Development tools

Release 2012.06.18

Patrick Vergain

June 22, 2012
## CONTENTS

1 Sphinx documentation ........................................ 3
   1.1 Documentation about sphinx ................................ 3

2 Development .................................................. 31
   2.1 Admin sys .................................................. 31
   2.2 Building software .......................................... 35
This is a documentation about some tools and environments used for some typical software engineering tasks:

hg clone ssh://hg@bitbucket.org/pvergain/devtools_doc
hg clone https://hg@bitbucket.org/pvergain/devtools_doc
Contents

• Introduction
• Sphinx documentation
• Development
  – Admin sys
  – Building software
  – Command Line interface
  – Communication
  – Configuration software
  – Desktop environment
  – Documentation
  – Data (SQL or NOSQL)
  – Drivers
  – Free Software
  – Firmwares
  – Glossary
  – GUI (Graphical User Interface)
  – IDEs
  – Installation
  – Internationalization (i18n)
  – Languages
  – Licences Libres
  – Localization (l10n)
  – Multimedia
  – Open data
  – Operating systems
  – Privacy
  – Programming
  – Security
  – Software engineering
  – Software frameworks
  – Test software
  – Version Control Systems
  – Web development

• Hardware
  – Hardware

• Languages
  – Languages
1.1 Documentation about sphinx

Date June 21, 2012

See Also:
- http://packages.python.org/an_example_pypi_project/sphinx.html

1.1.1 conf.py

See Also:
- http://sphinx.pocoo.org/latest/config.html?highlight=conf#conf

The configuration directory must contain a file named conf.py. This file (containing Python code) is called the “build configuration file” and contains all configuration needed to customize Sphinx input and output behavior. The configuration file is executed as Python code at build time (using execfile(), and with the current directory set to its containing directory), and therefore can execute arbitrarily complex code.

Sphinx then reads simple names from the file’s namespace as its configuration.

pypi Example

See Also:
- http://packages.python.org/an_example_pypi_project/sphinx.html#conf-py

In your doc/source directory is now a python file called conf.py. This is the file that controls the basics of how sphinx runs when you run a build. Here you can do this like:
- Change the version/release number by setting the version and release variables.
- Set the project name and author name.
- Setup a project logo.
- Set the default style to sphinx or default. Default is what the standard python docs use.

and much much more.

Browsing through this file will give you an understanding of the basics.
Exclude patterns

exclude_patterns

See Also:
http://sphinx.pocoo.org/latest/config.html?highlight=conf#confval-exclude_patterns

if conf_product==’mini’:
    exclude_patterns = [’interface/*.rst’,’dialogs/*.rst’]
elif conf_product==’main’:
    exclude_patterns = [’mini-indexes.rst’]

1.1.2 sphinx markup

See Also:
http://sphinx.pocoo.org/latest/markup/index.html

sphinx inline markup

See Also:
http://sphinx.pocoo.org/latest/markup/inline.html

Sphinx uses interpreted text roles to insert semantic markup into documents.

abbr

See Also:
http://sphinx.pocoo.org/latest/markup/inline.html?highlight=doc#role-doc

An abbreviation. If the role content contains a parenthesized explanation, it will be treated specially: it will be shown in a tool-tip in HTML, and output only once in LaTeX.
Example:
:abbr:’LIFO (last-in, first-out)’.
LIFO (last-in, first-out)

doc

See Also:
http://sphinx.pocoo.org/latest/markup/inline.html?highlight=doc#role-doc

Link to the specified document; the document name can be specified in absolute or relative fashion.
If no explicit link text is given the link caption will be the title of the given document.
Example 1: local link

For example, if the reference :doc:`command` occurs in the document inline/index, then the link refers to :file:`inline/command`.

For example, if the reference `command` occurs in the document inline/index, then the link refers to `inline/command`.

Example 2: with explicit link text

reference is :doc:`Index principal </index>` or :doc:`Index local <../index>`

reference is *Index principal* or *Index local*

Example 3 no explicit link text

reference is :doc:`/index` or :doc:`../index`

reference is *Introduction* or *sphinx markup*

command

See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=command#role-command

Example:

The name of an OS-level command, such as :command:`rm`.

The name of an OS-level command, such as `rm`.

download

See Also:

http://sphinx.pocoo.org/latest/markup/inline.html

This role lets you link to files within your source tree that are not ReST documents that can be viewed, but files that can be downloaded.

When you use this role, the referenced file is automatically marked for inclusion in the output when building (obviously, for HTML output only).

All downloadable files are put into the _downloads subdirectory of the output directory; duplicate filenames are handled.

An example:

See :download:`this example script <../example.py>`.

The given filename is usually relative to the directory the current source file is contained in, but if it is absolute (starting with `/`), it is taken as relative to the top source directory.

The example.py file will be copied to the output directory, and a suitable link generated to it.

1.1. Documentation about sphinx
glossary

This directive must contain a reST definition-list-like markup with terms and definitions. The definitions will then be referencable with the term role. Example:

```plaintext
.. glossary::

   environment
       A structure where information about all documents under the root is saved, and used for cross-referencing. The environment is pickled after the parsing stage, so that successive runs only need to read and parse new and changed documents.

   source directory
       The directory which, including its subdirectories, contains all source files for one Sphinx project.
```

In contrast to regular definition lists, multiple terms per entry are allowed, and inline markup is allowed in terms. You can link to all of the terms. For example:

```plaintext
.. glossary::

   term 1
   term 2
       Definition of both terms.
```

(When the glossary is sorted, the first term determines the sort order.) New in version 0.6: You can now give the glossary directive a :sorted: flag that will automatically sort the entries alphabetically. Changed in version 1.1: Now supports multiple terms and inline markup in terms.

guilabel

See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=guilabel#role-guilabel

Labels presented as part of an interactive user interface should be marked using :guilabel:. This includes labels from text-based interfaces such as those created using curses or other text-based libraries. Any label used in the interface should be marked with this role, including:

- button labels,
- window titles,
- field names,
- menu and
- menu selection names,
- and even values in selection lists.

Changed in version 1.0: An accelerator key for the GUI label can be included using an ampersand; this will be stripped and displayed underlined in the output (example: Cancel).

To include a literal ampersand, double it.
Example 1

button :guilabel:'Start'

button Start

Example 2 ampersand accelerators  guilabel also supports ampersand accelerators just like guilabel.

button :guilabel:'&Start'

button Start

menuselection

Menu selections should be marked using the menuselection role. This is used to mark a complete sequence of menu selections, including selecting submenus and choosing a specific operation, or any subsequence of such a sequence. The names of individual selections should be separated by --

Example 1  For example, to mark the selection Start -> Programs, use this markup:

:menuselection:'Start --> Programs'

Start → Programs

When including a selection that includes some trailing indicator, such as the ellipsis some operating systems use to indicate that the command opens a dialog, the indicator should be omitted from the selection name.

Example 2 ampersand accelerators  menuselection also supports ampersand accelerators just like guilabel.

:menuselection:'Start --> &Programs'

Start → Programs

program

See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=doc#role-doc

The name of an executable program.

This may differ from the file name for the executable for some platforms.

In particular, the .exe (or other) extension should be omitted for Windows programs.

Example

:program:'Geany.exe'

Geany.exe

1.1. Documentation about sphinx
term (very important)

See Also:
http://sphinx.pocoo.org/latest/markup/inline.html

Reference to a term in the glossary.
The glossary is created using the glossary directive containing a definition list with terms and definitions.
It does not have to be in the same file as the term markup, for example the Python docs have one global glossary in the glossary.rst file.
If you use a term that’s not explained in a glossary, you’ll get a warning during build.
Example:
See :term:`Sphinx`

See Sphinx

sphinx misc markup (very important)

index (very important)

Sphinx automatically creates index entries from all object descriptions (like functions, classes or attributes) like discussed in domains.
However, there is also explicit markup available, to make the index more comprehensive and enable index entries in documents where information is not mainly contained in information units, such as the language reference.

.. index:: <entries>
   This directive contains one or more index entries. Each entry consists of a type and a value, separated by a colon.

   For example:

   .. index::
      single: execution; context
      module: __main__
      module: sys
      triple: module; search; path

   The execution context
   ----------------------

   ...

   This directive contains five entries, which will be converted to entries in the generated index which link to the exact location of the index statement (or, in case of offline media, the corresponding page number).

   Since index directives generate cross-reference targets at their location in the source, it makes sense to put them before the thing they refer to – e.g. a heading, as in the example above.

! exclamation (important)

You can mark up “main” index entries by prefixing them with an exclamation mark. The references to “main” entries are emphasized in the generated index. For example, if two pages contain
and one page contains ::

.. index:: Pytho

then the backlink to the latter page is emphasized among the three backlinks.

For index directives containing only "single" entries, there is a shorthand notation: ::

.. index:: BNF, grammar, syntax, notation

This creates four index entries.

.. versionchanged:: 1.1
   Added "see" and "seealso" types, as well as marking main entries.

pair (very important)

pair: loop; statement is a shortcut that creates two index entries, namely loop; statement and statement; loop.

Example:

.. index::
   pair: sphinx; pair
   pair: sphinx important; contents

see

see: entry; other creates an index entry that refers from entry to other.

seealso

Like see, but inserts “see also” instead of “see”.

single

Creates a single index entry.

Can be made a subentry by separating the subentry text with a semicolon (this notation is also used below to describe what entries are created).

triple

Likewise, triple: module; search; path is a shortcut that creates three index entries, which are:

- module; search path
- search; path, module
- path; module search.

1.1. Documentation about sphinx
Deprecated: module, keyword, operator, object, exception, statement, builtin

module, keyword, operator, object, exception, statement, builtin These all create two index entries.

For example, module: hashlib creates the entries module; hashlib and hashlib; module.
(These are Python-specific and therefore deprecated.)

sphinx paragraph level markup

See Also:
http://sphinx.pocoo.org/latest/markup/para.html

These directives create short paragraphs and can be used inside information units as well as normal text:

contents (très important)

See Also:
• http://docutils.sourceforge.net/docs/ref/rst/directives.html#table-of-contents

Table-of-contents markup The toctree directive, which generates tables of contents of subdocuments, is described in The TOC tree.

For local tables of contents, use the standard reST contents directive.

Example 1

.. contents::
   :local:

Example 2

.. contents::
   :depth: 2

Contents

• contents (très important)
  – Table-of-contents markup
  – Example 1
  – Example 2

hlist

See Also:
http://sphinx.pocoo.org/latest/markup/para.html

These directives create short paragraphs and can be used inside information units as well as normal text:
This directive must contain a bullet list. It will transform it into a more compact list by either distributing more than one item horizontally, or reducing spacing between items, depending on the builder.

For builders that support the horizontal distribution, there is a columns option that specifies the number of columns; it defaults to 2. Example:

```plaintext
.. hlist::
   :columns: 3

   * A list of
   * short items
   * that should be
   * displayed
   * horizontally
```

New in version 0.6.

versionadded

See Also:


This directive documents the version of the project which added the described feature to the library or C API. When this applies to an entire module, it should be placed at the top of the module section before any prose.

The first argument must be given and is the version in question; you can add a second argument consisting of a brief explanation of the change.

Example:

```plaintext
.. versionadded:: 2.5
   The *spam* parameter.
```

Note that there must be no blank line between the directive head and the explanation; this is to make these blocks visually continuous in the markup.

1.1.3 sphinx domain

See Also:

- http://sphinx.pocoo.org/latest/domains.html

What is a Domain?

Originally, Sphinx was conceived for a single project, the documentation of the Python language. Shortly afterwards, it was made available for everyone as a documentation tool, but the documentation of Python modules remained deeply built in – the most fundamental directives, like function, were designed for Python objects. Since Sphinx has become somewhat popular, interest developed in using it for many different purposes: C/C++ projects, JavaScript, or even reStructuredText markup (like in this documentation).

While this was always possible, it is now much easier to easily support documentation of projects using different programming languages or even ones not supported by the main Sphinx distribution, by providing a domain for every such purpose.

A domain is a collection of markup (reStructuredText directives and roles) to describe and link to objects belonging together, e.g. elements of a programming language. Directive and role names in a domain have names like domain:name, e.g. py:function. Domains can also provide custom indices (like the Python Module Index).
Having domains means that there are no naming problems when one set of documentation wants to refer to e.g. C++ and Python classes. It also means that extensions that support the documentation of whole new languages are much easier to write.

**sphinx C domain**

See Also:

* [http://sphinx.pocoo.org/latest/domains.html](http://sphinx.pocoo.org/latest/domains.html)

The C domain (name `c`) is suited for documentation of C API.

```
.. c:function:: type name(signature)
   Describes a C function. The signature should be given as in C, e.g.:
      .. c:function:: PyObject* PyType_GenericAlloc(PyTypeObject *type, Py_ssize_t nitems)
   This is also used to describe function-like preprocessor macros. The names of the arguments should be given so they may be used in the description.
   Note that you don’t have to backslash-escape asterisks in the signature, as it is not parsed by the reST inliner.

.. c:member:: type name
   Describes a C struct member. Example signature:
      .. c:member:: PyObject* PyTypeObject.tp_bases
   The text of the description should include the range of values allowed, how the value should be interpreted, and whether the value can be changed. References to structure members in text should use the member role.

.. c:macro:: name
   Describes a “simple” C macro. Simple macros are macros which are used for code expansion, but which do not take arguments so cannot be described as functions. This is not to be used for simple constant definitions. Examples of its use in the Python documentation include `PyObject_HEAD` and `Py_BEGIN_ALLOW_THREADS`.

.. c:type:: name
   Describes a C type (whether defined by a typedef or struct). The signature should just be the type name.

.. c:var:: type name
   Describes a global C variable. The signature should include the type, such as:
      .. c:var:: PyObject* PyClass_Type
```

**Cross-referencing C constructs**

The following roles create cross-references to C-language constructs if they are defined in the documentation:

```
:c: data:
   Reference a C-language variable.

:c: func:
   Reference a C-language function. Should include trailing parentheses.

:c: macro:
   Reference a “simple” C macro, as defined above.

:c: type:
   Reference a C-language type.
```
sphinx C++ domain

See Also:

- http://sphinx.pocoo.org/latest/domains.html

The C++ Domain

The C++ domain (name `cpp`) supports documenting C++ projects.

The following directives are available:

```
.. cpp:class:: signatures
.. cpp:function:: signatures
.. cpp:member:: signatures
.. cpp:type:: signatures
```

Describe a C++ object. Full signature specification is supported – give the signature as you would in the declaration. Here some examples:

```
.. cpp:function:: bool namespaced::theclass::method(int arg1, std::string arg2)
   
   Describes a method with parameters and types.

.. cpp:function:: bool namespaced::theclass::method(arg1, arg2)
   
   Describes a method without types.

.. cpp:function:: const T &array<T>::operator[]() const
   
   Describes the constant indexing operator of a templated array.

.. cpp:function:: operator bool() const
   
   Describe a casting operator here.

.. cpp:function:: constexpr void foo(std::string &bar[2]) noexcept
   
   Describe a constexpr function here.

.. cpp:member:: std::string theclass::name

.. cpp:member:: std::string theclass::name[N][M]

.. cpp:type:: theclass::const_iterator
```

Will be rendered like this:

```
bool namespaced::theclass::method(int arg1, std::string arg2)
   
   Describes a method with parameters and types.

bool namespaced::theclass::method(arg1, arg2)
   
   Describes a method without types.

const T &array<T>::operator[]() const
   
   Describes the constant indexing operator of a templated array.

operator bool() const
   
   Describe a casting operator here.
```
constexpr void foo (std::string &bar[2]) noexcept

Describe a constexpr function here.

std::string theclass::name
std::string theclass::name[N][M]
type theclass::const_iterator

.. _cpp:namespace:: namespace

Select the current C++ namespace for the following objects.

These roles link to the given object types:

.. _cpp:class:
.. _cpp:func:
.. _cpp:member:
.. _cpp:type:

Reference a C++ object. You can give the full signature (and need to, for overloaded functions.)

Note: Sphinx’ syntax to give references a custom title can interfere with linking to template classes, if nothing follows the closing angle bracket, i.e. if the link looks like this: _cpp:class:'MyClass<T>'_. This is interpreted as a link to T with a title of MyClass. In this case, please escape the opening angle bracket with a backlash, like this: _cpp:class:`MyClass<T>`_.

Note on References

It is currently impossible to link to a specific version of an overloaded method. Currently the C++ domain is the first domain that has basic support for overloaded methods and until there is more data for comparison we don’t want to select a bad syntax to reference a specific overload. Currently Sphinx will link to the first overloaded version of the method / function.

1.1.4 reStructuredText Primer

This section is a brief introduction to reStructuredText (reST) concepts and syntax, intended to provide authors with enough information to author documents productively. Since reST was designed to be a simple, unobtrusive markup language, this will not take too long.

See Also:

* http://docutils.sourceforge.net/rst.html

**toctree**

See Also:

http://sphinx.pocoo.org/latest/markup/toctree.html

de  Luc Saffre <luc.saffre@gmail.com>
heure de l’expéditeur   Envoyé à 14:26 (GMT+02:00). Heure locale : 17:00.
répondre à  sphinx-dev@googlegroups.com
à   sphinx-dev <sphinx-dev@googlegroups.com>
date   18 février 2011 14:26
Hi,

I just discovered a useful thing that is not documented (at least not where I would expect it to be at http://sphinx.pocoo.org/markup/toctree.html)

A `toctree` with `:glob:` flag option not only supports `*` but also the `?` wildcard character.

I use this to have the correct sort order in an automatic index for pages are named using simple numbers, starting from `1.rst`. When I have more than 9 files (12 for example), then a simple `*` would yield a toctree sorted 1, 11, 12, 2, 3, 4. But if I use the following construct:

```plaintext
.. toctree::
   :maxdepth: 1
   :glob:

   ?
   ??
```

then I get the expected order 1, 2, 3, 4, 11, 12.

In short: Sphinx is great :-)

**Paragraphs**

The paragraph is the most basic block in a reST document. Paragraphs are simply chunks of text separated by one or more blank lines. As in Python, indentation is significant in reST, so all lines of the same paragraph must be left-aligned to the same level of indentation.

**Image**

See Also:

- http://sphinx.pocoo.org/latest/domains.html#directive-rst:role
- http://docutils.sourceforge.net/docs/ref/rst/directives.html#images

**image**

An “image” is a simple picture:

```plaintext
.. image:: picture.png
```

The URI for the image source file is specified in the directive argument. As with hyperlink targets, the image URI may begin on the same line as the explicit markup start and target name, or it may begin in an indented text block immediately following, with no intervening blank lines. If there are multiple lines in the link block, they are stripped of leading and trailing whitespace and joined together.

Optionally, the image link block may contain a flat field list, the image options.

For example:

```plaintext
.. image:: picture.jpeg
   :height: 100px
   :width: 200 px
   :scale: 50 %
```
Use:

.. image:: ../images/wiki_logo_openalea.png
to put an image

![OpenAlea](../images/wiki_logo_openalea.png)

**Note:** As mentionned earlier, a directive may have options put between two columns:

```
.. image:: ../images/wiki_logo_openalea.png
   :width: 200px
   :align: center
   :height: 100px
   :alt: alternate text
```

```
.. figure:: ../images/wiki_logo_openalea.png
   :width: 200px
   :align: center
   :height: 100px
   :alt: alternate text
   :figclass: align-center

   figure are like images but with a caption
   and whatever else you wish to add

   .. code-block:: python

       import image
```
gives

**Geoserver example**

**See Also:**

- [http://docs.geoserver.org/trunk/en/docguide/sphinx.html#images](http://docs.geoserver.org/trunk/en/docguide/sphinx.html#images)

Add images to your documentation when possible. Images, such as screenshots, are a very helpful way of making documentation understandable. When making screenshots, try to crop out unnecessary content (browser window, desktop, etc).
Avoid scaling the images, as the Sphinx theme automatically resizes large images.
It is also helpful to include a caption underneath the image.

Figure 1.2: *The GeoServer logo as shown on the homepage.*

This image is generated by the following code:

```
.. figure:: pagelogo_geoserver.png
   :align: center

   *The GeoServer logo as shown on the homepage.*
```

In this example, the image file exists in the same directory as the source page. If this is not the case, you can insert path information in the above command.

Substitutions

See Also:

Substitutions

Substitutions syntax is
The `biohazard` symbol must be used on containers used to dispose of medical waste.

Or if you want to do a literal text replacement use:

```
.. |doctest| replace:: :mod:`doctest`
```

I really like `doctest`.

Which renders like this:

The `biohazard` symbol must be used on containers used to dispose of medical waste.

I really like `doctest`.

**Note:** Substitutions are really useful, especially when put into a `global.rst` and included at the top of every file. See *Includes* for more.

### Lists and Quotes

List markup is natural: just place an asterisk at the start of a paragraph and indent properly. The same goes for numbered lists; they can also be autonumbered using a `#` sign:

* This is a bulleted list.
  * It has two items, the second item uses two lines.

1. This is a numbered list.
2. It has two items too.

#. This is a numbered list.
#. It has two items too.

Note that Sphinx disables the use of enumerated lists introduced by alphabetic or roman numerals, such as

A. First item
B. Second item

Nested lists are possible, but be aware that they must be separated from the parent list items by blank lines:

* this is
  * a list
    * with a nested list
      * and some subitems
    * and here the parent list continues

**Definition lists are created as follows:**

```
term (up to a line of text)
  Definition of the term, which must be indented

  and can even consist of multiple paragraphs
```

next term
  Description.
Paragraphs are quoted by just indenting them more than the surrounding paragraphs.

**Source Code**

Literal code blocks are introduced by ending a paragraph with the special marker ::. The literal block must be indented, to be able to include blank lines:

```
This is a normal text paragraph. The next paragraph is a code sample::

   It is not processed in any way, except
to make sure the indentation is removed.

   It can span multiple lines.
```

This is a normal text paragraph again.

The handling of the :: marker is smart:

- If it occurs as a paragraph of its own, that paragraph is completely left out of the document.
- If it is preceded by whitespace, the marker is removed.
- If it is preceded by non-whitespace, the marker is replaced by a single colon.

That way, the second sentence in the above example’s first paragraph would be rendered as “The next paragraph is a code sample:”.

**include**

See Also:

- [http://packages.python.org/an_example_pypi_project/sphinx.html#includes](http://packages.python.org/an_example_pypi_project/sphinx.html#includes)

The syntax:
```
.. include :: myfile.rst
```

Will ‘inline’ the given file. A common convention I use is create a global .rst file called `global.rst` and include that at the top of every page. Very useful for links to common images or common files links, etc.

**code-block (pygments, highlight)**

See Also:

- [http://sphinx.pocoo.org/latest/markup/code.html#index-0](http://sphinx.pocoo.org/latest/markup/code.html#index-0)
- [http://pygments.org/docs/lexers/](http://pygments.org/docs/lexers/)

Sphinx does syntax highlighting using the Pygments library.

For documents that have to show snippets in different languages, there’s also a `code-block` directive that is given the highlighting language directly:
```
.. code-block:: python

   Some python code.
```

You can specify different highlighting for a code block using the following syntax:
Default highlighter

With two colons you start a code block using the default highlighter:

```python
# Some Python code here
# The language defaults to Python, we don't need to set it
if 1 == 2:
    pass
```

With two colons you start a code block using the default highlighter:

```python
# Some Python code here
# The language defaults to Python, we don't need to set it
if 1 == 2:
    pass
```

**python highlighter**

You can specify the language used for syntax highlighting by using code-block:

```python
.. code-block:: python

    if "foo" == "bar":
        # This is Python code
        pass

if "foo" == "bar":
    # This is Python code
    pass
```

**xml highlighter**

For example, to specify XML:

```xml
.. code-block:: xml

    <somesnippet>Some XML</somesnippet>

<somesnippet>Some XML</somesnippet>
```

**console highlighter**

... or UNIX shell:

```console
.. code-block:: console

    # A comment
    sh myscript.sh

    # A comment
    sh myscript.sh
```
ini highlighter

... or a buildout.cfg:

```ini
[some-part]
# A random part in the buildout
recipe = collective.recipe.foo
option = value
```

```ini
[some-part]
# A random part in the buildout
recipe = collective.recipe.foo
option = value
```

pycon python console highlighter

... or interactive Python:

```pycon
>>> class Foo:
...
...    bar = 100
...
>>> f = Foo()
>>> f.bar
100
>>> f.bar / 0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero
```

```pycon
>>> class Foo:
...
...    bar = 100
...
>>> f = Foo()
>>> f.bar
100
>>> f.bar / 0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero
```

highlighting mode for the whole document

Setting the highlighting mode for the whole document:

```highlight:: console
All code blocks in this doc use console highlighting by default:
some shell commands
```

If syntax highlighting is not enabled for your code block, you probably have a syntax error and Pygments will fail silently.

1.1. Documentation about sphinx
rest Tables

Grid tables

Two forms of tables are supported. For *grid tables* you have to “paint” the cell grid yourself. They look like this:

<table>
<thead>
<tr>
<th>Header row, column 1</th>
<th>Header 2</th>
<th>Header 3</th>
<th>Header 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(header rows optional)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>body row 1, column 1</td>
<td>column 2</td>
<td>column 3</td>
<td>column 4</td>
</tr>
<tr>
<td>body row 2</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Simple tables

*Simple tables* are easier to write, but limited: they must contain more than one row, and the first column cannot contain multiple lines. They look like this:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A and B</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>

list-table

See Also:

- [http://docs.geoserver.org/trunk/en/docguide/sphinx.html#list-tables](http://docs.geoserver.org/trunk/en/docguide/sphinx.html#list-tables)

Bulleted lists can sometimes be cumbersome and hard to follow.

When dealing with a long list of items, use list-tables.

For example, to talk about a list of options, create a table that looks like this:

<table>
<thead>
<tr>
<th>Shapes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Square</td>
<td>Four sides of equal length, 90 degree angles</td>
</tr>
<tr>
<td>Rectangle</td>
<td>Four sides, 90 degree angles</td>
</tr>
</tbody>
</table>

This is done with the following code:

```markdown
.. list-table::
   :widths: 20 80
   :header-rows: 1

   * - Shapes
     - Description
   * - Square
     - Four sides of equal length, 90 degree angles
```
* - Rectangle
  - Four sides, 90 degree angles

**sphinx Hyperlinks**

**External links**

Use ‘Link text <http://target>’ for inline web links. If the link text should be the web address, you don’t need special markup at all, the parser finds links and mail addresses in ordinary text.

Example of external link: ‘reST role <http://sphinx.pocoo.org/latest/markup/inline.html#role-ref>’

Example of external link: reST role .

**Internal links**

See Also:
http://sphinx.pocoo.org/latest/markup/inline.html#role-ref

Internal linking is done via a special reST role .

:ref:`link to internal links <internal_links>`

**Sphinx Sections**

Section headers are created by underlining (and optionally overlining) the section title with a punctuation character, at least as long as the text:

```
=================
This is a heading
=================
```

Normally, there are no heading levels assigned to certain characters as the structure is determined from the succession of headings. However, for the Python documentation, we use this convention:

- # with overline, for parts
- * with overline, for chapters
- =, for sections
- –, for subsections
- ^, for subsubsections
- `, for paragraphs

**Explicit Markup**

“Explicit markup” is used in reST for most constructs that need special handling, such as footnotes, specially-highlighted paragraphs, comments, and generic directives.
An explicit markup block begins with a line starting with .. followed by whitespace and is terminated by the next paragraph at the same level of indentation. (There needs to be a blank line between explicit markup and normal paragraphs. This may all sound a bit complicated, but it is intuitive enough when you write it.)

**Directives**

*See Also:*

http://sphinx.readthedocs.org/en/1.1.0/modules.html#directive-rst:role

A directive is a generic block of explicit markup. Besides roles, it is one of the extension mechanisms of reST, and Sphinx makes heavy use of it.

Basically, a directive consists of a name, arguments, options and content. (Keep this terminology in mind, it is used in the next chapter describing custom directives.) Looking at this example,

```
.. function:: foo(x)
   foo(y, z)
   :bar: no

   Return a line of text input from the user.
```

`function` is the directive name. It is given two arguments here, the remainder of the first line and the second line, as well as one option `bar` (as you can see, options are given in the lines immediately following the arguments and indicated by the colons).

The directive content follows after a blank line and is indented relative to the directive start.

**Footnotes**

For footnotes, use [#]_ to mark the footnote location, and add the footnote body at the bottom of the document after a “Footnotes” rubric heading, like so:

```
Lorem ipsum [#]_ dolor sit amet ... [#]_
```

```
.. rubric:: Footnotes

.. [#] Text of the first footnote.
.. [#] Text of the second footnote.
```

You can also explicitly number the footnotes for better context.

**Comments**

Every explicit markup block which isn’t a valid markup construct (like the footnotes above) is regarded as a comment.

**Source encoding**

Since the easiest way to include special characters like em dashes or copyright signs in reST is to directly write them as Unicode characters, one has to specify an encoding:

All Python documentation source files must be in UTF-8 encoding, and the HTML documents written from them will be in that encoding as well.
Gotchas

There are some problems one commonly runs into while authoring reST documents:

- **Separation of inline markup:** As said above, inline markup spans must be separated from the surrounding text by non-word characters, you have to use an escaped space to get around that.

Explicit Markup

“Explicit markup” is used in reST for most constructs that need special handling, such as footnotes, specially-highlighted paragraphs, comments, and generic directives.

An explicit markup block begins with a line starting with .. followed by whitespace and is terminated by the next paragraph at the same level of indentation. (There needs to be a blank line between explicit markup and normal paragraphs. This may all sound a bit complicated, but it is intuitive enough when you write it.)

Sphinx Directives

See Also:

- [http://docutils.sourceforge.net/docs/ref/rst/directives.html#images](http://docutils.sourceforge.net/docs/ref/rst/directives.html#images)

A directive is a generic block of explicit markup. Besides roles, it is one of the extension mechanisms of reST, and Sphinx makes heavy use of it.

Basically, a directive consists of a name, arguments, options and content. (Keep this terminology in mind, it is used in the next chapter describing custom directives.) Looking at this example,

```plaintext
.. function:: foo(x)
    :bar: no

    foo(y, z)
```

function is the directive name. It is given two arguments here, the remainder of the first line and the second line, as well as one option bar (as you can see, options are given in the lines immediately following the arguments and indicated by the colons).

The directive content follows after a blank line and is indented relative to the directive start.

Sphinx Footnotes

For footnotes, use [#]_ to mark the footnote location, and add the footnote body at the bottom of the document after a “Footnotes” rubric heading, like so:

```plaintext
Lorem ipsum [#]_ dolor sit amet ...
```

.. rubric:: Footnotes

.. [#] Text of the first footnote.
.. [#] Text of the second footnote.

You can also explicitly number the footnotes for better context.
Comments

Every explicit markup block which isn’t a valid markup construct (like the footnotes above) is regarded as a comment.

Source encoding

Since the easiest way to include special characters like em dashes or copyright signs in reST is to directly write them as Unicode characters, one has to specify an encoding:

All Python documentation source files must be in UTF-8 encoding, and the HTML documents written from them will be in that encoding as well.

sidebar

See Also:


t is possible to create sibar

Sidebar Title

Optional Sidebar Subtitle

Subsequent indented lines comprise the body of the sidebar, and are interpreted as body elements.

using the following code:

```rest
.. sidebar:: Sidebar Title
   :subtitle: Optional Sidebar Subtitle

   Subsequent indented lines comprise
   the body of the sidebar, and are
   interpreted as body elements.
```

Note: sidebar appears as floating box and may not appear nicely.

Gotchas

There are some problems one commonly runs into while authoring reST documents:

- **Separation of inline markup**: As said above, inline markup spans must be separated from the surrounding text by non-word characters, you have to use an escaped space to get around that.

1.1.5 sphinx code

autodoc

See Also:

Development tools, Release 2012.06.18

- http://packages.python.org/an_example_pypi_project/sphinx.html#auto-directives

**literalinclude**

See Also:
http://sphinx.pocoo.org/markup/code.html

```plaintext
.. literalinclude:: example.py
   :language: python
   :linenos:
```

The file name is usually relative to the current file’s path. However, if it is absolute (starting with `/`), it is relative to the top source directory.

The directive also supports the `linenos` flag option to switch on line numbers, and a `language` option to select a language different from the current file’s standard language. Example with options:

```plaintext
.. literalinclude:: example.rb
   :language: ruby
   :linenos:
```

Include files are assumed to be encoded in the `source_encoding`. If the file has a different encoding, you can specify it with the `encoding` option:

```plaintext
.. literalinclude:: example.py
   :encoding: latin-1
```

The directive also supports including only parts of the file. If it is a Python module, you can select a class, function or method to include using the `pyobject` option:

```plaintext
.. literalinclude:: example.py
   :pyobject: Timer.start
```

This would only include the code lines belonging to the `start()` method in the `Timer` class within the file. Alternately, you can specify exactly which lines to include by giving a `lines` option:

```plaintext
.. literalinclude:: example.py
   :lines: 1,3,5-10,20-
```

This includes the lines 1, 3, 5 to 10 and lines 20 to the last line.

**Documenting parameters**

See Also:

- http://packages.python.org/an_example_pypi_project/sphinx.html

I’m wondering if people have suggestions on the best way to format function/method docstrings so it’s possible to get Sphinx’s fancy formatting and yet retain nice and readable docstrings from the Python prompt. I’m especially interested in how to document the function/ method input “parameters”.

I know about the :param name: stanza but when there’s a relatively long list of parameters, I don’t find it very readable from the Python prompt. Of course, once processed by Sphinx, it yields great looking documentation.

1.1. Documentation about sphinx
I also know about http://packages.python.org/an_example_pypi_project/sphinx.html and the googley and sphinxey variants. I’m thinking there must be a good compromise here.

For return values, I often use the following in my docstrings:

```markdown
returns

out1  something

out2  something else
```

Indentation and proper alignment make this easy to read and it looks great once processed by Sphinx. But I can’t find a similar syntax for parameters and keywords. I thought (and obviously I’m wrong) that Sphinx accepted :parameters: and :keywords: but those don’t look nice once processed, e.g., to html. In particular, :parameters: is converted to “Parameters :” (with a space), which often causes the colon to end on a new line below the word “Parameters”.

It would be great to be able to write:

```markdown
parameters

in1  description of in1

in2  description of in2

keywords

kw1  description of kw1
```

just the same way we can use :returns:. I’m happy to try and add this if it sounds like a good idea. I’d like to hear what people think anyways.

Thanks.

Example documenting parameters

Constants

**COMMAND_SUCCESS** The CR_S_SUCCESS value is returned when a command is successfull.

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>French Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000000L</td>
<td>COMMAND_SUCCESS</td>
<td>Opération réussie.</td>
</tr>
</tbody>
</table>

List of error codes

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>French Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x8130F001L</td>
<td>CR_E_FAILED</td>
<td>Une vérification de cohérence interne a échoué.</td>
</tr>
<tr>
<td>0x8130F002L</td>
<td>CR_E_TIMEOUT</td>
<td>Le délai a expiré.</td>
</tr>
<tr>
<td>0x8130F003L</td>
<td>CR_E_SEND_DATA_FAILED</td>
<td>L’envoi des données a échoué.</td>
</tr>
<tr>
<td>0x8130F004L</td>
<td>CR_E_OPEN_PORT_FAILED</td>
<td>L’ouverture du port COM a échoué.</td>
</tr>
<tr>
<td>0x8130F005L</td>
<td>CR_E_CLOSE_PORT_FAILED</td>
<td>La fermeture du port COM a échoué.</td>
</tr>
<tr>
<td>0x8130F006L</td>
<td>CR_E_BAD_SYNCHRO</td>
<td>Erreur de synchronisation avec la base.</td>
</tr>
<tr>
<td>0x8130F007L</td>
<td>CR_E_BAD_ADDRESS</td>
<td>Mauvaise adresse.</td>
</tr>
<tr>
<td>0x8130F008L</td>
<td>CR_E_BAD_SIZE</td>
<td>Taille incorrecte.</td>
</tr>
<tr>
<td>0x8130F009L</td>
<td>CR_E_BAD_CHANNEL</td>
<td>Mauvais canal.</td>
</tr>
<tr>
<td>0x8130F00AL</td>
<td>CR_E_BAD_STATUS</td>
<td>Mauvais statut retourné par la base.</td>
</tr>
<tr>
<td>0x8130F00BL</td>
<td>CR_E_OPEN_FILE_FAILED</td>
<td>Ouverture du fichier a échoué.</td>
</tr>
</tbody>
</table>

.. c:function:: DWORD CR_RFID_VerifierPIN(ST_CODE_PIN *pCodePIN)

Le code PIN permet d’autoriser l’utilisation des clés de chiffrement et de signature dans le lecteur RFID.
:Paramètres:
  :entree:
    :pCodePIN: le code PIN à présenter au lecteur RFID.

:Returns:
  :error: :ref:`see the error codes <list_error_codes>`
  :success: :ref:`COMMAND_SUCCESS <cr_success>`

**CR_RFID_VerifierPIN**

DWORD CR_RFID_VerifierPIN (ST_CODE_PIN *pCodePIN)

Le code PIN permet d’autoriser l’utilisation des clés de chiffrement et de signature dans le lecteur RFID.

**Paramètres**

  :entree
    :pCodePIN: le code PIN à présenter au lecteur RFID.

**Returns**

  :error: see the error codes
  :success: COMMAND_SUCCESS
2.1 Admin sys

2.1.1 Admin sys tool

Shinken

See Also:

- http://shinken.ideascale.com/

Shinken is a new monitoring tool in AGPLv3 written in Python and compatible with Nagios. The main goal of Shinken is to allow users to have a fully flexible architecture for their monitoring system that can easily scale to large environments. It’s as simple as in all marketing “cloud computing” slides, but here it’s real!

Shinken is compatible with the Nagios configuration standard and plugins. It works on any operating system and architecture that supports Python, which includes Windows and GNU/Linux.

Source

- git clone git://shinken.git.sourceforge.net/gitroot/shinken/shinken

Fabric

See Also:

- http://bitprophet.org
- http://docs.fabfile.org/1.0a/
Fabric is a Python library and command-line tool for streamlining the use of SSH for application deployment or systems administration tasks.

It provides a basic suite of operations for executing local or remote shell commands (normally or via `sudo`) and uploading/downloading files, as well as auxiliary functionality such as prompting the running user for input, or aborting execution.

Typical use involves creating a Python module containing one or more functions, then executing them via the `fab` command-line tool. Below is a small but complete “fabfile” containing a single task:

```
from fabric.api import run
def host_type():
    run('uname -s')
```

Once a task is defined, it may be run on one or more servers, like so:

```
$ fab -H localhost,linuxbox host_type
[localhost] run: uname -s
[localhost] out: Darwin
[linuxbox] run: uname -s
[linuxbox] out: Linux
```

Done.
Disconnecting from localhost... done.
Disconnecting from linuxbox... done.

In addition to use via the `fab` tool, Fabric’s components may be imported into other Python code, providing a Pythonic interface to the SSH protocol suite at a higher level than that provided by e.g. Paramiko (which Fabric itself leverages.)

### Development

The Fabric development team consists of two programmers, Jeff Forcier and Christian Vest Hansen, with Jeff taking the lead role.

However, dozens of other developers pitch in by submitting patches and ideas, via individual emails, Redmine, the mailing list and GitHub.

http://tav.espians.com/ See Also:

Fabric is an awesome tool. Like Capistrano and Vlad, it makes deployments a lot simpler than with shell scripts on their own.

However, once the complexity of your setup starts to grow, you very quickly start wishing for a cleaner and more powerful API.

And once you are deploying to more than 30 servers, you really wish that Fabric would run commands in parallel instead of doing them sequentially, one after another.

Having recently experienced this pain, I decided to rework the core of Fabric. I’ve documented it below — describing all the changes to the current Fabric 1.0a including funky new features like fab shell and staged deployments!
Fabtest

Test Fabric scripts on VirtualBox VMs

See Also:
- https://bitbucket.org/kmike/fabtest/src

Django fab deploy

django-fab-deploy is a collection of fabric scripts for deploying and managing django projects on Debian servers.

See Also:
http://packages.python.org/django-fab-deploy/

Other links

See Also:
http://imil.net/wp/tag/fabric/

LDAP

See Also:
https://secure.wikimedia.org/wikipedia/fr/wiki/LDAP

Lightweight Directory Access Protocol (LDAP) is à l’origine un protocole permettant l’interrogation et la modification des services d’annuaire.

Ce protocole repose sur TCP/IP.

Il a cependant évolué pour représenter une norme pour les systèmes d’annuaires, incluant un modèle de données, un modèle de nommage, un modèle fonctionnel basé sur le protocole LDAP, un modèle de sécurité et un modèle de réplication.

Un annuaire LDAP respecte généralement le modèle X.500 édicté par l’UIT-T : c’est une structure arborescente dont chacun des nœuds est constitué d’attributs associés à leurs valeurs.

Le nommage des éléments constituant l’arbre (racine, branches, feuilles) reflète souvent le modèle politique, géographique ou organisationnel de la structure représentée.

La tendance actuelle est d’utiliser le nommage DNS pour les éléments de base de l’annuaire (racine et premières branches, domain components ou dc=...).

Les branches plus profondes de l’annuaire peuvent représenter des unités organisationnelles ou des groupes (organizational units ou ou=...), des personnes (common name ou cn=... voire user identifier uid=...), ...

L’assemblage de tous les composants (du plus précis au plus général) d’un nom forme son distinguished name, l’exemple suivant en présente deux :

- cn=ordinateur,ou=machines,dc=EXEMPLE,dc=FR
La dernière version en date du protocole est LDAPv3. Cette version est définie par l’IETF dans plusieurs RFC en commençant par la RFC 4510.

**LDAP python tools**

- [http://pumpkin.prymitive.com/0.1/index.html](http://pumpkin.prymitive.com/0.1/index.html)

**Admin tools**

**pbs**

See Also:

- [https://github.com/amoffat/pbs](https://github.com/amoffat/pbs)

PBS is a unique subprocess wrapper that maps your system programs to Python functions dynamically.

PBS helps you write shell scripts in Python by giving you the good features of Bash (easy command calling, easy piping) with all the power and flexibility of Python.

```python
from pbs import ifconfig
print ifconfig("eth0")
```

PBS is not a collection of system commands implemented in Python.

**Remote Admin**

**x2go**

---

### Contents

- **x2go**
  - About X2Go
  - Installation

---

**About X2Go**  
x2go is an open (GPL/AGPL) source “server based computing” project. Combining the advantages of existing systems it features ease of use, performance and scalability.

x2go provides you with access to your desktop as an individual as well as a corporate user - from within your own network and via the internet.

x2go is not limited to particular hardware, it supports a variety of devices and architectures.

x2go is open source and open minded.

Like any open source project we welcome your support.

**Installation**
x2go install

See Also:


I have always used VNC to remotely connect to a Linux desktop.

On Windows it’s been RDP.

I came across a much better way than VNC to connect to my Kubuntu desktop: x2go.

The application is absolutely wonderful and dead easy to install on Kubuntu. But its documentation is a confusing mess. Fortunately, we don’t really need to use the documentation that much.

Server

sudo add-apt-repository ppa:x2go/stable
You are about to add the following PPA to your system:
x2go stable ppa
Quick howto to turn your machine into an x2go server:
sudo apt-add-repository ppa:x2go/stable
sudo apt-get update
sudo apt-get install x2goserver
sudo apt-get install x2gognomebindings # if you use GNOME
sudo apt-get install x2golxdebindings # if you use LXDE/lubuntu

x2goclient

sudo apt-add-repository ppa:x2go/stable
sudo apt-get update
sudo apt-get install x2goclient

2.2 Building software

2.2.1 Build automation

See Also:

- http://en.wikipedia.org/wiki/Make_%28software%29

autotools

See Also:


Autotools (ou GNU build system) est un terme général utilisé par la communauté anglosaxonne pour désigner l’ensemble des outils de build du projet GNU.

Autotools n’est pas un projet GNU, mais l’expression est souvent utilisée à l’extérieur comme à l’intérieur du projet pour désigner un ensemble de projets GNU, parmi lesquels:

- GNU AutoGen
- GNU Autoconf
• GNU Automake
• GNU Libtool
• GNU Make

Qt creator and autotools

See Also:
  • http://qt.gitorious.org/qt-creator/qt-creator/commit/6445c99ad2020182b5d105b29e373de6273cb5f5/diffs
  • http://qt.gitorious.org/~patriciasc

After all the work, I can already announce and celebrate, that the plugin for giving Qt Creator support for autotools projects, has been merged today into Qt Creator! I could not have finished my time at Openismus GmbH and series of posts related to the plugin in a better way. I am very happy of the result and hope it will be helpful for those who love autotools.

Bakefile

  • http://www.bakefile.org/index.html

Bakefile is cross-platform, cross-compiler native makefiles generator. It takes compiler-independent description of build tasks as input and generates native makefile (autoconf’s Makefile.in, Visual C++ project, bcc makefile etc.).

Bakefile’s task is to generate native makefiles, so that people can keep using their favorite tools. There are other cross-platform make solutions, but they either aren’t native and require the user to use unfamiliar tools (Boost.Build) or they are too limited (qmake).

Apache buildr

See Also:
http://buildr.apache.org/

Apache Buildr is a build system for Java-based applications, including support for Scala, Groovy and a growing number of JVM languages and tools. We wanted something that’s simple and intuitive to use, so we only need to tell it what to do, and it takes care of the rest. But also something we can easily extend for those one-off tasks, with a language that’s a joy to use. And of course, we wanted it to be fast, reliable and have outstanding dependency management.

buildroot (linux kernel tool)

See Also:
  • buildroot linux tool (Making Embedded Linux easy)
CMake est un « moteur de production » multiplate-forme. Il est comparable au programme Make dans le sens où le processus de construction logicielle est entièrement contrôlé par des fichiers de configuration, appelés CMakeLists.txt dans le cas de CMake.

Mais CMake ne produit pas directement le logiciel final, il s’occupe de la génération de fichiers de construction standards : makefiles sous Unix, et fichiers de projet Visual Studio sous Windows. Cela permet aux développeurs d’utiliser leur environnement de développement préféré comme à leur habitude.

C’est cette utilisation des outils habituels de développement qui distingue CMake des autres systèmes de production comme SCons ou les Autotools.

Le nom “CMake” est l’abréviation de “cross platform make”. Malgré l’utilisation de “make” dans son nom, CMake est une application séparée et de plus haut niveau que l’outil make.
Suite à un billet datant de 2008, je reviens vers vous pour ajouter quelques précisions sur la compilation de programme Qt avec CMake. En effet, pour mon programme XINX, j’ai modifié la chaîne de compilation actuelle utilisant QMake par une chaîne de compilation CMake.

CMake est un puissant générateur de Makefile, il permet de remplacer les anciens (mais pas complètement révolus) autotools.

**CMake ne remplace donc pas le programme make mais vient se placer en amont.**

CMake permet de compiler un programme à différents endroits du dossier des sources, ce qui permet de garder le répertoire des sources propre.

**Projects using Cmake**

http://brlcad.org/ and cmake  
See Also:

- http://freshmeat.net/projects/brl-cad

BRL-CAD 7.20.2 is now available! The 7.20 release marks the introduction of a new CMake-based build system for BRL-CAD.
libnfc and cmake  See Also:

http://code.google.com/p/libnfc/source/browse/trunk/CMakeLists.txt

Opencv and cmake  See Also:

- opencv image library

Do the cmake step from inside the folder where you want to build OpenCV. It is a good practice to keep your source and build files (the files that are going to be generated after the cmake step) separately.

Remember to specify the source folder while invoking cmake as the last argument. Otherwise cmake will scan for CMakeLists.txt file in the usual standard location /usr/local/src and will throw an error.

Unzipping all OpenCV directories from the source files to the src folder is not an elegant way of doing things in linux.

If you have a gui version of cmake installed, just do this from inside the build folder, $ ccmake <OpenCV source folder>

Another example for Windows users (assuming the .exe extracted files to C:OpenCV2.3)
Project automation is essential to the success of software projects. It should be straight-forward, easy and fun to implement. There is no one-size-fits-all process for builds. Therefore Gradle does not impose a rigid process over people. Yet we think finding and describing YOUR process is very important. And so, Gradle has the very best support for describing it.

We don’t believe in tools that save people from themselves. Gradle gives you all the freedom you need. Using Gradle you can create declarative, maintainable, concise and high-performance builds.

Declarative builds and build-by-convention

At the heart of Gradle lies a rich extensible Domain Specific Language (DSL) based on Groovy. Gradle pushes declarative builds to the next level by providing declarative language elements that you can assemble as you like. Those elements also provide build-by-convention support for Java, Groovy, OSGi, Web and Scala projects. Even more, this declarative language is extensible. Add your own new language elements or enhance the existing ones. Thus providing concise, maintainable and comprehensible builds.
Why Groovy? We think the advantages of an internal DSL (based on a dynamic language) over XML are tremendous in case of build scripts. There are a couple of dynamic languages out there. Why Groovy? The answer lies in the context Gradle is operating in. Although Gradle is a general purpose build tool at its core, its main focus are Java projects. In such projects obviously the team members know Java. We think a build should be as transparent as possible to all team members.

You might argue why not using Java then as the language for build scripts. We think this is a valid question. It would have the highest transparency for your team and the lowest learning curve. But due to limitations of Java such a build language would not be as nice, expressive and powerful as it could be.

Languages like Python, Groovy or Ruby do a much better job here. We have chosen Groovy as it offers by far the greatest transparency for Java people. Its base syntax is the same as Java’s as well as its type system, its package structure and other things. Groovy builds a lot on top of that. But on a common ground with Java.

For Java teams which share also Python or Ruby knowledge or are happy to learn it, the above arguments don’t apply. The Gradle design is well-suited for creating another build script engine in JRuby or Jython. It just doesn’t have the highest priority for us at the moment. We happily support any community effort to create additional build script engines.

gyp

GYP can Generate Your Projects.
- http://code.google.com/p/gyp/

Gnome jhbuild

See Also:
https://live.gnome.org/Jhbuild

JHBuild allows you to automatically download and compile “modules” (i.e. source code packages). Modules are listed in “module set” files, which also include dependency information so that JHBuild can discover what modules need to be built and in what order.

JHBuild was originally written for building GNOME, but has since been extended to be usable with other projects.

Kook

See Also:
- http://www.kuwata-lab.com/kook/

Kook is software build tool similar to Rake, Ant, SCons or Cook.

Kook liken build process to cooking:
- Recipe – Task or rule to generate output from input.
- Product – Output from recipe.
- Ingredient – Input for recipe.
- Method – Steps (= a set of commands) to generate product from ingredients.
- Spice – Additional argument (= command-line options) for recipe.
- Cookbook – A file containing recipe definitions.
The most interesting feature Kook has (and others doesn’t) is that it introduced meta programming concept into task definition.

**Warning:** Caution! pyKook is currently under experimental. It means that the design and specification of pyKook may change without prior notice.

### makefile

**See Also:**
- [http://en.wikipedia.org/wiki/Make_%28software%29](http://en.wikipedia.org/wiki/Make_%28software%29)

### Makefile generators

**See Also:**
- [http://en.wikipedia.org/wiki/Make_%28software%29](http://en.wikipedia.org/wiki/Make_%28software%29)

### qmake

**See Also:**

qmake generate 3 Makefiles from a qtcreator project:
- Makefile
- Makefile.Debug
- Makefile.Release

**Example**

```bash
c:\qtsdk_1.1.3\desktop\qt\4.7.4\mingw\bin\qmake.exe -spec c:\QtSDK_1.1.3\Desktop\Qt\4.7.4\mingw\mkspecs\win32-g++ qmake tutorial
```

**See Also:**

qmake is a tool that helps simplify the build process for development project across different platforms.

**qmake automates the generation of Makefiles** so that only a few lines of information are needed to create each Makefile.

qmake can be used for any software project, whether it is written in Qt or not.

qmake generates a Makefile based on the information in a project file.

Project files are created by the developer, and are usually simple, but more sophisticated project files can be created for complex projects.

qmake contains additional features to support development with Qt, automatically including build rules for moc and uic.

**qmake can also generate projects for Microsoft Visual studio** without requiring the developer to change the project file.
maven

See Also:

- http://maven.apache.org/
- http://maven.apache.org/what-is-maven.html
- http://en.wikipedia.org/wiki/Maven

A maven (also mavin) is a trusted expert in a particular field, who seeks to pass knowledge on to others. The word maven comes from the Hebrew (מaven), via Yiddish, and means one who understands, based on an accumulation of knowledge.

Apache Maven est un outil logiciel libre pour la gestion et l’automatisation de production des projets logiciels Java en général et Java EE en particulier. L’objectif recherché est comparable au système Make sous Unix : produire un logiciel à partir de ses sources, en optimisant les tâches réalisées à cette fin et en garantissant le bon ordre de fabrication.

Il est semblable à l’outil Ant, mais fournit des moyens de configuration plus simples, eux aussi basés sur le format XML. Maven est géré par l’organisation Apache Software Foundation. Précédemment Maven était une branche de l’organisation Jakarta Project.

Introduction

See Also:

http://maven.apache.org/what-is-maven.html

Maven, a Yiddish word meaning accumulator of knowledge, was originally started as an attempt to simplify the build processes in the Jakarta Turbine project. There were several projects each with their own Ant build files that were all slightly different and JARs were checked into CVS. We wanted a standard way to build the projects, a clear definition of what the project consisted of, an easy way to publish project information and a way to share JARs across several projects.

The result is a tool that can now be used for building and managing any Java-based project. We hope that we have created something that will make the day-to-day work of Java developers easier and generally help with the comprehension of any Java-based project.

Maven’s Objectives

Maven’s primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal there are several areas of concern that Maven attempts to deal with:

- Making the build process easy
- Providing a uniform build system
- Providing quality project information
- Providing guidelines for best practices development
- Allowing transparent migration to new features

Maven does encourage best practices, but we realise that some projects may not fit with these ideals for historical reasons. While Maven is designed to be flexible, to an extent, in these situations and to the needs of different projects, it can not cater to every situation without making compromises to the integrity of its objectives.

2.2. Building software
Project Object Model (POM)

Chaque projet ou sous-projet est configuré par un POM qui contient les informations nécessaires à Maven pour traiter le projet (nom du projet, numéro de version, dépendances vers d'autres projets, bibliothèques nécessaires à la compilation, noms des contributeurs etc.).

Ce POM se matérialise par un fichier pom.xml à la racine du projet. Cette approche permet l’héritage des propriétés du projet parent. Si une propriété est redéfinie dans le POM du projet, elle recouvre celle qui est définie dans le projet parent. Ceci introduit le concept de réutilisation de configuration. Le fichier pom du projet principal est nommé pom parent.

Examples

javal maven project  See Also:
http://code.google.com/p/javal/wiki/GettingStarted

Premake

See Also:
• http://fr.wikipedia.org/wiki/Premake
• http://industriousone.com/premake

Ninja

Ninja is a small build system closest in spirit to Make.

See the manual: http://martine.github.com/ninja/manual.html or manual.asciidoc included in the distribution for more background, including motivation and build instructions.

Though the code is copyright Google, don’t take that as an endorsement; I wrote this in my spare time for fun.
• http://news.ycombinator.com/item?id=2186392
• https://github.com/martine/ninja
• http://neugierig.org/software/chromium/notes/2011/02/ninja.html

qmake

See Also:
• http://doc.trolltech.com/3.0/qmake.html
• http://fr.wikipedia.org/wiki/Qt#qmake
• http://mingw-cross-env.nongnu.org/#tutorial
• http://paulf.free.fr/undocumented_qmake.html
• http://www.qtcentre.org/wiki/index.php?title=Deploying_Qt_Applications
Step 5c: Cross compile your Project (Qt)  If you have a Qt application, all you have to do is:

i686-pc-mingw32-qmake
make

If you are using Qt plugins such as the svg or ico image handlers, you should also have a look at the Qt documentation about static plugins.

Some thoughts on Qt

See Also:

Qt Build Suite (Qbs), Cubes

See Also:
• http://chaos.troll.no/~dmolkent/qbs-0.1/index.html
• http://labs.qt.nokia.com/2012/02/15/introducing-qbs/
• http://labs.qt.nokia.com/2009/10/12/to-make-or-not-to-make-qmake-and-beyond/
• http://gamesfromwithin.com/the-quest-for-the-perfect-build-system

Introducing qbs

Over the years we have developed a kind of a love-hate relationship towards qmake. It does its job but also has its quirks and is widely considered unmaintainable.

The blog post [TMQB] contains a wish list for a qmake replacement. We have considered the various tools on the market, but none satisfied us – see for example [WNCM].

So some time ago we have initiated an internal project to try some of the ideas.

This is the outcome of it: the Qt Build Suite, aka qbs (pronounced as “Cubes”).

It is not qmake

Unlike qmake, qbs is not tied to the Qt version, and it generates a proper build graph (dependency graph) from the high-level project description in the project file.
Also, classical makefile generators like qmake and CMake create makefiles and leave the execution of the actual build commands to tools like make or ninja.

Qbs on the other hand fills the role of a parallel make and directly calls the compiler, linker and other tools, pretty much like SCons and Ant do.

**Redo**

- https://github.com/apenwarr/redo

**Scons**

See Also:

- http://www.scons.org/

SCons is a computer software construction tool that automatically analyzes source code file dependencies and operating system adaptation requirements from a software project description and generates final binary executables for installation on the target operating system platform. Its function is analogous to the traditional GNU build system based on the make utility and the autoconf tools.

**Tup**

- http://gittup.org/tup/make_vs_tup.html

### 2.3 Command Line interface

#### 2.3.1 Command Line Interface (CLI)

See Also:

- http://www.commandlinefu.com
- http://google-opensource.blogspot.com/2010/06/introducing-google-command-line-tool.html
- Python cli management

**Introduction**

A command-line interface (CLI) is a mechanism for interacting with a computer operating system or software by typing commands to perform specific tasks.

The concept of the CLI originated when teletypewriter machines (TTY) were connected to computers in the 1950s, and offered results on demand, compared to batch oriented mechanical punched card input technology. Dedicated text-based CRT terminals followed, with faster interaction and more information visible at one time, then graphical terminals enriched the visual display of information.

Currently personal computers encapsulate all three functions (batch processing, CLI, GUI) in software.
Usage

A CLI is used whenever a large vocabulary of commands or queries, coupled with a wide (or arbitrary) range of options, can be entered more rapidly as text than with a pure GUI.

CLI and python

CLI and python  See Also:
http://packages.python.org/pyCLI/

The cli package is a framework for making simple, correct command line applications in Python.

With cli, you can quickly add standard command line

• parsing;
• logging;
• unit and functional testing;
• and profiling to your CLI apps.

To make it easier to do the right thing, cli wraps all of these tools into a single, consistent application interface

Perl

See Also:

• http://www.activestate.com/activeperl/

Contents

• Perl
  • perl usage
    • windows .bat file
    • UNIX .sh file

perl usage

windows .bat file

perl -p -e ‘s/\_ext/\_DOTDOT/\_DOTDOT//g’ %1
perl -p -e ‘s/\/$\{CND_PLATFORM\}//g’ %1

UNIX .sh file

#!/usr/bin/sh

perl -pi -e ‘s/\_ext/\_DOTDOT/\_DOTDOT//g’ $1
perl -pi -e ‘s/\/$\{CND_PLATFORM\}//g’ $1

2.3. Command Line interface
ack  See Also:

- http://betterthangrep.com/
- https://github.com/petdance/ack

ack is a grep-like tool optimized for working with large trees of source code. ack is not intended to be a general-purpose replacement for grep. ack is for searching source code of defined file types. For searching other files, you may want to stick with grep.

**Command Line Interface (CLI) on GNU/linux**

See Also:

- http://tirania.org/blog/archive/2011/Sep-06.html (Learning Unix)
- Command Line Interface (CLI)

Unix shells  See Also:

- http://tirania.org/blog/archive/2011/Sep-06.html (Learning Unix)
- Command Line Interface (CLI)

**Bash: The shell for the GNU operating system**  Bash is a free software Unix shell written for the GNU Project. Its name is an acronym which stands for Bourne-again shell. The name is a pun on the name of the Bourne shell (sh), an early and important Unix shell written by Stephen Bourne and distributed with Version 7 Unix circa 1978, and “born again”. Bash was created in 1987 by Brian Fox. In 1990 Chet Ramey became the primary maintainer.

Bash is the shell for the GNU operating system from the GNU Project. It can be run on most Unix-like operating systems. It is the default shell on most systems built on top of the Linux kernel as well as on Mac OS X and Darwin. It has also been ported to Microsoft Windows using Subsystem for UNIX-based Applications (SUA), or POSIX emulation provided by Cygwin and MSYS.

See Also:


**Commands Line Interface**  See Also:

http://www.commandlinefu.com

**Research of text inside files**

```bash
find . -name "*.c" -exec grep -iHA5 -B5 "ioutil.h" {} \
```

**Renaming files**

```bash
rename .c .cpp *.c
```

**Unix shell commands**  See Also:

Command Line Interface (CLI)
Comptage de fichiers  See Also:

Command Line Interface (CLI)

Contents

- Comptage de fichiers
  - Compter le nombre de fichiers include dans un projet
  - Compter le nombre de fichiers C/C++ dans un projet

Compter le nombre de fichiers include dans un projet

find . -name "*.h" | wc -l

Compter le nombre de fichiers C/C++ dans un projet

find . -name '.*c' -o -name '.*cpp' | wc -l

Comptage de lignes  See Also:

Command Line Interface (CLI)

Contents

- Comptage de lignes
  - Compter le nombre de lignes ‘.h’ dans un projet
  - Compter le nombre de lignes ‘C/C++’ dans un projet
  - Compter le nombre de lignes python dans un projet

Compter le nombre de lignes ‘.h’ dans un projet

find . -name "*.h" -print | xargs wc -l

Compter le nombre de lignes ‘C/C++’ dans un projet

find . -name '.*c' -o -name '.*cpp' -print | xargs wc -l

Compter le nombre de lignes python dans un projet

find . -name '.*py' -print | xargs wc -l

Delete svn files

With the find command + hg rename

2.3. Command Line interface
find . -depth -name .svn -type d -exec rm -fr {} \\

Delete some data in a file

```
source guilde@guilde.asso.fr
::
Je cherche un moyen de supprimer un bloc de manière automatique d’un fichier de conf de proftpd.
Exemple de conf : Plein de lignes à garder <IfUser BLa> <Limit LOGIN> Allow 1.2.3.4 5.6.7.8... DenyAll 
</Limit> </IfUser> Plein de lignes à garder :
Je pense le faire avec sed mais je ne m’en sors pas. Je connais le début du bloc (<IfUser BLa>) et la fin du bloc 
(le premier </IfUser> après le début du bloc). [...]
```

Réponse d’Edgar

```
sed ’/<IfUser BLa>/,<\/IfUser>/d’
```

find strings in files recursively

```
find . -name "*.rst" -exec grep -iHA5 -B5 "string" {} \\
find . -name "*.rst" -exec grep -iH "string" {} \\
find . -name "*.rst" -exec grep "string" {} \\
```

Renaming files

rename command    See Also:

http://www.commandlinefu.com/commands/tagged/404/ rename

Rename files from .c to .cpp

```
rename ’s/\.c/\.cpp/’ *.c
```

Rename files from .MOD to .MPG

```
rename ’s/\.MOD/\.MPG/’ *.MOD
```

With the find command + hg rename

Rename files from ".txt" to ".rst"

```
find . -name "*.txt" | awk '{ newFile=gensub("txt$", "rst", 1); system("hg rename " $0 ' " newFile) }' \\
```

replace string in files (recursively or not)    See Also:

recursive replacing

Dry run (‘s’ option)

rpl -Rs -x’.rst’ '2010-2012' '2010-2013' .
rpl -Rs olstring newstring .

Replace (without ‘s’ option)

rpl -R -x’.rst’ '2010-2012' '2010-2013' .
rpl -R olstring newstring .

no recursive replacing

grep -rl oldstring . | xargs sed -i -e 's/olstring/newstring/'

bar  This is a small shell script intended to be used in portable Unix install scripts for showing progress bars. The overall goal is to write a minimally complex shell script (thus a program that needs no compilation) that is as robust as possible to work on as many Bourne shells and operating systems as possible, and that implements ‘cat’ with an ASCII progress bar and some other nifty features.

This is pure Bourne shell code. (For sh, ash, ksh, zsh, bash, ...)

The script is mainly indented to be used in portable install scripts, where you can use the body of the script.


Gestion des groupes sous GNU/Linux

Ajouter un utilisateur à un groupe sous GNU/Linux  See Also:


Petit pense-bête à usage interne: gérer les utilisateurs dans ses groupes sous GNU/Linux et en ligne de commande.

Modification du groupe primaire d’un utilisateur  Pour changer le groupe primaire de l’utilisateur nicolargo à admin, il suffit d’utiliser la commande usermod:

usermod -g admin nicolargo

Ajout d’un groupe secondaire à un utilisateur existant  Pour ajouter un groupe secondaire networkadmin à un utilisateur existant nicolargo, c’est encore la commande usermod qu’il faut utiliser:

usermod -a -g networkadmin nicolargo

Ajout d’un nouvel utilisateur à un groupe primaire  Pour ajouter le nouvel utilisateur ritchy et lui configurer un comme groupe primaire admin, il suffit d’utiliser la commande useradd:

useradd -g admin nicolargo

2.3. Command Line interface
Ajout d’un nouvel utilisateur à un groupe secondaire   Pour ajouter le nouvel utilisateur ritchy et lui configurer un comme groupe secondaire networkadmin, il suffit d’utiliser la commande useradd:

```
useradd -G networkadmin nicolargo
```

A noter qu’il est possible d’utiliser l’option -G avec plusieurs groupes.
Exemples pour ajouter ritchy au groupe secondaire networkadmin et systemadmin:

```
useradd -G networkadmin,systemadmin nicolargo
```

Vérifier les groupes associés à un utilisateur   Rien de plus simple avec la commande groups:

```
groups ritchy
```

```
ritchy: networkadmin systemadmin
```

Resizing image files   See Also:

http://www.commandlinefu.com/commands/tagged/1066/resize

**mogrify command   See Also:**

*Les différents outils d’ImageMagick*

Resize all JPEGs in a directory

This command requires the *imagemagick libraries* and will resize all files with the .JPG extension to a width of 800 pixels and will keep the same proportions as the original image.

```
mogrify -resize 800 *.JPG
```

google picasa use case

```
cp *.JPG web
cd web
mogrify -resize 800 *.JPG
date; google picasa -d "2011-10-27" -n "Minou et Ali à Annecy le 27 octobre 2011" -t "Annecy, 27 octobre 2011" create "Annecy, 27 octobre 2011" *.JPG; date
```

Extract sound from a vidéo with ffmpeg   See Also:

- *ffmpeg video library*

Source: Linux Partique numéro 63, p.23

Vous venez de récupérer une vidéo depuis un site de publications du type Youtube ou DailyMotion, dont vous aimeriez bien récupérer la bande son ?

C’est idéal dans le cas de clip vidéo d’un artiste que vous appréciez par exemple...

Dans ce cas, la commande suivante, utilisant *ffmpeg* vous sera très utile.
L'option **-i** permet de définir le fichier vidéo en entrée.
L'option **-ar** définit la fréquence d'échantillonnage (par défaut 44100Hz)
L'option **-ac** permet de définir le nombre de canaux audio
L'option **-ab** désigne le bitrate audio (64k par défaut)

**write in file with python**  
See Also:
http://hg.piranha.org.ua/sphinxedhg/docs/test-hgrc.html

Write in a file with python

```
$ TEST='pwd'/test_insert.txt
$ export TEST

python -c "print '\[foo\]
bar = a
b
 c
 de
 fg
nbaz = bif cb
\n'" > $TEST
```

**Terminator**  
See Also:

- http://tirania.org/blog/archive/2011/Sep-06.html (Learning Unix)
- Command Line Interface (CLI)

## Terminator

Terminator is a cross-platform GPL terminal emulator with advanced features not yet found elsewhere.
Terminator will run on any modern OS with Java 5 or later. It replaces xterm, rxvt, xwsh and friends on X11 systems, GNOME Terminal, KDE’s Konsole, Apple’s Terminal.app, and PuTTY on MS Windows.

## Downloads

- Windows x86 .msi (Windows 2000 to Windows 7)*
- Requires Cygwin and Cygwin Ruby and 32 bit Java. Setting up Cygwin.

See Also:

- http://software.jessies.org/terminator/
- http://www.commandlinefu.com/commands/browse
Open Terminator  Click on the Terminator icon:

Type commands

Command Line Interface (CLI) on Windows

See Also:

- [Windows tools](http://fr.wikipedia.org/wiki/Cmd.exe)
- [Command.com](http://fr.wikipedia.org/wiki/Command.com)
- [Console with multiple tabs](http://fr.wikipedia.org/wiki/Interface_en_ligne_de_commande)

Contents

• Console with multiple tabs
  – Description

Description

Console is a Windows console window enhancement.

Console features include: multiple tabs, text editor-like text selection, different background types, alpha and color-key transparency, configurable font, different window styles.

Path editors

See Also:

• Windows tools
• http://www.redfernplace.com/software-projects/patheditor/

Windows PowerShell

See Also:

• http://fr.wikipedia.org/wiki/Powershell

Windows PowerShell, anciennement Microsoft Command Shell (MSH), nom de code Monad, est une interface en ligne de commande et un langage de script développé par Microsoft.

Il est inclus dans Windows 7 (y compris la version grand public) et fondé sur la programmation orientée objet (et le framework Microsoft .NET).


PowerShell est compatible avec toutes les versions de Windows qui supportent la version 2.0 de .NET.

Depuis le 24 mars 2009, PowerShell 1.0 pour Windows XP et Vista est distribué comme une mise à jour logicielle facultative par le service Windows Update de Microsoft.

Il est intégré natively dans Windows 7 en version 2.0

Windows PowerShell versions

Windows PowerShell 3.0

See Also:


If you haven’t seen the new Windows PowerShell version 3 yet, you should make a point to check it out.

Don Jones

The new Windows PowerShell is coming. Actually, Microsoft has just launched a Community Technology Preview (CTP) of Windows PowerShell version 3, although the final version 3 probably won’t ship until it comes out with Windows 8.

It also will be available for Windows 7 and Windows Server 2008 R2. The CTP will install on those OSes.

2.3. Command Line interface
A CTP is an excellent point in the development cycle for you to start experimenting with new Microsoft technology. Product teams can still accept and act on feedback.

Don’t like some new piece of syntax ? Say something.

Wish a feature did just one more useful thing ? Let them know.

For Windows PowerShell, that feedback is usually best sent through Microsoft Connect, a site that’s being used by more and more product teams. That can seem like a black hole, because in many cases Microsoft can’t tell you what they’re doing with your feedback until they’ve shipped the next version of the product.

The Windows PowerShell team does indeed read that stuff, though. In fact, there’s a whole mess of improvements in version 3 that came directly from suggestions on Connect.

**CLI Subversion tools   See Also:**
http://tools.tortoisesvn.net/index.html

**Open a windows console with the stex extension   See Also:**
http://code.google.com/p/stexbar/downloads/list

![StEx](image)

**Wmic   See Also:**
- [http://timgolden.me.uk/python/wmi/cookbook.html](http://timgolden.me.uk/python/wmi/cookbook.html)

**System, BIOS, Motherboard   This first example shows a few variations of the most common WMI query.**

We ask a WMI object (computersystem, or bios, or baseboard in the examples below) to return the values for a few of its properties. It returns the results in its default tabular format.

```
wmic computersystem get domain, EnableDaylightSavingsTime, Manufacturer, Model, PartOfDomain, TotalPhysicalMemory, username
```

<table>
<thead>
<tr>
<th>Domain</th>
<th>EnableDaylightSavingsTime</th>
<th>Manufacturer</th>
<th>Model</th>
<th>PartOfDomain</th>
<th>TotalPhysicalMemory</th>
<th>username</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID3GEN</td>
<td>TRUE</td>
<td>Hewlett-Packard</td>
<td>HP Compaq dx2400 Microtower PC</td>
<td>TRUE</td>
<td>347</td>
<td></td>
</tr>
</tbody>
</table>
Google cli

See Also:
- http://code.google.com/p/googlecl/
- http://code.google.com/p/googlecl/wiki/ExampleScripts#Picasa
- google python data modules

GoogleCL brings Google services to the command line.
We currently support the following Google services:

Blogger
$ google blogger post --title "foo" "command line posting"

Calendar
$ google calendar add "Lunch with Jim at noon tomorrow"

Contacts
$ google contacts list Bob name,email > the_bobs.csv

Docs
$ google docs edit "Shopping list"

Finance
$ google finance create-txn "Savings Portfolio" NASDAQ:GOOG Buy

Picasa
$ google picasa create "Cat Photos" ~/photos/cats/*.

2.3. Command Line interface
Youtube

$ google youtube post --category Education killer_robots.avi

Dependencies  GoogleCL requires Python 2.5 or 2.6 and the gdata python client library.
You can get the library from the project homepage:
http://code.google.com/p/gdata-python-client/

Gcalcli  See Also:
http://code.google.com/p/gcalcli/
gcalcli is a Python application that allows you to access your Google Calendar from a command line. It's easy to get your agenda, search for events, and quickly add new events. Additionally gcalcli can be used as a reminder service to execute any application you want.

2.4 Communication

2.4.1 Communication

Internet

See Also:

• http://inl.info.ucl.ac.be/cnp3
• http://fr.wikipedia.org/wiki/Internet

The Internet is a global system of interconnected computer networks that use the standard Internet Protocol Suite (TCP/IP) to serve billions of users worldwide. It is a network of networks that consists of millions of private, public, academic, business, and government networks, of local to global scope, that are linked by a broad array of electronic, wireless and optical networking technologies

Wireless Communication

See Also:

• http://tech.chambana.net/projects/commotion

RFID

See Also:

• https://secure.wikimedia.org/wikipedia/fr/wiki/RFID
• RFID
• http://www.openpcd.org/Live
What does RFID mean? RFID is a special kind of wireless communication to identify or count an object contactless.

On one side you have a RFID-reader, like a terminal or handheld device. On the other side you have a transponder, like a tag or a label.

Passive RFID system In a passive RFID system, the reader sends out a field of energy and data. The transponder uses the energy and data to read out his memory and sends the content back to the reader.

Active RFID system At an Active-RFID system the transponder has its own battery, which allows much bigger memory sizes, a wider range and a faster communication.
### Parameters table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Low frequency</th>
<th>High frequency</th>
<th>Ultra High</th>
<th>Microwave frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>125 kHz</td>
<td>13.56 MHz</td>
<td>868 MHz</td>
<td>2.4 GHz</td>
</tr>
<tr>
<td>Reading distance (max.)</td>
<td>1 m</td>
<td>1.5 m</td>
<td>10 m</td>
<td>80 m (active)</td>
</tr>
<tr>
<td>Reading rate</td>
<td>slow</td>
<td>Depending on ISO-Standards</td>
<td>fast very</td>
<td>fast (active reader)</td>
</tr>
<tr>
<td>Humidity</td>
<td>No influence</td>
<td>No influence</td>
<td>Negative influence</td>
<td>Negative influence</td>
</tr>
<tr>
<td>Metal</td>
<td>Negative influence</td>
<td>Negative influence</td>
<td>No influence</td>
<td>No influence</td>
</tr>
<tr>
<td>ISO standards</td>
<td>• 11784/85</td>
<td>• 14443</td>
<td>• 14443,</td>
<td>18000-4</td>
</tr>
<tr>
<td></td>
<td>• 14223</td>
<td>• 15693</td>
<td>• 15693</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• 18000-2</td>
<td>• 18000-3</td>
<td>• 18000-6</td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>• Admission control,</td>
<td>• scrubber agreement</td>
<td>• Pallet collection</td>
<td>container tracking</td>
</tr>
<tr>
<td></td>
<td>• going away barrier</td>
<td>• asset management</td>
<td>• container tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• scrubber agreement</td>
<td>• ticketing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• gas reading</td>
<td>• tracking and tracing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• group collection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### RFID tools

**http://www.openpcd.org/Live**  The bootable Live RFID Hacking System contains a ready-to-use set of hacking tools for breaking and analyzing MIFARE Classic RFID cards and other well known card formats. It is built around PCSC-lite, the CCID free software driver and `libnfc` that gives you access to some of the most common RFID readers.

You can download a bootable LiveCD of the RFID Live Hacking System.
NFC definition  Near field communication, or NFC, allow for simplified transactions, data exchange, and connections with a touch.[1] Formed in 2004, the Near Field Communication Forum (NFC Forum) promotes sharing, pairing, and transactions between NFC devices[2] and develops and certifies device compliance with NFC standards.[3]

A smartphone or tablet with an NFC chip could make a credit card payment or serve as keycard or ID card. NFC devices can read NFC tags on a museum or retail display to get more information or an audio or video presentation. NFC can share a contact, photo, song, application, or video or pair Bluetooth devices.

Communication en champ proche  See Also:


La communication en champ proche (Near Field Communication), habituellement appelée NFC, est une technologie de communication sans-fil à courte portée et haute fréquence, permettant l’échange d’informations entre des périphériques jusqu’à une distance d’environ 10 cm.

Cette technologie est une extension de la norme ISO/CEI 14443 standardisant les cartes de proximité utilisant la RFID (Radio Frequency IDentification), qui combinent l’interface d’une carte à puce et un lecteur au sein d’un seul périphérique.

Un périphérique NFC est capable de communiquer autant avec le matériel ISO/CEI 14443 existant qu’avec un autre périphérique NFC, et est tout autant compatible avec les infrastructures sans-contact existantes déjà en utilisation dans les transports en commun et les terminaux de paiement.

La NFC est à la base conçue pour un usage dans les téléphones mobiles.
Welcome to the official website of libnfc. On this site you will find information concerning the open source library for Near Field Communication (NFC).

libnfc is the first free NFC SDK and Programmers API released under the GNU Lesser General Public License. It provides complete transparency and royalty-free use for everyone.

Since the RFID market is spoiled by proprietary hard and software we want to contribute constructively by distributing a free library which can be used for various RFID and NFC applications.

This website mainly focuses itself on researches and developers that want to work with NFC hardware but don’t want the pain of programming NFC software on a very low level.

Furthermore, this library does not require any license fee or non-disclosure agreement to be signed for. This list shows the current supported features.

We hope this moves the NFC development to a more open culture where discussions about applications and techniques can be shared among the users.

libnfc sources See Also:

http://code.google.com/p/libnfc/

This is download/development site of libnfc, the Free/Libre Near Field Communication (NFC) Library

libnfc links See Also:

http://www.libnfc.org/links

http://code.google.com/p/pynfc/ See Also:

http://code.google.com/p/pynfc/source/browse/#svn%2Ftrunk%2Fsrc

Pynfc provides a pythonic interface for the libnfc library, allowing access to ISO-14443a and similar RFID/NFC cards supported by libnfc.

opencd.org live The bootable Live RFID Hacking System contains a ready-to-use set of hacking tools for breaking and analyzing MIFARE Classic RFID cards and other well known card formats. It is built around PCSC-lite, the CCID free software driver and libnfc that gives you access to some of the most common RFID readers.

NFC use cases

paiement See Also:


On l’avait vu la semaine dernière, les technologies de type NFC ne sont pas uniquement une occasion de révolutionner le porte monnaie traditionnel.

Aujourd’hui, j’aimerais partager avec vous d’autres opportunités qu’amènent ces technologies de communication en champs proche.
**votre téléphone = votre exposition personnalisée**  On connait tous le guide audio des musée qui nous donne des informations dans diverses langue.. Et bien ça pourrait très vite devenir une relique du passé (pourquoi pas un musée là-dessus !)

Avec un mobile NFC, on va directement au bureau d’accueil, on secoue son téléphone NFC et on télécharge un guide audio directement sur son smartphone.

Secouer son téléphone devant une pièce du même musée donnera de la même manière des informations.

Si on prend nos bottes de sept lieux, ce même téléphone sera la possibilité pour ce même musée de faire des expositions interactives, d’offrir des bons de réduction pour le magasin de souvenir du musée, etc.

Nokia a déjà commencé cette démarche avec le musée de Londres : en touchant des tags dispersés dans le musée, le visiteur peut s’enfoncer dans l’histoire du vieux Londres grâce à des expositions interactives.

**votre téléphone = vos clés de voiture**  De nombreux constructeurs automobiles (ou apparentés) tels que Continental, NXP ou BMW vont déjà dans la direction de clés NFC. Il sera possible de démarrer sa voiture ou la chauffer seulement grâce à cette clé.

Pendant que certaines entreprises cherchent à remplacer les clés traditionnelles par un téléphone NFC, Continental et NXP offrent la possibilité de secouer ses clés au-dessus de son smartphone pour accéder à des données pratiques : une carte pour retrouver sa voiture, un GPS, l’état de la voiture et un utilitaire de gestion des problèmes associés.

Et si on veut prêter sa voiture à quelqu’un ? Facile. On envoie un code et la voiture sait qui attendre : l’individu à qui on a envoyé le code peut utiliser la voiture.

**votre téléphone = vos promotions personnalisées**  Tout l’intérêt des bons de réduction réside dans le fait que les consommateurs possèdent un panorama de produits à prix réduits, le tout avec un risque personnel très faible. Imaginons maintenant que l’on utilise ce système dans une rue piétonne. En utilisant les technologies de géolocalisation, ces mêmes consommateurs pourraient se connecter à un service leur permettant de recevoir des bons de réductions ciblés géographiquement.

Ca marche comme ça : on accepte le message de ce service de bons de réduction et... Paf ! On se retrouve dans une nouvelle ville pour la première fois. On descend la rue et là, on reçoit un texto. Celui-ci nous invite à prendre un café gratuit au troquet du coin si on goûte un de leurs gâteaux. Tout ce que l’on a à faire est de se rendre au café en question, commander le tout et secouer son téléphone NFC devant un terminal. Ce dernier reconnaîtra le téléphone et la transaction sera faite, réductions comprises.

**bien plus qu’un téléphone**  Les technologies type NFC vont donc bien plus loin que le simple téléphone qui servait à... appeler ! Ceci dit, je suis sûr qu’il existe tout un tas d’applications possibles à ces nouvelles technologies. Si vous avez une idée, en tête, partagez-là ! ;-)  

Stewart

**téléphone**  See Also:  


La technologie du NFC, permettant la transmission sans fil d’informations à courte distance, trouve de plus en plus de marches et un fabricant pourrait accélérer les choses en facilitant la compatibilité des téléphones.

Principalement promue comme le nouveau standard pour effectuer facilement et rapidement des paiements électroniques et notamment depuis les smartphones de nouvelle génération, la technologie du NFC nécessite néanmoins d’avoir un téléphone compatible. Les derniers terminaux haut de gamme embarquent peu à peu la puce nécessaire mais malgré plusieurs expériences concluantes, la technologie reste encore loin d’être déployée massivement sur le Vieux Continent et encore moins dans les pays en voie de développement.

2.4. Communication
Le fabricant Inside Secure annonce cependant avoir réussi à embarquer la technologie du NFC directement au sein des cartes SIM traditionnelles, un tour de force qui, s’il s’avère satisfaisant, pourrait rendre compatibles même les plus vieux téléphones mobiles toujours en circulation sur le marché.

L’antenne NFC développée par Inside Secure ne mesurerait que 5 x 10 mm et trouverait alors sa place directement sur la carte SIM du téléphone.

Dans un communiqué officiel, Charles Walton, directeur des opérations chez Inside Secure, déclare : « Cette technologie de rupture va nous permettre de mettre en place des services NFC sur des marchés qui n’auraient pas pu être atteints immédiatement par le biais des ventes de smartphones, comme le Brésil, l’Inde, les pays africains ou n’importe quel autre marché avec une vaste base de téléphones portables pré 3G ».

Ce n’est pas la première fois que nous entendons parler d’un tel dispositif. Au mois de septembre Jason Rees, directeur de la division Mobile Payments & Ticketing chez Orange UK, expliquait avoir adopté une stratégie similaire.

En affirmant que l’utilisateur serait en mesure de changer de téléphone tout en conservant la technologie, il soulignait que cela contribuerait à fidéliser davantage le client.
Serial Communication

See Also:

- http://www.webalice.it/fede.tft/serial_port/serial_port.html

The serial port protocol is one of the most long lived protocols currently in use. According to wikipedia, it has been standardized in 1969. First, a note: here we’re talking about the RS232 serial protocol. This note is necessary because there are many other serial protocols, like SPI, I2C, CAN, and even USB and SATA.

Some time ago, when the Internet connections were done using a 56k modem, the serial port was the most common way of connecting a modem to a computer. Now that we have ADSL modems, the serial ports have disappeared from newer computers, but the protocol is still widely used.

In fact, most microcontrollers, even the newer ones have one or more peripherals capable of communicating using this protocol, and from the PC side, all operating systems provide a way of interfacing with serial devices. The problem of the lack of serial ports on computers was solved with USB to serial converters, often embedded into the device itself.

One of such devices is the Arduino. While the first models had a serial port connector, newer models has an USB port. However, nothing has changed on the microcontroller side, nor on the PC side.

Simply the newer Arduinos have a chip that performs the serial to USB conversion.

Serial C++ Communication

SerialPort See Also:

- http://www.webalice.it/fede.tft/serial_port/serial_port.html
- http://gitorious.org/serial-port

This article explains how to interface with serial ports from C++. Instead of presenting an API specific to a single operating system, the Boost.asio library was used. This library provides portable, high performance implementation of sockets and serial ports.
The chosen presentation is to provide code with a growing level of scalability, and a growing level of complexity. It starts with a simple class to wrap Asio’s serial ports to provide string write and read, and expands to topics like binary data transfer, timeouts, and asynchronous operation.

All the example presented here are available for download at the end of the article.

This article presents some classes that wrap the asio library. They can be used as-is, or its implementation can be studied to understand how to perform these tasks with asio.

**Serial Python Communication**

![PySerial Logo](image)

See Also:

- [http://wiki.python.org/moin/PySerial](http://wiki.python.org/moin/PySerial)

This module encapsulates the access for the serial port. It provides backends for Python running on Windows, Linux, BSD (possibly any POSIX compliant system), Jython and IronPython (.NET and Mono).

The module named “serial” automatically selects the appropriate backend.

**Serial port python examples**

**RS232 simulator**

The simulator python pyBGL2SIM_gen module

voir P5E627/Concept/Soft/PC/XLOG9X324_python_Simul

**Module csv**

See Also:

http://docs.python.org/library/csv.html#module-csv

**Module collections**

```python
from collections import namedtuple

BeagleRecord = namedtuple('BeagleRecord', 'Index, Time, Dur, Len, Err, Dev, Ep, Level, Record, Data')

def beagle_records(filename):
    for beagle_record in map(BeagleRecord._make, csv.reader(open(filename, "rb"))):
        yield beagle_record
```

---

66 Chapter 2. Development
collections.namedtuple (typename, field_names, verbose=False, rename=False)

Returns a new subclass of tuple with named fields.

```python
globals()['Point'] = namedtuple('Point', 'x y')
Point.__doc__
# docstring for the new class
p = Point(11, y=22)
# instantiate with positional args or keywords
p[0] + p[1]
# indexable like a plain tuple
x, y = p
# unpack like a regular tuple
x, y (11, 22)
p.x + p.y
# fields also accessible by name
33
d = p._asdict()
# convert to a dictionary
d['x']
11
Point(**d)
# convert from a dictionary
Point(x=11, y=22)
p._replace(x=100)
# _replace() is like str.replace() but targets named fields
Point(x=100, y=22)
```

Other example

```python
EmployeeRecord = namedtuple('EmployeeRecord', 'name, age, title, department, paygrade')
import csv
for emp in map(EmployeeRecord._make, csv.reader(open("employees.csv", "rb"))):
    print emp.name, emp.title
import sqlite3
conn = sqlite3.connect('/companydata')
cursor = conn.cursor()
cursor.execute('SELECT name, age, title, department, paygrade FROM employees')
for emp in map(EmployeeRecord._make, cursor.fetchall()):
    print emp.name, emp.title
```

See Also:
collections documentation: http://docs.python.org/library/collections.html
namedtuple: http://docs.python.org/library/collections.html#collections.namedtuple

Module serial  See Also:
pyserial documentation: http://pyserial.sourceforge.net/pyserial.html

Module binascii

```python
binascii.a2b_hex (hexstr) → s; Binary data of hexadecimal representation.
hexstr must contain an even number of hex digits (upper or lower case). This function is also available as
"unhexlify()"
```

Module string

```python
string.translate (s, table[, deletions]) → string
Return a copy of the string s, where all characters occurring in the optional argument deletions are removed, and
```
the remaining characters have been mapped through the given translation table, which must be a string of length 256. The deletions argument is not allowed for Unicode strings.

Module time

time.clock() → floating point number
Return the CPU time or real time since the start of the process or since the first call to clock(). This has as much precision as the system records.

time.sleep(seconds)
Delay execution for a given number of seconds. The argument may be a floating point number for subsecond precision.

pyBGL2SIM_gen.py

def beagle_records(filename):
    try:
        for beagle_record in map(BeagleRecord._make, csv.reader(open(filename, "rb"))):
            yield beagle_record
    except Exception as e:
        print('can t open file', filename)

if __name__ == '__main__':
    try:
        print("Opening serial port ...")
        ser = serial.Serial(COMPortNumber-1, 115200)
        try:
            Table = string.maketrans('','')
            RemoveNonHexCharsTable = string.translate(Table, Table, string.hexdigits)
            CheckBadCharTable = string.translate(Table, Table, string.punctuation)
UniformSepTable = string.maketrans(string.punctuation, ' ' * len(string.punctuation))

LastTime = time.clock()
PreviousTimeStamp = None

nb_records = 0;
for beagle_record in beagle_records(beagle_file):
    nb_records = nb_records + 1
    print('.', end = '')
    if (nb_records % 200) == 0:
        print('

', nb_records, ' beagle records')
    if nb_records > nb_max_records:
        break;

SplittedTime = string.translate(beagle_record.Time, UniformSepTable).split()
Minutes, Secondes, Millisecondes = int(SplittedTime[0]), int(SplittedTime[1]), int(SplittedTime[2])
Microsecondes, Nanosecondes = int(SplittedTime[3]), int(SplittedTime[4])

CurrentTimeStamp = (Minutes * 60) + Secondes + (Millisecondes / 1000.0) + (Microsecondes / 1000000.0) + (Nanosecondes / 1000000000.0)

if PreviousTimeStamp is not None:
    if CurrentTimeStamp > PreviousTimeStamp:
        time.sleep(CurrentTimeStamp - PreviousTimeStamp)
    else:
        PreviousTimeStamp = CurrentTimeStamp
else:
    PreviousTimeStamp = CurrentTimeStamp

if len(string.translate(beagle_record.Data, Table, CheckBadCharTable)) == 0:
    try:
        RawData = binascii.a2b_hex(string.translate(beagle_record.Data, Table, RemoveNonHexCharsTable))
        ser.write(RawData)
    except TypeError:
        pass

    PreviousTimeStamp = CurrentTimeStamp
finally:
    print('

', nb_records, 'records processed')
    print('Closing serial port ...')
    ser.close()
except serial.SerialException, e:
    print('Unable to open COM port %d, please check port number' % COMPortNumber)

Code source de pyBGL2SIM_gen.py  Download pyBGL2SIM_gen.py

# -*- coding: utf-8 -*-

"""
This module simulates the input of case frames by reading a file of records captured by the 'Beagle USB Protocol Analyzer' <http://www.totalphase.com/support/product/beagle_ism/>
- the fake frames are sent to a COM5 port.
- the reader (not provided here) reads a COM1 port.
"""

from __future__ import print_function

# Import standard modules
import string
import csv
import binascii
import time

# Import addons modules
import serial

## COM port number
# Please defines here the correct COM port number.
COMPortNumber = 5

from collections import namedtuple

BeagleRecord = namedtuple('BeagleRecord', 'Index, Time, Dur, Len, Err, Dev, Ep, Level, Record, Data')

def beagle_records(filename):
    print('opening the', filename, 'beagle file')
    try:
        for beagle_record in map(BeagleRecord._make, csv.reader(open(filename, "rb"))):
            yield beagle_record
    except Exception as e:
        print('can t open file', filename)

beagle_file = "CR_beagle_2009_11_09_all_gen.csv"

if __name__ == '__main__':
    try:
        print("Opening serial port ...")
        ser = serial.Serial(COMPortNumber-1, 115200)
    except Exception as e:
        print('can t open serial port', e)

    Table = string.maketrans('','')
    RemoveNonHexCharsTable = string.translate(Table, Table, string.hexdigits)
    CheckBadCharTable = string.translate(Table, Table, string.punctuation)
    UniformSepTable = string.maketrans(string.punctuation,' '.ljust(len(string.punctuation)))

    LastTime = time.clock()
    PreviousTimeStamp = None

    nb_records=0;
    for beagle_record in beagle_records(beagle_file):
        nb_records=nb_records+1
        print('.', end = '')
        if ((nb_records % 200) == 0):
            print("", nb_records, " beagle records")
        if (nb_records > nb_max_records):
            break;
        SplittedTime = string.translate(beagle_record.Time, UniformSepTable).split()
        Minutes, Secondes, Millisecondes = int(SplittedTime[0]), int(SplittedTime[1]), int(SplittedTime[2])
        Microsecondes, Nanosecondes = int(SplittedTime[3]), int(SplittedTime[4])
        CurrentTimeStamp = (Minutes * 60) + Secondes + (Millisecondes / 1000.0) + (Microsecondes / 1000000.0) + (Nanosecondes / 1000000000.0)
# to comment if no delay between frames.
if PreviousTimeStamp is not None:
    if CurrentTimeStamp > PreviousTimeStamp:
        time.sleep(CurrentTimeStamp - PreviousTimeStamp)
    else:
        PreviousTimeStamp = CurrentTimeStamp
else:
    PreviousTimeStamp = CurrentTimeStamp

if len(string.translate(beagle_record.Data, Table, CheckBadCharTable)) == 0:
    try:
        RawData = binascii.a2b_hex(string.translate(beagle_record.Data, Table, RemoveNonHexCharsTable))
        ser.write(RawData)
    except TypeError:
        pass

finally:
    PreviousTimeStamp = CurrentTimeStamp

print("\n", nb_records, "records processed")
print("Closing serial port ...")
ser.close()
except serial.SerialException, e:
    print("Unable to open COM port \d, please check port number" % COMPortNumber)

Twisted serial service  You should be able to implement your own Service class and create your SerialPort instances in the startService method. e.g. (untested)

```
from twisted.application import service
from twisted.internet import reactor
from twisted.internet.serialport import SerialPort
from twisted.protocols.basic import LineReceiver

class SerialService(service.Service):
    def startService(self):
        self.serial = SerialPort(LineReceiver, '/dev/tty/serialport', reactor)

multiService = service.MultiService()
serialService = SerialService()
serialService.setServiceParent(multiService)

# Add some other services...

application = service.Application("Serial MultiService Example")
multiService.setServiceParent(application)
```

See Also:

Hi,I try to use twisted for asynchronous serial port read().

Here is my simple twisted code:
```
from twisted.internet import reactor
from twisted.internet.serialport import SerialPort
```
from twisted.internet.protocol import Protocol

class Client(Protocol):
    def connectionMade(self):
        print('connected')

    def dataReceived(self, data):
        print(data)

SerialPort(Client(), 'COM4', reactor)
reactor.callLater(20, reactor.stop())
reactor.run()

==========================

It runs on my Mac with this line changed to SerialPort(Client,
'dev/tty.usbserial-A4001mLL', reactor) which is my Arduino board device.

But on Windows XP, it gave the error message: "AttributeError: 'module' object has no attribute 'addEvent'"

I tried to used different reactor, such as selectreactor, win32eventreactor, but the error message became "AssertionError: reactor already installed"

> Can twisted be used to access a serial port on OS X, MS Windows, linux, etc ???

Yes, via the twisted.internet.serial module.

> Is there any support for serial protocols (eg. BISYNC, PPP, etc) ???

You’d have to write them yourself, except for a couple, e.g. Logitech Serial Mouse
(http://twistedmatrix.com/trac/browser/tags/releases/twisted-8.2.0/twisted/protocols/mice/mouseman.py)

> Can twisted be used to create custom serial protocols ???

Yes. For example,
http://twistedmatrix.com/projects/core/documentation/examples/mouse.py shows how to use the mouse protocol.

> I’ve been able to create an application in Linux that reads/writes
> multiple serial ports asynchronously. The setup code that does this looks
> like this:
> ...
> from twisted.internet.qtreactor import install
> a = QApplication(argv)
> install(a)
> from twisted.internet import reactor
> from twisted.internet.serialport import SerialPort
> ...
> ports, badPorts = getGoodPorts()
> if not ports:
>     exit(1)
> data = ConfigData(join(sep, 'etc', 'qa.conf'))
dbInfo = copy(data['qadata'])
getLogin(dbInfo)
w = MainWindow(data, dbInfo, ports)
w.show()
reactor.addSystemEventTrigger('after', 'shutdown', a.quit)
a.connect(a, SIGNAL('lastWindowClosed()'), reactor.stop)
for portObj in w.portObjs:
    SerialPort(portObj.scanner, portObj.port, reactor, baudrate=38400)
    portObj.sendLine()
reactor.run()

Where the portObj.scanner is an instance of a descendent of Protocol.
Like I said, the above code works under Linux. Then I tried porting this
to Windows. The first problem I came across is that the qtreactor.py
would not work. I had to subclass QTReactor from Win32Reactor. It runs
without errors. However, I am not reading anything off of the serial
port. I can see the lights blink on the port when the portObj.sendLine()
is called, so I believe I am writing to it ok and data is coming back.
The data is just not read by the application. I think it must have
something to do with the SerialPort instance not getting an even that
data is ready. I suspect this is a Windows issue in that Windows is not
signaling an event when data is ready to read on the serial port.

Does anybody have any experience with this? Is there a work around? Am I
doing something wrong?

Well, I found out that it has something to do with the qtreactor. If I
use just a Win32Reactor, it will read/write the serial port just fine. I
played around with writing to the serial port using
scanner.transport.write('
') and found that
Win32Reactor.doWaitForMultipleEvents() is called when a Win32Reactor is
used. It is not called when a QTReactor(Win32Reactor) is used. The
question is why is this the case?

Ok, I was able to find a solution to my problem. I had to make some
changes to the original qt4reactor.py (which I renamed to qtreactor.py
and copied over the original twisted qtreactor.py). The changes I made
are:

1) Subclassed QTReactor from Win32Reactor.
2) Commented out the following methods so the base class methods are
   used:
   addReader()
   addWriter()
   removeReader()
   removeWriter()
   removeAll()
3) Modified the QTReactor.simulate() method so it calls doIteration()
after calling runUntilCurrent().

I've tested this with two devices and it works great. After I test this
in a production environment with six devices, I can create a patch
against the original qt4reactor.py. I'd like to make this available to
others who might need it. Where is the best place to post the patch?
This mailing list?
How can I access the reactor if I want to use the win32eventreactor? It seems I have to win32eventreactor.install() before importing the reactor from twisted.internet, otherwise I get a traceback saying AssertionError: reactor already installed.

Is this the recommended way to do it:

```python
from twisted.internet import win32eventreactor
win32eventreactor.install()
from twisted.internet import reactor
```

### pyserial examples

**Serial Python Communication**

```python
#!/usr/bin/env python

"""Enhanced Serial Port class
part of pyserial (http://pyserial.sf.net)  (C)2002 cliechti@gmx.net

another implementation of the readline and readlines method.
this one should be more efficient because a bunch of characters are read
on each access, but the drawback is that a timeout must be specified to
make it work (enforced by the class __init__).

this class could be enhanced with a read_until() method and more
like found in the telnetlib.
""

from serial import Serial

class EnhancedSerial(Serial):
    def __init__(self, *args, **kwargs):
        #ensure that a reasonable timeout is set
        timeout = kwgars.get('timeout',0.1)
        if timeout < 0.01: timeout = 0.1
        kwgars['timeout'] = timeout
        Serial.__init__(self, *args, **kwgars)
        self.buf = ''

    def readline(self, maxsize=None, timeout=1):
        """maxsize is ignored, timeout in seconds is the max time that is way for a complete line""

        tries = 0
        while 1:
            self.buf += self.read(512)
            pos = self.buf.find('
')
            if pos >= 0:
                line, self.buf = self.buf[:pos+1], self.buf[pos+1:]
                return line
            tries += 1
            if tries * self.timeout > timeout:
                break
                line, self.buf = self.buf, ''
        return line
```
def readlines(self, sizehint=None, timeout=1):
    """read all lines that are available. abort after timeout
    when no more data arrives.""
    lines = []
    while 1:
        line = self.readline(timeout=timeout)
        if line:
            lines.append(line)
        if not line or line[-1:] != '\n':
            break
    return lines

if __name__=='__main__':
    #do some simple tests with a Loopback HW (see test.py for details)
    PORT = 0
    #test, only with Loopback HW (shortcut RX/TX pins (3+4 on DSUB 9 and 25) )
    s = EnhancedSerial(PORT)
    #write out some test data lines
    s.write(\'\n\'.join("hello how are you".split()))
    #and read them back
    print s.readlines()
    #this one should print an empty list
    print s.readlines(timeout=0.4)

scan.py  download scan.py

#!/usr/bin/env python
"""
Scan for serial ports.

Part of pySerial (http://pyserial.sf.net)
(C) 2002-2003 <cliechti@gmx.net>

The scan function of this module tries to open each port number
from 0 to 255 and it builds a list of those ports where this was
successful.
"""

import serial

def scan():
    """scan for available ports. return a list of tuples (num, name)""
    available = []
    for i in range(256):
        try:
            s = serial.Serial(i)
            available.append( (i, s.portstr) )
            s.close()  # explicit close 'cause of delayed GC in java
        except serial.SerialException:
            pass
    return available

if __name__=='__main__':
    print "Found ports:"
    for n,s in scan():
        print "(%d) %s" % (n,s)
scanlinux.py  download scanlinux.py

```python
#!/usr/bin/env python

"""
Scan for serial ports. Linux specific variant that also includes USB/Serial adapters.

Part of pySerial (http://pyserial.sf.net)
(C) 2009 <cliechti@gmx.net>
"""

import serial
import glob

def scan():
    """scan for available ports. return a list of device names.
    """
    found_ports = glob.glob('/dev/ttyS*') + glob.glob('/dev/ttyUSB*') + glob.glob('/dev/ttyACM*') + glob.glob('/dev/serial/by-id/*')
    if __name__ == '__main__':
        print "Found ports:
        for name in scan():
            print name

scanwin32.py  download scanwin32.py

import ctypes
import re

def ValidHandle(value):
    if value == 0:
        raise ctypes.WinError()
    return value

NULL = 0
HDEVINFO = ctypes.c_int
BOOL = ctypes.c_int
CHAR = ctypes.c_char
PCTSTR = ctypes.c_char_p
HWND = ctypes.c_uint
DWORD = ctypes.c_ulong
PDWORD = ctypes.POINTER(DWORD)
ULONG = ctypes.c_ulong
ULONG_PTR = ctypes.POINTER(ULONG)
#~ PBYTE = ctypes.c_char_p
```
PBYTE = ctypes.c_void_p

class GUID(ctypes.Structure):
    _fields_ = [
        ('Data1', ctypes.c_ulong),
        ('Data2', ctypes.c_ushort),
        ('Data3', ctypes.c_ushort),
        ('Data4', ctypes.c_ubyte*8),
    ]
    def __str__(self):
        return "{:%08x-%04x-%04x-%s-%s}" % (self.Data1,
            self.Data2,
            self.Data3,
            ''.join(['%02x' % d for d in self.Data4[:2]]),
            ''.join(['%02x' % d for d in self.Data4[2:]])

class SP_DEVINFO_DATA(ctypes.Structure):
    _fields_ = [
        ('cbSize', DWORD),
        ('ClassGuid', GUID),
        ('DevInst', DWORD),
        ('Reserved', ULONG_PTR),
    ]
    def __str__(self):
        return "ClassGuid:%s DevInst:%s" % (self.ClassGuid, self.DevInst)
PSP_DEVINFO_DATA = ctypes.POINTER(SP_DEVINFO_DATA)

class SP_DEVICE_INTERFACE_DATA(ctypes.Structure):
    _fields_ = [
        ('cbSize', DWORD),
        ('InterfaceClassGuid', GUID),
        ('Flags', DWORD),
        ('Reserved', ULONG_PTR),
    ]
    def __str__(self):
        return "InterfaceClassGuid:%s Flags:%s" % (self.InterfaceClassGuid, self.Flags)
PSP_DEVICE_INTERFACE_DATA = ctypes.POINTER(SP_DEVICE_INTERFACE_DATA)
PSP_DEVICE_INTERFACE_DETAIL_DATA = ctypes.c_void_p

class dummy(ctypes.Structure):
    _fields_ = ["d1", DWORD], ["d2", CHAR]
    _pack_ = 1
SIZEOF_SP_DEVICE_INTERFACE_DETAIL_DATA_A = ctypes.sizeof(dummy())

SetupDiDestroyDeviceInfoList = ctypes.windll.setupapi.SetupDiDestroyDeviceInfoList
SetupDiDestroyDeviceInfoList.argtypes = [HDEVINFO]
SetupDiDestroyDeviceInfoList.restype = BOOL

SetupDiGetClassDevs = ctypes.windll.setupapi.SetupDiGetClassDevsA
SetupDiGetClassDevs.argtypes = [ctypes.GUID, PCTSTR, HWND, DWORD]
SetupDiGetClassDevs.restype = ValidHandle # HDEVINFO

SetupDiEnumDeviceInterfaces = ctypes.windll.setupapi.SetupDiEnumDeviceInterfaces
SetupDiEnumDeviceInterfaces.argtypes = [HDEVINFO, PSP_DEVINFO_DATA, ctypes.GUID, DWORD, PSP_DEVICE_INTERFACE_DATA]

2.4. Communication 77
SetupDiEnumDeviceInterfaces.restype = BOOL

SetupDiGetDeviceInterfaceDetail = ctypes.windll.setupapi.SetupDiGetDeviceInterfaceDetailA
SetupDiGetDeviceInterfaceDetail.argtypes = [HDEVINFO, PSP_DEVICE_INTERFACE_DATA, PSP_DEVICE_INTERFACE_DETAIL_DATA, DWORD, PDWORD, PSP_DEVINFO_DATA]
SetupDiGetDeviceInterfaceDetail.restype = BOOL

SetupDiGetDeviceRegistryProperty = ctypes.windll.setupapi.SetupDiGetDeviceRegistryPropertyA
SetupDiGetDeviceRegistryProperty.argtypes = [HDEVINFO, PSP_DEVINFO_DATA, DWORD, PDWORD, PBYTE, DWORD, PDWORD]
SetupDiGetDeviceRegistryProperty.restype = BOOL

GUID_CLASS_COMPORT = GUID(0x86e0d1e0L, 0x8089, 0x11d0,
                      (ctypes.c_ubyte*8)(0x9c, 0xe4, 0x08, 0x00, 0x3e, 0x30, 0x1f, 0x73))

DIGCF_PRESENT = 2
DIGCF_DEVICEINTERFACE = 16
INVALID_HANDLE_VALUE = 0
ERROR_INSUFFICIENT_BUFFER = 122
SPDRP_HARDWAREID = 1
SPDRP_FRIENDLYNAME = 12
SPDRP_LOCATION_INFORMATION = 13
ERROR_NO_MORE_ITEMS = 259

def comports(available_only=True):
    """This generator scans the device registry for com ports and yields
    (order, port, desc, hwid). If available_only is true only return currently
    existing ports. Order is a helper to get sorted lists. it can be ignored
    otherwise."""
    flags = DIGCF_DEVICEINTERFACE
    if available_only:
        flags |= DIGCF_PRESENT
    g_hdi = SetupDiGetClassDevs(ctypes.byref(GUID_CLASS_COMPORT), None, NULL, flags);
    #~ for i in range(256):
    for dwIndex in range(256):
        did = SP_DEVICE_INTERFACE_DATA()
        did.cbSize = ctypes.sizeof(did)
        if not SetupDiEnumDeviceInterfaces(g_hdi,
                                            None,
                                            ctypes.byref(GUID_CLASS_COMPORT),
                                            dwIndex,
                                            ctypes.byref(did)):
            if ctypes.GetLastError() != ERROR_NO_MORE_ITEMS:
                raise ctypes.WinError()
            break
        dwNeeded = DWORD()
        # get the size
        if not SetupDiGetDeviceInterfaceDetail(g_hdi,
                                                ctypes.byref(did),
                                                None, 0, ctypes.byref(dwNeeded),
                                                None):
            # Ignore ERROR_INSUFFICIENT_BUFFER
            if ctypes.GetLastError() != ERROR_INSUFFICIENT_BUFFER:
raise ctypes.WinError()

# allocate buffer
class SP_DEVICE_INTERFACE_DETAIL_DATA_A(ctypes.Structure):
    _fields_ = [
        ('cbSize', DWORD),
        ('DevicePath', CHAR*(dwNeeded.value - ctypes.sizeof(DWORD))),
    ]
def __str__(self):
    return "DevicePath: %s" % (self.DevicePath,)

idd = SP_DEVICE_INTERFACE_DETAIL_DATA_A()
idd.cbSize = SIZEOF_SP_DEVICE_INTERFACE_DETAIL_DATA_A
devinfo = SP_DEVINFO_DATA()
devinfo.cbSize = ctypes.sizeof(devinfo)
if not SetupDiGetDeviceInterfaceDetail(g_hdi, ctypes.byref(did),
    ctypes.byref(idd), dwNeeded, None, ctypes.byref(devinfo)):
    raise ctypes.WinError()

# hardware ID
szHardwareID = ctypes.create_string_buffer(250)
if not SetupDiGetDeviceRegistryProperty(
    g_hdi,
    ctypes.byref(devinfo),
    SPDRP_HARDWAREID,
    None,
    ctypes.byref(szHardwareID), ctypes.sizeof(szHardwareID) - 1,
    None
):
    # Ignore ERROR_INSUFFICIENT_BUFFER
    if ctypes.GetLastError() != ERROR_INSUFFICIENT_BUFFER:
        raise ctypes.WinError()

# friendly name
szFriendlyName = ctypes.create_string_buffer(1024)
if not SetupDiGetDeviceRegistryProperty(
    g_hdi,
    ctypes.byref(devinfo),
    SPDRP_FRIENDLYNAME,
    None,
    ctypes.byref(szFriendlyName), ctypes.sizeof(szFriendlyName) - 1,
    None
):
    # Ignore ERROR_INSUFFICIENT_BUFFER
    if ctypes.GetLastError() != ERROR_INSUFFICIENT_BUFFER:
        # raise ctypes.WinError()
        # not getting friendly name for com0com devices, try something else
        szFriendlyName = ctypes.create_string_buffer(1024)
    if SetupDiGetDeviceRegistryProperty(
        g_hdi,
        ctypes.byref(devinfo),
        SPDRP_LOCATION_INFORMATION,
        None,
        ctypes.byref(szFriendlyName), ctypes.sizeof(szFriendlyName) - 1,
        None
    ):
port_name = "\\.\\" + szFriendlyName.value
order = None
else:
    port_name = szFriendlyName.value
    order = None
else:
    try:
        m = re.search(r"\((.*?([d+])\)\)\", szFriendlyName.value)
        #~ print szFriendlyName.value, m.groups()
        port_name = m.group(1)
        order = int(m.group(2))
    except AttributeError, msg:
        port_name = szFriendlyName.value
        order = None
    yield order, port_name, szFriendlyName.value, szHardwareID.value

SetupDiDestroyDeviceInfoList(g_hdi)

if __name__ == '__main__':
    import serial
    print "-"*78
    print "Serial ports"
    print "-"*78
    for order, port, desc, hwid in sorted(comports()):
        print "%-10s: \s (%s) ->\% (port, desc, hwid),
        try:
            serial.Serial(port) # test open
        except serial.serialutil.SerialException:
            print "can't be openend"
        else:
            print "Ready"
    print 
    print 
    print "All serial ports (registry)"
    print "-"*78
    for order, port, desc, hwid in sorted(comports(False)):
        print "%-10s: \s (%s)" % (port, desc, hwid)

test_read.py  download test_read.py

#!/usr/bin/python2
"""
::
    python test_read_2.py
::
Msg086
Msg087
Msg088
Msg089
Msg090
import serial

device_port = serial.Serial("/dev/serial/by-id/usb-id3_semiconductors_MEABOARD_00000000-if00", 115200, timeout=30)

while 1:
    answer = device_port.read(6)
    print(answer)
device_port.close()

wxSerialConfigDialog.py  
download wxSerialConfigDialog.py

#!/usr/bin/env python
# generated by wxGlade 0.3.1 on Thu Oct 02 23:25:44 2003

# from wxPython.wx import *
import wx
import serial

SHOW_BAUDRATE = 1<<0
SHOW_FORMAT = 1<<1
SHOW_FLOW = 1<<2
SHOW_TIMEOUT = 1<<3
SHOW_ALL = SHOW_BAUDRATE|SHOW_FORMAT|SHOW_FLOW|SHOW_TIMEOUT

try:
    enumerate
except NameError:
    def enumerate(sequence):
        return zip(range(len(sequence)), sequence)

class SerialConfigDialog(wx.Dialog):
    """Serial Port configuration dialog, to be used with pyserial 2.0+
    When instantiating a class of this dialog, then the "serial" keyword
    argument is mandatory. It is a reference to a serial.Serial instance.
    the optional "show" keyword argument can be used to show/hide different
    settings. The default is SHOW_ALL which corresponds to
    SHOW_BAUDRATE|SHOW_FORMAT|SHOW_FLOW|SHOW_TIMEOUT. All constants can be
    found in this module (not the class).""

    def __init__(self, *args, **kwds):
        # grab the serial keyword and remove it from the dict
        self.serial = kwds['serial']
        del kwds['serial']
        self.show = SHOW_ALL
        if kwds.has_key('show'):
            self.show = kwds['show']
        del kwds['show']
        # begin wxGlade: SerialConfigDialog.__init__
Development tools, Release 2012.06.18

# end wxGlade
kwds["style"] = wx.DEFAULT_DIALOG_STYLE
wx.Dialog.__init__(self, *args, **kwds)
self.label_2 = wx.StaticText(self, -1, "Port")
self.combo_box_port = wx.ComboBox(self, -1, choices=["dummy1", "dummy2", "dummy3", "dummy4"],
if self.show & SHOW_BAUDRATE:
    self.label_1 = wx.StaticText(self, -1, "Baudrate")
self.choice_baudrate = wx.Choice(self, -1, choices=["choice 1"])
if self.show & SHOW_FORMAT:
    self.label_3 = wx.StaticText(self, -1, "Data Bits")
self.choice_databits = wx.Choice(self, -1, choices=["choice 1"])
self.label_4 = wx.StaticText(self, -1, "Stop Bits")
self.choice_stopbits = wx.Choice(self, -1, choices=["choice 1"])
self.label_5 = wx.StaticText(self, -1, "Parity")
self.choice_parity = wx.Choice(self, -1, choices=["choice 1"])
if self.show & SHOW_TIMEOUT:
    self.checkbox_timeout = wx.CheckBox(self, -1, "Use Timeout")
    self.text_ctrl_timeout = wx.TextCtrl(self, -1, "")
self.label_6 = wx.StaticText(self, -1, "seconds")
if self.show & SHOW_FLOW:
    self.checkbox_rtscts = wx.CheckBox(self, -1, "RTS/CTS")
    self.checkbox_xonxoff = wx.CheckBox(self, -1, "Xon/Xoff")
self.button_ok = wx.Button(self, -1, "OK")
self.button_cancel = wx.Button(self, -1, "Cancel")
self.__set_properties()
self.__do_layout()
#fill in ports and select current setting
index = 0
for n in range(4):
    portname = serial.device(n)
    self.combo_box_port.Append(portname)
    if self.serial.portstr == portname:
        index = n
if self.serial.portstr is not None:
    self.combo_box_port.SetValue(str(self.serial.portstr))
else:
    self.combo_box_port.SetSelection(index)
if self.show & SHOW_BAUDRATE:
    #fill in badrates and select current setting
    self.choice_baudrate.Clear()
    for n, baudrate in enumerate(self.serial.BAUDRATES):
        self.choice_baudrate.Append(str(baudrate))
        if self.serial.baudrate == baudrate:
            index = n
    self.choice_baudrate.SetSelection(index)
if self.show & SHOW_FORMAT:
    #fill in databits and select current setting
    self.choice_databits.Clear()
    for n, bytesize in enumerate(self.serial.BYTESIZES):
        self.choice_databits.Append(str(bytesize))
        if self.serial.bytesize == bytesize:
            index = n
    self.choice_databits.SetSelection(index)
    #fill in stopbits and select current setting
    self.choice_stopbits.Clear()
    for n, stopbits in enumerate(self.serial.STOPBITS):
Development tools, Release 2012.06.18

self.choice_stopbits.Append(str(stopbits))
if self.serial.stopbits == stopbits:
    index = n
self.choice_stopbits.SetSelection(index)
#fill in parities and select current setting
self.choice_parity.Clear()
for n, parity in enumerate(self.serial.PARITIES):
    self.choice_parity.Append(str(serial.PARITY_NAMES[parity]))
if self.serial.parity == parity:
    index = n
self.choice_parity.SetSelection(index)

if self.show & SHOW_TIMEOUT:
    #set the timeout mode and value
    if self.serial.timeout is None:
        self.checkbox_timeout.SetValue(False)
        self.text_ctrl_timeout.Enable(False)
    else:
        self.checkbox_timeout.SetValue(True)
        self.text_ctrl_timeout.Enable(True)
        self.text_ctrl_timeout.SetValue(str(self.serial.timeout))
if self.show & SHOW_FLOW:
    #set the rtscts mode
    self.checkbox_rtscts.SetValue(self.serial.rtscts)
    #set the rtscts mode
    self.checkbox_xonxoff.SetValue(self.serial.xonxoff)

#attach the event handlers
self.__attach_events()

def __set_properties(self):
    # begin wxGlade: SerialConfigDialog.__set_properties
    self.SetTitle("Serial Port Configuration")
    # end wxGlade
    if self.show & SHOW_TIMEOUT:
        self.text_ctrl_timeout.Enable(0)
    self.button_ok.SetDefault()

def __do_layout(self):
    # begin wxGlade: SerialConfigDialog.__do_layout
    sizer_2 = wx.BoxSizer(wx.VERTICAL)
    sizer_3 = wx.BoxSizer(wx.HORIZONTAL)
    sizer_basics = wx.StaticBoxSizer(wx.StaticBox(self, -1, "Basics"), wx.VERTICAL)
    sizer_5 = wx.BoxSizer(wx.HORIZONTAL)
    sizer_8 = wx.BoxSizer(wx.HORIZONTAL)
    sizer_7 = wx.BoxSizer(wx.HORIZONTAL)
    sizer_6 = wx.BoxSizer(wx.HORIZONTAL)
    sizer_format = wx.StaticBoxSizer(wx.StaticBox(self, -1, "Data Format"), wx.VERTICAL)
    sizer_6.Add(self.label_3, 1, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)
    sizer_5.Add(self.combo_box_port, 1, 0, 0)
    sizer_basics.Add(sizer_5, 0, wx.RIGHT|wx.EXPAND, 0)
    sizer_basics.Add(sizer_8, 0, wx.RIGHT|wx.EXPAND, 0)
    if self.show & SHOW_BAUDRATE:
        sizer_baudrate = wx.BoxSizer(wx.HORIZONTAL)
        sizer_baudrate.Add(self.label_1, 1, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)
        sizer_baudrate.Add(self.choice_baudrate, 1, wx.ALIGN_RIGHT, 0)
        sizer_basics.Add(sizer_baudrate, 0, wx.EXPAND, 0)
    sizer_2.Add(sizer_basics, 0, wx.EXPAND, 0)
    if self.show & SHOW_FORMAT:
        sizer_format = wx.StaticBoxSizer(wx.StaticBox(self, -1, "Data Format"), wx.VERTICAL)
        sizer_6.Add(self.label_3, 1, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)

2.4. Communication   83
if self.show & SHOW_TIMEOUT:
    sizer_timeout = wx.StaticBoxSizer(wx.StaticBox(self, -1, "Timeout"), wx.HORIZONTAL)
    sizer_timeout.Add(self.checkbox_timeout, 0, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)
    sizer_timeout.Add(self.text_ctrl_timeout, 0, 0, 0
    sizer_timeout.Add(self.label_6, 0, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)
    sizer_2.Add(sizer_timeout, 0, 0, 0)
if self.show & SHOW_FLOW:
    sizer_flow = wx.StaticBoxSizer(wx.StaticBox(self, -1, "Flow Control"), wx.HORIZONTAL)
    sizer_flow.Add(self.checkbox_rtscts, 0, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)
    sizer_flow.Add(self.checkbox_xonxoff, 0, wx.ALL|wx.ALIGN_CENTER_VERTICAL, 4)
    sizer_flow.Add((10,10), 1, wx.EXPAND, 0)
    sizer_2.Add(sizer_flow, 0, wx.EXPAND, 0)

self.SetAutoLayout(1)
self.SetSizer(sizer_2)
sizer_2.Fit(self)
sizer_2.SetSizeHints(self)
self.Layout()

def __attach_events(self):
    wx.EVT_BUTTON(self, self.button_ok.GetId(), self.OnOK)
    wx.EVT_BUTTON(self, self.button_cancel.GetId(), self.OnCancel)
    if self.show & SHOW_TIMEOUT:
        wx.EVT_CHECKBOX(self, self.checkbox_timeout.GetId(), self.OnTimeout)

def OnOK(self, events):
    success = True
    self.serial.port = str(self.combo_box_port.GetValue())
    if self.show & SHOW_BAUDRATE:
        self.serial.baudrate = self.serial.BAUDRATES[self.choice_baudrate.GetSelection()]
    if self.show & SHOW_FORMAT:
        self.serial.bytesize = self.serial.BYTESIZES[self.choice_databits.GetSelection()]
        self.serial.stopbits = self.serial.STOPBITS[self.choice_stopbits.GetSelection()]
        self.serial.parity = self.serial.PARITIES[self.choice_parity.GetSelection()]
    if self.show & SHOW_FLOW:
        self.serial.rtscts = self.checkbox_rtscts.GetValue()
        self.serial.xonxoff = self.checkbox_xonxoff.GetValue()
    if self.show & SHOW_TIMEOUT:
        try:
            self.serial.timeout = float(self.text_ctrl_timeout.GetValue())
        except ValueError:
           dlg = wx.MessageDialog(self, 'Timeout must be a numeric value', 'Value Error', wx.OK | wx.ICON_ERROR)
            dlg.ShowModal()
            dlg.Destroy()
            success = False

Chapter 2. Development
else:
    self.serial.timeout = None
if success:
    self.EndModal(wx.ID_OK)

def OnCancel(self, events):
    self.EndModal(wx.ID_CANCEL)

def OnTimeout(self, events):
    if self.checkbox_timeout.GetValue():
        self.text_ctrl_timeout.Enable(True)
    else:
        self.text_ctrl_timeout.Enable(False)

# end of class SerialConfigDialog

class MyApp(wx.App):
    """Test code""
    def OnInit(self):
        wx.InitAllImageHandlers()

        ser = serial.Serial()
        print ser

        #loop until cancel is pressed, old values are used as start for the next run
        #show the different views, one after the other
        #value are kept.
        for flags in (SHOW_BAUDRATE, SHOW_FLOW, SHOW_FORMAT, SHOW_TIMEOUT, SHOW_ALL):
            dialog_serial_cfg = SerialConfigDialog(None, -1, "", serial=ser, show=flags)
            self.SetTopWindow(dialog_serial_cfg)
            result = dialog_serial_cfg.ShowModal()
            print ser
            if result != wx.ID_OK:
                break

        #the user can play around with the values, CANCEL aborts the loop
        while 1:
            dialog_serial_cfg = SerialConfigDialog(None, -1, "", serial=ser)
            self.SetTopWindow(dialog_serial_cfg)
            result = dialog_serial_cfg.ShowModal()
            print ser
            if result != wx.ID_OK:
                break

        return 0

# end of class MyApp

if __name__ == '__main__':
    app = MyApp(0)
    app.MainLoop()
Communication tools

**links**

```
yum install links
```

**sox**

```
yum install sox
```

**net**

```
https://code.google.com/p/netzob/
```

**netzob (NETwork protocol modeliZatiOn By reverse engineering)**

NETZOB : NETwork protocol modeliZatiOn By reverse engineering

**See Also:**

- https://code.google.com/p/netzob/
- python netzob

Netzob simplifies the work for security auditors by providing a complete framework for the reverse engineering of communication protocols. It handles different types of protocols: text protocols (like HTTP and IRC), fixed fields protocols (like IP and TCP) and variable fields protocols (like ASN.1 based formats).

Netzob is therefore suitable for reversing network protocols, structured files and system and process flows (IPC and communication with drivers).

Netzob is provided with modules dedicated to capture data in multiple contexts (network, file, process and kernel data acquisition).

**Description of the functionalities**

- Handle the following inputs as initial data:
  - PCAP
  - Network capturing (with Scapy)
  - Structured files with unknown format
  - Intra Processus communication (API calls)
  - Inter Processus Communication (pipes, shared memory and sockets)
  - Kernel Memory (with a dedicated module)
- Metadata representation of inputs
- Clustering: Regroups equivalent messages using:
  - an UPGMA Algorithm to regroup similar messages
  - an openMP and MPI implementation
- Sequencing, Alignment: Identification of fields in messages:
  - Needleman & Wunsch Implementation
• Fields dependencies identification:
  – length fields and associated payloads
  – encapsulated messages identifications

• Fields type identification:
  – Primary types: binary, ascii, num, base64...
  – Definition domain, unique elements and intervals
  – Data carving (tar.gz, png, jpg, ...)
  – Semantic data identification (emails, IP ...)

• Fuzzing:
  – Live instrumentation through a dedicated proxy
  – Possibilities of variations:
    * Data and types
    * Length
    * Fields dependencies

• Results exports:
  – XML meta representations of inferred protocol,
  – Dedicated New Wireshark Disector

2.5 Configuration software

2.5.1 Configuration

See Also:

Mozillazine configuration

Profile folder

See Also:
  * http://kb.mozillazine.org/Profile_folder

Mozilla applications store a user’s personal information in a unique profile.

The first time you start any Mozilla application, it will automatically create a default profile; additional profiles can be created using the Profile Manager.

The settings which form a profile are stored in files within a special folder on your computer — this is the profile folder.

The installation directory also includes a “profile” folder but this folder contains program defaults, not your user profile data
Profile manager

See Also:
- http://kb.mozillazine.org/Profile_Manager

Multiple profiles Firefox

http://kb.mozillazine.org/Using_multiple_profiles_-_Firefox

Qt Configuration with qsettings

See Also:

Detailed Description

The QSettings class provides persistent platform-independent application settings.

Users normally expect an application to remember its settings (window sizes and positions, options, etc.) across sessions.

This information is often stored in the system registry on Windows, and in XML preferences files on Mac OS X. On Unix systems, in the absence of a standard, many applications (including the KDE applications) use INI text files.

QSettings is an abstraction around these technologies, enabling you to save and restore application settings in a portable manner. It also supports custom storage formats.

QSettings’s API is based on QVariant, allowing you to save most value-based types, such as QString, QRect, and QImage, with the minimum of effort.

QSettings stores settings. Each setting consists of a QString that specifies the setting’s name (the key) and a QVariant that stores the data associated with the key.

To write a setting, use setValue(). For example:

```
settings.setValue("editor/wrapMargin", 68);
```

2.6 Desktop environment

2.6.1 Desktop environment (environnement de bureau)

See Also:

In graphical computing, a desktop environment (DE) commonly refers to a particular implementation of graphical user interface (GUI) derived from the desktop metaphor that is seen on most modern personal computers.

These GUIs help the user in easily accessing, configuring, and modifying many important and frequently accessed specific operating system (OS) features.
The GUI usually does not afford access to all the many features found in an OS. Instead, the traditional command-line interface (CLI) is still used when full control over the OS is required in such cases.

A desktop environment typically consists of icons, windows, toolbars, folders, wallpapers and desktop widgets (see Elements of graphical user interfaces and WIMP).

A GUI might also provide drag and drop functionality and other features that make the desktop metaphor more complete.

A desktop environment aims to be an intuitive way for the user to interact with the computer using concepts which are similar to those used when interacting with the physical world, such as buttons and windows.

While the term desktop environment originally described a style of user interfaces following the desktop metaphor, it has also come to describe the programs that realize the metaphor itself.

This usage has been popularized by the Common Desktop Environment and the K Desktop Environment.

** GNOME **

See Also:

- http://www.gnome.org/
- twitter.com/gnome
- http://identi.ca/gnome

Figure 2.1: GNOME logo

** Presentation **

GNOME (pronounced /nom/ or /nom/) is a desktop environment and graphical user interface that runs on top of a computer operating system.

It is composed entirely of free and open source software.

It is an international project that includes creating software development frameworks, selecting application software for the desktop, and working on the programs that manage application launching, file handling, and window and task management.

GNOME is part of the GNU Project and can be used with various Unix-like operating systems, most notably Linux and as part of OpenSolaris Desktop.

** Contents **

- GNOME
  - Presentation
  - Métaphore d’interface
  - Name
  - Look and feel
  - GNOME applications
  - GNOME issue tracker
  - GNOME shells
  - GNOME versions

2.6. Desktop environment 89
Métaphore d’interface

La métaphore d’interface à la base de l’expérience utilisateur a radicalement changé avec la version 3.0.
Précédemment l’environnement GNOME adoptait la métaphore du bureau en présentant à l’utilisateur un bureau virtuel sur lequel il est possible de placer des icônes de toutes sortes (représentant des lanceurs d’applications, des dossiers ou des fichiers), complété d’un tableau de bord très personnalisable permettant de lancer des programmes ou d’afficher des notifications.
À partir de la version 3.0, GNOME adopte un nouveau paradigme pour l’interface utilisateur, orchestré par GNOME Shell.
La métaphore du bureau, jugée non optimale, est abandonnée.
La nouvelle interface met l’accent sur la possibilité de basculer facilement d’une tâche à l’autre et l’utilisation de multiples espaces de travail pour gérer les tâches.

Name

Initially “GNOME” was an acronym of GNU Network Object Model Environment, referring to the original intention of creating a distributed object framework similar to Microsoft’s OLE; but it was dropped[8] because this no longer reflects the core vision of the GNOME project.

Look and feel

Up until the release of GNOME 3.x, GNOME was designed around the traditional computing desktop metaphor. Users can change the appearance of their desktop through the use of themes, which usually consist of an icon set, a window manager border and GTK+ theme engine and parameters. The current default theme is Adwaita. The Human Interface Guidelines helps developers to produce applications that look and behave similarly to each other, which provides a cohesive GNOME experience.

 GNOME has evolved from a traditional desktop metaphor to a user interface where switching between different tasks and virtual workspaces takes place in a new area called the overview.

 The redesigned GNOME experience features several main changes: released as the new interface for Gnome, GNOME Shell replaces the original GNOME Panel; Mutter replaces Metacity as the default window manager; the minimize and maximize buttons are no longer placed on the titlebar by default.

 Many of the default GNOME applications have also gone through redesigns to provide a more consistent and unified user experience.

 In the default configuration of GNOME, the desktop has a top panel holding (from left to right) an activities button, clock, system status area and user menu. Clicking on the activities button or moving one’s mouse to the top-left hot corner, brings one to the overview.

 The system status area holds various system indicators, such as those for volume, Bluetooth, network, battery, and accessibility.

 The user menu holds a chat availability indicator, shortcuts to system settings, as well as session actions such as logging out, switching users, locking the screen, or suspending the computer.

 The overview (accessed by clicking on the activities button in the top panel, or touching the top-left hot corner) shows the window picker, the workspace changer on the right, the dash on the left, a windows button, an applications button, and a search bar. While in the overview, users can click on the windows and application buttons just under the top panel to switch between the window picker and the application picker. The window picker provides a way to switch to other open windows; a convenient way to close multiple windows easily; and gives users a quick overview of
current activities. The application provides an easy way to launch applications. The dash houses shortcuts to favorite applications and open windows.

Also in the default interface are a new system for notifications.

In GNOME 3, notifications popup from the bottom of the screen, as opposed to showing in the top-right of the screen as in GNOME 2.x

**GNOME applications**

**GNOME applications**  See Also:

http://www.gnome.org/applications/

**cheese**  See Also:

- http://projects.gnome.org/cheese/
- http://projects.gnome.org/cheese/tour

Cheese uses your webcam to take photos and videos, applies fancy special effects and lets you share the fun with others.

It was written as part of Google’s 2007 Summer of Code lead by daniel g. siegel and mentored by Raphaël Slinckx.

Under the hood, Cheese uses GStreamer to apply fancy effects to photos and videos.

With Cheese it is easy to take photos of you, your friends, pets or whatever you want and share them with others.

After a success of the Summer of Code, the development continued and we still are looking for people with nice ideas and patches ;)

**cheese versions**

**cheese 3.2**

**what’s changed in 3.3.2?**  Huge changes to the API reference documentation, and new hotplug support, as well as many other updates.

- Pre-release version bump to 3.3.2
- Remove marshaller generation rules The generic marshallers, supplied by GLib, are used instead.
- Make preferences dialog more netbook-friendly Fixes bug 663146.
- Convert deprecated GtkVBox and GtkHBox to GtkBox Fixes bug 661661.
- Add chapter IDs to documentation
- Add Cheese architecture diagram to documentation Fixes bug 664333.
- Set camerabin to playing before start-capture Fixes bug 663998, with the camerabin documentation being updated in bug 664048.
- Use private GObject pointer in Widget and Chooser Additionally, remove some dead code.
- Add gtk-doc comments to UmCropArea Currently, UmCropArea is internal (and will likely stay that way), so the comments are not included in the generated documentation.
- Use switch statement to select cases
• Use better API to notify and install properties Changed g_object_notify_by_pspec() to g_object_notify() and g_object_class_install_property() to g_object_class_install_properties(). Added properties static array to hold properties. Added enum constants identifying properties and sentinels for array length definitions. Fixes bug 663098.

• Improve CheeseAvatarChooser documentation Add documentation for private methods in CheeseAvatarChooser.

• Improve CheeseEffect documentation Document the private methods in CheeseEffect. Simplify effect construction, by making the name and pipeline-desc properties construct-only. Improve some variables names.

• Improve CheeseFileUtil filename handling Use GDateTime to format the time string. Use switch statements when selecting cases from an enum. Use g_build_filename() rather than g_strjoin(G_DIR_SEPARATOR_S, ...).

• Improve CheeseCamera documentation Document most of the private methods in CheeseCamera, and rename the device-name property to device-node.

• Improve CheeseCameraDevice documentation Document most of the private methods in CheeseCameraDevice, and rename some variables to more closely match the property names.

• Improve CheeseCameraDeviceMonitor documentation

• Make CheeseFlash a GtkWindow rather than a GObject Additionally, add further gtk-doc-like comments to the flash implementation.

• Comment Vala methods with Valadoc markup Add basic documentation to all methods in Vala sources.

• Add hotplug support to preferences dialog Adding and removing camera devices in the CheeseCamera is now propagated to the preferences dialog UI. Partially fixes bug 603612.

• Correct default brightness in the schema to zero

• Improve GSettings schema text and include ranges

• Also use the countdown duration for burst mode

• Add a countdown-duration key to GSettings schema Add a new GSettings key to allow configuration of the duration of the countdown when taking a photo. Partially fixes bug 594267.

• Bump required Vala version to 0.13.2 Required for Clutter.TableLayout in clutter-1.0.vapi, which was added when Vala switched to use GIR files.

• Use STYLE_PROVIDER_PRIORITY_USER

• Use Vala ‘as’ operator where possible Use the as operator when fetching widgets from GtkBuilder files, to do a runtime type-check.

• Include the version in the man page

• Add generated man page to CLEANFILES

• Fix a typo so that the man page is generated

• Fix gtk-doc checks when srmdir != builddir

• Add man page, generated with xsltproc

• Add help button to the preferences dialog

• Improve documentation to pass gtk-doc tests Add the deprecated API index to the documentation. Complete the rename of cheese_camera_set_device_by_dev_file() to cheese_camera_set_device_by_device_node(). Add missing documentation to reach 100% symbol coverage.

• Enable gtk-doc tests during make check
• Check for GStreamer plugins required at runtime. Add a GStreamer plugin check to configure.ac, which checks for individual plugins with gst-inspect, rather than relying on the plugins being provided by checking for GStreamer pkg-config files. The check is non-fatal, as the plugins are not build-time dependencies.

• Bump pkg-config requirement to 0.24. Version 0.24 or greater of pkg-config is required in order to avoid the duplicate AC_SUBST macro calls for PKG_CHECK_MODULES substitutions.

• Remove unused FULL_LIBEXECDIR from configure.ac

• Add private pointers to libcheese GObject structs. Speed up access to the private struct of the GObject by adding a pointer, so that _GET_PRIVATE does not have to be called each time. Create a typedef for the private structures in the headers, and hide the structs from the documentation by placing them inside a private subsection.

• Add basic documentation for remaining public API. Add cheese-widget-private.h back to files ignored by gtk-doc. Add basic documentation for the remainder of the public API. Use UUID instead of ID or UDI. Add blurb and nick to all documented properties. Add CheeseVideoFormat documentation. Use unsigned integers where the values are always positive. Sprinkle some const qualifiers. Add some filename GObject Introspection annotations.

• Update TODO

• Fix several compiler warnings. Add some missing prototypes, correct some pointer type mismatches, return a value from functions that returns values and improve GError handling.

• Use gnome-common compiler warnings. Additionally, enable silent Automake rules by default so that warnings are more visible.

• Add more classes to gtk-doc documentation. Add basic documentation for CheeseCamera, CheeseFileUtil and CheeseFlash classes. Split the API reference into libcheese and libcheese-gtk chapters.

• Add CheeseCameraDeviceMonitor::removed callback. Added CheeseCameraDeviceMonitor::removed signal callback function in CheeseCamera. Fixes bug 662852.

• Improved CheeseCameraDeviceMonitor::added signal. CheeseCameraDeviceMonitor::added has now a CheeseCameraDevice argument. CheeseCameraDeviceMonitor is also now a member of CheeseCamera.

• Add a long description to the DOAP file

• help: fixed another typo

• help: moving introduction page back to .page

• help: fixed typo in Makefile.am

• help: updated Makefile.am

• Overhaul the libcheese documentation. Add documentation for CheeseCameraDevice. Remove bogus XML included in the library overview. Add section documentation to all classes, and mark them as unstable. Add GObject and GObjectClass struct documentation.

• help: renamed pages and rearranged sections. Moved all pages, except introduction.page, into sections to make index look nicer.

• Use license-type in the about dialog

• Connect thumbnail nav button signals. The thumb nav widget crashed when clicking the buttons to scroll the thumbnail view. This was caused by connecting to the wrong signals: ‘button-pressed-event’ and ‘button-released-event’, rather than ‘pressed’ and ‘released’. There was also some duplicate code for creating the left button, which led to a GtkButton being leaked when the thumb nav was created. Fixes bug 660686.

• Remove obsolete MAINTAINERS file. http://live.gnome.org/Git/FAQ#How_do_I_add_a_description_to_the_git_web_view.3F__What_is_this_.22blah.doap.22.

• Remove obsolete cicl script

• Increase the photo count in burst mode to 100000. Fixes bug 659977

2.6. Desktop environment
• Use an idle handler to generate thumbnails A thread was used for generating thumbnails for CheeseThumbView, but this had problems with concurrent access to the GtkListStore which backs the thumb view, as in bug 648936. A simpler approach is to use an idle handler, which avoids the need for acquiring the GDK lock.

• Restore gudev checks during configure

• Added/Updated Translations
  • be, courtesy of Yuri Matsuk
  • de, courtesy of Mario Blättermann
  • es, courtesy of Daniel Mustieles
  • et, courtesy of Mattias Pöldaru
  • gl, courtesy of Fran Dieguez
  • he, courtesy of Yaron Shahrabani
  • lt, courtesy of Aurimas Ėrniius
  • nb, courtesy of Kjartan Maraas
  • sv, courtesy of Daniel Nylander
  • tr, courtesy of Muhammet Kara
  • xh, courtesy of Andiswa Mvanyashe

• Added/Updated Documentation
  • es, courtesy of Daniel Mustieles

what’s changed in 3.2.0?

• configure.ac: Bump Cheese version to 3.2.0
• Added/Updated Translations
  • as, courtesy of Nilamdyuti Goswami
  • gl, courtesy of Leandro Regueiro
  • or, courtesy of Manoj Kumar Giri
• Added/Updated Documentation
  • de, courtesy of Mario Blättermann
  • es, courtesy of Daniel Mustieles
  • gl, courtesy of Leandro Regueiro

**GNOME issue tracker**

**GNOME issue_tracker**  See Also:

https://bugzilla.gnome.org/
Gnome shells

Gnome shells

Gnome shell See Also:

http://fr.wikipedia.org/wiki/GNOME_Shell

Contents

- Gnome shell
  - Présentation

Présentation  Le GNOME Shell (distribué sous licence GNU GPL) est le cœur de l’interface graphique de l’environnement de bureau GNOME à partir de sa version 3.0.

Il sert à la fois de tableau de bord, de zone de notification et de sélecteur de fenêtres.

Il est basé sur le gestionnaire de fenêtres Mutter et est écrit en C et JavaScript. Son apparence est aisément configurable en CSS.

Un système d’extensions (écrites en JavaScript et stylisées en CSS) permet de personnaliser le GNOME Shell. À compter de la version 3.2, un service Web est mis en place permettant d’activer/désactiver une sélection d’extensions (chaque pourra proposer, au moyen de la même interface, des extensions qui seront mises en ligne une fois vérifié qu’elles ne contiennent pas de code malicieux).

Le GNOME Shell est fourni avec un débogueur nommé Looking Glass.

L’intégration poussée de GNOME Shell à Mutter a créé une controverse, puisque cela implique que d’autres gestionnaires de fenêtres, tels que Compiz, ne pourront plus servir en remplacement.

GNOME shell extensions  See Also:

- https://extensions.gnome.org

Contents

- GNOME shell extensions
  - What is GNOME Shell ?
  - What are GNOME Shell Extensions ?

What is GNOME Shell ?  GNOME Shell provides core user interface functions for GNOME, like switching to windows and launching applications.

User interface elements provided by GNOME Shell include the Panel at the top of the screen, the Activities Overview, and Message Tray at the bottom of the screen.

What are GNOME Shell Extensions ?  GNOME Shell extensions are small pieces of code written by third party developers that modify the way GNOME works. (If you are familiar with Chrome Extensions or Firefox Addons, GNOME Shell extensions are similar to them.)

You can find and install GNOME Shell extensions using this website.
Since extensions are created outside of the normal GNOME design and development process, they are supported by
their authors, rather than by the GNOME community.

Some features first implemented as extensions might find their way into future versions of GNOME.

Unity See Also:


Présentation

Unity est un shell (logiciel fournissant une interface pour l’utilisateur) pour l’environnement de bureau
GNOME développé par Canonical Ltd pour son système d’exploitation Ubuntu.

Il est prévu à l’origine pour utiliser de manière plus efficace le peu d’espace écran disponible sur les netbooks en
inclusant, par exemple, un lanceur d’application vertical.

Unity is a shell interface for the GNOME desktop environment developed by Canonical Ltd for its Ubuntu operating
system.

Unity debuted in the netbook edition of Ubuntu 10.10. It is designed to make more efficient use of space given the
limited screen size of netbooks, including, for example, a vertical application switcher[3] called the launcher.

Unlike GNOME, KDE Software Compilation, Xfce, or LXDE, Unity is not a collection of applications but is designed
to use existing GTK+ programs.

Unity is part of the Ayatana project, an initiative to improve the user experience within Ubuntu. In addition to Unity,
there are Application Indicators and other projects such as MeMenu, the notification system and the application Noti-
fyOSD gathered.

 GNOME versions

 GNOME versions

 GNOME 3.2 See Also:

http://library.gnome.org/misc/release-notes/3.2/

Une évolution de GNOME 3.0

En se basant sur le retour d’expériences d’utilisateurs, de nombreuses petites modifications ont été apportées à
GNOME 3.2

pour une utilisation plus agréable. Voici quelques points forts notables:

- L’agrandissement de la zone de redimensionnement des fenêtres les rend plus faciles à redimensionner.
- L’application Paramètres système inclut maintenant des liens vers d’autres réglages en rapport, situés à d’autres
endroits. Par exemple, la section Clavier possède maintenant un lien vers celle concernant l’agencement du
clavier.
• Les barres de titre, boutons et autres contrôles prennent moins de place, facilitant l’utilisation de GNOME sur les petits écrans.

• Les notifications dans le coin inférieur droit incluent à présent un compteur. Il est ainsi plus facile de savoir, par exemple, combien de nouveaux courriels sont arrivés (sans avoir à ouvrir votre application de messagerie), ou encore combien de messages ont été ratés dans un salon de discussion particulier.

• L’effet de mise en évidence qui indique qu’une application est déjà lancée, a été rendu plus visible.

• Dans le menu utilisateur, les notifications peuvent être configurées indépendamment du statut de la discussion.

• Le commutateur d’espace de travail dans la vue d’ensemble reste déployé en conservant toute sa largeur lorsque vous utilisez plus d’un espace de travail.

• L’application associée au calendrier peut maintenant être personnalisée, au lieu de supposer que c’est Evolution.

• Le niveau d’énergie de la batterie est maintenant affiché en utilisant une jauge à barres.

• La gestion de Focus-follows-mouse a été améliorée, même si elle nécessitera encore du travail.

Merci de continuer à nous transmettre votre retour d’expérience.

KDE

See Also:

• http://kde.org/

• http://en.wikipedia.org/wiki/KDE

Figure 2.2: KDE logo

KDE (play /kedii/) is an international free software community producing an integrated set of cross-platform applications designed to run on Linux, FreeBSD, Microsoft Windows, Solaris and Mac OS X systems.

It is best known for its Plasma Desktop, a desktop environment provided as the default working environment on many Linux distributions, such as Kubuntu, Pardus and openSUSE.

The goal of the community is to provide basic desktop functions and applications for daily needs as well as tools and documentation for developers to write stand-alone applications for the system. In this regard, the KDE project serves as an umbrella project for many standalone applications and smaller projects that are based on KDE technology. These include Calligra Suite, digiKam, Rekonq, K3b and many others.

KDE software is based on Qt framework. The original GPL version of this toolkit only existed for the X11 platform, but with the release of Qt 4, LGPL versions are available for all platforms.

This allows KDE software based on Qt 4 to also be distributed to Microsoft Windows and Mac OS X.

KDE versions

KDE versions

KDE 4.7 See Also:

http://kde.org/announcements/4.7/

KDE is delighted to announce its latest set of releases, providing major updates to the KDE Plasma Workspaces, KDE Applications, and the KDE Platform that provides the foundation for KDE software.

Version 4.7 of these releases provide many new features and improved stability and performance.
Plasma Workspaces Become More Portable Thanks to KWin  See Also:
http://kde.org/announcements/4.7/plasma.php
The KDE Plasma Workspaces 4.7. The Plasma Workspaces gain from extensive work to KDE’s compositing window manager, KWin, and from the leveraging of new Qt technologies such as Qt Quick.
For full details, read the Plasma Desktop and Plasma Netbook 4.7 release announcement.

Updated KDE Applications Bring Many Exciting Features  See Also:
http://kde.org/announcements/4.7/applications.php
The KDE Applications 4.7. Many KDE applications are updated. In particular, KDE’s groupware solution, Kontact, rejoins the main KDE release cycle with all major components ported to Akonadi.
The Digikam Software Collection, KDE’s feature-rich photo management tools, come with a major new version.
For full details, read the KDE Applications 4.7 release announcement.

Improved Multimedia, Instant Messaging and Semantic Capabilities in the KDE Platform  See Also:
http://kde.org/announcements/4.7/platform.php
The KDE Development Platform 4.7.0. A wide range of KDE and third party software will be able to take advantage of extensive work in Phonon and major improvements to the semantic desktop components, with enriched APIs and improved stability.
The new KDE Telepathy framework offers integration of instant messaging directly into workspaces and applications.
Performance and stability improvements in nearly all components lead to a smoother user experience and a reduced footprint of applications using the KDE Platform 4.7.
For full details, read the KDE Platform 4.7 release announcement.

New Instant Messaging integrated directly into desktop  The KDE-Telepathy team is proud to announce the technical preview and historic first release of the new IM solution for KDE. Even though it’s still in its early stages, you can already set up all sorts of accounts, including GTalk and Facebook Chat and use them in your everyday life.
The chat interface lets you choose among many appearances with its support of Adium themes. You can also put the Presence Plasma widget right into your panel to manage your online status. As this project is not yet mature enough to be part of the big KDE family, it is packaged and released separately, alongside the other major parts of KDE.
The source code for KDE-Telepathy 0.1.0 is available on http://download.kde.org.
Installation instructions are available on http://community.kde.org.

Stability As Well As Features  In addition to the many new features described in the release announcements, KDE contributors have closed over 12,000 bug reports (including over 2,000 unique bugs in the software released today) since the last major releases of KDE software. As a result, our software is more stable than ever before.

Desktop toolkits

GTK+

See Also:
• http://en.wikipedia.org/wiki/GTK%2B
GTK+ (GIMP Toolkit) is a cross-platform widget toolkit for creating graphical user interfaces. It is licensed under the terms of the GNU LGPL, allowing both free and proprietary software to use it.

It is one of the most popular toolkits for the X Window System, along with Qt.

The name GTK+ originates from GTK; the plus was added to distinguish an enhanced version.

It was originally created for the GNU Image Manipulation Program (GIMP), a free software raster graphics editor, in 1997 by Spencer Kimball and Peter Mattis, members of eXperimental Computing Facility (XCF) at the University of California, Berkeley. It is now maintained by members of the GNOME Foundation.

**Design** GTK+ is an object-oriented widget toolkit written in the C programming language; object-orientation is achieved by using the GLib object system.

On the X11 display server, GTK+ uses Xlib to draw widgets.

Using Xlib provides flexibility and allows GTK+ to be used on platforms where the X Window System is unavailable.

While GTK+ is primarily targeted at the X Window System, other platforms are supported, including Microsoft Windows (interfaced with the Windows API), and Mac OS X (interfaced with Quartz).

**HTML5 and Wayland backends are in development**

GTK+ can be configured to change the look of the widgets drawn; this is done using different display engines. Several display engines exist which try to emulate the look of the native widgets on the platform in use.

**Where can I use it?** Everywhere! GTK+ is cross-platform and boasts an easy to use API, speeding up your development time. Take a look at the screenshots to see a number of platforms GTK+ will run.

**What languages are supported?** GTK+ is written in C but has been designed from the ground up to support a wide range of languages, not only C/C++. Using GTK+ from languages such as Perl and Python (especially in combination with the Glade GUI builder) provides an effective method of rapid application development.

**Are there any licensing restrictions?** See Also:

http://www.gtk.org/commerce.php

GTK+ is free software and part of the GNU Project. However, the licensing terms for GTK+, the GNU LGPL, allow it to be used by all developers, including those developing proprietary software, without any license fees or royalties.

Get an overview of GTK+. Understand who started it, the basic architecture and why we use the license we do.

GTK+ has been involved in many projects and some big platforms.

To get a glimpse of what people think of GTK+ and how it has been used in commercial projects, read the success stories...

**GTK+ versions**
GTK+ 3.2  See Also:


What’s new in GTK+ 3.2

- Many improvements to the CSS theming support
- Width-for-height support in many more widgets: GtkPaned, GtkMenuBar, GtkNotebook,
- New experimental Wayland and HTML5 backends
- GtkFileChooser and GtkAssistant have received face-lifts
- The GtkFontSelection has been replaced by a new family of GtkFontChooser widgets
- New widgets: GtkLockButton and GtkOverlay

For more details and lists of fixed bugs, see the announcements for the 3.1.x development releases:


2.7 Documentation

2.7.1 Documentation

Documenting with sphinx

See Also:

- https://groups.google.com/forum/#!topic/sphinx-dev
- https://bitbucket.org/birkenfeld/sphinx

Figure 2.4: Sphinx logo
sphinx doc

Documentation about sphinx

Date June 21, 2012

See Also:
- http://packages.python.org/an_example_pypi_project/sphinx.html

conf.py  See Also:

- http://sphinx.pocoo.org/latest/config.html?highlight=conf#conf

The configuration directory must contain a file named conf.py. This file (containing Python code) is called the “build configuration file” and contains all configuration needed to customize Sphinx input and output behavior.

The configuration file is executed as Python code at build time (using `execfile()`), and with the current directory set to its containing directory), and therefore can execute arbitrarily complex code.

Sphinx then reads simple names from the file’s namespace as its configuration.

pypi Example  See Also:

- http://packages.python.org/an_example_pypi_project/sphinx.html#conf-py

In your doc/source directory is now a python file called conf.py.

This is the file that controls the basics of how sphinx runs when you run a build. Here you can do this like:

- Change the version/release number by setting the `version` and `release` variables.
- Set the project name and author name.
- Setup a project logo.
- Set the default style to `sphinx` or `default`. Default is what the standard python docs use.

and much much more.

Browsing through this file will give you an understanding of the basics.

Exclude patterns
**exclude_patterns**  See Also:

http://sphinx.pocoo.org/latest/config.html?highlight=conf#confval-exclude_patterns

```python
if conf_product=='mini':
    exclude_patterns = ['interface/*.rst','dialogs/*.rst']
elif conf_product=='main':
    exclude_patterns = ['mini-indexes.rst']
```

**sphinx markup**  See Also:

http://sphinx.pocoo.org/latest/markup/index.html

**sphinx inline markup**  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html

Sphinx uses interpreted text roles to insert semantic markup into documents.

**abbr**  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=doc#role-doc

An abbreviation. If the role content contains a parenthesized explanation, it will be treated specially: it will be shown in a tool-tip in HTML, and output only once in LaTeX.

Example:

```plaintext
:abbr:'LIFO (last-in, first-out)'.
```

LIFO

**doc**  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=doc#role-doc

Link to the specified document; the document name can be specified in absolute or relative fashion.

If no explicit link text is given the link caption will be the title of the given document.

**Example 1 : local link**

For example, if the reference `command` occurs in the document `inline/index`, then the link refers to `inline/command`.

For example, if the reference `command` occurs in the document `inline/index`, then the link refers to `inline/command`.

**Example 2 : with explicit link text**

Reference is `Index principal <index>` or `Index local <../index>`

Reference is *Index principal* or *Index local*
Example 3 no explicit link text
reference is :doc:`/index` or :doc:`../index`

reference is Introduction or sphinx markup

command See Also:
http://sphinx.pocoo.org/latest/markup/inline.html?highlight=command#role-command
Example:
The name of an OS-level command, such as :command:`rm`.
The name of an OS-level command, such as rm.

download See Also:
http://sphinx.pocoo.org/latest/markup/inline.html
This role lets you link to files within your source tree that are not reST documents that can be viewed, but files that can be downloaded.
When you use this role, the referenced file is automatically marked for inclusion in the output when building (obviously, for HTML output only).
All downloadable files are put into the _downloads subdirectory of the output directory; duplicate filenames are handled.
An example:
See :download:`this example script <../example.py>`.
The given filename is usually relative to the directory the current source file is contained in, but if it absolute (starting with /), it is taken as relative to the top source directory.
The example.py file will be copied to the output directory, and a suitable link generated to it.

glossary This directive must contain a reST definition-list-like markup with terms and definitions. The definitions will then be referencable with the term role. Example:

```
.. glossary::

  environment
    A structure where information about all documents under the root is saved, and used for cross-referencing. The environment is pickled after the parsing stage, so that successive runs only need to read and parse new and changed documents.

  source directory
    The directory which, including its subdirectories, contains all source files for one Sphinx project.
```

In contrast to regular definition lists, multiple terms per entry are allowed, and inline markup is allowed in terms. You can link to all of the terms. For example:
.. glossary::

   term 1
   term 2
   Definition of both terms.

(When the glossary is sorted, the first term determines the sort order.) New in version 0.6: You can now give the glossary directive a :sorted: flag that will automatically sort the entries alphabetically. Changed in version 1.1: Now supports multiple terms and inline markup in terms.

**guilabel**  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=guilabel#role-guilabel

Labels presented as part of an interactive user interface should be marked using :guilabel:. This includes labels from text-based interfaces such as those created using curses or other text-based libraries. Any label used in the interface should be marked with this role, including:

- button labels,
- window titles,
- field names,
- menu and
- menu selection names,
- and even values in selection lists.

Changed in version 1.0: An accelerator key for the GUI label can be included using an ampersand; this will be stripped and displayed underlined in the output (example: Cancel).

To include a literal ampersand, double it.

**Example 1**

```
button :guilabel:'Start'
```

button Start

**Example 2 ampersand accelerators**  guilabel also supports ampersand accelerators just like guilabel.

```
button :guilabel:'&Start'
```

button Start

**menuselection**  Menu selections should be marked using the menuselection role. This is used to mark a complete sequence of menu selections, including selecting submenus and choosing a specific operation, or any subsequence of such a sequence.

The names of individual selections should be separated by -->.
Example 1  For example, to mark the selection Start → Programs, use this markup:

```
:menuselection:'Start --> Programs'
```

**Start → Programs**

When including a selection that includes some trailing indicator, such as the ellipsis some operating systems use to indicate that the command opens a dialog, the indicator should be omitted from the selection name.

Example 2  **ampersand accelerators**  menuselection also supports ampersand accelerators just like `guilabel`.

```
:menuselection:'Start --> &Programs'
```

**Start → Programs**

program  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html?highlight=doc#role-doc

The name of an executable program.

This may differ from the file name for the executable for some platforms.

In particular, the .exe (or other) extension should be omitted for Windows programs.

Example

```
:program:'Geany.exe'
```

Geany.exe

term (very important)  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html

Reference to a term in the glossary.

The glossary is created using the `glossary` directive containing a definition list with terms and definitions.

It does not have to be in the same file as the term markup, for example the Python docs have one global glossary in the `glossary.rst` file.

If you use a term that’s not explained in a glossary, you’ll get a warning during build.

Example:

See :term:`Sphinx`

See *Sphinx*

sphinx misc markup (very important)
index (very important)  Sphinx automatically creates index entries from all object descriptions (like functions, classes or attributes) like discussed in domains.

However, there is also explicit markup available, to make the index more comprehensive and enable index entries in documents where information is not mainly contained in information units, such as the language reference.

.. index:: <entries>
   This directive contains one or more index entries. Each entry consists of a type and a value, separated by a colon.

   For example:

   .. index::
      single: execution; context
      module: __main__
      module: sys
      triple: module; search; path

      The execution context
      -----------------------
      ...

      This directive contains five entries, which will be converted to entries in the generated index which link to the exact location of the index statement (or, in case of offline media, the corresponding page number).

      Since index directives generate cross-reference targets at their location in the source, it makes sense to put them before the thing they refer to – e.g. a heading, as in the example above.

! exclamation (important)  You can mark up “main” index entries by prefixing them with an exclamation mark. The references to “main” entries are emphasized in the generated index. For example, if two pages contain

   .. index:: Python
   and one page contains ::

   .. index:: ! Python

   then the backlink to the latter page is emphasized among the three backlinks.

   For index directives containing only "single" entries, there is a shorthand notation::<

   .. index:: BNF, grammar, syntax, notation

   This creates four index entries.

   .. versionchanged:: 1.1
      Added "see" and "seealso" types, as well as marking main entries.

pair (very important) pair:  loop; statement is a shortcut that creates two index entries, namely loop; statement and statement; loop.

Example:
.. index::
    pair: sphinx ; pair
    pair: sphinx important; contents

see  see: entry; other creates an index entry that refers from entry to other.

seealso  Like see, but inserts “see also” instead of “see”.

single  Creates a single index entry.
Can be made a subentry by separating the subentry text with a semicolon (this notation is also used below to describe what entries are created).

triple  Likewise, triple: module; search; path is a shortcut that creates three index entries, which are:
    • module; search path
    • search; path, module
    • and path; module search.

Deprecated: module, keyword, operator, object, exception, statement, builtin
module, keyword, operator, object, exception, statement, builtin These all create two index entries.
    For example, module: hashlib creates the entries module; hashlib and hashlib; module.
    (These are Python-specific and therefore deprecated.)

sphinx paragraph level markup  See Also:
http://sphinx.pocoo.org/latest/markup/para.html
These directives create short paragraphs and can be used inside information units as well as normal text:

contents (très important)  See Also:
    • http://docutils.sourceforge.net/docs/ref/rst/directives.html#table-of-contents

Table-of-contents markup  The toctree directive, which generates tables of contents of subdocuments, is described in The TOC tree.
For local tables of contents, use the standard reST contents directive.

Example 1

.. contents::
    :local:
Example 2

.. contents::
   :depth: 2

Contents

- contents (très important)
  - Table-of-contents markup
  - Example 1
  - Example 2

hlist  See Also:

http://sphinx.pocoo.org/latest/markup/para.html

These directives create short paragraphs and can be used inside information units as well as normal text:

.. hlist::

   This directive must contain a bullet list. It will transform it into a more compact list by either distributing more than one item horizontally, or reducing spacing between items, depending on the builder.

   For builders that support the horizontal distribution, there is a columns option that specifies the number of columns; it defaults to 2. Example:

   .. hlist::
      :columns: 3

      * A list of
      * short items
      * that should be
      * displayed
      * horizontally

   New in version 0.6.

versionadded  See Also:


This directive documents the version of the project which added the described feature to the library or C API. When this applies to an entire module, it should be placed at the top of the module section before any prose.

The first argument must be given and is the version in question; you can add a second argument consisting of a brief explanation of the change.

Example:

.. versionadded:: 2.5
   The *spam* parameter.

Note that there must be no blank line between the directive head and the explanation; this is to make these blocks visually continuous in the markup.
What is a Domain?

Originally, Sphinx was conceived for a single project, the documentation of the Python language. Shortly afterwards, it was made available for everyone as a documentation tool, but the documentation of Python modules remained deeply built in – the most fundamental directives, like function, were designed for Python objects. Since Sphinx has become somewhat popular, interest developed in using it for many different purposes: C/C++ projects, JavaScript, or even reStructuredText markup (like in this documentation).

While this was always possible, it is now much easier to easily support documentation of projects using different programming languages or even ones not supported by the main Sphinx distribution, by providing a domain for every such purpose.

A domain is a collection of markup (reStructuredText directives and roles) to describe and link to objects belonging together, e.g. elements of a programming language. Directive and role names in a domain have names like domain:name, e.g. py:function. Domains can also provide custom indices (like the Python Module Index).

Having domains means that there are no naming problems when one set of documentation wants to refer to e.g. C++ and Python classes. It also means that extensions that support the documentation of whole new languages are much easier to write.

sphinx C domain  See Also:

- http://sphinx.pocoo.org/latest/domains.html

The C domain (name c) is suited for documentation of C API.

```rest
.. c:function::: type name(signature)
   Describes a C function. The signature should be given as in C, e.g.:

   .. c:function::: PyObject* PyType_GenericAlloc(PyTypeObject *type, Py_ssize_t nitems)
```

This is also used to describe function-like preprocessor macros. The names of the arguments should be given so they may be used in the description.

Note that you don’t have to backslash-escape asterisks in the signature, as it is not parsed by the reST inliner.

```rest
.. c:member::: type name
   Describes a C struct member. Example signature:

   .. c:member::: PyObject* PyTypeObject.tp_bases
```

The text of the description should include the range of values allowed, how the value should be interpreted, and whether the value can be changed. References to structure members in text should use the member role.

```rest
.. c:macro::: name
   Describes a “simple” C macro. Simple macros are macros which are used for code expansion, but which do not take arguments so cannot be described as functions. This is not to be used for simple constant definitions. Examples of its use in the Python documentation include PyObject_HEAD and Py_BEGIN_ALLOW_THREADS.

.. c:type::: name
   Describes a C type (whether defined by a typedef or struct). The signature should just be the type name.

.. c:var::: type name
   Describes a global C variable. The signature should include the type, such as:

   .. c:var::: PyObject* PyClass_Type
```
Cross-referencing C constructs The following roles create cross-references to C-language constructs if they are defined in the documentation:

:c:data:
Reference a C-language variable.

:c:func:
Reference a C-language function. Should include trailing parentheses.

:c:macro:
Reference a “simple” C macro, as defined above.

:c:type:
Reference a C-language type.

sphinx C++ domain See Also:

- http://sphinx.pocoo.org/latest/domains.html

The C++ Domain The C++ domain (name cpp) supports documenting C++ projects.

The following directives are available:

.. cpp:class:: signatures
.. cpp:function:: signatures
.. cpp:member:: signatures
.. cpp:type:: signatures

Describe a C++ object. Full signature specification is supported – give the signature as you would in the declaration. Here some examples:

.. cpp:function:: bool namespaced::theclass::method(int arg1, std::string arg2)

    Describes a method with parameters and types.

.. cpp:function:: bool namespaced::theclass::method(arg1, arg2)

    Describes a method without types.

.. cpp:function:: const T &array<T>::operator[](const

    Describes the constant indexing operator of a templated array.

.. cpp:function:: operator bool() const

    Describe a casting operator here.

.. cpp:function:: constexpr void foo(std::string &bar[2]) noexcept

    Describe a constexpr function here.

.. cpp:member:: std::string theclass::name

.. cpp:member:: std::string theclass::name[N][M]

.. cpp:type:: theclass::const_iterator

Will be rendered like this:
bool namespaced::theclass::method (int arg1, std::string arg2)
    Describes a method with parameters and types.

bool namespaced::theclass::method (arg1, arg2)
    Describes a method without types.

const T& array<T>::operator[] () const
    Describes the constant indexing operator of a templated array.

operator bool () const
    Describe a casting operator here.

constexpr void foo (std::string& bar[2]) noexcept
    Describe a constexpr function here.

std::string theclass::name

std::string theclass::name[N][M]

type theclass::const_iterator

.. cpp:namespace:: namespace
    Select the current C++ namespace for the following objects.

These roles link to the given object types:

:cpp:class:
    Reference a C++ object. You can give the full signature (and need to, for overloaded functions.)

Note: Sphinx’ syntax to give references a custom title can interfere with linking to template classes, if nothing
follows the closing angle bracket, i.e. if the link looks like this: :cpp:class: ’MyClass<T>‘. This is
interpreted as a link to T with a title of MyClass. In this case, please escape the opening angle bracket with a
backslash, like this: :cpp:class: ’MyClass\<T>‘.

Note on References

It is currently impossible to link to a specific version of an overloaded method. Currently the C++ domain is the first
domain that has basic support for overloaded methods and until there is more data for comparison we don’t want to
select a bad syntax to reference a specific overload. Currently Sphinx will link to the first overloaded version of the
method / function.

reStructuredText Primer  This section is a brief introduction to reStructuredText (reST) concepts and syntax, in-
tended to provide authors with enough information to author documents productively. Since reST was designed to be
a simple, unobtrusive markup language, this will not take too long.

See Also:

- http://docutils.sourceforge.net/rst.html
Hi,

I just discovered a useful thing that is not documented (at least not where I would expect it to be at http://sphinx.pocoo.org/markup/toctree.html)

A `toctree` with `:glob:` flag option not only supports `*` but also the `?` wildcard character.

I use this to have the correct sort order in an automatic index for pages are named using simple numbers, starting from 1.rst. When I have more than 9 files (12 for example), then a simple `*` would yield a toctree sorted 1, 11, 12, 2, 3, 4. But if I use the following construct:

```rst
.. toctree::
   :maxdepth: 1
   :glob:

   ?
   ??
```

then I get the expected order 1, 2, 3, 4, 11, 12.

In short: Sphinx is great :-)
Use:

.. image:: ../images/wiki_logo_openalea.png

to put an image

Note: As mentioned earlier, a directive may have options put between two columns:

.. image:: ../images/wiki_logo_openalea.png
   :width: 200px
   :align: center
   :height: 100px
   :alt: alternate text

figure

.. figure:: ../images/wiki_logo_openalea.png
   :width: 200px
   :align: center
   :height: 100px
   :alt: alternate text
   :figclass: align-center

   figure are like images but with a caption
   and whatever else you wish to add

   .. code-block:: python

      import image

gives

Geoserver example  See Also:

- http://docs.geoserver.org/trunk/en/docguide/sphinx.html#images
Add images to your documentation when possible. Images, such as screenshots, are a very helpful way of making documentation understandable. When making screenshots, try to crop out unnecessary content (browser window, desktop, etc).

Avoid scaling the images, as the Sphinx theme automatically resizes large images.

It is also helpful to include a caption underneath the image.

```
.. figure:: pagelogo_geoserver.png
   :align: center
   *The GeoServer logo as shown on the homepage.*
```

In this example, the image file exists in the same directory as the source page. If this is not the case, you can insert path information in the above command.

**Substitutions**

Substitutions syntax is
The `biohazard` symbol must be used on containers used to dispose of medical waste.

Or if you want to do a literal text replacement use:

```
.. |doctest| replace:: :mod:`doctest`
```

I really like `doctest`.

Which renders like this:

The ⚯ symbol must be used on containers used to dispose of medical waste.

I really like doctest.

**Note:** Substitutions are really useful, especially when put into a `global.rst` and included at the top of every file. See *Includes* for more.

### Lists and Quotes

List markup is natural: just place an asterisk at the start of a paragraph and indent properly. The same goes for numbered lists; they can also be autonumbered using a `#` sign:

- This is a bulleted list.
  - It has two items, the second item uses two lines.

1. This is a numbered list.
2. It has two items too.

#. This is a numbered list.
#. It has two items too.

Note that Sphinx disables the use of enumerated lists introduced by alphabetic or roman numerals, such as

A. First item
B. Second item

Nested lists are possible, but be aware that they must be separated from the parent list items by blank lines:

- this is
  - a list
    - with a nested list
      - and some subitems
    - and here the parent list continues

Definition lists are created as follows:

```
term (up to a line of text)
    Definition of the term, which must be indented
```

and can even consist of multiple paragraphs
Paragraphs are quoted by just indenting them more than the surrounding paragraphs.

**Source Code**  Literal code blocks are introduced by ending a paragraph with the special marker `::`. The literal block must be indented, to be able to include blank lines:

This is a normal text paragraph. The next paragraph is a code sample: ::

```
It is not processed in any way, except
that the indentation is removed.
```

```
It can span multiple lines.
```

This is a normal text paragraph again.

The handling of the `::` marker is smart:

- If it occurs as a paragraph of its own, that paragraph is completely left out of the document.
- If it is preceded by whitespace, the marker is removed.
- If it is preceded by non-whitespace, the marker is replaced by a single colon.

That way, the second sentence in the above example’s first paragraph would be rendered as “The next paragraph is a code sample::”.

**include**  See Also:

- [http://packages.python.org/an_example_pypi_project/sphinx.html#includes](http://packages.python.org/an_example_pypi_project/sphinx.html#includes)

The syntax:

```
.. include:: myfile.rst
```

Will ‘inline’ the given file. A common convention I use is create a global .rst file called `global.rst` and include that at the top of every page. Very useful for links to common images or common files links, etc.

**code-block (pygments, highlight)**  See Also:

- [http://sphinx.pocoo.org/latest/markup/code.html#index-0](http://sphinx.pocoo.org/latest/markup/code.html#index-0)
- [http://pygments.org/docs/lexers/](http://pygments.org/docs/lexers/)

Sphinx does syntax highlighting using the Pygments library.

For documents that have to show snippets in different languages, there’s also a code-block directive that is given the highlighting language directly:

```
.. code-block:: python
```

```
Some python code.
```

You can specify different highlighting for a code block using the following syntax:
Default highlighter

With two colons you start a code block using the default highlighter:

```python
# Some Python code here
# The language defaults to Python, we don’t need to set it
if 1 == 2:
    pass
```

With two colons you start a code block using the default highlighter:

```python
# Some Python code here
# The language defaults to Python, we don’t need to set it
if 1 == 2:
    pass
```

**python highlighter**  You can specify the language used for syntax highlighting by using code-block:

```python
.. code-block:: python

    if "foo" == "bar":
        # This is Python code
        pass

    if "foo" == "bar":
        # This is Python code
        pass
```

**xml highlighter**  For example, to specify XML:

```xml
.. code-block:: xml

    <somesnippet>Some XML</somesnippet>

<somesnippet>Some XML</somesnippet>
```

**console highlighter**  ... or UNIX shell:

```console
.. code-block:: console

    # A comment
    sh myscript.sh

    # A comment
    sh myscript.sh
```
ini highlighter  ... or a buildout.cfg:

.. code-block:: ini

[some-part]
# A random part in the buildout
recipe = collective.recipe.foo
option = value

[some-part]
# A random part in the buildout
recipe = collective.recipe.foo
option = value

pycon python console highlighter  ... or interactive Python:

.. code-block:: pycon

>>> class Foo:
...    bar = 100
...
>>> f = Foo()
>>> f.bar
100
>>> f.bar / 0
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero

>>> class Foo:
...    bar = 100
...
>>> f = Foo()
>>> f.bar
100
>>> f.bar / 0
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
ZeroDivisionError: integer division or modulo by zero

highlighting mode for the whole document  Setting the highlighting mode for the whole document:

.. highlight:: console

All code blocks in this doc use console highlighting by default:

some shell commands

If syntax highlighting is not enabled for your code block, you probably have a syntax error and Pygments will fail silently.

The full list of lexers and associated short names is here: http://pygments.org/docs/lexers/
rest Tables

**Grid tables**  Two forms of tables are supported. For *grid tables* you have to “paint” the cell grid yourself. They look like this:

```
<table>
<thead>
<tr>
<th>Header row, column 1</th>
<th>Header 2</th>
<th>Header 3</th>
<th>Header 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(header rows optional)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>body row 1, column 1</td>
<td>column 2</td>
<td>column 3</td>
<td>column 4</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>body row 2</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
```

**Simple tables**  *Simple tables* are easier to write, but limited: they must contain more than one row, and the first column cannot contain multiple lines. They look like this:

```
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A and B</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
```

**list-table**  See Also:

- [http://docs.geoserver.org/trunk/en/docguide/sphinx.html#list-tables](http://docs.geoserver.org/trunk/en/docguide/sphinx.html#list-tables)

Bulleted lists can sometimes be cumbersome and hard to follow.

When dealing with a long list of items, use list-tables.

For example, to talk about a list of options, create a table that looks like this:

```
<table>
<thead>
<tr>
<th>Shapes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Square</td>
<td>Four sides of equal length, 90 degree angles</td>
</tr>
<tr>
<td>Rectangle</td>
<td>Four sides, 90 degree angles</td>
</tr>
</tbody>
</table>
```

This is done with the following code:

```
.. list-table::
   :widths: 20 80
   :header-rows: 1
   *
   - Shapes
     - Description
   *
   - Square
     - Four sides of equal length, 90 degree angles
   *
   - Rectangle
     - Four sides, 90 degree angles
```

**sphinx Hyperlinks**
External links  Use ‘Link text <http://target>\_’ for inline web links. If the link text should be the web address, you don’t need special markup at all, the parser finds links and mail addresses in ordinary text.

Example of external link: ‘reST role <http://sphinx.pocoo.org/latest/markup/inline.html#role-ref>\_’

Example of external link: reST role .

Internal links  See Also:

http://sphinx.pocoo.org/latest/markup/inline.html#role-ref

Internal linking is done via a special reST role .

:ref:’link to internal links \<internal_links>’

link to internal links

Sphinx Sections  Section headers are created by underlining (and optionally overlining) the section title with a punctuation character, at least as long as the text:

=============
This is a heading
=============

Normally, there are no heading levels assigned to certain characters as the structure is determined from the succession of headings. However, for the Python documentation, we use this convention:

• # with overline, for parts
• * with overline, for chapters
• =, for sections
• –, for subsections
• ^, for subsubsections
• ”, for paragraphs

Explicit Markup  “Explicit markup” is used in reST for most constructs that need special handling, such as footnotes, specially-highlighted paragraphs, comments, and generic directives.

An explicit markup block begins with a line starting with .. followed by whitespace and is terminated by the next paragraph at the same level of indentation. (There needs to be a blank line between explicit markup and normal paragraphs. This may all sound a bit complicated, but it is intuitive enough when you write it.)

Directives  See Also:

http://sphinx.pocoo.org/latest/domains.html#directive-rst:role

A directive is a generic block of explicit markup. Besides roles, it is one of the extension mechanisms of reST, and Sphinx makes heavy use of it.

Basically, a directive consists of a name, arguments, options and content. (Keep this terminology in mind, it is used in the next chapter describing custom directives.) Looking at this example,
.. function:: foo(x)
    foo(y, z)

    :bar: no

    Return a line of text input from the user.

function is the directive name. It is given two arguments here, the remainder of the first line and the second line, as well as one option bar (as you can see, options are given in the lines immediately following the arguments and indicated by the colons).

The directive content follows after a blank line and is indented relative to the directive start.

Footnotes  For footnotes, use [^]_ to mark the footnote location, and add the footnote body at the bottom of the document after a “Footnotes” rubric heading, like so:

Lorem ipsum [^]_ dolor sit amet ... [^]_

.. rubric:: Footnotes

.. [^] Text of the first footnote.
.. [^] Text of the second footnote.

You can also explicitly number the footnotes for better context.

Comments  Every explicit markup block which isn’t a valid markup construct (like the footnotes above) is regarded as a comment.

Source encoding  Since the easiest way to include special characters like em dashes or copyright signs in reST is to directly write them as Unicode characters, one has to specify an encoding:

All Python documentation source files must be in UTF-8 encoding, and the HTML documents written from them will be in that encoding as well.

Gotchas  There are some problems one commonly runs into while authoring reST documents:

• Separation of inline markup: As said above, inline markup spans must be separated from the surrounding text by non-word characters, you have to use an escaped space to get around that.

Explicit Markup  “Explicit markup” is used in reST for most constructs that need special handling, such as footnotes, specially-highlighted paragraphs, comments, and generic directives.

An explicit markup block begins with a line starting with .. followed by whitespace and is terminated by the next paragraph at the same level of indentation. (There needs to be a blank line between explicit markup and normal paragraphs. This may all sound a bit complicated, but it is intuitive enough when you write it.)

Sphinx Directives  See Also:

• http://sphinx.pocoo.org/latest/domains.html#directive-rst:role
• http://docutils.sourceforge.net/docs/ref/rst/directives.html#images

2.7. Documentation
A directive is a generic block of explicit markup. Besides roles, it is one of the extension mechanisms of reST, and Sphinx makes heavy use of it.

Basically, a directive consists of a name, arguments, options and content. (Keep this terminology in mind, it is used in the next chapter describing custom directives.) Looking at this example,

```restructuredtext
.. function:: foo(x)
   foo(y, z)
   :bar: no

   Return a line of text input from the user.
```

function is the directive name. It is given two arguments here, the remainder of the first line and the second line, as well as one option bar (as you can see, options are given in the lines immediately following the arguments and indicated by the colons).

The directive content follows after a blank line and is indented relative to the directive start.

**Sphinx Footnotes** For footnotes, use [#]_ to mark the footnote location, and add the footnote body at the bottom of the document after a “Footnotes” rubric heading, like so:

```restructuredtext
Lorem ipsum [#]_ dolor sit amet ... [#]_
```

```restructuredtext
.. rubric:: Footnotes

.. [#] Text of the first footnote.
.. [#] Text of the second footnote.
```

You can also explicitly number the footnotes for better context.

**Comments** Every explicit markup block which isn’t a valid markup construct (like the footnotes above) is regarded as a comment.

**Source encoding** Since the easiest way to include special characters like em dashes or copyright signs in reST is to directly write them as Unicode characters, one has to specify an encoding:

All Python documentation source files must be in UTF-8 encoding, and the HTML documents written from them will be in that encoding as well.

**sidebar** See Also:

t is possible to create sibar

---

**Sidebar Title**

**Optional Sidebar Subtitle**

Subsequent indented lines comprise the body of the sidebar, and are interpreted as body elements.

using the following code:
.. sidebar:: Sidebar Title
   :subtitle: Optional Sidebar Subtitle

   Subsequent indented lines comprise
   the body of the sidebar, and are
   interpreted as body elements.

Note: sidebar appears as floating box and may not appear nicely.

Gotchas  There are some problems one commonly runs into while authoring reST documents:

  - **Separation of inline markup:** As said above, inline markup spans must be separated from the surrounding text
    by non-word characters, you have to use an escaped space to get around that.

sphinx code

autodoc  See Also:

- http://packages.python.org/an_example_pypi_project/sphinx.html#auto-directives

literalinclude  See Also:

http://sphinx.pocoo.org/markup/code.html

.. literalinclude:: filename
   :language: ruby
   :linenos:

   Longer displays of verbatim text may be included by storing the example text in an external file containing only
   plain text. The file may be included using the literalinclude directive. For example, to include the Python
   source file example.py, use:

   .. literalinclude:: example.py

   The file name is usually relative to the current file’s path. However, if it is absolute (starting with `/`), it is relative
   to the top source directory.

   Tabs in the input are expanded if you give a `tab-width` option with the desired tab width.

   The directive also supports the `linenos` flag option to switch on line numbers, and a `language` option to
   select a language different from the current file’s standard language. Example with options:

   .. literalinclude:: example.rb
      :language: ruby
      :linenos:

Include files are assumed to be encoded in the source_encoding. If the file has a different encoding, you can
specify it with the `encoding` option:

   .. literalinclude:: example.py
      :encoding: latin-1

The directive also supports including only parts of the file. If it is a Python module, you can select a class,
function or method to include using the `pyobject` option:
I’m wondering if people have suggestions on the best way to format function/method docstrings so it’s possible to get Sphinx’s fancy formatting and yet retain nice and readable docstrings from the Python prompt. I’m especially interested in how to document the function/method input “parameters”.

I know about the :param name: stanza but when there’s a relatively long list of parameters, I don’t find it very readable from the Python prompt. Of course, once processed by Sphinx, it yields great looking documentation.

I also know about http://packages.python.org/an_example_pypi_project/sphinx.html and the googley and sphinxey variants. I’m thinking there must be a good compromise here.

For return values, I often use the following in my docstrings:

    returns
    out1  something
    out2  something else

Indentation and proper alignment make this easy to read and it looks great once processed by Sphinx. But I can’t find a similar syntax for parameters and keywords. I thought (and obviously I’m wrong) that Sphinx accepted :parameters: and :keywords: but those don’t look nice once processed, e.g., to html. In particular, :parameters: is converted to “Parameters:” (with a space), which often causes the colon to end on a new line below the word “Parameters”.

It would be great to be able to write:

    parameters
    in1  description of in1
    in2  description of in2

    keywords
    kw1  description of kw1

just the same way we can use :returns:. I’m happy to try and add this if it sounds like a good idea. I’d like to hear what people think anyways.

Thanks.
The CR_S_SUCCESS value is returned when a command is successful.

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>French Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00000000L</td>
<td>COMMAND_SUCCESS</td>
<td>Opération réussie.</td>
</tr>
</tbody>
</table>

**List of error codes**

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>French Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x8130F01L</td>
<td>CR_E_FAILED</td>
<td>Une vérification de cohérence interne a échoué.</td>
</tr>
<tr>
<td>0x8130F02L</td>
<td>CR_E_TIMEOUT</td>
<td>Le délai a expiré.</td>
</tr>
<tr>
<td>0x8130F03L</td>
<td>CR_E_SEND_DATA_FAILED</td>
<td>L’envoi des données a échoué.</td>
</tr>
<tr>
<td>0x8130F04L</td>
<td>CR_E_OPEN_PORT_FAILED</td>
<td>L’ouverture du port COM a échoué.</td>
</tr>
<tr>
<td>0x8130F05L</td>
<td>CR_E_CLOSE_PORT_FAILED</td>
<td>La fermeture du port COM a échoué.</td>
</tr>
<tr>
<td>0x8130F06L</td>
<td>CR_E_BAD_SYNCHRO</td>
<td>Erreur de synchronisation avec la base.</td>
</tr>
<tr>
<td>0x8130F07L</td>
<td>CR_E_BAD_ADDRESS</td>
<td>Mauvaise adresse.</td>
</tr>
<tr>
<td>0x8130F08L</td>
<td>CR_E_BAD_SIZE</td>
<td>Taille incorrecte.</td>
</tr>
<tr>
<td>0x8130F09L</td>
<td>CR_E_BAD_CHANNEL</td>
<td>Mauvais canal.</td>
</tr>
<tr>
<td>0x8130F0AL</td>
<td>CR_E_BAD_STATUS</td>
<td>Mauvais statut retourné par la base.</td>
</tr>
<tr>
<td>0x8130F0BL</td>
<td>CR_E_OPEN_FILE_FAILED</td>
<td>Ouverture du fichier a échoué.</td>
</tr>
</tbody>
</table>

.. c:function:: DWORD CR_RFID_VerifierPIN(ST_CODE_PIN *pCodePIN)

Le code PIN permet d’autoriser l’utilisation des clés de chiffrement et de signature dans le lecteur RFID.

:Paramètres:
   :entree:
      :pCodePIN: le code PIN à présenter au lecteur RFID.

:Returns:
   :error: :ref:`see the error codes <list_error_codes>`
   :success: :ref:`COMMAND_SUCCESS <cr_success>`

**CR_RFID_VerifierPIN**

DWORD CR_RFID_VerifierPIN (ST_CODE_PIN *pCodePIN)

Le code PIN permet d’autoriser l’utilisation des clés de chiffrement et de signature dans le lecteur RFID.

**Paramètres**

   entree

   pCodePIN le code PIN à présenter au lecteur RFID.

**Returns**

   error :ref:`see the error codes`

   success :ref:`COMMAND_SUCCESS`

**sphinx installation**

Installation d’un projet sphinx  See Also:


2.7. Documentation 125
Installation d’une documentation Sphinx ‘reStructuredText’ sous Windows   See Also:

• *Open a windows console with the stex extension*


1. **ouvrir une fenêtre de commande**

2. taper les commandes suivantes
   
   > cd meta
   
   > sphinx-quickstart
Production de la documentation html

1. taper la commande suivante

   > make html
This is a tutorial introduction to quickly get you up and running with your own sphinx documentation system. We’ll cover installing sphinx, customizing the look and feel, using custom extensions for embedding plots, inheritance diagrams, syntax highlighted ipython sessions and more. If you follow along the tutorial, you’ll start with nothing and end up with this site – it’s the bootstrapping documentation tutorial that writes itself!

**Openalea**  See Also:
Plone  See Also:
  • https://github.com/collective/collective.developermanual

Github Fork and Edit button  See Also:
  • Github Fork and Edit button
  • https://github.com/blog/844-forking-with-the-edit-button
  • https://confluence.atlassian.com/display/BITBUCKET/Fork+a+Repo,+Compare+Code,+and+Create+a+Pull+Request
You can commit file edits through GitHub web interface using Fork and Edit button Alternative, clone the repository
using git, perform changes and push them back.
Plone collective GitHub repository has open-for-all contribution access. If you want to contribute changes without
asking the maintainers to merge them, please add your GitHub username to your profile on plone.org and request
access here.

Geoserver  See Also:
http://docs.geoserver.org/trunk/en/docguide/sphinx.html

Documentation style guide sphinx  See Also:
https://github.com/benoitbryon/documentation-style-guide-sphinx

Documentation style guide mercurial  See Also:
http://mercurial.selenic.com/wiki/HelpStyleGuide#Sections

Tools for sphinx

Tools for building sphinx

Qt program  See Also:
  • https://gist.github.com/1672347
  • http://pymolurus.blogspot.com/2012/01/documentation-viewer-for-sphinx.html
Eric Holscher, one of the creators of Read The Docs, recently posted about the importance of a documentation culture
in Open Source development, and about things that could be done to encourage this.
He makes some good points, and Read The Docs is a very nice looking showcase for documentation.
Writing good documentation is difficult enough at the best of times, and one practical problem that I face when working
on Sphinx documentation is that I often feel I have to break away from composing it to building it, to see how it looks
- because the look of it on the page will determine how I want to refine it.
What I’ve tended to do is work iteratively by making some changes to the ReST sources, invoking make html and
refreshing the browser to show how the built documentation looks.
This is OK, but does break the flow more than a little (for me, anyway, but I can’t believe I’m the only one).
I had the idea that it would be nice to streamline the process somewhat, so that all I would need to do is to save the
changed ReST source – the building and browser refresh would be automatically done, and if I had the editor and
I could achieve a sort of WYSIWYG effect with the changes appearing in the browser a second or two after I saved any changes.

I decided to experiment with this idea, and needed a browser which I could easily control (to get it to refresh on-demand). I decided to use Roberto Alsina’s 128-line browser, which is based on QtWebKit and PyQt.

The code (MIT licensed) is available from here. As it’s a single file standalone script, I haven’t considered putting it on PyPI – it’s probably easier to download it to a SHOME/bin or similar location, then you can invoke it in the docs directory of your project, run your editor, position the browser and editor windows suitably, and you’re ready to go!

```python
#!/usr/bin/env python
#
# Copyright (C) 2012 Vinay Sajip. Licensed under the MIT license.
#
# Based on Roberto Alsina’s 128-line web browser, see
#
#
import json
import os
import subprocess
import sys
import tempfile
from urllib import pathname2url

import sip
sip.setapi("QString", 2)
sip.setapi("QVariant", 2)

from PyQt4.QtGui import QtGui,QtCore,QtWebKit, QtNetwork

settings = QtCore.QSettings("Vinay Sajip", "DocWatch")

class Watcher(QtCore.QThread):
    ""
    A watcher which looks for source file changes, builds the documentation,
    and notifies the browser to refresh its contents
    ""
    def run(self):
        self._stop = False
        watch_command = ‘inotifywait -rq -e close_write --exclude ‘\’*.html\’ ‘.split()
        make_command = ‘make html’.split()
        while not self._stop:
            # Perhaps should put notifier access in a mutex - not bothering yet
            self.notifier = subprocess.Popen(watch_command)
            self.notifier.wait()
            if self._stop:
                break
            subprocess.call(make_command)
            # Refresh the UI ...
            self.parent().changed.emit()

    def stop(self):
        self._stop = True
        # Perhaps should put notifier access in a mutex - not bothering for now
        if self.notifier.poll() is None:  # not yet terminated ...
            self.notifier.terminate()
```

Chapter 2. Development
class MainWindow(QtGui.QMainWindow):
    
    A browser intended for viewing HTML documentation generated by Sphinx.
    
    changed = QtCore.pyqtSignal()

    def __init__(self, url):
        QtGui.QMainWindow.__init__(self)
        self.sb = self.statusBar()
        self.pbar = QtGui.QProgressBar()
        self.pbar.setMaximumWidth(120)
        self.wb = QtWebKit.QWebView(loadProgress = self.pbar.setValue, loadFinished = self.pbar.hide, loadStarted = self.pbar.show, titleChanged = self.setWindowTitle)
        self.setCentralWidget(self.wb)
        self.tb = self.addToolBar("Main Toolbar")
        for a in (QtWebKit.QWebPage.Back, QtWebKit.QWebPage.Forward, QtWebKit.QWebPage.Reload):
            self.tb.addAction(self.wb.pageAction(a))

        self.url = QtGui.QLineEdit(returnPressed = lambda: self.wb.setUrl(QtCore.QUrl.fromUserInput(self.url.text())))
        self.tb.addWidget(self.url)

        self.wb.urlChanged.connect(lambda u: self.url.setText(u.toString()))
        self.wb.urlChanged.connect(lambda: self.url.setCompleter(QtGui.QCompleter(QtCore.QStringList([QtCore.QString(i.url().toString()) for i in self.wb.history().items()]), caseSensitivity = QtCore.Qt.CaseInsensitive)))
        self.wb.statusBarMessage.connect(self.sb.showMessage)
        self.wb.page().linkHovered.connect(lambda l: self.sb.showMessage(l, 3000))

        self.search = QtGui.QLineEdit(returnPressed = lambda: self.wb.findText(self.search.text()))
        self.search.hide()
        self.showSearch = QtGui.QShortcut("Ctrl+F", self, activated = lambda: (self.search.show(), self.search.setFocus()))
        self.hideSearch = QtGui.QShortcut("Esc", self, activated = lambda: (self.search.hide(), self.wb.setFocus()))
        self.quit = QtGui.QShortcut("Ctrl+Q", self, activated = self.close)
        self.zoomIn = QtGui.QShortcut("Ctrl++", self, activated = lambda: self.wb.setZoomFactor(self.wb.zoomFactor()+.2))
        self.zoomOut = QtGui.QShortcut("Ctrl+-", self, activated = lambda: self.wb.setZoomFactor(self.wb.zoomFactor()-.2))
        self.zoomOne = QtGui.QShortcut("Ctrl+="., self, activated = lambda: self.wb.setZoomFactor(1))

        self.wb.settings().setAttribute(QtWebKit.QWebSettings.PluginsEnabled, True)
        self.sb.addPermanentWidget(self.search)
        self.sb.addPermanentWidget(self.pbar)

        self.load_settings()

        self.wb.load(url)
        self.watcher = Watcher(self)

        self.changed.connect(self.wb.reload)
        self.watcher.start()

    def load_settings(self):
        settings.beginGroup('mainwindow')
        pos = settings.value('pos')
        size = settings.value('size')
        if isinstance(pos, QtCore.QPoint):
            self.move(pos)
        if isinstance(size, QtCore.QSize):

        self.load_settings()
Development tools, Release 2012.06.18

```python
self.resize(size)
settings.endGroup()

def save_settings(self):
    settings.beginGroup('mainwindow')
    settings.setValue('pos', self.pos())
    settings.setValue('size', self.size())
    settings.endGroup()

def closeEvent(self, event):
    self.save_settings()
    self.watcher.stop()

if __name__ == '__main__':
    if not os.path.isdir('_build'):
        # very simplistic sanity check. Works for me, as I generally use
        # sphinx-quickstart defaults
        print('You must run this application from a Sphinx directory containing _build')
        rc = 1
    else:
        app = QtGui.QApplication(sys.argv)
        path = os.path.join('_build', 'html', 'index.html')
        url = 'file:///{}html/index.html'.format(os.path.abspath(path))
        url = QtCore.QUrl(url)
        wb = MainWindow(url)
        wb.show()
        rc = app.exec_()
        sys.exit(rc)
```

**Ironpython**

Update: Another advantage of using the subprocess / command line approach to notification is that it’s easy to slot in a solution for a platform which doesn’t support inotify.

Alternatives are available for both Windows and Mac OS X. For example, on Windows, if you have IronPython installed, the following script could be used to provide the equivalent functionality to inotifywait (for this specific application):

```python
import clr
import os

from System.IO import FileSystemWatcher, NotifyFilters

stop = False

def on_change(source, e):
    global stop
    if not e.Name.endswith('.html'):
        stop = True
        print('file: %s, stop = %s' % (e.FullPath, e.ChangeType, stop))

watcher = FileSystemWatcher(os.getcwd())
watcher.NotifyFilter = NotifyFilters.LastWrite | NotifyFilters.FileName
watcher.EnableRaisingEvents = True
watcher.IncludeSubdirectories = True
watcher.Changed += on_change
watcher.Created += on_change
```
while not stop:
    pass

**Mac OS X**  Whereas for Mac OS X, if you install the MacFSEvents package, the following script could be used to provide the equivalent functionality to inotifywait (again, for this specific application):

```python
#!/usr/bin/env python
import os
from fsevents import Observer, Stream

stop = False

def on_change(e):
    global stop
    path = e.name
    if os.path.isfile(path):
        if not path.endswith('.html'):
            stop = True
        print('%s: %s, stop = %s' % (e.name, e.mask, stop))

observer = Observer()
observer.start()
stream = Stream(on_change, os.getcwd(), file_events=True)
observer.schedule(stream)
try:
    while not stop:
        pass
finally:
    observer.unschedule(stream)
    observer.stop()
    observer.join()
```

**Automatically-build-sphinx-documentation**  See Also:


```bash
#!/bin/bash
## Automatically build Sphinx documentation upon file change
## Copyright (c) 2011 Samuele ~redShadow~ Santi - Under GPL

WORKDIR="$( dirname "$0" )"
while ;; do
    ## Wait for changes
    inotifywait -e modify,create,delete -r "$WORKDIR"
    ## Make html documentation
    make -C "$WORKDIR" html
done
```

**sphinx extensions**

**Sphinx extensions**  See Also:
• http://sphinx.pocoo.org/extensions.html
• https://bitbucket.org/birkenfeld/sphinx-contrib

**autodoc** sphinx extension  See Also:
  • http://sphinx.pocoo.org/ext/autodoc.html

This extension can import the modules you are documenting, and pull in documentation from docstrings in a semi-automatic way.

**Note:** For Sphinx (actually, the Python interpreter that executes Sphinx) to find your module, it must be importable. That means that the module or the package must be in one of the directories on `sys.path`—adapt your `sys.path` in the configuration file accordingly.

For this to work, the docstrings must of course be written in correct reStructuredText. You can then use all of the usual Sphinx markup in the docstrings, and it will end up correctly in the documentation. Together with hand-written documentation, this technique eases the pain of having to maintain two locations for documentation, while at the same time avoiding auto-generated-looking pure API documentation.

**autogen** sphinx extension  See Also:
  • http://sphinx.pocoo.org/ext/autosummary.html#sphinx-autogen-generate-autodoc-stub-pages

The `sphinx-autogen` script can be used to conveniently generate stub documentation pages for items included in `autosummary` listings.

For example, the command

```
$ sphinx-autogen -o generated *.rst
```

will read all `autosummary` tables in the `*.rst` files that have the `:toctree:` option set, and output corresponding stub pages in directory `generated` for all documented items. The generated pages by default contain text of the form:

```
sphinx.util.relative_uri
========================
.. autofunction:: sphinx.util.relative_uri
```

If the `-o` option is not given, the script will place the output files in the directories specified in the `:toctree:` options.

**Generating stub pages automatically**  If you do not want to create stub pages with `sphinx-autogen`, you can also use this new config value:

```
.. confval:: autosummary_generate
```

Boolean indicating whether to scan all found documents for autosummary directives, and to generate stub pages for each.

Can also be a list of documents for which stub pages should be generated.

The new files will be placed in the directories specified in the `:toctree:` options of the directives.
Once again thanks a lot for your prompt replies. In case you are curious or you ever have to face the same problem than I, I finally chose blockdiag.

With only a few statements:

```python
.. blockdiag::

    {planners -> seq-mco -> seq-mco-planner-11;
     planners -> seq-mco -> seq-mco-planner-12;
     planners -> seq-mco -> "seq-mco-planner-1...";
     planners -> seq-opt -> seq-opt-planner-21;
     planners -> seq-opt -> seq-opt-planner-22;
     planners -> seq-opt -> "seq-mco-planner-2...";
     planners -> seq-sat -> seq-sat-planner-31;
     planners -> seq-sat -> seq-sat-planner-32;
     planners -> seq-sat -> "seq-mco-planner-3...";
     planners -> tempo-sat -> tempo-sat-planner-41;
     planners -> tempo-sat -> tempo-sat-planner-42;
     planners -> tempo-sat -> "seq-mco-planner-4...";
    }
```

I could generate the attached figure and embed it in the html and pdf docs generated with sphinx.

Just awesome!!
Komiya Takeshi showed me his tool called blockdiag, which is a DSL you can use to add diagrams in your documentation. The nice thing is that it provides a Sphinx extension so you can add diagrams in your documentation through simple expressions, and have Sphinx generate for you the diagrams on the fly.

There’s even an interactive online shell: http://interactive.blockdiag.com/

I’ve challenged Komiya to write a few diagrams I have for some Mozilla projects using his tool, and it took a few seconds for him to build them. So, I am going to use this in the future.

This is a small package containing a Sphinx theme named “cloud”, along with some related Sphinx extensions.

See next page for installation instructions.

Dependancies

sphinxcontrib-cheeseshop See Also:
http://pypi.python.org/pypi/sphinxcontrib-cheeseshop

doxygen sphinx extension See Also:
http://www.stack.nl/~dimitri/doxygen/

Doxygen docs examples See Also:
• http://pcsclite.alioth.debian.org/api/index.html

doxygen installation on GNU/Linux

Prerequisite We must be root to install the doxygen library (for centos).
Under ubuntu, we must be root only for make install.
Install in /opt/doxygen/1.7.2

cd <yourpath>/doxygen-1.7.2

./configure --prefix /opt/doxygen/1.7.2 --with-doxywizard

Results

cd /tmp/doxygen-1.7.2
./configure --prefix /opt/doxygen/1.7.2 --with-doxywizard
make

other option for configure: --enable-debug-log --enable-examples-build

sudo make install

Make the link to the current doxygen version

su - root
cd /opt/doxygen
ln -s 1.7.2 current

export PATH=/opt/doxygen/current/bin:$PATH

breathe sphinx extension  See Also:

• https://github.com/michaeljones/breathe

This is an extension to restructured text and Sphinx to be able to read and render the Doxygen xml output.

It is an easy way to include Doxygen information in a set of documentation generated by Sphinx.

The aim is to produce an autodoc like support for people who enjoy using Sphinx but work with languages other than Python. The system relies on the Doxygen’s xml output.

Gammu breathe example  See Also:

• http://wammu.eu/docs/manual/index.html

Sphinx html5slides extension  See Also:

• http://code.google.com/p/html5slides/

Contents

• Sphinx html5slides extension
  – Introduction
  – Using Hieroglyph

Introduction  hieroglyph is an extension for Sphinx which builds HTML5 slides from ReStructured Text documents.
Using Hieroglyph  Add hieroglyph as a Sphinx extension to your configuration:

```python
extensions = [
    'hieroglyph',
]
```

Build your slides:

```
$ sphinx -b html5slides output/slides
```

sphinx odt2sphinx extension  See Also:

- https://bitbucket.org/cdevienne/odt2sphinx
- Open ODF

Transform a OOO Writer document into Sphinx documentation sources

Sphinx plantum extension  See Also:

- https://pypi.python.org/pypi/sphinxcontrib-plantuml
- _plant_uml

Usage  First, you may need to specify plantuml command in your conf.py:

```python
plantuml = ['java', '-jar', '/path/to/plantuml.jar']
```

Instead, you can install a wrapper script in your PATH:

```bash
% cat <<EOT > /usr/local/bin/plantuml
#!/bin/sh -e
java -jar /path/to/plantuml.jar "$@
EOT
% chmod +x /usr/local/bin/plantuml
```

Then, write PlantUML text under `.. uml::` directive:

```plaintext
.. uml::

    Alice -> Bob: Hi!
    Alice <- Bob: How are you?
```

rstspreadsheet sphinx extension  See Also:

- http://pypi.python.org/pypi/rstspreadsheet

Add the `spreadsheet` directive to reStructuredText for Docutils and Sphinx

rst2qhc (Qt)  See Also:

http://code.google.com/p/rst2qhc/

Convert a collection of restructured text files into a Qt Help file and (optional) a Qt Help Project file.
Robin: bridge between doxygen (XML) and Sphinx via mongodb

See Also:
mongodb database

Contents

- Robin: bridge between doxygen (XML) and Sphinx via mongodb
  - Robin
  - Features
  - Prerequisites
  - Getting started
  - Status

Robin

We’re happy to announce robin, a new Doxygen/C++ to Sphinx bridge.

Robin provides an easy-to-use, easy-to-hack integration of Doxygen documentation into Sphinx.

Robin is licensed under the BSD and can be found at Bitbucket: https://bitbucket.org/reima/robin

Features

- Robust extraction of Doxygen XML data via an easy-to-hack parser
- Intermediate data is stored in a database (mongodb) for simple extraction and processing
- Directive-driven output; each directive provides callbacks and hooks which allows for deep customization
- Automated generation of driver ReST documents: Similar to automodule; however, robin generates actual ReST documents which can be inspected

Prerequisites

Robin expects a running mongodb on the local host.

It uses a minimal set of external libraries: Pymongo, sphinx, progressbar.

All of the dependencies can be easily installed using pip or easy_install.

Robin has been developed with Python 2.7; we have not tested previous versions.

Getting started

- Run Doxygen to generate XML documentation (GENERATE_XML=YES)
- Run extract-doxygen <path to XML> <project name>
- Run create-rst <project name> This generates several directories (classes, groups, etc.) Include the groups.rst into your toc
- Add ‘robin.sphinx’ to the Sphinx extensions
- Build (make html) for TOC update
- Build again (make clean && make html)
Status  We’re using robin internally for a large C++ codebase, and there are a few minor issues left that we hope to resolve soon (all of them are tracked on Bitbucket.)

After that, we expect that robin will go into “maintenance” mode focusing on bug fixes only.

If someone is interested in contributing, please get in touch with us.

Cheers, the robin developers

integrate doxygen documentation in sphinx  There is an easy way to include Doxygen information in a set of documentation generated by Sphinx.

Create 2 directories for sphinx and doxygen

Copy the doxygen output files in _build/html/_downloads directory

copy_doxygen_doc.bat  Télécharger le fichier de commandes DOS

python copytree_doxygen.py

pause

Copy with python shutil.rmtree and shutil.copytree  Télécharger le fichier python

# -*- coding: utf-8 -*-

"""
Copie des données doxygen.

Les données générées par doxygen sont dans le répertoire doxygen/html

Il faut copier ces données dans le répertoire sphinx/_build/html/_downloads

Précondition: le répertoire destination doit être vide.

.. function:: copytree(src, dst[, symlink=False[, ignore=None]])

    Recursively copy an entire directory tree rooted at *src*. The destination
directory, named by *dst*, must not already exist; it will be created as well
as missing parent directories. Permissions and times of directories are
copied with :func:`copystat`, individual files are copied using
:func:`copy2`.

2.7. Documentation
.. function:: rmtree(path[, ignore_errors[, onerror]])

.. index:: single: directory; deleting

Delete an entire directory tree; `path` must point to a directory (but not a
symbolic link to a directory). If `ignore_errors` is true, errors resulting
from failed removals will be ignored; if false or omitted, such errors are
handled by calling a handler specified by `onerror` or, if that is omitted,
they raise an exception.

```
from shutil import copytree, rmtree

destination = '_build/html/_downloads'
# le répertoire destination doit être vide
rmtree(destination)

# copie vers le répertoire destination
copytree(src='../doxygen/html', dst=destination)
```

Download the index.html source file from sphinx documents

First sphinx document

Le but de la bibliothèque logicielle est de proposer une API C permettant la
communication entre:

- l’application :term:`EUCLIDE`
- un ensemble de 40 :term:`boîtiers`
- un :term:`lecteur de cartes à puces` (RFID)
- une :term:`clé RF`


Second sphinx document

.. _doxygen_documentation:

=============
Doxygen Documentation
=============


Tree docs directory

```
+-doxygen
 |   |
 |   TagFile.xml
 |   
```
Projects using sphinx

See Also:

- http://sphinx.pocoo.org/latest/examples.html

The Atomic Simulation Environment (ASE)

See Also:

- https://wiki.fysik.dtu.dk/ase/

The Atomic Simulation Environment (ASE) is the common part of the simulation tools developed at CAMd. ASE provides Python modules for manipulating atoms, analyzing simulations, visualization etc.

Askbot

See Also:

http://askbot.org/doc/index.html

Python USB API for Canon digital cameras

See Also:

http://packages.python.org/canon-remote/index.html
Buildout  See Also:
  • http://www.buildout.org/index.html

Buildout is a Python-based build system for creating, assembling and deploying applications from multiple parts, some of which may be non-Python-based. It lets you create a buildout configuration and reproduce the same software later.

Django (mais pas de show source, dommage)  See Also:
  • https://docs.djangoproject.com/en/dev/
  • https://docs.djangoproject.com/en/1.3/internals/documentation/

Everything you need to know about Django (and then some).

How the Django documentation works  See Also:
  • https://docs.djangoproject.com/en/1.3/internals/documentation/

... and how to contribute.

Django’s documentation uses the Sphinx documentation system, which in turn is based on docutils.

The basic idea is that lightly-formatted plain-text documentation is transformed into HTML, PDF, and any other output format.

To actually build the documentation locally, you’ll currently need to install Sphinx – easy_install Sphinx should do the trick.

Dpm  See Also:


dpm (data package manager) is a command line tool and python library and for working with Data Packages and interacting with data hubs like those powered by CKAN such http://thedatahub.org/.

dpm is a simple way to ‘package’ data building on existing packaging tools developed for code. By putting data in a package, it gets labelled with standardized metadata and can be put in a dpm repository, such as http://thedatahub.org/ or a local one. Once in such a repository, the packages are easy to find and retrieve.

Eyesopen  See Also:

http://www.eyesopen.com/documentation

Sphinx gammu documentation

Gammu sphinx documentation  See Also:

http://wammu.eu/docs/manual/index.html

<gsavix@gmail.com>
heure de l’expéditeur  Envoyé à 13:15 (GMT-02:00). Heure locale : 10:35.
répondre à  sphinx-dev@googlegroups.com
à  sphinx-dev@googlegroups.com
date 18 février 2011 13:15
objet Re: [sphinx-dev] GSoC and Breathe
liste de diffusion  <sphinx-dev@googlegroups.com> Filtrar les messages de cette liste de diffusion
i think this is very usefull, i use debian lenny and spend 1 week to put gammu documentation that use sphinx, breathe and doxygen to work properly (with help of http://wammu.eu michal cihar) and now i could use this in "telecentros" public libraries for social public projects (with open source) here in são paulo.

thanks for all effort in sphinx, breathe and doxygen!

Gammu is a project providing abstraction layer for cell phones access.

It covers wide range of phones, mostly focusing on AT compatible phones and Nokia phones.

This manual describes all parts of Gammu, starting with information about Gammu project, going through API documentation for both python-gammu API and libGammu and covering SMS Daemon as well.

Other document

- http://gitorious.org/gammu/gsm-docs

**github2 using sphinx**   See Also:

- http://packages.python.org/github2/
- seealso:: github2 module

This is a Python library implementing all of the features available in version 2 of the Github API.

**Example**

```python
class Github(object):
    def __init__(self, username=None, api_token=None, requests_per_second=None, access_token=None, cache=None, proxy_host=None, proxy_port=8080):
        
        An interface to GitHub’s API:
        http://develop.github.com/

        .. versionadded:: 0.2.0
        The `requests_per_second` parameter
        .. versionadded:: 0.3.0
        The `cache` and `access_token` parameters
        .. versionadded:: 0.4.0
        The `proxy_host` and `proxy_port` parameters

        :param str username: your own GitHub username.
        :param str api_token: can be found at https://github.com/account
            (while logged in as that user):
        :param str access_token: can be used when no `username` and/or
            `api_token` is used. The `access_token` is the OAuth access
token that is received after successful OAuth authentication.
        :param float requests_per_second: indicate the API rate limit you’re
            operating under (1 per second per GitHub at the moment),
            or None to disable delays. The default is to disable delays (for
            backwards compatibility).
        :param str cache: a directory for caching GitHub responses.
        :param str proxy_host: the hostname for the HTTP proxy, if needed.
        :param str proxy_port: the hostname for the HTTP proxy, if needed (will
default to 8080 if a proxy_host is set and no port is set).
```

2.7. Documentation 145
self.request = GithubRequest(username=username, api_token=api_token,
    requests_per_second=requests_per_second,
    access_token=access_token, cache=cache,
    proxy_host=proxy_host,
    proxy_port=proxy_port)

self.issues = Issues(self.request)
self.users = Users(self.request)
self.repos = Repositories(self.request)
self.commits = Commits(self.request)
self.organizations = Organizations(self.request)
self.teams = Teams(self.request)
self.pull_requests = PullRequests(self.request)

def get_network_meta(self, project):
    """Get Github metadata associated with a project
    :param str project: GitHub project
    """
    return self.request.raw_request("/".join([self.request.github_url,
        project,
        "network_meta"])), {}

def get_network_data(self, project, nethash, start=None, end=None):
    """Get chunk of Github network data
    :param str project: GitHub project
    :param str nethash: identifier provided by `get_network_meta`
    :param int start: optional start point for data
    :param int stop: optional end point for data
    """
    data = {"nethash": nethash}
    if start:
        data["start"] = start
    if end:
        data["end"] = end

    return self.request.raw_request("/".join([self.request.github_url,
        project,
        "network_data_chunk"])),
        data)

def _handle_naive_datetimes(f):
    """Decorator to make datetime arguments use GitHub timezone
    :param func f: Function to wrap
    """
    def wrapper(datetime_):
        if not datetime_.tzinfo:
            datetime_ = datetime_.replace(tzinfo=GITHUB_TZ)
        else:
            datetime_ = datetime_.astimezone(GITHUB_TZ)
        return f(datetime_)
    wrapped = wrapper
    wrapped.__name__ = f.__name__
    wrapped.__doc__ = (f.__doc__
        + """\n        .. note:: Supports naive and timezone-aware datetimes"""
    )
return wrapped

@_handle_naive_datetimes
def datetime_to_ghdate(datetime_):
    """Convert Python datetime to Github date string
    
    :param datetime datetime_: datetime object to convert
    """
    return datetime_.strftime(GITHUB_DATE_FORMAT)

Macaron: Python O/R Mapper  See Also:

• http://nobrin.github.com/macaron/

Macaron is a small and simple object-relational mapper (ORM) for SQLLite. It is distributed as a single file module which has no dependencies other than the Python Standard Library.

Macaron provides provides easy access methods to SQLLite database. And it supports Bottle web framework through plugin mechanism.

Example:

```python
>>> import macaron
>>> macaron.macaronage(dbfile="members.db")
>>> team = Team.create(name="Houkago Tea Time")
>>> team.members.append(name="Azusa", part="Gt2")
<Member object 1>
>>> macaron.bake()
>>> azu = Member.get("part=?", ["Gt2"])
>>> print azu
<Member ‘Azusa : Gt2’>
>>> macaron.db_close()
```

Passlib  See Also:

See Also:

• http://packages.python.org/passlib/index.html
• http://packages.python.org/passlib/contents.html
• passlib

Pylons  See Also:

• http://docs.pylonsproject.org/en/latest/index.html

PysSCes  See Also:

• http://packages.python.org/PySCeS/
• http://pysces.sourceforge.net
PySCeS: the Python Simulator for Cellular Systems is an extendable toolkit for the analysis and investigation of cellular systems. It is available for download at http://pysces.sourceforge.net

Python GTK+ 3 Tutorial See Also:

Python See Also:
- http://docs.python.org/
Welcome! This is the documentation for Python 2.7.2.

python py3k See Also:
http://docs.python.org/py3k/

python dev See Also:
http://docs.python.org/dev/

Sphinx neuronvisio documentation See Also:
http://mattions.github.com/neuronvisio/

Figure 2.8: Prody analysis and modeling of protein structural dynamics

Sphinx Prody documentation See Also:
http://www.csb.pitt.edu/ProDy/index.html

Sphinx The sphinx doc is written with sphinx of course!
See Also:
http://sphinx.pocoo.org/latest/index.html
SQLAlchemy 0.7 Documentation  See Also:
http://docs.sqlalchemy.org/en/latest/intro.html#documentation-overview

sphinx applications

sphinx applications

Editing sphinx doc on the web

Baow  See Also:
http://www.baow.com/help/
Baow is a tool that makes it easy to organize your internet resources and create intelligent and beautiful web pages within Firefox web browser.

Code review  See Also:
Code review

Gerrit  See Also:
Gerrit Code review

Github Fork and Edit button  seealso:
- :ref:`github_fork_and_edit_button`
- https://github.com/blog/844-forking-with-the-edit-button
- https://confluence.atlassian.com/display/BITBUCKET/Fork+a+Repo,+Compare+Code,+and+Create+a+Pull+Request

You can commit file edits through GitHub web interface using Fork and Edit button Alternative, clone the repository using git, perform changes and push them back.

Plone collective GitHub repository has open-for-all contribution access. If you want to contribute changes without asking the maintainers to merge them, please add your GitHub username to your profile on plone.org and request access here.

pydocweb  See Also:
https://github.com/pv/pydocweb
Tool for collaboratively documenting Python modules via the web.
You might check out the Numpy documentation project (http://docs.scipy.org/numpy/Front%20Page/). I think they have a wiki/rst/sphinx like solution for editing documentation (http://code.google.com/p/pydocweb/) that is written in Django.

copypasta and sphinx  See Also:

- https://copypasta.credibl.es/
- Django ucomment

Copypasta is a collaborative editing tool. Readers submit edits, authors approve changes, everyone wins.

garlicsims and copypasta  See Also:

http://blog.garlicsim.org/post/3897688973/garlicsims-documentation-is-now-user-editable

A while ago I stumbled upon a very cool tool by Kurt Mackey called Copypasta. As the website describes it: “Copypasta is a collaborative editing tool. Readers submit edits, authors approve changes, everyone wins.” It’s still a young and experimental project, but it’s quite promising. It often happens that I’m reading a friend’s blog post and I see something that I want to fix, like a typo or a grammar mistake.

What I usually do is fire an email to that friend alerting him to the typo. This is of course inefficient. Copypasta lets you edit the page yourself in the browser, and then it shoots an email to the site owner with your proposed changes. Currently it requires the site owner to put the Copypasta button on the website for it to work; in the future Kurt might release a browser plugin that will let you edit any site on the internet.

That will be really awesome.

So I decided to put Copypasta on the GarlicSim documentation site. (This is a Sphinx-based documentation site.) Now anyone can offer fixes for GarlicSim’s docs!

Let me know if there are any bugs.

I hope that more Python package maintainers will do this for their projects’ documentation so we could all help to improve each other’s documentation.

Now what I want is a Sphinx backend for Copypasta which will be able to automatically change the documentation source in the repository, possibly creating a GitHub pull request with the changes to the docs source...

But now I’m fantasizing :)

comment from Henning  You could also try http://ucomment.org/

comment from Jonas Obrist (https://github.com/ojii/)  About the last paragraph, I implemented exactly that for readthedocs.org unfortunately it’s not 100% working yet, but stay tuned!

comment from Robert Kern  ou may want to give the Numpy Documentation Editor a look. It’s a web documentation editor for Sphinx documentation checked into a repository.

- http://docs.scipy.org/numpy/Front%20Page/
- http://code.google.com/p/pydocweb/
coolRR 1 day ago in reply to Robert Kern  Yeah, I remember I considered using it for the GarlicSim docs a few months ago but decided against it, don’t remember why.
Looking at it now I see that it’s not very easy to use. I now got into the NumPy docs and pressed “edit” and I got a confusing screen where I don’t understand how I’m supposed to edit anything. Possibly UX is the reason why pydocweb hasn’t gained wide adoption in the Python world.

Read the docs
See Also:
- http://readthedocs.org/
- http://readthedocs.org/dashboard/
- http://github.com/rtfd/readthedocs.org.git

Read the Docs hosts documentation for the open source community.
It supports Sphinx docs written with reStructuredText, and can pull from your Subversion, Bazaar, Git, and Mercurial repositories.
The code is open source, and available on github:
git clone http://github.com/rtfd/readthedocs.org.git

sphinx-wiki  See Also:
- https://bitbucket.org/kevindunn/sphinx-wiki/wiki/Home
A Mediawiki extension that allows ReStructuredText markup, compiled via Sphinx, to be used.

Examples  This extension is used on several sites operated by the author. Two that are publicly available as university courses:
- Statistics for Engineering
- Numerical methods
On both sites you can click Edit at the top of the page to see the page’s source.

sphinx tinkerer application  See Also:
- http://tinkerer.me/
- http://tinkerer.me/pages/documentation.html
What is Tinkerer?  Tinkerer is a blogging engine/static website generator powered by Sphinx. It allows blogging in reStructuredText format, comes with out-of-the-box support for post publishing dates, authors, categories, tags, post archive, RSS feed generation, comments powered by Disqus and more. Tinkerer is also highly customizable through Sphinx extensions.

Why Tinkerer?
- Because “hacker” is way overused nowadays.
- Because static websites are cool.
- Because you already write documentation in RST format and you love it.
- Because you have Pygments for sharing highlighted code.
- Because you can use your favorite Sphinx extensions right away.

Hosting on bitbucket  See Also:
http://tinkerer.me/doc/deploying.html#hosting-on-bitbucket

Versions

tinkerer versions

tinkerer 0.3  See Also:
http://tinkerer.bitbucket.org/2012/02/09/tinkerer_beta_0_3_released.html

What's New  Tinkerer went international! Spanish and Catalan translations are now available. To have your Tinkerer blog displayed in Spanish, add the following line to your conf.py:

```python
teasure = "es"
```

For Catalan, add:

```python
teasure = "ca"
```
Limited support for Facebook Comments as an alternative to Disqus comments

RSS feed enhancements: RSS feed auto-discovery and feed categories

Multiple bugfixes and minor style tweaks to the Modern theme

**tinkerer 0.2**

- Support for Drafts.
- Fixes for cross-references and embedded images which were not displaying correctly on home page and RSS feed.
- Ensure Tinkerer runs only from the blog root directory unless when setting up a new blog (this prevents accidental deletes and mysterious exceptions).
- Minimal support for documentation - prev and next links will be displayed on pages under doc/ or docs/ path.
- Many other small extension fixes.
- CSS fixes (gradient not showing in Firefox, page not scaling correctly on retina displays).

**Upgrading from 0.1**  There are a couple of steps required if upgrading from 0.1:

In your conf.py replace:

```python
# Add file patterns to exclude from build
exclude_patterns = []
```

with:

```python
# Add file patterns to exclude from build
exclude_patterns = ["drafts/*"]
```

This will make Sphinx stop warning you about drafts not being included in the build.

Make sure your master.rst file ends with a blank line. If not, append a blank line at the end of it.

Thank You!

A big Thank You to everyone who showed interest in the project and for the valuable feedback you provided.

**sphinx versions**

**Sphinx versions**

**Release 1.1.1 (Nov 1, 2011)**

- #791: Fix QtHelp, DevHelp and HtmlHelp index entry links.
- #792: Include “sphinx-apidoc” in the source distribution.
- #797: Don’t crash on a misformatted glossary.
- #801: Make intersphinx work properly without SSL support.
- #805: Make the Sphinx.add_index_to_domain method work correctly.
- #780: Fix Python 2.5 compatibility.
Release 1.1 (Oct 9, 2011)  See Also:

- http://sphinx.pocoo.org/changes.html#release-1-1-oct-9-2011

Incompatible changes

- The `py:module` directive doesn’t output its `platform` option value anymore. (It was the only thing that the directive did output, and therefore quite inconsistent.)
- Removed support for old dependency versions; requirements are now:
  - Pygments >= 1.2
  - Docutils >= 0.7
  - Jinja2 >= 2.3

Features added

- Added Python 3.x support.
- New builders and subsystems:
  - Added a Texinfo builder.
  - Added i18n support for content, a `gettext` builder and related utilities.
  - Added the `websupport` library and builder.
  - #98: Added a `sphinx-apidoc` script that autogenerates a hierarchy of source files containing autodoc directives to document modules and packages.
  - #273: Add an API for adding full-text search support for languages other than English. Add support for Japanese.
- Markup:
  - #138: Added an `index` role, to make inline index entries.
  - #454: Added more index markup capabilities: marking see/seealso entries, and main entries for a given key.
  - #460: Allowed limiting the depth of section numbers for HTML using the `toctree's numbered` option.
  - #586: Implemented improved `glossary` markup which allows multiple terms per definition.
  - #478: Added `py:decorator` directive to describe decorators.
  - C++ domain now supports array definitions.
  - C++ domain now supports doc fields (`:param x:` inside directives).
  - Section headings in `only` directives are now correctly handled.
  - Added `emphasize-lines` option to source code directives.
  - #678: C++ domain now supports superclasses.
- HTML builder:
  - Added `pyramid` theme.
  - #559: `::html_add_permalinks` is now a string giving the text to display in permalinks.
  - #259: HTML table rows now have even/odd CSS classes to enable “Zebra styling”.

Chapter 2. Development
- #554: Add theme option \texttt{sidebarwidth} to the basic theme.

- **Other builders:**
  - #516: Added new value of the ::latex_show_urls option to show the URLs in footnotes.
  - #209: Added ::text_newlines and ::text_sectionchars config values.
  - Added ::man_show_urls config value.
  - #472: linkcheck builder: Check links in parallel, use HTTP HEAD requests and allow configuring the timeout. New config values: ::linkcheck_timeout and ::linkcheck_workers.
  - #521: Added ::linkcheck_ignore config value.
  - #28: Support row/colspans in tables in the LaTeX builder.

- **Configuration and extensibility:**
  - #537: Added ::nitpick_ignore.
  - #306: Added env-get-outdated event.
  - Application.add_stylesheet() now accepts full URIs.

- **Autodoc:**
  - #564: Add ::autodoc_docstring_signature. When enabled (the default), autodoc retrieves the signature from the first line of the docstring, if it is found there.
  - #176: Provide private-members option for autodoc directives.
  - #520: Provide special-members option for autodoc directives.
  - #431: Doc comments for attributes can now be given on the same line as the assignment.
  - #437: autodoc now shows values of class data attributes.
  - autodoc now supports documenting the signatures of \texttt{functools.partial} objects.

- **Other extensions:**
  - Added the \texttt{sphinx.ext.mathjax} extension.
  - #443: Allow referencing external graphviz files.
  - Added inline option to graphviz directives, and fixed the default (block-style) in LaTeX output.
  - #590: Added caption option to graphviz directives.
  - #553: Added testcleanup blocks in the doctest extension.
  - #594: ::trim_doctest_flags now also removes \texttt{<BLANKLINE>} indicators.
  - #367: Added automatic exclusion of hidden members in inheritance diagrams, and an option to selectively enable it.
  - Added ::pngmath_add_tooltips.
  - The math extension displaymath directives now support \texttt{name} in addition to \texttt{label} for giving the equation label, for compatibility with Docutils.

- **New locales:**
  - #221: Added Swedish locale.
  - #526: Added Iranian locale.
  - #694: Added Latvian locale.
Development tools, Release 2012.06.18

- Added Nepali locale.
- #714: Added Korean locale.
- #766: Added Estonian locale.

- Bugs fixed:
  - #778: Fix “hide search matches” link on pages linked by search.
  - Fix the source positions referenced by the “viewcode” extension.

**Doxygen**

**See Also:**
- *doxygen sphinx extension*

---

**Figure 2.9: Doxygen logo**

Doxygen is a documentation system for C++, C, Java, Objective-C, Python, IDL (Corba and Microsoft flavors), Fortran, VHDL, PHP, C#, and to some extent D.

It can help you in three ways:

- It can generate an on-line documentation browser (in HTML) and/or an off-line reference manual (in $\text{LaTeX}$) from a set of documented source files. There is also support for generating output in RTF (MS-Word), PostScript, hyperlinked PDF, compressed HTML, and Unix man pages. The documentation is extracted directly from the sources, which makes it much easier to keep the documentation consistent with the source code.

- You can configure doxygen to extract the code structure from undocumented source files. This is very useful to quickly find your way in large source distributions. You can also visualize the relations between the various elements by means of include dependency graphs, inheritance diagrams, and collaboration diagrams, which are all generated automatically.

- You can even *abuse* doxygen for creating normal documentation (as I did for this manual).

Doxygen is developed under Linux and Mac OS X, but is set-up to be highly portable. As a result, it runs on most other Unix flavors as well. Furthermore, executables for Windows are available.

Furthermore, executables for Windows are available.

**Example**

///
/// <summary>
/// CR_RFID_DecouvrirLecteurs() Retourne la liste des lecteurs PS/SC de type
/// sous la forme d une liste de noms de la forme : "CodeRousseau RFID".
/// </summary>
/// <param name="ppListeNomsLecteursRFID"> [out] pointeur sur une liste de noms de lecteurs RFID.</param>
Projects using doxygen

Introduction  Doxygen supports a number of output formats where HTML is the most popular one. I’ve gathered some nice examples of real-life projects using doxygen. These are part of a larger list of projects that use doxygen. If you know other projects, let me know and I’ll add them.

C projects

C doxygen projects

libpcsclite  See Also:


libusbk  See Also:

- http://code.google.com/p/usb-travis/

C# projects

C Projects using doxygen

ini_parser  See Also:

C ini-parser modules
using System;

namespace IniParser
{
    /// <summary>
    /// Represents an error occurred while parsing data
    /// </summary>
    public class ParsingException : Exception
    {
        /// <summary>
        /// Initializes a new instance of the <see cref="ParsingException"/> class.
        /// </summary>
        public ParsingException()
        {
        }

        /// <summary>
        /// Initializes a new instance of the <see cref="ParsingException"/> class.
        /// </summary>
        /// <param name="msg">The message describing the exception cause.</param>
        public ParsingException(string msg)
        :
            base(msg) {}
        
        /// <summary>
        /// Initializes a new instance of the <see cref="ParsingException"/> class.
        /// </summary>
        /// <param name="msg">The message describing the exception cause.</param>
        /// <param name="innerException">An inner exception.</param>
        public ParsingException(string msg, Exception innerException)
        :
            base(msg, innerException) {}
    }
}

using System;
using System.Collections;
using System.Collections.Generic;

namespace IniParser
{
    /// <summary>
    /// Represents a collection of Keydata.
    /// </summary>
    public class KeyDataCollection : ICloneable, IEnumerable<KeyData>
    {
        #region Initialization

        /// <summary>
        /// Initializes a new instance of the <see cref="KeyDataCollection"/> class.
        /// </summary>
        public KeyDataCollection()
        {
        }
    }
}
_keyData = new Dictionary<string, KeyData>();

/// <summary>
/// Initializes a new instance of the <see cref="KeyDataCollection"/> class
/// from a previous instance of <see cref="KeyDataCollection"/>.
/// </summary>
/// <remarks>
/// Data is deeply copied
/// </remarks>
/// <param name="ori">
/// The instance of the <see cref="KeyDataCollection"/> class
/// used to create the new instance.</param>
public KeyDataCollection(KeyDataCollection ori)
{
    _keyData = new Dictionary<string, KeyData>();
    foreach (string key in _keyData.Keys)
    {
        _keyData.Add(key, (KeyData)ori._keyData[key].Clone());
    }
}

#region Properties
/// <summary>
/// Gets or sets the value of a concrete key.
/// </summary>
/// <remarks>
/// If we try to assign the value of a key which doesn’t exists,
/// a new key is added with the name and the value is assigned to it.
/// </remarks>
/// <param name="keyName">Name of the key</param>
/// <returns>
/// The string with key's value or null
/// if the key was not found.
/// </returns>
public string this[string keyName]
{
    get
    {
        if (_keyData.ContainsKey(keyName))
            return _keyData[keyName].Value;
        return null;
    }
    set
    {
        if (!_keyData.ContainsKey(keyName))
            return;
        _keyData[keyName].Value = value;
    }
}
#endregion
/// <summary>
/// An integer with the number of keys in the collection.</summary>
public int Count
{
    get { return _keyData.Count; }
}

#region Public Methods

/// <summary>
/// Adds a new key with the specified name and empty value and comments
/// </summary>
/// <remarks>
/// A valid key name is a string with NO blank spaces.
/// </remarks>
/// <param name="keyName">New key to be added.</param>
/// <returns>
/// <c>true</c> if a new empty key was added
/// <c>false</c> otherwise.
/// </returns>
/// <exception cref="ArgumentException">If the key name is not valid.</exception>
public bool AddKey(string keyName)
{
    // Checks valid key name
    // if ( !Assert.StringHasNoBlankSpaces(keyName) )
    //     throw new ArgumentException("Key name is not valid");

    if ( !_keyData.ContainsKey(keyName) )
    {
        _keyData.Add(keyName, new KeyData(keyName));
        return true;
    }

    return false;
}

/// <summary>
/// Adds a new key with the specified name and value and comments
/// </summary>
/// <remarks>
/// A valid key name is a string with NO blank spaces.
/// </remarks>
/// <param name="keyName">New key to be added.</param>
/// <param name="keyData">KeyData instance.</param>
/// <returns>
/// <c>true</c> if a new empty key was added
/// <c>false</c> otherwise.
/// </returns>
/// <exception cref="ArgumentException">If the key name is not valid.</exception>
public bool AddKey(string keyName, KeyData keyData)
{
    if (AddKey(keyName))
    {
        _keyData[keyName] = keyData;
        return true;
    }
return false;

}  
/// <summary>
/// Adds a new key with the specified name and value and comments
/// </summary>
/// <remarks>
/// A valid key name is a string with NO blank spaces.
/// </remarks>
/// <param name="keyName">New key to be added.</param>
/// <param name="keyValue">Value associated to the key.</param>
/// <returns>
/// <c>true</c> if a new empty key was added
/// <c>false</c> otherwise.
/// </returns>
/// <exception cref="ArgumentException">If the key name is not valid.</exception>
public bool AddKey(string keyName, string keyValue)
{
    if (AddKey(keyName))
    {
        _keyData[keyName].Value = keyValue;
        return true;
    }
    return false;
}

/// <summary>
/// Retrieves the data for a specified key given its name
/// </summary>
/// <param name="keyName">Name of the key to retrieve.</param>
/// <returns>
/// A <see cref="KeyData"/> instance holding
/// the key information or <c>null</c> if the key wasn’t found.
/// </returns>
public KeyData GetKeyData(string keyName)
{
    if (_keyData.ContainsKey(keyName))
    {
        return _keyData[keyName];
    }
    return null;
}

/// <summary>
/// Sets the key data associated to a specified key.
/// </summary>
/// <param name="data">The new <see cref="KeyData"/> for the key.</param>
public void SetKeyData(KeyData data)
{
    if (data != null)
    {
        if (_keyData.ContainsKey(data.KeyName))
            RemoveKey(data.KeyName);

        AddKey(data.KeyName, data);
    }
}
/// <summary>
/// Gets if a specified key name exists in the collection.
/// </summary>
/// <param name="keyName">Key name to search</param>
/// <returns><c>true</c> if a key with the specified name exists in the collection
/// <c>false</c> otherwise</returns>
public bool ContainsKey(string keyName)
{
    return _keyData.ContainsKey(keyName);
}

/// <summary>
/// Deletes a previously existing key, including its associated data.
/// </summary>
/// <param name="keyName">The key to be removed.</param>
/// <returns>
/// <c>true</c> if a key with the specified name was removed
/// <c>false</c> otherwise.
/// </returns>
public bool RemoveKey(string keyName)
{
    return _keyData.Remove(keyName);
}

#region IEnumerable<KeyData> Members

/// <summary>
/// Allows iteration through the collection.
/// </summary>
/// <returns>A strong-typed IEnumerator</returns>
public IEnumerator<KeyData> GetEnumerator()
{
    foreach (string key in _keyData.Keys)
    {
        yield return _keyData[key];
    }
}

#endregion

#region IEnumerable Members

/// <summary>
/// Implementation needed
/// </summary>
/// <returns>A weak-typed IEnumerator</returns>
IEnumerator IEnumerable.GetEnumerator()
{
    return _keyData.GetEnumerator();
}

#endregion

#region ICloneable Members

/// <summary>
/// Creates a new object that is a copy of the current instance.
/// </summary>

#endregion
/// <summary>
/// A new object that is a copy of this instance.
/// </summary>
public object Clone()
{
    return new KeyDataCollection(this);
}

#region Non-public Members
/// <summary>
/// Collection of KeyData for a given section
/// </summary>
private readonly Dictionary<string, KeyData> _keyData;
#endregion

---

### Doxygen versions

**Doxygen 1.8.0 (25-02-2012)**  See Also:

http://www.stack.nl/~dimitri/doxygen/changelog.html

**Contents**

- Doxygen 1.8.0 (25-02-2012)
  - Changes
    - Updated the manual and improved the look
    - doxytag removed
    - New features
      - Markdown support
      - tableofcontents
      - targets for 64 bits

**Changes**  Auto list items can now consist of multiple paragraphs.
The indentation of the (first line) of a new paragraph determines to which list item the paragraph belongs or if it marks the end of the list.

When UML_LOOK is enabled, relations shown on the edge of a graph are not shown as attributes (conform to the UML notation)

**Updated the manual and improved the look**  Made the contents in the navigation tree more consistent for groups, pages with subpages, and grouped subpages.

---

2.7. Documentation 163
id 669079: Latex: made the margins of latex page layout smaller using the geometry package.

**doxytag removed** The tool doxytag has been declared obsolete and is removed (it wasn’t working properly anyway). Same goes for the installdox script.

Updated the copyright in source code and doxywizard “about” to 2012. id 668008: HTML version of the manual now has the treeview enabled for easier navigation.

**New features**

**Markdown support** See Also:

Added support for *Markdown* formatting. This is enabled by default, but can be disabled by setting MARKDOWN_SUPPORT to NO.

When enabled the following is processed differently:

- tabs are converted to spaces according to TAB_SIZE.
- blockquotes are created for lines that start with one or more >’s (amount of >’s determne the indentation level).
- emphasis using *emphasize this* or _emphasis this_ or strong emphasis using emphasis this. Unlike classic Markdown ‘some_great_indentifier’ is not touched.
- code spans can be created using back-ticks, i.e. *here’s an example*
- Using three or more -’s or *’s alone on a line with only spaces will produce a horizontal ruler.
- A header can be created by putting a ===== (for h1) or —– (for h2) on the next line or by using 1 to 6 #’s at the start of a line for h1-h6.
- auto lists item can also start with + or * instead of only -
- ordered lists can be made using 1. 2. ... labels.
- verbatim blocks can be produced by indenting 4 additional spaces. Note that doxygen uses a relative indent of 4 spaces, not an absolute indent like Markdown does.
- Markdown style hyperlinks and hyperlink references.
- Simple tables can be created using the *Markdown Extra format*.
- *Fenced code blocks* are also supported, include language selection.
- files with extension .md or .markdown are converted to related pages.
- See the section about Markdown support in the manual for details.

It is now possible to add user defined tabs or groups of tabs to the navigation menu using the layout file (see the section of the manual about customizing the output for details).

**tableofcontents** Added new command tableofcontents (or [TOC] if you prefer Markdown) which can be used in a related page with sections to produce a table of contents at the top of the HTML page (for other formats the command has no effect).

When using SVG images and INTERACTIVE_SVG is enabled, a print icon will be visible along with the navigation controls to facilitate printing of the part of the graph that is visible on screen.
Added obfuscation of email addresses for the HTML output to make email harvesting more difficult.

**targets for 64 bits**  Added build targets for 64 bit Windows (thanks to Vladimir Simonov). The installer script is also updated to install a 64 bit version of doxygen on 64 bit systems and the 32 bit version on 32 bit systems.

Added support for using the HTML tag `<blockquote>` in comments.

Included patch by Gauthier Haderer that fixes some issues with the dbus XML parser.

Added support for Markdown style fenced code blocks.

Added option to `@code` command to force parsing and syntax highlighting according to a particular language.

Section of pages are now added to the navigation index.

Added support for cell alignment and table header shading in LaTeX and RTF output.

Added `-d` filteroutput option to show the output of an input filter (thanks to Albert for the patch).

id 668010: Latex: for Windows doxygen new generates a makepdf.bat file in the latex output dir to create the latex documentation.

**Doxygen 1.7.6.1**  See Also:

http://www.stack.nl/~dimitri/doxygen/changelog.html

**Changes**  Doxygen now reports its cache usage (for the symbol and the lookup cache) at the end of a run (if QUIET=NO), and recommends settings for SYMBOL_CACHE_SIZE and LOOKUP_CACHE_SIZE for your project if either cache is too small.

**New features**  Added new option LOOKUP_CACHE_SIZE to control the internal cache doxygen uses to find symbols given their name and a context.

- Python: added support for `@staticmethod`

**Markdown**

See Also:


**Introduction**

Markdown est un langage de balisage léger créé par John Gruber et Aaron Swartz. Le but de la syntaxe Markdown est d’offrir une syntaxe facile à lire et à écrire.

C’est-à-dire qu’un document formaté selon Markdown devrait pouvoir être publié comme tel, en texte, sans donner l’impression qu’il a été marqué par des balises ou des instructions de formatage.

Bien que la syntaxe Markdown ait été influencée par plusieurs filtres de conversion de texte vers HTML existants — incluant Setext, atx, Textile, reStructuredText, Grutatext et EtText — la source d’inspiration principale de la syntaxe Markdown est le format du courrier électronique en mode texte.

2.7. **Documentation**  165
Tutorials

Markdown tutorials

Documenting with mynt

See Also:

- http://mynt.mirroredwhite.com/

Overview

Another static site generator?

With the ever growing population of static site generators, all filling a certain need, I’ve yet to find one that allows the generation of anything but the simplest of blogs.

That’s where mynt comes in, being designed to give you all the features of a CMS with none of the often rigid implementations of those features.

Highlights

- Multiple ways of getting posts how you want, whether ordered by year and month, or by tag.
- Automatic generation of common pages such as those for archives and tags.
- Freedom in the markup language and rendering engine used.

Where to start?

See Also:

- http://mynt.mirroredwhite.com/quickstart/
- http://mynt.mirroredwhite.com/docs/

If you’re looking to get up and running quickly, I’d recommend giving the aptly named quickstart page a read.

If you’re pretty familiar with static site generators, I’d still recommend giving the quickstart page a read, although the docs may be a better jumping off point for you.

I have a question / want to help!

See Also:

- https://github.com/Anomareh/mynt
- http://mynt.mirroredwhite.com/community/

There are 2 main places to find other people messing around with mynt, GitHub and IRC (#mynt on irc.freenode.net).

For more info, head on over to the community page.
Rationale  The aim of this PEP is to standardize the high-level structure of docstrings: what they should contain, and how to say it (without touching on any markup syntax within docstrings).

The PEP contains conventions, not laws or syntax.

"A universal convention supplies all of maintainability, clarity, consistency, and a foundation for good programming habits too. What it doesn’t do is insist that you follow it against your will. That’s Python !"

--Tim Peters on comp.lang.python, 2001-06-16

If you violate these conventions, the worst you’ll get is some dirty looks.

But some software (such as the Docutils docstring processing system pep0256 ) will be aware of the conventions, so following them will get you the best results.

Documenting python projects with sphinx  See Also:

•  Documenting with sphinx
  http://sphinx.pocoo.org/latest
  http://packages.python.org/an_example_pypi_project/sphinx.html
  http://docs.python.org/dev/documenting/

sphinx  See Also:

•  http://sphinx.pocoo.org/latest

an example pypi project  See Also:

•  http://packages.python.org/an_example_pypi_project/sphinx.html

http://docs.python.org/dev/documenting  See Also:

http://docs.python.org/dev/documenting/
Development tools, Release 2012.06.18

geoserver See Also:
http://docs.geoserver.org/trunk/en/docguide/sphinx.html
This page contains syntax rules, tips, and tricks for using Sphinx and reStructuredText. For more information, please see this comprehensive guide to reStructuredText, as well as the Sphinx reStructuredText Primer.

Rest Documentation
See Also:
reStructuredText Primer

Docutils Rest Documentation
See Also:
• http://docutils.sourceforge.net/rst.html

Openalea Rest Documentation
See Also:
• http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/_sources/source/sphinx/rest_syntax.txt

Rest tools
http://rst.ninjs.org See Also:
• http://rst.ninjs.org/
• https://github.com/anru/rsted
Simple online editor for reStructuredText on Flask.

Documenting with socrates
See Also:
• http://pypi.python.org/pypi/socrates
Socrates is a simple static site generator. It’s geared towards blogs.
You write your posts in your favorite plain text to HTML language (e.g. Markdown, textile, reStructuredText) and save them as text files on your harddrive.
Socrates then takes them, and creates a full HTML site for you.
For free, you will get a home page which lists:
• latest posts,
• single post pages,
• category pages,
- archive pages,
- an about page
- and an atom feed.

Features

- Familiar Django and Jinja2 templates
- Simple install via pip
- Markdown, reStructuredText, Textile support
- YAML configuration
- Atom feed
- Github pages compatible
- Real HTML punctuation

Documentation

See Also:
The documentation is contained within the docs directory and is written in reStructuredText using Sphinx.
The documentation is easily read in a standard text editor.
However, you can build an HTML version like so:
$ pip install sphinx $ cd docs/ $ make html $ open _build/html/index.html
Or, you can view the online version of the latest documentation.

Documentation Advices

See Also:
- Parse Designing great api docs

Beautiful docs

See Also:
- https://github.com/PharkMillups/beautiful-docs

Twilio

See Also:
- http://twilio.com/engineering/2012/01/18/dont-skimp-on-documentation
Stripe

See Also:
https://stripe.com/docs/api?lang=python#top

Plone

See Also:

Designing Great API Docs 11 Jan 2012

See Also:
- http://blog.parse.com/2012/01/11/designing-great-api-docs/

Zotero documentation

See Also:
- https://www.zotero.org/start
- https://www.zotero.org/static/videos/zotero_1_5_cast.flv

Zotero [zoh-TAIR-oh] is a free, easy-to-use tool to help you collect, organize, cite, and share your research sources. It lives right where you do your work—in the web browser itself.

Zotero is, at the most basic level, a citation manager. It is designed to store, manage, and cite bibliographic references, such as books and articles.

In Zotero, each of these references constitutes an item.

Guide

See Also:
- https://www.zotero.org/support/quick_start_guide

Good documentation

Contents

- Good documentation
  - Raphael
  - Passlib
2.8 Data (SQL or NOSQL)

2.8.1 Data processing

SQL Data processing

Python camelot

See Also:
http://www.python-camelot.com/

Camelot is an open source RAD framework that leverages Python, Sqlalchemy and Qt to build rich desktop applications. Many built in features make applications built with Camelot user and developer friendly.

When you use Camelot, your applications will be:

- User and developer friendly
- Responsive, even with very large data sets, slow network connections and high CPU load.
- Integrated with Excel / Word / Outlook or OpenOffice

sqlite

See Also:

- https://www.sqlite.org/
- https://secure.wikimedia.org/wikipedia/fr/wiki/SQLite
Introduction SQLite est une bibliothèque écrite en C qui propose un moteur de base de données relationnelles accessible par le langage SQL.

SQLite implémente en grande partie le standard SQL-92 et des propriétés ACID.

Contrairement aux serveurs de bases de données traditionnels, comme MySQL ou PostgreSQL, sa particularité est de ne pas reproduire le schéma habituel client-serveur mais d’être directement intégrée aux programmes.

L’intégralité de la base de données (déclarations, tables, index et données) est stockée dans un fichier indépendant de la plateforme.

D. Richard Hipp, le créateur de SQLite, a choisi de mettre cette bibliothèque ainsi que son code source dans le domaine public, ce qui permet son utilisation sans restriction aussi bien dans les projets open source que dans les projets propriétaires.

sqlite tools

ohraimeur See Also:

- http://mbg-sqlclient.developpez.com/
- http://projets.developpez.com/projects/show/ohraimeur
- http://www.framasoft.net/article5155.html

Ohraimeur est avant tout une console offrant une coloration syntaxique et une autocomplémentation permettant de dialoguer interactivement avec le SGBD.

Cette application propose également des outils pour faciliter l’édition de requêtes.

- Affichage du MLD/R de la base de données
- Affichage du code source des objets SQL
- Génération d’ORM
- Interface graphique pour la création de tables.

sqlitedict See Also:

http://pypi.python.org/pypi/sqlitedict
Tabular Data processing

Python tablib module

See Also:

http://tablib.org/intro/

This part of the documentation covers all the interfaces of Tablib.

Tablib is a format-agnostic tabular dataset library, written in Python. It allows you to Pythonically import, export, and manipulate tabular data sets. Advanced features include, segregation, dynamic columns, tags / filtering, and seamless format import/export.

No-SQL database

IndexedDB database

See Also:

• https://developer.mozilla.org/en/IndexedDB
• http://www.html5rocks.com/en/tutorials/indexeddb/todo/

Contents

• IndexedDB database
  – Introduction
  – Basic Concepts About IndexedDB
  – Soma IndexedDB applications

Introduction  IndexedDB is an API for client-side storage of significant amounts of structured data and for high performance searches on this data using indexes.

While DOM Storage is useful for storing smaller amounts of data, it is less useful for storing larger amounts of structured data. IndexedDB provides a solution.

IndexedDB provides separate APIs for synchronous and asynchronous access.

The synchronous API is intended to be used inside workers.

Basic Concepts About IndexedDB  See Also:


IndexedDB is a way for you to persistently store data inside a user’s browser.

Because it lets you create web applications with rich query abilities, these applications can work both online and offline.

IndexedDB is useful for applications that store a large amount of data (for example, a catalog of DVDs in a lending library) and applications that don’t need persistent internet connectivity to work (for example, mail clients, to-do lists, and notepads)
Development tools, Release 2012.06.18

Soma IndexedDB applications  See Also:
http://hacks.mozilla.org/2012/02/announcing-the-december-dev-derby-winners/
  • 1st Place: eLibri by mar.castelluccio
  • 2nd Place: FileSystemDB by mar.castelluccio
  • 3rd Place: IndexedDB Editor by twolfson

mongodb database

See Also:
  • http://www.mongodb.org/

Contents
  • mongodb database
      – Presentation
      – Documentation
      – Mongodb on windows
      – Python tools

Presentation  MongoDB (from “humongous”) is a scalable, high-performance, open source, document-oriented database.

MongoDB is a high-performance schemaless database that allows you to store and retrieve JSON-like documents. MongoDB stores these documents in collections, which are analogous to SQL tables.

Because MongoDB is schemaless, there are no guarantees given to the database client of the format of the data that may be returned from a query; you can put any kind of document into a collection that you want.

Documentation  See Also:
http://docs.mongodb.org/manual/
  • http://docs.mongodb.org/manual/#getting-started

Mongodb on windows
  • http://www.mongodb.org/display/DOCS/Quickstart+Windows

Python tools
jaraco.modb  See Also:
   • http://pypi.python.org/pypi/jaraco.modb

MongoDB Object DataBase (MODB) for Python objects

Python ming framework  See Also:
   • http://merciless.sourceforge.net/index.html

MongoDB is a high-performance schemaless database that allows you to store and retrieve JSON-like documents. MongoDB stores these documents in collections, which are analogous to SQL tables.

Because MongoDB is schemaless, there are no guarantees given to the database client of the format of the data that may be returned from a query; you can put any kind of document into a collection that you want.

While this dynamic behavior is handy in a rapid development environment where you might delete and re-create the database many times a day, it starts to be a problem when you need to make guarantees of the type of data in a collection (because you code depends on it).

The goal of Ming is to allow you to specify the schema for your data in Python code and then develop in confidence, knowing the format of data you get from a query.

leveldb database

See Also:
   • http://code.google.com/p/leveldb/

Leveldb is a fast key-value storage library written at Google that provides an ordered mapping from string keys to string values.

Features
   • Keys and values are arbitrary byte arrays.
   • Data is stored sorted by key.
   • Callers can provide a custom comparison function to override the sort order.
   • The basic operations are Put(key,value), Get(key), Delete(key).
   • Multiple changes can be made in one atomic batch.
   • Users can create a transient snapshot to get a consistent view of data.
   • Forward and backward iteration is supported over the data.
   • Data is automatically compressed using the Snappy compression library.
   • External activity (file system operations etc.) is relayed through a virtual interface so users can customize the operating system interactions.
   • Detailed documentation about how to use the library is included with the source code.
Data Format

Data csv Format

See Also:


The comma-separated values (CSV) pseudo-file format is a set of file formats used to store tabular data in which numbers and text are stored in plain-text form that can be easily written and read in a text editor.

In fact, because the goal of reading and writing the format take precedence over consistency, there effectively is no CSV standard: only the understanding that plain text is delimited by a symbol.

Traditionally, lines in the text file represent rows in a table, and commas separate the columns.

Django csv See Also:

- Django CSV importer

CSV importer is a tool which allow you to transform easily a csv file into a python object or a django model instance. It is based on a django-style declarative model.

2.9 Drivers

2.9.1 Drivers

In computing, a device driver or software driver is a computer program allowing higher-level computer programs to interact with a hardware device.

A driver typically communicates with the device through the computer bus or communications subsystem to which the hardware connects. When a calling program invokes a routine in the driver, the driver issues commands to the device. Once the device sends data back to the driver, the driver may invoke routines in the original calling program. Drivers are hardware-dependent and operating-system-specific. They usually provide the interrupt handling required for any necessary asynchronous time-dependent hardware interface.

See Also:


linux module drivers

See Also:

- http://tjworld.net/books/ldd3
- http://kernelnewbies.org/Drivers
- http://kernelnewbies.org/USB
- http://www.linux-usb.org/
USB drivers

Universal Serial Bus

USB (Universal Serial Bus) is a specification to establish communication between devices and a host controller (usually personal computers). USB is intended to replace many varieties of serial and parallel ports. USB can connect computer peripherals such as mice, keyboards, digital cameras, printers, personal media players, flash drives, and external hard drives. For many of those devices, USB has become the standard connection method. USB was designed for personal computers, but it has become commonplace on other devices such as smartphones, PDAs and video game consoles, and as a power cord between a device and an AC adapter plugged into a wall plug for charging.

The design of USB is standardized by the USB Implementers Forum (USB-IF), an industry standards body incorporating leading companies from the computer and electronics industries.

Notable members have included Agere (now merged with LSI Corporation), Apple Inc., Hewlett-Packard, Intel, Microsoft, Sony and NEC.
Caractéristiques générales  L’Universal Serial Bus est une connexion à haute vitesse qui permet de connecter des périphériques externes à un ordinateur (hôte dans la terminologie USB). Il permet le branchement simultané de 127 périphériques par contrôleur (hôte). Le bus autorise les branchements et débranchements à chaud (« Hot-Plug », sans avoir besoin de redémarrer l’ordinateur) et fournit l’alimentation électrique des périphériques sous 5 V, dans la limite de 500 mA.

Le bus possède une topologie arborescente (dite également en étoile) : les feuilles de cet arbre sont les périphériques ; les nœuds internes sont des hubs qui permettent de greffer des sous-arborescences dans l’arborescence principale. On trouve dans le commerce ces hubs sous forme de petits boîtiers alimentés soit sur le bus, soit sur le secteur, et qui s’utilisent comme des multiprises. Certains périphériques intègrent également un hub (moniteurs, claviers…). Cependant, tout bus USB possède au moins un hub situé sur le contrôleur : le hub racine, qui peut gérer les prises USB de l’ordinateur.

Le nombre de hubs connectés en cascade est limité : hub racine compris, il ne doit pas exister plus de 7 couches dans l’arborescence.

USB 1.x  La version 1.x du bus peut communiquer dans deux modes:
- mode lent (1,5 Mbit/s) (« Low Speed ») permet de connecter des périphériques qui ont besoin de transférer peu de données, comme les claviers et souris ;
- mode rapide (12 Mbit/s, soit 1,5 Mo/s) (« Full Speed ») est utilisé pour connecter des imprimantes, scanners, disques durs, graveurs de CD et autres périphériques ayant besoin de plus de rapidité. Néanmoins il est insuffisant pour beaucoup de périphériques de stockage de masse (ce mode permet la vitesse « 10 X » des CD).

USB 2.0  USB 2.0 introduit un troisième mode permettant de communiquer à 480 Mbit/s (soit 60 Mo/s). Ce mode est appelé « high Speed ». Il est utilisé par les périphériques rapides : disques durs, graveurs… Mais en 2009, la plupart des périphériques ont une vitesse inférieure à ce que permet l’USB 2.0.

USB 3.0  La dernière version, l’USB 3.0, comporte un quatrième mode (« Super Speed ») permettant de communiquer à 4,8 Gbit/s (soit 600 Mo/s).

Les premiers appareils commercialisés sont prévus pour 2010.

Lorsque l’on parle d’un équipement USB, il est nécessaire de préciser la révision de la norme (1.1 ou 2.0) mais également la vitesse (Low, Full ou High Speed).

Une clef USB spécifiée en USB 2.0 n’est pas forcément High Speed si cela n’est pas précisé par un logo « High Speed ».

Le bus USB reste plus lent que des bus internes comme PCI ou AGP.


Lors de la connexion du périphérique à l’hôte, ce dernier détecte l’ajout du nouvel élément grâce au changement de la tension entre les fils D+ et D-. À ce moment, l’ordinateur envoie un signal d’initialisation au périphérique pendant 10 ms, puis lui fournit du courant grâce aux fils GND et VBUS (jusqu’à 100 mA). Le périphérique est alors alimenté en courant électrique et peut utiliser temporairement l’adresse par défaut (l’adresse 0).

L’étape suivante consiste à lui fournir son adresse définitive et à obtenir sa description : c’est la procédure d’énumération.

En effet, après avoir reçu son adresse, le périphérique transmet à l’hôte une liste de caractéristiques qui permettent à ce dernier de l’identifier (type, constructeur, nom, version). L’hôte, disposant de toutes les caractéristiques nécessaires est alors en mesure de charger le pilote approprié.
Les périphériques sont regroupés en types ou classes dans la terminologie USB. Tous les dispositifs d’une classe donnée reconnaissent le même protocole normalisé.

Il existe par exemple une classe pour les périphériques de stockage de masse (mass storage class, MSC), implémentée par la quasi-totalité des clés USB, disques durs externes, appareils photo et par certains baladeurs.

La plupart des systèmes d’exploitation possèdent des pilotes génériques, pour chaque type de périphérique. Ces pilotes génériques donnent accès aux fonctions de base, mais des fonctions avancées peuvent manquer.

**USB device classes**

**USB device classes**  USB defines class codes used to identify a device’s functionality and to load a device driver based on that functionality.

This enables every device driver writer to support devices from different manufacturers that comply with a given class code.
<table>
<thead>
<tr>
<th>Class</th>
<th>Usage</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>Device</td>
<td>Unspecified class 0</td>
<td>(Device class is unspecified. Interface descriptors are used for determining the required drivers)</td>
</tr>
<tr>
<td>0x01</td>
<td>Interface</td>
<td>Audio</td>
<td>Speaker, microphone, sound card</td>
</tr>
<tr>
<td>0x02</td>
<td>Both</td>
<td>Communications and CDC Control</td>
<td>Ethernet adapter, modem</td>
</tr>
<tr>
<td>0x03</td>
<td>Interface</td>
<td>Human Interface Device (HID)</td>
<td>Keyboard, mouse, joystick</td>
</tr>
<tr>
<td>0x05</td>
<td>Interface</td>
<td>Physical Interface Device (PID)</td>
<td>Force feedback joystick</td>
</tr>
<tr>
<td>0x06</td>
<td>Interface</td>
<td>Image</td>
<td>Webcam, scanner</td>
</tr>
<tr>
<td>0x07</td>
<td>Interface</td>
<td>Printer</td>
<td>Laser printer, inkjet printer, CNC machine</td>
</tr>
<tr>
<td>0x08</td>
<td>Interface</td>
<td>Mass Storage</td>
<td>USB flash drive, memory card reader, digital audio player, digital camera, external drive</td>
</tr>
<tr>
<td>0x09</td>
<td>Device</td>
<td>USB hub</td>
<td>Full bandwidth hub</td>
</tr>
<tr>
<td>0x0a</td>
<td>Interface</td>
<td>CDC-Data</td>
<td>(This class is used together with class 02h - Communications and CDC Control.)</td>
</tr>
<tr>
<td>0x0b</td>
<td>Interface</td>
<td>Smart Card</td>
<td>USB smart card reader</td>
</tr>
<tr>
<td>0x0d</td>
<td>Interface</td>
<td>Content Security</td>
<td>Fingerprint reader</td>
</tr>
<tr>
<td>0x0e</td>
<td>Interface</td>
<td>Video</td>
<td>Webcam</td>
</tr>
<tr>
<td>0x0f</td>
<td>Interface</td>
<td>Personal Healthcare</td>
<td></td>
</tr>
<tr>
<td>0xdc</td>
<td>Both</td>
<td>Diagnostic Device</td>
<td>USB compliance testing device</td>
</tr>
<tr>
<td>0xe0</td>
<td>Interface</td>
<td>Wireless Controller</td>
<td>Wi-Fi adapter, Bluetooth adapter</td>
</tr>
<tr>
<td>0xef</td>
<td>Both</td>
<td>Miscellaneous</td>
<td>ActiveSync device</td>
</tr>
<tr>
<td>0xfe</td>
<td>Interface</td>
<td>Application Specific</td>
<td>IrDA Bridge, Test &amp; Measurement Class (US-BTMC), DFU</td>
</tr>
<tr>
<td>0xff</td>
<td>Both</td>
<td>Vendor Specific</td>
<td>(This class code indicates that the device needs vendor specific drivers.)</td>
</tr>
</tbody>
</table>

See Also:

DFU (Device Firmware Upgrade) overview

Application Specific class example See Also:
dfu.c

/*
 * dfu-programmer
 * $Id: dfu.c 2706 2010-03-04 13:27:49Z pvergain $
USB devices and hosts, OTG (On-The-Go) Il existe deux types de périphériques USB:
  
  • ‘host’ : un périphérique de type ordinateur;
  
  • ‘device’ : un périphérique moins évolué, comme un appareil photo, une souris, un clavier.

Cependant, certains processeurs embarqués (comme l’AT91-RM9200) intègrent à la fois une interface host et une interface device.

La nouvelle norme OTG (On-The-Go) permet de connecter 2 périphériques devices entre eux sans passer par un host.

Les hosts USB En USB 1.0 et 1.1, il existe 2 types de contrôleurs host:

  • l’UHCI (Universal Host Controller Interface) développé par Intel Universal Host Controller Interface (UHCI) was created by Intel for USB 1.0 (full and low speeds). Far from being “universal”, it is actually proprietary and is incompatible with OHCI. Intel and VIA controllers generally use UHCI, while other vendors use OHCI.
  
  • l’OHCI (Open Host Controller Interface), un standard développé par Compaq, Microsoft et National Semiconductor Open Host Controller Interface, or OHCI, is an open standard.
Pour l’USB2.0 intervient également le contrôleur EHCI (Enhanced Host Controller Interface)
Linux gère les 3 types de contrôleurs: les pilotes sont chargés automatiquement selon les besoins.

A host controller interface (HCI)  See Also:
http://en.wikipedia.org/wiki/EHCI

A host controller interface (HCI) is a register level interface which allows a host controller for USB or FireWire to communicate with the operating system of a personal computer.

On the expansion card or motherboard controller, this involves much custom logic, with digital logic engines in FPGAs plus analog circuitry managing the high speed differential signals. On the software side, it requires a device driver (called a Host Controller Driver, or HCD).

USB device  Il existe deux types de périphériques USB (host ou device):
• Un host (hôte est en général un périphérique de type ordinateur;
• un device est en général un périphérique moins évolué, comme un appareil photo, une souris, un clavier.

See Also:
DFU (Device Firmware Upgrade) overview

Les périphériques USB vides  Todo
Les périphériques USB vides

Dongle  See Also:

Un dongle peut désigner toutes sortes de matériels comme des périphériques de stockage (clés USB), des clés permettant de se connecter à un réseau Wi-Fi, bluetooth, 3G ou infrarouge, ou encore de recevoir la TNT (relié à une antenne).

Historiquement, ils étaient branchés sur le port parallèle de l’ordinateur. Aujourd’hui il est possible de se brancher sur un port USB. L’encyclopédie Universalis a longtemps utilisé ce type de protection contre le piratage. Dans les années 1980, ce terme désignait des matériels destinés à valider le droit d’utiliser un logiciel, remplissant le rôle de « verrous matériels ».

USB on the go (OTG)  See Also:
• http://en.wikipedia.org/wiki/USB_On-The-Go
• http://fr.wikipedia.org/wiki/USB_On-The-Go
• http://de.wikipedia.org/wiki/Universal_Serial_Bus

Contents
• USB on the go (OTG)
  – Architecture
  – Principe
Architecture  Standard USB uses a master/slave architecture; a USB host acts as the protocol master, and a USB ‘Device’ acts as the slave. Only the Host can schedule the configuration and data transfers over the link.

The Devices cannot initiate data transfers, they only respond to requests given by a host. OTG introduces the concept that a ‘Device’ can perform both the master and slave roles, and so subtly changes the terminology.

Now a ‘Device’ can be either a ‘Host’ (acting as the link master) or a ‘Peripheral’ (acting as the link slave). The Device connected to the ‘A’ end of the cable at start-up acts as the Default Host, while the ‘B’ end acts as the Default Peripheral.

USB On-The-Go does not preclude using a USB hub, but it describes Host/Peripheral role swapping only for the case of a one-to-one connection where two OTG devices are directly connected.

Role swapping does not work through a standard hub, and one device will act as the Host and the other as the Peripheral until the hub is disconnected.

Principe  Le standard USB (USB 1.1/2.0) utilise une architecture maître/esclave un concentrateur USB (en anglais host USB) agit comme un Maître USB et un périphérique USB agit comme un Esclave.

Seuls les concentrateurs USB peuvent gérer la configuration et le transfert des données lors de la connexion.

La norme USB-OTG change cette situation. Les périphériques compatibles avec la norme USB-OTG sont capables d’ouvrir une session, contrôler la connexion et, échanger les rôles maître / périphérique.

Pour ce faire, USB-OTG introduit deux nouveaux protocoles : SRP (Session Request Protocol) et HNP (Host Negotiation Protocol).

Les périphériques USB ON-The-Go sont compatibles avec les normes USB 1.1/2.0 et se comportent comme un périphérique USB standard lorsqu’ils sont connectés avec des périphériques USB traditionnels (non OTG).

usb.ids

2  #       List of USB ID’s
3  #
4  #       Maintained by Stephen J. Gowdy <sgowdy+usb.ids@gmail.com>
5  #       If you have any new entries, send them to the maintainer.
6  #       Send entries as patches (diff -u old new).
7  #       The latest version can be obtained from
8  #       http://www.linux-usb.org/usb.ids

See Also:

- http://git.kernel.org/?p=linux/kernel/git/gregkh/usbutils.git;a=.blob;f=usb.ids;h=55e310e20da2d1264fd4c349ddd1149cbc8a611f;h=55e310e20da2d1264fd4c349ddd1149cbc8a611f
- http://www.linux-usb.org/usb.ids

http://www.linux-usb.org/usb-ids.html  This is a public repository of all known ID’s used in USB devices: ID’s of vendors, devices, subsystems and device classes. It is used in various programs (e.g., The USB Utilities) to display full human-readable names instead of cryptic numeric codes.

See Also:

- https://usb-ids.gowdy.us/read/UD/
- http://sourceforge.net/projects/pciids/
id3semiconductors Vendor ID (VID) 0x0b81  The id3semiconductors Vendor ID (VID) is 0x0b81

The id3semiconductor’s VID should be in this list:

- https://usb-ids.gowdy.us/read/UD/

<table>
<thead>
<tr>
<th>Product ID</th>
<th>Product Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Biothentic II smartcard reader with fingerprint sensor</td>
</tr>
<tr>
<td>0002</td>
<td>DFU-Enabled Devices (DFU)</td>
</tr>
<tr>
<td>0012</td>
<td>BioPAD biometric module (DFU + CDC)</td>
</tr>
<tr>
<td>0102</td>
<td>Certis V1 fingerprint reader</td>
</tr>
<tr>
<td>0103</td>
<td>Certis V2 fingerprint reader</td>
</tr>
<tr>
<td>0200</td>
<td>CL1356T / CL1356T5 / CL1356A smartcard readers (CCID)</td>
</tr>
<tr>
<td>0201</td>
<td>CL1356T / CL1356T5 / CL1356A smartcard readers (DFU + CCID)</td>
</tr>
<tr>
<td>0220</td>
<td>CL1356A FFPJP smartcard reader (CCID + HID)</td>
</tr>
<tr>
<td>0221</td>
<td>CL1356A smartcard reader (DFU + CCID + HID)</td>
</tr>
</tbody>
</table>

History  Update the list by the web. But be careful, the update is not immediate and must be approved by Stephen J. Gowdy <gowdy+usb.ids@gmail.com>.

From - Fri Mar 05 08:29:35 2010
Date: Thu, 4 Mar 2010 17:21:48 +0100 (CET)
From: "Stephen J. Gowdy" <gowdy@cern.ch>
X-X-Sender: gowdy@localhost
To: Patrick Vergain <patrick.vergain@id3.eu>
Subject: Re: diff -u usb.ids usb_with_id3.ids

It needed approved. I’ve just down that. They should appear online overnight.

On Thu, 4 Mar 2010, Patrick Vergain wrote:

> Le 04/03/2010 16:49, Stephen J. Gowdy a écrit :
> can you try using the web interface?
> I did try but unfortunately the vendor and product names do not appear and I don’t know for which reason

First insertion of id3 Semiconductors vendorId

$ diff -u usb.ids usb_with_id3.ids

--- usb.ids 2010-03-03 17:07:04.718750000 +0100
+++ usb_with_id3.ids 2010-03-04 15:06:03.406250000 +0100
@@ -8863,6 +8863,11 @@
  0b7b Taiko Denki Co., Ltd
  0b7c ITRAN Communications, Ltd
  0b7d Astrodienst, Inc.
+0b81 id3 Semiconductors
+  0102 Certis V1 fingerprint reader
+  0103 Certis V2 fingerprint reader
+  0200 CL1356T/T5 smartcard reader
+  0220 CL1356A smartcard reader
The usb.ids file on centos

// found with the command
// locate usb.ids
// ===============================
define FILE_USB_IDS "*/usr/share/hwdata/usb.ids"

update usb.ids on debian/ubuntu

sudo update-usbids

Locations of usb.ids on ubuntu

/usr/share/hwdata/usb.ids
/usr/share/misc/usb.ids
/var/lib/usbutils/usb.ids

Linux kernel usb documentation

linux kernel USB documentation

mailing list

linux-usb

• http://dir.gmane.org/gmane.linux.usb.general
• http://marc.info/?l=linux-usb
• http://www.spinics.net/lists/linux-usb/

Doc sur les sources Linux USB

Commandes USB linux usbutils    See Also:

• http://git.kernel.org/?p=linux/kernel/git/gregkh/usbutils.git;a=tree;h=refs/heads/master;hb=refs/heads/master
• http://www.kernel.org/pub/linux/utils/usb/usbutils/

\---usbutils-0.86
  aclocal.m4
  AUTHORS
  ChangeLog
  config.h.in
  configure
  configure.ac

2.9. Drivers
lsusb

lsusb

Comment votre système reconnaît-il la marque et le modèle du matériel ? Eh bien la commande **lsusb** utilise un fichier `usb.ids` (/var/lib/usbutils/usb.ids sous Debian, /var/lib/misc/usb.ids sous ubuntu). Notez la présence des lignes terminées par “root usb”. En fait, la commande vous propose plusieurs ports USB, mais la plupart du temps, ces ports appartiennent à un ou plusieurs hubs USB internes, qui s’accompagnent de contrôleurs (faisant office d’interface entre les périphériques et le système).

**See Also:**

- *Linux 2.6.31 USB sources*
- *usb.ids*

lspci

lspci -v | grep USB

La commande **lspci** permet d’identifier les contrôleurs USB mis en jeu.

**linux kernel 2.6.31 USB documentation and sources**

**linux kernel 2.6.31 USB sources**

**Linux 2.6.31 USB sources**
Development tools, Release 2012.06.18

| | | usb.h |
| | | +---gadget |
| | | amd5536udc.c |
| | | amd5536udc.h |
| | | at91_udc.c |
| | | at91_udc.h |
| | | atmel_usba_udc.c |
| | | atmel_usba_udc.h |
| | | audio.c |
| | | cdc2.c |
| | | ci13xxx_udc.c |
| | | ci13xxx_udc.h |
| | | composite.c |
| | | config.c |
| | | dummy_hcd.c |
| | | epautoconf.c |
| | | ether.c |
| | | file_storage.c |
| | | fsl_mx3_udc.c |
| | | fsl_qe_udc.c |
| | | fsl_qe_udc.h |
| | | fsl_udc_core.c |
| | | fsl_usb2_udc.h |
| | | f_acm.c |
| | | f_audio.c |
| | | f_ecm.c |
| | | f_loopback.c |
| | | f_obex.c |
| | | f_phonet.c |
| | | f_rndis.c |
| | | f_serial.c |
| | | f_sourcesink.c |
| | | f_subset.c |
| | | gadget_chips.h |
| | | gmidi.c |
| | | goku_udc.c |
| | | goku_udc.h |
| | | q_zero.h |
| | | imx_udc.c |
| | | imx_udc.h |
| | | inode.c |
| | | Kconfig |
| | | langwell_udc.c |
| | | langwell_udc.h |
| | | lb7a40x_udc.c |
| | | lb7a40x_udc.h |
| | | m66592_udc.c |
| | | m66592_udc.h |
| | | Makefile |
| | | ndis.h |
| | | net2280.c |
| | | net2280.h |
| | | omap_udc.c |
| | | omap_udc.h |
| | | printer.c |
| | | pxa25x_udc.c |
| | | pxa25x_udc.h |
Development tools, Release 2012.06.18

2.9. Drivers 189
Chapter 2. Development
To understand all the Linux-USB framework  To understand all the Linux-USB framework, you’ll use these resources:

- This source code. This is necessarily an evolving work, and includes kerneldoc that should help you get a current overview. (“make pdfdocs”, and then look at “usb.pdf” for host side and “gadget.pdf” for peripheral side.) Also, Documentation/usb has more information.

- The USB 2.0 specification (from http://www.usb.org), with supplements such as those for USB OTG and the various device classes. The USB specification has a good overview chapter, and USB peripherals conform to the widely known “Chapter 9”.

- Chip specifications for USB controllers. Examples include:
  - host controllers (on PCs, servers, and more)
  - peripheral controllers (in devices with Linux firmware, like printers or cell phones);
  - and hard-wired peripherals like Ethernet adapters.
Specifications for other protocols implemented by USB peripheral functions. Some are vendor-specific; others are vendor-neutral but just standardized outside of the www.usb.org team.

Here is a list of what each subdirectory here is, and what is contained in them:

**core**  This is for the core USB host code, including the usbfs files and the hub class driver (“khubd”).

**host**  This is for USB host controller drivers. This includes:
  - UHCI,
  - OHCI,
  - EHCI,
  - and others that might be used with more specialized “embedded” systems.

**gadget**  This is for USB peripheral controller drivers and the various gadget drivers which talk to them.

**image**  This is for still image drivers, like scanners or digital cameras.

**input**  This is for any driver that uses the input subsystem, like keyboard, mice, touchscreens, tablets, etc.

**media**  This is for multimedia drivers, like video cameras, radios, and any other drivers that talk to the v4l subsystem.

**net/usb**  This is for network drivers.

**serial**  This is for USB to serial drivers.

**storage**  This is for USB mass-storage drivers.

**class**  This is for all USB device drivers that do not fit into any of the above categories, and work for a range of USB Class specified devices.

**misc**  This is for all USB device drivers that do not fit into any of the above categories.

**Linux-USB sub-directories**

**usbcore**  This is for the core USB host code, including the usbfs files and the hub class driver (“khubd”).

```c
/*
 * drivers/usb/core/usb.c
 *
 * (C) Copyright Linus Torvalds 1999
 * (C) Copyright Johannes Erdfelt 1999-2001
 * (C) Copyright Andreas Gal 1999
 * (C) Copyright Gregory P. Smith 1999
 * (C) Copyright Deti Fliegl 1999 (new USB architecture)
```
NOTE! This is not actually a driver at all, rather this is
just a collection of helper routines that implement the
generic USB things that the real drivers can use.

Think of this as a "USB library" rather than anything else.
It should be considered a slave, with no callbacks. Callbacks
are evil.

*/

#include <linux/module.h>
#include <linux/moduleparam.h>
#include <linux/string.h>
#include <linux/bitops.h>
#include <linux/slab.h>
#include <linux/interrupt.h>  /* for in_interrupt() */
#include <linux/kmod.h>
#include <linux/init.h>
#include <linux/spinlock.h>
#include <linux/errno.h>
#include <linux/usb.h>
#include <linux/mutex.h>
#include <linux/workqueue.h>
#include <linux/debugfs.h>

#include <asm/io.h>
#include <linux/scatterlist.h>
#include <linux/mm.h>
#include <linux/dma-mapping.h>

#include "hcd.h"
#include "usb.h"

const char *usbcore_name = "usbcore";

2.9. Drivers
Driver for USB Mass Storage compliant devices

Current development and maintenance by:
(c) 1999-2003 Matthew Dharm (mdharm-usb@one-eyed-alien.net)

Developed with the assistance of:
(c) 2000 David L. Brown, Jr. (usb-storage@davidb.org)
(c) 2003-2009 Alan Stern (stern@rowland.harvard.edu)

Initial work by:
(c) 1999 Michael Gee (michael@linuxspecific.com)

usb_device_id support by Adam J. Richter (adam@yggdrasil.com):
(c) 2000 Yggdrasil Computing, Inc.

This driver is based on the ‘USB Mass Storage Class’ document. This
describes in detail the protocol used to communicate with such
devices. Clearly, the designers had SCSI and ATAPI commands in
mind when they created this document. The commands are all very
similar to commands in the SCSI-II and ATAPI specifications.

It is important to note that in a number of cases this class
exhibits class-specific exemptions from the USB specification.
Notably the usage of NAK, STALL and ACK differs from the norm, in
that they are used to communicate wait, failed and OK on commands.

Also, for certain devices, the interrupt endpoint is used to convey
status of a command.

Please see http://www.one-eyed-alien.net/~mdharm/linux-usb for more
information about this driver.

This program is free software; you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the
Free Software Foundation; either version 2, or (at your option) any
later version.

This program is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
General Public License for more details.

You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
675 Mass Ave, Cambridge, MA 02139, USA.

/*
#include <linux/sched.h>
#include <linux/errno.h>
#include <linux/freezer.h>
#include <linux/module.h>
#include <linux/init.h>
#include <linux/slab.h>
#include <linux/kthread.h>
#include <linux/mutex.h>
#include <linux/utsname.h>

#include <scsi/scsi.h>
#include <scsi/scsi_cmnd.h>
#include <scsi/scsi_device.h>

#include "usb.h"
#include "scsiglue.h"
#include "transport.h"
#include "protocol.h"
#include "debug.h"
#include "initializers.h"

#include "sierra_ms.h"
#include "option_ms.h"

/* Some informational data */
MODULE_AUTHOR("Matthew Dharm <mdharm-usb@one-eyed-alien.net>");
MODULE_DESCRIPTION("USB Mass Storage driver for Linux");
MODULE_LICENSE("GPL");

static unsigned int delay_use = 5;
module_param(delay_use, uint, S_IRUGO | S_IWUSR);
MODULE_PARM_DESC(delay_use, "seconds to delay before using a new device");

static char quirks[128];
module_param_string(quirks, quirks, sizeof(quirks), S_IRUGO | S_IWUSR);
MODULE_PARM_DESC(quirks, "supplemental list of device IDs and their quirks");
Development tools, Release 2012.06.18

| scsiglue.h |
| sddr09.c |
| sddr55.c |
| shuttle_usbat.c |
| sierra_ms.c |
| sierra_ms.h |
| transport.c |
| transport.h |
| unusual_alauda.h |
| unusual_cypress.h |
| unusual_datafab.h |
| unusual_devs.h |
| unusual_freecom.h |
| unusual_isd200.h |
| unusual_jumpshot.h |
| unusual_karma.h |
| unusual_onetouch.h |
| unusual_sddr09.h |
| unusual_sddr55.h |
| unusual_usbat.h |
| usb.c |
| usb.h |
| usual-tables.c |

**usbnnet**

```c
/*
 * USB Network driver infrastructure
 * Copyright (C) 2000-2005 by David Brownell
 * Copyright (C) 2003-2005 David Hollis <dhollis@davehollis.com>
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
 */
```

/*
 * This is a generic "USB networking" framework that works with several
 * kinds of full and high speed networking devices: host-to-host cables,
 * smart usb peripherals, and actual Ethernet adapters.
 *
 * These devices usually differ in terms of control protocols (if they
 * even have one!) and sometimes they define new framing to wrap or batch
 * Ethernet packets.  Otherwise, they talk to USB pretty much the same,
 * so interface (un)binding, endpoint I/O queues, fault handling, and other
 * issues can usefully be addressed by this framework.
 */
// #define DEBUG     // error path messages, extra info
// #define VERBOSE // more; success messages

#include <linux/module.h>
#include <linux/init.h>
#include <linux/netdevice.h>
#include <linux/etherdevice.h>
#include <linux/ctype.h>
#include <linux/ethtool.h>
#include <linux/workqueue.h>
#include <linux/mii.h>
#include <linux/usb.h>
#include <linux/usb/usbnet.h>
#define DRIVER_VERSION "22-Aug-2005"

/*-------------------------------------------------------------------------*/

\*
* Nineteen USB 1.1 max size bulk transactions per frame (ms), max.
* Several dozen bytes of IPv4 data can fit in two such transactions.
* One maximum size Ethernet packet takes twenty four of them.
* For high speed, each frame comfortably fits almost 36 max size
* Ethernet packets (so queues should be bigger).
*
* REVISIT qlens should be members of ‘struct usbnet’; the goal is to
* let the USB host controller be busy for 5msec or more before an irq
* is required, under load. Jumbograms change the equation.
* /
#define RX_MAX_QUEUE_MEMORY (60 * 1518)
#define RX_QLEN(dev) (((dev)->udev->speed == USB_SPEED_HIGH) ?
    (RX_MAX_QUEUE_MEMORY/(dev)->rx_urb_size) : 4)
#define TX_QLEN(dev) (((dev)->udev->speed == USB_SPEED_HIGH) ?
    (RX_MAX_QUEUE_MEMORY/(dev)->hard_mtu) : 4)

// reawaken network queue this soon after stopping; else watchdog barks
#define TX_TIMEOUT_JIFFIES (5*HZ)

// throttle rx/tx briefly after some faults, so khubd might disconnect()
// us (it polls at HZ/4 usually) before we report too many false errors.
#define THROTTLE_JIFFIES (HZ/8)

// between wakeups
#define UNLINK_TIMEOUT_MS 3

/*-------------------------------------------------------------------------*/

// randomly generated ethernet address
static u8 node_id [ETH_ALEN];

static const char driver_name [] = "usbnet";

+---net
|   \\
|    \---usb
|    |    asix.c
|    |    catc.c
|    |    cdc-phonet.c
|    |    cdc_eem.c
|    |    cdc_ether.c
|    |    cdc_subset.c

2.9. Drivers 197
USB HID

/*
 * USB HID support for Linux
 *
 * Copyright (c) 1999 Andreas Gal
 * Copyright (c) 2000-2005 Vojtech Pavlik <vojtech@suse.cz>
 * Copyright (c) 2005 Michael Haboustak <mike-@cinci.rr.com> for Concept2, Inc
 * Copyright (c) 2006-2008 Jiri Kosina
 * Copyright (c) 2007-2008 Oliver Neukum
 */

/*
 * This program is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License as published by the Free
 * Software Foundation; either version 2 of the License, or (at your option)
 * any later version.
 */

#include <linux/module.h>
#include <linux/slab.h>
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/list.h>
#include <linux/mm.h>
#include <linux/mutex.h>
#include <linux/spinlock.h>
#include <asm/unaligned.h>
#include <asm/byteorder.h>
#include <linux/input.h>
#include <linux/wait.h>
#include <linux/workqueue.h>

#include <linux/usb.h>

#include <linux/hid.h>
#include <linux/hiddev.h>
#include <linux/hid-debug.h>
#include <linux/hidraw.h>
#include "usbhid.h"

/*@ Version Information */
#define DRIVER_VERSION "v2.6"
#define DRIVER_AUTHOR "Andreas Gal, Vojtech Pavlik, Jiri Kosina"
#define DRIVER_DESC "USB HID core driver"
#define DRIVER_LICENSE "GPL"

---hid
|   | hid-a4tech.c
|   | hid-apple.c
|   | hid-belkin.c
|   | hid-cherry.c
|   | hid-chicony.c
|   | hid-core.c
|   | hid-cypress.c
|   | hid-debug.c
|   | hid-drff.c
|   | hid-ezkey.c
|   | hid-gaff.c
|   | hid-gyration.c
|   | hid-ids.h
|   | hid-input.c
|   | hid-kensington.c
|   | hid-kye.c
|   | hid-lg.c
|   | hid-lg.h
|   | hid-lg2ff.c
|   | hid-lgff.c
|   | hid-microsoft.c
|   | hid-monterey.c
|   | hid-ntrig.c
|   | hid-petalynx.c
|   | hid-pl.c
|   | hid-samsung.c
|   | hid-sjoy.c
|   | hid-sony.c
|   | hid-sunplus.c
|   | hid-tmff.c
|   | hid-topseed.c
|   | hid-wacom.c
|   | hid-zpff.c
|   | hidraw.c
| | Kconfig
| | Makefile
| | 
| \---usbhid
|   | hid-core.c
|   | hid-pidff.c
|   | hid-quirks.c
|   | hiddev.c
|   | Kconfig
|   | Makefile
|   | usbhid.h

2.9. Drivers
| usbkbd.c
| usbmouse.c

**usb_image**

```
/*
 * Version Information
 */
#define DRIVER_VERSION "v0.7.5 (30/10/2000)"
#define DRIVER_AUTHOR "Henning Zabel <henning@uni-paderborn.de>"
#define DRIVER_DESC "USB Driver for Mustek MDC800 Digital Camera"

/*
 * Version Information
 */
#define DRIVER_VERSION "v0.4.3"
#define DRIVER_AUTHOR "John Fremlin <vii@penguinpowered.com>, Oliver Neukum <Oliver.Neukum@lrz.uni-muenchen.de>"
#define DRIVER_DESC "Microtek Scanmaker X6 USB scanner driver"
```

```---image
| Kconfig
| Makefile
| mdc800.c
| microtek.c
| microtek.h
```

**linux kernel 2.6.31 USB documentation**

```---Documentation
| acm.txt
| anchors.txt
| authorization.txt
| callbacks.txt
| CREDITS
| dma.txt
| ehci.txt
| error-codes.txt
| gadget_printer.txt
| gadget_serial.txt
| hiddev.txt
| hotplug.txt
| iuuPhoenix.txt
| linux.inf
| misc_usbsevseg.txt
| mtouchusb.txt
| ohci.txt
| persist.txt
| power-management.txt
| proc_usb_info.txt
| rio.txt
| URB.txt
| usb-help.txt
| usb-serial.txt
| usbmon.txt
```
USB authorizations Source: GNU/Linux magazine N°101, janvier 2008, Mathieu Barthélémy

Linux 2.6.24 marque le début du support de contrôle d’autorisation d’accès au système pour les périphériques USB. Ce framework vise à permettre de contrôler si un tel périphérique a le droit de se connecter et de fournir une interaction avec l’espace utilisateur (notification de nouveau matériel essayant de se connecter) pour permettre la prise de décision.

Le moyen le plus basique d’interagir avec ce nouveau système est:

- l’entrée sysfs: `/sys/bus/usb/devices/<périphérique>/authorized` (0=interdit, 1=permis), qui définit le droit pour un périphérique de se connecter
- l’entrée `/sys/bus/usb/devices/usbX/authorized_default` (0=interdit, 1=permis), qui attribue une politique globale à chaque contrôleur.

Concrètement, une fois le matériel autorisé à se connecter, nous pouvons très bien souhaiter définir des critères personnalisés pour déterminer s’il pourra être utilisé: par quels utilisateurs, suivant le type de matériel... Ainsi, une organisation peut par exemple appliquer des politiques de restriction des clefs USB (pour éviter le vol de données confidentielles à l’organisation/entreprise), mais accepter les claviers ou souris. Dans ce cas, `authorized_default` sera positionnée à 0, et ce sera un mécanisme entièrement en espace utilisateur qui positionnera éventuellement à 1 cette entrée.

Illustrons par un exemple: créons un script à valeur pédagogique, appelé grâce à une règle udev à chaque découverte d’un nouveau périphérique USB. Il n’autorisera que les périphériques dont l’identifiant est présent dans un fichier de configuration. Nous formerons cet identifiant à partir du `vendorID` et du `productID`

```
checkUSBAuth.sh

#!/bin/sh
if [ ! -z "grep $2 /etc/authorized_dev_ids" ]; then
  # si le grep renvoie quelque chose, le périph est autorisé
  echo 1 > /sys$1/device/authorized
else
  # sinon ... interdit
  echo 0 > /sys$1/device/authorized
fi
```

Et, enfin la règle udev, à placer avant toute autre règle sur l’USB:

2.9. Drivers
#USB authorization

SUBSYSTEM=="usb_device", ACTION="add", RUN="/root/checkUSBAuth.sh %p %s{idVendor}%s{idProduct}"

Cette règle s’applique à tout le sous-système USB (SUBSYSTEM), seulement dans le cas d’un ajout (ACTION), et lance donc systématiquement notre script (RUN) avec en paramètre le chemin du périphérique (%p).

Le branchement d’un périphérique sur un bus ayant l’accès par défaut interdit doit provoquer l’apparition du message:

usb X-Y: Device is not authorized for usage

dans le log système.

Lors de son activation/autorisation, la mention:

usb X-Y: authorized to connect

Un `chmod u+x` sur le script et un `udevcontrol reload_rules` plus tard et le système est prêt.

### Compiling a Linux Driver

As the first step, we need to look a bit at how modules must be built. The build process for modules differs significantly from that used for user-space applications; the kernel is a large, stand alone program with detailed and explicit requirements on how its pieces are put together. The build process also differs from how things were done with previous versions of the kernel; the new build system is simpler to use and produces more correct results, but it looks very different from what came before. The kernel build system is a complex beast, and we just look at a tiny piece of it.

The files found in the Documentation/kbuild directory in the kernel source are required reading for anybody wanting to understand all that is really going on beneath the surface.

There are some prerequisites that you must get out of the way before you can build kernel modules. The first is to ensure that you have sufficiently current versions of the compiler, module utilities, and other necessary tools.

The file Documentation/Changes in the kernel documentation directory always lists the required tool versions; you should consult it before going any further. Trying to build a kernel (and its modules) with the wrong tool versions can lead to no end of subtle, difficult problems. Note that, occasionally, a version of the compiler that is too new can be just as problematic as one that is too old; the kernel source makes a great many assumptions about the compiler, and new releases can sometimes break things for a while.

If you still do not have a kernel tree handy, or have not yet configured and built that kernel, now is the time to go do it. You cannot build loadable modules for a 2.6 kernel without this tree on your filesystem. It is also helpful (though not required) to be actually running the kernel that you are building for.

Once you have everything set up, creating a makefile for your module is straightforward. In fact, for the “hello world” example shown earlier in this chapter, a single line will suffice:

```
obj-m := hello.o
```

Readers who are familiar with make, but not with the 2.6 kernel build system, are likely to be wondering how this makefile works. The above line is not how a traditional makefile looks, after all. The answer, of course, is that the kernel build system handles the rest. The assignment above (which takes advantage of the extended syntax provided by GNU make) states that there is one module to be built from the object file hello.o. The resulting module is named hello.ko after being built from the object file.
If, instead, you have a module called module.ko that is generated from two source files (called, say, file1.c and file2.c), the correct incantation would be:

```
obj-m := module.o
module-objs := file1.o file2.o
```

For a makefile like those shown above to work, it must be invoked within the context of the larger kernel build system. If your kernel source tree is located in, say, your ~/kernel-2.6 directory, the make command required to build your module (typed in the directory containing the module source and makefile) would be:

```
make -C ~/kernel-2.6 M='pwd' modules
```

This command starts by changing its directory to the one provided with the -C option (that is, your kernel source directory). There it finds the kernel’s top-level makefile. The M= option causes that makefile to move back into your module source directory before trying to build the modules target. This target, in turn, refers to the list of modules found in the obj-m variable, which we’ve set to module.o in our examples.

Typing the previous make command can get tiresome after a while, so the kernel developers have developed a sort of makefile idiom, which makes life easier for those building modules outside of the kernel tree. The trick is to write your makefile as follows:

```make
# If KERNELRELEASE is defined, we’ve been invoked from the
# kernel build system and can use its language.
ifeq ($(KERNELRELEASE),)
    obj-m := hello.o
# Otherwise we were called directly from the command line; invoke the kernel build system.
else
    KERNELDIR ?= /lib/modules/$(shell uname -r)/build
    PWD := $(shell pwd)
else
    KERNELDIR ?= /lib/modules/$(shell uname -r)/build
    PWD := $(shell pwd)
endif

default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
```

Once again, we are seeing the extended GNU make syntax in action. This makefile is read twice on a typical build. When the makefile is invoked from the command line, it notices that the KERNELRELEASE variable has not been set. It locates the kernel source directory by taking advantage of the fact that the symbolic link build in the installed modules directory points back at the kernel build tree. If you are not actually running the kernel that you are building for, you can supply a KERNELDIR= option on the command line, set the KERNELDIR environment variable, or rewrite the line that sets KERNELDIR in the makefile. Once the kernel source tree has been found, the makefile invokes the default: target, which runs a second make command (parameterized in the makefile as $(MAKE)) to invoke the kernel build system as described previously. On the second reading, the makefile sets obj-m, and the kernel makefiles take care of actually building the module.

This mechanism for building modules may strike you as a bit unwieldy and obscure. Once you get used to it, however, you will likely appreciate the capabilities that have been programmed into the kernel build system. Do note that the above is not a complete makefile; a real makefile includes the usual sort of targets for cleaning up.

**USB GNU/Linux tools**

**linux USB tools**

2.9. Drivers
usbutils linux  usbutils contains a utility for inspecting devices connected to the USB bus.
It requires a Linux kernel version 2.3.15 or newer (supporting the `/proc/bus/usb` interface).

See Also:
- http://github.com/gregkh/usbutils
- http://github.com/gregkh/usbutils/tree/master

usbutils news

usbutils 0.87 release

from  Greg KH <greg@kroah.com>
to   linux-usb@vger.kernel.org
cc   linux-kernel@vger.kernel.org
date Thu, Mar 18, 2010 at 2:51 AM
subject usbutils 0.87 release
mailing list <linux-usb.vger.kernel.org> Filter messages from this mailing list
mailed-by vger.kernel.org

hide details 2:51 AM (8 hours ago)

Here’s the 0.87 release of usbutils.

There are some fixes for people having problems with the latest libusb
version, minor code edits, and some new USB class logic. The whole
changelog is below.

The package can be downloaded from kernel.org:
http://www.kernel.org/pub/linux/utils/usb/usbutils/

We’ve switched over to using git for development now, which makes things
much easier than the old cvs tree. The tree can be found on both
kernel.org and github.com if you want to fork it and send us changes
easier:
http://git.kernel.org/?p=linux/kernel/git/gregkh/usbutils.git
http://github.com/gregkh/usbutils/tree/master

thanks,

greg k-h

--------

Shortlog of changes since last release (0.86):

Aurelien Jarno (1):
    lsusb.c: correctly dump ccid devices

Greg Kroah-Hartman (11):
    lots of trailing whitespace removed.
    add autogen.sh script
    coding style cleanups for .h files.
    names.c: fix up some compiler warnings
    coding style cleanups for usbmisc.c
    names.c: fix lots of coding style issues
devtree.c: coding style cleanups
lsusb.c: coding style fixes
lsusb.c: fix some build warnings.
usb.ids: Reserve EEM Gadget id for the Linux Foundation
add lsusb.py from Kurt Garloff <garloff@suse.de>

Philip A. Prindeville (1):
Fix build issue with libusb location

**lsusb.py correctly dump ccid devices**  See Also:
http://github.com/gregkh/usbutils/commit/277327cd6a61d96f9c322289e055ec2b4a129096

**lsusb.c correctly dump ccid devices**

CCID has now an official class according to:
http://www.usb.org/developers/defined_class/

This patch removes the hard-coded value and replace it by a #define, and also call dump_ccid_device() when use in a common class interface descriptor.

Signed-off-by: Aurelien Jarno <aurelien@aurel32.net>
Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

```c
+#ifndef USB_CLASS_CCID
+#define USB_CLASS_CCID 0x0b
+#endif
```

**lsusb.py**

**usbutils**  See Also:
http://github.com/gregkh/usbutils

**usbview**  See Also:
http://github.com/gregkh/usbview/

USBView is a small GTK application to show what the device tree of the USB bus looks like. It shows a graphical representation of the devices that are currently plugged in, showing the topology of the USB bus. It also displays information on each individual device on the bus.

**usbutils/lsusb.py**  See Also:
http://github.com/gregkh/usbutils/blob/v0.87/lsusb.py

**USB classes**
Contents

USB ccid driver

- GNU/Linux USB ccid driver
- Windows USB ccid driver
- Computer/Smart Card (PC/SC)
- CCID and PC/SC people
- Glossary

See Also:

- CCID Chip/Smart Card Interface Devices Glossary

GNU/Linux USB ccid driver

linux CCID

libccid

Introduction

This package provides the source code for a generic USB CCID (Chip/Smart Card Interface Devices) driver and ICCD (Integrated Circuit(s) Card Devices).

See the USB CCID and ICCD specifications from the USB working group.

See Also:

- https://alioth.debian.org/frs/?group_id=30105&release_id=1377
- http://pcsclite.alioth.debian.org/ccid.html
- http://pcsclite.alioth.debian.org/

CCID driver

- http://ludovicrousseau.blogspot.com/
- http://pcsclite.alioth.debian.org/
- Project page:
  - Sending usefull logs
  - Checking for reader compliance
  - Reader matrix
- Download libccid
• Subversion repository:
  – viewsvn
  – wsvn
  – svn co svn://svn.debian.org/pcsclite/trunk/Drivers/ccid

Main CCID/ICCD features supported
• Exchange levels:
  – short APDU
  – extended APDU (with some limitations)
  – TPDU
  – character
• multi-slot readers
• composite CCID device
• PC/SC v2 part 10 features:
  – GET_FEATURE_REQUEST
  – secure PIN verify (FEATUREVERIFY_PIN_DIRECT)
  – modify PIN entry (FEATURE_MODIFY_PIN_DIRECT)
  – reader PIN properties (FEATURE_IFD_PIN_PROPERTIES)
  – Multifunctional Card Terminal reader direct (FEATURE_MCT_READER_DIRECT)
  – retrieve reader properties in TLV form (FEATURE_GET_TLV_PROPERTIES)
• Data rates list
• Localize LCD display messages (Gemalto GemPC PIN PAD, Covadis Véga-Alpha)
• Extended APDU (for T=1 cards only and if your reader is in TPDU mode or extended APDU mode)
• SCardGetAttrib() attributes
• ICCD versions A and B

libccid installation

libccid installation on GNU/Linux    See Also:

CCID

libccid installation on Debian/ubuntu

    sudo aptitude install libccid-dev

libccid manual Installation    Source: http://pcsclite.alioth.debian.org/ccid.html#CCID_compliant

Warning: pcsclite must be installed before libccid.
libccid Installation on GNU/Linux

bunzip2 ccid-x.y.z.tar.bz2
tar xvf ccid-x.y.z.tar
cd ccid-x.y.z
./configure PCSC_CFLAGS=-I/opt/pcsclite/current/include/PCSC LIBUSB_CFLAGS=-I/opt/libusb/current/include/libusb-1.0 -I/.opt/libusb/current/include/libusb-1.0 -L/opt/pcsclite/current/lib -lpcsclite LIBUSB_LIBS=-L/opt/libusb/current/lib -lusb-1.0 -fvisibility=hidden

Example for libccid 1.4.3

./configure PCSC_CFLAGS=-I/opt/pcsclite/current/include/PCSC LIBUSB_CFLAGS=-I/opt/libusb/current/include/libusb-1.0 -I/.opt/libusb/current/include/libusb-1.0 -L/opt/pcsclite/current/lib -lpcsclite LIBUSB_LIBS=-L/opt/libusb/current/lib -lusb-1.0 -fvisibility=hidden

libccid has been configured with following options:
Version: 1.4.3
User binaries: /opt/ccid/1.4.3/bin
Configuration files: /opt/ccid/1.4.3/etc
Host: i686-pc-linux-gnu
Compiler: gcc
Preprocessor flags:
Compiler flags: -g -O2
Preprocessor flags:
Linker flags:
Libraries:
PCSC_CFLAGS: -I/opt/pcsclite/current/include/PCSC
PCSC_LIBS: -L/opt/pcsclite/current/lib -lpcsclite
PTHREAD_CFLAGS: -pthread
PTHREAD_LIBS:
BUNDLE_HOST: Linux
DYN_LIB_EXT: so
LIBUSB_CFLAGS: -I/opt/libusb/current/include/libusb-1.0
LIBUSB_LIBS: -L/opt/libusb/current/lib -lusb-1.0
SYMBOL_VISIBILITY: -fvisibility=hidden
NOCLASS:
libusb support: yes
composite as multisolot: no
multi threading: yes
bundle directory name: ifd-ccid.bundle
USB drop directory: /opt/pcsclite/current/drivers
serial Twin support: no
serial twin install dir: /serial
compiled for pcsc-lite: yes
class driver: yes

make

make install
copy the src/92_pcscd_ccid.rules file in udev directory (/etc/udev/rules.d/)

After make install See Also:

usb.ids
GNU/Linux CCID compliant reader  See Also:

- [http://pcsclite.alioth.debian.org/ccid.html#CCID_compliant](http://pcsclite.alioth.debian.org/ccid.html#CCID_compliant)

## Contents

- GNU/Linux CCID compliant reader
  - How to become CCID compliant
  - `sudo apt-cache search “PC/SC”`
  - `sudo aptitude install libusb-dev`
  - `sudo aptitude install libpcsclite-dev`
    * ./configure
    * make
  - `/etc/init.d/pcscd stop`
  - `sudo ./src/parse > output.txt`
  - CL1356T5 output.txt
  - CL1356A+ output.txt

### How to become CCID compliant
Just get the source code of this driver and do:

tar xzvf ccid-x.y.z.tar.gz
cd ccid-x.y.z
./configure
make
sudo ./src/parse > output.txt

and send me (ludovic.rousseau@free.fr) the generated output.txt file.

If your reader is CCID compliant and you would like to add it to the list of readers I would also need:

- The URL of a web page describing the reader. Typically the web page of the reader description on the manufacturer web site.
- A picture of the reader. In general a picture is already available on the manufacturer web page described above.

```
sudo apt-cache search “PC/SC”
```

```
patrick@vercors:~$ sudo apt-cache search "PC/SC"
libpcsc-lite-dev - Middleware to access a smart card using PC/SC (development files)
libpcsc-lite1 - Middleware to access a smart card using PC/SC(library)
libacrm38u - PC/SC driver for the ACR38U smart card reader
libasedrive-serial - PC/SC driver for the Athena ASEDDrive IIIe serial smart card reader
libasedrive-usb - PC/SC driver for the Athena ASEDDrive IIIe USB smart card reader
libccid - PC/SC driver for USB CCID smart card readers
libgcr410 - PC/SC driver for GemPlus GCR410 serial SmartCard interface
libgempc410 - PC/SC driver for the GemPC 410, 412, 413 and 415 smart card readers
libgempc4130 - PC/SC driver for the GemPC 410, 412, 413 and 415 smart card readers
libopenct1 - middleware framework for smart card terminals (libraries)
libopenct1-dbg - middleware framework for smart card terminals (libraries)
libopenct1-dev - headers and development libraries for libopenct1
libpcsc-perl - Perl interface to the PC/SC middleware (development)
libpcsc-perl1 - Perl interface to the PC/SC middleware
libpcscada-dev - Ada bindings to PC/SC middleware (development)
libpcscada0 - Ada bindings to PC/SC middleware
openct - middleware framework for smart card terminals
pcc-utils - Some tools to use with smart cards and PC/SC
pccscada-dbg - Ada bindings to PC/SC middleware (debug)
pccscd - Middleware to access a smart card using PC/SC (daemon side)
python-pyscard - Python wrapper above PC/SC API
pccsc-omnikey - PC/SC driver for Omnikey Cardman Smartcard readers (binary-only)
```

```
sudo aptitude install libusb-dev
```

**See Also:**

- [libccid Installation on GNU/Linux](#)
- [libpcsclite Installation](#)

```
patrick@vercors:~$ sudo aptitude install libusb-dev
Lecture des listes de paquets... Fait
Construction de l’arbre des dépendances
Lecture des informations d’état... Fait
Lecture de l’information d’état étendu
Initialisation de l’état des paquets... Fait
```
Les NOUVEAUX paquets suivants vont être installés : libusb-dev
0 paquets mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de télécharger 38,8ko d’archives. Après dépaquetage, 344ko
seront utilisés.
Écriture de l’information d’état étendu... Fait
[38,8kB]
38,8ko téléchargés en 0s (102ko/s)
Sélection du paquet libusb-dev précédemment désélectionné.
(lecture de la base de données... 464692 fichiers et répertoires déjà
installés.)
Dépaquetage de libusb-dev (à partir de
.../libusb-dev_2%3a0.1.12-13_i386.deb) ...
Traitement des actions différées (« triggers ») pour « doc-base »...
Processing 1 added doc-base file(s)...
Registering documents with scrollkeeper...
Traitement des actions différées (« triggers ») pour « man-db »...
Paramétrage de libusb-dev (2:0.1.12-13) ...

**sudo aptitude install libpcsclite-dev**

patrick@vercors:~$ sudo aptitude install libpcsclite-dev
[sudo] password for patrick:
Lecture des listes de paquets... Fait
Construction de l’arbre des dépendances
Lecture des informations d’état... Fait
Lecture de l’information d’état étendu
Initialisation de l’état des paquets... Fait
Les NOUVEAUX paquets suivants vont être installés :
libpcsclite-dev
0 paquets mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à
jour.
Il est nécessaire de télécharger 62,1ko d’archives. Après dépaquetage, 205ko
seront utilisés.
Écriture de l’information d’état étendu... Fait
1.5.3-lubuntu1 [62,1kB]
62,1ko téléchargés en 0s (151ko/s)
Sélection du paquet libpcsclite-dev précédemment désélectionné.
(lecture de la base de données... 464675 fichiers et répertoires déjà
installés.)
Dépaquetage de libpcsclite-dev (à partir de
.../libpcsclite-dev_1.5.3-lubuntu1_i386.deb) ...
Paramétrage de libpcsclite-dev (1.5.3-lubuntu1) ...
Lecture des listes de paquets... Fait
Construction de l’arbre des dépendances
Lecture des informations d’état... Fait
Lecture de l’information d’état étendu
Initialisation de l’état des paquets... Fait
Écriture de l’information d’état étendu... Fait

`./configure`

libccid has been configured with following options:

Version: 1.3.11

2.9. Drivers
Development tools, Release 2012.06.18

User binaries: NONE/bin  
Configuration files: NONE/etc

Host: i686-pc-linux-gnu  
Compiler: gcc  
Preprocessor flags: 
Compiler flags: -g -O2  
Preprocessor flags: 
Linker flags: 
Libraries:

PCSC_CFLAGS: -I/usr/include/PCSC  
PCSC_LIBS: -lpcsclite  
PTHREAD_CFLAGS: -pthread  
PTHREAD_LIBS: 
BUNDLE_HOST: Linux  
DYN_LIB_EXT: so  
LIBUSB_CFLAGS: 
LIBUSB_LIBS: -lusb  
SYMBOL_VISIBILITY: -fvisibility=hidden  
NOCLASS:

make

/etc/init.d/pcscd stop

/etc/init.d/pcscd stop

sudo ./src/parse > output.txt

patrick@vercors:/home/vercors/pcscd-1.3.11$ sudo ./src/parse > output.txt

Parsing USB bus/device: 001/002
idVendor: 0x0BDA iManufacturer: Generic
idProduct: 0x0151 iProduct: USB2.0-CRW
NOT a CCID/ICCD device
 Parsing USB bus/device: 001/001
idVendor: 0x1D6B iManufacturer: Linux 2.6.31-19-generic ehci_hcd
idProduct: 0x0002 iProduct: EHCI Host Controller
NOT a CCID/ICCD device
 Parsing USB bus/device: 002/004
idVendor: 0x0B81 iManufacturer: id3 Semiconductors
idProduct: 0x0200 iProduct: Contactless Reader
Found a CCID/ICCD device at interface 0
 Parsing USB bus/device: 002/001
idVendor: 0x1D6B iManufacturer: Linux 2.6.31-19-generic ohci_hcd
idProduct: 0x0001 iProduct: OHCI Host Controller
NOT a CCID/ICCD device

CL1356T5 output.txt  See Also:

• http://pcsclite.alioth.debian.org/shouldwork.html#0x0B810x0200
• http://pcsclite.alioth.debian.org/readers/id3_CL1356T5.txt
CL1356A+ output.txt  See Also:

usb.ids

idVendor: 0x0B81
iManufacturer: id3 Semiconductors
idProduct: 0x0200
iProduct: CL1356A+
bcdDevice: 11.06 (firmware release?)
bLength: 9
bDescriptorType: 4
bInterfaceNumber: 0
bAlternateSetting: 0
bNumEndpoints: 3
bulk-IN, bulk-OUT and Interrupt-IN
bInterfaceClass: 0x0B [Chip Card Interface Device Class (CCID)]
bInterfaceSubClass: 0
bInterfaceProtocol: 0
bulk transfer, optional interrupt-IN (CCID)
Can’t get iInterface string
CCID Class Descriptor
bLength: 0x36
bDescriptorType: 0x21
bcdCCID: 1.10
bMaxSlotIndex: 0x00
bVoltageSupport: 0x07
5.0V
3.0V
1.8V
dwProtocols: 0x0000 0x0003
T=0
T=1
dwDefaultClock: 3.390 MHz
dwMaximumClock: 13.560 MHz
bNumClockSupported: 0 (will use whatever is returned)
IFD does not support GET CLOCK FREQUENCIES request: Success
dwDataRate: 106000 bps
dwMaxDataRate: 848000 bps
bNumDataRatesSupported: 4
Support 106000 bps
Support 212000 bps
Support 424000 bps
Support 848000 bps
dwMaxIFSD: 254
dwSynchProtocols: 0x00000000
dwMechanical: 0x00000000
No special characteristics
dwFeatures: 0x000406FE
....02 Automatic parameter configuration based on ATR data
....04 Automatic activation of ICC on inserting
....08 Automatic ICC voltage selection
....10 Automatic ICC clock frequency change according to parameters
....20 Automatic baud rate change according to frequency and Fi, Di params
....40 Automatic parameters negotiation made by the CCID
....80 Automatic PPS made by the CCID
..02.. NAD value other than 00 accepted (T=1)
..04.. Automatic IFSD exchange as first exchange (T=1)
04.... Short and Extended APDU level exchange
dwMaxCCIDMessageLength: 65535 bytes

2.9. Drivers
bClassGetResponse: 0xFF
echoes the APDU class
bClassEnveloppe: 0xFF
echoes the APDU class
wLcdLayout: 0x0000
bPINSupport: 0x00
bMaxCCIDBusySlots: 1

libccid versions

Changes for libccid 1.4.0  See Also:

4 days ago I released a new version 1.4.0 of my CCID driver (libccid).

Changes 1.4.0 - 4 August 2010, Ludovic Rousseau:
- add support of Kingtrust Multi-Reader, Dectel CI692, Todos CX00, C3PO LTC36, ACS AET65, Broadcom 5880, Tianyu Smart Card Reader, Gemalto Hybrid Smartcard Reader
- Add support of the SCM SDI 010 again. At least the contact interface can be used.
- Use libusb-1.0 instead of libusb-0.1
- add support of TAG_IFD_STOP_POLLING_THREAD and use of the asynchronous libusb API to be able to stop a transfer.
- Request pcsc-lite 1.6.2 minimum (instead of 1.6.0) to have TAG_IFD_STOP_POLLING_THREAD defined
- The O2MICRO OZ776 patch (for OZ776, OZ776_7772, REINER_SCT and BLUDRIVEII_CCID) is no more supported with libusb-1.0
- correctly get the IFSC from the ATR (ATR parsing was not always correct)
- some minor bugs removed

The change that will interest us here is the support of TAG_IFD_STOP_POLLING_THREAD. This is used so that pcsd can ask the driver to stop its polling function.

But the support of TAG_IFD_STOP_POLLING_THREAD has only be included in pcsc-lite 1.6.2. This is why you need pcsc-lite 1.6.2 to compile libccid 1.4.0.

## Contents

- Changes for libccid 1.3.13
  - 2010-06-04
  - 2010-06-01
  - 2010-05-27
  - 2010-05-20
  - 2010-05-15
  - 2010-05-10
2010-06-04
Ludovic Rousseau
• [r4979] README, configure.in: release 1.3.13

2010-06-01 Ludovic Rousseau
• [r4975] readers/GemPCCTwin.txt: update
• [r4973] src/ccid.c, src/ccid.h, src/ccid_ifdhandler.h, src/ccid_serial.c, src/ccid_serial.h, src/ccid_usb.c, src/ccid_usb.h, src/commands.h, src/convert_version.pl, src/create_Info_plist.pl, src/debug.c, src/debug.h, src/defs.h, src/ifdhandler.c, src/parse.c, src/utils.c, src/utils.h: update copyright date
• [r4972] src/parse.c: No need to #include <usb.h> since revision 4907
• [r4971] configure.in: remove duplicate errno.h in AC_CHECK_HEADERS()
• [r4970] configure.in, m4/as-ac-expand.m4: Correctly display $bindir and $sysconfdir values when they are not changed from the default values (instead of NONE/bin and NONE/etc)

2010-05-27 Ludovic Rousseau
• [r4965] readers/Lenovo.txt: update
• [r4963] src/ccid_usb.c: OpenUSBByName(): allow the combination of USE_COMPOSITE_AS_MULTISLOT and libhal. It does work so do not prevent its use.

2010-05-20 Ludovic Rousseau
• [r4956] src/ccid.h, src/ccid_serial.c, src/ccid_usb.c, src/ifdhandler.c: Add support of SCARD_ATTR_VENDOR_IFD_SERIAL_NO

2010-05-15 Ludovic Rousseau
• [r4943] readers/GemPCPnпадv2.txt: Gemalto PC pinpad v1+

2010-05-10 Ludovic Rousseau
• [r4939] src/commands.c: Revert revision 4936 (stupid me)
• [r4938] src/ifdhandler.c: IFDHCreateChannelByName() & IFDHCreateChannel(): use the low level CmdGetSlotStatus() instead of IFDHICCPresence() to be able to fix the read timeout.
   We use a read timeout of 100 milliseconds instead of 2 seconds. The maximum wait time is now 200 milliseconds instead of 4 seconds. This increases the startup time a lot (up to 95%) when pcsd is auto started.
• [r4937] src/ccid.c, src/ccid.h, src/ccid_serial.c, src/ccid_usb.c, src/commands.c, src/defs.h, src/ifdhandler.c: change read timeout from second to millisecond unit to have a sub-second control
• [r4936] src/commands.c: SecurePINVerify() & SecurePINModify(): use min() instead of max() to get a minimum of 30 seconds for the read timeout

2.9. Drivers
Contents

- Changes for libccid 1.3.12
  - Changes
  - 2010-05-08
  - 2010-05-07
  - 2010-05-03
  - 2010-05-02
  - 2010-05-01
  - 2010-04-29
  - 2010-04-25
  - 2010-04-24
  - 2010-04-18
  - 2010-04-16
  - 2010-04-09
  - 2010-04-06
  - 2010-03-31
  - 2010-03-30
  - 2010-03-27
  - 2010-03-14
  - 2010-03-12
  - 2010-03-05
  - 2010-02-26
  - 2010-02-24
  - 2010-02-23
  - 2010-02-12
  - 2010-02-09
  - 2010-02-05
  - 2010-02-04
  - 2010-01-29
  - 2010-01-22
  - 2010-01-21
  - 2010-01-19
  - 2010-01-13
  - 2010-01-11
  - 2010-01-04
  - 2010-01-02
  - 2009-12-16
  - 2009-12-15
  - 2009-12-13
  - 2009-12-09
  - 2009-12-05
  - 2009-12-02
  - 2009-11-18
  - 2009-11-17
  - 2009-11-13
  - 2009-10-28
  - 2009-10-25
  - 2009-10-24
  - 2009-10-21
  - 2009-10-18
  - 2009-10-14
  - 2009-10-08
  - 2009-10-02
  - 2009-10-01
  - 2009-09-30

Changes for libccid 1.3.12

2.9. Drivers

- 2009-09-28
- 2009-09-25
- 2009-09-22
- 2009-09-10
- 2009-08-30
Changes

- add support of Todos AGM2 CCID, Cherry SmartTerminal XX7X, Smart SBV280, Ask CPL108, German Privacy Foundation Crypto Stick v1.2
- better support of Dell keyboard
- better support of multislot readers (like the GemCore SIM Pro)
- better support of SCM SCR3310
- The Covadis Véga-Alpha reader is a GemPC pinpad inside. So we use the same code to:
  - load the strings for the display
  - avoid limitation of the reader
- IFDHControl() the (proprietary) get firmware version escape command is allowed with a Gemalto reader:
  - the (proprietary) switch interface escape command is allowed on the Gemalto GemProx DU
  - return IFD_ERROR_NOT_SUPPORTED instead of IFD_COMMUNICATION_ERROR if the dwControlCode value is not supported
  - return IFD_ERROR_INSUFFICIENT_BUFFER when appropriate
- IFDHGetCapabilities(): add support of SCARD_ATTR_ICC_PRESENCE and SCARD_ATTR_ICC_INTERFACE_STATUS
- support extended APDU of up to 64kB with APDU readers
- get the language selected during Mac OS X installation as language to use for Covadis Véga-Alpha and Gemalto GemPC PinPad pinpad readers
- some minor bugs removed

2010-05-08  Ludovic Rousseau
- [r4932] configure.in: Minimum version of pcsc-lite is 1.6.0 instead of the unreleased 1.5.6
- [r4931] README, configure.in: release 1.3.12

2010-05-07  Ludovic Rousseau
- [r4930] MacOSX/configure: - use libusb-1.0 and libusb-compat-0.1 build for Snow Leopard
- [r4928] src/ifdhandler.c: IFDHControl(): set PCSCv2_PART10_PROPERTY_bEntryValidationCondition specific value only for the Gemalto PC Pinpad V1 & Covadis Véga-Alpha readers.
- [r4927] examples/scardcontrol.c: Reformat output
- [r4926] src/ifdhandler.c: IFDHControl(): The Covadis Véga-Alpha share the same firmware with the Gemalto PC Pinpad V1
- [r4925] src/ifdhandler.c: IFDHControl(): add comments
- [r4924] src/ifdhandler.c: IFDHControl(): add support of IOCTL_FEATURE_GET_TLV_PROPERTIES bMinPINSize & bMaxPINSize for Gemalto Pinpad V1

2010-05-03  Ludovic Rousseau
- [r4914] readers/SCR3500.txt: add SCM SCR3500 (same idProduct as SCR355 but different firmware)
2010-05-02  Ludovic Rousseau

  • [r4908] readers/Athena_IDProtect_Key.txt, readers/supported_readers.txt: add Athena IDProtect Key (unsupported see http://www.opensc-project.org/pipermail/opensc-user/2010-May/004023.html)

2010-05-01  Ludovic Rousseau

  • [r4907] src/ccid_usb.h: Do not try to be smart and always #include <usb.h> Should fix FreeBSD issues

2010-04-29  Ludovic Rousseau

  • [r4902] src/commands.c: SecurePINVerify() & SecurePINModify(): with a TPDU reader and a T=1 card the ns & nr sequence numbers were not correctly handled if the CCID command was rejected at the reader level (not sent to the card). The next APDU sent to the card would fail.

2010-04-25  Ludovic Rousseau

  • [r4897] src/ifdhandler.c: IFDHControl(): reuse ccid_descriptor variable when available

  • [r4896] src/ccid.h, src/ifdhandler.c: FEATURE_MCT_READER_DIRECT is also supported by the Kobil mIDentity visual

2010-04-24  Ludovic Rousseau

  • [r4893] readers/Kobil_Smart_Token.txt, readers/Kobil_mIDentity_4smart.txt, readers/Kobil_mIDentity_4smart_AES.txt, readers/Kobil_mIDentity_fullsize.txt, readers/Kobil_mIDentity_fullsize_AES.txt, readers/Kobil_mIDentity_visual.txt, readers/supported_readers.txt: add KOBIL Smart Token, KOBIL mIDentity 4smart, KOBIL mIDentity 4smart AES, KOBIL mIDentity visual, KOBIL mIDentity fullsize, KOBIL mIDentity 4smart fullsize AES

2010-04-18  Ludovic Rousseau

  • [r4886] src/commands.c: CmdXfrBlockCHAR_T0(): debug ICCD type A algorithm Thanks to Alexander Abarzhi for the patch

2010-04-16  Ludovic Rousseau

  • [r4879] src/ifdhandler.c: Fix Studio CC warning “ifdhandler.c”, line 1275: warning: initializer does not fit or is out of range: 248

  • [r4878] src/ifdhandler.c: Fix Sun Studio CC warnings “ifdhandler.c”, line 910: warning: initializer does not fit or is out of range: 250 “ifdhandler.c”, line 910: warning: initializer does not fit or is out of range: 255 “ifdhandler.c”, line 910: warning: initializer does not fit or is out of range: 129 “ifdhandler.c”, line 911: warning: initializer does not fit or is out of range: 128 “ifdhandler.c”, line 911: warning: initializer does not fit or is out of range: 193 “ifdhandler.c”, line 911: warning: initializer does not fit or is out of range: 192 “ifdhandler.c”, line 912: warning: initializer does not fit or is out of range: 144 “ifdhandler.c”, line 912: warning: initializer does not fit or is out of range: 177

  • [r4876] readers/SCL01x.txt: SCM SCL01x Contactless Reader

2.9. Drivers
2010-04-09 Ludovic Rousseau


2010-04-06 Ludovic Rousseau

- [r4856] readers/supported_readers.txt: Remove duplicate Vid/Pid entry for Alcor Micro SCR001 and Micro AU9520

2010-03-31 Ludovic Rousseau

- [r4849] readers/GoldKey_PIV_Token.txt, readers/supported_readers.txt: add GoldKey PIV Token

2010-03-30 Ludovic Rousseau

- [r4847] src/ccid_ifdhandler.h, src/ifdhandler.c: add support of FEATURE_GET_TLV_PROPERTIES
- [r4846] examples/scardcontrol.c: add support of FEATURE_GET_TLV_PROPERTIES

2010-03-27 Ludovic Rousseau

- [r4844] src/Info.plist.src, src/commands.c, src/ifdhandler.c: remove spaces and tabs at end of line

2010-03-14 Ludovic Rousseau

- [r4815] readers/supported_readers.txt: Enable the Broadcom 5880 reader. It should work after a firmware upgrade.

2010-03-12 Ludovic Rousseau

- [r4814] INSTALL, examples/scardcontrol.c, readers/supported_readers.txt, src/ccid_serial.c, src/reader.conf.in: Rename Gemplus in Gemalto

2010-03-05 Ludovic Rousseau

- [r4796] src/commands.c: CmdXfrBlockCHAR_T0(): limit the received length to 0x1000 for ICCD Version A length is 16-bits and usb_control_msg() fails with “Invalid argument” if the length is > 0x100 The same patch was already present in CmdXfrBlockAPDU_extended for ICCD Version B Thanks to Alexander Abarzhi for the patch
- [r4795] src/commands.c: CCID_Receive(): set the received length for a ICCD Version A device Thanks to El Tuba for the patch

2010-02-26 Ludovic Rousseau

- [r4780] readers/supported_readers.txt: SCM SDI 010 removed on manufacturer request since not supported by my driver

Chapter 2. Development
2010-02-24  Ludovic Rousseau

- [r4776] readers/supported_readers.txt: Removed Smart SBV280 on manufacturer request. They use libusb directly.
- [r4775] readers/Broadcom_5880.txt: regenerate

2010-02-23  Ludovic Rousseau

- [r4771] readers/Covadis_Auriga.txt, readers/supported_readers.txt: add Covadis Auriga

2010-02-12  Ludovic Rousseau

- [r4761] readers/id3_CL1356D.txt, readers/supported_readers.txt: id3_CL1356D.txt is a duplicate of id3_CL1356T5.txt
- [r4759] readers/id3_CL1356T5.txt, readers/supported_readers.txt: add id3 CL1356T5
- [r4757] readers/Gemalto_PDT.txt, readers/supported_readers.txt: update Gemalto PDT

2010-02-09  Ludovic Rousseau

- [r4750] src/ccid_usb.c: Fix 1 compiler warning ccid_usb.c: In function “InterruptRead”: ccid_usb.c:987: warning: pointer targets in passing argument 3 of “log_xxd” differ in signedness /usr/include/PCSC/debuglog.h:83: note: expected “const unsigned char pointer” but argument is of type “char pointer”
- [r4749] src/commands.c: Fix 2 compiler warnings commands.c: In function “CCID_Transmit”: commands.c:1107: warning: passing argument 5 of “ControlUSB” discards qualifiers from pointer target type ccid_usb.h:43: note: expected “unsigned char pointer” but argument is of type “const unsigned char pointer” commands.c:1130: warning: passing argument 5 of “ControlUSB” discards qualifiers from pointer target type ccid_usb.h:43: note: expected “unsigned char pointer” but argument is of type “const unsigned char pointer”
- [r4747] readers/Makefile.am: Do not include the reader descriptions in the archive, only the supported_readers.txt file
- [r4746] Makefile.am, configure.in: use readers/ again (revert revision 4745)
- [r4745] Makefile.am, configure.in: Do not include the readers/* files in the archive

2010-02-05  Ludovic Rousseau

- [r4712] readers/CL1356T.txt, readers/id3_CL1356T.txt: rename CL1356T.txt in id3_CL1356T.txt like the other id3_* readers

2010-02-04  Ludovic Rousseau

- [r4709] src/commands.c: add Copyright (C) 2005 Martin Paljak and update my copyright date

2010-01-29  Ludovic Rousseau

- [r4694] readers/GPFCryptoStick.txt, readers/supported_readers.txt: add German Privacy Foundation Crypto Stick v1.2

2.9. Drivers
2010-01-22  Ludovic Rousseau
  • [r4684] configure.in: use LT_INIT(disable-static) instead of the deprecated AM_DISABLE_STATIC

2010-01-21  Ludovic Rousseau
  • [r4680] configure.in: Static lib is disabled by default. Use –enable-static if needed

2010-01-19  Ludovic Rousseau
  • [r4676] readers/GemPC_Express.txt: update

2010-01-13  Ludovic Rousseau
  • [r4665] readers/Ask_CPL108.txt, readers/supported_readers.txt: ass Ask CPL108

2010-01-11  Ludovic Rousseau
  • [r4655] readers/supported_readers.txt: update Gemalto Prox-DU and Prox-SU names
  • [r4654] readers/supported_readers.txt: update “Gemalto Prox DU” name

2010-01-04  Ludovic Rousseau
  • [r4639] src/debug.c: debug.c: In function “log_msg”: debug.c:38: warning: unused parameter “priority”
  • [r4638] src/ifdhandler.c: ifdhandler.c: In function “IFDHSetCapabilities”: ifdhandler.c:553: warning: unused parameter “Length”
  • [r4637] contrib/Kobil_mIDentity_switch/Kobil_mIDentity_switch.c: Kobil_mIDentity_switch.c: In function “main”: Kobil_mIDentity_switch.c:136: warning: unused parameter “argc”
  • [r4636] src/openct/proto-t1.c: t1_release(): fix compilation warning openct/proto-t1.c:116: warning: unused parameter “t1”

2010-01-02  Ludovic Rousseau
  • [r4631] readers/Broadcom_5880v2.txt, readers/supported_readers.txt: another Broadcom 5880 reader (iProduct: 5880) which looks like to work correctly

2009-12-16  Ludovic Rousseau
  • [r4619] src/ccid_usb.c: Todos Argos Mini II with firmware before 1.01 has a bogus CCID descriptor: “Automatic IFSD exchange as first exchange (T=1)” is missing. You can’t use a T=1 card with this reader.
  • [r4618] src/ccid_usb.c: Precise Biometrics Precise 250 MC with firmware before 50.00 is bogus: time extension requests are not sent back to the host
  • [r4617] readers/Todos_AGM2_CCID.txt: firmware 1.01
2009-12-15  Ludovic Rousseau

• [r4614] examples/scardcontrol.c, src/ccid_ifdhandler.h, src/ifdhandler.c: rename FEATURE_MCT_READERDIRECT in FEATURE_MCT_READER_DIRECT

2009-12-13  Ludovic Rousseau

• [r4610] src/ccid.c: ccid_open_hack_post(): get the language selected during Mac OS X installation as language to use for Covadis Véga-Alpha and Gemalto GemPC PinPad pinpad readers

2009-12-09  Ludovic Rousseau

• [r4597] readers/Precise_250_MC.txt: firmware update
• [r4594] src/parse.c: do not generate extra space at end of line

2009-12-05  Ludovic Rousseau

• [r4592] src/ccid_usb.c: Precise Biometrics Precise 200 MC with firmware before 50.00 is bogus: time extension requests are not sent back to the host
• [r4590] readers/support readers.txt: improve documentation of bogus readers
• [r4589] readers/support readers.txt: add OCS ID-One Cosmo Card (with ProductID 0x6356) in a commented line (unsupported)
• [r4588] readers/Gemalto_HybridSmartcardReader.txt: Gemalto Hybrid Smartcard Reader

2009-12-02  Ludovic Rousseau

• [r4584] readers/Precise_200_MC.txt: new firmware

2009-11-18  Ludovic Rousseau

• [r4556] readers/Oberthur-CosmoCard1.txt: other version of the OCS ID-One Cosmo Card

2009-11-17  Ludovic Rousseau

• [r4550] src/ccid.c, src/ccid.h, src/commands.c: The Covadis Véga-Alpha reader is a GemPC pinpad inside. So we use the same code to: - load the strings for the display - avoid limitation of the reader Thanks to Loïs Lherbier for the patch

2009-11-13  Ludovic Rousseau

• [r4545] src/commands.c: CmdGetSlotStatus(): the SCM SCR3310 also reports an error 0xFE (ICC_MUTE) when no card is inserted. So extend the special case to all readers and not just the O2MICRO OZ776. Thanks to Ivan Vilata i Balaguer for the bug report (Debian bug #555837)

2009-10-28  Ludovic Rousseau

• [r4521] src/ccid_serial.c: set_ccid_descriptor(): reset dwSlotStatus to IFD_ICC_PRESENT for the other slots of a multislot reader (like a GemCore SIM Pro). This is needed because the state of dwSlotStatus may have already been changed to IFD_ICC_NOT_PRESENT (by the polling thread) when the second slot is created. The polling thread of the second slot would then never check for a card since this check is only done once. Slots are SAMs and the card is always present or absent. The problem was already dealt with on USB from the beginning but not on serial. Thanks to Emmanuel Deloget for the patch.

2.9. Drivers 223
2009-10-25  Ludovic Rousseau

- [r4510] src/defs.h: change CMD_BUF_SIZE to support extended APDU of up to 64kB. We need this size for readers in APDU mode to be able to receive the card response in one block (chaining is not always possible in this direction)
- [r4509] src/defs.h: remove useless #define
- [r4508] src/ifdhandler.c: FDHPowerICC(): use the exact length for the PowerOn output buffer
- [r4507] src/defs.h: remove useless #defines

2009-10-24  Ludovic Rousseau

- [r4505] configure.in: check for IFD_ERROR_INSUFFICIENT_BUFFER in ifdhandler.h and simplify the PCSC checking code

2009-10-21  Ludovic Rousseau

- [r4502] readers/Precise_250_MC.txt: new firmware
- [r4501] readers/Precise_200_MC.txt: new firmware

2009-10-18  Ludovic Rousseau

- [r4499] src/ifdhandler.c: FDHGetCapabilities(): add support of SCARD_ATTR_ICC_PRESENCE Required to support the Windows middleware that’s used for French Healthcare cards. Thanks to David Markowitz for the patch.
- [r4498] src/ifdhandler.c: FDHGetCapabilities(): add support of SCARD_ATTR_ICC_INTERFACE_STATUS Required to support the Windows middleware that’s used for French Healthcare cards. Thanks to David Markowitz for the patch.

2009-10-14  Ludovic Rousseau

- [r4493] readers/Smart_SBV280.txt, readers/supported_readers.txt: add Smart SBV280

2009-10-08  Ludovic Rousseau

- [r4450] src/ccid_ifdhandler.h, src/ifdhandler.c: IFDHControl(): do not check if FEATURE_IFD_PIN_PROPERTIES is defined since we now require pcsc-lite >= 1.5.6 (with FEATURE_IFD_PIN_PROPERTIES defined)
- [r4449] src/ifdhandler.c: IFDHGetCapabilities() & IFDHControl(): return IFD_ERROR_INSUFFICIENT_BUFFER when appropriate
- [r4448] configure.in, src/commands.c: Require to have pcsc-lite >= 1.5.6 to have IFD_ERROR_INSUFFICIENT_BUFFER defined in ifdhandler.h
- [r4446] src/ccid_usb.c: Use usb_strerror() instead of strerror(erno) to also get the libusb specific error messages
- [r4441] README: The supported, should work and unsupported lists are now online only. The information in the README file was not up to date and hard to sync.
2009-10-02  Ludovic Rousseau

•  [r4417] src/ifdhandler.c: revert change in revision 4414. It is a bug in the reader not the driver
•  [r4416] ylwrap: update
•  [r4414] src/ifdhandler.c: IFDHSetProtocolParameters(): with a T=1 card, do not try to negotiate IFSD if the reader works in APDU mode
•  [r4413] readers/Todos_AG2_CCID.txt: update

2009-10-01  Ludovic Rousseau

•  [r4411] src/ifdhandler.c: IFDHControl(): typo in comment

2009-09-30  Ludovic Rousseau

•  [r4410] src/ifdhandler.c: IFDHControl(): PIN_PROPERTIES_STRUCTURE structure do not have the wLcdMaxCharacters and wLcdMaxLines fields anymore. Conform with Revision 2.02.06, April 2009 of PCSCv2 part 10. Modified in pcsc-lite > 1.5.5 (revision 4378, 2009-09-09)

2009-09-28  Ludovic Rousseau

•  [r4401] src/ccid_usb.c: OpenUSBByName(): make the libhal scheme parsing more robust. Readers serial “numbers” may contain ‘_’ characters

2009-09-25  Ludovic Rousseau

•  [r4397] src/ifdhandler.c: IFDHPowerICC(): remove a useless ;

2009-09-22  Ludovic Rousseau

•  [r4392] readers/CherrySmartTerminalXX7X.txt, readers/supported_readers.txt: add Cherry SmartTerminal XX7X
•  [r4390] examples/GPL-2, examples/Makefile.am: sample code is GPLv2+
•  [r4389] src/commands.c: SecurePINVerify(): circumvent a Dell keyboard problem avoid the command rejection because the Enter key is still pressed. Wait a bit (250ms) for the (Enter) key to be released.

2009-09-10  Ludovic Rousseau

•  [r4383] src/ccid.h, src/commands.c: circumvent bugs of the Dell 413c:2100 keyboard
•  [r4382] readers/DellSK-3106.txt: regenerate
•  [r4380] src/Info.plist.src: typo in comment

2009-08-30  Ludovic Rousseau

•  [r4372] readers/Todos_AG2_CCID.txt, readers/supported_readers.txt: add Todos AG2 CCID

2.9. Drivers
2009-08-27 Ludovic Rousseau

- [r4368] SCARDCONTOL.txt: List of SCardControl() commands supported by the CCID driver
- [r4366] src/ifdhandler.c: IFDHControl(): return IFD_ERROR_NOT_SUPPORTED instead of IFD_COMMUNICATION_ERROR if the dwControlCode value is not supported

2009-07-31 Ludovic Rousseau

- [r4360] src/ccid.c: ccid_open_hack_pre(): do not call InterruptRead() on Mac OS X. The libusb does not timeout and blocks forever.
- [r4358] src/ifdhandler.c: IFDHControl(): the (proprietary) switch interface escape command is allowed on the Gemalto GemProx DU
- [r4356] src/ifdhandler.c: IFDHControl(): the (proprietary) get firmware version escape command is allowed with a Gemalto reader
- [r4355] src/ccid.h: add GET_VENDOR macro

Windows USB ccid driver

![Figure 2.10: usbccid windows driver](image.png)

usbcid driver windows installation

Windows usbcid driver

See Also:

CCID driver manager (id3 Semiconductors)

Contents

- Windows usbcid driver
  - windows usbcid driver
  - Old installation

windows usbcid driver

Contents

- Windows x86 USB ccid driver
  - Introduction
  - Certis Bio Driver example
  - Installation of the x86 usbcid driver under Vista and Windows 7
  - Installation of the x86 usbcid driver under Windows XP
Introduction  La famille x86 regroupe les microprocesseurs compatibles avec le jeu d’instructions de l’Intel 8086. Cette série est nommée IA-32 (pour Intel architecture 32 bits) par Intel pour ses processeurs à partir du Pentium. Un constructeur de microprocesseur pour PC doit maintenir une compatibilité ascendante avec ce jeu d’instruction s’il veut que les logiciels déjà écrits fonctionnent sur les nouveaux microprocesseurs.

See Also:

• http://fr.wikipedia.org/wiki/X86
• x86

Certis Bio Driver example

Installation of the x86 usbccid driver under Vista and Windows 7

dpinst32.exe /q /f /path .

After installation, the id3usbccid.sys is under the “%systemroot%\system32\drivers” directory.

See Also:

File System Redirector

Installation of the x86 usbccid driver under Windows XP

DPInst.exe
DPInst_32bits_usbccid_driver.bat
install_usbccid_driver.bat
usbccid.cat
usbccid.inf
usbccid.sys

DPInst.exe -q -f /path .

After installation, the usbccid.sys is under the “%systemroot%\system32\drivers” directory.

2.9. Drivers 227
See Also:

DPInst Command-Line Switches

See Also:

File System Redirector

## Contents

### Windows x64 USB ccid driver

- Installation of the x64 usbccid driver under Vista and Windows 7
- Installation of the x64 usbccid driver under Windows XP

#### Installation of the x64 usbccid driver under Vista and Windows 7

dpinst64.exe /q /f /path .

See Also:

- File System Redirector
- x86

#### Installation of the x64 usbccid driver under Windows XP

DPInst.exe /q /f /path .

See Also:

DPInst Command-Line Switches

See Also:
File System Redirector

Old installation

Old Windows x86 USB ccid driver installation

Introduction

**usbccid.sys Driver**  The list of files

usbccid.cat
usbccid.inf
usbccid.sys
id3Semiconductors.p7b (for Windows Vista and Windows 7)

Installation under Windows XP, Vista and Windows 7

cd Program Files/Orcanthus/CL1356A/Driver/{XP, Vista}
DPInst.exe -q /F /path .

After installation, the usbccid.sys is under the “c:\WINDOWS\system32\drivers” directory.

Installation under Windows Vista and Windows 7  For Vista and Win7 there is a certificate file: id3Semiconductors.p7b.

    cd Program Files/Orcanthus/CL1356A/Driver/Vista
    certmgr.exe -add -c id3SemiConductors.p7b -s -r localmachine trustedpublisher

*id3Semiconductors.p7b*  An id3Semiconductors certificate file trusted by VeriSign Trust Network.

**usbccid.inf**

```
[Version]
Signature="$Windows NT$"
Class=SmartCardReader
ClassGuid={50DD5230-BA8A-11D1-BF5D-0000F805F530}
Provider=%MS%
DriverVer=08/01/2006,5.2.3790.2724
CatalogFile=usbccid.cat

[SourceDisksNames]
l=%Distribution%,usbccid.sys,,

[SourceDisksFiles]
usbccid.sys = 1

[ClassInstall32]
Addreg=SmartCardReaderClassReg
```

2.9. Drivers
[ClassInstall]
Addreg=SmartCardReaderClassReg

[SmartCardReaderClassReg]
HKR,,0,%ClassName%
HKR,,Icon,,25
HKR,,NoInstallClass,,1
HKR,,Installer32,,"winscard.dll,ClassInstall32"

[Manufacturer]
%CCID%=CCID

; =========== Add reg for all readers =============

[Reader.Install.AddReg]
HKLM,Software\Microsoft\Cryptography\Calais\Readers,",
HKLM, System\CurrentControlSet\Services\SCardSvr,Start,0x00010001,2
HKLM, System\CurrentControlSet\Services\CertPropSvc,Start,0x00010001,2

; ***********************
; USB CCID Compliant Readers
; ***********************

[CCID]
%USBCCID.DeviceDesc% = USBCCID.Install,USB\Class_0B&SubClass_00

[USBCCID.Install.NT]
AddReg = Reader.Install.AddReg
CopyFiles = USBCCID.Install.CopyFiles

[USBCCID.Install.NT.HW]
AddReg = USBCCID.Install.AddReg.HW

[USBCCID.Install.AddReg.HW]
HKR,,"VendorName",,"Generic"
HKR,,"IfdType",,"USB Smartcard"

[USBCCID.Install.CopyFiles]
usbcid.sys,,2

[USBCCID.Install.NT.Services]
AddService = USBCCID, 2, USBCCID.Service

[USBCCID.Service]
DisplayName = %USBCCID.DeviceDesc%
ServiceType = 1 ;%SERVICE_KERNEL_DRIVER%
StartType = 3 ;%SERVICE_AUTO_START%
ErrorControl = 1 ;%SERVICE_ERROR_NORMAL%
ServiceBinary = %12%\usbcid.sys

[DestinationDirs]
USBCCID.Install.CopyFiles = 12

[Strings]
MS = "Microsoft"
Distribution = "Windows USB CCID Driver Disk"
ClassName = "Smart card readers"
CCID = "USB CCID Compliant"
USBCCID.DeviceDesc = "USB Smart Card reader"
HKLM\SYSTEM\CurrentControlSet\Enum\USB  See Also:

devcon (mt)

“Device Parameters” key

EscapeCommandEnable value  See Also:

http://www.microsoft.com/whdc/device/input/smartcard/USB_CCID.mspx

In order to send or receive an Escape command to a reader, the DWORD registry value EscapeCommandEnable must be added and set to a non-zero value under the HKLM\SYSTEM\CurrentControlSet\Enum\USB\Vid*Pid*\Device Parameters key.

Then the vendor IOCTL for the Escape command is defined as follows:

#define IOCTL_CCID_ESCAPE SCARD_CTL_CODE(3500).

With the enabled Escape Command, security against malicious escape commands becomes the reader’s responsibility.

Dpinst.exe  source:


**DPIInst Command-Line Switches**  DPIInst supports the following command-line switches that set the DPIInst configuration flags that are described in Setting DPIInst Configuration Flags and additional switches that further modify DPIInst operation:

/a   Sets the installAllOrNone flag to ON, which configures DPIInst to install drivers in a driver package only if all of the drivers in the installation package can be installed.

/c   Configures DPIInst, if DPIInst is called at a command prompt, to display the text of log messages in the Command Prompt window.

/d   Sets the deleteBinaries flag to ON, which configures DPIInst to delete the binary files that were copied to a system when a driver package was installed.

Note: Starting with Windows 7, the /d switch is ignored by the operating system. Binary files, which were copied to a system when a driver package was installed, can no longer be deleted by using DPIInst.

/el   Sets the enableNotListedLanguages flag to ON, which configures DPIInst to enable all of the supported languages that are not explicitly enabled by language XML elements in a DPIInst descriptor file.

/f   Sets the forceIfDriverIsNotBetter flag to ON, which configures DPIInst to install a driver on a device even if the driver that is currently installed on the device is a better match than the new driver.

/h, /?, or /help  Configures DPIInst to display help information about DPIInst command-line options. If DPIInst was called at a command prompt, DPIInst displays the help text in the Command Prompt window; otherwise, DPIInst displays the help text in a message box.

2.9. Drivers  231
Confirms DPInst to display the text, bitmaps, and icon for the language that is specified by a language-ID value. The language-ID value can be specified in either hexadecimal or decimal format. For a list of the languages and corresponding language identifiers, see DPInst Localization Support.

If DPInst supports the specified language and runs on a Windows operating system in which the language is installed, DPInst will use the specified language. For more information about how to use the /L switch, see Testing Language Customization.

Sets the legacyMode flag to ON. In legacy mode, DPInst accepts unsigned driver packages without performing signature verification. However, be aware that Windows driver signing requirements apply to the installation of a driver package from the DIFx driver store. These requirements depend on the Windows version, the signature type, the driver package type, and the driver signing options that are set for a computer.

Sets the quietInstall flag to ON, which suppresses the display of wizard pages, user dialog boxes, and other user messages that DPInst and Windows generate. The quietInstall flag works with the presence of an end-user license agreement (EULA) page and the suppressEulaPage flag.

Sets the promptIfDriverIsNotBetter flag to ON, which configures DPInst to display a user dialog box if a new driver is not a better match to a device than a driver that is currently installed on the device. The user dialog box informs a user of this situation and provides an option to replace the driver that is currently installed on the device with the new driver. see /sh flag

Sets the DPInst working directory to DPInst-working-directory. DPInst searches for driver packages in the DPInst working directory and in subdirectories under the working directory that are specified by subDirectory XML elements in a DPInst descriptor file. For more information about how to set the DPInst working directory and specifying subdirectories under the working directory, see Specifying the Location of a Driver Package.

Sets the suppressAddRemovePrograms flag to ON, which configures DPInst to suppress the addition of entries to Programs and Features in Control Panel. These entries represent the driver packages and driver package groups that DPInst installs.

In versions of Windows earlier than Windows Vista, DPInst added the entry for the driver package or driver package group to Add or Remove Programs in Control Panel.

Sets the suppressEulaPage flag to ON, which configures DPInst to suppress the display of a EULA page. The suppressEulaPage flag works with the quiet Install flag and the suppress Wizard flag.

Sets the scanHardware flag to ON, which configures DPInst to install a driver package for a Plug and Play (PnP) function driver only if the driver package matches a device that is currently configured in a computer and if the driver package is a better match for the device than the driver package that is currently installed on the device.

Sets the suppressWizard flag to ON, which suppresses the display of wizard pages and other user messages that DPInst generates. The suppressWizard flag works with the presence of a EULA page and the suppressEulaPage flag.

Configures DPInst to uninstall a driver package whose INF file is specified by inf-file-path. The path that is specified by inf-file-path is relative to the directory
that contains DPInst.exe.

**DPInst calling examples**  The following command includes the /q and /se command-line switches, which configure DPInst to operate in quiet-install mode, whether a eula XML element is included in a DPInst descriptor file. This command suppresses all of the wizard pages, user dialog boxes, and other user messages that DPInst and Windows generate.

```
dpinst.exe /q /se
```

The following command includes the /q command-line switch, which configure DPInst to operate in quiet-install mode. However, DPInst will operate in quiet-install mode only if a eula XML element is not included in a DPInst descriptor file. If a eula element is included, DPInst does not operate in quiet-install mode.

```
dpinst.exe /q
```

The following command includes a /se command-line switch, which suppresses the display of only the EULA page.

```
dpinst.exe /se
```

The following command includes the /q and /se command-line switches, which configure DPInst to operate in quiet-install mode, including suppressing the EULA page. The command also includes the /f command-line switch, which forces the installation of a new driver package for a device that is configured in a computer, even if the driver package that is currently installed on the device is a better match for the device than the new driver package.

```
dpinst.exe /q /se /f
```

The following command includes the /sh command-line switch, which configures DPInst to install a driver package on a device that is configured in a computer only if the new driver package is a better match for the device than the driver package that is currently installed on the device. This command facilitates using an installation package to batch install driver packages for devices that are configured in a computer. Although the installation package might contain a large number of driver packages, DPInst will install only driver packages that are better matches for the devices than the driver packages that are currently installed on the devices.

```
dpinst.exe /sh
```

The following command includes the /u command-line switch, followed by the Abc.inf INF file, and the /d command-line switch. The /u command-line switch uninstalls the driver package that is associated with the Abc.inf INF file from all of the devices on which the package is installed. The /d command-line switch deletes the binaries that were copied to the system when the package was installed and deletes the package from the DIFx driver store. If the driver package is signed, the corresponding catalog file must be present in the same directory in which Abc.inf is located.

```
dpinst.exe /u abc.inf /d
```

**CL1356 DPInst calling examples**
Display a user dialog box if a new driver is not a better...

DPInst.exe -p /path .

See Also:

- DPInst Command-Line Switches
- DPInst calling examples

Computer/Smart Card (PC/SC)

Personal Computer/Smart Card (PC/SC)  See Also:

- smartcards
- smartcard tools
The PC/SC specification defines how to integrate smart card readers and smart cards with the computing environment and how to allow multiple applications to share smart card devices.

Personal computer/Smart Card (PC/SC) is a specification for SmartCard integration in computing environment. PC/SC is implemented in Microsoft Windows 200x/XP and available under Microsoft Windows NT/9x. A free implementation of PC/SC, PC/SC Lite, is available under Linux and bundled with Mac OS X.


PC/SC Linux

libpcsclite  
See Also:

- Ludovic Rousseau

Contents

- libpcsclite
  - Introduction
  - Documentation
  - PC/SC API spy
  - pcsclite installation
  - Versions
  - Debug
  - pcscl and DBUS

Introduction  
Middleware to access a smart card using SCard API (PC/SC).

Source code available from http://svn.debian.org/wsvn/pcsclite/trunk/PCSC/

- http://ludovicrousseau.blogspot.com/
- http://pcsclite.alioth.debian.org/
- Project page
- Download
- Subversion repository:
  - viewsvn
  - wsvn
  - svn co svn://svn.debian.org/pcsclite/trunk/PCSC

Ludovic Rousseau’s blog about PC/SC and smart cards

Documentation

- PC/SC Lite API (WinSCard)
- IFD Handler API v3.0
- PC/SC internals documented with Doxygen

PC/SC API spy
PC/SC API spy  See Also:


pcsclite installation

pcsclite installation  See Also:

PCSC Lite

pcsclite installation on Debian/ubuntu

sudo aptitude install libpcsclite-dev

pcsclite installation on linux

libpcsclite installation  See Also:

- PCSC Lite

Prerequisite  See Also:

- pcscd now uses libudev instead of libhal
- libudev

sudo aptitude install libudev-dev

Les NOUVEAUX paquets suivants vont être installés :

libudev-dev

0 paquets mis à jour, 1 nouvellement installés, 0 à enlever et 20 non mis à jour.
Il est nécessaire de télécharger 153ko d’archives. Après dépaquetage, 504ko seront utilisés.
153ko téléchargés en 1s (153ko/s)
 Sélection du paquet libudev-dev précédemment désélectionné.
 (Lecture de la base de données... 251274 fichiers et répertoires déjà installés.)
Dépaquetage de libudev-dev (à partir de .../libudev-dev_162-2.2_i386.deb) ...
Paramétrage de libudev-dev (162-2.2) ...

Package installation

on fedora

yum install libpcsclite-dev
Manual installation

Warning: libusb must be installed before libpcsclite.

bunzip2 pcsc-lite-x.y.z.tar.bz2

tar xvf pcsc-lite-x.y.z.tar

cd pcsc-lite-x.y.z

chmod +x configure

./configure --disable-libhal LIBUSB_LIBS="-L/opt/libusb/current/lib -lusb-1.0" --prefix=/opt/pcsclite

make

Exemple for the last version 1.x.y

The last version of libpcsclite is 1.7.4 (june 23, 2011)

cd pcsc-lite-1.x.y

chmod +x configure

./configure --disable-libhal LIBUSB_LIBS="-L/opt/libusb/current -lusb-1.0" --prefix=/opt/pcsclite/1.x.y

make

PC/SC lite has been configured with following options:

Version: 1.x.y
System binaries: /opt/pcsclite/1.x.y /sbin
Configuration dir: /opt/pcsclite/1.x.y /etc/reader.conf.d
Host: i686-pc-linux-gnu
Compiler: gcc
Preprocessor flags: -I$(top_srcdir)/src
Compiler flags: -Wall -fno-common -g -O2
Preprocessor flags: -I$(top_srcdir)/src
Linker flags:
Libraries:

PTHREAD_CFLAGS: -pthread
PTHREAD_LIBS: Linux

pcscd binary /opt/pcsclite/1.x.y /sbin/pcscd
libudev support: yes
libusb support: no
USB drop directory: /opt/pcsclite/1.x.y /lib/pcsc/drivers
ATR parsing messages: false
ipcdir: /var/run/pcscd
use serial: yes
use usb: yes

make

make install

Libraries have been installed in: /opt/pcsclite/1.x.y /lib

If you ever happen to want to link against installed libraries in a given directory, LIBDIR, you must either use libtool, and specify the full pathname of the library, or use the ‘-LLIBDIR’ flag during linking and do at least one of the following:

- add LIBDIR to the ‘LD_LIBRARY_PATH’ environment variable during execution

2.9. Drivers
- add LIBDIR to the 'LD_RUN_PATH' environment variable during linking
- use the '-Wl,-rpath -Wl,LIBDIR' linker flag
- have your system administrator add LIBDIR to '/etc/ld.so.conf'

After make install In /opt/pcsclite/1.x.y

```
-- include
  | -- PCSC
  |   -- debuglog.h
  |   -- ifdhandler.h
  |   -- pcsclite.h
  |   -- reader.h
  |   -- winscard.h
  |   -- wintypes.h
-- lib
  |   -- libpcsclite.la
  |   -- libpcsclite.so -> libpcsclite.so.1.0.0
  |   -- libpcsclite.so.1 -> libpcsclite.so.1.0.0
  |   -- libpcsclite.so.1.0.0
  |   -- pkgconfig
  |   -- libpcsclite.pc
-- sbin
  | -- pcscd
-- share
  -- doc
    | -- pcsc-lite
    |   -- README.DAEMON
  -- man
    -- man5
    | -- reader.conf.5
    -- man8
      -- pcscd.8
```

Link to the the current libusb version

cd /opt/pcsclite/1.x.y
ln -s 1.x.y current
ls -als

drwxr-xr-x 3 root root 4096 2011-04-07 13:56 ./
drwxr-xr-x 5 root root 4096 2011-04-07 13:54 ../
drwxr-xr-x 6 root root 4096 2011-04-07 13:54 1.x.y /
lrwxrwxrwx 1 root root 5 2011-04-07 13:56 current -> 1.x.y /

The pcscd daemon

pcscd help

cd /opt/pcsclite/current/sbin
sudo pcscd -h
Usage: ./pcscd options

Options:
- a, --apdu log APDU commands and results
- c, --config path to reader.conf
- f, --foreground run in foreground (no daemon),
  send logs to stderr instead of syslog
- h, --help display usage information
- H, --hotplug ask the daemon to rescan the available readers
- v, --version display the program version number
- d, --debug display lower level debug messages
--info display info level debug messages (default level)
- e --error display error level debug messages
- C --critical display critical only level debug messages
--force-reader-polling ignore the IFD_GENERATE_HOTPLUG reader capability
- t, --max-thread maximum number of threads (default 200)
- s, --max-card-handle-per-thread maximum number of card handle per thread
  (default: 200)
- r, --max-card-handle-per-reader maximum number of card handle per reader
  (default: 200)

libpcsclite installation on centos

yum install libpcsclite
yum install libccid
yum install libusbutils

Versions

libpcsclite versions See Also:

- PCSC Lite
- https://alioth.debian.org/frs/?group_id=30105&release_id=1756

libpcsclite 2.0.0

Changes for pcsc-lite pcsc-lite-2.0.0 BETA: Ludovic Rousseau

10 Feb 2010

- redesign the client/server communication:
  - no more shared memory used (allow pcscd and libpcsclite1.so to be on different computer and talk
    over a network)
  - no more difference between short and extended APDU
  - no more use of a /var/run/pcscd/pcscd.events/ directory. events are sent through the socket
  - simpler command format between client and server The side effect is that you are not able to mix an old
    pcscd with a new libpcsclite1.so or the reverse. SCardEstablishContext() will fail unless you update both
    sides of the communication.
• **Use lists instead of fixed size arrays to store handles.** It is now possible to have:
  – 200 simultaneous PC/SC clients instead of 16
  – 200 SCardConnect per client instead of 16
  – 200 clients per reader instead of 16

  The default value of 200 can be changed by giving an argument to pcsd --max-thread --max-card-handle-per-thread --max-card-handle-per-reader

  Thanks to Jean-Luc Giraud for the big patch

• Make SCardReconnect(), SCardStatus() and SCardTransmit() block instead of returning SCARD_E_SHARING_VIOLATION immediately. These functions will then behave like on Windows. This can happen if these functions are called when the reader is locked by a PCSC transaction(SCardBeginTransaction/SCardEndTransaction). You can define the environment variable PCSCLITE_NO_BLOCKING to use the old behavior. Thanks to Jean-Luc Giraud for the patch.


• SCardEstablishContext(): try to start the pcsd daemon if not already running.
  – pcsd will suicide itself after 60 seconds of inactivity if it is started using --auto-exit. This is the default behavior when pcsd is started by libpcsclite
  – Set PCSCLITE_PCIEARGS with the argument you want to pass to pcsd in autostart Only one argument is passed. The space character is not a separator. example: export PCSCLITE_PCIEARGS=-df

• SCardListReaders(): can use SCARD_AUTOALLOCATE

• SCardGetAttrib(): return SCARD_E_INSUFFICIENT_BUFFER if the driver returns IFD_ERROR_INSUFFICIENT_BUFFER
  – add support of SCARD_ATTR_DEVICE_FRIENDLY_NAME as it is better implemented in pcsd (it knows the friendly name)

• SCardGetStatusChange(): Calling with cReaders == 0 will now just return SCARD_S_SUCCESS
  – Use the special reader name “\?PnP\?Notification” to wait for a reader event notification

• SCardTransmit(): do not limit the minimum size of an APDU to 4 bytes. non ISO 7816-4 compliant cards (like Mifare DESFIRE) may use shorter commands

• SCardStatus(): returns SCARD_E_SHARING_VIOLATION if the reader is already used More conform to Windows

• PCSC/reader.h: update sruct PIN_PROPERTIES_STRUCTURE to be conform with Revision 2.02.06, April 2009 of PCSCv2 part 10 Fields wLcdMaxCharacters and wLcdMaxLines have been removed
  – rename FEATURE_MCT_READERDIRECT in FEATURE_MCT_READER_DIRECT to be conform with ch. 2.3 of PCSC v2 part 10

• SCardControl() return SCARD_E_UNSUPPORTED_FEATURE if the driver returned IFD_ERROR_NOT_SUPPORTED or IFD_NOT_SUPPORTED This is used to separate an unsupported value of ControlCode from a general error

• Use the standard --sysconfdir=DIR ($prefix/etc by default) instead of --enable-confdir=DIR for defining the directory containing reader.conf

• remove SCF support (PC/SC over Smart Card Framework). I never used this feature and SCF is now dead and replaced by JSR 268 (javax.smartcardio)

• Better handling of PCSCLITE_STATIC_DRIVER as can be used on platforms using Clinux (without dynamic loader). This is used to statically link the reader driver to pcsd. Since the link is static you must define the IFDHandler API version at compilation time. Either define IFDHANDLERv1, IFDHANDLERv2 or IFDHANDLERv3
• Use dynamic instead of static allocation for the driver library filename. The filename is no more limited to 100 characters. Closes: [#312332] MAX_LIBNAME too short?
• force the return codes SCARD_* to be long since the SCard* functions return a LONG type
• some other minor improvements and bug corrections
– Dr. Ludovic Rousseau

libpcsclite 1.8.2 (janvier 2012) See Also:
• http://ludovicrousseau.blogspot.com/2012/01/new-version-of-pcsc-lite-182.html
I just released a new version of pcsc-lite 1.8.2. No big changes expect for pcsc-spy I talked about in a previous article PCSC API spy, third try.

Changelog 18 January 2012
• rename pcsc-spy.py to pcsc-spy and install it as a normal binary (in /usr/local/bin by default)
• write a pcsc-spy.1 manpage
• fix a bug with a multi-slot reader
• Info.plist parser: avoid a buffer read overflow in management
• Some Doxygen improvements

libpcsclite 1.8.0 See Also:

Introduction

I just released a new version of pcsc-lite 1.8.0.
As always, please report any issue on this mailing list or using the project bugtracker.

Bye

Changes, pcsc-lite-1.8.0: Ludovic Rousseau 19 November 2011
• PC/SC spy tool
• Support systemd socket activation (the auto start of pcsd from the library has been removed. Use systemd instead)
• SCardGetStatusChange(): check all the readers are already known and return SCARD_E_UNKNOWN_READER if a reader name is not present. Windows XP has this behavior.
• SCardEstablishContext(): Invalidate all the handles in the son after a fork
• Add define of FEATURE_EXECUTE_PACE from PCSC v2 Part 10 Amendment 1 2011-06-03
• Fix some memory leaks repoted by Coverity
• Enable silent build by default
• log_line(): correctly calculate delta time when no color is used The update of last_time was only done in case of colorization (LogDoColor). So on unsupported consoles the time was wrong.
• log_xxd_always(): Use a variable-length array The debug message buffer is no more with a fixed size (around 600 bytes of buffer to log) but uses a variable-length array. It is now possible to log extended APDU of 64kB. The variable-length array feature is available in GCC in C90 mode and is mandatory in C99 standard.

• Some other minor improvements and bug corrections

> On Sat, Nov 19, 2011 at 5:20 PM, Ludovic Rousseau
> <ludovic.rousseau@gmail.com> wrote:
>> - PC/SC spy tool
>>
>> Is there more information available somewhere?
>> The only difference I got when building 1.8.0 was that there is a new
>> libpcscspy. But I’m not sure how it can be used.

You have a README file in src/spy/

More information on PCSC API spy, third try . I added the link in my blog article about pcsc-lite version 1.8.0. But forgot to also add it in the email to muscle. Sorry.

Bye

libpcsclite 1.7.3  See Also:


Changelog  I just released new version of pcsc-lite 1.7.3.

• COPYING: Add my name as copyright holder
• hotplug libudev: support libudev >= 171
• hotplug libusb: Fix a memory leak
• pcsd: exit immediately in case of SIGTERM Closes Debian bug #620305 “pcsclite slows down shutdown/restart”
• Send logs to stdout instead of stderr It is now possible to use tee(1) to redirect logs in a file without first redirecting stderr to stdout
• Add command line option -T, –color: force use of colored logs The idea is to have colored logs even if they are redirected to a file or a pipe.
• Define g_rgSCardT?Pci as const structures to be more Windows like I do not expect a regression or compilation problem in WinSCard API users but how knows...
• log at level PCSC_LOG_DEBUG instead of PCSC_LOG_ERROR to avoid filling the system log file
• Remove the deprecated define FEATURE_MCT_READERDIRECT (replaced by FEATURE_MCT_READER_DIRECT)
• better Hurd support

some other minor improvements and bug corrections

libpcsclite 1.7.1
Hello,

I just released a new version of pcsc-lite 1.7.1.

**Changelog**  pcsc-lite-1.7.1: Ludovic Rousseau, 30 March 2011

- use libudev only on Linux and libusb elsewhere. The configuration now works by default on GNU/kFreeBSD systems
- Try to use a (CCID) class driver if a specific driver fails to use the reader.
- fix a potential crash

https://alioth.debian.org/frs/?group_id=30105&release_id=1666#pcsclite-pcsc-lite-1-7-1-title-content

Bye

**libpcsclite 1.6.2**  The libpcsclite 1.6.2 version is an important version which is adapted to the *libccid 1.4.0 version*


The same day of the libcid 1.4.0 release I also released pcsc-lite 1.6.2.

Changes pcsc-lite-1.6.2: Ludovic Rousseau, 4 August 2010:

- implement a “Forced suicide” mechanism. After 3 Ctrl-C without much reaction from pcscd (in fact the drivers) we force the suicide. Sometimes libusb is blocked in a kind of dead-lock and kill -9 was the only option.
- Add support of TAG_IFD_STOP_POLLING_THREAD to request the stop of the driver polling function.
- Avoid a division by 0. Closes [#312555] “simclist bug in pcsc-lite”
- if pcscd is started by libpcsclite then close all file handles except stdin, stdout and stderr so that pcscd does not confiscate resources allocated by the application
- in case of auto exit create a new session so that Ctrl-C on the application will not also quit pcscd
- src/hotplug_libusb.c: port from libusb-0.1 to libusb-1.0
- default configuration is now $sysconfdir/reader.conf.d
- fix crash with empty config dir
- src/PCSC/winscard.h: Remove definitions of SCARD_READERSTATE_A, PSCARD_READERSTATE_A and LPSCARD_READERSTATE_A types
- some other minor improvements and bug corrections

**Debug**

**pcsclite debug**

Executive summary

1. Copy the file ltrace.conf to ~/.ltrace.conf
2. Use with: ltrace -l/usr/lib/libpcsclite.so your_application
   • Call to other libraries (lib C in particular) are no more displayed
   • PCSC functions arguments are displayed in a human form

If I reuse the same SCardConnect example we now have:

SCardConnect(0x103d6cb, "Lenovo Integrated Smart Card Rea"..., SCARD_SHARE_SHARED, SCARD_PROTOCOL_T0_SCARD_PROTOCOL_T1, 126261, SCARD_PROTOCOL_T1) = SCARD_S_SUCCESS

We get:

• Connection context to the PC/SC Resource Manager. It is just a random numeric value: 0x103d6cb
• Reader name to connect to: “Lenovo Integrated Smart Card Rea”...
• Mode of connection type: exclusive or shared. SCARD_SHARE_SHARED
• Desired protocol use: SCARD_PROTOCOL_T0_SCARD_PROTOCOL_T1
• Handle to this connection: 126261
• Established protocol to this connection: SCARD_PROTOCOL_T1
• Return code: SCARD_S_SUCCESS

It is easy to use ltrace to trace a program. I will use the testpcsc program included in pcsc-lite as a PC/SC sample code:

$ ltrace .libs/testpcsc

Limitations By default ltrace lists all the calls from all the libraries. You can restrict the tracing to one specific library using:

-l/usr/lib/libpcsclite.so

The PC/SC API calls are not really useful. You have to parse the argument by hand.

It is not easy to know what SCardConnect(0x103a3a0, 0x96ea118, 2, 3, 0xbfdfacdc) = 0 is really doing.

ltrace PCSC configuration file ltrace can use a configuration file to parse the arguments of the functions and display a (more) human readable version.

The configuration file uses a very simple format. For example for SCardConnect I used:

scarderror SCardConnect(scardcontext, string, share_mode, protocol, +scardhandle*, +protocol*);

A ltrace configuration file is now included in the pcsc-lite project in the new contrib/ directory. Just copy the ltrace.conf to ~/.ltrace.conf.

./pcscd -a

DO_PROFILE define in src/winscard_clnt.c To have more logs on the client side you have to edit src/winscard_clnt.c and #define DO_PROFILE. Then rebuild and reinstall pcsc-lite.
pcsc and DBUS

#!/usr/bin/env python

""
pcsc-watcher.py
(c) 2011 by Florian "floe" Echtler <floe@butterbrot.org>
""

import gobject
import dbus
import dbus.service
from dbus.mainloop.glib import DBusGMainLoop
from smartcard.CardMonitoring import *
from smartcard.util import *

# dbus stuff

service = None
interface = "org.debian.alioth.pcsclite" # also used as bus name
path = "/org/debian/alioth/pcsclite/reader0/slot0"

# very simple DBus service class with one signal
class PCSC_DBus_Service(dbus.service.Object):
    def __init__(self,object_path):
        bus_name = dbus.service.BusName(interface, bus=dbus.SessionBus())
        dbus.service.Object.__init__(self, bus_name, object_path)

    @dbus.service.signal(interface)
    def CardPresenceChanged(self, atr, added):
        pass

# smartcard stuff

# simple card observer class which calls the DBus service
class PCSC_Card_Observer(CardObserver):
    def update(self, observable, (addedcards, removedcards)):
        for card in addedcards:
            service.CardPresenceChanged(toHexString(card.atr),True)
        for card in removedcards:
            service.CardPresenceChanged(toHexString(card.atr),False)

# main

DBusGMainLoop(set_as_default=True)
gobject.threads_init() # very important to avoid starving the PCSC thread

service = PCSC_DBus_Service(path)

cardmonitor = CardMonitor()
cardobserver = PCSC_Card_Observer()
Development tools, Release 2012.06.18

cardmonitor.addObserver(cardobserver)

gobject.MainLoop().run()

 pcs muscle  Le projet MUSCLE ou la bibliothèque PC/SC Lite.

See Also:
GNU/Linux magazine N° 39, p.18 de Ludovic Rousseau et David Corcoran.

ATR  Windows identifie une carte en fonction de son ATR (Answer to Reset).
C’est avec l’ATR qu’une carte est entrée dans la base (de registres) avec ScardIntroduceCardType().
Le problème est que de nos jours une carte à puce est une plateforme (JavaCard, .NET ou même MULTOS) sur laquelle il est possible de charger plusieurs applications.

See Also:
http://smartcard-atr.appspot.com/
The parsing code is part of pycard and is available at parseATR.py The list of known ATR is also available online at smartcard_list.txt.

MUSCLE news

request for a WinSCard logger

from  Ludovic Rousseau <ludovic.rousseau@gmail.com> 
reply-to  MUSCLE <muscle@lists.musclecard.com> 
to  MUSCLE <muscle@lists.musclecard.com> 
date  Fri, Apr 9, 2010 at 11:29 AM 
subject  Re: [Muscle] request for a WinSCard logger

WinSCard APDU View Utility  After more research I found:

• WinSCard APDU View Utility http://www.fernandes.org/apduview/index.html It is GPL but for Windows and only logs SCardTransmit

ScardSpy

• ScardSpy http://www.idrix.fr/Root/content/category/7/25/48/ It is proprietary and for Windows but logs other winscard functions (but not all)

PC/SC windows

Smart Cards Windows news
What’s New in Smart Cards


Updated: February 9, 2009

Applies To: Windows 7, Windows Server 2008 R2

Windows® 7 includes new features that make smart cards easier to use and to deploy, and makes it possible to use smart cards to complete a greater variety of tasks.

The new smart card features are available in all versions of Windows 7.

What’s new in smart cards? Windows 7 features enhanced support for smart card–related Plug and Play and the Personal Identity Verification (PIV) standard from the National Institute of Standards and Technology (NIST).

This means that users of Windows 7 can use smart cards from vendors who have published their drivers through Windows Update without needing special middleware.

These drivers are downloaded in the same way as drivers for other devices in Windows.

When a PIV-compliant smart card is inserted into a smart card reader, Windows attempts to download the driver from Windows Update. If an appropriate driver is not available from Windows Update, a PIV-compliant minidriver that is included with Windows 7 is used for the card.

Who will want to use smart cards? Network administrators who want to enhance the security of the organization’s computers, particularly portable computers used by remote users, will appreciate the simplified deployment and use scenarios made possible by smart card Plug and Play PIV support.

Users will appreciate the ability to use smart cards to perform critical business tasks in a secure manner.

What are the benefits of the new and changed features? The new smart card support options in Windows 7 include:

- Encrypting drives with BitLocker Drive Encryption. In the Windows 7 Enterprise and Windows 7 Ultimate operating systems, users can choose to encrypt their removable media by turning on BitLocker and then choosing the smart card option to unlock the drive. At run time, Windows retrieves the correct minidriver for the smart card and allows the operation to complete.

- Smart card domain logon by using the PKINIT protocol. In Windows 7, the correct minidriver for a smart card is retrieved automatically, enabling a new smart card to authenticate to the domain without requiring the user to install or configure additional middleware.

- Document and e-mail signing. Windows 7 users can rely on Windows to retrieve the correct minidriver for a smart card at run time to sign an e-mail or document. In addition, XML Paper Specification (XPS) documents can be signed without the need for additional software.

- Use with line-of-business applications. In Windows 7, any application that uses Cryptography Next Generation (CNG) or CryptoAPI to enable the application to use certificates can rely on Windows to retrieve the correct minidriver for a smart card at run time so that no additional middleware is needed.

What’s the impact of these changes on smart card usage? Smart card usage is expanding rapidly. To encourage more organizations and users to adopt smart cards for enhanced security, the process to provision and use new smart cards is simplified and supports more end user scenarios.
Troubleshooting Smart Card Plug and Play Issues


Updated: November 3, 2009 Applies To: Windows 7, Windows Server 2008 R2

Symptom  After installing or upgrading to Windows 7, smart card Plug and Play detection may not work as expected, and a user or local administrator needs to find and resolve problems that prevent smart card Plug and Play detection on Windows 7 from functioning correctly.

Cause  This section of the guide provides troubleshooting information that helps you find and resolve smart card Plug and Play issues in Windows 7.

Resolution  Before you begin troubleshooting smart card Plug and Play, you should ensure that you can provide administrative credentials.

You should also understand smart card solutions that are not compatible with Plug and Play and how smart cards work with Remote Desktop connections.

Administrative credentials  You must be a member of the local Administrators group on the Windows 7–based computer on which you are troubleshooting smart card issues or know the user name and password of a local administrator account.

If you are not logged on with an administrator account, you must provide administrator credentials to perform many of the tasks in this guide.

Smart card solutions that are not compatible with Plug and Play  Smart card Plug and Play only supports smart cards that require drivers to function.

Not all smart card solutions require drivers for integrating with Windows. These solutions do not use the Windows Smart Card Framework and must be installed on the computer before using the smart card for the first time.

The following solutions are not compatible with smart card Plug and Play:

- Custom cryptographic service provider (CSP)-based solutions.
- Custom key storage provider (KSP)-based solutions.
- Public Key Cryptography Standard #11 (PKCS #11)-based solutions.
- Smart card driver packages without complete INF files or with incorrect device identifications.
- Some multislot smart card readers that create only one device for all available slots in the smart card reader.

Each time a smart card is inserted in the computer, Windows attempts to download and install the smart card driver if it is not already available on the computer.

You may see a Plug and Play error when you insert a non-Plug and Play smart card on the computer. This does not necessarily mean that there is a problem with the smart card.

If your deployment uses only non-Plug and Play smart card solutions, smart card Plug and Play can be disabled by a local administrator on a client computer.

Disabling smart card Plug and Play prevents smart card drivers, also known as smart card mini-drivers, from downloading and prevents smart card Plug and Play prompts.

To disable smart card Plug and Play in local Group Policy

- On a client computer, click Start, type gedit.msc in the Search programs and files box, and then press ENTER.
In the console tree under Computer Configuration, click Administrative Templates.

In the details panel, double-click Windows Components, and then double-click Smart Card.

Right-click Turn on Smart Card Plug and Play service, and then click Edit.

Click Disabled, and then click OK.

For enterprise deployments, smart card Plug and Play can be disabled by deploying a Group Policy.

For information about administrative templates, see Administrative templates overview for GPMC (http://go.microsoft.com/fwlink/?LinkId=152390).

**Warning:** For commercial deployments that target end-users (such as online banking) and environments that include both Plug and Play and non-Plug and Play smart cards, using Group Policy to disable Plug and Play for smart cards is strongly discouraged because it will affect all the smart cards in your environment.

---

**Remote Desktop connections and smart cards** Smart card Plug and Play works only for local sessions on a computer.

The smart card driver must be installed on the local computer before attempting to use smart cards with Remote Desktop connections.

The driver can be installed by inserting a Plug and Play–compatible smart card in a smart card reader on the local computer or by manually installing the driver.

For information about manually installing drivers, see Manually Install a Smart Card Driver in this guide.

**Steps for troubleshooting smart card Plug and Play** The steps to identify and resolve issues associated with smart card detection and driver installation are demonstrated in the following flowchart.

**Smart Card troubleshooting steps** The table follows the troubleshooting steps in the flowchart and provides links to the information about the troubleshooting step that may help you resolve the issue.

**Troubleshooting step Description**

**Does the smart card work as expected?** If the smart card works as expected, the user can use the smart card. If the smart card does not work as expected, begin the troubleshooting process.

**Is the logon screen displayed?** If the logon screen is displayed, log on to the computer as a local administrator, and then see Verify that the Smart Card Reader Device and Driver Are Installed Correctly.

If the computer is configured to allow only smart card logon, see Log On in Safe Mode to Configure the Computer for Password Logon.

If the logon screen is not displayed, see Verify that the Smart Card Reader Device and Driver Are Installed Correctly.

**Are the smart card reader device and driver installed correctly?** To check the smart card reader device and driver, see Verify that the Smart Card Reader Device and Driver Are Installed Correctly. If you reinstalled the smart card device driver, reinsert the smart card into the smart card reader.

If the smart card reader device and driver are installed correctly, see Verify that the Smart Card Is Installed Correctly.
Is the Smart cards node listed in Device Manager? If the smart card is listed in the Smart cards node in Device Manager, the user can use the smart card.

If the Smart cards node is not listed in Device Manager, see Verify Network Connectivity.

Is network connectivity available? If network connectivity is available, see Verify that Windows Update Is Enabled.

If network connectivity is not available, enable network connectivity, and then reinsert the smart card into the smart card reader.

Is Windows Update enabled? If Windows Update is enabled, see Verify that the Certificate Propagation and Smart Card Services Are Running. If Windows Update is not enabled, enable Windows Update, and then see Verify that the Smart Card Is Installed Correctly.

Are the Certificate Propagation and Smart Card services running? If the services are running, see Manually Install a Smart Card Driver, and then reinsert the smart card into the smart card reader.

If the services are not running, start the services, and then see Verify that the Smart Card Is Installed Correctly.

Winscard win32 The windows version of the libpcsclite linux version.

Winscard.h

/*****

Copyright (c) 1996 Microsoft Corporation

Module Name:

    WinSCard

Abstract:

    This header file provides the definitions and symbols necessary for an
    Application or Smart Card Service Provider to access the Smartcard
    Subsystem.

Environment:

    Win32

Notes:

    --*/

    //
    //******************************************************************************/
    //
    // Reader Control Routines
    //
    // The following services provide for direct, low-level manipulation of the
    // reader by the calling application allowing it control over the
attributes of the communications with the card.

extern WINSCARDAPI LONG WINAPI SCardControl(
    IN SCARDHANDLE hCard,
    IN DWORD dwControlCode,
    IN LPCVOID lpInBuffer,
    IN DWORD nInBufferSize,
    OUT LPVOID lpOutBuffer,
    IN DWORD nOutBufferSize,
    OUT LPDWORD lpBytesReturned);

PC/SC tools and libraries

See Also:

- smartcard tools

pycard PCSC

See Also:

- pyscard

pypassword

Introduction

See Also:

http://code.google.com/p/pypassport/

pypassword is an API written in Python that allows to interact with electronic passports compliant with ICAO Doc9303. It provides an interface with the following functionalities:

- Perform the Basic Access Control and Secure Messaging.
- Perform the passive and active authentications.
- Read the content of the ePassport, including pictures in JPEG2000 format.
- Create “valid” auto-signed ePassport content.

pypassword is released under GNU Lesser General Public License, Version 3 License.

Links

- The JMRTD website provides some Country Signing Certificates that can be used with pypassword.
- You can learn more about electronic passports: “Belgian Biometric Passport does not get a pass...” by Gildas Avoine, Kassem Kalach, and Jean-Jacques Quisquater (2007).
- Some related softwares:
  - Golden Reader Tool by BSI
  - RFIDIOt : RFID tools (Python)
  - wzMRTD : ePassport API by Johann Dantant (C++)
DBUS and pesc

#!/usr/bin/env python

""
pcsc-watcher.py
(c) 2011 by Florian "floe" Echtler <floe@butterbrot.org>
""

import gobject
import dbus
import dbus.service
from dbus.mainloop.glib import DBusGMainLoop
from smartcard.CardMonitoring import *
from smartcard.util import *

# dbus stuff

service = None
interface = "org.debian.alioth.pcsclite" # also used as bus name
path = "/org/debian/alioth/pcsclite/reader0/slot0"

# very simple DBus service class with one signal
class PCSC_DBus_Service(dbus.service.Object):
    def __init__(self,object_path):
        bus_name = dbus.service.BusName(interface, bus=dbus.SessionBus())
        dbus.service.Object.__init__(self, bus_name, object_path)
    @dbus.service.signal(interface)
    def CardPresenceChanged(self, atr, added):
        pass

# smartcard stuff

# simple card observer class which calls the DBus service
class PCSC_Card_Observer(CardObserver):
    def update(self, observable, (addedcards, removedcards)):
        for card in addedcards:
            service.CardPresenceChanged(toHexString(card.atr),True)
        for card in removedcards:
            service.CardPresenceChanged(toHexString(card.atr),False)

# main

DBusGMainLoop(set_as_default=True)
gobject.threads_init() # very important to avoid starving the PCSC thread

service = PCSC_DBus_Service(path)

cardmonitor = CardMonitor()
cardobserver = PCSC_Card_Observer()
cardmonitor.addObserver(cardobserver)

gobject.MainLoop().run()

Proprietary commands

EscapeCommandEnable  Les commandes propriétaires doivent être activées dans la base de registres pour pouvoir être utilisées par le pilote usbccid de Microsoft (MS).

In order to send or receive an Escape command to a reader, the DWORD registry value EscapeCommandEnable must be added and set to a non-zero value under the HKLM\SYSTEM\CurrentControlSet\Enum\USB\Vid*Pid*Device Properties key.

See Also:
  • http://www.microsoft.com/whdc/device/input/smartcard/USB_CCID.mspx
  • ush.ids

PC/SC commands

APDU Wrapping : proposition de comportement de l’API

-----Message d’origine-----
De : Philippe BOURGAULT [mailto:philippe.bourgault@id3.eu]
Envoyé : mardi 13 janvier 2009 11:31
À : ’Fabien Brun’; Patrick Vergain
Objet : Command Passing Method and APDU Wrapping : proposition de comportement de l’API.

Introduction  Managing specific contactless reader functionalities requires lots of commands. Some commands are defined by PC/SC 2.01 specifications but some other commands are vendor specific = proprietary.

When defined by PC/SC 2.01 specifications, commands are formatted like APDU and passed through a transmit() method call (PC/SC SCardTransmit method). Regarding proprietary commands, the CL1356A reader allows them to be passed either through a control() method call (PC/SC SCardControl method) or through a transmit() method call (PC/SC SCardTransmit method) using a APDU wrapping mechanism.
**ScardControl**  
SCardControl passing method advantages are:

- Commands be passed using a direct connection to reader avoiding the need for a card to be present on reader. Changing reader parameters, loading cryptographic keys in reader, etc., can be performed without asking user to place a card on reader.
- Commands can also be passed if a shared connection has been established.

SCardControl passing method drawbacks are:

- This passing method is considered unsecured and both Windows and Linux CCID driver forbid the use of this method by default.
- Under Windows, a special registry value named “EscapeCommandEnable” must be added for to allow the driver to forward commands to reader but this requires administrative rights to modify registry.
- Under Linux, a bit must be set inside a driver configuration file requiring root login.

**ScardTransmit**  
SCardTransmit passing method advantages are:

- This passing method is considered secured and requires only basic user rights on both Windows and Linux.

SCardTransmit passing method drawbacks are:

- Commands will be denied if a direct connection to reader has been established. Only shared connection are allowed and thus requires asking user to place a card on reader even when commands only involve the reader and not the card like changing a communication parameter.

**Proposition**  
Command passing may be set to AUTOMATIC, CONTROL or TRANSMIT

When set to TRANSMIT, a successful connect() or connect(false) method call should have been performed to establish a shared connection to a card. As a consequence, a card must be physically present on the reader. Specific reader commands will be passed to the reader using an APDU wrapping mechanism through a transmit() method call (PC/SC SCardTransmit method). When defined by PC/SC 2.01, commands will be passed using PC/SC defined APDU format through a transmit() method call (PC/SC SCardTransmit method).

When set to CONTROL, a successful connectDirect() or connect(true) method call should have been performed to get a direct connection to the reader. Specific reader commands will be passed to reader using proprietary format through a control() method call (PC/SC SCardControl method). It is not possible to pass commands in PC/SC 2.01 APDU format when direct connection is established.

When set to AUTOMATIC, appropriate command passing method will be chosen depending of the current connection type; direct or shared. If currently established connection is direct, proprietary commands will be passed through a control() method call (PC/SC SCardControl method). Commands only defined by PC/SC 2.01 will fail. If currently established connection is shared, proprietary commands will be passed through a transmit() method call (PC/SC SCardTransmit method) using a APDU wrapping mechanism and commands defined by PC/SC 2.01 will be passed through a transmit() method call (PC/SC SCardTransmit method).

Passing method defaults to AUTOMATIC.

Please note that when a card is present on the reader, it is also possible to use a connect() or a connect(false) method call and to use the CONTROL command passing method. Specific reader commands will be passed through a control() method call (PC/SC SCardControl method) and PC/SC 2.01 commands will be passed through a transmit() method call (PC/SC SCardTransmit method).

Pour avis, Philippe.

**CCID and PC/SC people**
USB ccid PC/SC people

Ludovic Rousseau  See Also:
  • http://ludovic.rousseau.free.fr
  • http://ludovicrousseau.blogspot.com/
  • http://pcsclite.alioth.debian.org/

Big matrix

Smartcard list  See Also:
  • http://ludovic.rousseau.free.fr/softwares/pcsc-tools/smartcard_list.txt

Blog

Blog de Ludovic Rousseau

Blog de Ludovic Rousseau 2012

New version of pcsc-lite: 1.8.2 (18 January 2012)  See Also:
  • http://ludovicrousseau.blogspot.com/2012/01/new-version-of-pcsc-lite-182.html
  • https://alioth.debian.org/frs/?group_id=30105&release_id=1765#pcsclite-_1.8.2-title-content
  • http://anonscm.debian.org/viewvc/pcsclite/trunk/PCSC/

I just released a new version of pcsc-lite 1.8.2.
No big changes expect for pcsc-spy I talked about in a previous article PCSC API spy, third try

Changelog 18 January 2012
  • rename pcsc-spy.py to pcsc-spy and install it as a normal binary (in /usr/local/bin by default)
  • write a pcsc-spy.1 manpage
  • fix a bug with a multi-slot reader
  • Info.plist parser: avoid a buffer read overflow in management
  • Some Doxygen improvements

Blog de Ludovic Rousseau 2011

2.9. Drivers
PC/SC API spy (18 novembre 2011)  See Also:

- **PC/SC API spy**

I already blogged about how to spy the PCSC API in PCSC API spy for GNU systems and PCSC API spy, another way.

But I am still not happy with the limitations and side effects.

**pcscd now uses libudev instead of libhal**  See Also:


**pcsc-lite** first started (in the previous century) when only serial readers were in use.

The serial readers are configured using the /etc/reader.conf file.

The problem is that the configuration is static and can’t be used with a USB reader.

**History of USB plug-n-play mechanism**  To detect the insertion or removal of a USB reader different mechanisms have been used in the last 10 years.

**Linux**  hotplug_linux.c: introduced at least in (or before) version 1.0.2beta2 (20 Dec, 2001).

The problem is that the code is Linux specific.

**libusb**

**hotplug_libusb.c**  introduced in version 1.2.0-rc1 (26 August, 2003). From the changelog:

- src/hotplug_libusb.c: Add support of libusb. Allow to use USB readers on BSD or any plateform supported by libusb. Thanks to Toni Andjelkovic for the great job.

The problem of libusb is that pcscd is constantly polling the USB bus to detect reader hotplug.

**libhal**

**hotplug_libhal.c**  introduced in version 1.4.100 (23 March 2008).

From the changelog:

- add libhal support to avoid polling the USB bus. libusb is still supported but libhal is now the default

The problem is that **libhal has been deprecated** upstream in May 2008 (**2 months after pcscd started using it**) !. Some distributions are actively migrating out of libhal.

See Debian bug 587979 “pcscd: uses deprecated HAL” for example. And the wiki page HALRemoval.

**libudev**

**hotplug_libudev.c**  introduced in version 1.6.8 (not yet released as of February 2011).

It looks like **libudev** is a good choice for the future. I don’t know if it is supported by other systems than GNU/Linux.

I removed the support of libhal but support of libusb is still present.

See Also:
Mac OS X  Mac OS X has its own hotplug mechanism in hotplug_macosx.c. Apple maintains its own forked version of pcsc-lite. So I do not plan to change anything Mac OS X specific in pcsc-lite.

Conclusion  The hotplug mechanism is a fast moving target on GNU/Linux. I hope libudev will last at least a few months before I have to move to something else.

Blog de Ludovic Rousseau 2010

I now have a blog to talk about smart cards http://ludovicrousseau.blogspot.com/

from  Ludovic Rousseau <ludovic.rousseau@gmail.com>
reply-to  MUSCLE <muscle@lists.musclecard.com>
to  MUSCLE <muscle@lists.musclecard.com>
date  Thu, Apr 8, 2010 at 1:37 PM
subject  [Muscle] I now have a blog to talk about smart cards

Hello,

I created a blog to talk about smart card in general and pcsc-lite, libccid and other projects I work on in particular. I am not a blogger, this is my first blog.

- http://ludovicrousseau.blogspot.com/
- http://smartcard-atr.appspot.com/

I have two posts for now:

- Parsing an ATR
- PC/SC sample in different languages

I don’t know if I should also post my blog messages on this mailing list. Do you have any opinion on this?

Enjoy

from  Burak Ilgicioglu <ilgicioglu@gmail.com>
reply-to  MUSCLE <muscle@lists.musclecard.com>
to  MUSCLE <muscle@lists.musclecard.com>
date  Fri, Apr 9, 2010 at 4:30 PM
subject  Re: [Muscle] I now have a blog to talk about smart cards

I have blog on smart cards, but on the contactless and dual interface ones :)

http://www.contactless-world.com

Mainly focused on the application and functionality level, rather than coding like Ludovic’s one.

Regards,

Glossary
CCID Chip/Smart Card Interface Devices Glossary  See Also:
  • Smartcard Glossary

APDU Command Header  The four byte sequence that begins an APDU; CLA INS P1 P2 (ISO/IEC 7816-4 § 5.3.1)

CCID  Chip/Smart Card Interface Devices. The USB CCID specification from the USB working group aims to nor-

malize USB smartcard readers, in order to have a single driver (supplied once for all with the operating system)

for virtually any reader from any manufacturer.

Smart Card  Any of a number of similar devices conforming to ISO/IEC 7816-3.

T=0 Command Header  The sequence of five bytes; CLA INS P1 P2 P3 [ISO/IEC 7816-3 § 8.3.2].

WI  Waiting time Integer for protocol T = 0

USB Composite Class Devices (CDC)  USB Composite Class Devices

The USB specification defines a composite class device as a device whose device-descriptor fields for device class
(bDeviceClass) and device subclass (bDeviceSubClass) both have the value 0. A composite class device appears to
the system as a USB device using a single bus address that may present multiple interfaces, each of which represents
a separate function.

A good example of a composite class device is a multifunction device, such as a device that performs printing, scan-
ning, and faxing. In such a device, each function is represented by a separate interface.

In Mac OS X, the I/O Kit loads the AppleUSBComposite device driver for composite class devices that do not already
have vendor-specific device drivers to drive them.

The AppleUSBComposite driver configures the device and causes drivers to be loaded for each USB interface.
Although most multifunction USB devices are composite class devices, not all composite class devices are multifunc-
tion devices.

The manufacturer of a single-function USB device is at liberty to classify the device as a composite class device as
long as the device meets the USB specifications.

USB links

Tutorials on USB
  • http://www.usbmadesimple.co.uk/
  • http://www.beyondlogic.org/usbnutshell/

Linux USB  See Also:
  • http://www.linux-usb.org/
  • http://linux-hotplug.sourceforge.net
  • http://www.linuxmaine.fr/article/31?page=user/get_informations

Snooping the USB data stream  See Also:
http://www.linuxjournal.com/node/7582
Writing a real driver in user space  See Also:
http://www.linuxjournal.com/node/7466

Writing a simple USB driver  See Also:
http://www.linuxjournal.com/node/7353

Linux Device drivers, Third edition  See Also:
http://lwn.net/Kernel/LDD3/

USB in a nutshell  See Also:
http://www.beyondlogic.org/usbnutshell/

Programming guide for Linux USB device drivers  See Also:
http://www.lrr.in.tum.de/Par/arch/usb/usbdoc/

Tutorial on USB with Linux  See Also:
http://www.cs.indiana.edu/~bpisupat/work/usb.html

Cours USB de G. Fondeville  See Also:
http://g.fondeville.free.fr/usb_cours.html

USB FAQs
  • http://www.usb.org/developers/usbfaq/
  • http://www.linux-usb.org/FAQ.html

USB TIPS

Tip 1 : mount -t usbfs usbfs /proc/bus/usb  See Also:

PCSCD got segmentation fault on ARM v5 with uClibc

dep Ludovic Rousseau ludovic.rousseau@gmail.com
heure de l’expéditeur Envoyé à 14:07 (GMT+02:00). Heure locale : 14:36.
répondre à MUSCLE <muscle@lists.musclecard.com>
à MUSCLE <muscle@lists.musclecard.com>
date 8 juillet 2011 14:07
objet Re: [Muscle] PCSCD got segmentation fault on ARM v5 with uClibc
I want to say sorry for wasting your time in this issue, but then I found out the root cause is that the usbfs is not mounted. Too bad, this information is not available when executing pcscd. The issue is solved by simply:

```
mount -t usbfs usbfs /proc/bus/usb
```

I would to propose a patch to check the return value of libusb_init() call inside src/libusb/hotplug_libusb.c so the user will be notified early if there is something wrong with libusb.

Not checking the libusb_init() returned value was a real bug. It is now fixed thanks to you.

Thanks for your time and response. Keep the good work!

You are welcome.

Bye

Linix and windows USB tools

USB linux tools

usbmon_helper

```
#!/usr/bin/python

„„
Thibaut Colar
http://wiki.colar.net/
Quick (&dirty) script that takes in an usbmon log (USB sniffer) and dumps out an easier to read html file.

This is used to debug/rev. engineer USB protocols under linux/unix.
What it adds to a usbmon log:
- colors so it’s easier to scan through
- timestamps shown as time offsets - easier to see where we are at
- data packets shown in ascii as well as hex (ascii helps seing commands)
e tc...

Example of use: :

Doing an USBMon trace

```
# mount -t debugfs none_debugs /sys/kernel/debug

if not in kernel:

```
# sudo modprobe usbmon

cat /sys/kernel/debug/usbmon/4u > /tmp/usb.log
==========================================================================

If you want to scan a specific device, find it’s device number (sudo lsusb):

$ cat /sys/kernel/debug/usbmon/4u > /tmp/usb.log

run some usb transactions of some kind :-) when done kill the
‘cat’ command (^c)

Converting to HTML
==================

# python usbmon_helper.py /tmp/usb.log > /tmp/usb.html
Open /tmp/usb.html in browser and enjoy!

USBMon Doc:
http://www.mjmwired.net/kernel/Documentation/usb/usbmon.txt

usbmon  See the documentation about usbmon.

File output example

Time: Time offset from first packet (MM:SS.mmmm)
Type: S: submission, C: control, E: Error
Address: Address (URB type and direction, Bus number, Device address, Endpoint nb)
          - URB type and direction:
            Ci Co Control input and output
            Zi Zo Insochronous input and output
            Ii Io Interrupt input and output
            Bi Bo Bulk input and output
Status: Status Code returned

usbmon  The name “usbmon” in lowercase refers to a facility in kernel which is used to collect traces of I/O on
the USB bus. This function is analogous to a packet socket used by network monitoring tools such as tcpdump(1) or
Ethereal. Similarly, it is expected that a tool such as usbdump or USBMon (with uppercase letters) is used to examine
raw traces produced by usbmon.

The usbmon reports requests made by peripheral-specific drivers to Host Controller Drivers (HCD). So, if HCD is
buggy, the traces reported by usbmon may not correspond to bus transactions precisely. This is the same situation as
with tcpdump.

• How to use usbmon to collect raw text traces

Unlike the packet socket, usbmon has an interface which provides traces in a text format. This is used for two purposes.
First, it serves as a common trace exchange format for tools while more sophisticated formats are finalized. Second,
humans can read it in case tools are not available.

To collect a raw text trace, execute following steps.

debugfs and usbmon (if not built into the kernel)
Mount debugfs  Mount debugfs (it has to be enabled in your kernel configuration):

mount -t debugfs none_debugs /sys/kernel/debug

usbmon module  Load the usbmon module (if built as module). This step is skipped if usbmon is built into the kernel:

modprobe usbmon

In our linux kernel configuartion the usbmon module is built with the kernel.

Verify that bus sockets are present.  With linux 2.6.18

[root@houx usbmon]# cd /sys/kernel/debug/usbmon; ls;pwd
ls 1t 2s 2t 3s 3t 4s 4t 5s 5t 6s 6t
/sys/kernel/debug/usbmon

Now you can choose to either:

• use the socket ‘0u’ (to capture packets on all buses), and skip to step #3,
• or find the bus used by your device with step #2.

This allows to filter away annoying devices that talk continuously.

Find which bus connects to the desired device

cat /proc/bus/usb/devices  Run:

cat /proc/bus/usb/devices

and find the T-line which corresponds to the device. Usually you do it by looking for the vendor string. If you have many similar devices, unplug one and compare two /proc/bus/usb/devices outputs. The T-line will have a bus number.

Example:

T: Bus=04 Lev=01 Prnt=01 Port=01 Cnt=01 Dev#= 2 Spd=12 MxCh= 0
D: Ver= 2.00 Cls=ff(vend.) Sub=ff Prot=ff MxPS= 8 #Cfgs= 1
P: Vendor=0b81 ProdID=0103 Rev= 0.01
S: Manufacturer=id3 Semiconductors
S: Product=CERTIS 2
S: SerialNumber=0137064
C:* #Ifs= 1 Cfg#= 1 Atr=80 MxPwr=200mA
I: If#= 0 Alt= 0 #EpS= 0 Cls=ff(vend.) Sub=00 Prot=00 Driver=(none)
I: If#= 0 Alt= 1 #EpS= 3 Cls=ff(vend.) Sub=00 Prot=00 Driver=(none)
E: Ad=83(I) Atr=01(Isoc) MxPS= 532 Iv=1ms
E: Ad=02(O) Atr=02(Bulk) MxPS= 32 Iv=0ms
E: Ad=82(I) Atr=02(Bulk) MxPS= 32 Iv=0ms
I: If#= 0 Alt= 2 #EpS= 3 Cls=ff(vend.) Sub=00 Prot=00 Driver=(none)
E: Ad=83(I) Atr=01(Isoc) MxPS= 692 Iv=1ms
E: Ad=02(O) Atr=02(Bulk) MxPS= 32 Iv=0ms
E: Ad=82(I) Atr=02(Bulk) MxPS= 32 Iv=0ms
I: If#= 0 Alt= 3 #EpS= 3 Cls=ff(vend.) Sub=00 Prot=00 Driver=(none)
E: Ad=83(I) Atr=01(Isoc) MxPS= 868 Iv=1ms
lsusb or:

lsusb

[root@houx ~]# lsusb
Bus 001 Device 001: ID 0000:0000
Bus 001 Device 002: ID 0bda:0111 Realtek Semiconductor Corp. Card Reader
Bus 003 Device 001: ID 0000:0000
Bus 004 Device 002: ID 0b81:0103 id3 Semiconductors
Bus 004 Device 001: ID 0000:0000
Bus 005 Device 001: ID 0000:0000
Bus 006 Device 001: ID 0000:0000
Bus 002 Device 001: ID 0000:0000
[root@houx ~]#

Bus=04 means it’s bus 4

Start ‘cat’

sudo cat /sys/kernel/debug/usbmon/4t > /tmp/usb.log

to listen on a single bus.

This process will be reading until killed. Naturally, the output can be redirected to a desirable location. This is preferred, because it is going to be quite long.

Perform the desired operation on the USB bus This is where you do something that creates the traffic: plug in a flash key, copy files, control a webcam, etc.

Kill cat Usually it’s done with a keyboard interrupt (Control-C).

At this point the output file (/tmp/1.mon.out in this example) can be saved, sent by e-mail, or inspected with a text editor. In the last case make sure that the file size is not excessive for your favourite editor.

Raw text data formats [1t, 1u] Two formats are supported currently: the original, or ‘1t’ format, and the ‘1u’ format.

1t format The ‘1t’ format is deprecated in kernel 2.6.21.
The ‘1u’ format adds a few fields, such as ISO frame descriptors, interval, etc. It produces slightly longer lines, but otherwise is a perfect superset of ‘1t’ format.

If it is desired to recognize one from the other in a program, look at the “address” word (see below), where ‘1u’ format adds a bus number. If 2 colons are present, it’s the ‘1t’ format, otherwise ‘1u’.

Any text format data consists of a stream of events, such as URB submission, URB callback, submission error. Every event is a text line, which consists of whitespace separated words. The number or position of words may depend on the event type, but there is a set of words, common for all types.

Here is the list of words, from left to right:

- **URB Tag.** This is used to identify URBs, and is normally an in-kernel address of the URB structure in hexadecimal, but can be a sequence number or any other unique string, within reason.

- **Timestamp in microseconds,** a decimal number. The timestamp’s resolution depends on available clock, and so it can be much worse than a microsecond (if the implementation uses jiffies, for example).

- **Event Type.** This type refers to the format of the event, not URB type. Available types are: S - submission, C - callback, E - submission error.

- **“Address” word (formerly a “pipe”).** It consists of four fields, separated by colons: URB type and direction, Bus number, Device address, Endpoint number. Type and direction are encoded with two bytes in the following manner:
  - Ci Co Control input and output
  - Zi Zo Isochronous input and output
  - Ii Io Interrupt input and output
  - Bi Bo Bulk input and output

  Bus number, Device address, and Endpoint are decimal numbers, but they may have leading zeros, for the sake of human readers.

- **URB Status word.** This is either a letter, or several numbers separated by colons: URB status, interval, start frame, and error count. Unlike the “address” word, all fields save the status are optional. Interval is printed only for interrupt and isochronous URBs. Start frame is printed only for isochronous URBs. Error count is printed only for isochronous callback events.

  The status field is a decimal number, sometimes negative, which represents a “status” field of the URB. This field makes no sense for submissions, but is present anyway to help scripts with parsing. When an error occurs, the field contains the error code.

  In case of a submission of a Control packet, this field contains a Setup Tag instead of an group of numbers. It is easy to tell whether the Setup Tag is present because it is never a number. Thus if scripts find a set of numbers in this word, they proceed to read Data Length (except for isochronous URBs). If they find something else, like a letter, they read the setup packet before reading the Data Length or isochronous descriptors.

- **Setup packet, if present, consists of 5 words:** one of each for bmRequestType, bRequest, wValue, wIndex, wLength, as specified by the USB Specification 2.0. These words are safe to decode if Setup Tag was ‘s’. Otherwise, the setup packet was present, but not captured, and the fields contain filler.

- **Number of isochronous frame descriptors and descriptors themselves.** If an Isochronous transfer event has a set of descriptors, a total number of them in an URB is printed first, then a word per descriptor, up to a total of 5. The word consists of 3 colon-separated decimal numbers for status, offset, and length respectively. For submissions, initial length is reported. For callbacks, actual length is reported.

- **Data Length.** For submissions, this is the requested length. For callbacks, this is the actual length.

- **Data tag.** The usbmon may not always capture data, even if length is nonzero. The data words are present only if this tag is ‘=’.
• Data words follow, in big endian hexadecimal format. Notice that they are not machine words, but really just a byte stream split into words to make it easier to read. Thus, the last word may contain from one to four bytes. The length of collected data is limited and can be less than the data length report in Data Length word.

Linux Usbview

• http://github.com/gregkh/usbview/

A version of USBView is available for Linux from http://www.kroah.com/linux-usb/. As with the Windows version, this displays a connection tree of all the USB devices connected to the PC. A version compiled for x86_64 Linux can be downloaded here.

The source

git clone https://github.com/gregkh/usbview.git
Cloning into usbview...
remote: Counting objects: 312, done.
remote: Compressing objects: 100% (171/171), done.
remote: Total 312 (delta 168), reused 233 (delta 131)
Receiving objects: 100% (312/312), 662.53 KiB | 195 KiB/s, done.
Resolving deltas: 100% (168/168), done.

Tree

|-- AUTHORS
|-- COPYING
|-- ChangeLog
|-- INSTALL
|-- Makefile.am
|-- NEWS
|-- README
|-- TODO
|-- about-dialog.c
|-- autogen.sh
|-- callbacks.c
|-- config.h.in
|-- configure-dialog.c
|-- configure.in
|-- debian
| |-- changelog
| |-- compat
| |-- control
| |-- copyright
| |-- rules
| |-- usbview.desktop
| |-- usbview.dirs
| |-- usbview.docs
| |-- usbview.manpages
| '-- usbview.menu
|-- interface.c
|-- main.c
|-- showmessage.c
|-- usb_icon.xpm
|-- usbparse.c
|-- usbparse.h
|-- usbtree.c
Development tools, Release 2012.06.18

|-- usbtree.h
|-- usbview.8
|-- usbview.spec
|-- usbview_logo.xcf
`-- usbview_logo.xpm

Installation  > ./autogen.sh >

Usbutils


A version of USBView is available for Linux from [http://www.kroah.com/linux-usb/](http://www.kroah.com/linux-usb/). As with the Windows version, this displays a connection tree of all the USB devices connected to the PC. A version compiled for x86_64 Linux can be downloaded [here](http://www.kroah.com/linux-usb/).

News


Big thing is support for the audio class descriptors.

Other than that, it’s just a lot of little things, as can be seen in the changelog below.

The package can be downloaded from kernel.org: [http://www.kernel.org/pub/linux/utils/usb/usbutils/](http://www.kernel.org/pub/linux/utils/usb/usbutils/)

We’ve switched over to using git for development now, which makes things much easier than the old cvs tree. The tree can be found on both kernel.org and github.com if you want to fork it and send us changes easier:

- [http://github.com/gregkh/usbutils/tree/master](http://github.com/gregkh/usbutils/tree/master)

thanks,

vusb-analyzer


vusb-analyzer announce


Subject: Open source USB sniffer GUI
Date: Tuesday, January 13, 2009 - 10:54 am

Hi,

I apologize in advance if this isn’t the right place for such an announcement, but I thought this might be of interest to the Linux USB driver community.

I’d like to announce a new open source GUI tool for visualizing USB sniffer logs. It’s written in Python with PyGTK, and it has some novel
Development tools, Release 2012.06.18

features including a graphical timeline view. We built this tool at VMware for analyzing sniffer logs produced by our USB virtualization stack. There are instructions on the web site for capturing sniffer logs using VMware products (including the free-as-in-beer VMware Player). This is pretty useful, since you capture logs from a device’s Windows driver, then capture logs from your own Linux driver and compare the two side-by-side. But if VMware isn’t your cup of tea, there’s also an architecture in place for pluggable log file decoder modules. We already have support for decoding the logs produced by Ellisys hardware analyzers, and we’d love to see someone add support for usbmon. If you’re interested, the project web site has source code, documentation, and plenty of screenshots:
http://vusb-analyzer.sourceforge.net/

This is a GUI tool for analyzing logs of traced USB communications.

dependencies

- python 2.6 (not python 2.7)
- gtk+
- pyGTK

Versions

vusb-analyser 1.1 Version 1.1 Includes support for Linux usbmon logs. (beta) This code was contributed by Christoph Zimmermann. The usbmon support hasn’t been tested as extensively as the other formats, and it doesn’t support all vusb-analyzer features, but other than that it should be quite usable and stable.

python vusb-analyser See the documentation about usbmon.

```bash
drwxr-xr-x 3 5724 201 4096 aoû 23 14:31 VUsbTools
[root@houx vusb-analyzer-1.1]# python vusb-analyzer
```

usage: vusb-analyzer [-t] vmx.log [vmx.log]

PyGTK frontend for the virtual USB analyzer
Micah Dowty <micah@vmware.com>
Version 1.1

- t Tail mode, start from the end of a growing log file.

Supported log formats:
    VMware VMX log file (*.log)
    Exported XML from Ellisys Visual USB (*.xml)
    Linux usbmon log, raw ASCII format (*.mon)

Also supports transparent decompression of gzipped (*.gz) files. Logs may be appended to while this program is running.

For best results with Ellisys logs, enable 'Expand

2.9. Drivers
transactions packets’ but not ‘Expand consecutive elements’ while exporting.

Two log files can be specified, in order to invoke diff mode.

**usbmon**  See the documentation about **usbmon**.

**Compilation**

**GTK dependencies**

- fontconfig
- pixman
- cairo
- glib
- atk
- pango
- gtk

after

- libffi
- gobject

**fontconfig**

- [http://fontconfig.org/release/](http://fontconfig.org/release/)

**pixman**


**cairo**

- [http://www.cairographics.org/](http://www.cairographics.org/)

**glib**


**atk**

**pango**
### gtk+2.24

- gtk > 2.10

```bash
pkg-config --modversion gtk+-2.0
```

```
[root@houx glib-2.24.1]# pkg-config --modversion gtk+-2.0
2.10.4
[root@houx glib-2.24.1]#
```

### libffi


### gobject-introspection

### Compilations examples

Instructions to compile the current GIMP source code (master version in GIT repository) with Ubuntu Linux 9.04 + 9.10

Important: The current GIT master version is a snapshot, an intermediate version that can be used to follow current developments (for instance, the single window mode). For production use, always use a stable GIMP version (2.6)!

The procedure is the same for Ubuntu 9.04 and 9.10, but with Ubuntu 9.04 you have to download & compile the newest Gtk version.

```bash
# (as non-root user)

cd ~
mkdir -p tmp
cd tmp

# Important! These variables have to be set and # are required for all further commands. If you close the # terminal window, you have to give the 3 export commands # again.

export PATH=$PATH:/opt/gimp-2.7/bin
export PKG_CONFIG_PATH=/opt/gimp-2.7/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gimp-2.7/lib

# Fetch the most important packages
sudo apt-get build-dep gimp

# Additional packages
sudo aptitude install checkinstall git-core libtool libopenexr-dev libopenraw-dev libspiro-dev

### BEGIN: ONLY NEEDED FOR UBUNTU 9.04 ###

# Fetch, compile, install Glib (needed for Gtk)
wget http://ftp.gnome.org/pub/gnome/sources/glib/2.22/glib-2.22.2.tar.bz2
tar -xjf glib-2.22.2.tar.bz2
cd glib-2.22.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install Gtk
wget http://ftp.gnome.org/pub/gnome/sources/gtk+/2.18/gtk+-2.18.2.tar.bz2
```

2.9. Drivers
tar -xjf gtk+-2.18.2.tar.bz2
cd gtk+-2.18.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..
### END: ONLY NEEDED FOR UBUNTU 9.04 ###

# Fetch, compile, install BABL
git clone --depth 1 git://git.gnome.org/babl
cd babl
./autogen.sh --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install GEGL
git clone --depth 1 git://git.gnome.org/gegl
cd gegl
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install GIMP
git clone --depth 1 git://git.gnome.org/gimp
cd gimp
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..
Launch GIMP with:
/opt/gimp-2.7/bin/gimp-2.7

For gtk+

export PATH=/opt/gtk2/bin:$PATH
export PKG_CONFIG_PATH=/opt/gtk2/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gtk2/lib

./configure --prefix=/opt/gtk2

cairo

cairo (version 1.8.10 [release]) will be compiled with:

The following surface backends:
Image: yes (always builtin)
Xlib: yes
Xlib Xrender: yes
Quartz: no (requires CoreGraphics framework)
Quartz-image: no (disabled, use --enable-quartz-image to enable)
XCB: no (disabled, use --enable-xcb to enable)
Win32: no (requires a Win32 platform)
OS2: no (disabled, use --enable-os2 to enable)
PostScript: yes
HAL http://en.wikipedia.org/wiki/HAL_%28software%29

HAL is a software project providing a hardware abstraction layer for Unix-like computer systems. It aims to allow
desktop applications to discover and use the hardware of the host system through a simple, portable and abstract API,
regardless of the type of the underlying hardware.

HAL was originally envisioned by Havoc Pennington and is now a freedesktop.org project, being a key part of the
software stack of the GNOME and KDE desktop environments. It is free software, dual-licensed under both the GNU
General Public License and the Academic Free License.

**Warning:** HAL is now deprecated, with functionality being merged into udev as of 2008–2010;

HAL is deprecated As of 2009, distributions such as Ubuntu, Debian, and Fedora, and projects such as GNOME
and X.org are in the process of deprecating HAL as it has “become a large monolithic unmaintainable mess”.

It is in the process of being merged into udev (main udev, libudev, and udev-extras) and existing udev and kernel
functionality.

Ubuntu version 10.04 removes HAL from the boot process.

Initially a new daemon DeviceKit was planned to replace certain aspects of HAL, but in March 2009, DeviceKit was
deprecated in favor of adding the same code to udev as a package: udev-extras, and some functions have now moved
to udev proper.

**hal_device**

An extract of this command is:

```
1: udi = ‘/org/freedesktop/Hal/devices/usb_device_b81_103_0137064_if0’
info.udi = ‘/org/freedesktop/Hal/devices/usb_device_b81_103_0137064_if0’ (string)
linux.subsystem = ‘usb’ (string)
```
Development tools, Release 2012.06.18

```
linux.hotplug_type = 1 (0x1) (int)
info.product = 'USB Vendor Specific Interface' (string)
usb.interface.protocol = 0 (0x0) (int)
usb.interface.subclass = 0 (0x0) (int)
usb.interface.class = 255 (0xff) (int)
usb.interface.number = 0 (0x0) (int)
usb.linux.sysfs_path = '/sys/devices/pci0000:00/0000:00:13.1/usb3/3-2/3-2:1.0' (string)
usb.configuration_value = 1 (0x1) (int)
usb.num_configurations = 1 (0x1) (int)
usb.num_interfaces = 1 (0x1) (int)
usb.device_class = 255 (0xff) (int)
usb.device_subclass = 255 (0xff) (int)
usb.device_protocol = 255 (0xff) (int)
usb.vendor_id = 2945 (0xb81) (int)
usb.product_id = 259 (0x103) (int)
usb.vendor = 'id3 Semiconductors' (string)
usb.product = 'USB Vendor Specific Interface' (string)
usb.device_revision_bcd = 1 (0x1) (int)
usb.max_power = 200 (0xc8) (int)
usb.num_ports = 0 (0x0) (int)
usb.linux.device_number = 2 (0x2) (int)
usb.serial = '0137064' (string)
usb.speed_bcd = 4608 (0x1200) (int)
usb.version_bcd = 512 (0x200) (int)
usb.is_self_powered = false (bool)
usb.can_wake_up = false (bool)
usb.bus_number = 3 (0x3) (int)
info.bus = 'usb' (string)
info.parent = '/org/freedesktop/Hal/devices/usb_device_b81_103_0137064' (string)
linux.sysfs_path_device = '/sys/devices/pci0000:00/0000:00:13.1/usb3/3-2/3-2:1.0' (string)
linux.sysfs_path = '/sys/devices/pci0000:00/0000:00:13.1/usb3/3-2/3-2:1.0' (string)
```

2: udi = '/org/freedesktop/Hal/devices/usb_device_b81_103_0137064'
info.udi = '/org/freedesktop/Hal/devices/usb_device_b81_103_0137064' (string)
linux.subsystem = 'usb' (string)
usb_device.bus_number = 3 (0x3) (int)
usb_device.can_wake_up = false (bool)
usb_device.is_self_powered = false (bool)
usb_device.version_bcd = 512 (0x200) (int)
usb_device.speed_bcd = 4608 (0x1200) (int)
usb_device.serial = '0137064' (string)
usb_device.device_class = 255 (0xff) (int)
usb_device.device_subclass = 255 (0xff) (int)
usb_device.device_protocol = 255 (0xff) (int)
usb_device.device_revision_bcd = 1 (0x1) (int)
info.product = 'CERTIS 2' (string)
usb_device.product = 'CERTIS 2' (string)
info.vendor = 'id3 Semiconductors' (string)
usb_device.vendor = 'id3 Semiconductors' (string)
usb_device.product_id = 259 (0x103) (int)
usb_device.vendor_id = 2945 (0xb81) (int)
usb_device.device_policy = 255 (0xff) (int)
usb_device.device_subclass = 255 (0xff) (int)
usb_device.device_class = 255 (0xff) (int)
usb_device.num_interfaces = 1 (0x1) (int)
usb_device.num_configurations = 1 (0x1) (int)
usb_device.configuration_value = 1 (0x1) (int)
```
wireshark and pcap

Wireshark

Introduction

Sources:

- http://wiki.wireshark.org/CaptureSetup/USB

usbmon

To dump USB traffic on Linux, you need the **usbmon** module, which has existed since Linux 2.6.11. Information on that module is available in `/usr/src/linux/Documentation/usb/usbmon.txt` in the Linux source tree.

Depending on the distribution you’re using, and the version of that distribution, that module might be built into the kernel, or might be a loadable module; if it’s a loadable module, depending on the distribution you’re using, and the version of that distribution, it might or might not be loaded for you.

**Note:** On centos it’s built into the kernel

If it’s a loadable module, and not loaded, you will have to load it with the command:

```
modprobe usbmon
```

which must be run as root.

libcap

libpcap releases prior to 1.0 do not include USB support, so you will need at least libpcap 1.0.0.

**Warning:** on centos we have libpcap-0.9.4-15.el5.i386

linux kernel

For versions of the kernel prior to 2.6.21, the only USB traffic capture mechanism available is a text-based mechanism that limits the total amount of data captured for each raw USB block to about 30 bytes.

**Warning:** on centos the linux kernel is 2.6.18

There is no way to change this without patching the kernel.

If debugfs is not already mounted on `/sys/kernel/debug`, ensure that it is mounted there by issuing the following command as root:

```
mount -t debugfs / /sys/kernel/debug
```
Note: For kernel version 2.6.21 and later, there is a binary protocol for tracing USB packets which doesn’t have that size limitation.

For that kernel version, you will need *libpcap 1.1.0 or newer*, because the libpcap 1.0.x USB support uses, but does not correctly handle, the memory-mapped mechanism for USB traffic, which libpcap will use if available - it cannot be made unavailable, so libpcap will always use it.

- In libpcap 1.0.x, the devices for capturing on USB have the name usbn, where n is the number of the bus.
- In libpcap 1.1.0 and later, they have the name usbmonn.

You will also need a [Wireshark 1.2.x or newer](http://wiki.wireshark.org/Development).

**Warning:** on centos we have wireshark-1.0.15-1.el5_5.1.i386

---

Wireshark Installation

NOTE: this document applies to the Wireshark source releases and buildbot source tarballs.

It does not apply to source code checked out directly from Subversion, as files such as the configuration script are not checked into Subversion, but need to be generated from the autoconf and automake files.

See [http://wiki.wireshark.org/Development](http://wiki.wireshark.org/Development) if you would like to build the source code checked out directly from Subversion.

## Installation on centos

### GTK configuration

```bash
export PATH=/opt/gtk2/bin:$PATH
export PKG_CONFIG_PATH=/opt/gtk2/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gtk2/lib:$LD_LIBRARY_PATH
pkg-config glib-2.0 --modversion
```

**2.24.1**

### libpcap configuration

```
--with-pcap=DIR
Use this to tell Wireshark where you have libpcap installed, if it is installed in a non-standard location.
```

### autogen.sh

```
> ./autogen.sh
```

Checking for python. You must have autoconf 2.60 or later installed to compile Wireshark. Download the appropriate package for your distribution/OS, or get the source tarball at [ftp://ftp.gnu.org/pub/gnu/autoconf/](ftp://ftp.gnu.org/pub/gnu/autoconf/)
Warning: autoconf-2.59-12.noarch on centos

See Also:

autoconf

> autogen.sh

Results

$ wireshark-1.4.2$ ./autogen.sh
checking for python.
aclocal -I ./aclocal-fallback
libtoolize --copy --force
libtoolize: putting auxiliary files in ".".
libtoolize: copying file "/ltmain.sh"
libtoolize: Consider adding ‘AC_CONFIG_MACRO_DIR([m4])’ to configure.in and
libtoolize: rerunning libtoolize, to keep the correct libtool macros in-tree.
libtoolize: Consider adding ‘-I m4’ to ACLOCAL_AMFLAGS in Makefile.am.
autoheader
automake --add-missing --gnu
configure.in: installing "/ylwrap"
autoconf
Now type "/configure [options]" and "make" to compile Wireshark.

./configure --prefix=/opt/wireshark --with-pcap=/opt/libpcap/current -with-python

...  
checking for pkg-config... /usr/bin/pkg-config
checking for GTK+ - version >= 2.4.0... yes (version 2.20.1)
checking for pkg-config... (cached) /usr/bin/pkg-config
checking for GLIB - version >= 2.4.0... yes (version 2.24.1)
checking for GLIB - version >= 2.14.0... yes
checking whether GLib supports loadable modules... yes
...

> configure --prefix=/opt/wireshark --with-pcap=/opt/libpcap/current -with-python
The Wireshark package has been configured with the following options.
Build wireshark: yes
   Build tshark: yes
   Build capinfos: yes
   Build editcap: yes
   Build dumpcap: yes
   Build mergecap: yes
   Build text2pcap: yes
   Build iod12wrs: yes
   Build randpkt: yes
   Build dftest: yes
   Build rawshark: yes
Install dumpcap with capabilities: no
Install dumpcap setuid: no
   Use dumpcap group: (none)
   Use plugins: yes
   Use lua library: no
   Use python binding: yes

2.9. Drivers
Development tools, Release 2012.06.18

Build rtp_player : no
Use threads : no
Build profile binaries : no
Use pcap library : yes
Use zlib library : yes
Use pcre library : no (using GRegex instead)
Use kerberos library : yes (MIT)
Use c-ares library : no
Use GNU ADNS library : no
Use SMI MIB library : no
Use GNU crypto library : yes
Use SSL crypto library : no
Use IPv6 name resolution : yes
Use gnutls library : no
Use POSIX capabilities library : yes
Use GeoIP library : no

make 30 minutes to compile.

make install

|-- bin
  |-- capinfos
  |-- dftest
  |-- dumpcap
  |-- editcap
  |-- id12wrs
  |-- mergecap
  |-- randpkt
  |-- rawshark
  |-- text2pcap
  |-- tshark
  `-- wireshark
|-- lib
  |-- libwireshark.la
  |-- libwireshark.so -> libwireshark.so.0.0.1
  |-- libwireshark.so.0 -> libwireshark.so.0.0.1
  |-- libwireshark.so.0.0.1
  |-- libwiretap.la
  |-- libwiretap.so -> libwiretap.so.0.0.1
  |-- libwiretap.so.0 -> libwiretap.so.0.0.1
  |-- libwiretap.so.0.0.1
  |-- libwsutil.la
  |-- libwsutil.so -> libwsutil.so.0.0.0
  |-- libwsutil.so.0 -> libwsutil.so.0.0.0
  `-- libwsutil.so.0.0.0
    |-- plugins
    `-- 1.4.0rc2
      `-- 1.4.0rc2

276 Chapter 2. Development
2.9. Drivers

277
2.9. Drivers

279
Development tools, Release 2012.06.18

|-- dictionary.nomadix
|-- dictionary.nortel
|-- dictionary.ntua
|-- dictionary.openser
|-- dictionary.packeteer
|-- dictionary.patton
|-- dictionary.propel
|-- dictionary.prosoft
|-- dictionary.quiconnect
|-- dictionary.quintum
|-- dictionary.redback
|-- dictionary.redcreek
|-- dictionary.rfc2865
|-- dictionary.rfc2866
|-- dictionary.rfc2867
|-- dictionary.rfc2868
|-- dictionary.rfc2869
|-- dictionary.rfc3162
|-- dictionary.rfc3576
|-- dictionary.rfc3580
|-- dictionary.rfc4072
|-- dictionary.rfc4372
|-- dictionary.rfc4675
|-- dictionary.rfc4679
|-- dictionary.rfc4818
|-- dictionary.rfc4849
|-- dictionary.rfc5090
|-- dictionary.rfc5176
|-- dictionary.riverstone
|-- dictionary.roaringpenguin
|-- dictionary.shasta
|-- dictionary.shiva
|-- dictionary.slipstream
|-- dictionary.sofaware
|-- dictionary.sonicwall
|-- dictionary.springtide
|-- dictionary.starent
|-- dictionary.t_systems_nova
|-- dictionary.telebit
|-- dictionary.telkom
|-- dictionary.trapeze
|-- dictionary.tropos
|-- dictionary.tunnel
|-- dictionary.unisphere
|-- dictionary.unix
|-- dictionary.usr
|-- dictionary.utstarcom
|-- dictionary.valemount
|-- dictionary.versanet
|-- dictionary.vqp
|-- dictionary.walabi
|-- dictionary.waverider
|-- dictionary.wimax
|-- dictionary.wispr
|-- dictionary.xemia
|-- dictionary.xylan
|-- rawshark.html
Wireshark python  See Also:  
http://wiki.wireshark.org/Python

Extending Wireshark with Python  Note: This isn’t yet supported on Windows (see Bug 3500)
The projects aim is to give the possibility to developers to easily extend Wireshark with Python.
It is a project in development and therefore is experimental. It is better to not use this in production for now. It is good
though for prototyping as the syntax is rather concise.
It is better to have read doc/README.developer and doc/README.python before attempting to play with the Python API.

Requirements  You must have a valid Python environment (python >= 2.3) and ctypes. ctypes is part of the Python
package from the version 2.5. If you have an older version, you have to install it yourself.
Compile with Python support:

    ./configure --with-python

export LD_LIBRARY_PATH=/opt/libpcap/current:$LD_LIBRARY_PATH export PATH=/opt/wireshark/bin:$PATH

tcpdump and libpcap  See Also:  
- http://www.tcpdump.org/
http://www.tcpdump.org/  This is the official web site of:

• tcpdump, a powerful command-line packet analyzer;
• and libpcap, a portable C/C++ library for network traffic capture.

In this page, you’ll find the latest stable version of tcpdump and libpcap, as well as current development snapshots, a complete documentation, and information about how to report bugs or contribute patches.

Current Development Version  The current development version is freely accessible through the anonymous GIT server.

To checkout a copy of libpcap or tcpdump, do:

```bash
git clone git://bpf.tcpdump.org/tcpdump
git clone git://bpf.tcpdump.org/libpcap
```

**Warning:** can’t git clone `git://bpf.tcpdump.org/libpcap` on centos

One can then configure and compile the source via the normal GNU `autoconf` method.

You can also find a nightly update at git hub: libpcap and git hub: tcpdump and you are encouraged to do your initial pull from there.

You are also encouraged to submit patches in the form of git trees hosted on github or elsewhere.

Installation

```bash
./configure --prefix=/opt/libpcap/1.1
make
make install
```

pcap tree

```
|-- bin
 |  `-- pcap-config
|-- include
 |  |-- pcap
 |     |-- bluetooth.h
 |     |-- bpf.h
 |     |-- ipnet.h
 |     `-- namedb.h
 |     `-- pcap.h
 |     `-- sll.h
 |     `-- usb.h
 |     `-- vlan.h
 |     `-- pcap-bpf.h
 | `-- pcap-namedb.h
 | `-- pcap.h
|-- lib
 |  |-- libpcap.a
 |  `-- libpcap.so -> libpcap.so.1
 | `-- libpcap.so.1 -> libpcap.so.1.1
 | `-- libpcap.so.1.1
|-- share
 | `-- man
```
|-- man1
|   `-- pcap-config.1
|-- man3
|   |-- pcap.3pcap
|   |   |-- pcap_activate.3pcap
|   |   |-- pcap_breakloop.3pcap
|   |   |-- pcap_can_set_rfmon.3pcap
|   |   |-- pcap_close.3pcap
|   |   |-- pcap_compile.3pcap
|   |   |-- pcap_create.3pcap
|   |   |-- pcap_datalink.3pcap
|   |   |-- pcap_datalink_name_to_val.3pcap
|   |   |-- pcap_datalink_val_to_description.3pcap
|   |   |-- pcap_datalink_val_to_name.3pcap
|   |   |-- pcap_dispatch.3pcap
|   |   |-- pcap_dump.3pcap
|   |   |-- pcap_dump_close.3pcap
|   |   |-- pcap_dump_file.3pcap
|   |   |-- pcap_dump_flush.3pcap
|   |   |-- pcap_dump_fopen.3pcap
|   |   |-- pcap_dump_ftell.3pcap
|   |   |-- pcap_dump_open.3pcap
|   |   |-- pcap_file.3pcap
|   |   |-- pcap_fileno.3pcap
|   |   |-- pcap_findalldevs.3pcap
|   |   |-- pcap_fopen_offline.3pcap
|   |   |-- pcap_free_datalinks.3pcap
|   |   |-- pcap_freealldevs.3pcap
|   |   |-- pcap_freecode.3pcap
|   |   |-- pcap_get_selectable_fd.3pcap
|   |   |-- pcap_geterr.3pcap
|   |   |-- pcap_getnonblock.3pcap
|   |   |-- pcap_inject.3pcap
|   |   |-- pcap_is_swapped.3pcap
|   |   |-- pcap_lib_version.3pcap
|   |   |-- pcap_list_datalinks.3pcap
|   |   |-- pcap_lookupdev.3pcap
|   |   |-- pcap_lookupnet.3pcap
|   |   |-- pcap_loop.3pcap
|   |   |-- pcap_major_version.3pcap
|   |   |-- pcap_minor_version.3pcap
|   |   |-- pcap_next.3pcap
|   |   |-- pcap_next_ex.3pcap
|   |   |-- pcap_offline_filter.3pcap
|   |   |-- pcap_open_dead.3pcap
|   |   |-- pcap_open_live.3pcap
|   |   |-- pcap_open_offline.3pcap
|   |   |-- pcap_perror.3pcap
|   |   |-- pcap_sendpacket.3pcap
|   |   |-- pcap_set_buffer_size.3pcap
|   |   |-- pcap_set_datalink.3pcap
|   |   |-- pcap_set_promisc.3pcap
|   |   |-- pcap_set_rfmon.3pcap
|   |   |-- pcap_set_snaplen.3pcap
|   |   |-- pcap_set_timeout.3pcap
|   |   |-- pcap_setdirection.3pcap
|   |   |-- pcap_setfilter.3pcap
|   |   |-- pcap_setnonblock.3pcap

2.9. Drivers
Development tools, Release 2012.06.18

| |-- pcap_snapshot.3pcap
| |-- pcap_stats.3pcap
| |-- pcap_statustostr.3pcap
| `-- pcap_strerror.3pcap
|-- man5
| `-- pcap-savefile.5
`-- man7
     |-- pcap-filter.7
     `-- pcap-linktype.7

10 directories, 79 files

autoconf

> ./autogen.sh

Checking for python.
You must have autoconf 2.60 or later installed to compile Wireshark.
Download the appropriate package for your distribution/OS,
or get the source tarball at ftp://ftp.gnu.org/pub/gnu/autoconf/

Warning: autoconf-2.59-12.noarch on centos

Installation

> configure
> make
> make install

Checking the version

> autoconf --version

autoconf (GNU Autoconf) 2.68
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+/Autoconf: GNU GPL version 3 or later
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Written by David J. MacKenzie and Akim Demaille.

Warning: http://wiki.wireshark.org/CaptureSetup/USB
For versions of the kernel prior to 2.6.21, the only USB traffic capture mechanism available is a tex-
amism that limits the total amount of data captured for each raw USB block to about 30 bytes. There
change this without patching the kernel.

new linux kernel for wireshark

How to install a new kernel on centOS  See Also:

- http://www.howtoforge.com/kernel_compilation_centos
Development tools, Release 2012.06.18

- http://wiki.centos.org/HowTos/Custom_Kernel
- http://elrepo.org/tiki/kernel-ml
- http://people.centos.org/hughesjr/kernel-rt/i386/

Extract the sources

cd cd /dvcs_repositories/
gunzip linux-2.6.31.14.tar.gz
tar xvf linux-2.6.31.14.tar.gz -C /usr/src
cd /usr/src
ln -sf linux-2.6.31.14 linux
11

Results:

```text
drwxr-xr-x 2 root root 4096 jan 26 2010 debug
drwxr-xr-x 2 root root 4096 jan 26 2010 kernels
lrwxrwxrwx 1 root root 15 déc 17 09:07 linux -> linux-2.6.31.14
drwxrwxr-x 23 root root 4096 jui 5 19:11 linux-2.6.31.14
drwxr-xr-x 7 root root 4096 déc 16 16:57 redhat
```

Install the development tools

yum install gcc make bison ncurses-devel rpm-build

Results:

```
[rroot@houx src]# yum install gcc make bison ncurses-devel rpm-build
Loaded plugins: fastestmirror, priorities
Loading mirror speeds from cached hostfile
214 packages excluded due to repository priority protections
Setting up Install Process
Package gcc-4.1.2-48.el5.i386 already installed and latest version
Package 1:make-3.81-3.el5.i386 already installed and latest version
Package bison-2.3-2.1.i386 already installed and latest version
Package ncurses-devel-5.5-24.20060715.i386 already installed and latest version
Package rpm-build-4.4.2.3-20.el5_5.1.i386 already installed and latest version
Nothing to do
```

Keep the old config  >  `uname -r`

```
2.6.18-194.26.1.el5
```

copy the old config file

```
cd /usr/src/linux
cp /boot/config-`
```

2.9. Drivers
Development tools, Release 2012.06.18

[root@houx linux]# make menuconfig
HOSTCC  scripts/basic/fixdep
HOSTCC  scripts/basic/docproc
HOSTCC  scripts/basic/hash
HOSTCC  scripts/kconfig/conf.o
HOSTCC  scripts/kconfig/kxgettext.o
HOSTCC  scripts/kconfig/lxdialog/checklist.o
HOSTCC  scripts/kconfig/lxdialog/inputbox.o
HOSTCC  scripts/kconfig/lxdialog/textbox.o
HOSTCC  scripts/kconfig/lxdialog/util.o
HOSTCC  scripts/kconfig/lxdialog/yesno.o
HOSTCC  scripts/kconfig/mconf.o
SHIPPED  scripts/kconfig/zconf.tab.c
SHIPPED  scripts/kconfig/lex.zconf.c
SHIPPED  scripts/kconfig/zconf.hash.c
HOSTCC  scripts/kconfig/zconf.tab.o
HOSTLD  scripts/kconfig/mconf
scripts/kconfig/mconf arch/x86/Kconfig
  .config:684:warning: symbol value 'm' invalid for IP_DCCP_CCID3
  .config:685:warning: symbol value 'm' invalid for IP_DCCP_TFRC_LIB
  .config:1424:warning: symbol value 'm' invalid for FIXED_PHY
  .config:1731:warning: symbol value 'm' invalid for ISDN
  .config:2865:warning: symbol value 'm' invalid for RTC_INTF_SYSFS
  .config:2866:warning: symbol value 'm' invalid for RTC_INTF_PROC
  .config:2867:warning: symbol value 'm' invalid for RTC_INTF_DEV
  .config:2889:warning: symbol value 'm' invalid for DMA_ENGINE
  .config:2943:warning: symbol value 'm' invalid for GFS2_FS_LOCKING_DLM

make rpm (RPM of the new kernel)

[root@houx linux]# make rpm HOSTLD scripts/kconfig/conf scripts/kconfig/conf -s arch/x86/Kconfig
  make clean set -e; cd ..; ln -sf /usr/src/linux-2.6.31.14 kernel-2.6.31.14id3_17_12_2010
  /bin/sh -e /var/tmp/rpm-tmp.99947 + umask 022 + cd /usr/src/redhat/BUILD + cd kernel-2.6.31.14id3_17_12_2010 + rm -rf /var/tmp/kernel-2.6.31.14id3_17_12_2010-root + exit 0 rm
  /kernel-2.6.31.14id3_17_12_2010.tar.gz

Installing the new kernel
date; rpm -ivh /usr/src/redhat/RPMS/i386/kernel-2.6.31.14id3_17_12_2010-1.i386.rpm; date

Results

[root@houx linux]# date; rpm -ivh /usr/src/redhat/RPMS/i386/kernel-2.6.31.14id3_17_12_2010-1.i386.rpm; date
ven déc 17 10:49:53 CET 2010
Préparation... ########################################### [100%]
1:kernel ########################################### [100%]
ven déc 17 10:50:18 CET 2010

Confirming that the new kernel was installed

ls /boot

[root@houx linux]# ls /boot
config-2.6.18-194.11.4.el5 initrd-2.6.18-194.11.4.el5.img symvers-2.6.18-194.17.1.el5.gz vmlinuz-2.6.18-194.11.4.el5
config-2.6.18-194.17.1.el5 initrd-2.6.18-194.17.1.el5.img symvers-2.6.18-194.26.1.el5.gz vmlinuz-2.6.18-194.17.1.el5
config-2.6.31.14id3_17_12_2010 lost+found System.map-2.6.18-194.17.1.el5 vmlinuz-2.6.31.14id3_17_12_2010
extlinux
grub symvers-2.6.18-194.11.4.el5.gz System.map-2.6.31.14id3_17_12_2010

Creating the initrd for the new kernel

mkinitrd -v /boot/in

USB windows Utilities  See Also:

SniffUSB 2.0, un autre espion de bus USB (libre, windows XP only)  See Also:
http://www.pcausa.com/Utilities/UsbSnoop/default.htm

Windows Usbview  USBView is a free utility from Microsoft that displays the USB connection tree and shows the USB devices that are connected to it together with their configuration data. This is very useful for debugging USB enumeration errors.
USBView runs under Windows 98, ME, 2000 and XP. USBView is available for download as a .zip file by clicking http://www.ftdichip.com/Resources/Utilities/usbview.zip

Windows USBdevview  See Also:
  - http://www.nirsoft.net/utils/usb_devices_view.html
  - http://www.nirsoft.net/utils/usbdevview.zip

2.9. Drivers
USBDeview is a small utility that lists all USB devices that currently connected to your computer, as well as all USB devices that you previously used.

For each USB device, extended information is displayed: Device name/description, device type, serial number (for mass storage devices), the date/time that device was added, VendorID, ProductID, and more...

USBDeview also allows you to uninstall USB devices that you previously used, disconnect USB devices that are currently connected to your computer, as well as to disable and enable USB devices.

You can also use USBDeview on a remote computer, as long as you login to that computer with admin user.

**Windows Device Driver Uninstaller**  
CDM Uninstaller is a free application that can selectively remove Windows device drivers from the user’s system as specified by the device Vendor ID and Product ID. This application comes as a command driven application or as a GUI executable.

The readme for the command line version is available [CLI](#) and the readme for the GUI version can be viewed [GUI](#).

Both application come as zipped executables that need to be extracted prior to running, the command line version is available to download here and the GUI version is available here. Please refer to the respective readmes for running the application and operation guides.

- Download CDM Uninstaller (command line version)
- Download GUI CDM Uninstaller (GUI executable version)

**Windows device manager**  

**libusbx**

**See Also:**

- [http://www.libusbx.org/](http://www.libusbx.org/)

### Contents

- **libusbx**
  - Overview
  - Is libusbx a fork of libusb?

**Overview**  
libusbx is a library that provides generic access to USB devices. As a library, it is meant to be used by developers, to facilitate the development of applications that communicate with USB hardware.

It is portable: Using a single cross-platform API, it provides access to USB devices on Linux, OS X, Windows and OpenBSD.

It is user-mode: No special privilege or elevation is required for the application to communicate with a device.

It is version-agnostic: All versions of the USB protocol, from 1.0 to 3.0 (latest), are supported.

**Is libusbx a fork of libusb?**  
Yes it is.

The reason for the fork is that, despite having dedicated members, libusbx has still not been able to produce a new release for the past 2 years.

When a project fails to produce regular releases, we consider that you, its user, are paying the ultimate price. This is because it means that patches and new feature are being withheld and you end up wasting your time.
We are the same dedicated team who tirelessly tried to improve libusb but saw our efforts being wasted there.

After using libusbx for a while and after dealing with our great community, we hope that you will be as convinced as we are that there exists a better way!

libusb

libusb 1.0

libusb-1.0

libusb-1.0 is an almost-rewrite of the previous stable branch, libusb-0.1. It is a lightweight library that can be efficiently integrated into applications of any kind, with several new features. libusb-1.0 development is being lead by Daniel Drake. Contributions encouraged!

See Also:

pyUSB

libusb linux

See Also:

• http://git.libusb.org/?p=libusb.git;a=tree
• http://www.libusb.org/wiki/Libusb1.0
• http://pyusb/wiki.sourceforge.net/
• http://www.libusb.org/wiki/windows_backend
• https://lists.sourceforge.net/lists/listinfo/libusb-cvs
• http://libusb.sourceforge.net/api-1.0/

libusb is a library for USB device access from Linux userspace.

libusb 1.0 linux

C library for writing portable USB drivers in userspace.

Sources

• http://git.libusb.org/
• http://git.libusb.org/?p=libusb.git;a=summary

http://libusb.sourceforge.net/api-1.0/

Introduction

libusb is an open source library that allows you to communicate with USB devices from userspace.

It’s possible to write USB driver is userspace. See the USB driver API moves to EXPORT_SYMBOL_GPL (February 2008)

This documentation is aimed at application developers wishing to communicate with USB peripherals from their own software. After reviewing this documentation, feedback and questions can be sent to the libusb-devel mailing list.

This documentation assumes knowledge of how to operate USB devices from a software standpoint (descriptors, configurations, interfaces, endpoints, control/bulk/interrupt/isochronous transfers, etc).
See Also:

- http://sourceforge.net/projects/libusb/files/libusb-1.0/
- http://git.libusb.org/
- http://git.libusb.org/?p=libusb.git;a=summary
- http://www.linux-mips.org/wiki/The_perfect_patch
- http://userweb.kernel.org/~akpm/stuff/tpp.txt (inspiration for the above)
- http://lkml.indiana.edu/hypermail/linux/kernel/0801.0/0373.html
- http://lwn.net/Articles/160191/
- http://www.usbmadesimple.co.uk/index.html

Library features

- All transfer types supported (control/bulk/interrupt/isochronous)
- 2 transfer interfaces:
  1. Synchronous (simple)
  2. Asynchronous (more complicated, but more powerful)
- Thread safe (although the asynchronous interface means that you usually won’t need to thread)
- Lightweight with lean API
- Compatible with libusb-0.1 through the libusb-compat-0.1 translation layer

libusb and root permission (libusb and udev)


One issue about libusb under Linux and BSDs is how to run the program without using the root privilege.
Under Linux, the standard solution is to use udev rules.

Here are some good references about udev:

- Writing udev rules http://www.reactivated.net/udevrules.php
- Proper place to ask questions about udev rules http://vger.kernel.org/vger-lists.html#linux-hotplug

libusb versions

libusb versions

libusb 1.0.9  See Also:

- Wednesday, March 30, 2011 from Ludovic Rousseau
libusb-1.0.9 soon Fri, Jul 2, 2010 at 4:19 PM?

El 02/07/10 05:19, Ludovic Rousseau escribió:

> Hello Daniel and all,
> > Bugs have been reported and either corrected or patches are available
> > since libusb-1.0.8 is out. Is there a plan to release libusb-1.0.9
> > with all the improvements?

Thanks for keeping on top of things. I’ll try and get to it soon. At the moment I don’t have much connectivity which is holding me back. Next week should be better.

libusb-1.09 will be able to integrate the Windows backend

I would really hope that libusb-1.0.9 will be able to integrate the Windows backend even though the Windows backend will really be labeled as experimental or beta.

libusb 1.0.8


libusb-1.0 has a new API and, in particular, an asynchronous API. Using the asynchronous API it is possible to cancel an ongoing call. This is exactly what is done when TAG_IFD_STOP_POLLING_THREAD is requested by pcsd to the driver.

But libusb-1.0 is relatively new and has bugs. The latest stable version 1.0.8 is not yet bug free and I found some bugs in particular cases. Bugs have been reported to the libusb project and most are already corrected.

ChangeLog
Development tools, Release 2012.06.18

commit a04cbb0095a78aed3f65af06c46069d0ac184a
Author: Daniel Drake <dan@reactivated.net>
Date:   Tue May 4 19:01:59 2010 -0300

v1.0.8 release

commit 53b47299531974fa8901b74a163f3c8ebae4ee9
Author: Daniel Drake <dan@reactivated.net>
Date:   Thu Apr 22 18:12:51 2010 -0300

Linux: don’t set SHORT_NOT_OK on bulk out URBs (#20)

Setting this flag is illegal, and the behaviour we’re looking for
is already in place for host-to-device transfers without this flag.

commit 1519828b7eelfce46d4c51fc097d52e01b8e0bb4
Author: Daniel Drake <dan@reactivated.net>
Date:   Tue Apr 20 20:14:09 2010 -0300

Linux: Handle failure to read active configuration during enumeration

commit 45168627cc15aee3875192f34286110dbbd27095
Author: Nathan Hjelm <hjelmn@me.com>
Date:   Tue Apr 20 19:43:06 2010 -0300

Darwin: don’t reuse cached descriptors during enumeration

Descriptor reuse was causing scans to return invalid information when
the device at a location has changed.

libusb installation

libusb installation

libusb installation  See Also:

• http://git.libusb.org/?p=libusb.git;a=summary.js=1
• Wednesday, March 30, 2011 from Ludovic Rousseau
• Prerequisite
• libusb maintenance problem

Warning: libusb must be installed before libpcsclite.

Installation on GNU/Linux  We must be root to install the libusb libray.

Options for /configure  Optional Features:

--disable-option-checking  ignore unrecognized --enable/--with options
--disable-FEATURE  do not include FEATURE (same as --enable-FEATURE=no)
--enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
Development tools, Release 2012.06.18

--enable-silent-rules    less verbose build output (undo: ‘make V=1’)  
--disable-silent-rules   verbose build output (undo: ‘make V=0’)  
--disable-dependency-tracking  do not reject slow dependency extractors  
--enable-dependency-tracking  speeds up one-time build  
--enable-shared[=PKGS]     build shared libraries [default=yes]  
--enable-static[=PKGS]      build static libraries [default=yes]  
--enable-fast-install[=PKGS] optimize for fast installation [default=yes]  
--disable-libtool-lock    avoid locking (might break parallel builds)  
--enable-timerfd          use timerfd for timing (default auto)  
--disable-log             disable all logging  
--enable-debug-log        enable debug logging (default n)  
--enable-examples-build   build example applications (default n)  

Optional Packages:

--with-PACKAGE[=ARG] use PACKAGE [ARG=yes]  
--without-PACKAGE do not use PACKAGE (same as --with-PACKAGE=no)  
--with-pic try to use only PIC/non-PIC objects [default=use both]  
--with-gnu-ld assume the C compiler uses GNU ld [default=no]  

Some influential environment variables:

CC    C compiler command  
CFLAGS C compiler flags  
LDFLAGS linker flags, e.g. -L<lib dir> if you have libraries in a nonstandard directory <lib dir>  
LIBS   libraries to pass to the linker, e.g. -l<library>  
CPPFLAGS (Objective) C/C++ preprocessor flags, e.g. -I<include dir> if you have headers in a nonstandard directory <include dir>  
CPP    C preprocessor  

Use these variables to override the choices made by configure or to help it to find libraries and programs with non-standard names/locations.

Report bugs to the package provider.

Install the development version (git version)

wget http://git.libusb.org/?p=libusb.git;a=tree;hb=HEAD  
cd libusb-HEAD-be523f1  
./autogen.sh  
./configure  
make  
make install

maketi install

[root@agave latest]# make install  
Making install in libusb  
make[1]: entrant dans le répertoire « /tmp/latest/libusb »  
make[2]: entrant dans le répertoire « /tmp/latest/libusb »  
test -z "/usr/local/lib" || /bin/mkdir -p "/usr/local/lib"  
/bin/sh ../.libtool --mode=install /usr/bin/install -c libusb-1.0.1a "/usr/local/lib"

2.9. Drivers 293
Development tools, Release 2012.06.18

libtool: install: /usr/bin/install -c .libs/libusb-1.0.so.0.0.0 /usr/local/lib/libusb-1.0.so.0.0.0
libtool: install: (cd /usr/local/lib && ln -s -f libusb-1.0.so.0.0.0 libusb-1.0.so.0 || rm -f libusb-1.0.so.0.0.0)
libtool: install: (cd /usr/local/lib && ln -s -f libusb-1.0.so.0.0.0 libusb-1.0.so || rm -f libusb-1.0.so.0.0.0)
libtool: install: /usr/bin/install -c .libs/libusb-1.0.lai /usr/local/lib/libusb-1.0.la
libtool: install: /usr/bin/install -c .libs/libusb-1.0.a /usr/local/lib/libusb-1.0.a
libtool: install: chmod 644 /usr/local/lib/libusb-1.0.a
libtool: install: ranlib /usr/local/lib/libusb-1.0.a
libtool: finish: PATH="/usr/lib/qt-3.3/bin:/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/sbin" ldconfig -n /usr/local/lib

Libraries have been installed in:
/usr/local/lib

If you ever happen to want to link against installed libraries in a given directory, LIBDIR, you must either use libtool, and specify the full pathname of the library, or use the '-LLIBDIR' flag during linking and do at least one of the following:
- add LIBDIR to the 'LD_LIBRARY_PATH' environment variable during execution
- add LIBDIR to the 'LD_RUN_PATH' environment variable during linking
- use the '-Wl,-rpath -Wl,LIBDIR' linker flag
- have your system administrator add LIBDIR to '/etc/ld.so.conf'

See any operating system documentation about shared libraries for more information, such as the ld(1) and ld.so(8) manual pages.

----------------------------------------------------------------------
test -z "/usr/local/include/libusb-1.0" || /bin/mkdir -p "/usr/local/include/libusb-1.0"
/usr/bin/install -c -m 644 libusb.h "/usr/local/include/libusb-1.0"
make[2]: quittant le répertoire « /tmp/latest/libusb »
maker[1]: quittant le répertoire « /tmp/latest/libusb »
Making install in doc
make[2]: entrant dans le répertoire « /tmp/latest/doc »
maker[2]: Rien à faire pour « install-exec-am ». 
maker[2]: Rien à faire pour « install-data-am ». 
maker[2]: quittant le répertoire « /tmp/latest/doc »
maker[1]: quittant le répertoire « /tmp/latest/doc »
maker[2]: entrant dans le répertoire « /tmp/latest »
maker[2]: entrant dans le répertoire « /tmp/latest »
maker[2]: Rien à faire pour « install-exec-am ».
test -z "/usr/local/lib/pkgconfig" || /bin/mkdir -p "/usr/local/lib/pkgconfig"
/usr/bin/install -c -m 644 libusb-1.0.pc "/usr/local/lib/pkgconfig"
make[2]: quittant le répertoire « /tmp/latest »
maker[1]: quittant le répertoire « /tmp/latest »
[root@agave latest]#

Install in /opt/libusb/version The last version is the libusb1.0.9 rc3.

See Also:
- http://git.libusb.org/?p=libusb.git;a=commit:h=f07a4a78533b44d124dfe06c6f42afa7fb267359;js=1

download under /tmp After dowload, uncompress the files under /tmp.

./autogen.sh (as root user)
cd /tmp/libusb-xxxx
./autogen.sh

configure (as root user)

cd /tmp/libusb-xxxx
./configure --prefix=/opt/libusb/xxxx --enable-debug-log --enable-examples-build
make

sudo make install

Libraries have been installed in:
   /opt/libusb/1.0.9/lib

If you ever happen to want to link against installed libraries in a given directory, LIBDIR, you must either use libtool, and specify the full pathname of the library, or use the ‘-LLIBDIR’ flag during linking and do at least one of the following:

- add LIBDIR to the ‘LD_LIBRARY_PATH’ environment variable during execution
- add LIBDIR to the ‘LD_RUN_PATH’ environment variable during linking
- use the ‘-Wl,-rpath -Wl,LIBDIR’ linker flag
- have your system administrator add LIBDIR to ‘/etc/ld.so.conf’

Link to the the ‘current’ libusb version

cd /opt/libusb
ln -s 1.0.9 current
ls -als

libusb maintenance problem  See Also:

libusb july 2011 maintenance problem  See Also:

http://sourceforge.net/mailarchive/forum.php?thread_name=CA572755.147C1%25kustaa.nyholm%40planmeca.com&forum_name=libusb-devel

From: Pete Batard <pete@ak...> - 2011-07-28 09:21
Just a couple of quick questions:

1. Are you reading the libusb mailing list at all?
2. If not, do you plan to be involved with libusb ever again?
3. If not, could you please appoint a new project lead for libusb, with the ability to appoint maintainers?

In case you haven’t been following the situation, because of the lack of release, users are actively ditching libusb, and with a single project maintainer, arbitration seems unattainable.

We need at least 3 different people, with either administrative or maintenance authority, to be actively involved in libusb in order to stop the current deadlock. As such your action is very much required.

Regards,
/Pete

libusb may 2011 maintenance problem  See Also:

- https://github.com/LudovicRousseau/libusb

I an previous blog article I talked about the situation of libusb:

- a stable libusb has not been released since May 2010 (exactly one year ago)
- bugs are still open even if they have patches attached in the bug report

I already provide a git snapshot of the upstream (official) repository in the file libusb-1.0.8-git.tar.bz2 from http://ludovic.rousseau.free.fr/softwares/pcsc-lite/.
I now also provide a git snapshot of my own github version with ticket #56 fixed in the file libusb-1.0.8-github.tar.bz2.

libusb march 2011 libusb maintenance problem  See Also:

- [http://sourceforge.net/mailarchive/forum.php?thread_name=AANLkTinp3vS0bemKC4Us6-23g4QyaE%2BgRjS5MyuT1HNr%40mail.gmail.com&forum_name=libusb-devel](http://sourceforge.net/mailarchive/forum.php?thread_name=AANLkTinp3vS0bemKC4Us6-23g4QyaE%2BgRjS5MyuT1HNr%40mail.gmail.com&forum_name=libusb-devel)

The latest version 1.0.8 of libusb-1.0 was released in May 2010 (nearly a year ago). Since then many bugs have been reported and most have been fixed in the git repository. The problem is that no new stable or 1.0.9 version has been released since May 2010. So if you suffer from a bug in libusb it is not easy to update it.
To ease the use of a more up-to-date version of libusb-1.0 I made a snapshot of the version in the git repository and provide it on my web page of beta versions or http://ludovic.rousseau.free.fr/softwares/pcsc-lite/

libusb august 2011 maintenance problem   See Also:
http://sourceforge.net/mailarchive/forum.php?thread_name=CAGstE8BvGF-%3Dku0XGzg74aQj3PzUFj6yMT88FeJNBxHy0hhXrQ%3D-devel

Hello Peter and others,

Once again a user got hit by a bug in libusb 1.0.8 that is _already_ fixed.

The user worked on the problem and wrote a patch for my program. The problem is not in my program but in libusb. Upgrading libusb solved the problem.

He lost time identifying the problem.
He lost time writing a patch.
He lost time filling a bug [1].
I lost time answering the bug.
I lose time writing this email.

So much time lost that could be used for something else.

Peter, if you care (a little) about your users then please release libusb-1.0.9 as it is now. Many minutes/hours will be saved for many people.

Thanks

See Also:
http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=637022

Other libraries using libusb
libfprint linux library

See Also:

- http://www.reactivated.net/fprint/wiki/Main_Page
- http://www.reactivated.net/fprint/api/

libfprint  libfprint is the centre of our efforts. libfprint is the component which does the dirty work of talking to fingerprint reading devices, and processing fingerprint data.

If you’re a user, you probably aren’t interested in libfprint, instead you want to find some software which uses libfprint (see the integration project).

If you’re an application developer looking to add support for some kind of fingerprinting to your software, libfprint is exactly what you are looking for. It provides a simple API for you to enroll fingerprints and then identify users later on.

Download

- http://www.reactivated.net/fprint/wiki/Download
- http://cgit.freedesktop.org/libfprint/libfprint/

Clone

- git://anongit.freedesktop.org/libfprint/libfprint
- ssh://git.freedesktop.org/git/libfprint/libfprint

Integration  The integration project details our efforts to integrate libfprint with existing applications, so that users can use their fingerprint reading hardware.

fprint_demo  fprint_demo is a simple GUI application used to demonstrate and test libfprint’s capabilities.
fprint_demo homepage  fprintd is a daemon that provides fingerprint scanning functionality over D-Bus.

libfprint linux library  See Also:

- http://www.reactivated.net/fprint/api/
- http://www.reactivated.net/fprint/wiki/Libfprint
- http://www.reactivated.net/fprint/wiki/Main_Page
- http://www.reactivated.net/fprint/wiki/Libfprint:Supported_devices
- http://www.reactivated.net/fprint/wiki/Driver_quality

libfprint is an open source software library designed to make it easy for application developers to add support for consumer fingerprint readers to their software.

Features

- Written in C
- Licensed as LGPL-2.1
- Depends on libusb for USB communication and glib
- Primarily developed for linux, but should be fairly portable
- Offers a single API to application developers to access the entire range of supported devices
- Supports imaging - downloading live fingerprint scans from the device
- Includes image processing/matching code
- Supports enrollment/verification - enrolling a print from a known user, and then later comparing a live scan to the enrolled print

Fingerprint driver quality  See Also:

http://www.reactivated.net/fprint/wiki/Driver_quality

libfprint includes a number of drivers which are all compiled and included by default. This will not change, because we want to promote further development of all drivers and ensure they continue to compile OK (even the ones that are not so high quality).

However, if you’re distributing libfprint, you may want to selectively disable the drivers which are known not to work so well. This page will guide you on those decisions.

In the context of this page, we refer to “quality” from a user perspective how well does the driver accept fingers that match, and reject ones that don’t? It does not necessary relate to the code quality of that driver, which may be different!
The quality of most drivers could be improved by the methods described at Imaging performance.

libfprintd  See Also:
   • http://github.com/dsd/fprintd
   • http://www.reactivated.net/fprint/wiki/Fprintd

 Daemon to offer libfprint functionality over D-Bus Currently in early stages. Might eat your kangaroo.

 Written in C. Requires bleeding edge libfprint (libusb-1.0 port).

 Licensed under the GPL version 2 (see COPYING).

 An experimental PAM login module is included in the ‘pam’ directory. This will be moved to a separate package once the system has matured.

 API use cases
   • User wants to use the fingerprint reader, and enroll his fingerprints, or remove some fingerprints from the database
   • Administrator wants to enroll fingerprints for a particular user, or remove fingerprints for a particular user
   • Laptop/desktop authentication: * Check for fingerprint devices * Check whether a particular user has any fingerprints enrolled * Verify a fingerprint for a particular user, or, if the device supports it, verify that the fingerprint matches against any of the fingerprints enrolled
   • Point Of Sale authentication (in a bar, the fingerprint reader is used to see who accesses a particular point of sale/till, in place of PIN code authentication and/or tokens) Given a list of users, verify which one has scanned their finger

 Contributions encouraged. Please submit patches on the mailing list.

 fprint_demo  See Also:
   • http://github.com/dsd/fprint_demo

 fprint_demo is a simple GTK+ application to demonstrate and test libfprint’s capabilities. It is written in C and licensed under the GNU GPL v2.

 It provides access to many of the features offered by the backing library, libfprint.

 libfprint linux library  See Also:
 http://www.reactivated.net/fprint/wiki/Libfprint

 libfprint is an open source software library designed to make it easy for application developers to add support for consumer fingerprint readers to their software.

 Features
   • Written in C
   • Licensed as LGPL-2.1
   • Depends on libusb for USB communication and glib
   • Primarily developed for linux, but should be fairly portable
   • Offers a single API to application developers to access the entire range of supported devices
• Supports imaging - downloading live fingerprint scans from the device
• Includes image processing/matching code
• Supports enrollment/verification - enrolling a print from a known user, and then later comparing a live scan to the enrolled print

libfprint linux library  See Also:
http://github.com/dsd/fprintd

libfrprint news

2010-08-19 15:28:12 GMT, New releases, new repos, new substitute maintainer


From: Bastien Nocera <hadess <at> hadess.net>
Subject: New releases, new repos, new substitute maintainer
Newsgroups: gmane.linux.fprint
Date: 2010-08-19 15:28:12 GMT (5 hours and 32 minutes ago)

Heya,

Daniel is very busy, and as such, hasn’t been able to handle patches and requests coming in from users and developers. With his approval, I now moved the canonical git repositories to freedesktop.org, and made releases so people can stop shipping pre-release tarballs, and git snapshots.

The most significant changes from the pre-releases are the new UPEK EikonII driver, from Jorge Suarez de Lis, and Guido Grazioli’s work on updating the AES1610 driver.

If you have patches that are still lingering somewhere, let me know, and we can start looking at them.

You’ll also notice that fprint demo and pam_fprint haven’t moved. I consider them both obsoleted by front-ends to fprintd, though somebody is more than welcome doing work on those.

Finally, we have a bugzilla coming up, see this bug for the progress: https://bugs.freedesktop.org/show_bug.cgi?id=29610 and I’ll be updating the reactivated.net Wiki as soon as Daniel gives me access to it.

Releases:

• http://freedesktop.org/~hadess/libfprint-0.2.0.tar.bz2
• http://freedesktop.org/~hadess/fprintd-0.2.0.tar.bz2

git web interface:

• http://cgit.freedesktop.org/libfprint/libfprint/
• http://cgit.freedesktop.org/libfprint/fprintd/

libusb.h

core.c
libusb tutorials

- http://www.libusb.org/wiki/Libusb1.0
- http://libusb.sourceforge.net/api-1.0/
- http://pyusb.sourceforge.net/docs/1.0/tutorial.html

libusb examples

- http://www.reactivated.net/fprint/wiki/

libusb isochronous examples

- http://git.tuxfamily.org/microdiausp/microdiauserspace.git
- https://sourceforge.net/scm/?type=git&group_id=8157


In the article, there is an interesting example of Isochrons USB Transfer example based on lpcusb. So I use lpc21isp and download the code to my Olimex LPC-P2148 board which the developers are also using. They have also the Linux host example based on usbfs which seems to work right out of the box. Now I need to read the firmware in more detail.

There are also quite some examples from the psas site. http://psas.pdx.edu/
lpcusb (get the svn version) http://sourceforge.net/projects/lpcusb
lpc21isp http://sourceforge.net/projects/lpc21isp

libusb news

Wednesday, March 30, 2011 from Ludovic Rousseau See Also:

- http://ludovic.rousseau.free.fr/softwares/pcsc-lite/

My CCID driver uses libusb-1.0 to talk to USB readers.

libusb evolution stalled The latest version 1.0.8 of libusb-1.0 was released in May 2010 (nearly a year ago). Since then many bugs have been reported and most have been fixed in the git repository.

The problem is that now new stable or 1.0.9 version has been released since May 2010. So if you suffer from a bug in libusb it is not easy to update it.

libusb git snapshot To ease the use of a more up-to-date version of libusb-1.0 I made a snapshot of the version in the git repository and provide it on my web page of beta versions or http://ludovic.rousseau.free.fr/softwares/pcsc-lite/

The files in the libusb git repository have not evolved in the last 6 months so I do not expect to have to make a new snapshot within the next 6 months.

I sent a mail on the libusb mailing list.

2.9. Drivers
**Conclusion**  So before reporting a bug in libccid first try a newer version of libusb. If the bug is at the communication level it may already be solved in the libusb snapshot version.

**[Libusb-devel] GIT snapshot available**

From: Ludovic Rousseau <ludovic.rousseau@gm...> - 2011-03-30 07:10

Hello,

Some users report problems with my application but the problems are in fact in libusb. Often/always the problems have already been fixed but the code is not yet released in a stable libusb (1.0.9) version.

So I decided to distribute a .tar.bz2 [1] from GIT so that my users can rebuild libusb themself without using git and autotools.

The last commit in the libusb.git repo dates back to October 2010 (6 months ago). And I do not expect any change in the next 6 months. So a nightly, weekly, monthly or yearly snapshot is the same. Updating my snapshot to a newer version should not take more than 5 minutes each year.

The libusb project is making me lose my time, energy and motivation. Not releasing a stable version has external costs on many people (me and my users at least).

Bye

libusb windows

libusb library

libusb  See Also:

- http://libusb.org/wiki/windows_backend

About  The aim of this project is to bring a Windows backend to the mainline libusb 1.0 branch, so that libusb 1.0 can easily be used on Windows platforms.

Status  The Windows backend is now in pre-release stages. As of 2010.02.17, it include all of the main libusb v1.0 features, for both HID and WinUSB devices, apart from isochronous transfers (which is a limitation from the WinUSB Microsoft API). It currently supports all Windows platforms from Windows XP, including 64 bit versions of Windows (Windows 2003 is untested).

The benefits of this WinUSB backend is that it will work for 64bit Windows like Vista and Windows 7. The WinUSB backend will not support isochronous transfer. So it is not a complement replacement of libusb-win32 yet.

libusb-winusb
libusb-winusb will be integrated to the main libusb-1.0 tree (using git). Once it is mature, I think it will be very good replacement for libusb-win32. More backend will probably be added, like the HID backend (HID support) and the libusb-win32 device driver backend (isochronous support, Windows 2k support).

Please try this branch if you are interested.

Xiaofan http://mcuee.blogspot.com

libusb_news

Testers welcome for libusb-win32 snapshot release 20100505

from Xiaofan Chen <xiaofanc@gmail.com>
to libusb-devel@lists.sourceforge.net
date Thu, May 6, 2010 at 1:39 AM
mailing list libusb-devel@lists.sourceforge.net Filter messages from this mailing list
mailed-by lists.sourceforge.net
unsubscribe Unsubscribe from this mailing-list

Travis Robinson has pushed the new snapshot release 20100505, Please go to libusb-win32 project page at Sourceforge to download this release. http://sourceforge.net/projects/libusb-win32/

Major changes:

1. Travis is now the lead developer for libusb-win32 project.
2. License is updated to LGPL V3 (library) and GPL V3 (driver, installer, test program).
3. Fixed bug 2658937 (Filter driver should not be power policy owner). This should make the filter safer to use under XP/Vista/Win7.
4. If running as a device driver, default configuration will be set. If running as a filter driver, no configuration will be set.
5. Drop support for Windows 98SE and Windows ME. Win2k will the minimum required OS version.
6. Bump up the version (1.1.14.0) to facilitate Microsoft WHQL submission.
7. Change the build system to Microsoft WDK.

We have done reasonable tests under different operating systems for both the filter driver mode and the device driver mode. We are reasonably sure that the new filter driver would not cause BSODs or make the USB subsystem malfunction. Nevertheless, this is a snapshot release and quite possibly there are still bugs so that you still need to carry out due diligence especially with the filter driver.

Early adopters are welcome to test this new snapshot release and send in success/failure reports.

The support facilities are described here. http://sourceforge.net/projects/libusb-win32/support

Mailing list is the preferred support channel. Bug report, patches and feature request are also welcome through the tacker system. The forum and support requests at SourceForge will be closed in the future.

The project website is now redirected to the following Trac Wiki. http://sourceforge.net/apps/trac/libusb-win32/wiki
Identify the physical USB ports used by a driver handle

from Xiaofan Chen <xiaofanc@gmail.com>
reply-to libusb-win32-devel@lists.sourceforge.net
to libusb-win32-devel@lists.sourceforge.net
date Fri, Feb 19, 2010 at 3:47 PM
subject Re: [Libusb-win32-devel] Identify the physical USB ports used by a driver handle
mailing list <libusb-win32-devel.lists.sourceforge.net> Filter messages from this mailing list
mailed-by lists.sourceforge.net
unsubscribe Unsubscribe from this mailing-list

On Fri, Feb 19, 2010 at 6:10 PM, Patrik Thalin <patrik.thalin@stericsson.com> wrote: > Hi all, > > I am looking for a solution to identify the physical USB ports used by a device. I have looked at devcon in WDK to find the DeviceID eg. USBVID_xxxx&PID_yyyy6&6B42B9C&0&4. This seem to be a unique for the port used. I have successfully duplicated this to my application. By calling SetupDiGetDeviceRegistryProperty and SetupDiGetDeviceInstanceId. > But I can’t find a way to associate this to the driver handle. Can this be done? > > Any other suggestion on how identify the port is also welcome! Note that I can not use a serial number in the device it has to be unique for the USB port on the computer. Also note that I have several indentical units > connected. >

I do not know the answer. You may have better luck trying out some other lists. The best one may be this newsgroup. microsoft.public.development.device.drivers http://groups.google.com/group/microsoft.public.development.device.drivers/topics

But it is said to be non-trivial for any OS. http://old.nabble.com/Re%3A-how-to-get-device-info-from-mount-point-p16593540.html


– Xiaofan http://mcuee.blogspot.com - Show quoted text -

Question: libusb for 64-bit systems

from Don Raikes <DON.RAIKES@oracle.com>
reply-to libusb-win32-devel@lists.sourceforge.net
to libusb-win32-devel@lists.sourceforge.net
date Fri, Feb 26, 2010 at 9:16 PM
subject [Libusb-win32-devel] libusb for 64-bit systems
mailing list <libusb-win32-devel.lists.sourceforge.net> Filter messages from this mailing list
mailed-by lists.sourceforge.net

Hello, I am new to this list, and in fact haven’t used libusb-win32 at all yet. I am upgrading to a 64-bit machine with windows 7 64-bit on it over this weekend. My question is: is there a version of libusb-win that will work for windows 64-bit operating systems?

Response 1 There are many discussions of this topic in the mailing list. You can check the archive.

Current version libusb-win32 already works under XP64. It will also work under Vista 64 and Windows 7 64bit if you play some tricks. The major problems is the digital signing of the kernel driver. For example, this is a whole thread about this issue. http://old.nabble.com/Windows-2008-x64-td24807435.html

Now there is finally a viable alternative to libusb-win32: the new libusb 1.0 Windows backend, it is now under pre-release mode but there are issues to be sorted out so that it is integrated into the main libusb 1.0 tree. It uses WinUSB and/or Windows HID as the backend. Unless your device is using isochronous transfer, WinUSB should work for you and it will work under all the 64bit Windows system. URL: http://libusb.org/wiki/windows_backend

I have tried it and it works fine for me (both WinUSB backend and HID backend). There are still issues (multi-thread and some other issues) to be sorted out though. If you are interested, give it a try and it will help if you subscribed to libusb mailing list. If you encountered issues, do not hesitate to raise the questions there. https://lists.sourceforge.net/lists/listinfo/libusb-devel Archive: http://old.nabble.com/LibUSB-f14231.html

Response 2  Of course, since one company has done the digital signing. If you are doing things for your employer (Oracle), then you can probably get it (paid by your company). The best is of course to contribute the things back to the community and allow it to be used with their projects. http://old.nabble.com/Building-Libusb-for-64-bit-Windows-Vista-and-Seven-td27260978.html

I am not sure if Pruftechnik will allow others to use their driver (WHQLed) for other projects. If yes, then people can use it as well.

The driver is now ptlibusb0.sys (not libusb0.sys). According to the following libusb-win32 forum thread (log-in to access it), you need to change the driver name but it seems the dll is still okay. https://sourceforge.net/projects/libusb-win32 fora1ums/forum/266688/topic/3536300

Quote: It took me quite some effort and testing to come to this solution. I could have lived with running windows in testmode, but this solution is of course much better.

Now we only need to find out, which information in the inf file can be changed without breaking the certification. Some things can be renamed, but not everything.

Ciao, Steffen

PS: While the driver and dll are renamed from libusb0 to ptlibusb0 which would make it necessary to change the application which uses the lib, it seems that the original libusb0.dll can be used with the ptlibusb0.sys, so even existing binaries linking to libusb0.dll can work with this driver.

Response 3  On Tue, Mar 2, 2010 at 6:36 AM, Don Raikes <DON.RAIKES@oracle.com> wrote: > Hello, > So I am in the process of installing libwinbackend. >> I have downloaded the files, and went into device manager to >> determine the vendor/product id for the device. >> I see the device listed in device manager as >> “alva satellite 544” >> but it has no device driver installed. I cannot figure >> out how to get the necessary vid and pid information. I checked >> all three > tabs and none of them listed any numbers that would indicate these values.

There is information about VID/PID in device manager (called hardware id).

Anyway, you should use USBView instead since it is easier and give you more information. It is part of WDK.
http://www.microsoft.com/whdc/devtools/wdk/wdkpkg.mspx


You can of course get the information from Linux as well

lsusb -vvv

http://libusbdotnet.sourceforge.net/V2/Index.html  libusbdotnetV2 seems to have some big improvement.

It supports libusb 1.0 under Linux and libusb-win32/WinUSB under Windows. Maybe this will be of some interests to some C# or DotNet users. I can only barely use C myself.

I found out this from the following Microchip forum thread. http://www.microchip.com/forums/tm.aspx?m=480008
Some nice features I tested: 1) INF wizard to generate the INF file for WinUSB and libusb-win32 2) Device notification
http://libusbdotnet.sourceforge.net/V2/html/8c7cc7dc-5b65-4fab-b2a2-54cf0b727a19.htm

Example output if I reset the device:

```
[DeviceType:DeviceInterface] [EventType:DeviceRemoveComplete]
FullName:USB#VID_0925&PID_1456#5&207b166d&0&7#{a5dcbf10-6530-11d2-901f-00c04fb951ed}
Vid:0x0925
Pid:0x1456
SerialNumber:5&207b166d&0&7
ClassGuid:a5dcbf10-6530-11d2-901f-00c04fb951ed

[DeviceType:DeviceInterface] [EventType:DeviceArrival]
FullName:USB#VID_0925&PID_1456#5&207b166d&0&7#{a5dcbf10-6530-11d2-901f-00c04fb951ed}
Vid:0x0925
Pid:0x1456
SerialNumber:5&207b166d&0&7
ClassGuid:a5dcbf10-6530-11d2-901f-00c04fb951ed
```

> Some nice features I tested:
> 1) INF wizard to generate the INF file for WinUSB and libusb-win32
> 2) Device notification

For what is worth, both of these features will be added to libusb after our first release, and there’s actual active
development going on for the inf/autoinstaller feature (that is, when other libusb stuff leaves enough time for that):
See
http://git.libusb.org/?p=libusb-pbatard.git;a=shortlog;h=refs/heads/winusb-autoinstall

But being able to check working auto-notification code on Windows might prove quite helpful actually ;)

See Also:

```
usb.ids
```

Unified .inf

from Graeme Gill <graeme2@argyllcms.com>
reply-to graeme@argyllcms.com
to libusb-devel <libusb-devel@lists.sourceforge.net>
date Fri, Mar 19, 2010 at 6:12 AM
subject [Libusb-devel] Unified .inf
mailing list libusb-devel@lists.sourceforge.net Filter messages from this mailing list
mailed-by lists.sourceforge.net unsubscribe Unsubscribe from this mailing-list

Here <http://www.argyllcms.com/example.inf> is an example of a unified libusb0.sys & WinUSB .inf file, that I’ve had some success with, when combined with my libusb V1.0 libusb0.sys support.

You can set which driver install (libusb0.sys, WinUSB with the CoInstallers, or WinUSB without the CoInstallers) for each Platform version (Win2K/XP/Vista/Win7/64).

cheers,

Graeme Gill.
libusb-1.0 Windows Backend on cygwin  See Also:

http://cygwin.com/cgi-bin2/package-grep.cgi?grep=libusb


I have not tried it yet. The version is called libusb1.0-1.0.5+git03e9371a.

from Xiaofan Chen <xiaofanc@gmail.com>
to René Hansen <renehh@gmail.com>
cc libusb-devel@lists.sourceforge.net
date Tue, Jun 1, 2010 at 3:53 AM
mailing list libusb-devel@lists.sourceforge.net Filter messages from this mailing list

On Tue, Jun 1, 2010 at 1:51 AM, René Hansen <renehh@gmail.com> wrote:

> I've been in contact with a company called All.com and on their behalf, I'm currently looking into the feasibility of porting a > library like libgpod cross platform, with initial focus on Windows. > Being able to to sync media devices like iPods and so on without > iTunes is key for their project.

Not so sure if this helps. Yesterday I happened to come across this library usbmuxd by Hector Martin. http://www.libusb.org/wiki/Libusb1.0 http://marcansoft.com/blog/iphonelinux/usbmuxd/

As it is using libusb-1.0 and with bulk tansfer, probably the porting of the libusb layer to Windows is not that difficult (with WinUSB backend). It seems to use asynchronous transfer which is not heavily tested for the Windows backend as far as I know. But hopefully it should work.

Is libgpod using usbmuxd?

I do not have any iPod/iPhone though since I do not buy things from Apple (and Sony except Sony Radios) in general.

libusb FAQ

Libusb on windows 7 64bits

from Xiaofan Chen <xiaofanc@gmail.com>
reply-to libusb-win32-devel@lists.sourceforge.net
to libusb-win32-devel@lists.sourceforge.net
date Fri, Jan 22, 2010 at 3:07 AM
subject Re: [Libusb-win32-devel] Building Libusb for 64-bit Windows Vista and Seven
mailing list <libusb-win32-devel@lists.sourceforge.net> Filter messages from this mailing list
mailed-by lists.sourceforge.net
unsubscribe Unsubscribe from this mailing-list

> I have read on a forum that the digital signing process requires the major > version number of the DLL to be greater or equal to 1. The current major > version number of Libusb is 0.

Yes that is said to be the major thing to be changed to get WHQL for libusb-win32 based driver.

> Could you please tell me how to build these 64-bit files? You need DDK/WDK.

One company has already got their libusb-win32 based device driver WHQL certified. Please refer to the following archived thread. http://old.nabble.com/Win-Vista-32-64-Build.-td17313102.html

The file change_mfh.txt list the changes.

from Bachelier, Georges <georges.bachelier@atmel.com>
reply-to libusb-win32-devel@lists.sourceforge.net
to libusb-win32-devel@lists.sourceforge.net
date Mon, Jan 25, 2010 at 4:03 PM
subject Re: [Libusb-win32-devel] Building Libusb for 64-bit WindowsVista and Seven
mailing list <libusb-win32-devel.lists.sourceforge.net> Filter messages from this mailing list
mailed-by lists.sourceforge.net
unsubscribe Unsubscribe from this mailing-list

Yes, we are using our own certificate to sign the driver package. The INF file seems to be OK according to the Pruftechnik one and examples I have found.
I use inf2cat to build the catalog files, then signtool to sign them.

```
signtool sign /f E:\labo\tools\atmel.pfx /p **** atmel_usb_dfu.cat
signtool sign /f E:\labo\tools\atmel.pfx /p **** atmel_usb_dfu_x64.cat
```

I am currently reading this paper: http://www.microsoft.com/whdc/winlogo/drvsign/kmcs_walkthrough.mspx
It contains detailed information about the signing process. I will let you know about the results.

from Bachelier, Georges <georges.bachelier@atmel.com>
reply-to libusb-win32-devel@lists.sourceforge.net
to libusb-win32-devel@lists.sourceforge.net
date Tue, Jan 26, 2010 at 5:39 PM
subject Re: [Libusb-win32-devel] Building Libusb for 64-bit WindowsVista and Seven
mailing list <libusb-win32-devel.lists.sourceforge.net> Filter messages from this mailing list
mailed-by lists.sourceforge.net
unsubscribe Unsubscribe from this mailing-list

Hi Xiaofan!

After some tweaking in our driver building and signing processes, we finally got a successful Vista and Seven 64-bit LibUsb driver installation! We also have had to modify our INF file accordingly to the Pruftechnik’s one.

Thanks a lot for your support on this topic, and thanks to all contributors who sent me solutions.

Kind regards,

libwdi

**libwdi**  
See Also:

- http://libusb.org/wiki/libwdi
- http://git.libusb.org/?p=libusb-pbatard.git;a=shortlog;h=refs/heads/wdi

The aim of this project is to create a library that facilitates the foolproof installation of any USB driver required to use a libusb application on Windows.

Features

- all required driver files are provided by the library - no need for additional downloads
- automated inf generation
• compatible with 32 and 64 bit Windows platforms, starting with Windows XP, with the possibility to either produce 2 separate libraries, for 32 or 64 bit, or a single library that covers both 32 and 64 bit at the same time.

• can either be integrated in your libusb application, so that a single executable application can be redistributed without the need for an additional installer, or to create a separate installer application (the library itself does have any libusb dependency)

Status  As of 2010.03.29, libwdi is in functional beta.

libwdi news

libusb 1.0 Windows backend WDI branch testing (Tue, Mar 30, 2010 at 4:18 PM)

I just build the WDI branch with WDK and it seems to work. Tested with the libusbdotnet Benchmark firmware (just change the PID to 0054 instead of 0053 and can called the Windows wizard).

C:\cygwin\home\mcuee\mcu\libusb\win32\git\wdi\libusb-pbatard\WDK\Win32>setdrv.exe
libwdi:debug [wdi_create_list] got hardware ID: USB\VID_04D8&PID_0054&REV_0000
libwdi:debug [wdi_create_list] Driverless USB device (11): USB\VID_04D8&PID_0054\6&1DA2B8C1&0&3
libwdi:debug [wdi_create_list] Device description: Microchip WinUSB Example Device
Found driverless USB device: "Microchip WinUSB Example Device" (VID_04D8:PID_0054)
Do you want to install a driver for this device (y/n)?
y
libwdi:debug [extract_binaries] successfully extracted files to C:\test
libwdi:debug [wdi_install_driver] all clean
libwdi:debug [process_message] [installer process] got parameter
libusb-device.inf
libwdi:debug [process_message] got request for device_id
libwdi:debug [process_message] [installer process] got device_id: USB\VID_04D8&PID_0054\6&1DA2B8C1&0&3
libwdi:debug [process_message] got request for hardware_id
libwdi:debug [process_message] [installer process] got hardware_id: USB\VID_04D8 &PID_0054&REV_0000
libwdi:debug [process_message] switching timeout to infinite
libwdi:debug [process_message] [installer process] Installing driver - please wait...
libwdi:debug [process_message] switching timeout back to finite
libwdi:debug [process_message] [installer process] driver update completed

2.9. Drivers
libwdi:debug [process_message] [installer process] re-enumerating driver node USB\VID_04D8&PID_0054\6&1DA2B8C1&0&3...
libwdi:debug [process_message] [installer process] re-enumeration succeeded...

--
Xiaofan http://mcuee.blogspot.com

Re: [Libusb-devel] libusb 1.0 Windows backend WDI branch testing Tue, Mar 30, 2010 at 4:55 PM

On 2010.03.30 15:18, Xiaofan Chen wrote:
> I just build the WDI branch with WDK and it seems to work.

Yeah, I was meaning to make a formal announcement about that once I have a GUI version of the setdvr application, but if you’re in a hurry, and need a one size fits all WinUSB installer, you can pick up the 32+64 bit precompiled setdrv.exe, from the "Driver Installation" section on the Windows Backend page (http://libusb.org/wiki/windows_backend).

setdrv, which relies on the Windows Driver Installer library - libwdi, will detect and install a WinUSB driver for any plugged USB device that doesn’t have a driver, for any Windows platform starting with XP (including 64 bit ones). Ultimately, it’ll also install libusb0.sys or any other USB driver we might require for the backend.

One of the main goals of libwdi is to be able to produce libusb executables that also contain *all* the required driver files, so that a single executable can be redistributed, without worrying about users having to manually download and install their drivers.

libwdi can also be used standalone, for the writing of installer type applications. I’m also examining the possibility of optionally providing the ability to use downloadable driver content, instead of embedded, which would reduce the general footprint of the library where needed (a 32+64 bit compatible version is about 4.8 MB because of WinUSB DLLs).

Note that even though the current git tree is a branch of libusb, libwdi doesn’t share or use any files from libusb and can be compiled as a standalone.

More info: http://libusb.org/wiki/libwdi

Regards,

/Pete

PS: I’m still planning to make a formal announcement once I have a more presentable, and foolproof, GUI installer application.
Development tools, Release 2012.06.18

pyusb, pylimbusb

pylibusb tool  See Also:
ref:usb_gnu_linux_tools

pylibusb  See Also:
http://pypi.python.org/pypi/pylibusb/0.2
libusb wrapper (ctypes based).

pylibusb.py

pylibusb version

pyUSB  See Also:
• http://pyusb.wiki.sourceforge.net/
• http://pyusb.berlios.de

Contents

• pyUSB
  – pyUSB tutorial
  – pyUSB news

pyUSB tutorial

Contents

• Programming with PyUSB 1.0
  – Let me introduce myself
  – Enough talk, let’s code!
    • Who’s who
    • Let’s get it started
    • What’s wrong?
    • Where are you?
    • Describe yourself
    • How am I supposed to work?
    • Talk to me, honey
  – Additional Topics
    • Behind every great abstraction, there’s a great implementation
    • Don’t be selfish
    • Oldschool rules
    • Help me, please
    • What do you think about it?
Let me introduce myself

PyUSB 1.0 is a Python library allowing easy USB access. It has the following features:

100% written in Python: Unlike the 0.x version, which is written in C, 1.0 version is written in Python. This allows Python programmers with no background in C to understand better how PyUSB works.

Platform neutrality: 1.0 version implements a frontend-backend scheme. This isolates the API from system specific implementation details. The glue between the two layers is the IBackend interface. PyUSB comes with builtin backends for libusb 0.1, libusb 1.0 and OpenUSB. You can write your own backend if you desire to.

Portability: PyUSB should run on any platform with Python >= 2.3, ctypes and at least one of the supported builtin backends.

Easeiness: Communicating with an USB device has never been so easy! USB is a complex protocol, but PyUSB has good defaults for most common configurations.

Support for isochronous transfers: PyUSB supports isochronous transfers if the underline backend supports it.

Although PyUSB makes USB programming less painful, it is assumed in this tutorial that you have a minimal USB protocol background. If you don’t know anything about USB, I recommend you the excellent Jan Axelson’s book USB Complete.

Enough talk, let’s code!

Who’s who

First of all, let’s give an overview on the PyUSB modules. PyUSB modules are under the usb package. This package has the following modules:

<table>
<thead>
<tr>
<th>Content</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>core</td>
<td>The main USB module.</td>
</tr>
<tr>
<td>util</td>
<td>Utility functions</td>
</tr>
<tr>
<td>legacy</td>
<td>The 0.x compatibility layer</td>
</tr>
<tr>
<td>backend</td>
<td>A subpackage containing the builtin backends</td>
</tr>
</tbody>
</table>

For example, to import the core module, you do as so:

```python
>>> import usb.core
>>> dev = usb.core.find()
```

Let’s get it started

Following is a simplistic program that sends the ‘test’ string to the first OUT endpoint found:

```python
import usb.core
import usb.util

# find our device
dev = usb.core.find(idVendor=0xfffe, idProduct=0x0001)

# was it found?
if dev is None:
    raise ValueError('Device not found')

# set the active configuration. With no arguments, the first
# configuration will be the active one
dev.set_configuration()

# get an endpoint instance
ep = usb.util.find_descriptor(
    dev.get_interface_altsetting(),  # first interface
```
# match the first OUT endpoint
custom_match = lambda e:
  usb.util.endpoint_direction(e.bEndpointAddress) == usb.util.ENDPOINT_OUT

assert ep is not None

# write the data
ep.write('test')

The first two lines import PyUSB package modules. `usb.core` is the main module, and `usb.util` contains utility functions. The next command searches our device and returns an instance object if it is found. If not, `None` is returned. After, we set the configuration to use. Note that no argument indicating what configuration we want was supplied. As you will see, many PyUSB functions have defaults for most common devices. In this case, the configuration set is the first one found.

Then, we look for the endpoint we are interested. We search it inside the first interface we have. After finding the endpoint, we send data to it.

If we know the endpoint address in advance, we could just call the `write` function from the device object:

dev.write(1, 'test', 0)

Here we write the string ‘test’ at endpoint address 1 of the interface number 0. All these functions will be detailed in the next sections.

**What’s wrong?** Every function in PyUSB raises an exception in case of an error. Besides the Python standard exceptions, PyUSB defines the `usb.core.USBError` for USB related errors.

You can also use the PyUSB log functionality. It uses the `logging` module. To enable it, define the environment variable `PYUSB_DEBUG_LEVEL` with one of the following level names: `critical`, `error`, `warning`, `info` or `debug`.

By default the messages are sent to `sys.stderr`. If you want to, you can redirect log messages to a file by defining the `PYUSB_LOG_FILENAME` environment variable. If its value is a valid file path, messages will be written to it, otherwise it will be sent to `sys.stderr`.

**Where are you?** The `find()` function in the `core` module is used to find and enumerate devices connected to the system. For example, let’s say that our device has a vendor id equals to 0xfffe and product id equals to 0x0001. If we would like to find it, we would do so:

```python
import usb.core
dev = usb.core.find(idVendor=0xfffe, idProduct=0x0001)
if dev is None:
    raise ValueError('Our device is not connected')
```

Just it, the function will return an `usb.core.Device` object representing our device. If the device is not found, it returns `None`. Actually, you can use any field of the Device Descriptor you desire. For example, what if we would like to discover if there is an USB printer connected to the system? This is far easy:

```python
# actually this is not the whole history, keep reading
if usb.core.find(bDeviceClass=7) is None:
    raise ValueError('No printer found')
```
The 7 is the code for the printer class according to the USB standard. Hey, wait, what if I want to enumerate all printers present? No problem:

```python
# this is not the whole history yet...
printers = usb.core.find(find_all=True, bDeviceClass=7)

def _call__(self, device):
    # first, let's check the device
    if device.bDeviceClass == self._class:
        return True
    # ok, transverse all devices to find an
    # interface that matches our class
    for intf in device:
        # find_descriptor: what's it?
        if intf is not None:
            return True
    return False
```

What happened? Well, it is time for a little explanation... `find` has a parameter called `find_all` that defaults to False. When it is false, `find` will return the first device found that matches the specified criteria (more on it soon). If you give it a true value, `find` instead will return a list with all devices matching the criteria. That’s it! Simple, doesn’t it?

Finished? No! I have not told you the whole history: many devices actually put their class information in the Interface Descriptor instead of the Device Descriptor. So, to really find all printers connected to the system, we would need to transverse all configurations, and then all interfaces and check if one of the interfaces has its bInterfaceClass field equals to 7. “Man, too much work for a programmer”, you might say. Yes, I am one two, that’s why I have implemented some stuff to make our lives a bit easier. First, let’s give a look on the final code to find all printers connected:

```python
import sys
sys.stdout.write('There are ' + len(printers) + ' in the system

import usb.core
import usb.util
import sys

class find_class(object):
    def __init__(self, class_):
        self._class = class_
    def __call__(self, device):
        # first, let’s check the device
        if device.bDeviceClass == self._class:
            return True
        # ok, transverse all devices to find an
        # interface that matches our class
        for intf in device:
            # find_descriptor: what’s it?
            if intf is not None:
                return True
        return False

printers = usb.core.find(find_all=1, custom_match=find_all(7))
```

The `custom_match` parameter accepts any callable object that receives the device object. It must return true for a matching device, and false for a non-matching device. You can also combine `custom_match` with device fields if you want:

```python
# find all printers that belongs to our vendor:
printers = usb.core.find(find_all=1, custom_match=find_class(7), idVendor=0xfffe)
```

---

[^1]: When I say True or False (capitalized), I mean the respectively values of the Python language. And when I say true and false, I mean any expression in Python which evals to true or false.
Here we are only interested in the printers of the 0xfffe vendor.

**Describe yourself**  Ok, we’ve found our device, but before talking to it, we would like to know more about it, you know, configurations, interfaces, endpoints, transfer types...

If you have a device, you can access any device descriptor fields as object properties:

```python
>>> dev.bLength
>>> dev.bNumConfigurations
>>> dev.bDeviceClass
>>> # ...
```

To access the configurations available in the device, you can iterate over the device:

```python
for cfg in dev:
    sys.stdout.write(str(cfg.bConfigurationValue) + '
')
```

In the same way, you can iterate over a configuration to access the interfaces, and iterate over the interfaces to access their endpoints. Each kind of object has as attributes the fields of the respective descriptor. Let’s see an example:

```python
for cfg in dev:
    sys.stdout.write(str(cfg.bConfigurationValue) + '
')
    for intf in cfg:
        sys.stdout.write('   ' + str(intf.bInterfaceNumber) + ', ' + str(intf.bAlternateSetting) + '    '
        + str(intf.bEndpointAddress) + '    
')
```

You can also use the subscript operator to access the descriptors randomly, like that:

```python
>>> # access the second configuration
>>> cfg = dev[1]
>>> # access the first interface
>>> intf = cfg[(0,0)]
>>> # third endpoint
>>> ep = intf[2]
```

As you can see, the index is zero based. But wait! There is something weird in the way I access an interface... Yes, you are right, the subscript operator in the Configuration accepts a sequence of two items, with the first one being the index of the Interface and the second one, the alternate setting. So, to access the first interface, but its second alternate setting, we write `cfg[(0,1)]`.

Now it’s time to we learn a powerful way to find descriptors, the `find_descriptor` utility function. We have already seen it in the printer finding example. `find_descriptor` works in almost the same way as `find`, with two exceptions:

- `find_descriptor` receives as its first parameter the parent descriptor that you will search on.
- There is no backend parameter.

---

2 See backend specific documentation.
For example, if we have a configuration descriptor \texttt{cfg} and want to find all alternate settings of the interface 1, we do so:

```python
import usb.util
alt = usb.util.find_descriptor(cfg, find_all=True, bInterfaceNumber=1)
```

Note that \texttt{find_descriptor} is in the \texttt{usb.util} module. It also accepts the early described \texttt{custom_match} parameter.

**How am I supposed to work?** USB devices after connected must be configured through a few standard requests.

When I got started to study USB spec, I found myself confused with descriptors, configurations, interfaces, alternate settings, transfer types and all this stuff... And worst, you cannot simply ignore them, a device does not work without setting a configuration, even if it has just one! PyUSB tries to make your life as easy as possible. For example, after getting your device object, one of the first things you need to do before communicating with it is issuing a \texttt{set_configuration} request. The parameter for this request is the \texttt{bConfigurationValue} of the configuration you are interested in. Most devices has no more than one configuration, and tracking the configuration value to use is annoying (although most code I have seem simply hardcode it). Therefore, in PyUSB, you can just issue a \texttt{set_configuration} call with no parameters. In this case, it will set the first configuration found (if your device has just one, you don't need to worry about the configuration value at all). For example, let's imagine you have a device with one configuration descriptor with its \texttt{bConfigurationValue} field equals to 5

```
>>> dev.set_configuration(5)
>>> dev.set_configuration() # we assume the configuration 5 is the first one
>>> cfg = util.find_descriptor(dev, bConfiguration=5)
>>> cfg.set()
>>> dev.set_configuration(cfg)
```

Wow! You can use a \texttt{Configuration} object as a parameter to \texttt{set_configuration}! Yes, and also it has a \texttt{set} method to configure itself as the current configuration.

The other setting you might or might not have to configure is the interface alternate setting. Each device can have only one activated configuration at a time, and each configuration may have more than one interface, and you can use all interfaces at the same time. You better understand this concept if you think of an interface as a logical device. For example, let's imagine a multifunction printer, which is at the same time a printer and a scanner. To keep things simple (or at least as simple as we can), let's consider it has just one configuration. As we have a printer and a scanner, the configuration has two interfaces, one for the printer and one for the scanner. A device with more than one interface is called a composite device. When you connect your multifunction printer to your computer, the Operating System would load two different drivers: one for each “logical” peripheral you have.

And about the alternate setting? Good you have asked. An interface has one or more alternate settings. An interface with just one alternate setting is considered to not having an alternate setting. Alternate settings are for interfaces which configurations are for devices, i.e, for each interface, you can have only one alternate setting active. For example, USB spec says that a device cannot have a isochronous endpoint in its primary alternate setting, so a streaming device must have at least two alternate settings, with the second one having the isochronous endpoint(s). But as opposed to configurations, interfaces with just one alternate setting don't need to be set. You select an interface alternate setting through the \texttt{set_interface_altsetting} function:

```
>>> dev.set_interface_altsetting(interface = 0, alternate_setting = 0)
```

---

3 USB spec does not impose any sequential value to the configuration value. The same is true for interface and alternate setting numbers.

4 Actually things are a little more complex, but this simple explanation is enough for us.

5 I know it sounds weird.

6 This is because if there is no bandwidth for isochronous transfer at the device configuration time, the device can be successfully enumerated.

7 This does not happen for configurations because a device is allowed to be in an unconfigured state.
Warning: The USB spec says that a device is allowed to return an error in case it receives a SET_INTERFACE request for an interface that has no additional alternate settings. So, if you are not sure if the interface has more than one alternate setting or it accepts a SET_INTERFACE request, the safety way is to call `set_interface_altsetting` inside an `try-except` block, like so:

```python
try:
    dev.set_interface_altsetting(...)
except USBError:
    pass
```

You can also use an `Interface` object as a parameter to the function, the `interface` and `alternate_setting` parameters are automatically inferred from `bInterfaceNumber` and `bAlternateSetting` fields. Example:

```python
>>> intf = find_descriptor(...)  
>>> dev.set_interface_altsetting(intf)  
>>> intf.set_altsetting()  # wow! Interface also has a method for it
```

Warning: The `Interface` object must belong to the active configuration descriptor.

Talk to me, honey  Now it’s time to learn how to communicate with USB devices. USB has four flavors of transfers: bulk, interrupt, isochronous and control. I don’t intend to explain the purpose of each transfer and the differences among them. Therefore, I assume you know at least the basics of the USB transfers.

Control transfer is the unique transfer that has structured data described in the spec, the others just send and receive raw data from USB point of view. Because of it, you have a different function to deal with control transfers, all the other transfers are managed by the same functions.

You issue a control transfer through the `ctrl_transfer` method. It is used both for OUT and IN transfers. The transfer direction is determined from the `bmRequestType` parameter.

The `ctrl_transfer` parameters are almost equal to the control request structure. Following is an example of how to do a control transfer 8:

```python
>>> msg = 'test'  
>>> assert dev.ctrl_transfer(0x40, CTRL_LOOPBACK_WRITE, 0, 0, msg) == len(msg)  
>>> ret = dev.ctrl_transfer(0x40, CTRL_LOOPBACK_READ, 0, 0, len(msg))  
>>> sret = ''.join([chr(x) for x in ret])  
>>> assert sret == msg
```

In this example, it is assumed that our device implements two custom control requests that act as a loopback pipe. What you write with the `CTRL_LOOPBACK_WRITE` message, you can read with the `CTRL_LOOPBACK_READ` message.

The first four parameters are the `bmRequestType`, `bmRequest`, `wValue` and `wIndex` fields of the standard control transfer structure. The fifth parameter is either the data payload for an OUT transfer or the number of bytes to read in an IN transfer. The data payload can be any sequence type that can be used as a parameter for the `array __init__` method. If there is no data payload, the parameter should be `None` (or 0 in case of an IN transfer). There is one last optional parameter specifying the timeout of the operation. If you don’t supply it, a default timeout will be used (more on that later). In an OUT transfer, the return value is the number of bytes really sent to the device. In an IN transfer, the return value is an `array` object with the data read.

8 In PyUSB, control transfers are only issued in the endpoint 0. It’s very very rare a device having an alternate control endpoint (I’ve never seem such device).
For the other transfers, you use the methods `write` and `read`, respectively, to write and read data. You don’t need to worry about the transfer type; it is automatically determined from the endpoint address. Here is our loopback example assuming the we have a loopback pipe in the endpoint 1:

```python
>>> msg = 'test'
>>> assert len(dev.write(1, msg, 0, 100)) == len(msg)
>>> ret = dev.read(0x81, len(msg), 0, 100)
>>> sret = ''.join([chr(x) for x in ret])
>>> assert sret == msg
```

The first, third and fourth parameters are equal for both methods, they are the endpoint address, interface number and timeout, respectively. The second parameter is the data payload (write) or the number of bytes to read (read). The return of the `read` function is an instance of the `array` object or the number of bytes written for the `write` method.

As in `ctrl_transfer`, the `timeout` parameter is optional. When the `timeout` is omitted, it is used the `Device.default_timeout` property as the operation timeout.

**Additional Topics**

**Behind every great abstraction, there’s a great implementation**  On early days, there was only `libusb`. Then came `libusb 1.0`, and now we had `libusb 0.1` and `1.0`. After, they created `OpenUSB`, and now we live at the `Tower of Babel` of the USB libraries. How does `PyUSB` deal with it? Well, `PyUSB` is a democratic library, you may choose whatever library you want. Actually, you can write your own USB library from scratch and tell `PyUSB` to use it.

The `find` function has one more parameter that I haven’t told you. It is the `backend` parameter. If you don’t supply it, it will be used one of the builtin backends. A backend is a object derived from `usb.backend.IBackend`, responsible to implement the operating system specific USB stuff. As you might guess, the builts are `libusb 0.1`, `libusb 1.0` and `OpenUSB` backends.

You can create your own backend and use it. Just inherit from `IBackend` and implement the methods necessary. You might want to give a look at `backend` package documentation to learn how to do that.

**Don’t be selfish**  Python has what we say automatic memory management. This means that the virtual machine will take care about when to release objects from the memory. Under the hoods, `PyUSB` manages all low level resource management it needs to work (interface claiming, device handles, etc.) internally and most of users don’t need to worry about that. But, because of the nonderterministic nature of automatic object destruction of Python, users cannot predict when the resources allocated will be released. Some applications need to allocate and free the resources deterministically. For these kind of applications, the `usb.util` module has a set of functions to deal with resource management.

If you want to claim and release interfaces manually, you may use the `claim_interface` and `release_interface` functions. `claim_interface` will claim the specified interface if the device has not done it yet. If the device already claimed the interface, it does nothing. In a similar way, `release_interface` will release the specified interface if it is claimed. If the interface is not claimed, it does nothing. You can use manual interface claim to solve the configuration selection problem described in the `libusb` documentation.

If you want to free all resources allocated by the device object (including interfaces claimed), you can use the `dispose_resources` function. It releases all resources allocated and put the device object (but not the device hardware itself) in the state it was at the time when the `find` function returned.

---

9 It’s just a joke, don’t take it serious. Many choices is better than no choice.
**Oldschool rules** If you wrote an application using the old PyUSB API (0.whatever), you may be asking yourself if you need to update your code to use the new API. Well, you should, but you don’t need to. PyUSB 1.0 comes with the `usb.legacy` compatibility module. It implements the older API above the new API. “So, do I have just to replace my `import usb` statement with `import usb.legacy as usb` to get my application working?”, you ask. The answer is yes, it will, but you don’t have to. If you run your application untouched it will just work, because the `import usb` statement will import all public symbols from `usb.legacy`. If you face a problem, probably you found a bug.

**Help me, please** If you need help, **do not email me**, the mailing list is there for this. Subscribe instructions can be found at the PyUSB website.

**What do you think about it?** At alpha stage, users of PyUSB are invited to give their opinion about the PyUSB API. If you think a feature is hard to use and you have a better idea, open a new thread in the mailing list so we can discuss about that.

**pyusb install**

**ldconfig and LD_LIBRARY_PATH**

```bash
vim ld.so.conf
add /opt/libusb/current/lib
run ldconfig -v | grep libusb
ldconfig -v | grep libusb
```

```
[root@houx etc]# ldconfig -v | grep libusb
/opt/libusb/current/lib:
  libusb-1.0.so.0 -> libusb-1.0.so.0.0.0
  libusb-0.1.so.4 -> libusb.so
  libusbpp-0.1.so.4 -> libusp
```

**set the new library** See Also:

http://docs.python.org/library/ctypes.html

```python
from ctypes import *
import ctypes.util
from ctypes.util import find_library
libname_usb10 = find_library('libusb-1.0')

def _load_library():
    candidates = ('usb-1.0', 'libusb-1.0', 'usb')
    for candidate in candidates:
        libname = find_library(candidate)
        if libname is not None: break
```

**pyusb examples**
Example1

On Tue, Aug 24, 2010 at 3:43 PM, Max Teo <maxdaw@gmail.com> wrote:
> Surprisingly as I had mentioned previously in the comments above, using a
> bus analyzer, the CSW status can be retrieved if I
> were to use another standalone Microsoft program to get the status back.
> What do you think could have caused this?
>
> Something is preventing within this version that’s preventing to read back
> the CSW. I still think using lib-usb1.0 is a great start
> and would like to contribute back if I were to find anything.
>
> Otherwise, what would you recommend?

I think libusb-1.0 should work if the other program works. A few suggestions:

- Post your pyusb program and Wander or others may point out whether your python pyusb code is correct or not.
- Enable debugging. libusb-1.0 Windows backend has the togglable debugging information (by calling libusb_set_debug(4)). pyusb should have similar things.
- Or you can directly use libusb-1.0 and C for testing purpose.

Since you have the bus analyzer, you can see the differences on the wire.

Response I used both option 1 and 2 then :)

Its not even a program yet as its a trial to see if libusb can achieve what I desired. Some snippets are cut out (not interesting and too much info)

```python
>>> import usb.core
>>> import usb.util
>>> import usb.backend.libusb10 as libusb10

backend = libusb10.get_backend()
libusb:debug [libusb_init] created default context
libusb:debug [libusb_init]
libusb:debug [init_polling] Will use CancelIo for I/O cancellation
libusb:debug [windows_clock_gettime_threaded] hires timer available (Frequency: 2000040000 Hz)
libusb:debug [usb_add_pollfd] add fd 3 events 1

backend
<usb.backend.libusb10._LibUSB object at 0x01B2C4B0>

>>> dev=usb.core.find(idVendor=0x152d, idProduct=0x2339, backend=backend)
...
libusb:debug [set_device_paths] path (3:2): \.\USB#VID_152D&PID_2339#02000BFFFFFF#{A5DCBF10-6530-11
D2-901F-00C04FB951ED}
libusb:debug [set_device_paths] driver(s): WINUSB
libusb:debug [set_device_paths] matched driver name against WinUSB API
...

>>> dev.set_configuration()
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_open] open 3.2
libusb:debug [libusb_set_configuration] configuration 1
libusb:debug [libusb_claim_interface] interface 0
libusb:debug [winusb_claim_interface] claimed interface 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [windows_assign_endpoints] (re)assigned endpoint 81 to interface 0
libusb:debug [windows_assign_endpoints] (re)assigned endpoint 02 to interface 0
libusb:debug [auto_claim] auto-claimed interface 0 for control request
```
libusb:debug [winusb_submit_control_transfer] will use interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [libusb_get_next_timeout] next timeout in 0.99120s
libusb:debug [handle_events] poll() 2 fds with timeout in 992ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with events = 0000
libusb:debug [windows_handle_events] checking fd 4 with events = 0001
libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 0
libusb:debug [libusb_release_interface] interface 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [windows_assign_endpoints] (re)assigned endpoint 81 to interface 0
libusb:debug [windows_assign_endpoints] (re)assigned endpoint 02 to interface 0
libusb:debug [auto_release] auto-released interface 0
libusb:debug [ctrl_transfer_cb] actual_length=0

>>> intf=dev.get_interface_altsetting()
libusb:debug [libusb_get_config_descriptor] index 0
>>> out = usb.util.find_descriptor(...
... intf, custom_match = 
... lambda e: 
... usb.util.endpoint_direction(e.bEndpointAddress) == 
... usb.util.ENDPOINT_OUT)
libusb:debug [libusb_get_config_descriptor] index 0
>>> ind = usb.util.find_descriptor(...
... intf, custom_match = 
... lambda e: 
... usb.util.endpoint_direction(e.bEndpointAddress) == 
... usb.util.ENDPOINT_IN)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0

# The commands below are trusted out commands; technically, if another session is sent out again
# it will cause an abort reflected in the CSW. As mentioned, the first command was sent out successfully

>>> security_trusted_out = hex_digit_pairs_to_byte_array("55:53:42:11:34:66:78:00:02:00:00:80:00:0F:85:16:03:00:01:00:01:00:00:00:FE:00:07:00:5F:00", ":")
>>> out.write(security_trusted_out)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_claim_interface] interface 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [windows_assign_endpoints] (re)assigned endpoint 81 to interface 0
libusb:debug [windows_assign_endpoints] (re)assigned endpoint 02 to interface 0
libusb:debug [winusb_submit_bulk_transfer] matched endpoint 02 with interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [winusb_submit_bulk_transfer] writing 31 bytes
libusb:debug [usbi_add_pollfd] add fd 4 events 4
libusb:debug [libusb_get_next_timeout] next timeout in 0.990349s
libusb:debug [handle_events] poll() 2 fds with timeout in 991ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with events = 0000
libusb:debug [windows_handle_events] checking fd 4 with events = 0004
libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 0
libusb:debug [bulk_transfer_cb] actual_length=31
Development tools, Release 2012.06.18

>>> open_session = pad_blob_to_size(hex_digit_pairs_to_byte_array("00 00 00 00 07 FE 00 00 00 00 00
00 00 00 00 00 00 00 00 50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 38 0
0 00 00 00 00 00 00 00 00 00 00 29 F8 A8 00 00 00 00 00 00 00 FF A8 00 00 00 00 00 00 FF 02 F0 83 01
2E 13 A8 00 00 02 05 00 00 00 01 01 F1 F9 F0 00 00 00 F1", " ").as_blob(), 512)
>>> out.write(open_session)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [winusb_submit_bulk_transfer] matched endpoint 02 with interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [winusb_submit_bulk_transfer] writing 512 bytes
libusb:debug [usbi_add_pollfd] add fd 4 events 4
libusb:debug [libusb_get_next_timeout] next timeout in 0.995021s
libusb:debug [handle_events] poll() 2 fds with timeout in 996ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with revents = 0000
libusb:debug [windows_handle_events] checking fd 4 with revents = 0004
libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 0
libusb:debug [bulk_transfer_cb] actual_length=512
>>> ind.read(13, timeout=5000)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [winusb_submit_bulk_transfer] matched endpoint 81 with interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [winusb_submit_bulk_transfer] reading 13 bytes
libusb:debug [usbi_add_pollfd] add fd 4 events 1
libusb:debug [libusb_get_next_timeout] next timeout in 4.990351s
libusb:debug [handle_events] poll() 2 fds with timeout in 4991ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with revents = 0000
libusb:debug [windows_handle_events] checking fd 4 with revents = 0001
libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 0
libusb:debug [bulk_transfer_cb] actual_length=13
array(’B’, [85, 83, 66, 83, 17, 52, 102, 120, 0, 0, 0, 0, 0])
>>>

<--- this is important which is CSW (

# repeating with the same loop as above
>>> security_trusted_out = hex_digit_pairs_to_byte_array("55:53:42:43:11:34:66:78:00:02:00:00:80:00:
0F:85:16:03:00:01:00:01:00:00:00:FE:00:07:00:5F:00", ":")
>>> out.write(security_trusted_out)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [winusb_submit_bulk_transfer] matched endpoint 02 with interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [winusb_submit_bulk_transfer] writing 31 bytes
libusb:debug [usbi_add_pollfd] add fd 4 events 4
libusb:debug [libusb_get_next_timeout] next timeout in 0.999222s
libusb:debug [handle_events] poll() 2 fds with timeout in 1000ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with revents = 0000
libusb:debug [windows_handle_events] checking fd 4 with revents = 0004

324

Chapter 2. Development


Development tools, Release 2012.06.18

libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 0
libusb:debug [bulk_transfer_cb] actual_length=31
>>> open_session = pad_blob_to_size(hex_digit_pairs_to_byte_array("00 00 00 00 07 FE 00 00 00 00 00
00 00 00 00 00 00 00 00 50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 38 0
0 00 00 00 00 00 00 00 00 00 00 29 F8 A8 00 00 00 00 00 00 00 FF A8 00 00 00 00 00 00 FF 02 F0 83 01
2E 13 A8 00 00 02 05 00 00 00 01 01 F1 F9 F0 00 00 00 F1", " ").as_blob(), 512)
>>> out.write(open_session)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [winusb_submit_bulk_transfer] matched endpoint 02 with interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [winusb_submit_bulk_transfer] writing 512 bytes
libusb:debug [usbi_add_pollfd] add fd 4 events 4
libusb:debug [libusb_get_next_timeout] next timeout in 0.999240s
libusb:debug [handle_events] poll() 2 fds with timeout in 1000ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with revents = 0000
libusb:debug [windows_handle_events] checking fd 4 with revents = 0004
libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 0
libusb:debug [bulk_transfer_cb] actual_length=512
>>> ind.read(13, timeout=5000)
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [libusb_get_config_descriptor] index 0
libusb:debug [winusb_submit_bulk_transfer] matched endpoint 81 with interface 0
libusb:debug [usbi_create_fd] could not duplicate handle for CancelIo - using original one
libusb:debug [winusb_submit_bulk_transfer] reading 13 bytes
libusb:debug [usbi_add_pollfd] add fd 4 events 1
libusb:debug [libusb_get_next_timeout] next timeout in 4.999235s
libusb:debug [handle_events] poll() 2 fds with timeout in 5000ms
libusb:debug [handle_events] poll() returned 1
libusb:debug [windows_handle_events] checking fd 3 with revents = 0000
libusb:debug [windows_handle_events] checking fd 4 with revents = 0001
libusb:debug [usbi_remove_pollfd] remove fd 4
libusb:debug [windows_transfer_callback] handling I/O completion with errcode 31
libusb:debug [windows_transfer_callback] detected endpoint stall
libusb:debug [bulk_transfer_cb] actual_length=0
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "usb\core.py", line 273, in read
File "usb\core.py", line 624, in read
File "usb\_debug.py", line 53, in do_trace
File "C:\sandbox\openseapy\Kwai_Phase2_Internal_Review\toolbox\usb\backend\libusb10.py", line 483,
in bulk_read
timeout)
File "C:\sandbox\openseapy\Kwai_Phase2_Internal_Review\toolbox\usb\backend\libusb10.py", line 581,
in __read
timeout))
File "C:\sandbox\openseapy\Kwai_Phase2_Internal_Review\toolbox\usb\backend\libusb10.py", line 353,
in _check
raise USBError(_str_error[retval.value])
usb.core.USBError: Pipe error
>>>

2.9. Drivers

325


I tried polling for a few times by sending a libusb_clear_halt, (technically it's not required for the other standalone program) but got back the Pipe error.

pyUSB news

INF file (for WinUSB)

from Xiaofan Chen <xiaofanc@gmail.com>
reply-to pyusb-users@lists.sourceforge.net
to pyusb-users@lists.sourceforge.net
date Mon, May 31, 2010 at 1:13 AM
subject Re: [pyusb-users] XP

On Sun, May 30, 2010 at 11:31 PM, Karl Palsson <tweak@tweak.net.au> wrote: > If you’re screwing around with inf files, (in my opinion) you’re doing it wrong. > > I (finally) got windows XP and ubuntu working with (almost) the same code, and no > inf files. > > http://github.com/karlp/karlnet/tree/usbmaster/producers/pyhid/ has the simple > code, and the README there is.... >

I see you are using libusb-1.0, that is fine. Moreover you are using HID device, So you do not need to generate your own INF file. But you still need to generate the INF (for WinUSB) if you are using non-HID device.

The OP can of course try libusb-1.0 and WinUSB as well. For driver installation, zadig/libwdi is quite nice. The git version actually supports both:

- winusb.sys and
- libusb0.sys.

http://www.libusb.org/wiki/windows_backend

from Xiaofan Chen <xiaofanc@gmail.com>
date Wed, Jun 2, 2010 at 12:46 AM
subject Re: [pyusb-users] XP

On Wed, Jun 2, 2010 at 5:27 AM, Mark <mhh@absamail.co.za> wrote: > I created the INF file using inf-wizard.exe. My lsusb output is: >... > cannot read device status, Broken pipe (32)

This is not a real problem. If you run as root or sudo, lsusb may be fine.

What is the output of testlibusb-win.exe under Windows?

PyUSB 1.0 alpha 0 release notification

from Wander Lairson <wander.lairson@gmail.com>
reply-to pyusb-users@lists.sourceforge.net
to pyusb-users@lists.sourceforge.net
date Fri, Apr 16, 2010 at 5:14 PM
subject [pyusb-users] PyUSB 1.0 alpha 0 release notification

Dear all,

This is the first PyUSB 1.0 series public release. This is an alpha release, which means that most of the features described in the README file and on the website are not yet stable or even implemented.
Features not implemented

- Full support for legacy 0.4 legacy code (although partial support is provided).
- OpenUSB backend.
- libusb 1.0 windows backend stability (although it is reasonable usable).
- Support for several standard control requests (including GET_STRING).
- Python < 2.6 and Python 3 not yet fully tested.

Known issues

- ‘reset’ method fails under FreeUSB (libusb 1.0 backend).
- ‘reset’ method hangs under Windows (libusb 1.0 backend).
- Sometimes occurs read timeout on Windows (libusb 1.0 backend).
- Test cases fail to run under cygwin.

Best regards, Wander

Dotnet usb

USB dotnet  See Also:


usblib (SharpUSBLib) is a wrapper around the libusb project (WIN32), thus you must have it installed prior to using usblib.

Mike started this project because he wanted to program a power switch (GEMBIRD SIS-PM) with a USB port and didn’t find any .NET USB library.

One goal is to provide a platform independent (Linux/Win32 solution) USB access layer for .NET.

License  The library is dual-licensed: GPL or LGPL. The latter allows you to use the library in closed-source applications (when you link against the library, including the source code is still not allowed - it has to be in a separate assembly).

Download  Downloads are hosted on sf.net together with our #develop and #ziplib projects: Get the latest version (source, binary, samples, documentation)

python-libusb1

python-libusb1  See Also:

- https://github.com/vpelletier/python-libusb1
- https://github.com/vpelletier/python-libusb1.git

python-libusb1 is a Python wrapper for libusb-1.0.

python-libusb1 was created to get a python wrapper supporting asynchronous features of libusb1.
python-libusb1 is a Python wrapper for libusb-1.0

URL: http://github.com/vpelletier/python-libusb1

Requirements:

- Python 2.4+ (2.6+ recommended) http://www.python.org/ Not compatible with Python 3
- ctypes (included in Python 2.5+) http://python.net/crew/theller/ctypes/
- libusb-1.0 http://www.libusb.org/wiki/libusb-1.0

OS

python-libusb1 is expected to work on any OS supported by libusb. It is known to be used on:

- Linux
- Windows
- OSX

Contents

- libusb1.py Bare ctype wrapper, inspired from library C header file.
- usb1.py Python-ish (classes, exceptions, ...) wrapper around libusb1.py. See docstrings (pydoc recommended) for usage.
- setup.py To package as python egg.
- stdeb.cfg To package as Debian package. See https://github.com/astraw/stdeb.
- testUSB1.py Very limited regression test, only testing functions which do not require a USB device.

Install

python setup.py install

(you might need root access to do this)

History

python-libusb1 was created to get a python wrapper supporting asynchronous features of libusb1.

0.0.1 Initial release

0.1.1 Massive rework of usb1.py, making it more python-ish and fixing some memory leaks.

0.1.2 Deprecate “transfer” constructor parameter to allow instance reuse.

See also (other projects, different author):

- pyusb: another python wrapper for (among others) libusb1 http://sourceforge.net/projects/pyusb/

libusb haskell See Also:

http://hackage.haskell.org/package/usb
The usb package  This library allows you to communicate with USB devices from userspace. It is implemented as a high-level wrapper around bindings-libusb which is a low-level binding to the C library: libusb-1.*.

This documentation assumes knowledge of how to operate USB devices from a software standpoint (descriptors, configurations, interfaces, endpoints, control/bulk/interrupt/isochronous transfers, etc). Full information can be found in the USB 2.0 Specification.

For an example how to use this library see the ls-usb package at:

• http://hackage.haskell.org/package/ls-usb

Also see the usb-safe package which wraps this package and provides some strong safety guarantees for working with USB devices:

• http://hackage.haskell.org/package/usb-safe

Finally have a look at the usb-enumerator package which provides iteratee enumerators for enumerating bulk and interrupt endpoints:

http://hackage.haskell.org/package/usb-enumerator

Besides this API documentation the following sources might be interesting:

• The libusb 1.0 documentation at: http://libusb.sourceforge.net/api-1.0/
• The USB 2.0 specification at: http://www.usb.org/developers/docs/
• The bindings-libusb documentation at: http://hackage.haskell.org/package/bindings-libusb
• “USB in a NutShell” at: http://www.beyondlogic.org/usbnutshell/usb1.htm

libopenusb

libopenusb  Openusb is an enhanced version of the libusb 0.1.* library which allows userspace application access to USB devices. Documents can be found in the project’s website.

The official web site is: http://openusb.sourceforge.net/

See Also:

pyUSB

README

libhid

libhid  See Also:

• https://github.com/bfoz/libhid/
• http://www.wooji-juice.com/free/pyhid/readme.html

libhid linux

libhid news
libusb-ng

On Mon, Jun 13, 2011 at 11:58 PM, Ludovic Rousseau <ludovic.rousseau@gmail.com> wrote:
> So it will be hard to reuse the domain name libusb.org or the libusb
> sourceforge project.

That is true but not a big problem. github and Sourceforge both support git and Sourceforge can be a good place if
you want to start libusb-ng project.

> The only viable option I see now is to fork libusb in a new project
> (libusb-ng?) maintained by people working for the library users.

What do you have in mind for libusb-ng? More frequent release? Care more about the library users? Faster admission
of new features?

In terms of features, what do you have in mind?

libhid C++ hid

de Xiaofan Chen xiaofanc@gmail.com

Website: http://bfoz.github.com/libhid/

Right now Linux is not supported, Windows and Mac OS X are supported.

According to the author, Linux support is on his todo list. Perl/Ruby/Python bindings are also on the list after the
Linux support.

Also according to the author, the license is 2-clause BSD even though he does not really care what people do with it
so long as he does not get sued.

The library looks quite interesting and seems hotplug is supported as well. It is unfortunately named the same as the
existing libhid.

Missing first byte when reading HID Feature Reports with ID

OK, after much fruitless search, I found that I could get somewhat of an
automated HID parser report using libhid and the smali
extract_report_descriptor.py tool from libhid on Linux
(http://libhid.alioth.debian.org/). Yet another tool that takes a custom
HID report format, instead of good old universal binary dump – they really don’t want to make HID troubleshooting easy, do they...

The automated analysis seems to confirm that there’s only one feature report in the HID generic device, and thus no valid reason for report ID / ID prefixes to be used.

```
[0x0000] 0x06 0xa0 0xff (value: 0xffa0 / 65440)
  bSize = 2 byte(s)
  bType = 0x01 (Global)
  bTag  = 0x00 (Usage page)

[0x0003] 0x09 0x01 (value: 0x1 / 1)
  bSize = 1 byte(s)
  bType = 0x02 (Local)
  bTag  = 0x00 (Usage)
```

from Xiaofan Chen <xiaofanc@gmail.com>
to Pete Batard <pbatard@gmail.com>
cc libusb-devel@lists.sourceforge.net
date Thu, Apr 8, 2010 at 1:39 AM
subject Re: [Libusb-devel] Missing first byte when reading HID Feature Reports with ID

On Thu, Apr 8, 2010 at 1:23 AM, Pete Batard <pbatard@gmail.com> wrote:
>> It seems to me that it should be used when there are multiple
>> feature reports, just like Axel’s device.
>>
> Do we have multiple Feature Reports in the descriptor above? I only see
> one B1 entry?

No, there is only one Feature Report.

> I know realise that I’ve been spending more time trying to avoid to
> manually parse the data above than I would have spent doing so.
> I tried to use lsusb -vvv on Linux (which is supposed to provide HID
> report analysis after you detach the driver) but I get a
> ** UNRECOGNIZED: 09 21 11 01 00 01 22 2f 00
> before the endpoint descriptors.

You need to detach the kernel HID driver to get the report.

> I’ve also been trying to use the HID Descriptor Tool tool from the USB
> website, but it takes a custom .hid format as an input (what was wrong
> with using the actual binary report format from the USB devices?).
> Anybody knows of a tool that can produce a nicely formatted HID report
> text data from a binary report?

There is one new tool under Linux using libhid.
HIDRD:
https://sourceforge.net/projects/digimend/files/hidrd/0.1/hidrd-0.1.1.tar.gz/download
You can also visit the hidrd section on the DIGImend project page.
http://digimend.sourceforge.net/

2.9. Drivers 331
Brilliant! Just tried it, and the XML output is a lot more human readable than anything else I’ve seen so far.

I might add an option to xusb to dump the raw HID reports, so that we can then use something like: hidrd-convert raw.bin -o xml

Help with reading data from HID device

All in all, you will be annoyed the lack of a generic cross-platform HID library and Python wrapper.

Actually there was no native HID python wrapper under Windows for a long time until now. http://old.nabble.com/Python-HID-wrapper-for-Win32-td13329336.html Long long ago people asked for it (at least back in 2004). But it is not materialized until year 2010. It is still in development. http://code.google.com/p/pywinusb/

For Mac: pyhid http://www.wooji-juice.com/free/pyhid/readme.html

For Linux, Mac OS X and BSDs. libhid has the Python wrapper. But it is not working for Windows. I am not so sure how well it works under Mac OS X as it is still depending on libusb 0.1. http://libhid.alioth.debian.org/

With the new libusb 1.0 Windows HID backend, pyusb should work for HID device under Windows and Linux (and FreeBSD 8/9 where every USB interface seems to have a ugen driver associated with it). But someone needs to write the native libusb 1.0 HID backend for Mac OS X.

USB hid annexes

See Also:

- http://www.alanmacek.com/usb/
- https://github.com/bfoz/libhid/

USB and Human Interface Devices (HID) Human Interface Devices (HID) are a class of USB devices that give structure to the data that will be transferred between the device and the host computer. During the enumeration process, the device describes the information that it can receive and send. This allows a host computer to handle the data being received from the USB device without requiring a specially designed device driver.

HID class The HID class is supposed to include devices such as a mouse, joystick, keyboard, etc. Because the host computer knows what the data means a device driver is not necessary for HID devices, the operating system can supply a generic HID driver. For instance, if you plug in a USB Mouse, it will immediately work because the OS knows how to interpret information received from a mouse.
Information on the HID class can be found at http://www.usb.org/developers/hidpage. You can see examples of HID descriptors starting on page 89 of the HID 1.1 spec from http://usb.org.

A very useful tool for designing HID class devices is the HID Descriptor Tool also available from http://usb.org. This tool allows you to put together the HID description and run it through the HID parser.

Some other useful tools are available from Intel University Press. They include ‘USB Single Step’, ‘USBView’, ‘HIDView’ and the HID Descriptor Tool described above.

If you are using Visual C++, then the ‘hview’ sample program that comes with the Windows DDK is good for examining the HID descriptor and values. Unfortunately the program is more complicated than it needs to be and is not a good example of using the USB functions.

Links

TtyLinux


The very frugal ttylinux distro is available in version 9.1. The newest aspect of ttylinux is that the kernel now supports USB Human Interface Devices (HID), which primarily serve in connecting USB keyboards. It also has a new init script and config files for the cron daemon.

The resource-saving ttylinux is ready-made for i486 machines with 24 MBytes RAM in Live memory. A hard drive installation requires only 8 MBytes disk space and 16 MBytes RAM. The Linux distro also supports Intel 64-bit CPUs.

Ttylinux 9.1 is available for download from the project page as ISO images and source code archive.

usb ip

USB IP

usbip  http://usbip.sourceforge.net/

The USB/IP Project aims to develop a general USB device sharing system over IP network. To share USB devices between computers with their full functionality, USB/IP encapsulates “USB I/O messages” into TCP/IP payloads and transmits them between computers. Original USB device drivers and applications can be also used for remote USB devices without any modification of them. A computer can use remote USB devices as if they were directly attached; for example, we can ...

- USB storage devices: fdisk, mkfs, mount/umount, file operations, play a DVD movie and record a DVD-R media.
- USB keyboards and USB mice: use with linux console and X Window System.
- USB webcams and USB speakers: view webcam, capture image data and play some music.
- USB printers, USB scanners, USB serial converters and USB Ethernet interfaces: ok, use fine.

It is currently implemented as Linux device drivers and available under the open source license GPL. Its I/O performance is enough practical in local area network for all types of devices, including isochronous devices, without any modification of Linux-original USB device drivers.
Development tools, Release 2012.06.18

pyusbip This project provides a Python wrapper for the USB/IP shared library. The wrapper was generated using ctypesgen (http://code.google.com/p/ctypesgen/). The goal this project is to maintain these wrappers and provide GUI (pygtk) tools for server-side and client-side USB/IP userland operations.

This project is still in its larval stage :)

http://code.google.com/p/pyusbip/

Hi all,

Just an FYI if anyone is interested. I generated some Python wrappers for the usb/ip shared libraries. This is a new realm for me, so I don’t even know how useful the wrappers are in their current state. However, I created them several weeks ago and haven’t used them since, so I wanted to let everyone know there is at least something available. My end goal was to get some nice pygtk userland tools developed.

Here’s the link:

http://code.google.com/p/pyusbip/

If anyone decides to do any work in this realm, please let me know. I’d be happy to participate.

Thanks, Brian

USB dfu

USB DFU Device Firmware Upgrade

USB DFU Device Firmware Upgrade See Also:

• http://www.usb.org/developers/devclass_docs/usbdfu10.pdf
• http://dfu-programmer.sourceforge.net/
• ATMEL dfu-programmer

DFU (Device Firmware Upgrade) overview See Also:

• http://www.usb.org/developers/devclass_docs/usbdfu10.pdf
• Connexion à chaud et Plug and Play : processus d’enumération

Users that have purchased USB devices require the ability to upgrade the firmware of those devices with improved versions as they become available from manufacturers.

Device Firmware Upgrade is the mechanism for accomplishing that task.

Any class of USB device can exploit this capability by supporting the requirements specified in this document.

This document focuses on installing product enhancements and patches to devices that are already deployed in the field. Other potential uses for the firmware upgrade capability are beyond the scope of this document. Because it is impractical for a device to concurrently perform both DFU operations and its normal runtime activities, those normal activities must cease for the duration of the DFU operations. Doing so means that the device must change its operating mode; i.e., a printer is not a printer while it is undergoing a firmware upgrade; it is a PROM programmer.

However, a device that supports DFU is not capable of changing its mode of operation on its own volition. External (human or host operating system) intervention is required.

There are four distinct phases required to accomplish a firmware upgrade:
The device informs the host of its capabilities. A DFU class-interface descriptor and associated functional descriptor embedded within the device’s normal run-time descriptors serves this purpose and provides a target for class-specific requests over the control pipe.

The host and the device agree to initiate a firmware upgrade. The host issues a USB reset to the device, and the device then exports a second set of descriptors in preparation for the Transfer phase. This deactivates the run-time device drivers associated with the device and allows the DFU driver to reprogram the device’s firmware unhindered by any other communications traffic targeting the device.

The host transfers the firmware image to the device. The parameters specified in the functional descriptor are used to ensure correct block sizes and timing for programming the nonvolatile memories. Status requests are employed to maintain synchronization between the host and the device.

Once the device reports to the host that it has completed the reprogramming operations, the host issues a USB reset to the device. The device re-enumerates and executes the upgraded firmware. The device’s vendor ID, product ID, and serial number can be used to form an identifier used by the host operating system to uniquely identify the device. However, certain operating systems may use only the vendor and product IDs reported by a device to determine which drivers to load, regardless of the device class code reported by the device. (Host operating systems typically do not expect a device to change classes.) Therefore, to ensure that only the DFU driver is loaded, it is considered necessary to change the idProduct field of the device when it enumerates the DFU descriptor set. This ensures that the DFU driver will be loaded in cases where the operating system simply matches the vendor ID and product ID to a specific driver.

This document does not attempt to specify how a vendor might alter the device’s product ID except to suggest that adding one, setting the high bit, or using FFFFh are all valid possibilities. Vendors may use any scheme that they choose.

ATMEL dfu-programmer See Also:

- http://dfu-programmer.sourceforge.net/
- https://lists.sourceforge.net/lists/listinfo/dfu-programmer-user

Why dfu-programmer The need for this tool came about when I needed to flash an at89c51snd1c chip that had the USB bootloader on it, but the Atmel provided tool (FLIP) didn’t support USB flashing in Linux.

After a few days of web searching and scraping together a windows machine to do the job, I found that Atmel provided a document describing the communications protocol used. I happily spent the next week hacking together the start of dfu-programmer.

README file
dfu-programmer is an implementation of the Device Firmware Upgrade class USB driver that enables firmware upgrades for various USB enabled (with the correct bootloader) Atmel chips. This program was created because the Atmel "FLIP" program for flashing devices does not support flashing via USB on Linux, and because standard DFU loaders do not work for Atmel’s chips.

Check out the Atmel website for more information. They are kind enough to provide generally correct specifications this implementation is based on.
The project website is http://dfu-programmer.sourceforge.net and you can use that to check for updates.

Currently Supported Chips
==========================
8051 based:
  at89c51snd1c
  at89c5130
  at89c5131
  at89c5132

AVR based:
  at90usb1287
  at90usb1286
  at90usb647
  at90usb646

Simple install procedure
=========================

% tar -xzf dfu-programmer-<version>.tar.gz # unpack the sources
% cd dfu-programmer # change to the top-level directory

[ If the source was checked-out from CVS, run the following command ]
% ./bootstrap.sh # regenerate base config files
% ./configure # regenerate configure and run it

[ Optionally you can specify where dfu-programmer gets installed using the --prefix= option to the ./configure command. See ]
% ./configure --help for more details. ]
% make # build dfu-programmer
% make install # install dfu-programmer

Building RPM Binary Packages
============================
This section is intended to make it easier for those people that wish to build RPMs from the source included in this package, but aren’t sure how.

1) Copy dfu-programmer-<version>.tar.gz to your RPM SOURCES directory. Usually this is /usr/src/redhat/SOURCES/.

2) Extract or copy dfu-programmer.spec into your RPM SPECS directory. Usually this is /usr/src/redhat/SPECS/.

3) In your RPM SPECS directory, issue the command ’rpm -ba dfu-programmer.spec’. This will cause rpm to extract the dfu-programmer sources to a temporary directory, build them, and build rpm packages based on the information in the spec file. The binary rpms will be put into your RPM RPMS directory. Usually this is /usr/src/redhat/RPMS/<platform>/.

If you have any further questions, please refer to the RPM documentation.
dfu.c

dfu_device.h

arguments.h

pylibftdi

pylibftdi  See Also:

• http://pypi.python.org/pypi/pylibftdi/

pylibftdi is a minimal Pythonic interface to FTDI devices using libftdi.

Usage  The primary interface is the Device class in the pylibftdi package; this gives serial access on relevant FTDI devices (e.g. the UM232R), providing a file-like interface (read, write). Baudrate is controlled with the baudrate property.

If a Device instance is created with mode='t' (text mode) then read() and write() can use the given encoding (defaulting to latin-1).

This doesn’t make a lot of difference on Python 2 (and can be omitted), but allows easier integration with passing unicode strings between devices in Python 3.

Multiple devices are supported by passing the desired device serial number (as a string) in the device_id parameter - this is the first parameter in both Device() and BitBangDevice() constructors.

```python
>>> from pylibftdi import Device

>>> with Device(mode='t') as dev:
...    dev.baudrate = 115200
...    dev.write('Hello World')
```

The pylibftdi.BitBangDevice wrapper provides access to the parallel IO mode of operation through the port and direction properties. These provide an 8 bit IO port including all the relevant bit operations to make things simple:

```python
>>> from pylibftdi import BitBangDevice

>>> with BitBangDevice('FTE00P4L') as bb:
...    bb.direction = 0x0F  # four LSB are output(1), four MSB are input(0)
...    bb.port |= 2        # set bit 1
...    bb.port &= 0xFE     # clear bit 0
```

source

hg clone https://bitbucket.org/codedstructure/pylibftdi

Mac OS X usb

2.9. Drivers
USB macOS X  See Also:

win32 usb

USB win32

pywinusb  See Also:
  • http://code.google.com/p/pywinusb/

This project aims to be a simple USB/HID user application space (hence no drivers needed) 100% python package. Initially targeting simple HID devices management, also planned is support for WinUSB high level wrapping. The vision for this project is to be something similar to PySerial or PyParallel for USB/HID hardware enthusiasts.

Advantages
  • All python code, using ctypes
  • Top level handling of HID events (usage events calling hook handlers)

Limitations  Depending on your application you might find these limitations:
  • Windows only (so far...)

Using the package

Examples  There are some scripts included in the .examples folder, take a look

hook_button.py
  • hook_button.py, shows how to deal with input report usages, the basic idea is to use “event handlers”.

simple_send.py
  • simple_send.py, toggles (clicks) an hypothetical usage signal, two output reports are sent.

raw_data.py
  • raw_data.py, this script shows a list of available HID devices, you can pick up one in order to setup a simple custom input reports raw data handler (it just shows the data). Actually is a good simple utility to test your device.

pnp_sample.py
  • pnp_sample.py, shows how to use the Plug and Play helpers. The PnP helpers just require a window handler (hWnd) of your main frame. So, it does not depend on a particular window library (wxPython, pyQt, etc), but the sample uses wxPython as an example.
windows drivers

Windows Drivers install

Windows Drivers install whql

Windows libusb0.sys back end considerations

from Orin Eman <orin.eman@gmail.com> 
to graeme@argyllcms.com 
cc libusb-devel <libusb-devel@lists.sourceforge.net> 
date Tue, Apr 27, 2010 at 8:13 AM 
subject Re: [Libusb-devel] Windows libusb0.sys back end considerations.

On Mon, Apr 26, 2010 at 10:44 PM, Graeme Gill <graeme2@argyllcms.com> wrote:

Xiaofan Chen wrote:
> When you submit the driver packages for WHQL
> (with a Verisign certificate), the resultant sys file will also
> embedded the digital signature. This is what I expect.

I don’t think it works that way. The Microsoft documentation
assumes that you would only do this for test signing, and assumes
that you create a catalog file for the "real thing".
This is the case for the PRUFTECHNIK package.

Well, since I put a couple of drivers through WHQL last year... Only the cat file gets signed by WHQL. For the sys
files, you sign the sys files with your own code signing certificate along with a datetamp and cross-certificate that
enables the kernel to validate the signature on the driver. The WHQL signature enables the driver to be installed
silently - if you sign/datemstamp the cat file with your own certificate, the OS will bring up the “Do you trust...” dialog
when you try to install the driver. Keyboard filter drivers at least can be installed without WHQL signing - the driver
still needs to be signed/datemstamped and the cross-certificate included, but installation of the driver is just a matter of
adjusting a registry entry and service. Whether this would extend to a USB filter driver, I don’t know.

Orin.

from Orin Eman <orin.eman@gmail.com> 
to
cc libusb-devel <libusb-devel@lists.sourceforge.net> 
date Tue, Apr 27, 2010 at 8:20 AM 
subject Re: [Libusb-devel] Windows libusb0.sys back end considerations.

Xiaofan Chen wrote:
> When you submit the driver packages for WHQL
> (with a Verisign certificate), the resultant sys file will also
> embedded the digital signature. This is what I expect.

I don’t think it works that way. The Microsoft documentation
assumes that you would only do this for test signing, and assumes
that you create a catalog file for the "real thing".
This is the case for the PRUFTECHNIK package.

Well, since I put a couple of drivers through WHQL last year...
Only the cat file gets signed by WHQL. For the sys files, you sign the sys files with your own code signing certificate along with a datsertamp and cross-certificate that enables the kernel to validate the signature on the driver.

BTW, the OS will consider a signature invalid if you don’t include a timestamp and the current date and time falls out the range of valid dates for the code signing certificate... I’ve seen signed software from reputable sources reported as “Unknown Publisher” because the signature wasn’t timestamped and the certificate had run out.

Orin.

from Xiaofan Chen <xiaofanc@gmail.com> to graeme@argyllcms.com
cc libusb-devel <libusb-devel@lists.sourceforge.net>
date Tue, Apr 27, 2010 at 1:49 AM
subject Re: [Libusb-devel] Windows libusb0.sys back end considerations.

On Tue, Apr 27, 2010 at 1:44 PM, Graeme Gill <graeme2@argyllcms.com> wrote:
> Xiaofan Chen wrote:
>> When you submit the driver packages for WHQL
>> (with a Verisign certificate), the resultant sys file will also
>> embed the digital signature. This is what I expect.
> >
> I don’t think it works that way. The Microsoft documentation
> assumes that you would only do this for test signing, and assumes
> that you create a catalog file for the "real thing".
> > This is the case for the PRUFTECHNIK package.
> >
> I see. But somehow I have used the driver with no issue
> under my Win74 64bit box. I need to try again.

And supposedly Pruftechnik can use the same Verisign digital signature for both the WHQL and the KMCS since they already paid that. I would expect this to be the norm. Of course they can have a separate code signing certificate for the .sys driver and the Verisign one for WHQL.


“First there are two types of signing. There is digital signing to identify the vendor required for 64-bit Vista, that is what the link you listed below is about. Second there is the digital signature for passing WHQL and for accessing the WinQual database of crashes (when you get that nice prompt after a reboot from a crash that asks if you want to report this to Microsoft, it goes to WinQual). For the second VeriSign is the only one that is accepted. So basically, unless you only care about the digital signature needed for Vista 64-bit you need to go to Verisign.”

“I can confirm this. The certificate you use for submitting the WHQL stuff can be used to sign binaries for Vista. I’ve used it to sign a driver for Vista x64. In this case you just need to remember to use the cross-signing procedures described in the Vista x64 driver signing walkthrough.”

Official page: https://winqual.microsoft.com/help/default.htm#winqual_requirements.htm

And it seems they have a offer, now it is US$99.

Windows Drivers install zadig
Zadig usb_driver output [was: Automated libusb driver installation in Windows]

de  Pete Batard pete@akeo.ie via lists.sourceforge.net
à  libusb-devel@lists.sourceforge.net
date 26 septembre 2011 22:17
objet Re: [Libusb-devel] Zadig usb_driver output [was: Automated libusb driver installation in Windows]

On 2011.09.26 21:04, Peter Stuge wrote:
> No my issue has nothing to do with MODs. I just associated to my
> problem when I read "pre-install".

Ah OK.

Then I’d suggest you have a look at your C:Windowssetupapi.log after you plug your printer and it fails, as it should
tell us what Windows did during the driver installation (not sure if driver eligibility will be mentioned on XP). I
believe that the eligibility rules will still apply, i.e., if there’s a competition between your old WHQL driver and the
new WinUSB one, WHQL will win.

You may have to delete your old driver manually for WinUSB to work as a pre-installed driver in such a situation.

Smartcard storage driver (minidriver)  See Also:

• http://www.microsoft.com/whdc/device/input/smartcard/sc-minidriver.mspx

Smart card vendors can write card minidrivers to present a consistent interface to their smart card type to the Microsoft
Smart Card Base Cryptographic Service Provider (CSP) or Crypto Next Generation (CNG) Key Storage Provider
(KSP) and to the Smart Card Management Interface. These card minidrivers plug in to Windows operating system
code. The functionality in a card minidriver is narrowly scoped and carefully defined so that the card-dependent code
is simple to implement and easy to verify functionally.
OrcanthusStorageCard.inf

; Null Driver for Mifare Smartcard installation x86 and x64 package.

[Version]
Signature="$Windows NT$"
Class=SmartCard
ClassGuid={990A2BD7-E738-46c7-B26F-1CF8FB9F1391}
Provider=%ProviderName%
DriverVer=04/28/2010,1.0.0.0
CatalogFile=OrcanthusStorageCard.cat

[Manufacturer]
%ProviderName%=Minidriver,NTamd64,NTamd64.6.1,NTx86,NTx86.6.1

[Minidriver.NTamd64]
;This driver has no applicability on OS versions earlier than Windows 7

[Minidriver.NTx86]
;This driver has no applicability on OS versions earlier than Windows 7

[Minidriver.NTamd64.6.1]
; ghost card
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306000000000000000
; 14443A-1
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306010000000000000
; 14443A-2
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306020000000000000
; 14443A-3
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306030000000000000
; 14443A-4
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306040000000000000
; 14443A-5
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306050000000000000
; 14443A-6
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306060000000000000
; 14443A-7
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306070000000000000
; 14443A-8
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306080000000000000
; 14443A-9
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306090000000000000
; 14443A-10
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA0000003060A0000000000000
; 14443A-11
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA0000003060B0000000000000
; 14443A-12
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA0000003060C0000000000000
; 14443A-13

[Minidriver.NT686.6.1]
; ghost card
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306000000000000000
; 14443A-1
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306010000000000000
; 14443A-2
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306020000000000000
; 14443A-3
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306030000000000000
; 14443A-4
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306040000000000000
; 14443A-5
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306050000000000000
; 14443A-6
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306060000000000000
; 14443A-7
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306070000000000000
; 14443A-8
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306080000000000000
; 14443A-9
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA000000306090000000000000
; 14443A-10
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA0000003060A0000000000000
; 14443A-11
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA0000003060B0000000000000
; 14443A-12
%CardDeviceName%=Minidriver64_Install,SCFILTER\CID_804F0CA0000003060C0000000000000
; 14443A-13

;Leave the following sections blank
[DefaultInstall]
[DefaultInstall.ntamd64]
[DefaultInstall.NT686]
[DefaultInstall.ntamd64.6.1]
[DefaultInstall.NT686.6.1]
[Minidriver64_Install.NT]
[Minidriver32_Install.NT]
[Minidriver64_Install.NT.Services]
AddService = ,2,
[Minidriver32_Init.Install.NT.Services]
AddService = ,2,

; ============ Generic ============
[Strings]
ProviderName = "Orcanthus"
CardDeviceName = "Orcanthus Storage Card"

## 2.10 Free Software

### 2.10.1 Logiciel libre, Free software

See Also:
- [https://secure.wikimedia.org/wikipedia/fr/wiki/Free_software](https://secure.wikimedia.org/wikipedia/fr/wiki/Free_software)

Un logiciel libre est un logiciel dont l’utilisation, l’étude, la modification et la duplication en vue de sa diffusion sont permises (techniquement et légalement).

### Libre associations

#### April - Promouvoir et défendre le logiciel libre

See Also:
- [http://www.april.org/](http://www.april.org/)

Figure 2.11: *Logo de l’APRIL*

L’April est la principale association de promotion et la défense du logiciel libre dans l’espace francophone. À ce titre, elle entretient depuis le début des rapports étroits avec la Free Software Foundation, et était chargé de la traduction de la section philosophie du projet GNU.

En novembre 1996, des étudiants du laboratoire informatique de l’Université Paris VIII de Saint Denis créent une association dont le but est de promouvoir le logiciel libre en France. Le nom choisi est « April - Association pour la Promotion et la Recherche en Informatique Libre ».

Plus tard, cet acronyme sera abandonné pour être remplacé par le nom de l’association puis le slogan : « April - Promouvoir et défendre le logiciel libre ».

#### APRIL cours sur les logiciels libres

See Also:
- [http://wiki.april.org/w/Cours_logiciels_libres](http://wiki.april.org/w/Cours_logiciels_libres)

Pour ceux qui sont intéressé par la partie enseignement de l’informatique, nous avons commencé à degrossir ici : [http://wiki.april.org/w/Enseignement_de_l’informatique](http://wiki.april.org/w/Enseignement_de_l’informatique)
Framasoft  See Also:

- http://www.framasoft.net/

Figure 2.12: Logo Framasoft

Issu du monde éducatif, Framasoft est un réseau de sites web collaboratifs à géométrie variable dont le dénominateur commun est le logiciel libre, sa culture et son état d’esprit.

Il vise à diffuser le logiciel libre et à le faire connaître auprès du plus large public.

Lieu d’orientation, d’informations, d’actualités, d’échanges et de projets, Framasoft, de par la diversité et le dynamisme de son réseau, est aujourd’hui l’une des principales portes d’entrée francophones du logiciel libre.

Sa communauté d’utilisateurs est créatrice de ressources et apporte assistance et conseil à ceux qui découvrent et font leur premiers pas avec les logiciels libres.

Elle accompagne ceux qui désirent remplacer leurs logiciels propriétaires par des logiciels libres en attachant une attention toute particulière au processus de migration du système d’exploitation Microsoft Windows vers GNU/Linux.

Tout ce qui est produit est placé sous licence libre, favorisant ainsi la participation et garantissant que le travail effectué demeurera sans appropriation au bénéfice de tous.

Framasoft a été créé en novembre 2001 par Alexis Kauffmann. En janvier 2004 une association homonyme a vu le jour pour soutenir le développement du réseau.

GUILDE Guilde des Utilisateurs d’Informatique Libre du Dauphiné  See Also:

- http://www.guilde.asso.fr/

La Guilde des Utilisateurs d’Informatique Libre du Dauphiné (GUILDE) est une association loi 1901 ayant pour objectif de promouvoir les logiciels libres, et particulièrement le système GNU/Linux, auprès des particuliers et des professionnels et de rassembler les utilisateurs de la région du Dauphiné.

Listes de diffusion  See Also:

http://www.guilde.asso.fr/lurker/splash/index.html
Figure 2.13: *Logo de l’GUILDE*
LOG  See Also:
- http://www.logre.org/

Présentation  The LOG is a French hackerspace where geeks, artists and people interested in diy and funky activities alike meet and share ideas and fun.

Le LOG, c’est une association loi 1901 de passionnés de bidouillages en tout genre.

Electronique, informatique, mécanique, chimie, arts numériques, cuisine expérimentale, etc. : chacun amène son savoir-faire et sa curiosité dans un melting-pot ludique de création, bricolage et de détournement technologique en tout genre.

Vous avez un projet à partager ? C’est l’occasion de le présenter, et peut-être trouver de nouveaux contributeurs.

L’une des vocations du LOG est de fournir à ses membres un lieu pour héberger leurs projets ainsi que d’organiser des présentations publiques. Il est aussi ouvert sur l’extérieur par le biais de partenariats avec les institutionnels, animations ou participation à des événements.

Le LOG est une émanation du mouvement mondial Hackerspace et se veut ouvert aux échanges et collaborations avec les autres groupes.

Listes de diffusion  See Also:
http://www.logre.org/#liens

Réseaux sociaux
- identica http://identi.ca/logre

Conferences sur le Libre

Conférences Logiciel libre

Conférences RMLL  See Also:

Les Rencontres mondiales du logiciel libre, ou RMLL, sont, selon leurs organisateurs, le plus grand rendez-vous non commercial au monde entièrement consacré au logiciel libre

RMLL 2011  See Also:
http://2011.rmll.info

RMLL 2010  See Also:
http://2010.rmll.info

Conférences Fosdem  See Also:
- http://fosdem.org
- http://video.fosdem.org/
**Contents**

- Conférences Fosdem
  - Presentation
  - FOSDEM Videos
  - FOSDEM conferences

**Presentation**

FOSDEM is the biggest free and non-commercial event organized by and for the community. Its goal is to provide Free and Open Source developers a place to meet. No registration necessary.

La FOSDEM (Free and Open Source Software Developers’ European Meeting), réunion européenne des développeurs de logiciels libres et open source) est un ensemble de conférences qui se déroulent annuellement pendant un week-end.

**FOSDEM Videos**

See Also:

http://video.fosdem.org/

These talks have been recorded at FOSDEM, The Free and Open Source Software Developers’ European Meeting - http://www.fosdem.org

All videos are available in:

- DIRECT DOWNLOAD on http://video.fosdem.org
- STREAMED on http://youtube.com/fosdemtalks

Special thanks to:

- NamurLUG for the recording <http://www.namurlug.org>
- Belnet, Heanet, AARNET, dotsrc.org, geeksoc.org for the bandwidth

Should you have any question, please contact <info@fosdem.org>

**FOSDEM conferences**

**Fosdem 2012**

See Also:

http://fosdem.org/2012/

**Contents**

- Fosdem 2012
  - Blogs Links
  - Videos

**Blogs Links**

See Also:

- http://toscalix.blogspot.com/2012/02/fosdem-2012-for-toscalix.html

**Videos**

See Also:

http://video.fosdem.org/2012/
Fosdem 2011

Vidéos 2011   See Also:

http://linuxfr.org/news/fosdem-2011%C2%A0-les-vid%C3%A9os-sont-d%C3%A9j%C3%A0-en-ligne%E2%80%AF

Le Linux User Group de Namur (Belgique) réalise bénévolement depuis plusieurs années la capture vidéo des principales conférences du FOSDEM (Free and Open source Software Developer’s European Meeting), le grand rendez-vous européen des développeurs de logiciels libres et «open source» qui se déroule traditionnellement à Bruxelles au début du mois de février.

Cette année, nous avons encore amélioré notre prestation en fournissant un flux audio en direct, et un flux vidéo en léger différé durant tout le week-end (5-6 février 2011). Un tutoriel décrivant les procédures utilisées (100% logiciel libre sous Linux, évidemment) viendra bientôt s’ajouter à ceux des années précédentes, déjà disponibles sur notre site.

Toutes les vidéos 2011 ont maintenant été ré-encodées en XVID (conteneur AVI) et sont disponibles en accès libre sur le serveur vidéo du FOSDEM comme celles de toutes les éditions depuis 2005.

Videos   These talks have been recorded @FOSDEM The Free and OpenSource Software Developers’ European Meeting http://www.fosdem.org

All videos are available in:

• DIRECT DOWNLOAD on http://video.fosdem.org
• STREAMED on http://youtube.com/fosdemtalks

Special thanks to:

• NamurLUG for the recording <http://www.namurlug.org>
• Belnet, Heanet, Unixheads, dotsrc.org, geeksoc.org for the bandwidth

for any question, please contact <info@fosdem.org>

Conférences Fossa

Fossa 2011   See Also:

• http://fossa.inria.fr/
• http://philippe.scoffoni.net/je-serais-a-fossa-2011/

Fossa 2010   See Also:

http://fossa2010.inrialpes.fr/

Fossa 2009   See Also:

Open source projects sites

Open Source projects site

NASA Open Source projects site  See Also:
http://code.nasa.gov/

Figure 2.14: Nasa plan

Today we are launching code.nasa.gov, the latest member of the open NASA web family. Through this website, we will continue, unify, and expand NASA’s open source activities. The site will serve to surface existing projects, provide a forum for discussing projects and processes, and guide internal and external groups in open development, release, and contribution.

2.11 Firmwares

2.11.1 Firmware

Board Support Package

See Also:

- Thomas Petazzoni
- Linux embedded bootloaders

Contents

- Board Support Package
  - Intro (français)
  - Intro (anglais)
Intro (français)

Un Board Support Package ou BSP est un logiciel bas niveau de support de cartes-mères, c’est-à-dire entre l’OS et la carte mère, dans le domaine de l’informatique embarquée.
Par exemple, Linaro construit le BSP pour les processeurs ARM.
SHAI sera l’interface standardisée entre le système d’exploitation symbian, et les BSP supportant différents matériels.

Intro (anglais)

In embedded systems, a board support package (BSP) is implementation specific support code for a given (device motherboard) board that conforms to a given operating system.
It is commonly built with a bootloader that contains the minimal device support to load the operating system and device drivers for all the devices on the board.
Some suppliers also provide a root file system, a toolchain for making programs to run on the embedded system (which would be part of the architecture support package), and configurators for the devices (while running).

Firmware linux

Firmware linux tests

fwts  See Also:
- http://www.hecticgeek.com/2012/02/firmware-test-debugger-ubuntu-linux/

2.12 Glossary

2.12.1 Development Glossary

See Also:
Smartcard Glossary
DFU  Device Firmware Upgrade
Firmware  Executable software stored in a write-able, nonvolatile memory on a USB device.
FMDS  fiabilité, maintenabilité, disponibilité, sécurité
KibiByte
    See Also:
RAMS  Reliability, Availability, Maintainability and Safety
Smart Card  Any of a number of similar devices conforming to ISO/IEC 7816-3.
sphinx  Sphinx is a tool that makes it easy to create intelligent and beautiful documentation, written by Georg Brandl and licensed under the BSD license.

See Also:

http://sphinx.pocoo.org/latest/

USB  USB (Universal Serial Bus) is a specification to establish communication between devices and a host controller (usually personal computers). USB is intended to replace many varieties of serial and parallel ports. USB can connect computer peripherals such as mice, keyboards, digital cameras, printers, personal media players, flash drives, and external hard drives.

See Also:

Universal Serial Bus

2.13 GUI (Graphical User Interface)

2.13.1 Graphical User Interface

Android design

See Also:

http://developer.android.com/design/index.html

AVC

See Also:

• http://avc.inrim.it/html/

AVC is a multiplatform, fully automatic, live connection among graphical interface widgets and application variables for the python language. AVC supports in a uniform way the most popular widget toolkits: GTK+, Qt3, Qt4, Tk, wxWidgets.

The Swing widget toolkit for the java environment is also supported via the jython compiler. AVC is a python package that can be imported by any python or jython application.

gnome

See Also:

• http://en.wikipedia.org/wiki/GNOME
  • GNOME

GNOME (abbreviation of GNU Network Object Model Environment) is a desktop environment—a graphical user interface that runs on top of a computer operating system—composed entirely of free and open source software. It was created by two Mexican programmers, Miguel de Icaza and Federico Mena. It is an international project that includes creating software development frameworks, selecting application software for the desktop, and working on the programs that manage application launching, file handling, and window and task management.

GNOME is part of the GNU Project and can be used with various Unix-like operating systems, most notably GNU/Linux, and as part of the Java Desktop System in Solaris.
Major subprojects

GNOME is built from a large number of different projects. A few of the major ones are listed below:

- Bonobo – a (obsolete in current releases) compound document technology.
- GConf – for storing application settings.
- GVFS – a virtual file system.
- GNOME Keyring – for storing encryption keys and security information.
- GNOME Translation Project – translate documentation and applications into different languages.
- GTK+ – a widget toolkit used for constructing graphical applications. The use of GTK+ as the base widget toolkit allows GNOME to benefit from certain features such as theming (the ability to change the look of an application) and smooth anti-aliased graphics. Sub-projects of GTK+ provide object-oriented programming support (GObject), extensive support of international character sets and text layout (Pango) and accessibility (ATK). GTK+ reduces the amount of work required to port GNOME applications to other platforms such as Windows and Mac OS X.
- Human interface guidelines (HIG) – research and documentation on building easy-to-use GNOME applications.
- LibXML – an XML library.

A number of language bindings are available allowing applications to be written in a variety of programming languages, such as C++ (gtkmm), Java (java-gnome), Ruby (ruby-gnome2), C# (Gtk#), Python (PyGTK), Perl (gtk2-perl) and many others. The only languages currently used in applications that are part of an official GNOME desktop release are C, C#, Python and Vala.

fontconfig

- http://www.fontconfig.org/wiki/
- http://fontconfig.org/release/

pixman


cairo

- http://www.cairographics.org/

glib


GLib is a cross-platform software utility library that began as part of the GTK+ project, however, before releasing version 2 of GTK+, the project’s developers decided to separate non-GUI-specific code from the GTK+ platform, thus creating GLib as a separate product. GLib was released as a separate library so other developers, those that did not make use of the GUI-related portions of GTK+, could make use of the non-GUI portions of the library without the overhead of depending on a full-blown GUI library.
Since GLib is a cross-platform library, applications using it to interface with the operating system are usually portable across different operating systems without major changes.

**Similar projects**  For many applications, C with GLib is an alternative to C++ with STL (see GObject for a detailed comparison).

The Apache Portable Runtime has a large functional overlap with GLib, and provides many similar OS-portable threading, network and data structure implementations in C.

Other widget toolkits usually also provide low-level functions and implementations of data structures. For instance, in the wxWidgets library the non-GUI functions are in the wxBase library, and in Qt the non-GUI parts are in the QtCore module, which is written in C++.

**atk**


In computing, Accessibility Toolkit (ATK) refers in particular to the GNOME ATK.

The GNOME ATK, a developer toolkit, allows programmers to use common GNOME accessibility features in their applications. This includes such features as high-contrast visual themes for the visually-impaired and keyboard behaviour modifiers (e.g. sticky keys) for those with diminished motor control.

**pango**

- [http://www.pango.org/](http://www.pango.org/)

Pango is an LGPL licensed open source computing library used by software developers for laying out and rendering text in high quality, emphasising support for multilingual text. Different font backends can be used, allowing cross-platform support, so that Pango-rendered text will appear similar under different operating systems, such as Linux, Apple’s MacOS and Microsoft Windows.

**GTK+**


GTK+ (GIMP ToolKit) is a cross-platform widget toolkit for creating graphical user interfaces. It is one of the most popular toolkits for the X Window System, along with Qt.

GTK+ was initially created for the GNU Image Manipulation Program (GIMP), a raster graphics editor, in 1997 by Spencer Kimball and Peter Mattis, members of eXperimental Computing Facility (XCF) at University of California, Berkeley.

GTK+ is licensed under the LGPL free software license and is part of the GNU Project, which aims to create a whole free-software operating system.

**History**  GTK+ was originally designed and used in the GNU Image Manipulation Program (GIMP) as a replacement of the Motif toolkit; at some point Peter Mattis became disenchanted with Motif and began to write his own GUI toolkit called the GIMP toolkit and had successfully replaced Motif by the 0.60 release of GIMP.[2] Finally GTK was re-written to be object oriented and was renamed GTK+. This was first used in the 0.99 release of GIMP.
GTK+ 2 has succeeded GTK+. Its new features include improved text rendering using Pango, a new theme engine, improved accessibility using the Accessibility Toolkit, complete transition to Unicode using UTF-8 strings, and a more flexible API. However, GTK+ 2 lacks compatibility with GTK+ 1, and programmers must port applications to it.

Versions

2.20.1


Common questions:

- http://www.gtk.org/faq/

GTK+ dependencies

- fontconfig
- pixman
- cairo
- glib
- atk
- pango
- gtk+

Compiling gtk+ under centos 5.5 To compile the gtk+ library the following libraries must be compiled in this order:

- fontconfig
- pixmap
- cairo
- glib
- atk
- pango
- gtk+

Installing from sources

Setting the environment variables, PATH, PKG_CONFIG_PATH, LD_LIBRARY_PATH

export PATH=$PATH:/opt/gtk2/bin
export PKG_CONFIG_PATH=/opt/gtk2/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gtk2/lib
tar xvf gtk+-X.Y.Z.tar.bz2

cd /tmp

tar xvf gtk+-2.20.1.tar.bz2

cd gtk+-X.Y.Z

cd gtk+-2.20.1

make distclean

make distclean

./configure --prefix=/opt/gtk2

./configure --prefix=/opt/gtk2

make

make

make install

make install

**Compilations examples for GIMP** Instructions to compile the current GIMP source code (master version in GIT repository) with Ubuntu Linux 9.04 + 9.10

Important: The current GIT master version is a snapshot, an intermediate version that can be used to follow current developments (for instance, the single window mode). For production use, always use a stable GIMP version (2.6)!

The procedure is the same for Ubuntu 9.04 and 9.10, but with Ubuntu 9.04 you have to download & compile the newest Gtk version.

# (as non-root user)

```
cd ~
mkdir -p tmp
cd tmp

# Important! These variables have to be set and # are required for all further commands. If you close the # terminal window, you have to give the 3 export commands # again.

export PATH=$PATH:/opt/gimp-2.7/bin
export PKG_CONFIG_PATH=/opt/gimp-2.7/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gimp-2.7/lib

# Fetch the most important packages
sudo apt-get build-dep gimp
```
# Additional packages
sudo aptitude install checkinstall git-core libtool libopenexr-dev libopenraw-dev libspiro-dev
### BEGIN: ONLY NEEDED FOR UBUNTU 9.04 ###
# Fetch, compile, install Glib (needed for Gtk)
wget http://ftp.gnome.org/pub/gnome/sources/glib/2.22/glib-2.22.2.tar.bz2
tar -xjf glib-2.22.2.tar.bz2
cd glib-2.22.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..
# Fetch, compile, install Gtk
wget http://ftp.gnome.org/pub/gnome/sources/gtk+/2.18/gtk+-2.18.2.tar.bz2
tar -xjf gtk+-2.18.2.tar.bz2
cd gtk+-2.18.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..
### END: ONLY NEEDED FOR UBUNTU 9.04 ###
# Fetch, compile, install BABL
git clone --depth 1 git://git.gnome.org/babl
cd babl
./autogen.sh --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..
# Fetch, compile, install GEGL
git clone --depth 1 git://git.gnome.org/gegl
cd gegl
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..
# Fetch, compile, install GIMP
git clone --depth 1 git://git.gnome.org/gimp
cd gimp
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..
Launch GIMP with:
/opt/gimp-2.7/bin/gimp-2.7

See also


**Compiling gtk+ under centos 5.5** To compile the gtk+ library the following libraries must be compiled in this order:

- fontconfig
- pixmap
- cairo
- glib
Development tools, Release 2012.06.18

• atk
• pango
• gtk+

Setting the environment variables

export PATH=$PATH:/opt/gtk2/bin
export PKG_CONFIG_PATH=/opt/gtk2/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gtk2/lib

./configure --prefix=/opt/gtk2

Compilations examples for GIMP  Instructions to compile the current GIMP source code (master version in GIT repository) with Ubuntu Linux 9.04 + 9.10

Important: The current GIT master version is a snapshot, an intermediate version that can be used to follow current developments (for instance, the single window mode). For production use, always use a stable GIMP version (2.6)!

The procedure is the same for Ubuntu 9.04 and 9.10, but with Ubuntu 9.04 you have to download & compile the newest Gtk version.

# (as non-root user)

cd ~
mkdir -p tmp
cd tmp

# Important! These variables have to be set and # are required for all further commands. If you close the # terminal window, you have to give the 3 export commands # again.

export PATH=$PATH:/opt/gimp-2.7/bin
export PKG_CONFIG_PATH=/opt/gimp-2.7/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gimp-2.7/lib

# Fetch the most important packages
sudo apt-get build-dep gimp
# Additional packages
sudo aptitude install checkinstall git-core libtool libopenexr-dev libopenraw-dev libspiro-dev
### BEGIN: ONLY NEEDED FOR UBUNTU 9.04 ###
# Fetch, compile, install Glib (needed for Gtk)
wget http://ftp.gnome.org/pub/gnome/sources/glib/2.22/glib-2.22.2.tar.bz2
tar -xjf glib-2.22.2.tar.bz2
cd glib-2.22.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..
# Fetch, compile, install Gtk
wget http://ftp.gnome.org/pub/gnome/sources/gtk+/2.18/gtk+-2.18.2.tar.bz2
tar -xjf gtk+-2.18.2.tar.bz2
cd gtk+-2.18.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..
### END: ONLY NEEDED FOR UBUNTU 9.04 ###

# Fetch, compile, install BABL
git clone --depth 1 git://git.gnome.org/babl
cd babl
./autogen.sh --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install GEGL
git clone --depth 1 git://git.gnome.org/gegl
cd gegl
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install GIMP
git clone --depth 1 git://git.gnome.org/gimp
cd gimp
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..
Launch GIMP with:
/opt/gimp-2.7/bin/gimp-2.7

pygobject


Dependencies

- [gobject introspection](http://live.gnome.org/)

Versions

2.21.5 PyGObject requires glib >= 2.22.4 and Python >= 2.3.5 to build. GIO bindings require glib >= 2.22.4.


pyGTK

- [http://live.gnome.org/PyGTK](http://live.gnome.org/PyGTK)

PyGTK is a set of Python wrappers for the GTK+ graphical user interface library. PyGTK is free software and licensed under the LGPL. Other popular alternatives are PyQt and wxPython. Its original author is the prominent GNOME developer James Henstridge. Today there are six people in the core development team, with various other people who have submitted patches and bug reports. PyGTK has been selected as the environment of choice for applications running on the One Laptop Per Child systems. Developers and interested parties can usually be found on the IRC channel #pygtk on irc.gnome.org.

2.13. GUI (Graphical User Interface)
Dependencies

- GTK+
- pygobject

GNOME dependencies  If you want to make use of the Gnome libraries in your application, you may wish to install the gnome-python and gnome-python-desktop packages. In addition to this, gnome-python-extras is also available (contains less frequently used packages not in gnome-python). All are available from the Gnome FTP site and its mirrors:

Development

- git clone git://git.gnome.org/pygtk
- git clone git://git.gnome.org/gnome-python
- git clone git://git.gnome.org/gnome-python-desktop
- git clone git://git.gnome.org/gnome-python-extras

Versions

2.16.0

See also

- http://aciresnippets.wordpress.com/ (The Acire Project is simple: to provide a library of Python snippets and examples that demonstrate how to perform specific tasks. Many of us learn by example, and having a library of examples help us learn different modules faster and easier, helping us to write awesome programs faster and easier.

Compiling gtk+ under centos 5.5  To compile the gtk+ library the following libraries must be compiled in this order:

- compile gtk+
- python 2.6 (not python 2.7)
- gobject

Installing from sources

Setting the environment variables

```
export PATH=/opt/gtk2/bin:$PATH
export PKG_CONFIG_PATH=/opt/gtk2/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gtk2/lib
```
Extracting the sources

cd /tmp
tar xvf gtk+-2.20.1.tar.bz2

cd
cd gtk+-2.20.1

make distclean
make distclean

./configure --prefix=/opt/gtk2
./configure --prefix=/opt/gtk2

make
make

make install
make install

Compilations examples for GIMP Instructions to compile the current GIMP source code (master version in GIT repository) with Ubuntu Linux 9.04 + 9.10

Important: The current GIT master version is a snapshot, an intermediate version that can be used to follow current developments (for instance, the single window mode). For production use, always use a stable GIMP version (2.6)!

The procedure is the same for Ubuntu 9.04 and 9.10, but with Ubuntu 9.04 you have to download & compile the newest Gtk version.

# (as non-root user)

cd ~
mkdir -p tmp
cd tmp

# Important! These variables have to be set and # are required for all further commands. If you close the # terminal window, you have to give the 3 export commands # again.

export PATH=$PATH:/opt/gimp-2.7/bin
export PKG_CONFIG_PATH=/opt/gimp-2.7/lib/pkgconfig
export LD_LIBRARY_PATH=/opt/gimp-2.7/lib

# Fetch the most important packages
sudo apt-get build-dep gimp
# Additional packages
sudo aptitude install checkinstall git-core libtool libopenexr-dev libopenraw-dev libspiro-dev

### BEGIN: ONLY NEEDED FOR UBUNTU 9.04 ###

# Fetch, compile, install Glib (needed for Gtk)
wget http://ftp.gnome.org/pub/gnome/sources/glib/2.22/glib-2.22.2.tar.bz2
tar -xjf glib-2.22.2.tar.bz2
cd glib-2.22.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install Gtk
wget http://ftp.gnome.org/pub/gnome/sources/gtk+/2.18/gtk+-2.18.2.tar.bz2
tar -xjf gtk+-2.18.2.tar.bz2
cd gtk+-2.18.2
./configure --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..

### END: ONLY NEEDED FOR UBUNTU 9.04 ###

# Fetch, compile, install BABL
git clone --depth 1 git://git.gnome.org/babl
cd babl
./autogen.sh --prefix=/opt/gimp-2.7
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install GEGL
git clone --depth 1 git://git.gnome.org/gegl
cd gegl
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..

# Fetch, compile, install GIMP
git clone --depth 1 git://git.gnome.org/gimp
cd gimp
./autogen.sh --prefix=/opt/gimp-2.7 --disable-gtk-doc
make -j3
sudo make install -j3
cd ..

Launch GIMP with:
/opt/gimp-2.7/bin/gimp-2.7

**pyCairo**
- http://cairographics.org/pycairo/

**gnome python**

**Installation** The following modules will be built:
- gnome
• gnomecanvas
• gnomevfs
• gnome-vfs python module support
• gconf

The following modules will NOT be built:
• gnome.ui
• gnomevfs.bonobo
• bonobo
• bonobo.activation
• bonobo.ui

Note: PyORBit 2.0.1 or later is required by bonobo and gnomeui bindings. Download it from:
http://ftp.gnome.org/pub/GNOME/sources/pyorbit/2.0/

Versions

2.28.1
• http://ftp.gnome.org/pub/GNOME/sources/gnome-python/2.28/

gobject
• http://en.wikipedia.org/wiki/GObject

The GLib Object System, or GObject, is a free software library (covered by the LGPL) providing a portable object system and transparent cross-language interoperability. GObject is designed for use both directly in C programs to provide object-oriented C-based APIs, and through bindings to other languages to provide transparent cross-language interoperability.

History  Depending only on GLib and libc, GObject is a cornerstone of GNOME and is used throughout GTK+, Pango, Accessibility Toolkit, and most higher-level GNOME libraries and applications. Prior to GTK+ 2.0, the GObject code was part of the GTK+ codebase. (The name “GObject” was not yet in use — the common baseclass was called GtkObject.)

The release of GTK+ 2.0 had the object system extracted into a separate library due to its general utility. In the process, most non-GUI-specific parts of the GtkObject class were moved up into GObject, the new common baseclass. Having existed as a separate library since March 11, 2002 (the release date of GTK+ 2.0), the GObject library is now used by many non-GUI programs such as command-line and server applications.

Relation to GLib  Though GObject has its own separate set of documentation and is usually compiled into its own shared library file, the source code for GObject resides in the GLib source tree and is distributed along with GLib. For this reason, GObject uses the GLib version numbers and is typically packaged together with GLib (for example, Debian puts GObject in its libglib2.0 package family).

gobject introspection
• http://live.gnome.org/GObjectIntrospection
Cela fait presque sept ans que je fais du PAC sans le savoir !!!!! J’avais commencé avec RealBasic sur un Mac 68K sous MacOS 7.1 ! Bon, il faut dire que je ne suis pas informaticien professionnel alors, je peux me passer de concept fumeux des commerciaux et autres thèseux. Comme je suis un pur hacker (j’ai appris sur le tas et je perds un tas de temps à expérimenter) je vais te donner mes règles d’or en matière de programmation orienté objet :

1. Un Objet graphique se gère lui-même. (ex : Une fenêtre ne gère que les événements qui lui sont dédiés)
2. Dès qu’un objet graphique devient complexe (plusieurs événements à gérer) il faut en faire un objet à part.
3. Les données doivent être le plus proche possible de la couche logiciel de base afin de gagner en rapidité et en portabilité. ex : Utilisation des STL en C++ ou de la lib standard en Python, etc. Quel est l’intérêt d’implémenter QHttp en python alors qu’il y a la lib urllib2 et httplib, meilleurs et plus rapide ?
4. Documentez, Documentez, il en restera toujours quelque chose
5. Une chose, un objet, un fichier (bon d’accord, des fois on a un objet de six lignes #include compris)
6. Tout ce que je viens de dire est idiot. En gros, fait ce qu’il te plaît.

Alors, pas mal non ?

PS : Tu veux un coup de main (pas trop fort sur la joue) pour ton prog, j’aimerais t’aider, je tourne un peu en rond en ce moment.

Qt GUI

See Also:

- Qt Integrated Development Environment
Contents

- Qt GUI
  - Introduction
  - Qt on ubuntu
  - Qt doc
  - pyside, pyqt, pythonqt

Introduction

C++ a été écrit à de nombreuses reprises dans cet article mais tout cela s’applique également à tout langage tel que Python, Ruby ou tout autre langage avec les bindings Qt.

L’objectif n’est pas d’apprendre à réaliser une interface utilisateur en utilisant le C++ et Qt. Au lieu de cela, pourquoi ne pas regarder comment garantir que l’interface utilisateur et la partie logique soient gardées séparées ?

Lorsque l’on s’appuie sur Qt Quick ou HTML 5, on est forcé de définir une interface entre la partie logique et l’interface utilisateur. Lors de l’utilisation du C++ pour réaliser à la fois la partie logique et l’interface utilisateur, il est nécessaire de placer ces restrictions sur soi-même.

La clé ici est que tout code affectant uniquement l’interface utilisateur fait partie de l’interface utilisateur tandis que tout code affectant l’état de l’application (par exemple, le contenu d’un document) est membre de la partie logique.

La partie logique nécessite d’être maintenue dans un ensemble séparé de classes. Pour réellement se forcer à atteindre cet objectif, un objet de document pour chaque fenêtre de document est une bonne idée.

Pour les autres types d’applications, l’objet contextuel en provenance d’une intégration Qt Quick peut être réutilisé.

Différents utilisateurs attendent différentes interfaces utilisateur. Dans des systèmes multiutilisateurs, ces interfaces pourraient se différencier non seulement par l’apparence et les fonctionnalités, mais encore par la manière dont elles sont implémentées.

Qt supporte des interfaces classiques basées sur des widgets, tout comme des interfaces plus modernes via Qt Quick. En plus de ces deux technologies spécifiques à Qt, HTML5 est aussi supporté, grâce à l’intégration avec WebKit.

En séparant l’application en des parties séparées et réutilisables, on peut utiliser le même code logique pour créer des expériences utilisateur pour tous les utilisateurs - le tout avec Qt.

Qt on ubuntu

See Also:

http://www.markshuttleworth.com/archives/568

As part of our planning for Natty+1, we’ll need to find some space on the CD for Qt libraries, and we will evaluate applications developed with Qt for inclusion on the CD and default install of Ubuntu.

Ease of use, and effective integration, are key values in our user experience. We care that the applications we choose are harmonious with one another and the system as a whole.

Historically, that has meant that we’ve given very strong preference to applications written using Gtk, because a certain amount of harmony comes by default from the use of the same developer toolkit.

That said, with OpenOffice and Firefox having been there from the start, Gtk is clearly not an absolute requirement.

What I’m arguing now is that it’s the values which are important, and the toolkit is only a means to that end. We should evaluate apps on the basis of how well they meet the requirement, not prejudice them on the basis of technical choices made by the developer.
In evaluating an app for the Ubuntu default install, we should ask:

- is it free software?
- is it best-in-class?
- does it integrate with the system settings and preferences?
- does it integrate with other applications?
- is it accessible to people who cannot use a mouse, or keyboard?
- does it look and feel consistent with the rest of the system?

Of course, the developer’s choice of Qt has no influence on the first two.

Qt itself has been available under the GPL for a long time, and more recently became available under the LGPL. And there’s plenty of best-in-class software written with Qt, it’s a very capable toolkit.

...  

The decision to be open to Qt is in no way a criticism of GNOME. It’s a celebration of free software’s diversity and complexity. Those values of ease of use and integration remain shared values with GNOME, and a great basis for collaboration with GNOME developers and project members.

Perhaps GNOME itself will embrace Qt, perhaps not, but if it does then our willingness to blaze this trail would be a contribution in leadership. It’s much easier to make a vibrant ecosystem if you accept a certain amount of divergence from the canonical way, so to speak Our work on design is centered around GNOME, with settings and preferences the current focus as we move to GNOME 3.0 and gtk3.

Of course, this is a perfect opportunity for those who would poke fun at that relationship to do so, but in my view what matters most is the solid relationship we have with people who actually write applications under the GNOME banner. We want to be the very best way to make the hard work of those free software developers matter, by which we mean, the best way to ensure it makes a real difference in millions of lives every day, and the best way to connect them to their users.

To the good folks at Trolltech, now Nokia, who have made Qt a great toolkit – thank you.

To developers who wish to use it and be part of the Ubuntu experience – welcome.

See Also:


Qt doc

Qt accessibility

See Also:

http://blog.didrocks.fr/post/Accessible-Qt-now-in-Oneiric!

When we announced that Unity 2D will be the fallback if you have no required hardware or driver for unity 3D, one of the compulsory request for that is that Unity 2D should be accessible. Even if one of the goal for the cycle is to make Unity fully accessible[1] we should still consider people needed accessibility and not filling the requirements for it.
In addition to that, we have now in Oneiric Qt installed by default as previously decided, and if we want to be toolkit agnostic, we need a good accessibility story on that toolkit as well… and Qt is known to have a blurry story about accessibility :)

Unity 2D is written in QML, and so based on Qt technology. So what happens?

It’s been already some months that Qt people are working on accessibility in Qt and QML itself, basically working with at-spi2 technology (the same used in GNOME).

Last week, a lot of Qt contributors gathered at the and I had the chance to spend some time in Berlin there too. The awesome and very friendly Frederik Gladhorn worked to backport the QML accessible branch (done in an development 4.8 branch, with some nice features of Qt 5 :) ) to cut all non relevant parts down and backport to a 4.7 branch (as we have 4.7.3 in ubuntu and will probably stick with 4.7.4 in Oneiric looking at the schedule).

I just started from it, rebased on 4.7.3 and uploaded today (after some staging test days in the ubuntu-desktop ppa) to Oneiric!

So, if you upgrade your oneiric box and install as well the new qt-at-spi package (in universe right now, but should be soon land in main), you will get an accessible Qt and QML! To activate it (it’s not activated by the global configuration over dbus right now), launch Orca or accerciser (common accessibility tools) and run your Qt application with QT_ACCESSIBILITY=1.

We know right now that it’s a little crashy (only with the environment variable set), but do not hesitate to test! Please report crashes/issues to the Qt package in launchpad and tag them a11y so that it’s easier for us to find and leverage issues upstream. We can’t do it without your helps and we need feedbacks, so please test. :)

That won’t unfortunately make unity 2D instantly accessible though. Indeed, QML has not right now a standard desktop toolkit[2], every QML project creates right now its own components, and so, there is no accessibility magic to now what to expose or not. Consequently, the unity 2D guys will implement and test the new property to be set so that Orca (for instance), can read the widgets content. Accerciser though already introspects unity 2D and other QML projects content, which means that it seems to work. :)

Thanks again to Frederik, it’s a pleasure to work with him and thanks to all guys working on Qt making that possible! Let’s continue working together and push accessibility in Qt and QML as far as possible.

Qt GUI applications

pyQt4 2ManDVD application See Also:

• images 2mandvd tool

The Face Recognition System based on QT and OpenCV See Also:


An essay that imitate the format of intel cup:The Face Recognition System based on QT and OpenCV

Related downloads http://min.us/m1m2L9I0r The files in this package are suited to FriendlyARM’s Tiny6410,it contains OpenCV2.3 and QT4.7.3,it has been compiled and can be used directly. Please put the files in opencv-lib in folder /lib,and uncompress qt4.7.3.tgz to /opt. It also contains ORL face database which i scaled to 70x80, and some xml files. faceRec is for command-line training in this project(it can also used for command-line recognition).

This project used FriendlyARM’s Tiny6410, actually PC’s program is much simpler and faster. This program can train and recognize human face. It uses common USB camera to capture images through v4l2, and detect face with AdaBoost algorithm, and then recognize face in PCA method. If it recognized a face in the face database, it will send character ‘1’ to RS232, if a person stand in front of the camera for about a minute and not logged in, then it will send ‘2’.

The speed is not ideal, about 1 frame per second. As for desktop linux, you can just change the device name and compile the desktop version of OpenCV lib, that’s enough. Welcome to fork.
Specially thanks to Shervin Emami’s article: http://www.shervinemami.info/faceRecognition.html I’m trying to translate it into Chinese, but it still doesn’t reach 50%. I hope more people will participate it.

(I will still improve the wiki, and I hope Computer Vision lovers will continue the project...)

You are free:

- to Share — to copy, distribute and transmit the work to Remix — to adapt the work Under the following conditions:
  - Attribution You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
  - Noncommercial — You may not use this work for commercial purposes.
  - Share Alike — If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

**FET Taurus application  See Also:**

- http://lalescu.ro/liviu/fet/
- http://freshmeat.net/projects/fet#release_329879

FET is open source free software for automatically scheduling the timetable of a school, high-school or university.

It uses a fast and efficient timetabling algorithm. It is licensed under GNU GPL.

Usually, FET is able to solve a complicated timetable in maximum 5-20 minutes.

For simpler timetables, it may take a shorter time, under 5 minutes (in some cases, a matter of seconds).

For extremely difficult timetables, it may take a longer time, a matter of hours.

**pyQt thg application  See Also:**

- http://tortoisehg.bitbucket.org/
- http://tortoisehg.bitbucket.org/manual/2.0/
- TortoiseHg

**pyQt Taurus application  See Also:**


Taurus is a python framework for both CLI and GUI tango applications. It is build on top of PyTango and PyQt.

Taurus stands for TAngo User interface ‘R’ US.

**Qt winmerge 3.0 application  See Also:**

- https://bitbucket.org/grimmdp/winmerge
- https://bitbucket.org/grimmdp/winmerge/wiki/Home
- https://bitbucket.org/grimmdp/winmerge/wiki/Installer

WinMerge 3 will be modern compare/synchronization tool. It will be based on Qt library and cross-platform. You can use the same tool in Windows and in Linux.

WinMerge will not be just WinMerge 2 with new GUI. There will be radical changes in how WinMerge works and how it looks. Many features won’t be available anymore and new features will be added.
Source

hg clone https://bitbucket.org/grimmdp/winmerge

pyQt scapy GUI IPv6    See Also:

- https://bitbucket.org/grimmdp/winmerge
- https://bitbucket.org/grimmdp/winmerge/wiki/Home
- https://bitbucket.org/grimmdp/winmerge/wiki/Installer

The project builds a GUI for the Python network tool Scapy. It serves as a learning tool for packet generation with python and as a tool for rapid IPv6 packet generation.

- Installation Describes how to install and run the program
- Screenshots Give a quick overview and explanation of the current program features

Source


Other GUI programs

Sujet: Re: [scapy.ml] GUI for IPv6 packet building
Date : Thu, 31 Mar 2011 21:08:14 +0200
De : Francesco Piccinno <stack.box@gmail.com>
Répondre à : scapy.ml@secdev.org
Pour : scapy.ml@secdev.org
Copie à : scheffler@cs.uni-potsdam.de

If you want to use something ready as a starting point I suggest you to look at my software which aims to be a backend independent packet manipulator tool. Scapy is of course supported. You can download it from:
http://manipulator.umitproject.org

Actually if some student is interested to improve it UMIT is also taking part to GSoC 2011 program.

Some useful links:

- http://www.umitproject.org/
- http://www.umitproject.org/ideas

Scribus Qt application    See Also:

- http://wiki.scribus.net/canvas/1.4.0_Release
The sigrok project aims at creating a portable, cross-platform, Free/Libre/Open-Source logic analyzer software that supports various (usually USB-based) logic analyzer hardware products.

The code is licensed under the terms of the GNU GPL.

On debian  See Also:

• http://packages.qa.debian.org/s/sigrok.html

Installing sigrok

Qt extensions

qxt Qt extension

libqxt  See Also:

• http://dev.libqxt.org/libqxt/wiki/Home
• http://libqxt.bitbucket.org/doc/

LibQxt is an extension library for Qt providing a suite of cross-platform utility classes to add functionality not readily available in the Qt toolkit by Trolltech, a Nokia company.

This is the home for our source repository on bitbucket, reachable by http://dev.libqxt.org/libqxt or http://bitbucket.org/libqxt/libqxt.

Our main website is hosted at http://libqxt.org. You can find the documentation at http://doc.libqxt.org/

Dialog generator for Qt

Dialog generator for Qt  See Also:

http://www.holgerschurig.de/projects/mvg.html#dialog_code_generation

Qt’s Model/View Programming might be extremely flexibly.

This flexibility comes at the price of tedious programming, even for simple case.
What could be simple than having a struct filed with data, an array of such struct (either C++ array, or QList, or QVector, whatever), and a widget where you can see all those items and edit them.

A simple task, needed in many programs, no ?

Yet is is quite tedious to implement this in Qt 4.x:

- You have to hand-code an editor (Dialog) for it.
- You have to hand-code a model for it.
- You have to hand-code a view for it, or re-use QTableView.

mvg.py is a Model-View Generator and is a tool to solve this.

Qt GUI HTML5  See Also:


Dans quelques situations, Qt Quick se complète avec HTML 5. Utiliser HTML 5 signifie que l’on peut obtenir du code réutilisable si l’on possède une interface utilisateur que l’on souhaite déployer à la fois sur des moteurs Web standard et en tant qu’application personnalisée.

En utilisant le module QtWebKit, il est entièrement possible de réaliser une interface utilisateur autour d’une page Web, c’est-à-dire du code HTML 5. L’intégration de QtWebKit n’inclut pas seulement le rendu HTML et un moteur JavaScript, elle met également en place un pont permettant l’intégration de multiples QObject à l’intérieur de JavaScript.


Il est bon de noter que l’on peut accéder à une frame (ou plusieurs) d’une page à travers la mainFrame:

```cpp
QObject *obj = new MyLogic();
QWebFrame *frame = webpage->mainFrame();
frame->addToJavaScriptWindowObject("logic", obj);
```

Il n’existe aucun moyen d’exposer les objets contextuels de la même manière que pour l’intégration C++ et QML. Au lieu de cela, chaque objet doit être ajouté en tant qu’objet séparé au contexte.

Il est possible d’accéder et de modifier les propriétés de QObject depuis le JavaScript. On peut également appeler des méthodes depuis le C++ vers le JavaScript et vice-versa, tout comme émettre des signaux et réaliser des connexions depuis les deux camps.

Qt news

QML (Qt Meta-Object Language)  See Also:

- http://qt.gitorious.org/qt-components/desktop
- javascript language

2.13. GUI (Graphical User Interface)
QML (Qt Meta-Object Language) is a JavaScript-based, declarative language for designing user interface centric applications.

It is part of the Nokia Qt framework. QML is mainly used for mobile applications where touch input, fluid animations (60 FPS) and user experience are crucial.

QML documents describe an object tree of elements. QML elements[2] shipped with Qt are a sophisticated set of building blocks, graphical (e.g., rectangle, image) and behavioral (e.g., state, transition, animation).

These elements can be combined to build components ranging in complexity from simple buttons and sliders, to complete internet-enabled programs.

QML elements can be augmented by standard JavaScript both inline and via included .js files.

Elements can also be seamlessly integrated and extended by C++ components using the Qt framework.

The language name is QML. The runtime name is Qt Declarative.

QML components for desktop  See Also:
http://labs.qt.nokia.com/2011/03/10/qml-components-for-desktop/

While waiting for things to calm down a bit around here, I have been happily hacking on a new research project.

Since most of you have probably guessed what it is about by now, I will start by pointing you to this video:

Quick has been the main focus of Nokia for some time now. I wanted to see if we could bring some of the fun and joy of working with QML to the good old desktop. It is true that you can easily make your own widgets with Qt Quick, but obviously not everyone wants to do that. Having personally invested quite a bit of time in developing among others the GTK+ and Vista styles in Qt, I also wanted to prove that we could make use of that effort even in the brave new world of Qt Quick.

See Also:

Préambule  Je pense que la plupart d’entre vous ont entendu parler de QML ou de declarative UI, mais sans vraiment trop savoir de quoi ça parle.

Je vais essayer de résumer.

L’architecture “widget” est une architecture robuste et structurée qui a fait ses preuves. Mais cette architecture est peu flexible, peu adaptée aux composants non rectangulaires et aux animations. Qui fournit donc des IHM très statiques. Seulement, le besoin change. Et à quoi devrait ressembler une IHM dans le futur pour vous ? Sûrement à des IHM vivantes avec une plus grande interaction avec l’utilisateur avec des petits effets visuels. Le meilleur exemple son les téléphones mobiles et leurs interfaces de plus en plus attractives.

Il suffit de regarder l’interface de l’iPhone, Android, HTC et compagnie. Rien à voir avec les logiciels d’aujourd’hui. Tout est en mouvement : on zoom, on fait des rotations... On exploite le stylet, le doigt, la luminosité ambiante, l’orientation de l’appareil... L’application de visualisation de photo de l’iPhone est un très bon exemple. Et bien sûr, la mode commence à s’étendre sur les PC.

En gros voilà declarative UI est un projet R&D de Nokia sur le développement de ces IHM, et QML un langage pour exprimer de manière lisible (par un humain) ces IHM.

Voici deux vidéos très intéressantes :

- La deuxième sur les futures interfaces sur mobile (mais ça reste valide pour un PC) : New user interface paradigms on mobile devices <http://blip.tv/file/2561463/>
Pour être plus clair, voici quelques vidéos en plus.

Deux démonstrations utilisant QGraphics et QtAnimation. Et donc ce qui sera déjà possible avec Qt 4.6 :

- YouTube- Qt Kinetic Animated Tiles Example <http://www.youtube.com/watch?v=NEVrwYAAUvU&feature=channel_page>
- YouTube- TubeWiz: Unleashing the power of Qt for S60 <http://www.youtube.com/watch?v=NjxfEud_B6U&feature=channel_page>

Deux démonstrations utilisant QML et du javascript montrant ses premières possibilités:

- YouTube- QML Same Game Demo <http://www.youtube.com/watch?v=8Bvm4E819UY>
- YouTube- QML flickr browser demo <http://www.youtube.com/watch?v=xoo_Ows1ExU>

De plus, Developpez.com met des binaires précompilés de Qt à votre disposition. Dans lesquel vous trouverez une version compilée pour visual 2008 SP1 de la branche /kinetic-declarative-ui/ et donc voir à quoi cela va ressembler et bien sûr de jouer avec

Des binaires Qt à disposition ! <http://www.developpez.net/forums/m4657309-10/>

Et vous que pensez-vous de tout cela ? Vous êtes pour ? Contre ? Des remarques positives ou négatives ?

**Introduction**

Une interface utilisateur Qt Quick est construite depuis QML. Tout le code QML est exécuté dans un contexte. Chaque contexte est représenté par un objet QDeclarativeContext.

Pour QDeclarativeEngine, il y a un contexte racine qui contient les objets et valeurs globalement disponibles.

Pour exposer un QObject en tant que pièce de code QML, l’objet est défini en tant que valeur d’une propriété contextuelle, comme montré dans le code suivant:

```cpp
QObject *obj = new MyLogic();
QDeclarativeEngine *engine = ...;
engine->rootContext()->setContextProperty("logic", obj);
```

Cela expose l’objet au code QML dans lequel on peut y accéder en utilisant le nom logic:

```cpp
MouseArea {
...
    onClicked: { logic.actOnClick(); }  
}
```

La bonne chose dans l’exposition de propriétés contextuelles à QML est qu’elles s’associent bien avec la nature de binding de QML. Cela signifie que, si l’on définit la propriété contextuelle sceneWidth à 400 et qu’on la modifie par la suite à 500 (ou toute autre valeur), toutes les expressions dans QML dépendant de sceneWidth seront automatiquement actualisées.

**Qt quick applications**

See Also:

- [https://projects.forum.nokia.com/qmultiwinexample](https://projects.forum.nokia.com/qmultiwinexample)
- [https://projects.forum.nokia.com/qtbubblelevel](https://projects.forum.nokia.com/qtbubblelevel)

**Qt defis**

See Also:

- [http://qt.developpez.com/defis/02-tablette-hopital/resultats](http://qt.developpez.com/defis/02-tablette-hopital/resultats)
Capteurs et QML   See Also:


En QML, il y a un élément pour chaque type de capteur. Pour les accéléromètres, il s’agit d’Accelerometer. Les autres éléments sont nommés d’une manière très similaire aux classes C++, comme RotationSensor et ProximitySensor:

```qml
Accelerometer {
    id: sensor

    onReadingChanged: {
        xText.text = "X: " + reading.x.toFixed(2)
        yText.text = "Y: " + reading.y.toFixed(2)
        zText.text = "Z: " + reading.z.toFixed(2)

        var subFromAngle = screen.width > screen.height ? Math.PI : Math.PI/2
        var angle = Math.atan2(reading.y, -reading.x)
        theColumn.rotation = (angle-subFromAngle) * (180/Math.PI)
        ...
    }
}
```

Pour lire les données, on utilise le gestionnaire de signal onReadingChanged, appelé quand QSensor émet QSensor::readingChanged(). Les données sont disponibles dans la propriété Accelerometer::reading. theColumn est une Column qui contient trois éléments Text, qui affichent les données de l’accéléromètre.

QML and Qt Quick   See Also:

http://labs.qt.nokia.com/2011/02/18/buckets-of-cold-water/

L.MCH February 19, 2011 at 3:57 am   @David & Will Stokes

**QML & Qt Quick are a lot more relevant for desktop applications than it seems.** Once you start mastering Qt Quick and getting the mindset behind its design, it simplifies app development not just for smartphones, but on desktops an embedded devices too.

QML has very basic components but they are a lot more flexible than QWidgets coded in C++.

For example: a QML Listview is not “something like a listview”, instead is “something like a listview and much more” that you can use as a listview, but also as a multipage panel, or turn it into a “versatile treeview” by using the proper models and delegates.

QML UI items look more spartan than QWidgets, but they are easier and faster to compose into new UI elements you can reuse and extend further.

I’m using Qt to develop “simple” SCADA/DCS applications running on windows desktops and windows ce devices and it simply rocks *(really simple and basic, not the really big SCADA stuff, my applications are more like distributed HMIs on steroids).*

Low-level I/O and synchronization on serial links, tcp/ip and CAN bus is handled by a reusable “SCADA engine framework” coded in Qt/C++ and using C++ coded QWidgets for the fixed parts of the UI, while all the higher level and customized interactive visualization and instrumentation is handled by QML.

In other words it makes a lot easier to develop a “base application” written in Qt/C++ and adapt and extend it by scripting in QML/Javascript parts of its user interface and most of its higher level logic.

The best part is that I’m the **C++ guy that handles the weird and complex stuff** while the (boring) customized SCADA/DCS/HMI UIs for the final customers are coded in QML by the same guys that usually develop the plc.
programs on the remote devices, this way THEY write the UI as they like it (and of course they test it too) and they really like it too, because they feel they have full control of the system and development cycle is faster.

Of course some things need to be improved and there are problems (i.e. using Qt 4.7.1 there are some memory leaks that are really troublesome on memory-constrained Windows CE devices, but most of them are already fixed in the git repository so I just had to download a commit with the relevant patches) but it is a lot better than coding everything in C++ or using Python/"pure javascript"/Lua/ecc. for the scriptable parts (because in my humble opinion QML integrates a lot better the lower level Qt/C++ parts with the higher level UI coding and application logic).

Mine may seem a very peculiar situation, but almost every application that grows beyond a certain size/complexity ends up needing what Qt Quick provides (i.e. high level scripting and “flexible UI modding”).

Curt February 19, 2011 at 5:09 pm  @vohi

Thanks to you and the whole Troll team for great software and documentation over the years and for strong, helpful communication right now. I agree with the statement that QML will improve the desktop. In fact, we are currently involved in a project utilizing Qt Quick widgets as part of a desktop application and it is working well for us. We will keep working in Qt for the coming years.

Thanks again for addressing questions from the community, the Qt “love” team, as one of the earlier posts mentioned. Take care.

Qt GUI widgets

Qt Qspinbox  See Also:

• c GUI numericupdown

Detailed Description  The QAbstractSpinBox class provides a spinbox and a line edit to display values.

The class is designed as a common super class for widgets like QSpinBox, QDoubleSpinBox and QDateTimeEdit. Here are the main properties of the class:

• text: The text that is displayed in the QAbstractSpinBox.
• alignment: The alignment of the text in the QAbstractSpinBox.
• wrapping: Whether the QAbstractSpinBox wraps from the minimum value to the maximum value and vica versa.

Qt Qwt widgets  See Also:

http://sourceforge.net/projects/qwt/

Qwt is a graphics extension to the Qt GUI application framework from Trolltech AS of Norway. It provides a 2D plotting widget and more.

Qt ui files

uic.exe generator

c:/QtSDK/Desktop/Qt/4.7.2/mingw/bin/uic.exe src/mainwindow.ui -o src/ui_mainwindow.h

2.13. GUI (Graphical User Interface) 375
pyside, pyqt, pythonqt

pyside  See Also:
  • pyqt

pyside-examples

git clone git://gitorious.org/pyside/pyside-examples.git

Sites
  • http://pyqt.developpez.com/
  • http://www.qtrac.eu/pyqtbook.html#pyside

See Also:
  watchvideo

Compiling pyside

Hi,

2011/3/10 Mark Summerfield <list@qtrac.plus.com>:
> Hi,
> > When I build PySide on debian testing it finds the system’s Qt 4.6.3.
> > However, I’d like it to use my locally built $HOME/opt/qt470, but I
> > don’t know how to tell cmake to do that. Can anyone tell me?
> >

You can try to set the cmake parameter -DQT_QMAKE_EXECUTABLE=<path to qmake> for example -DQT_QMAKE_EXECUTABLE=$HOME/opt/qt470/bin/qmake

Regards

GUI compiling with pyside-uic  See Also:

http://www.qtrac.eu/pyqtbook.html#pyside

Hi,

2011/5/15 Thorsten Kampe <thorsten@thorstenkampe.de>: > I’ve ported a simple PyQt4 (4.8.4) application to PySide 1.0.0 according > to http://www.qtrac.eu/pyqtbook.html#pyside. This works pretty well, > except that I cannot use the PyQt4 UI code generator generated UI file:

You should use the utility “pyside-uic” instead of “pyuic4” if you want to use PySide. It’s basically the same utility, but for PySide.

pyside news

pyside PySide nightly build packages, 30 mars 2011 15:58
Hello PySide users,

To improve your experience and feedback, we are making available the PySide nightly build packages[1]. These packages can be used to try what are coming in the next PySide releases, and get some bug fixes before the release. But we do not recommend use these packages on production environment, use these only for test and development. The files are being updated with the latest git version every night at 1:00AM GMT-3.

BR


pyqt  See Also:

  • [http://fr.wikipedia.org/wiki/PyQt](http://fr.wikipedia.org/wiki/PyQt)
  • [pyside](http://fr.wikipedia.org/wiki/PyQt)

PyQt est un module libre qui permet de lier le langage Python avec la bibliothèque Qt distribué sous deux licences : une commerciale et la GNU GPL.

Il permet ainsi de créer des interfaces graphiques en Python.

Une extension de QtDesigner (utilitaire graphique de création d’interfaces Qt) permet de générer le code Python d’interfaces graphiques.

Le 18 août 2009, après des pourparlers avec les auteurs de PyQt pour un changement de licence, Nokia distribue sa propre bibliothèque de lien entre Python et Qt : PySide, sous une licence plus permissive, la licence publique générale limitée GNU (LGPL).

pyqt examples  See Also:


pythonqt  See Also:


Contents

  • pythonqt
    – Introduction

Introduction  PythonQt is a dynamic Python binding for the Qt framework. It offers an easy way to embed the Python scripting language into your C++ Qt applications.

It makes heavy use of the QMetaObject system and thus requires Qt 4.x.

The focus of PythonQt is on embedding Python into an existing C++ application, not on writing the whole application completely in Python.

If you want to write your whole application in Python, you should use PyQt or PySide instead.
If you are looking for a simple way to embed Python objects into your C++/Qt Application and to script parts of your application via Python, PythonQt is the way to go!

**icons**

**Software-Development-Icons**

- [http://www.iconarchive.com](http://www.iconarchive.com)

**C GUI**

See Also:
- [http://www.dotnetperls.com/windows](http://www.dotnetperls.com/windows)

**c GUI MessageBox**

See Also:

Affiche un message pouvant contenir du texte, des boutons et des symboles donnant des informations et des instructions diverses à l’utilisateur

```csharp
private void btn_selection_ok_Click(object sender, EventArgs e)
{
    int nb_selected_puits = getNbSelectedPuits();

    if (nb_selected_puits > 0)
        tabControl_VideoCell.SelectedTab = tabControl_VideoCell.TabPages[1];
    else
    {
        const string message = "Vous n’avez pas sélectionné de puits";
        const string caption = "Erreur de sélection";
        MessageBox.Show(message, caption, MessageBoxButtons.OK);
    }
}
```

**c GUI numericupdown**

See Also:
- [http://www.dotnetperls.com/numericupdown](http://www.dotnetperls.com/numericupdown)

**Qt equivalent widget** See Also:

*Qt Qspinbox*
c GUI tab control

See Also:

- http://www.dotnetperls.com/numericupdown

Un TabControl contient des pages d’onglets qui sont représentées par des objets TabPage que vous ajoutez via la propriété TabPages.

L’ordre des pages d’onglets dans cette collection correspond à l’ordre des onglets dans le contrôle.

L’utilisateur peut modifier le TabPage actuel en cliquant sur l’un des onglets du contrôle.

Vous pouvez également modifier par programme le TabPage actuel en utilisant l’une des propriétés TabControl suivantes :

- SelectedIndex
- SelectedTab

Dans Microsoft .NET Framework version 2.0, vous pouvez également utiliser l’une des méthodes suivantes :

- SelectTab
- DeselectTab

```csharp
private void btn_selection_ok_Click(object sender, EventArgs e)
{
   .tabControl_VideoCell.SelectedTab = tabControl_VideoCell.TabPages[1];
}
```

c GUI textbox

**c GUI maskedtextbox**

See Also:

http://www.dotnetperls.com/maskedtextbox

When creating Windows Forms programs that accept input, you often need to ensure that the input is of a specific format. For example, you need phone numbers to have their area code and also the correct number of digits.

To solve this problem, you can use the MaskedTextBox control, as described in this article.

Get started (MaskInputRejected)

To get started, please add a MaskedTextBox control to your Windows Forms program by double-clicking on the icon in the toolbox.

Next, try changing the properties of the MaskedTextBox by right-clicking on it and selecting properties.

**Importantly, change the Mask property:** you can find predefined masks for:

- integer numbers,
- phone numbers,
- dates in various formats,
- Social Security Numbers,
- time in various formats,
- and zip codes.
GUI evolutions

GUI evolution

GUI evolution 2012

Modern Cross Platform Development  See Also:

http://www.dodgycoder.net/2012/01/modern-cross-platform-development.html

Why isn’t there a modern technology available for using the same codebase to produce native apps on all of the currently popular platforms -

I’m talking iOS (iPhone/iPad/iPod Touch), Android, Windows, Mac and Linux?

That was my original question before I started looking, and since then I’ve discovered there actually are plenty of new options out there for cross platform development catering for all of the above platforms.

The main advantage of the browser app however, and its a massive one, is that its everwhere and supported by every device, so if HTML5 takes off, then it could become the defacto standard for developing apps of any kind, desktop or not.

As Jeff Atwood discusses in his seminal blog post All Programming is Web Programming: “the web is the most efficient, most pervasive, most immediate distribution network for software ever created - its almost completely frictionless”.

As Atwood mentions, more and more users gravitate towards preferring to run their apps inside the browser, on smartphones and on tablets where things ‘just work’ and there’s generally nowhere near as much friction caused by the traditional desktop software setup and update process.

A third option available now is the cross platform mobile development SDK as detailed below (Mono, Appcelerator Titanium, Rhodes, PhoneGap, MoSync and Moai).

Smartphone and tablet based applications are taking off, and these SDKs allow ou to write once and run on any of these new class of devices, as well as in some cases also being able to run on the traditional desktop.

These would seem to offer some of the ‘power’ of the desktop in terms of native code execution speed and access to hardware functionality, along with the enourmous ‘frictionless’ benefit that you get when publishing to an app store, where installing an app is literally a one click operation.

Aug 14, 2009 All Programming is Web Programming  See Also:

http://www.codinghorror.com/blog/2009/08/all-programming-is-web-programming.html

2.14 IDEs

2.14.1 Integrated Development Environments

See Also:

ACE Integrated Development Environment

See Also:

- https://github.com/ajaxorg/ace

Ace is a standalone code editor written in JavaScript. Our goal is to create a browser based editor that matches and extends the features, usability and performance of existing native editors such as TextMate, Vim or Eclipse.

It can be easily embedded in any web page or JavaScript application. Ace is developed as the primary editor for Cloud9 IDE and the successor of the Mozilla Skywriter (Bespin) Project.

Ace based editors

Plone editor See Also:

http://code.google.com/p/ploneide/

PloneIDE is an IDE intended to make Plone development faster and easier. It runs on top of Plone as a third party product, so it runs inside the browser. For this, it uses Ace, the editor used in the Ajax.org’s Cloud9 IDE.

I bet i lost some of you when i mentioned the fact that you will be coding inside the browser, but give it a try. It's amazingly fast, so you'll forget you're inside a browser.

Borland Integrated Development Environment

Borland

Borland update firmware

README

===============
README.txt file
===============

The ‘firmware.rc’ file is used by 2 GUI borland projects:

- XLOG8X235_UpdateReaderFirmwareLight
- XLOG8X236_UpdateAllReadersFirmwareLight

This file contains the last firmwares for the CL1356T & T5 readers.

Update rules

===============

When a new firmware is available, follow these instructions:

- update the Firmware.rc file
- compile the file in the Borland IDE

update the Firmware.rc file
1 RCDATA ".\XLOG8X238_Firmware_Hex_BIN\ReaderT_Standard\XLOG6N093_VE1.23.hex"
2 RCDATA ".\XLOG8X238_Firmware_Hex_BIN\RFModuleT_Standard\XLOG5H054_v255.bin"
3 RCDATA ".\XLOG8X238_Firmware_Hex_BIN\ReaderT_5\XLOG6S108_VN1.23.hex"
4 RCDATA ".\XLOG8X238_Firmware_Hex_BIN\RFModuleT_5\XLOG6R539_v032.bin"
2.14. IDEs

Development tools, Release 2012.06.18
Eclipse Integrated Development Environment

See Also:


Eclipse is a multi-language software development environment comprising an integrated development environment (IDE) and an extensible plug-in system.

It is written mostly in Java and can be used to develop applications in Java and, by means of various plug-ins, other programming languages including Ada, C, C++, COBOL, Perl, PHP, Python, Ruby (including Ruby on Rails framework), Scala, Clojure, and Scheme.

The IDE is often called Eclipse ADT for Ada, Eclipse CDT for C/C++, Eclipse JDT for Java, and Eclipse PDT for PHP.

Eclipse plugins

**pydev plugin**    See Also:

- http://pydev.org/index.html

What is PyDev?

PyDev is a Python IDE for Eclipse, which may be used in Python, Jython and IronPython development.

It comes with many goodies such as:

- Django integration
- Code completion
- Code completion with auto import
- Syntax highlighting
- Code analysis
- Go to definition
- Refactoring
- Mark occurrences
- Debugger
- Remote debugger
- Tokens browser
- Interactive console
- Unittest integration

For more details on the provided features, check the Features Matrix.

**Important**    First time users are strongly advised to read the Getting started guide which explains how to properly configure PyDev
Editra

See Also:

- http://editra.org/

Editra is a multi-platform text editor with an implementation that focuses on creating an easy to use interface and features that aid in code development. Currently it supports syntax highlighting and variety of other useful features for over 60 programming languages.

Editra is freely available for personal use under the terms of the wxWindows License.

Currently the project is in the alpha development phase but test builds of “stable” points are available for download and trial as Windows and Mac OSX(Universal) binaries, currently other Unix and Linux based systems will have to install from source using the included setup script. Please feel free give it a try and to report bugs and request features.

Eric

See Also:


Eric4

See Also:


Getting the eric4 code  Eric4 uses Subversion (svn) for managing its code.

Assuming you have Subversion installed, the following command in a terminal will fetch the most recent code for you:

```bash
tool co http://die-offenbachs.homelinux.org/svn/eric4repository/eric4/trunk/
```

After you got the eric4 code you have to run the `compileUiFiles.py` script in order to convert the forms files (*.ui) to valid Python code.

Stable and snapshot releases of eric4 are available at the eric download page

Eric5

See Also:

Getting the Eric5 code  Eric5 uses Mercurial (hg) for managing its code.
Assuming you have Mercurial installed, the following command in a terminal will fetch the most recent code for you:

```
hg clone http://die-offenbachs.homelinux.org/hg/eric5
```

After you got the eric5 code you have to run the `compileUiFiles.py` script in order to convert the forms files (*.ui) to valid Python code.

Stable and snapshot releases of eric5 are available at the eric download page:

```
http://eric5.plugins.die-offenbachs.homelinux.org
```

In order to get the code of the various plugins, enter the following command in a terminal:

```
hg clone http://die-offenbachs.homelinux.org/hg/eric5plugins/<path of the plugin>
```

To see the available plugin paths please follow the link below to the eric5 repositories.
Stable and snapshot releases of the eric5 plugins are available at the eric download page:

Geany editor

Geany is a lightweight cross-platform GTK+ text editor[2] based on Scintilla and including basic Integrated Development Environment (IDE) features.

It is designed to have short load times, with limited dependency on separate packages or external libraries. It is available for a wide range of operating systems, such as BSD, Linux, Mac OS X[3], Solaris and Windows.

Among the supported programming languages are C, Java, JavaScript, PHP, HTML, CSS, Python, Perl, Ruby, Pascal and Haskell and reStructuredText !

It is free software licensed under the terms of the GNU GPL version 2 or later

See Also:

- http://www.geany.org/Main/HomePage

Documentation


Ubuntu

Geany is available through the official Ubuntu archives (Universe section):

```
aptitude install geany
```

You might find newer versions in the Ubuntu Geany PPA at:

https://launchpad.net/~geany-dev/+archive/ppa
Geany Portable for Windows

See Also:
- http://www.geanyportable.org
- http://geanyportable.org/blog/
- https://twitter.com/#!/geanyportable

There is a portable version of Geany by Oliver Krystal to carry on your USB stick and use on any Windows computer GeanyPortable.org, in partnership with PortableApps.com and The Geany Team is pleased to announce the first official Geany Portable release for GeanyPortable.org and PortableApps.com

plugins

See Also:

Netbeans Integrated Development Environment

See Also:

The netbeans IDE uses:
- the GNU/MinGW development tools
- the cygwin command utilities like ‘rm’

NetBeans refers to both a platform framework for Java desktop applications, and an integrated development environment (IDE) for developing with Java, JavaScript, PHP, Python, Groovy, C, C++, Scala, Clojure, and others.

The NetBeans IDE 7.0 no longer supports Ruby and Ruby on Rails.

Netbeans

The C Netbeans IDE  NetBeans refers to both a platform for the development of applications for the network (using Java, JavaScript, PHP, Python, Ruby, Groovy, C, and C++), and an integrated development environment (IDE) developed using the NetBeans Platform. The NetBeans IDE is an open-source integrated development environment written entirely in Java using the NetBeans Platform. NetBeans IDE supports development of all Java application types (Java SE, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project system, version control (supporting CVS, Subversion, Mercurial and Clearcase) and refactoring.

See Also:
- http://code.google.com/p/winant/
License  From July 2006 through 2007, NetBeans IDE was licensed under Sun’s Common Development and Distribution License (CDDL), a license based on the Mozilla Public License (MPL).

In October 2007, Sun announced that NetBeans would henceforth be offered under a dual license of the CDDL and the GPL version 2 licenses, with the GPL linking exception for GNU Classpath.

Netbeans tool collection  The free GCC tools use are the Minimalist GNU for Windows (MinGW).

Netbeans windows version resource  See Also:

MinGW Resource Compiler (windres)

To add the version of the id3Image.dll file we must:

- create a resource file: ‘windows_resource/version.rc’

```
1 VERSIONINFO
FILEVERSION 0,1,9,0
PRODUCTVERSION 0,1,9,0
FILEFLAGSMASK 0x17L
FILEOS 0x4L
FILETYPE 0x2L
FILESUBTYPE 0x0L
BEGIN
  BLOCK "StringFileInfo"
  BEGIN
    BLOCK "040c04b0"
    BEGIN
      VALUE "CompanyName", "id3 Semiconductors"
      VALUE "FileDescription", "Bibliotheque id3Image.dll"
      VALUE "FileVersion", "0, 1, 9, 0"
      VALUE "InternalName", "id3Image"
      VALUE "LegalCopyright", "Copyright (C) 2009-2010"
      VALUE "OriginalFilename", "id3Image.dll"
      VALUE "ProductName", "id3Image.dll"
      VALUE "ProductVersion", "0, 1, 9, 0"
    END
  END
  BLOCK "VarFileInfo"
  BEGIN
    VALUE "Translation", 0x40c, 1200
  END
END
```

- compile the version.rc file with the MinGW Resource Compiler (windres)

```
#!/bin/sh
windres -o version.o version.rc
```

- link against the id3Image.dll

```
-Wl,--output-def,$(CND_DISTDIR)/$(CND_CONF)/id3Image.def, --out-implib,$(CND_DISTDIR)/$(CND_CONF)/id3Image.a version.o
```

The id3Image.dll with the version information
NetBeans IDE Bundle for C  

The NetBeans IDE Bundle for C adds support for C developers since NetBeans IDE 5.5.

This pack lets C developers use their specified set of compilers and tools in conjunction with NetBeans IDE to build native applications for Windows, Mac OS X, Linux, and Solaris.

The NetBeans C bundle does not include a C compiler.

The free GNU C Compiler used is the GNU Compiler Collection (GCC).

### Netbeans problems

**Warning:** Netbeans automatically generates the following paths:

- OBJECTDIR=build/${CND_CONF}/${CND_PLATFORM}
- ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/dir/obj.o

These paths are tooooo long and the linker buffer cant link all the files.

The solution is to update the netbeans makefiles (Makefile-Debug.mk, Makefile-Release.mk) in order to use the following paths:

- OBJECTDIR=build/${CND_CONF}/
- ${OBJECTDIR}/dir/obj.o

**clean_up.sh**  

This Netbeans problems can be resolved with this script:

```bash
#!/usr/bin/sh
perl -pi -e 's//_ext/_DOTDOT/_DOTDOT//g' $1
perl -pi -e 's//${CND_PLATFORM}//g' $1
```

The script can be run with the following command:

```
pvergain@euphorbe /cygdrive/y/projects_c/XLOG9Y315_LIB_id3Image/binaries/id3Image/nbproject $ ./clean_up.sh Makefile-Debug.mk
```

### Netbeans Makefile-{Debug,Release}.mk

**Warning:** if you add or remove a new ”.c” file or even a ”.h” file, you have to update the makefiles instead of:

```bash
# Object Directory
OBJECTDIR=build/${CND_CONF}/${CND_PLATFORM}

# Object Files
OBJECTFILES=  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/mindtct/results.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpegb/wrrle.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpegb/jdtrans.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/libjasper/base/jas_init.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpeg/jpegl/jmemansi.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/mindtct/getmin.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpeg/jpegl/util.o  
```

you should have:

```bash
# Object Directory
OBJECTDIR=build/${CND_CONF}/${CND_PLATFORM}

# Object Files
OBJECTFILES=  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/mindtct/results.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpegb/wrrle.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpegb/jdtrans.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/libjasper/base/jas_init.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpeg/jpegl/jmemansi.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/mindtct/getmin.o  
   ${OBJECTDIR}/_ext/_DOTDOT/_DOTDOT/jpeg/jpegl/util.o  
```
The Netbeans properties

Preprocessor definitions  In the nbproject/configurations.xml file

```
<preprocessorList>
  <Elem>WINDOWS</Elem>
  <Elem>__MSYS__</Elem>
  <Elem>__NBISLE__</Elem>
  <Elem>__NBIS_JASPER__</Elem>
  <Elem>__NBIS_PNG__</Elem>
</preprocessorList>
```

Netbeans versions

Netbeans 7.2  See Also:
- http://netbeans.org/community/releases/72/relnotes.html
- http://wiki.netbeans.org/NewAndNoteworthyNB72
**Contents**

- Netbeans 7.2
  - What’s New in 7.2 beta

**What’s New in 7.2 beta** NetBeans IDE 7.2 Beta is an update to NetBeans IDE 7.1.2 and includes the following changes:

- Significant performance increase on remote filesystems, improved project scanning speed, and background project scanning
- Java Editor enhancements
- JavaSE Development Kit 7 Update 4 support
- Full support of JavaFX 2.1 SDK on Windows and OS X 10.7
- Redesigned search history in supported version control systems and support for Mercurial Queues
- Ant 1.8.3 is supported and bundled with the IDE
- Bundled version of Maven is upgraded to 3.0.4
- Debugger improvements
- Oracle Public Cloud support
- PHP 5.4 support
- Debugging and code assistance improvements in C++ projects and C++11 support
- Additional enhancements are listed on the NetBeans IDE 7.2 New and Noteworthy page.

For more about this release, see the NetBeans IDE 7.2 Beta Release Information page.

**Netbeans 7.1** See Also:

- [http://netbeans.org/community/releases/71/index.html](http://netbeans.org/community/releases/71/index.html)

**Contents**

- Netbeans 7.1
  - What’s New in 7.1
  - What’s New in 7.1.2

**What’s New in 7.1** NetBeans IDE 7.1 is a significant update to NetBeans IDE 7.0.1 and includes the following changes:

- JavaFX 2.0.2 SDK support
- Batch refactoring tool for bulk changes in Java Editor
- New UI Debugger for JavaFX and Swing
- Support for GlassFish 3.1.1 and WebLogic 12c
- Improved support for CDI, EJB, JSF, Spring, Web Services
- Subversion and Mercurial enhancements and bundled Git support
• Maven enhancements
• PHP enhancements

What’s New in 7.1.2  NetBeans IDE 7.1.2 is an update to NetBeans IDE 7.1 and NetBeans IDE 7.1.1, and contains the following highlights:

• Java SE 7u4 Support, which includes the first Oracle JDK release for Mac OS X
• JavaFX 2.1 Support (Bundled with the JDK)
• Ant upgrade to version 1.8.3
• Support for GlassFish 3.1.2
• Integration of recent patches
• Minor performance improvements

IntelliJ IDEA

See Also:
• http://sonatype.com/Services/Training/Maven-Fundamentals

IntelliJ IDEA is a code-centric IDE focused on developer productivity. The editor deeply understands your code and knows its way around the codebase, makes great suggestions right when you need them, and is always ready to help you shape your code.

Intelligent code editor understanding your code, with refactorings, code inspections, intention actions, easy navigation and full Java 7 support.

JUnit and TestNG integration and a convenient test runner UI

Maven and Ant-based project setup and building. Comprehensive Groovy programming language support plus plugins for Scala and Clojure.

Google Android development including latest SDK support. Unified UI and experience for most popular version control systems:

• Subversion,
• Git,
• Mercurial
• CVS.

Visual merge, diff and changelists.

Powerfull XML editor with XML-Java interoperability.

The Swing UI designer for developing Java desktop applications

monodevelop

See Also:
• http://monodevelop.com/
MonoDevelop is an open-source integrated development environment for the Linux platform, Mac OS X, and Microsoft Windows, primarily targeted for the development of software that uses both the Mono and Microsoft .NET frameworks.

MonoDevelop integrates features similar to those of NetBeans and Microsoft Visual Studio, such as automatic code completion, source control, a graphical user interface (GUI) and Web designer.

MonoDevelop integrates a Gtk# GUI designer called Stetic.

It currently has language support for C#, Java, Boo, Visual Basic.NET, Oxygene, CIL, Python, Vala, C and C++.

**monodevelop versions**

**monodevelop 3.0.0 (May 14, 2012)**

See Also:

- http://monodevelop.com/Download/What%27s_new_in_MonoDevelop_3.0

**New Features and Improvements**

This release contains lots of new features and improvements. Here is a summary of the new features. You will find a more detailed description in the What’s new in MonoDevelop 3.0 page.

- New C# code completion engine:
  - More accurate and reliable code completion and navigation
  - Much faster Find References and Rename operations
  - Semantic highlighting for C# files
  - More reliable on the fly code formatter
  - Experimental source analysis and contextual fixes
- Virtual indent mode in the text editor
- New MonoMac addin
- **Visual UI Designer for Mono for Android**
  - Revamped Assembly Browser
  - Preliminary support for Portable Library Projects (PLP)
  - Faster loading and building of large projects

And plenty of bug fixes in all MonoDevelop features.

Please see What’s new in MonoDevelop 3.0 for more details.
Development tools, Release 2012.06.18

Android UI Designer  See Also:
http://docs.xamarin.com/android/tutorials/Designer_Overview
The Xamarin packages of MonoDevelop on Mac and Windows include a new visual drag-and-drop UI designer for editing Android XML layouts in Mono for Android projects.
It has full support for resources and qualifiers, and can be used to view and edit the layout in various simulated device specifications.
For more information, see the Mono for Android Designer Tutorial.

monodevelop 2.8.5 (December 13, 2011)  This release includes the following fixes and improvements:

**Workarounds for some Xcode integration issues**  Fixed the case where Xcode could spontaneously restart certain circumstances after it was exited. Added a workaround to try to avoid a deadlock in Xcode when it starts up.

**Better iPhone debugging support**  A predefined port is no longer required.

**Fixed rendering glitches on Windows**  Fixed the case where some windows would turn black the second (and subsequent) time they were viewed and would never re-render properly.

**Better support for both case-insensitive and case-sensitive file systems**  Correctly handle changing the case of a filename on a case-insensitive filesystem. MacOS is now treated as case-insensitive (the default for the filesystem).

**Several fixes and enhancements to the Version Control support and to SVN support**  In particular, adding a project which contains linked files to version control is now handled correctly.

**Several fixes to drag and drop support**  Dragging and dropping folders within MonoDevelop while under SVN version control is supported now. If an unexpected error happens during a drag and drop operation, MonoDevelop will no longer hard crash.

**Updater**  Fixed a cause of the updater crashing on startup on Windows.

Notepad++ editor

See Also:
- http://notepad-plus-plus.org/
plugins

Python Script for Notepad++  See Also:

- http://nppythonscript.sourceforge.net/
- https://github.com/davegb3/PythonScript
- Full programmatic access to Notepad++ features and menus
- Full programmatic access to all of Scintilla features
- Call other plugin menu items
- Assign menu items, shortcuts and toolbar icons to scripts
- Process Notepad++ and Scintilla events, direct from a Python script
- Python console built-in
- Full regular expression support for search and replace - script Python regular expression replaces
- Start external programs and pipe the output direct to a Notepad++ document, or filter it, or simply to the console window
- Full documentation for all the objects and methods

Sphinx documentation  See Also:

http://nppythonscript.sourceforge.net/docs/latest/

Source

Ninja Integrated Development Environment

NINJA-IDE (from: “Ninja Is Not Just Another IDE”), is a cross-platform integrated development environment specially design to build Python Applications.

NINJA-IDE provides tools to simplify the Python-software development and handles all kinds of situations thanks to its rich extensibility.

See Also:

- http://ninja-ide.appspot.com/
- http://qt.nokia.com/qt-in-use/ambassadors/project?id=a0F20000006KWGyEAO

Mercurial sources

hg clone https://ninja-ide.googlecode.com/hg/ ninja-ide

pyscripter

See Also:

- http://code.google.com/p/pyscripter/
PyScripter is a free and open-source Python Integrated Development Environment (IDE) created with the ambition to become competitive in functionality with commercial Windows-based IDEs available for other languages. Being built in a compiled language is rather snappier than some of the other Python IDEs and provides an extensive blend of features that make it a productive Python development environment.

**pydev**

**See Also:**

- [http://pydev.org/](http://pydev.org/)

![pydev logo](image)

Figure 2.16: *pydev logo*

PyDev is a Python IDE for Eclipse, which may be used in Python, Jython and IronPython development. It comes with many goodies such as:

- Django integration
- Code completion
- Code completion with auto import
- Syntax highlighting
- Code analysis
- Go to definition
- Refactoring
- Mark occurrences
- Debugger
- Remote debugger
- Tokens browser
- Interactive console
- Unittest integration
- Code coverage
- and many others:

**Versions**

**pydev versions**

**pydev 2.3.0  Release 2.3.0**

- Pep8.py integrated (must be enabled in PyDev > Editor > Code Analysis > pep8.py).
- Faster PyDev startup (internal Jython upgraded to version 2.2.1 – and also optimized for PyDev).
- Action to select/deselect scope (Shift+Alt+Up/Down).
- Fix: cache issue where the PYTHONPATH in memory became different from the PYTHONPATH configured for a project.
• Fix: OutOfMemoryError when dealing with PyOpenGL.
• Fix: deadlock (could occur in a race condition when importing a project with an existing Python configuration).
• Fix: code-completion integration issue with IPython 011 (patch from jonahkichwacoders).
• Fix: annotation could remain in editor after removing a marker.
• Fix: BadLocationException on extract local refactoring.

**pydev 2.2.4**

**Cython** Cython is now supported in PyDev (.pyx files may be opened with the PyDev editor).

**Globals Token Browser (Ctrl+Shift+T)** Packages/Modules can now be reached through the globals browser (so, __init__.py files can now be easily gotten through the package they represent)

**Handling external files** External libraries configured in a project appearing in the PyDev Package Explorer Show in
> PyDev Package Explorer working for files that are under the interpreter or external libraries. Show in > PyDev Package Explorer working for files inside .zip archives. External files that were opened when Eclipse is closed are properly reopened.

**Editor** New option in the code-formatter to only apply code-formatting on changed lines on save. from __future__ import now properly appears as first even if grouping is enabled. it’s now possible to have a minimap of the code in the overview ruler (enable in preferences > PyDev > Editor > Overview Ruler Minimap).

**Unittest runner** exc_clear() no longer called if it’s not available. Fixed issue where class tearDown was executed twice.

**Debugger** It’s now possible to enable/disable stepping into properties while in the debugger. Menu: Run > Disable step into properties (patch by Hussain Bohra) Show in outline view activated in debug perspective (patch by Hussain Bohra) Watch expressions can be properly expanded in the watch view (patch by Hussain Bohra) Breakpoints in external files are properly shown. Remote debugger: starting the remote debugger no longer shows a launch configuration Remote debugger: when the server is stopped, the server socket is properly closed

**Minors** Fixed issue in rename (Alt+Shift+R) / find references (Ctrl+Shift+G) on top level module variables. Fixed issue where doing create class/method/field action on file with tabs ended up adding spaces.

**Qt Integrated Development Environment**

**See Also:**
- [http://qt-project.org/](http://qt-project.org/)
- [Qt C++ nzmqt library](http://qt.developpez.com/faq/)
- [Qt GUI](http://qt.developpez.com/faq/)

```
echo Setting up environment for Qt usage...
set QTDIR=C:\QtSDK_1.1.4\Desktop\Qt\4.7.4\mingw
set PATH=C:\QtSDK_1.1.4\mingw\bin;%PATH%
set PATH=%QTDIR%\bin;%PATH%
```

2.14. IDEs
Qt project

See Also:

http://qt-project.org/

The Qt Project governs the open source development of Qt. It allows anybody wanting to contribute to join the effort, through a meritocratic structure of approvers and maintainers.

All development is driven by the people contributing to the project.

To learn more, visit the resources linked on this site and subscribe to our mailing-lists.

Write Qt Code  Naturally, the Qt Project is mostly about code. There are plenty of ways to contribute code, to learn and grow, and to build one’s reputation.

To get started contributing code, get in touch with the relevant module maintainer before you start on a patch.

- fix bugs
- write tests
- review Qt code
- write Qt code
- participate in the release process

Qt Coding Guidelines  See Also:

http://wiki.qt-project.org/Main_Page
http://wiki.qt-project.org/Coding_Style
http://wiki.qt-project.org/Coding_Conventions
http://wiki.qt-project.org/API_Design_Principles
http://wiki.qt-project.org/Binary_Compatibility_Workarounds
http://wiki.qt-project.org/Branch_Guidelines
http://wiki.qt-project.org/Qt_Localization
http://wiki.qt-project.org/Qt_In_Namepsace
http://wiki.qt-project.org/Transition_from_Qt_4.x_to_Qt5

Things you need to know before you start writing Qt code.

- Qt Framework Coding Style and Coding Conventions
• Qt Creator Coding Style & Conventions
• API Design Principles
• Binary Compatibility Workarounds
• Branch Guidelines
• Qt Localization
• Qt in Namespace
• Transition from Qt 4.x to Qt5

Creating a new module or tool for Qt

• Project playground
• Naming guidelines
• Module repository structure

Qt Software Quality Engineering  See Also:

• Qualité logiciel et test du logiciel
• http://wiki.qt-project.org/Main_Page
• http://wiki.qt-project.org/Writing_Unit_Tests
• http://wiki.qt-project.org/Qt_Quality_Gate_Criteria
• http://wiki.qt-project.org/Public_Autotest_Infrastructure
• http://wiki.qt-project.org/New_Unit_Test_Structure

Things you need to know to ensure good code quality.

• Writing Unit Tests
• Qt Quality Gate Criteria
• Public Autotest Infrastructure
• New Unit Test Structure

Tooling  See Also:

• http://wiki.qt-project.org/Main_Page
• http://wiki.qt-project.org/Setting_up_Gerrit
• http://wiki.qt-project.org/Gerrit_Introduction
• http://wiki.qt-project.org/Code_Reviews
• http://wiki.qt-project.org/Early_Warning_System

Things you need to know before you submit your code.

• Setting up Gerrit
• Gerrit Introduction
• Contributing and Reviewing Code
• Early Warning System
Qt creator

See Also:

- http://en.wikipedia.org/wiki/Qt_Creator
- http://qt.gitorious.org/qt-creator
- http://qt.gitorious.org/qt-creator/pages/Home
- http://planetqt.org/
- http://qt.gitorious.org/qt/qt
- http://bitbucket.org/razvanpetru/qt-components/
- http://qt-apps.org/

Qt Creator is a cross-platform C++ integrated development environment which is part of the Qt SDK.
It includes a visual debugger and an integrated GUI layout and forms designer.
The editor’s features includes syntax highlighting and autocompletion.
Qt Creator is a new cross-platform integrated development environment (IDE) tailored to the needs of Qt developers.
It includes:

- An advanced C++ code editor
- Integrated GUI layout and forms designer
- Project and build management tools
- Integrated, context-sensitive help system
- Visual debugger
- Rapid code navigation tools
- Supports multiple platforms

Qt creator versions  See Also:


Qt Creator 2.5.0 (May 9, 2012)  See Also:

- http://qt-project.org/news/view/qt-creator-2.5.0-released
- http://labs.qt.nokia.com/2012/05/09/qt-creator-2.5.0-released/

The Qt tools team in Berlin has released a new stable version of Qt Creator today.
You can read about all the great improvements and features on Eike’s release blog. Or just get it straight away from our downloads page.
The Qt Creator 2.5.0 final has been released! There are lots of new features and improvements in this release, I’ll highlight a few here, some others are probably already mentioned in our beta blog post, and you’ll find a more complete list in our changes file.

So, new features and improvements include but are not limited to:

- You can repeat a recent search with the same parameters with a simple click on “Search Again”
- “Execute” Locator filter lets you run arbitrary commands in a shell from Qt Creator (“! <some command>”) (thanks to Yuchen Deng!)
- Experimental plugin that shows “TODO” items from your sources (thanks to Dmitry Savchenko and Vasily Sorokin!)
- Experimental plugin for autotools based projects (thanks to Patricia Santana Cruz and Openismus GmbH!)
- Mac OS X Lion users will be happy to know that QTCREATORBUG-6222 which prevented adding some Qt Versions has finally been fixed
- A very basic version of a C++ refactoring action that adds an #include for an unknown identifier has been added (move cursor on identifier, press Alt+Return (Option+Return on Mac OS X))
- A very basic version of a C++ “extract method” refactoring action
- Improved support of C++11 (nullptr, constexpr, static_assert, noexcept, inline namespaces, auto, lambdas)
- Rearrange C++ method arguments (thanks to Bojan Petrovic!)
- New hints and warnings for QML code, including an option to prevent them for specific lines (with a special comment)

Qt Creator 2.4.0 See Also:

- http://qt.developpez.com/actu/40030/Qt-Creator-2-4-est-sorti-cette-nouvelle-version-de-l-EDI-pour-Qt-ameliore-le-support-de-QML/

We are happy to announce that finally Qt Creator 2.4.0 is ready to be published.

I’ll summarize a few of the great amount of features and improvements in this release.

You find a longer list in our change log, and you can also have a look at what we fixed for 2.4.0 in our bugtracker.

The commit log from our repository is only recommended if you have a lot of time: Around 1300 commits have been accepted since 2.3.1.

Lots of thanks go to the contributors, you find a list at the end of the change log :) 

My personal highlight:

- The “synchronize declaration and definition” quick fix, and the fixes for the “create definition from declaration”. When you change either declaration or definition you’ll notice a little light bulb appearing. If you now press “alt+return” (or click the bulb) your changes are applied to the counter part.
- “Schemes” for the coding styles (C++ and QML), reusable between projects
• QML rename usages and semantic highlighting
• Access most recent searches
• Encoding fixes for search & replace and the refactorings
• Subversion 1.7 support
• Netbook users will be glad to hear that the preferences dialog now has a decent size again

At the end a few words of advice: Because of the coding schemes, you’ll now need to change the coding style to change tab and indentation settings for code (was done in text editor settings before).

Qt Creator tries to migrate existing text editor settings to the new style for your projects, but if you had project specific settings before you’ll need to set them again.

If you are confused where the Qt version settings have gone: They are now in the “Build & Run” category, together with the tool chain and the general project settings.

You can download the standalone release from DevNet, and the release is going out as an update to users of the Qt SDK (might take a few hours to spread through the world).

Qt creator 2.3.0  See Also:


We are delighted to announce the new Qt Creator 2.3.0 release today.

The release adds lots of smaller and bigger improvements targeting the general developer experience. Since you might not have followed the prereleases I will repeat some of the most important ones here:

• Coding style options for C++ have been vastly improved and can be defined globally and on a per project basis. Also see the blog post.
• Profiling has moved to its own “Analyze” mode, and additionally supports profiling QML applications and profiling with Valgrind’s Callgrind.
• Support for “generic remote Linux devices” has been added. You define your connection for a “Linux Device” in the preferences, and add corresponding deploy and run configurations to your project’s run settings, and there you go. You’ll have to make sure that you use a suitable toolchain for building your project yourself though.
• Debugging and profiling Qt Quick applications now works for Symbian and Meego 1.2 Harmattan devices with Qt 4.7.4 installed.
• Various other improvements to Qt Quick support, like views, models and delegates an Qt Quick Designer including mockup data, improved Live Preview (i.e. modifying your QML while running in a preview), …

Qt creator 2.2.0  See Also:


While we were stabilizing Qt Creator 2.1.0 for the Qt SDK, we were already heavily working on the features and fixes for the next version.

So we are able to proudly present a feature complete beta release of Qt Creator 2.2 today (24 march 2011). Of course this release is overshadowed by the death of our beloved polar bear, Knut, so, in memory of him this release is codenamed “Knut Creator”.

402 Chapter 2. Development
Qt creator 2.1.0  See Also:

http://labs.qt.nokia.com/2011/03/01/qt-creator-2-1-0-released/

Today we release Qt Creator 2.1.0 as well as the Qt SDK 1.1 beta and Qt 4.7.2.

We did a few important fixes since the 2.1.0 release candidate, including adaptions of the mobile application project wizards and deployment related files, and updated documentation. Together with the Qt SDK 1.1 beta we are now confident that Qt Creator 2.1.0 is ready to provide a great experience.

The overall agenda for Qt Creator 2.1.0 was enhanced Qt Quick and mobile application support.

And of course it got lots of improvements in all other areas as well.

For all of you who didn’t follow the pre-releases closely, here are again a few random things from the release (no specific order, no specific relevance, not complete):

- Semantic C++ highlighting: Highlighting of types (no “Q…” magic anymore, hurray), local variables vs members, virtual methods
- Generic highlighting adds highlighting for various file types based on the Kate highlight definition specification
- C++ class view and image viewer contributed by Denis Mingulov
- Outline views for C++ and QML
- Searching for C++ symbols matching a pattern (via advanced find dialog)
- Wizard for adding libraries to pro files, including all the necessary fancy magic for include paths and static libs
- Various debugging improvements on all platforms
- Improved QML code editing with a faster code model and a new indenter
- Find usages and improved follow symbols in QML code
- Graphical QML tool bar (that you can get on request) for setting e.g. fonts and colors in the QML code editor
- Project wizards for Qt Quick applications that also handle deployment to devices, and for custom QML extension plugins
- Mobile application project wizards that provide you with the needed setup for Symbian and Maemo, and packaging and deployment
- Various improvements to deployment to Maemo targets and Symbian support
- Click on QObject::connect warnings in application output to jump to the code
- More, more, more

So grab the Qt SDK 1.1 beta release (includes Qt Creator 2.1.0) or get the Qt Creator-only binary packages from our download server.

Qt core

See Also:

- http://qt.developpez.com/faq/?page=qt4Core
- http://qt.developpez.com/faq/
Qt i18n

See Also:

• Internationalization (i18n)
  • http://doc.qt.nokia.com/4.6/internationalization.html
  • http://doc.qt.nokia.com/4.6/i18n-source-translation.html

Internationalization with Qt  The internationalization of an application is the process of making the application usable by people in countries other than one’s own.

Relevant Qt Classes and APIs  These classes support internationalizing of Qt applications.

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QInputContext</td>
<td>Abstracts the input method dependent data and composing state</td>
</tr>
<tr>
<td>QLocale</td>
<td>Converts between numbers and their string representations in various languages</td>
</tr>
<tr>
<td>QSystemLocale</td>
<td>Can be used to finetune the system locale of the user</td>
</tr>
<tr>
<td>QTextCodec</td>
<td>Conversions between text encodings</td>
</tr>
<tr>
<td>QTextDecoder</td>
<td>State-based decoder</td>
</tr>
<tr>
<td>QTextEncoder</td>
<td>State-based encoder</td>
</tr>
<tr>
<td>QTranslator</td>
<td>Internationalization support for text output</td>
</tr>
<tr>
<td>Translation Rules for Plurals</td>
<td>A summary of the translation rules for plurals produced by Qt’s i18n tools.</td>
</tr>
</tbody>
</table>
- Special line breaking behavior. Some of the Asian languages are written without spaces between words. Line breaking can occur either after every character (with exceptions) as in Chinese, Japanese and Korean, or after logical word boundaries as in Thai.

- Bidirectional writing. Arabic and Hebrew are written from right to left, except for numbers and embedded English text which is written left to right. The exact behavior is defined in the Unicode Technical Annex #9.

- Non-spacing or diacritical marks (accents or umlauts in European languages). Some languages such as Vietnamese make extensive use of these marks and some characters can have more than one mark at the same time to clarify pronunciation.

- Ligatures. In special contexts, some pairs of characters get replaced by a combined glyph forming a ligature. Common examples are the fl and fi ligatures used in typesetting US and European books.

https://secure.wikimedia.org/wikipedia/fr/wiki/Qt#Internationalisation  Qt intègre son propre système de traduction, qui n’est pas foncièrement différent dans le principe de la bibliothèque gettext.

Selon le manuel de Qt Linguist, l’internationalisation est assurée par la collaboration de trois types de personnes : les développeurs, le chef de projet et les traducteurs.

Dans leur code source, les développeurs entrent des chaînes de caractères dans leur propre langue. Ils doivent permettre la traduction de ces chaînes grâce à la méthode tr().

En cas d’ambiguïté sur le sens d’une expression, ils peuvent également indiquer des commentaires destinés à aider les traducteurs.

Le chef de projet déclare les fichiers de traduction (un pour chaque langue) dans le fichier de projet. L’utilitaire lupdate parcourt les sources à la recherche de chaînes à traduire et synchronise les fichiers de traduction avec les sources.

Fichier de traduction XML (.ts et .qm)  Les fichiers de traductions sont des fichiers XML portant l’extension .ts.

Les traducteurs utilisent Qt Linguist pour renseigner les fichiers de traduction.

Quand les traductions sont finies, le chef de projet peut compiler les fichiers .ts à l’aide de l’utilitaire lrelease qui génère des fichiers binaires portant l’extension .qm, exploitables par le programme.

Ces fichiers sont lus à l’exécution et les chaînes de caractères qui y sont trouvées remplacent celles qui ont été écrites par les développeurs.

Qt unicode support  See Also:


Translation of a Qt application is a four-step process  Translation of a Qt application is a three-step process:

0. Prepare the source code for the translation

1. Run lupdate to extract translatable text from:
   - the C++ source code of the Qt application,
   - the .ui file produced by QtCreator

resulting in a message file for translators (a .ts file).

The utility recognizes the tr() construct and the QT_TR_*_NOOP() macros described above and produces TS files (usually one per language).
2. Provide translations for the source texts in the TS file, using Qt Linguist. .. note:: Since TS files are in XML format, you can also edit them by hand.

3. Run lrelease to obtain a light-weight message file (a QM file) from the TS file, suitable only for end use. Think of the TS files as “source files”, and QM files as “object files”. The translator edits the TS files, but the users of your application only need the QM files. Both kinds of files are platform and locale independent.

Typically, you will repeat these steps for every release of your application. The lupdate utility does its best to reuse the translations from previous releases.

Qt prepare translation

Prepare the source code for the translation

Use QString for All User-Visible Text   See Also:

• unicode 6.0
• unicode 5.0

Since QString uses the Unicode 5.1 encoding internally, every language in the world can be processed transparently using familiar text processing operations. Also, since all Qt functions that present text to the user take a QString as a parameter, there is no char * to QString conversion overhead.

Strings that are in “programmer space” (such as QObject names and file format texts) need not use QString; the traditional char * or the QByteArray class will suffice.

You’re unlikely to notice that you are using Unicode; QString, and QChar are just like easier versions of the crude const char * and char from traditional C.

Use tr() for All Literal Text   Wherever your program uses “quoted text” for text that will be presented to the user, ensure that it is processed by the QCoreApplication::translate() function.

Essentially all that is necessary to achieve this is to use QObject::tr(). For example, assuming the LoginWidget is a subclass of QWidget:

```cpp
LoginWidget::LoginWidget()
{
    QLabel *label = new QLabel(tr("Password:"));
    ...}
```

Note: This accounts for 99% of the user-visible strings you’re likely to write.

If the quoted text is not in a member function of a QObject subclass, use either the tr() function of an appropriate class, or the QCoreApplication::translate() function directly:

```cpp
void some_global_function(LoginWidget *logwid)
{
    QLabel *label = new QLabel(
        LoginWidget::tr("Password:"), logwid);
}

void same_global_function(LoginWidget *logwid)
{
```
QLabel *label = new QLabel(
    qApp->translate("LoginWidget", "Password:"), logwid);
}

If you need to have translatable text completely outside a function, there are two macros to help:

- QT_TR_NOOP()
- and QT_TRANSLATE_NOOP().

They merely mark the text for extraction by the lupdate utility described below. The macros expand to just the text (without the context).

Example of QT_TR_NOOP():

QString FriendlyConversation::greeting(int type)
{
    static const char *greeting_strings[] = {
        QT_TR_NOOP("Hello"),
        QT_TR_NOOP("Goodbye")
    };
    return tr(greeting_strings[type]);
}

Example of QT_TRANSLATE_NOOP():

static const char *greeting_strings[] = {
    QT_TRANSLATE_NOOP("FriendlyConversation", "Hello"),
    QT_TRANSLATE_NOOP("FriendlyConversation", "Goodbye")
};

QString FriendlyConversation::greeting(int type)
{
    return tr(greeting_strings[type]);
}

QString global_greeting(int type)
{
    return qApp->translate("FriendlyConversation",
        greeting_strings[type]);
}

If you disable the const char * to QString automatic conversion by compiling your software with the macro QT_NO_CAST_FROM_ASCII defined, you’ll be very likely to catch any strings you are missing. See QString::fromLatin1() for more information. Disabling the conversion can make programming a bit cumbersome.

If your source language uses characters outside Latin1, you might find QObject::trUtf8() more convenient than QObject::tr(), as tr() depends on the QTextCodec::codecForTr(), which makes it more fragile than QObject::trUtf8().

Use QString::arg() for Dynamic Text  The QString::arg() functions offer a simple means for substituting arguments:

```cpp
void FileCopier::showProgress(int done, int total,
    const QString &currentFile)
{
    label.setText(tr("%1 of %2 files copied.
        Copying: %3")
        .arg(done)
        .arg(total)
```

2.14. IDEs 407
In some languages the order of arguments may need to change, and this can easily be achieved by changing the order of the % arguments.

For example:

```cpp
QString s1 = "%1 of %2 files copied. Copying: %3";
QString s2 = "Kopierer nu %3. Av totalt %2 filer er %1 kopiert.";
qDebug() << s1.arg(5).arg(10).arg("somefile.txt");
qDebug() << s2.arg(5).arg(10).arg("somefile.txt");
```

produces the correct output in English and Norwegian:

5 of 10 files copied. Copying: somefile.txt
Kopierer nu somefile.txt. Av totalt 10 filer er 5 kopiert.

Qt produce translations  See Also:

Qt lupdate program  lupdate is part of Qt’s Linguist tool chain.
It extracts translatable messages from:

- Qt UI files
- C++
- Java
- JavaScript/QtScript

Extracted messages are stored in textual translation source files (typically Qt TS XML).
New and modified messages can be merged into existing TS files.

lupdate

Usage:

```
lupdate [options] [project-file]...
lupdate [options] [source-file|path|@lst-file]... -ts ts-files|@lst-file
```

lupdate is part of Qt’s Linguist tool chain. It extracts translatable messages from Qt UI files, C++, Java and JavaScript/QtScript source code. Extracted messages are stored in textual translation source files (typically Qt TS XML). New and modified messages can be merged into existing TS files.

Options:

- `-help` Display this information and exit.
  - `-no-obsolete`
    - Drop all obsolete strings.
  - `-extensions <ext>[,<ext>...]`
    - Process files with the given extensions only.
The extension list must be separated with commas, not with whitespace.
Default: ‘java,jui,ui,c,c++,cc,cpp,cxx,ch,h,h++,hh,hpp,hxx,js,qs,qml’

-pluralonly
  Only include plural form messages.
-silent
  Do not explain what is being done.
-no-sort
  Do not sort contexts in TS files.
-no-recursive
  Do not recursively scan the following directories.
-recursive
  Recursively scan the following directories (default).
-I <includepath> or -I<includepath>
  Additional location to look for include files.
    May be specified multiple times.
-locations (absolute|relative|none)
  Specify/override how source code references are saved in TS files.
    Default is absolute.
-no-ui-lines
  Do not record line numbers in references to UI files.
-disable-heuristic {sametext|similartext|number}
  Disable the named merge heuristic. Can be specified multiple times.
-pro <filename>
  Name of a .pro file. Useful for files with .pro file syntax but
different file suffix. Projects are recursed into and merged.
-source-language <language>[_<region>]
  Specify the language of the source strings for new files.
    Defaults to POSIX if not specified.
-target-language <language>[_<region>]
  Specify the language of the translations for new files.
    Guessed from the file name if not specified.
-ts <ts-file>...
  Specify the output file(s). This will override the TRANSLATIONS
  and nullify the CODECFORTR from possibly specified project files.
-codecfortr <codec>
  Specify the codec assumed for tr() calls. Effective only with -ts.
-version
  Display the version of lupdate and exit.
@lst-file
  Read additional file names (one per line) from lst-file.

**lupdate Usage**

lupdate myproject.pro

**lupdate** is a command line tool that finds the translatable strings in the specified source, header and Qt Designer interface files, and produces or updates .ts translation files.

The files to process and the files to update can be set at the command line, or provided in a .pro file specified as an command line argument.

The produced translation files are given to the translator who uses Qt Linguist to read the files and insert the translations.

Companies that have their own translators in-house may find it useful to run lupdate regularly, perhaps monthly, as the application develops.
This will lead to a fairly low volume of translation work spread evenly over the life of the project and will allow the translators to support a number of projects simultaneously.

Companies that hire in translators as required may prefer to run lupdate only a few times in the application’s life cycle, the first time might be just before the first test phase.

This will provide the translator with a substantial single block of work and any bugs that the translator detects may easily be included with those found during the initial test phase.

The second and any subsequent lupdate runs would probably take place during the final beta phase.

The TS file format is a simple human-readable XML format that can be used with version control systems if required. lupdate can also process Localization Interchange File Format (XLIFF) format files; files in this format typically have file names that end with the .xlf suffix.

Note: The minimum supported version for XLIFF format files is 1.1. XLIFF 1.0 version files are not supported.

Produce the .ts files from a .ui file  In this example we want to have 5 translations:

- english
- deutsch
- français
- italiano
- español

In order to produce .ts files from a .ui file 2 steps:

1. add the following lines in your .pro file:

   TRANSLATIONS = your_project_en.ts
   TRANSLATIONS = your_project_de.ts
   TRANSLATIONS = your_project_fr.ts
   TRANSLATIONS = your_project_it.ts
   TRANSLATIONS = your_project_es.ts

2. type the following command in your terminal:

   lupdate your_project.pro

   Updating ‘your_project_en.ts’...
   Found 170 source text(s) (0 new and 170 already existing)
   Updating ‘your_project_de.ts’...
   Found 170 source text(s) (170 new and 0 already existing)
   Updating ‘your_project_fr.ts’...
   Found 170 source text(s) (170 new and 0 already existing)
   Updating ‘your_project_it.ts’...
   Found 170 source text(s) (170 new and 0 already existing)
   Updating ‘your_project_es.ts’...
   Found 170 source text(s) (170 new and 0 already existing)

Qt translate with Qt linguist GUI
Qt use translations  See Also:

lrelease

Usage

lrelease -h

Usage:
   lrelease [options] project-file
   lrelease [options] ts-files [-qm qm-file]

lrelease is part of Qt’s Linguist tool chain. It can be used as a stand-alone tool to convert XML-based translations files in the TS format into the ‘compiled’ QM format used by QTranslator objects.
Options:
- `help` Display this information and exit
- `idbased`
  Use IDs instead of source strings for message keying
- `compress`
  Compress the .qm files
- `noundefinished`
  Do not include unfinished translations
- `removeidentical`
  If the translated text is the same as the source text, do not include the message
- `markuntranslated <prefix>`
  If a message has no real translation, use the source text prefixed with the given string instead
- `silent`
  Do not explain what is being done
- `version`
  Display the version of lrelease and exit

**QTranslator class**  See Also:

*List of ISO 639-1 codes*

The QTranslator class provides internationalization support for text output.

An object of this class contains a set of translations from a source language to a target language.

QTranslator provides functions to look up translations in a translation file.

Translation files are created using Qt Linguist.

The most common use of QTranslator is to:

- load a translation file,
- install it using QApplication::installTranslator(),
- use it via QObject::tr().

```cpp
// i18n translation
QTranslator qtTranslator;
QString locale = QLocale::system().name().section('_', 0, 0);
qtTranslator.load("control_panel_" + locale);
control_panel.installTranslator(&qtTranslator);
```

**Qt where to put translations**  See Also:

http://stackoverflow.com/questions/4034158/loading-qm-file

**In a given directory**  Where are the .qm files located? Your code is attempting to load the file from the current working directory, which can be anything during runtime.

Specify a directory path in the call to QTranslator::load:
QTranslator* translator = new QTranslator();
if (translator->load("hellotr_la", "/path/to/folder/with/qm/files")) {
    app.installTranslator(translator);
}

In Qt resource
Translations can be loaded from Qt resources, so it is a good idea to bundle them inside your executables. Then you would load them somewhat like this:

QTranslator* translator = new QTranslator();
if (translator->load("hellotr_la", ":/translations")) {
    app.installTranslator(translator);
}

Support for unicode encodings  See Also:

- http://jimmyg.org/work/code/stringconvert/0.3.0/manual.html

The QTextCodec class and the facilities in QTextStream make it easy to support many input and output encodings for your users data.
When an application starts, the locale of the machine will determine the 8-bit encoding used when dealing with 8-bit data: such as for font selection, text display, 8-bit text I/O, and character input.

The application may occasionally require encodings other than the default local 8-bit encoding.

For example, an application in a Cyrillic KOI8-R locale (the de-facto standard locale in Russia) might need to output Cyrillic in the ISO 8859-5 encoding.

Code for this would be:

```cpp
QString string = ...; // some Unicode text
QTextCodec *codec = QTextCodec::codecForName("ISO 8859-5");
QByteArray encodedString = codec->fromUnicode(string);
```

**Unicode to UTF-8 encoding**

For converting Unicode to local 8-bit encodings, a shortcut is available: the `QString::toLocal8Bit()` function returns such 8-bit data.

Another useful shortcut is `QString::toUtf8()`, which returns text in the 8-bit UTF-8 encoding: this perfectly preserves Unicode information while looking like plain ASCII if the text is wholly ASCII.

**UTF-8 to unicode**

For converting the other way, there are the `QString::fromUtf8()` and `QString::fromLocal8Bit()` convenience functions, or the general code, demonstrated by this conversion from ISO 8859-5 Cyrillic to Unicode conversion:

```cpp
QByteArray encodedString = ...; // some ISO 8859-5 encoded text
QTextCodec *codec = QTextCodec::codecForName("ISO 8859-5");
QString string = codec->toUnicode(encodedString);
```

**Note:** Ideally Unicode I/O should be used as this maximizes the portability of documents between users around the world, but in reality it is useful to support all the appropriate encodings that your users will need to process existing documents.

In general, Unicode (UTF-16 or UTF-8) is best for information transferred between arbitrary people, while within a language or national group, a local standard is often more appropriate.

**codec for locale**

The most important encoding to support is the one returned by `QTextCodec::codecForLocale()`, as this is the one the user is most likely to need for communicating with other people and applications (this is the codec used by `local8Bit()`).

Qt supports most of the more frequently used encodings natively.

For a complete list of supported encodings see the `QTextCodec` documentation.

In some cases and for less frequently used encodings it may be necessary to write your own `QTextCodec` subclass.

Depending on the urgency, it may be useful to contact Qt’s technical support team or ask on the `qt-interest` mailing list to see if someone else is already working on supporting the encoding.

**Translating Non-Qt Classes**

See Also:


It is sometimes necessary to provide internationalization support for strings used in classes that do not inherit `QObject` or use the `Q_OBJECT` macro to enable translation features.
Since Qt translates strings at run-time based on the class they are associated with and \texttt{lupdate} looks for translatable strings in the source code, \textbf{non-Qt classes must use mechanisms that also provide this information.}

One way to do this is to add translation support to a non-Qt class using the \texttt{Q_DECLARE_TR_FUNCTIONS()} macro; for example:

```cpp
class MyClass
{
    Q_DECLARE_TR_FUNCTIONS(MyClass)

    public:
        MyClass();

    ...}
```

This provides the class with tr()_ functions that can be used to translate strings associated with the class, and makes it possible for \texttt{lupdate} to find translatable strings in the source code.

Alternatively, the \texttt{QCoreApplication::translate()} function can be called with a specific context, and this will be recognized by \texttt{lupdate} and \texttt{Qt Linguist}.

\textbf{Qt dynamic translation} Some applications, such as Qt Linguist, must be able to support changes to the user's language settings while they are still running.

To make widgets aware of changes to the installed QTranslators, reimplement the widget's \texttt{changeEvent()} function to check whether the event is a LanguageChange event, and update the text displayed by widgets using the tr() function in the usual way. For example:

```cpp
void MyWidget::changeEvent(QEvent *event)
{
    if (e->type() == QEvent::LanguageChange) {
        titleLabel->setText(tr("Document Title"));
        ...
        okPushButton->setText(tr("&OK"));
    } else
        QWidget::changeEvent(event);
}
```

All other change events should be passed on by calling the default implementation of the function.

The list of installed translators might change in reaction to a LocaleChange event, or the \textbf{application might provide a user interface that allows the user to change the current application language.}

The default event handler for QWidget subclasses responds to the QEvent::LanguageChange event, and will call this function when necessary.

\texttt{LanguageChange} events are posted when a new translation is installed using the \texttt{QCoreApplication::installTranslator()} function.

Additionally, other application components can also force widgets to update themselves by posting LanguageChange events to them.

\textbf{Qt unicode system support} \hfill \textbf{Warning:} Some of the operating systems and windowing systems that Qt runs on only have limited support for \texttt{Unicode.}

The level of support available in the underlying system has some influence on the support that Qt can provide on those platforms, although in general Qt applications need not be too concerned with platform-specific limitations.
Unix/X11

- Locale-oriented fonts and input methods. Qt hides these and provides Unicode input and output.
- Filesystem conventions such as UTF-8 are under development in some Unix variants. All Qt file functions allow Unicode, but convert filenames to the local 8-bit encoding, as this is the Unix convention (see QFile::setEncodingFunction() to explore alternative encodings).
- File I/O defaults to the local 8-bit encoding, with Unicode options in QTextStream.
- Many Unix distributions contain only partial support for some locales. For example, if you have a /usr/share/locale/ja_JP.EUC directory, this does not necessarily mean you can display Japanese text; you also need JIS encoded fonts (or Unicode fonts), and the /usr/share/locale/ja_JP.EUC directory needs to be complete. For best results, use complete locales from your system vendor.

Windows

- Qt provides full Unicode support, including input methods, fonts, clipboard, drag-and-drop and file names.
- File I/O defaults to Latin1, with Unicode options in QTextStream. Note that some Windows programs do not understand big-endian Unicode text files even though that is the order prescribed by the Unicode Standard in the absence of higher-level protocols.
- Unlike programs written with MFC or plain winlib, Qt programs are portable between Windows 98 and Windows NT. You do not need different binaries to support Unicode.

Mac OS X For details on Mac-specific translation, refer to the Qt/Mac Specific Issues document here.

Qt localization (l10n)

See Also:

Time localization Localization is the process of adapting to local conventions, for example presenting dates and times using the locally preferred formats.

Such localizations can be accomplished using appropriate tr()_ strings..

In the example, for the US we would leave the translation of “AMPM” as it is and thereby use the 12-hour clock branch; but in Europe we would translate it as something else and this will make the code use the 24-hour clock branch.

For localized numbers use the QLocale class.

For localized numbers use the QLocale class Localization is not recommended. Choose clear icons that are appropriate for all localities, rather than relying on local puns or stretched metaphors.

The exception is for images of left and right pointing arrows which may need to be reversed for Arabic and Hebrew locales.

Qt links
Contents

- Qt links
  - Links
  - QObject
  - Properties
  - Bytes, bits
  - Conteneurs
  - Qt creator
  - PyQt
  - livres

Links

- http://qt.gitorious.org/ ('Qt is a cross-platform application and UI framework that allows developers to write applications once and deploy them across many desktop and embedded operating systems without rewriting the source code.
- http://doc.trolltech.com/
- http://qt.nokia.com/developer
- http://qt.gitorious.org/qt/pages/QtCodingStyle
- http://qt.gitorious.org/qt/pages/CodingConventions
- http://qt.gitorious.org/qt/pages/ApiDesignPrinciples:
  - http://qt.gitorious.org/qt/pages/ApiDesignPrinciples#The+Art+of+Naming ('Naming is probably the single most important issue when designing an API. What should the classes be called? What should the member functions be called?')
  - http://chaos.troll.no/~shausman/api-design/api-design.pdf
- http://doc.qt.nokia.com/4.6/object.html (Qt Object Model)
- http://doc.qt.nokia.com/4.6/signalsandslots.html:
  - présentation des signaux et slots qui sont une implémentation du patron de conception observateur
  - http://www.pyside.org/docs/pseps/psep-0100.html
  - http://www.riverbankcomputing.co.uk/static/Docs/PyQt4/pyqt4ref.html#new-style-signal-and-slot-support

QObject

- http://doc.qt.nokia.com/4.6/qobject.html#Q_OBJECT
- http://doc.qt.nokia.com/4.6/qobject.html#Q_FLAGS
Properties

- [http://doc.qt.nokia.com/4.6/properties.html](http://doc.qt.nokia.com/4.6/properties.html) (‘Qt provides a sophisticated property system similar to the ones supplied by some compiler vendors. However, as a compiler- and platform-independent library, Qt does not rely on non-standard compiler features like __property or [property]. The Qt solution works with any standard C++ compiler on every platform Qt supports. It is based on the Meta-Object System that also provides inter-object communication via signals and slots.’)

- [http://doc.qt.nokia.com/4.6/qobject.html#Q_PROPERTY](http://doc.qt.nokia.com/4.6/qobject.html#Q_PROPERTY)


Bytes, bits

- [http://doc.qt.nokia.com/4.6/qbytearray.html](http://doc.qt.nokia.com/4.6/qbytearray.html) (‘The QByteArray class provides an array of bytes’)


  A QBitArray is an array that gives access to individual bits and provides operators (AND, OR, XOR, and NOT) that work on entire arrays of bits’)

- [http://doc.qt.nokia.com/4.6/qbitarray.html#toggleBit](http://doc.qt.nokia.com/4.6/qbitarray.html#toggleBit)

- [http://doc.qt.nokia.com/4.6/qdatastream.html](http://doc.qt.nokia.com/4.6/qdatastream.html) (‘A data stream is a binary stream of encoded information which is 100% independent of the host computer’s operating system, CPU or byte order. For example, a data stream that is written by a PC under Windows can be read by a Sun SPARC running Solaris.’) String, unicode, i18n (internationalisation)

- [http://doc.qt.nokia.com/4.6/unicode.html](http://doc.qt.nokia.com/4.6/unicode.html) (‘To fully benefit from Unicode, we recommend using QString for storing all user-visible strings, and performing all text file I/O using QTextStream’)

- [http://doc.qt.nokia.com/4.6/i18n.html](http://doc.qt.nokia.com/4.6/i18n.html)

- [http://doc.qt.nokia.com/4.6/linguist-programmers.html#tutorials](http://doc.qt.nokia.com/4.6/linguist-programmers.html#tutorials) * (‘Support for multiple languages is extremely simple in Qt applications, and adds little overhead to the programmer’s workload.’)


- [http://doc.qt.nokia.com/4.6/qtextstream.html](http://doc.qt.nokia.com/4.6/qtextstream.html) (‘The QTextStream class provides a convenient interface for reading and writing text. QTextStream can operate on a QIODevice, a QByteArray or a QString.’)


- [http://doc.qt.nokia.com/4.6/qstring.html#fromUtf8](http://doc.qt.nokia.com/4.6/qstring.html#fromUtf8)


- [http://doc.qt.nokia.com/4.6/qobject.html#tr](http://doc.qt.nokia.com/4.6/qobject.html#tr)

Conteneurs

- [http://doc.qt.nokia.com/4.6/containers.html](http://doc.qt.nokia.com/4.6/containers.html) (‘These container classes are designed to be lighter, safer, and easier to use than the STL containers. If you are unfamiliar with the STL, or prefer to do things the “Qt way”, you can use these classes instead of the STL classes’)

- [http://doc.qt.nokia.com/4.6/containers.html#the-container-classes](http://doc.qt.nokia.com/4.6/containers.html#the-container-classes)


- [http://doc.qt.nokia.com/4.6/containers.html#the-iterator-classes](http://doc.qt.nokia.com/4.6/containers.html#the-iterator-classes) (‘Iterators provide a uniform means to access items in a container. Qt’s container classes provide two types of iterators: Java-style iterators and STL-style iterators’)
- [http://doc.qt.nokia.com/4.6/containers.html#java-style-iterators](http://doc.qt.nokia.com/4.6/containers.html#java-style-iterators) (‘The Java-style iterators are new in Qt 4 and are the standard ones used in Qt applications. They are more convenient to use than the STL-style iterators, at the price of being slightly less efficient. Their API is modelled on Java’s iterator classes.’)
- [http://doc.qt.nokia.com/4.6/containers.html#stl-style-iterators](http://doc.qt.nokia.com/4.6/containers.html#stl-style-iterators) (‘STL-style iterators have been available since the release of Qt 2.0. They are compatible with Qt’s and STL’s generic algorithms and are optimized for speed. Unlike Java-style iterators, STL-style iterators point directly at items’)
- [http://doc.qt.nokia.com/4.6/containers.html#the-foreach-keyword](http://doc.qt.nokia.com/4.6/containers.html#the-foreach-keyword) (‘if you just want to iterate over all the items in a container in order, you can use Qt’s foreach keyword. The keyword is a Qt-specific addition to the C++-language, and is implemented using the preprocessor’)
- Types de base
  - ...
- debug
- logging for Qt
  - [http://bitbucket.org/razvanpetru/qt-components](http://bitbucket.org/razvanpetru/qt-components)
- test
- qmake
• GUI

• http://doc.qt.nokia.com/4.6/model-view-introduction.html

• http://doc.qt.nokia.com/4.6/model-view-introduction.html#the-model-view-architecture (‘Models, views, and delegates communicate with each other using signals and slots. Generally, the model/view classes can be separated into the three groups described above: models, views, and delegates. Each of these components is defined by abstract classes that provide common interfaces and, in some cases, default implementations of features. Abstract classes are meant to be subclassed in order to provide the full set of functionality expected by other components; this also allows specialized components to be written’)

• http://doc.qt.nokia.com/4.6/qt.html#ItemFlag-enum


• http://doc.qt.nokia.com/4.6/qtuitools.html


Qt creator

• qt creator (‘The goal of Qt Creator is to provide a cross-platform, complete Integrated Development Environment (IDE) to develop Qt projects. It is available for the Linux, Mac OS X and Windows platforms.’)

• http://qt.gitorious.org/qt-creator/qt-creator/trees/master

• http://qt.nokia.com/about/news/qt-creator-2.0-released (‘Oslo, 22 June 2010 – Nokia today released Qt Creator 2.0, an updated version of its cross-platform integrated development environment (IDE) for use with the Qt framework. Qt Creator 2.0 brings a range of new features specifically designed to make mobile development for Symbian, MeeGo and other platforms faster and easier. Qt Creator 2.0 is available as part of the Nokia Qt SDK 1.0 – also released today – as part of an updated build of the Qt SDK, or as a standalone binary or source download. All Qt releases are available for download from http://qt.nokia.com/downloads’)

• http://linuxfr.org/2010/06/25/27060.html (‘...La prise en charge de Qt Quick (voir plus bas pour la définition) était une des fonctionnalités les plus attendues de cette nouvelle version. Il a malheureusement été désactivé par défaut. La raison est que Qt SDK embarque Qt Creator 2.0 et Qt 4.6.3 ; QML (une des technologies nécessaires à Qt Quick) quant à lui arrive avec Qt 4.7, qui n’est pas encore stabilisé. Quand Qt 4.7 sera stabilisé, Qt Creator 2.1 sortira afin de pouvoir exploiter Qt Quick.

• Qu’est-ce que Qt Quick ? C’est l’union des 3 technologies suivantes :

• QML : un langage déclaratif qui permet la création de GUI (Graphical User Interface) ;

• Qt Declarative : la glue entre les fichiers QML et l’application ;

• Qt Creator : cet IDE va embarquer la gestion de QML afin de pouvoir créer des interfaces facilement et rapidement. À bientôt pour la sortie de Qt Creator 2.1 et Qt 4.7 !’)

• http://qt.nokia.com/developer/changes/changes-qtcreator-2.0

• http://doc.trolltech.com/qtcreator-2.0/index.html

• git clone git://gitorious.org/qt-creator/qt-creator.git

• git log –cherry-pick –pretty=oneline v1.3.1...v2.0.0

• http://qt.nokia.com/about/news/nokia-releases-qt-4.6.3 (‘Oslo, 08 June 2010’)

• http://qt.nokia.com/developer/changes/changes-4.6.3

• http://doc.trolltech.com/4.6/index.html

• http://doc.trolltech.com/4.6/how-to-learn-qt.html
Development tools, Release 2012.06.18

• http://www.forum.nokia.com/Develop/Qt/Tools/

pyQt
• http://www.riverbankcomputing.co.uk/software/pyqt/download
• http://pypi.python.org/pypi/spyder/1.1.0
• pySide (‘The PySide project provides LGPL-licensed Python bindings for the Qt cross-platform application and UI framework. PySide Qt bindings allow both free open source and proprietary software development and ultimately aim to support all of the platforms as Qt itself’)
• http://en.wikipedia.org/wiki/PySide
• http://qt.gitorious.org/pyside
• http://www.pyside.org/faq/
• http://www.pyside.org/docs/pseps/
• http://qt.gitorious.org/pyside/pyside-examples/trees/master/examples
• http://qt.gitorious.org/pyside/pseps/trees/master
• http://qt.gitorious.org/pyside/pyside/trees/master
• http://www.pyside.org/downloads/ (‘Unfortunately, PySide has not quite yet been ported to Windows. Stay tuned!’)
• http://www.pyside.org/docs/pyside-dev/ (documentation écrite avec sphinx/rst)
• http://www.pyside.org/files/pyside-qt4.6+0.3.2.tar.bz2
• http://twitter.com/qtbynokia

livres
• http://www.informit.com/content/images/9780132354165/examples/qt-book-examples.zip

The Qt news

Qt 4.7.2 has now been released!

andy.shaw@nokia.com
heure de l’expéditeur Envoyé à 11:30 (GMT+01:00). Heure locale : 14:06.
à qt-announce@trolltech.com
date 1 mars 2011 11:30
objet [Qt-announce] Qt 4.7.2 has now been released!

Qt 4.7.2 has now been released.
This is a maintenance release which contains bug-fixes based on feedback and contributions since the Qt 4.7.1 release back in November.

Note: It also marks the official release of Qt Quick.
**Qt Quick, QML language**  Qt Quick is a new feature within Qt that makes it easy to create light-weight apps and UIs.

It contains the new QML language, the Qt Declarative module, and new tooling in Qt Creator to easily build apps.

Packages are available from the Qt Download page, and for those who prefer to get it directly from the public git repository at [http://qt.gitorious.org/](http://qt.gitorious.org/), there will be a “v4.7.2 tag soon.

Thank you to all of you that have provided bug reports and/or bug fixes since the Qt 4.7.1 release, some of these will have already been incorporated for Qt 4.7.2 and there will be more coming in future releases.

If you want to see what has changed since Qt 4.7.1 then you can see the Qt 4.7.2 changelog and the list of bugs reported fixed for Qt 4.7.2 in the Qt Bug Tracker.

For a more detailed view then you can compare the changes between the v4.7.1 and v4.7.2 tags in the public git repository.

To provide feedback, you can use the Qt Bug Tracker. To contribute code, documentation updates or autotests for Qt, then all you need to get started is at [http://qt.gitorious.org/](http://qt.gitorious.org/).

Regards, Qt Development Team

**Qt Quick 1.0 and beyond (or, the post of many links)**  See Also:


UI developers rejoice; Today we are launching Qt Quick 1.0! For those of you who have been active members of our growing community, there might not be a whole lot of zero-day news here, except perhaps a bit of clarity on where this path leads you. For those of you who have only recently discovered how great Qt Quick is, let me summarize a bit for you;

**Qt Quick is a collection of three technologies**

**QtDeclarative**  The native library in Qt deliving native integration and the Qt Quick UI runtime

See Also:


**QML**  the Qt MetaObject Language, allowing declaration of User Interfaces and Experiences

See Also:


**Qt Creator**  support for QML-based projects and code integration.

See Also:


Qt Quick and the QML language is declarative, using JavaScript in bindings and property values; so while QML reduces the footprint of your code while providing blistering performance, keep in mind that JavaScriptCore (the JavaScript engine we currently use) will require an additional memory overhead.

For the last year, QML and QtDeclarative has been available to both early adopters as well as mainstream developers through code pushed to gitorious as well as Qt 4.7.0 and 4.7.1 releases.
Since the first release of QML we have both validated the incredible productivity benefits of the engine and declarative language, but also received tremendous help from our community in making sure performance and stability is up to scratch.

A massive thanks to everybody who have embraced and help lift this technology up!

**Today we are launching the final piece of the puzzle; Qt Creator with the Qt Quick Designer!**

Qt Quick Designer is a WYSIWYG editor for QML UI’s, allowing you to visually create and arrange your UI’s while staying in perfect sync with your QML code.

No, this does not use a separate generative step like Qt Designer did – but directly manipulates your QML code.

You can get Qt Creator along with the Qt Quick Designer and all the other features supporting Qt Quick from our downloads page.

Our team of wizards and magicians in Berlin have been concocting this out in the open on gitorious.org, so you might have seen it already. If you’re interested in seeing future developments – please try out the master branch!

In the near-term future we are working on some additional features for Qt Quick that we think will become useful and increasingly relevant;

This includes use-case specific elements for pinching touch interaction, free form touch interaction, support for Right-To-Left layouting and many other features and suggestions that come from the community via bugreports.qt.nokia.com.

We also started last fall to build Qt Quick components for MeeGo and Symbian – and as you might have seen, we stopped pushing commits to Qt Quick components for MeeGo a while back. While work is still continuing at a brisk pace on the MeeGo style, the Symbian style has now been fully transitioned to gitorious.org, allowing anyone to try it out on their Symbian3 device. This makes creating new Apps with native look and feel a lot easier and obsoletes the task of each developer having to creating reusable elements such as Button’s, ListItem’s and so forth.


To provide feedback you can use bugreports, and if you want to discuss you can use the mailing list or IRC.

So, what about the desktop you say? Well, the desktop support in Qt is still going strong; we have our supported platforms that are tried and tested every day, with new tests and features being added all the time.

You can also use Qt Quick to spice up your traditional desktop application, with partial QML content that is fluently animated and adds visual differntiation to your UI.

Or, go all the way like mixd.tv have done. But as mentioned in Volker’s post, there isn’t a whole lot of innovation happening on the desktop (Yes I know, Lion will make me eat my words). But, we’re preparing for the future in our own way and are exploring ways to make Qt Quick components relevant on the desktop as well. <tease>More on that in an upcoming blog post…</tease>

There’s also even more greatness on the horizon. As graphic acceleration is becoming more mainstream in devices, the need for a canvas architecture driven by painter’s algorithm is diminishing, replaced by more direct approaches that gives the end-user more bang for the buck. Here’s where scenegraph will make your UI feel like velvet, and allow even more amazing UI’s based on OpenGL ES.

In addition, we are also exploring ways to simplify 3D creation and integration of 3D into UI with our Qt/3D integration to QML making it possible for your grandpa to make a teapot.

So, throwing our glove; Qt Quick 1.0 has landed. Time to conquer the world.

**Qt SDK**

**Qt SDK versions**
Contents
Qt SDK 1.2
  • Qt SDK 1.2
    – What Is New
    – Qt 4.8.0 news

What Is New  This update of the Qt SDK brings you new versions of the following components:

  • Final Qt 4.8.0 for desktop release that delivers Qt Quick 1.1, Qt platform abstraction, Qt WebKit 2.2, and threaded OpenGL. For more information about deprecated items and additions to the Qt API, see What’s New in Qt 4.8.

  • Update to the MeeGo Harmattan tool chain.

  • Update to the Symbian Complementary Package that delivers the Analyze Tool plugin in Qt Creator 2.4 and adds the CODA 1.0.6 installation package (which will co-exist with version 1.0.5).

  • Qt Creator 2.4.1 bug fix update to Qt Creator 2.4.0 that introduced new editor functions, such as refactoring actions and specifying and sharing editor settings more flexibly.

  • Mobility examples for MeeGo Harmattan and Symbian that are integrated to the Qt Creator Welcome mode. The examples cover a wide set of the Mobility API features, such as maps and Bluetooth.

  • An update to Notifications API that improves the end user experience and fixes issues in the Nokia N9 implementation of the API.

  • A new feature has been added to the Qt SDK Installer that enables you to specify network proxy settings.

Use the Qt SDK Updater to update the components.

Qt 4.8.0 news

Qt 4.8.0 , December 15, 2011  See Also:

  • http://labs.qt.nokia.com/2011/12/15/qt-4-8-0-released/
  • http://developer.qt.nokia.com/doc/qt-4.8/qt4-8-intro.html

Posted by Sinan Tanilkan on December 15, 2011 · 101 comments

Qt has reached another important evolutionary milestone today. We are very proud to announce that Qt 4.8.0 has now been released. Many people have worked long and hard to deliver Qt 4.8.0.

Today that hard work reaches final release maturity, and we are celebrating!

Featuring Qt Platform Abstraction, threaded OpenGL support, multithreaded HTTP and optimized file system access, Qt 4.8.0 can be downloaded as binary or source packages

Content  Those of you that have been testing and using Qt 4.8 through its earlier stages will know the key benefits it brings.

If you haven’t been following it, here are some key features that are new in Qt 4.8.0:
Qt Quick 1.1 introduces new changes such as new properties and better performance.
- Right-To-Left text support
- Improved image caching
- Text input improvements - Support for split-screen virtual keyboard
- PinchArea Element - enables simple pinch gesture handling
- New properties for QML Elements.

Qt Platform Abstraction (QPA) – Lighthouse QPA restructures the GUI stack to enable easier porting of Qt to different windowing systems and devices. More info on: Lighthouse has grown up now.

Threaded OpenGL support Enables those of us that are not OpenGL-ninjas to render OpenGL from more than one thread concurrently. More info on: Threaded OpenGL in 4.8.

Multithreaded HTTP HTTP requests are now handled in a separate thread by default. This should make application guis smoother, as networking will no longer use the main event loop.

Optimized file system acces The file system stack received some heavy lifting under the hood. The result is better I/O performance, achieved by reducing the number of system calls performed for I/O and by better use of cached data, when available. The improvements in performance can be seen across all platforms. Qt 4.8.0 ships with QtWebKit 2.2.1. More details of the major changes can be found in the Qt 4.8 beta blog post.

Qt 4.8 introduces changes to the Qt API

C++ 11 support Qt supports some of the features of the C++11 standard
- QList, QVector and QStringList can be initialized with initializer lists.
- Most of the tool classes have a move operator.
- It is possible to use lambda functions in some of the QtConcurrent functions.

Localization API Changes to the Localization APIs include improvements to QLocale and more support for different language code formats.
- QLocale::quoteString() - for localized quotes
- QLocale::createSeparatedList() - for localized list separation (e.g. “1, 2 and 3”)
- QLocale::bcp47Name() - for locale names in the canonical form according to RFC 5646 - BCP47
- QLocale::matchingLocales() - to get a list of locales that match a criteria - e.g. a list of locales that use French language.
- QLocale::firstDayOfWeek()
- QLocale::weekdays()
• QLocale::currencySymbol()
• QLocale::toCurrencyString() - number formatting for currencies
• QLocale::uiLanguages()
• QLocale::nativeLanguageName()
• QLocale::nativeCountryName()

IP Multicast API

Multithreaded HTTP

QThreadLocalStorage can now store simple objects in addition to pointers

Qt SDK version 1.1.4  See Also:
This 1.1.4 update of Qt SDK is very much focusing on bringing the SDK build targets to the level of the latest Symbian devices and the Nokia N9.
It means that you can now start developing apps for these devices using:
• Qt 4.7.4 / Qt Quick 1.1 introducing among other things, right-to-left and pinch area support. These things are also supported by today’s Qt Quick Components 1.1 update. More about that and other features in Sami’s blog post.
• Qt Mobility 1.2 which introduces new NFC (near field communication) and Bluetooth APIs, as well as quality improvements to the existing APIs.

Qt framework

Qt guides  See Also:
http://qt.developpez.com/defis/02-tablette-hopital/resultats/

Qt people  See Also:
• http://louis-du-verdier.developpez.com/

PyQt scientific modules

Pierre Raybaut
• http://www.projet-plume.org/files/PRaybaut.pdf

python x/y  http://www.pythonxy.com/
Python(x,y) est un logiciel libre scientifique de calcul numérique basé sur le langage Python, les interfaces graphiques Qt (et le cadre de développement associé), l’environnement de développement Eclipse et l’environnement de développement scientifique interactif Spyder.
guidata

- https://code.google.com/p/guidata/

Based on the Qt Python binding module PyQt4, guidata is a Python library generating graphical user interfaces for easy dataset editing and display.

It also provides helpers and application development tools for PyQt4.

Camelot


guiqwt

- https://code.google.com/p/guiqwt/

Based on PyQwt (plotting widgets for PyQt4 graphical user interfaces) and on the scientific modules NumPy and SciPy, guiqwt is a Python library providing efficient 2D data-plotting features (curve/image visualization and related tools) for interactive computing and signal/image processing application development.

The guiqwt library is available for Windows XP/Vista/7, GNU/Linux (official package for Debian) and MacOS X.

pyqwt

- http://pyqwt.sourceforge.net/

PyQwt is a set of Python bindings for the Qwt C++ class library which extends the Qt framework with widgets for scientific and engineering applications. It provides a widget to plot 2-dimensional data and various widgets to display and control bounded or unbounded floating point values. http://pyqwt.sourceforge.net/cli-examples.html

Spyderlib  https://code.google.com/p/spyderlib/

Spyder is part of spyderlib, a Python module based on PyQt4, pyflakes, rope and sphinx (QScintilla’s dependency has been removed in version 2.0 and rope features have been integrated since this version as well) providing powerful PyQt4 widgets like source code editors, Python console (easy to embed in your own application: see this example) or GUI-based dictionary, lists/tuples and NumPy array editors.

Qt versions

Qt 5.0.0  See Also:

- http://qt-project.org/wiki/Qt-5
- http://qt-project.org/wiki/Category:Developing_with_Qt::Qt-5

Contents

- Qt 5.0.0
  - Mission of release
  - Releases
Mission of release  Qt 5.0 will be the foundation for a new way of developing applications, where Qt Quick is in the center of Qt.

Qt 5 continues to offer all of the power of native Qt C++, and we don’t want Qt 5 to be disruptive for existing code developed for Qt 4.

Qt 5 enables highly sophisticated user experiences, offering applications the full capabilities of OpenGL/OpenGL ES graphics acceleration.

In this respect, Qt 5.0 is a feature-driven release with time-to-market requirements especially for embedded environments. This implies that we should keep the scope of Qt 5.0 limited to the essential, and add more features and add-on modules in the upcoming minor releases.

Releases

• Alpha released on April 3, 2012.
• Beta release expected by April – May.
• Final release expected by the end of June.

Qt 4.8.0  The release notes for this version are as follows:

Qt Platform Abstraction (QPA)  Qt Platform Abstraction (QPA) restructures the GUI stack to enable easier porting of Qt to different windowing systems and devices.

Threaded OpenGL  Threaded OpenGL enables rendering OpenGL from more than one thread concurrently.

HTTP requests  HTTP requests are now handled in a separate thread by default.

I/O performance  The filesystem stack received some heavy lifting under the hood. The result is better I/O performance.

The Qt project for library

Qt .pro file

The lib Template  The lib template tells qmake to generate a Makefile that will build a library. When using this template, in addition to the system variables mentioned above for the app template the VERSION variable is supported. You should use these in your .pro file to specify information about the library.

With the delivered Qt project we can test the certis2 bio API.

TODO

Configuration

http://live.gnome.org/dconf  dconf is a low-level configuration system. Its main purpose is to provide a backend to GSettings on platforms that don’t already have configuration storage systems.
http://ubuntu-news.org/2011/01/18/some-further-notes-on-qt-in-ubuntu/ Mark then goes on to outline some of the challenges (e.g. system settings), some of the solutions (e.g. Canonical are funding development from Ryan Lortie to build dconf support into Qt), and he also discusses how Qt apps should be welcome in the Ubuntu installation if they represent best-of-breed for the Free Software desktop. I couldn’t agree more.

Personally, I think this is a great step forward. I used to hack with Qt many moons ago, and while I changed to use GTK as my preferred toolkit, recent innovations in Qt (such as the incredible QML) and it’s popularity with developers, makes this not only a wise choice for app authors who want to build Qt apps on Ubuntu, but also for Ubuntu users who will have a rich set of Qt apps open to them. This doesn’t change our relationship with GNOME or GTK, it is purely an additive decision, and I think it will serve our users well.

Quickly

Contents

• Quickly
  – https://wiki.ubuntu.com/Quickly
  – Utilisation de Quickly

Quickly

See Also:

• https://wiki.ubuntu.com/Quickly
• http://doc.ubuntu-fr.org/quickly

To make Ubuntu development as sleek, simple, and powerful as possible, we have created a tool called Quickly that gets you up and running right away.

Quickly integrates project creation, code editing, GUI editing, running and debugging, as well as packaging and sharing via Launchpad.net, all in one easy to use command line interface.

Quickly generates your project ready to run with everything you need for a complete Ubuntu Application.

Quickly also comes with great tutorials and a great community of developers to help you create whatever applications you can imagine.

Quickly is not an IDE.

Rather, Quickly works by first choosing the right technologies for creating the kind of project you want to create for Ubuntu. And then it integrates those choices into the desktop and into Launchpad.net for you, all controlled with a few easy commands.

Quickly has templates to help you create a Ubuntu application, a command line application, and a game, with more templates being created all the time.

Quickly provides:

• An Easy Choice – When you start an application with Quickly, all the hard decisions have been made.
• Easy and Fun Application Generation – When you write an application with Quickly, all the boiler plate has been written for you.
• Packaging and Sharing – Quickly applications are super easy to package and distribute.
• Help – Quickly comes with tutorials and detailed help, and there is even a #quickly irc channel.

https://wiki.ubuntu.com/Quickly

Given the richness and variety of the Linux platform, programming on Linux should be easy and fun. However, it’s not easy and fun because it is too hard to learn. It is too hard to learn because there are too many choices, and too much information to wade through. Quickly strives to make programming easy and fun by bringing opinionated choices about how to write different kinds of programs to developers. Included is a Ubuntu application template for making applications that integrate smoothly into the Ubuntu software infrastructure.

Utilisation de Quickly

Rapporte 8 points si Quickly est utilisé.

See Also:

http://qt.developpez.com/defis/02-tablette-hopital/resultats

shardevelop

See Also:

• http://www.icsharpcode.net/OpenSource/SD/Default.aspx
• https://en.wikipedia.org/wiki/SharpDevelop
• http://community.sharpdevelop.net/

Figure 2.18: Logo shardevelop

SharpDevelop is a free and open source IDE for the programming languages C#, Visual Basic .NET (VB.NET), Boo, F# (starting at version 3.0), IronPython, and Clarion.

It is typically used as an alternative to Microsoft’s Visual Studio .NET

Early in its development there was a fork to Mono/Gtk# called MonoDevelop which includes cross-platform support.

shardevelop versions

shardevelop 4.1 See Also:
The following features and improvements can be found in SharpDevelop 4.1:

- NuGet
- T4
- Debugger improvements
- Web development improvements
- Text editor performance improvements

Details on the new features can be found in the Beta release announcement.

SharpDevelop 4.1 is built against .NET Framework 4.0 Full – you must install it first. Download

Please note that .NET 3.5 SP1 is required too, and the Windows SDK is strongly recommended.

SharpDevelop can take advantage of the following software if you install it:

- Microsoft StyleCop
- TortoiseSVN (Source Control)
- Microsoft F#

**Vim editor**

**See Also:**


Vim est un éditeur de texte, c’est-à-dire un logiciel permettant la manipulation de fichiers texte. Il est directement inspiré de vi (un éditeur très répandu sur les systèmes d’exploitation de type UNIX), dont il est le clone le plus populaire. Son nom signifie d’ailleurs Vi IMproved, que l’on peut traduire par **VI aMélioré**.

**Microsoft Visual studio**

**See Also:**


**Versions**

**Visual studio versions**  **See Also:**


Microsoft Visual Studio est une suite de logiciels de développement pour Windows conçue par Microsoft.

La dernière version s’appelle Visual Studio 2010.

**Visual studio 2011**  **See Also:**

Visual studio 2010  See Also:

- http://www.microsoft.com/express/vcsharp/

Cette version Dev10 est sortie en version finale le 12 avril 2010.

Cette nouvelle version utilise la version 4.0 du .NET Framework dans le logiciel ainsi qu’une refonte de l’interface graphique propulsé par WPF.

Un nouveau langage de programmation fait officiellement son apparition : F#

English  See Also:

http://en.wikipedia.org/wiki/Microsoft_Visual_Studio#Visual_Studio_2010


Visual Studio 2010 features a new UI developed using WPF.

Visual Studio 2010 IDE has been redesigned which, according to Microsoft, clears the UI organization and “reduces clutter and complexity”.

The new IDE better supports multiple document windows and floating tool windows, while offering better multi-monitor support.

The IDE shell has been rewritten using the Windows Presentation Foundation (WPF), whereas the internals have been redesigned using Managed Extensibility Framework (MEF) that offers more extensibility points than previous versions of the IDE that enabled add-ins to modify the behavior of the IDE.

Licence
COMME DÉCRIT CI-DESSOUS, EN UTILISANT CERTAINES FONCTIONS, VOUS CONSENTEZ À CE QUE MICROSOFT RECUEILLE DES DONNÉES DE DÉPANNAGE ET DE DÉPANNAGE CONNEXION. VOS DONNÉES SERONT GÉRÉES SELON LA POLITIQUE DE PRIVATE POLICY.

1. PRÉSENTATION.

a. Logiciel. Le logiciel contient les outils de développement, les programmes logiciels et la documentation.

b. Modèle de licence. Le logiciel est concédé sous licence en vertu d’une licence par utilisateur.

c. Programme Logiciel. Le logiciel contient d’autres programmes Microsoft. Les présents termes ne s’appliquent pas à ces programmes.

d. Programmes tiers. Le logiciel contient des programmes tiers. Si d’autres termes accompagnent ces programmes, votre utilisation doit se conformer à ces termes.


2. INSTALLATION ET DROITS D’UTILISATION.


b. Programmes Logiciels Microsoft Fournis. Le logiciel contient d’autres programmes Microsoft. Les présents termes ne s’appliquent pas à ces programmes.

c. Programmes tiers. Le logiciel contient des programmes tiers. Si d’autres termes accompagnent ces programmes, votre utilisation doit se conformer à ces termes.

d. Fichiers texte. Microsoft peut inclure des fichiers texte pour des services Internet.

e. Code Distribuable. Le logiciel contient du code que vous êtes autorisé à distribuer ou déployer dans vos programmes.

3. CONDITIONS DE LICENCE ET/OU DROITS D’UTILISATION SUPPLÉMENTAIRES.

a. Test des utilisateurs. Vos utilisateurs finaux peuvent accéder au logiciel pour effectuer des tests.

b. Interface utilisateur Microsoft Office. Les termes de ce contrat de licence ne vous autorisent pas à modifier la conception de l’interface utilisateur Microsoft Office.

c. Utilitaires. Le logiciel contient certains composants qui sont répertoriés dans la Liste des Utilitaires.

d. Fichiers BUILDERVER.TXT. Si votre version du logiciel contient un fichier BUILDERVER.TXT, vous êtes autorisé à copier et à distribuer le fichier.


4. Services INTERNET. Microsoft fournit des services internet avec le logiciel. Ces services peuvent être modifiés ou interrompus à tout moment.

a. Consentement pour les Services Internet. Les fonctionnalités du logiciel décrites ci-après se connecteront à Internet.

b. Consentement pour les Services Internet. Les fonctionnalités du logiciel décrites ci-après se connecteront à Internet.

c. Consentement pour les Services Internet. Les fonctionnalités du logiciel décrites ci-après se connecteront à Internet.

2.14. IDEs
Elle s’applique également, même si :

- la réparation, le remplacement ou le remboursement du logiciel ne compense pas intégralement tout préjudice subi ; ou
- les réclamations pour rupture de contrat ou violation de garantie, les réclamations en cas de responsabilité sans faute, de négligence ou autre délit dans la limite autorisée par la loi en vigueur.
- toute affaire liée au logiciel, aux services ou au contenu (y compris le code) figurant sur des sites Internet tiers ou dans des programmes tiers ; et
- les réclamations pour rupture de contrat ou violation de garantie, les réclamations en cas de responsabilité sans faute, de négligence ou autre délit dans la limite autorisée par la loi en vigueur.

Cette limitation concerne :

- toute affaire liée au logiciel, aux services ou au contenu (y compris le code) figurant sur des sites Internet tiers ou dans des programmes tiers ; et
- les réclamations pour rupture de contrat ou violation de garantie, les réclamations en cas de responsabilité sans faute, de négligence ou autre délit dans la limite autorisée par la loi en vigueur.

Elle s’applique également, même si :

- la réparation, le remplacement ou le remboursement du logiciel ne compense pas intégralement tout préjudice subi ; ou
- Microsoft avait ou aurait dû avoir connaissance de l’éventualité de tels dommages.
Certains États n’autorisent pas l’exclusion de garanties ou la limitation de responsabilité pour les dommages indirects, accessoires ou de quelque nature que ce soit. 

GARANTIE LIMITÉE
A. GARANTIE LIMITÉE. Si vous suivez les instructions, le logiciel fonctionnera, pour l’essentiel, tel que décrit dans la documentation de Microsoft reçue avec ou dans le logiciel.

Gallery

Visual studio 2010 gallery  See Also:
• http://visualstudiogallery.msdn.microsoft.com/

nuget  See Also:
• http://nuget.org/
• http://nuget.codeplex.com/
• http://docs.nuget.org/

NuGet is a Visual Studio extension that makes it easy to install and update open source libraries and tools in Visual Studio.

NuGet is a free, open source developer focused package management system for the .NET platform intent on simplifying the process of incorporating third party libraries into a .NET application during development.

NuGet is a member of the ASP.NET Gallery in the Outercurve Foundation (see the press release).

There are a large number of useful 3rd party open source libraries out there for the .NET platform, but for those not familiar with the OSS ecosystem, it can be a pain to pull these libraries into a project.

Let’s take ELMAH as an example. It’s a fine error logging utility which has no dependencies on other libraries, but is still a challenge to integrate into a project. These are the steps it takes:

• Find ELMAH
• Download the correct zip package.
• “Unblock” the package.
• Verify its hash against the one provided by the hosting environment.
• Unzip the package contents into a specific location in the solution.
• Add an assembly reference to the assembly.
• Update web.config with the correct settings which a developer needs to search for.

And this is for a library that has no dependencies. Imagine doing this for NHibernate.Linq which has multiple dependencies each needing similar steps.

We can do much better!

NuGet automates all these common and tedious tasks for a package as well as its dependencies. It removes nearly all of the challenges of incorporating a third party open source library into a project’s source tree. Of course, using that library properly is still up to the developer.

Nuget versions

nuget versions

**nuget 1.6** If you are running VS 2010 SP1, you might run into an installation error when attempting to upgrade NuGet if you have an older version installed.

The workaround is to simply uninstall NuGet and then install it from the VS Extension Gallery.

See [http://support.microsoft.com/kb/2581019](http://support.microsoft.com/kb/2581019) for more information.

Note: If Visual Studio won’t allow you to uninstall the extension (the Uninstall button is disabled), then you likely need to restart Visual Studio using “Run as Administrator.”

Features

**Support for Semantic Versioning and Prerelease Packages** NuGet 1.6 introduces support for Semantic Versioning (SemVer).

For more details on how it uses SemVer, read the Versioning documentation.

**Using NuGet Without Checking In Packages (Package Restore)** NuGet 1.6 now has first class support for the workflow in which NuGet packages are not added to source control, but instead are restored at build time if missing.

For more details, read the Using NuGet without committing packages to source control topic.

**Item Templates That Install NuGet Packages** Building on the work to support preinstalled NuGet package to Visual Studio project templates, NuGet 1.6 also adds support for Visual Studio item templates.

Item templates can have associated NuGet packages that are installed when the template in invoked.

For more details on how to change a project/item template to install NuGet packages, read the Packages in Visual Studio Templates topic.
Support for disabling package sources  When multiple package sources are configured, NuGet will look in each one for packages during installation of a package and its dependencies.

A package source that is down for some reason can severely slow down NuGet.

Prior to NuGet 1.6, you could remove the package source, but then you have to remember the details for when you want to add it back in.

NuGet 1.6 allows unchecking a package source to disable it, but keep it around.

Visual studio 2008  See Also:


Cette version (nom de code Orcas) est sortie en version finale à la fin de l’année 2007. Parmi les nouveautés de cette version1.2 :

• elle est fondée sur le .NET Framework 3.5
• des outils de métrologie relatifs au code (indicateurs du nombre de lignes, profondeur des routines, calcul de la complexité cyclomatique), relatifs à la performance (mémoire utilisée, temps d’exécution)
• gestion complète du développement collaboratif et des versions (auteurs et révisions du code) en intégrant l’outil PowerTools
• possibilité d’automatiser les processus de compilation et intégration (avec des triggers)
• meilleure gestion des tests avec possibilité d’élaborer des scénarios de test, module de test spécifique aux applications Ajax
• suppression des wizards assistant et de la bibliothèque ATL pour faire des services web en C++
• amélioration de prise en charge des fichiers XML (validation IE...).

Le numéro de version interne de Visual Studio 2008 est 9.0.

Visual c++ 2008  Visual C++ 2008 (known also as Visual C++ 9.0) was released in November 2007. This version supports .NET 3.5, and it is currently the latest stable release. Managed C++ for CLI is still available via compiler options. By default, all applications compiled against the Visual C++ 2008 Runtimes (static and dynamic linking) will only work under Windows 2000 and later.[16] A feature pack released for VC9, later included into SP1, added support for C++ TR1 library extensions.

MSVC64  Seven Steps of Migrating a Program to a 64-bit System [http://www.viva64.com/art-1-2-850243650.html](http://www.viva64.com/art-1-2-850243650.html)

How to estimate the process of 64-bit migration of C/C++ applications?  [http://www.viva64.com/art-1-2-1638146190.html](http://www.viva64.com/art-1-2-1638146190.html)

Problems of testing 64-bit applications [http://www.viva64.com/art-1-2-1289354852.html](http://www.viva64.com/art-1-2-1289354852.html)

20 issues of porting C++ code on the 64-bit platform [http://www.viva64.com/art-1-2-599168895.html](http://www.viva64.com/art-1-2-599168895.html)

Tool for check 64-bit code: [http://www.viva64.com/viva64-tool/](http://www.viva64.com/viva64-tool/)

Tools

Visual Studio Tools
pytools An integrated environment for developing Python in VS2010

See Also:

- http://pytools.codeplex.com/
- https://hg01.codeplex.com/pytools
- http://pytools.codeplex.com/documentation

Figure 2.20: pytools logo

What, Why, Who, ... ?


PTVS enables developers to use all the major productivity features of Visual Studio to build Python code using either CPython or IronPython and adds new features such as using High Performance Computing clusters to scale your code.

Together with one of the standard distros, you can turn Visual Studio into a powerful Technical Computing IDE...

Note: PTVS is not a Python distribution; it works with your existing Python/IronPython installation to provide you an integrated editing and debugging experience.

Installer

Adrian Buehlmann <adrian@cadifra.com>

See Also:

- http://pytools.codeplex.com/

pytools versions

See Also:

- http://pytools.codeplex.com/
- https://hg01.codeplex.com/pytools

i18n

MSVC i18n

See Also:

http://morpheus.developpez.com/localisationdotnet/
MinGW and cygwin

Minimalist GNU for Windows MinGW

See Also:

- http://www.mingw.org

Core MinGW

Minimalist GNU for Windows (MinGW)  See Also:

- http://www.mingw.org

Figure 2.21: GNU

Contents

- Minimalist GNU for Windows (MinGW)
  - Introduction
  - Minimal GNU for Windows (MinGW) License
  - MinGW for netbeans
  - Last MinGW version used

Introduction  MinGW (Minimalist GNU for Windows), formerly mingw32, is a native software port of the GNU Compiler Collection (GCC) to Microsoft Windows, along with a set of freely distributable import libraries and header files for the Windows API.

MinGW allows developers to create native Microsoft Windows applications. Included in MinGW are extensions to the Microsoft Visual C++ runtime library to support C99 functionality.

Note:  MinGW forked from version 1.3.3 of Cygwin. Although both Cygwin and MinGW can be used to port Unix software to Windows, they have different approaches: Cygwin aims to provide a complete POSIX layer (similar to that found in a Linux or other Unix systems) on top of Windows, sacrificing performance where necessary for compatibility. Accordingly, this approach requires Win32 programs written with Cygwin to run on top of a copylefted compatibility library that must be distributed with the program, along with the program’s source code.

10 http://sourceforge.net/projects/mingw/
MinGW aims to provide native functionality and performance via direct Windows API calls.

**Warning:** Unlike Cygwin, MinGW does not require a compatibility layer DLL and thus programs do not need to be distributed with source code.

**See Also:**
- http://www.mingw.org/wiki/MSVC_and_MinGW_DLLs

**Minimal GNU for Windows (MinGW) License**

Minimal GNU for Windows
Version 5.1.6
http://www.mingw.org/

License, Use and Redistribution

MinGW contains several different packages. Some of those packages are licensed by the GNU Public License (GPL), some are licensed in the Public Domain and some have their own versions of a license.

You may use MinGW on any number of systems. There is no restriction to your use. You may use MinGW commercially as well as privately. You the user assume the responsibility for the use of the files, binary or text, and there is no guarantee or warranty, expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. You assume all responsibility and agree to hold no entity, copyright holder or distributors liable for any loss of data or inaccurate representations of data as a result of using MinGW.

You may redistribute MinGW in part or in whole as long as you follow the guidelines of redistribution of each license contained within. To be certain that you are being legally compliant, always distribute the source. Distribution of source is your responsibility should you decide to redistribute MinGW. If you distribute MinGW via a web site then you must put a copy of the source for that version of MinGW on your web site as well. If you distribute MinGW via removable media then you must distribute that version of MinGW source with that same type of removable media.

Binaries created from the use of MinGW and of MSYS are not bound by any license found within this package unless you use a library that is itself covered by the GPL license. If you wish to create proprietary software then don’t use libiberty.a or any other GPL licensed library. A library licensed with LGPL (Lesser GPL) may be used by proprietary software without GPL infection as special permission within the LGPL has given you this right.

Please read and reread the COPYING and COPYING.LIB found in the <prefix>/doc/mingw directory.

Earnie.
Earnie@users.sf.net

**MinGW for netbeans** MinGW tools in netbeans.
Building an application (.exe)

Last MinGW version used

<table>
<thead>
<tr>
<th>Fichier</th>
<th>MD5</th>
<th>SHA1</th>
<th>Size in bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MinGW-5.1.6.exe</td>
<td>9cf4ab0b4c9f858d32f5d5c89009c4db399983b03115c9f9e8121dd212799fc6c154</td>
<td>4438842</td>
<td></td>
</tr>
</tbody>
</table>

See Also:
- GCC (GNU Compiler Collection)
- MinGW Resource Compiler (windres)

Msy

Installing MSYS/MinGW  The combination of MinGW and MSYS provides a small, self-contained environment that can be loaded onto removable media without leaving entries in the registry or files on the computer. By providing more functionality, Cygwin becomes more complicated to install and maintain.

See Also:
- http://mingw.org/node/18

Last msys version used

February 8th 2010  Check http://sourceforge.net/projects/mingw/files/ for more recent versions of all these files.

<table>
<thead>
<tr>
<th>Fichier</th>
<th>MD5</th>
<th>SHA1</th>
<th>Size in bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSYS-1.0.11.exe</td>
<td>cf95067cc749b00bf5b81deb40a8e8</td>
<td>6bedad9d1b67b48c92781b3ccddbd1cc9ee64c47</td>
<td>78283940</td>
</tr>
</tbody>
</table>

http://downloads.sourceforge.net/mingw/MSYS-1.0.11.exe

windres minGW

MinGW windres resource manager

MinGW Resource Compiler (windres)  See Also:
- http://oldwiki.mingw.org/index.php/MS%20resource%20compiler
- http://www.mingw.org/wiki/MS_resource_compiler
The MinGW32 compilers come with a fairly decent resource compiler for Windows resources. There is no visual editor though, so you will have to your own resource scripts and have some other means to create ICO files and the like.

The main difference between using the MS resource tools and the GNU tools is that MS rc generates a ".res" file in a special binary resource format, which can be passed directly to MS link, while the GNU linker ld only supports resources in ".o" (same as ".obj") format (although windres can output in both formats).

Therefore, to convert commands like this:

```
rc foo.rc
link -out:foo.exe foo.obj foo.res
```

You need something like the following for the GNU tools:

```
windres foo.rc foores.o
gcc -o foo.exe foo.o foores.o
```

windres -h

Usage: windres [options] [fichier-d-entrée] [fichier-de-sortie]
The options are:
- i --input=<file> Name input file
- o --output=<file> Name output file
- J --input-format=<format> Specify input format
- O --output-format=<format> Specify output format
- F --target=<target> Specify COFF target
--preprocessor=<program> Program to use to preprocess rc file
- I --include-dir=<dir> Include directory when preprocessing rc file
- D --define <sym>[=<val>] Define SYM when preprocessing rc file
- U --undefine <sym> Undefine SYM when preprocessing rc file
- v --verbose Verbose - tells you what it’s doing
- c --codepage=<codepage> Specify default codepage
- l --language=<val> Set language when reading rc file
--use-temp-file Use a temporary file instead of popen to read
the preprocessor output
--no-use-temp-file Use popen (default)

Les options sont:
- r ignoré pour la compatibilité avec rc
@<fichier> lire les options à partir du <fichier>
- h --help afficher l’aide-mémoire
- V --version afficher le nom et la version du logiciel

FORMAT est soit rc, res, ou coff, et est déduit à partir l’extension
du nom de fichier, si non spécifié. Un nom simple de fichier comme fichier d’entrée
L’entrée par défaut se fait à partir de stdin, par défaut pour rc.
La sortie par défaut est stdout, par défaut pour rc.
windres: cibles supportées: pe-i386 pei-i386 elf32-i386 elf32-little elf32-big srec
symbolsrec verilog tekhex binary ihex
Rapporter toutes anomalies à <http://www.sourceforge.org/bugzilla/>

See Also:
windres -h

C:\MinGW\bin>windres -V
GNU windres (GNU Binutils) 2.20
Copyright 2009 Free Software Foundation, Inc.
This program is free software; you may redistribute it under the terms of
the GNU General Public License version 3 or (at your option) any later version.
This program has absolutely no warranty.

windres examples

If you have resources from a resource file (.rc) that also need to be
added to your executable, you’ll need to compile the resource file as
well as your other source files and include the compiled resources
when linking to create the executable. Here’s an example that shows
how to compile and link in a resource file named resfile.rc.

windres -o resfile.o resfile.rc
gcc -o hello hello.o resfile.o -mwindows

See Also:
• http://blog.stranadurakov.com/tag/windres/
• http://www.cs.colorado.edu/~main/cs1300/doc/mingwfaq.html

windres syntax  See Also:

• http://www.cygwin.com/cygwin-ug-net/windres.html

What follows is a quick-reference to the syntax windres supports.

id ACCELERATORS suboptions
BEG
"^C" 12
"Q" 12
65 12
65 12 , VIRTKEY ASCII NOINVERT SHIFT CONTROL ALT
(12 is an acc_id)
END

SHIFT, CONTROL, ALT require VIRTKEY

id BITMAP memflags "filename"
memflags defaults to MOVEABLE

id CURSOR memflags "filename"
memflags defaults to MOVEABLE,DISCARDABLE

id DIALOG memflags exstyle x,y,width,height styles BEG controls END
id DIALOGEX memflags exstyle x,y,width,height styles BEG controls END
id DIALOGEX memflags exstyle x,y,width,height,helpid styles BEG controls END
memflags defaults to MOVEABLE
exstyle may be EXSTYLE=number
styles: CAPTION "string"
CLASS id
    STYLE FOO | NOT FOO | (12)
    EXSTYLE number
    FONT number, "name"
    FONT number, "name", weight, italic
    MENU id
    CHARACTERISTICS number
    LANGUAGE number, number
    VERSIONK number
controls:
    AUTO3STATE params
    AUTOCHECKBOX params
    AUTORADIOBUTTON params
    BEDIT params
    CHECKBOX params
    COMBOBOX params
    CONTROL ["name",] id, class, style, x,y,w,h [,exstyle] [data]
    CONTROL ["name",] id, class, style, x,y,w,h, exstyle, helpid [data]
    CTEXT params
    DEFPUSHBUTTON params
    EDITTEXT params
    GROUPBOX params
    HEDIT params
    ICON ["name",] id, x,y [data]
    ICON ["name",] id, x,y,w,h, style, exstyle [data]
    ICON ["name",] id, x,y,w,h, style, exstyle, helpid [data]
    IEDIT params
    LISTBOX params
    LTEXT params
    PUSHBOX params
    PUSHBUTTON params
    RADIOBUTTON params
    RTEXT params
    SCROLLBAR params
    STATE3 params
    USERBUTTON "string", id, x,y,w,h, style, exstyle
    params:
        ["name",] id, x, y, w, h, [data]
        ["name",] id, x, y, w, h, style [,exstyle] [data]
        ["name",] id, x, y, w, h, style, exstyle, helpid [data]
    [data] is optional BEG (string|number) [,(string|number)] (etc) END

id FONT memflags "filename"
memflags defaults to MOVEABLE|DISCARDABLE

id ICON memflags "filename"
memflags defaults to MOVEABLE|DISCARDABLE

LANGUAGE num, num

id MENU options BEG items END
items:
    "string", id, flags
SEPARATOR
POPUP "string" flags BEG menuitems END
flags:
CHECKED
GRAYED
HELP
INACTIVE
MENUBARBREAK
MENUBREAK

id MENUEX suboptions BEG items END
items:
  MENUITEM "string"
  MENUITEM "string", id
  MENUITEM "string", id, type [,state]
  POPUP "string" BEG items END
  POPUP "string", id BEG items END
  POPUP "string", id, type BEG items END
  POPUP "string", id, type, state [,helpid] BEG items END

id MESSAGETABLE memflags "filename"
memflags defaults to MOVEABLE

id RCDATA suboptions BEG (string|number) [, (string|number)] (etc) END

STRINGTABLE suboptions BEG strings END
strings:
  id "string"
  id, "string"

(User data)
id id suboptions BEG (string|number) [, (string|number)] (etc) END

id VERSIONINFO stuffs BEG verblocks END
stuffs: FILEVERSION num, num, num, num
  PRODUCTVERSION num, num, num, num
  FILEFLAGSMASK num
  FILEOS num
  FILETYPE num
 FILESUBTYPE num
verblocks:
  BLOCK "StringFileInfo" BEG BLOCK BEG vervals END END
  BLOCK "VarFileInfo" BEG BLOCK BEG vertrans END END
vervals: VALUE "foo", "bar"
vertrans: VALUE num, num

suboptions:
  memflags
  CHARACTERISTICS num
  LANGUAGE num, num
  VERSIONK num

memflags are MOVEABLE/FIXED PURE/IMPURE PRELOAD/LOADONCALL DISCARDABLE

id3Image windows resource file example (resfile.rc)
2.15 Installation

2.15.1 Installation programs

See Also:
http://en.wikipedia.org/wiki/Installation_%28computer_programs%29

Cygwin commands

cygwin tools used by netbeans:

- C:\cygwin\bin\rm.exe
- C:\cygwin\bin\gdb.exe
0Install

Introduction

The antidote to app-stores.

Imagine a world where:

- anyone can distribute software: If you have a web-site, you can distribute your software; Create one package that works everywhere; With dependency handling and automatic updates; There is no central point of control
- security is central: Users don’t need administrator access just to install a word-processor; Digital signatures are always checked before new software is run; Users share libraries without having to trust each other; Use of sandboxes and virtualisation is routine
- you control your own computer: You don’t have to guess what happens during installation; Mix and match stable and experimental programs on a single system

coApp

See Also:

- http://coapp.org/index.html

See Also:

- http://0install.net/

2.15. Installation
People

Garrett Serack  See Also:
http://fosdem.org/2012/schedule/speaker/garrett_serack

Garrett Serack worked as an independent software development consultant in Calgary, Canada, for 15 years, with clients in fields such as government, telecom, petroleum, and railways.

Joining Microsoft in the fall of 2005 as the Community Program Manager of the Federated Identity team, Garrett has worked with the companies and the Open Source community to build digital identity frameworks, tools, and standards that are shaping the future of Internet commerce and strengthening the fight against fraud.

In the summer of 2007, he transitioned to Open Source Technology Center at Microsoft where he now works as a Senior Open Source Software Development Engineer working closely with Open Source communities to improve the quality and performance of their software on the Windows Platform.

Garrett currently spends most of his time working on CoApp—an open source Package Management system for Windows, and has started a number of Open Source projects along with working as a committer on several other projects.

Conferences

See Also:
http://fosdem.org/2012/schedule/event/coapp

CoApp: Packaging Open Source software for Windows

Open Source Software has long enjoyed the ability to be trivially acquired, installed and maintained on Linux via packages, where each package contains within it all the information required to find and configure the necessary dependencies.

However, in many situations there is a strong need to install OSS on Windows, where traditional package management systems like RPM and DPKG are not only unavailable—but couldn’t support the platform in a manner consistent with the other software.

This presentation looks into the technical details of CoApp, a new fully-open source package management system fills this gap by providing all the tools to easily create, publish, discover, and install software packages, including automatically handling dependencies, managing updates and providing a frictionless end-user experience, all in a method that is well supported by the platform.

InstallJammer installer

InstallJammer is a multiplatform GUI installer designed to be completely cross-platform and function on Windows and most all versions of UNIX with support for MacOS X coming soon.

InstallJammer features a very powerful install builder with support for multiple themes and a high level of configurability for installers. Installers are built as single executable files for easy distribution over the web and handle installing everything you need for your application in a simple, cross-platform way.

InstallJammer is written almost entirely in Tcl/Tk with a little bit of help from some extensions written in C. Since all scripting in InstallJammer is done through Tcl/Tk, it might be helpful for you to learn a little about Tcl if you plan on doing any kind of complex operations with your installer. Below are some links to get you started.

See Also:

•  http://www.tcl.tk/about/compare.html
InstallJammer about

About InstallJammer is a multiplatform GUI installer and builder designed to be completely cross-platform and function on Windows and most all version of UNIX with eventual support for Mac OS X.

Features

- A full-featured install builder.
- Installs are packaged in single binary executables for each platform making web distribution easy.
- Support for multiple install themes.
- Default install themes that resemble popular, commercial installers.
- A high level of configurability.
- Built-in support for Windows install conventions.
- Automatic creation of an uninstaller.
- Easily extended to new platforms.

Installing You’re already done. InstallJammer only needs to be unpacked, and it’s ready to go. If you have binaries available for your platform, you can type: ./installjammer to get going.

Availability

Home Page http://www.installjammer.com/
Downloads http://www.installjammer.com/download/

Innosetup

Introduction

Inno Setup is a free, feature-packed installation builder. The application’s features include a Windows 2000-style wizard interface; the ability to create a single EXE for easy, online distribution; support for disk spanning; and full uninstall capabilities. The program also includes customizable setup types, integrated file compression, support for installing shared files and OCXs, and the creation of Start menu icons, INI entries, and registry entries.

Le logiciel InnoSetup est sous http://inno-setup.en.softonic.com/

- http://www.jrsoftware.org/ispphelp/
- http://www.jrsoftware.org/ishelp/
- http://www.jrsoftware.org/
- dernière version utilisée: 5.3.8
Inno Setup License

Except where otherwise noted, all of the documentation and software included in the Inno Setup package is copyrighted by Jordan Russell.

Copyright (C) 1997-2010 Jordan Russell. All rights reserved. Portions Copyright (C) 2000-2010 Martijn Laan. All rights reserved.

This software is provided “as-is,” without any express or implied warranty. In no event shall the author be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter and redistribute it, provided that the following conditions are met:

1. All redistributions of source code files must retain all copyright notices that are currently in place, and this list of conditions without modification.
2. All redistributions in binary form must retain all occurrences of the above copyright notice and web site addresses that are currently in place (for example, in the About boxes).
3. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software to distribute a product, an acknowledgment in the product documentation would be appreciated but is not required.
4. Modified versions in source or binary form must be plainly marked as such, and must not be misrepresented as being the original software.

Jordan Russell jr-2010 AT jrsoftware.org http://www.jrsoftware.org/

“Inno Setup” options

Les fenêtres d’installation se présentent de cette façon: Setup - “Nom de l’application”.

Les commutateurs les plus utilisé pour les applications utilisant la technologie “Inno Setup” sont :

• SP- : désactive le message d’avertissement du début vous demandant si vous souhaitez réellement installer l’application.
• /SILENT : seule la barre de progression est montrée.
• /VERYSILENT : l’installation se fera de manière complètement silencieuse et sans même que la barre de progression soit visible.
• /NORESTART : demande à ce que l’ordinateur ne soit pas redémarré sauf si cela est absolument nécessaire.
• /SAVEINF="Nom_Fichier" : demande au programme d’installation de sauvegarder les paramètres d’installation dans le fichier spécifié. Cela vous permet d’automatiser les installations.
• /LOADINF="Nom_Fichier" : demande au programme de charger les paramètres d’installation à partir du fichier spécifié.
• /DIR="x:Nom_Répertoire” : permet de remplacer le répertoire d’installation défini par défaut par celui que vous aurez spécifié.
• /GROUP="Nom_Répertoire” : permet de remplacer le répertoire d’installation du menu Démarrer défini par défaut par celui que vous aurez spécifié.

D’autres possibilités sont expliquées sur ce site : http://www.jrsoftware.org
Innosetup example (.iss file)

install_windows_usbccid_driver

; -- install_windows_ccid_driver.iss --
; install a ccid driver depending on:
; - the architectures (x86, x64, ia64)
; - the windows operating system (Windows XP, Vista, 7)

[Setup]
AppCopyright=Copyright 2009-2010 Orcanthus
AppName=Windows USB CCID Driver Install
AppVerName=USBCCID Driver Install
OutputDir=innosetup
DefaultDirName=c:\Tmp
DefaultGroupName=USBCCID_Driver
UninstallDisplayIcon={app}\USBCCID_Driver.exe
Compression=lzma
SolidCompression=yes
DisableStartupPrompt=yes

; OutputDir=userdocs:Inno Setup Examples Output
; "ArchitecturesInstallIn64BitMode=x64 ia64" requests that the install
; be done in "64-bit mode" on x64 & Itanium, meaning it should use the
; native 64-bit Program Files directory and the 64-bit view of the
; registry. On all other architectures it will install in "32-bit mode".
ArchitecturesInstallIn64BitMode=x64 ia64

[Files]
; Install the usbccid driver depending on the architectures ans windows operating systems
Source: x86\Livraison\XP\*.*; DestDir: "\{app}\"; Check: IsOtherArch and IsWindowsXP
Source: x86\Livraison\Vista_7\*.*; DestDir: "\{app}\"; Check: IsOtherArch and IsWindowsVista_or_7
Source: x64\Livraison\XP\*.*; DestDir: "\{app}\"; Check: IsX64 and IsWindowsXP
Source: x64\Livraison\Vista_7\*.*; DestDir: "\{app}\"; Check: IsX64 and IsWindowsVista_or_7
; Source: "MyProg.chm"; DestDir: "\{app}\"
; Source: "README.txt"; DestDir: "\{app}\"; Flags: isreadme

[Icons]
Name: \{group}\Uninstall USBCCID_Driver; Filename: \{uninstall.exe\}

[Run]
Filename: "\{app\}\install_usbccid_driver.bat"; Parameters: "\{app\}"; StatusMsg: "Installing USB CCID driver ...

[Code]
function IsX64: Boolean;
begin
  Result := Is64BitInstallMode and (ProcessorArchitecture = paX64);
end;

function IsIA64: Boolean;
begin
  Result := Is64BitInstallMode and (ProcessorArchitecture = paIA64);
end;
function IsOtherArch: Boolean;
begin
  Result := not IsX64 and not IsIA64;
end;

function IsWindowsXP: Boolean;
var
  Version: TWindowsVersion;
  S: String;
begin
  GetWindowsVersionEx(Version);
  Result := Version.NTPlatform and (Version.Major = 5) and (Version.Minor = 1);
end;

function IsWindowsVista: Boolean;
var
  Version: TWindowsVersion;
  S: String;
begin
  GetWindowsVersionEx(Version);
  Result := Version.NTPlatform and (Version.Major= 6) and (Version.Minor = 0);
end;

function IsWindows7: Boolean;
var
  Version: TWindowsVersion;
  S: String;
begin
  GetWindowsVersionEx(Version);
  Result := Version.NTPlatform and (Version.Major = 6) and (Version.Minor = 1);
end;

function IsWindowsVista_or_7: Boolean;
var
  Version: TWindowsVersion;
  S: String;
begin
  GetWindowsVersionEx(Version);
  Result := Version.NTPlatform and (Version.Major = 6) and ((Version.Minor = 1) or (Version.Minor = 0));
end;

py2nsis

See Also:
- https://code.google.com/p/py2nsis/
- https://codescience.wordpress.com/2011/01/01/10

py2Nsis is a Simple to Use, Nsis installers generator for python projects.
It also make the setup.py for py2exe.

Nsis is a professional open source system to create Windows installers.

py2exe is a Python Distutils extension which converts Python scripts into executable Windows programs, able to run without requiring a Python installation.

py2Nsis combines the capabilities of these ones and generates directly a Windows installer with just a few information about your python code!

The installer for this project was generated with py2nsis itself.

Here’s a tutorial about py2nsis: https://codescience.wordpress.com/2011/01/01/10

Wise (Altiris, symantec)

See Also:

Introduction

Wise Solutions, Inc. is an American company that makes software tools for creating application installers. Their primary product, Wise was one of the most widely used installation packages for Windows. Their main competitor is InstallShield by Flexera Software. Wise Solutions was acquired and became a wholly owned subsidiary of Altiris, Inc. in December 2003.

In 2007, Altiris has been acquired by Symantec Corporation.

Wise Solutions was started as a shareware tool originally distributed via CompuServe. Over time the company grew to roughly 300 employees spread across the headquarters in Plymouth, Michigan and a European office in The Hague, Netherlands. After the 2003 acquisition by Altiris the European office was consolidated with an existing European Altiris office. The Plymouth Michigan office is now primarily a regional software development and support site for existing Wise branded products and some Altiris branded products.
Comment passer d’une version à l’autre

Conservant l’upgrade code:

- `{3795FA12-513E-4129-98AB-18D212FE92A2}` pour CL1356A
- `{81BA2998-64AF-4189-888E-98E927647AF3}` pour CL1356T/T5

puis incrémenter le product code.

Comme on peut le voir sur l’image ci-dessus, il faut cliquer sur ‘Non’.

Wix

See Also:


Introduction

The Windows Installer XML (WiX) is a toolset that builds Windows installation packages from XML source code.
The toolset supports a command line environment that developers may integrate into their build processes to build MSI and MSM setup packages

**Download**

Wix35.msi  See Also:
http://wix.codeplex.com/releases/view/60102#DownloadId=204417

### 2.16 Internationalization (i18n)

#### 2.16.1 Internationalization (i18n)

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Unicode
- https://secure.wikimedia.org/wikipedia/fr/wiki/Internationalisation_de_logiciel
- http://www.python.org/dev/peps/pep-0393/

In computing, internationalization and localization (also spelled internationalisation and localisation, see spelling differences) are means of adapting computer software to:

- different languages
- regional differences.

**Internationalization** is the process of designing a software application so that it can be adapted to various languages and regions without engineering changes.

**Localization** is the process of adapting internationalized software for a specific region or language by adding locale-specific components and translating text.

The terms are frequently abbreviated to the numeronyms i18n (where 18 stands for the number of letters between the first i and last n in internationalization, a usage coined at DEC in the 1970s or 80s[1]) and L10n respectively, due to the length of the words. The capital L in L10n helps to distinguish it from the lowercase i in i18n.

Some companies, like Microsoft, IBM and Sun Microsystems, use the term “globalization” for the combination of internationalization and localization. Globalization can also be abbreviated to g11n

https://secure.wikimedia.org/wikipedia/fr/wiki/Internationalisation_de_logiciel

L’internationalisation d’un logiciel consiste à préparer son adaptation à des langues et des cultures différentes.

Contrairement à la régionalisation, qui nécessite surtout des compétences en langues, l’internationalisation est un travail essentiellement technique, mené par des programmeurs.

**Links**

- http://www.cl.cam.ac.uk/~mgk25/unicode.html
- http://french.joelonsoftware.com/Articles/Unicode.html
List of ISO 639-1 codes

ISO 639 is a standardized nomenclature used to classify all known languages. Each language is assigned a 2-letter (639-1) and 3-letter (639-2 and 639-3), lowercase abbreviation, amended in later versions of the nomenclature. The system is highly useful for linguists and ethnographers to categorize the languages spoken on a regional basis, and to compute analysis in the field of lexicostatistics.

Best of Practice 47 (BCP_47)

IETF language tags are abbreviated language codes; for examples, “en” for English or “en-CA” for Canadian English. They are defined by the BCP_47 standard track, which is currently composed of normative RFC 5646 and RFC 4647 (also referencing the related informational RFC 5645), along with the normative content of the IANA Language Subtag Registry regulated by these RFCs.

Relations to ISO 639-3 (individual languages and macro-languages) and some parts of ISO 639-1

The obsoleted RFC 4646 (as well as its current successor RFC 5646), unlike its predecessors, defined the concept of an “extended language subtag”, although it still did not permit the registration of such subtags.

However, in the newer RFC 5645 and RFC 5646, all individual languages and macro-languages of ISO 639-3 were finally registered as (primary) language subtags, with a new language matching algorithm that allows a resource whose localization is missing in an individual language to be looked for in its macro-language, whose code is now present in the IANA database along with other classification information coming from ISO 639-3 (and also ISO 639-5 for language families).

See Also:

unicode

See Also:
http://docs.python.org/howto/unicode.html
http://www.haypocalc.com/wiki/Python_Unicode
http://sebsauvage.net/python/charsets_et_encoding.html
UTF-8 UCS transformation format 8 bits

Introduction

UTF-8 (UCS transformation format 8 bits) est un moyen de coder les caractères Unicode sous forme de séquences de un à quatre octets.

La norme Unicode définit entre autres un ensemble (ou répertoire) de caractères. Chaque caractère est repéré dans cet ensemble par un index entier aussi appelé « point de code ».

Par exemple le caractère “€” (euro) est le 8365ème caractère du répertoire Unicode, son index, ou point de code est donc 8364 (on commence à compter à partir de 0).

Le répertoire Unicode peut contenir plus d’un million de caractères, les points de code sont compris entre 0 et 0x10FFFF ce qui est bien trop grand pour tenir dans un seul octet (limité à des valeurs entre 0 et 255).

La norme Unicode définit donc des méthodes standardisées pour coder et stocker cet index sous forme de séquence d’octets : UTF-8 est l’une d’entre elles, avec UTF-16, UTF32 et leurs différentes variantes.

La principale caractéristique d’UTF-8 est qu’elle est rétro-compatible avec la norme ASCII, c’est-à-dire que tout caractère ASCII se code en UTF-8 sous forme d’un unique octet, identique au code ASCII.

Par exemple A (A majuscule) a pour code ASCII 65 et se code en UTF-8 par l’octet 65.

Chaque caractère dont le point de code est supérieur à 127 (caractère non ASCII) se code sur 2 à 4 octets.

Le caractère “€” (euro) se code par exemple sur 3 octets : 0xE2, 0x82, 0xAC (ou 226, 130, 172 en déc

unicode 6.0

See Also:

- http://www.unicode.org/versions/Unicode6.0.0/
- http://www.w3.org/blog/International/2011/02/22/unicode_version_6_0_complete_text_of_cor

The Unicode 6.0 core specification includes information on scripts newly encoded in Unicode 6.0, as well as many updates and clarifications to other sections of the text.

The release of the core specification completes the definitive documentation of the Unicode Standard, Version 6.0.

In Version 6.0, the standard grew by 2,088 characters. Over 1,000 of these characters are symbols used for text exchange on mobile phones.

The Unicode Standard now also includes the recently created official symbol for the Indian rupee. After computers and mobile phones update to Version 6.0, the rupee sign will be available for use like the $ or £ now.
In addition, this version adds many CJK Unified Ideographs in common use in China, Taiwan, and Japan, as well as characters for African language support, including extensions to the Tifinagh, Ethiopic, and Bamum scripts.

Three scripts are supported for the first time: Mandaic, Batak, and Brahmi.

In October of 2010, the other portions of Unicode 6.0 were released: the Unicode Standard Annexes, code charts, and the Unicode Character Database. This allowed vendors to update their implementations of Unicode 6.0 as quickly as possible.

For more information on all of The Unicode Standard, Version 6.0, see http://www.unicode.org/versions/Unicode6.0.0/

http://www.unicode.org/versions/Unicode6.0.0/ Unicode 6.0.0 is a major version of the Unicode Standard and supersedes all previous versions. This page summarizes the important changes for the Unicode Standard, Version 6.0.0. In the discussion below, shortened references to “Unicode 6.0” or “Version 6.0” specifically refer to Version 6.0.0.

Summary The Unicode Standard, Version 6.0 is the first major version of the Unicode Standard to be published solely in online format.

unicode 5.0 See Also:

• http://www.unicode.org/versions/Unicode5.0.0/

Version 5.0.0 has been superseded by the latest version of the Unicode Standard.

unicode CLDR (Common Locale Data Repository) See Also:

• http://cldr.unicode.org/
• https://secure.wikimedia.org/wikipedia/fr/wiki/CLDR
• https://secure.wikimedia.org/wikipedia/en/wiki/Common_Locale_Data_Repository

The Unicode CLDR provides key building blocks for software to support the world’s languages, with the largest and most extensive standard repository of locale data available. This data is used by a wide spectrum of companies for their software internationalization and localization, adapting software to the conventions of different languages for such common software tasks as:

• formatting of dates, times, and time zones,
• formatting numbers and currency values
• sorting text
• choosing languages or countries by name

Python cldr_countries

• https://github.com/matthiask/cldr_countries

unicode python See Also:

• http://www.haypocalc.com/wiki/Python_Unicode
This choice is a big portability issue. Mac OS X, most Linux distro, BSD systems use UTF-8 local encoding, whereas Windows use legacy code pages like cp1252 (something like ISO-8859-1) or cp952 (shift jis).

But sometimes, the locale is “C” (e.g. on our buildbots) and programs start to fail with Unicode errors...

I see two options to improve the situation

(1) **hard way** change open() API to make encoding a mandatory argument.

**Problem** it breaks compatibility with Python 3.0, 3.1 and 3.2 (ooops!); the encoding argument is the 4th argument, you have to use a keyword or choose a value for the buffering argument.

I proposed to change open() API in Python 3.1 to make all arguments -except the filename and the mode- keyword-only arguments, but Guido rejected my idea:

"Remember, for 3.0 we’re trying to get a release out of the door, not cram in new features, no matter how small."

http://bugs.python.org/issue4121

(2) **soft way** add a warning if the encoding is implicit (locale encoding).

I don’t know what is the best warning type, and if it should be always displayed, only once, or not by default.

Even if it is hidden by default, a careful developer will be able to use -Werror to fix bugs...

I suspect that most tests fail if open() raises an exception if the encoding is not specified (e.g. see distutils/packaging issues about the file encoding).

Victor

**Python tools**  See Also:

http://pypi.python.org/pypi/Unidecode

**unicode tutorials**  See Also:

- http://www.shapecatcher.com/
i18n projects

icu (International Components for Unicode)  See Also:
   • http://site.icu-project.org/
   • http://site.icu-project.org/#TOC-Who-Uses-ICU-

ICU is a mature, widely used set of C/C++ and Java libraries providing Unicode and Globalization support for software applications. ICU is widely portable and gives applications the same results on all platforms and between C/C++ and Java software.

pyICU  See Also:
   • http://pyicu.osafoundation.org/
   • http://pypi.python.org/pypi/PyICU/

Welcome to PyICU, a Python extension wrapping IBM’s International Components for Unicode C++ library (ICU). PyICU is a project maintained by the Open Source Applications Foundation.

multilingualweb  See Also:
http://www.multilingualweb.eu/

rushed conversion tool  See Also:
   • http://rishida.net/tools/conversion/
   • http://rishida.net/blog/?p=547

Exemple: http://rishida.net/tools/conversion/?q=CrU%200EApes

http://rishida.net/blog/?p=547  It has always been possible to pass a string to the converter in the URI, but that was never documented.

Now it is, and you can pass a string using the q parameter. For example, http://rishida.net/tools/conversion/?q=Crêpes. You can also pass a string with escapes in it, but you will need to be especially careful to percent escape characters such as &, + and # which affect the URI syntax. For example, http://rishida.net/tools/conversion/?q=CrU%2B00EApes.

Django i18n projects  See Also:
   • http://docs.djangoproject.com/en/dev/topics/i18n/

Overview  Django has full support for internationalization of text in code and templates, and format localization of dates and numbers. Here’s how it works.

Essentially, Django does two things:
   • It allows developers and template authors to specify which parts of their apps should be translatable.
   • It uses these hooks to translate Web apps for particular users according to their language preferences.
The complete process can be seen as divided in three stages. It is also possible to identify an identical number of roles with very well defined responsibilities associated with each of these tasks (although it’s perfectly normal if you find yourself performing more than one of these roles):

- For application authors wishing to make sure their Django apps can be used in different locales: Internationalization.
- For translators wanting to translate Django apps: Localization.
- For system administrators/final users setting up internationalized apps or developers integrating third party apps: Deployment of translations.

For more general information about the topic, see the GNU gettext documentation and the Wikipedia article.

**Django transmeta** See Also:

**Introduction** Transmeta is an application for translatable content in Django’s models. Each language is stored and managed automatically in a different column at database level.

**Features**
- Automatic schema creation with translatable fields.
- Translatable fields integrated into Django’s admin interface.
- Command to synchronize database schema to add new translatable fields and new languages.

**Python i18n modules** See Also:
- [http://www.catonmat.net/blog/python-library-for-google-translate/](http://www.catonmat.net/blog/python-library-for-google-translate/)
- [http://www.python.org/dev/peps/pep-0393/](http://www.python.org/dev/peps/pep-0393/)

**Babel** [http://babel.edgewall.org/](http://babel.edgewall.org/)
A collection of tools for internationalizing Python applications

Babel is composed of two major parts:
- tools to build and work with gettext message catalogs
- a Python interface to the CLDR (Common Locale Data Repository), providing access to various locale display names, localized number and date formatting, etc.

**Auto-translator** See Also:
Python google translate (xgoogle)  See Also:

- https://github.com/pkrumins/xgoogle
- http://www.catonmat.net/blog/python-library-for-google-translate/

The new module is called “xgoogle.translate” and it implements two classes

- “Translator”
- “LanguageDetector”.

The “Translator” class can be used to translate text. It provides a function called “translate” that takes three arguments - “message”, “lang_from” and “lang_to”.

It returns the translated text as a Unicode string. Don’t forget to encode it to the right encoding before outputting, otherwise you’ll get errors such as “UnicodeEncodeError: ‘latin-1’ codec can’t encode characters in position 0-3: ordinal not in range(256)”

Readme  This is a Google library called ‘xgoogle’. Current version is 1.3.

It’s written by Peteris Krumins (peter@catonmat.net).

His blog is at http://www.catonmat.net – good coders code, great reuse.

License  The code is licensed under MIT license.

Contains  At the moment it contains:

- Google Search module xgoogle/search.py  http://www.catonmat.net/blog/python-library-for-google-search/
- Google Sponsored Links Search module xgoogle/sponsoredlinks.py  http://www.catonmat.net/blog/python-library-for-google-sponsored-links-search/
- Google Sets module xgoogle/googlesets.py  http://www.catonmat.net/blog/python-library-for-google-sets/
- Google Translate module xgoogle/translate.py  http://www.catonmat.net/blog/python-library-for-google-translate/

Download  

- https://github.com/pkrumins/xgoogle

Example  Here is an example usage of Google Translate module:

```python
>>> from xgoogle.translate import Translator

>>> translate = Translator().translate

>>> print translate("Mani sauc Pteris", lang_to="ru").encode('utf-8')

>>> print translate("Mani sauc Pteris", lang_to="en")
My name is Peter

>>> print translate(" ")
My name is Peter
```
The “translate” function takes three arguments - “message”, “lang_from” and “lang_to”. If “lang_from” is not given, Google’s translation service auto-detects it. If “lang_to” is not given, it defaults to “en” (English).

In case of an error the “translate” function throws “TranslationError” exception. Make sure to wrap your code in try/except block to catch it:

```python
>>> from xgoogle.translate import Translator, TranslationError
>>> try:
>>>     translate = Translator().translate
>>>     print translate"
>>> except TranslationError, e:
>>>     print e
Failed translating: invalid text
```

The Google Translate module also provides “LanguageDetector” class that can be used to detect the language of the text.

Here is an example usage of LanguageDetector:

```python
>>> from xgoogle.translate import LanguageDetector, DetectionError
>>> detect = LanguageDetector().detect
>>> english = detect("This is a wonderful library.")
>>> english.lang_code
'en'
>>> english.lang
'English'
>>> english.confidence
0.28078437000000001
>>> english.is_reliable
True
```

The “DetectionError” may get raised if the detection failed.

**Version history**

**v1.0** initial release, xgoogle library contains just the Google Search.

**v1.1** added Google Sponsored Links Search. fixed a bug in browser.py that might have thrown an unexpected exception.

**v1.2** added Google Sets module

**v1.3** added Google Translate module fixed a bug in browser.py when KeyboardInterrupt did not get propagated.

**Qt linguist** See Also:

*Qt i18n*

**i18n news**


- http://usb.org/developers/docs/USB_LANGIDs.pdf
On Aug 18, 2010, at 9:01 PM, Peter Stuge wrote:

> Alan Ott wrote:
> >>>> mapping from locale names (like en_US) to USB language IDs
> >>>> (eg: 0x0409).
> >>>
> >>> http://usb.org/developers/docs/USB_LANGIDs.pdf
> >>
> >> I assume you’re suggesting that I make my own mapping. That seems
> >> quite error prone.
> >
> > ?!
> >
> >
> >> I’m looking for a "right" way to do it, something like "check out the
> >> source code for...." or "use libXYZ" or "the USB language codes
> >> correspond to .... which is the same as Y and so you can use...."
> >
> > How about "write the source code for it" ?
> >
> > Doing it myself seems like a bad idea.
> >
> I think it’s a great idea. Have someone review it if you’re anxious
> about getting it right. It should be a quick matter, and a very
> straightforward struct [].

Ok, you convinced me. The reason I was hesitant is because I’m kind of
new to i18n, and it seems like the kind of thing it’s easy to have a
simplified understanding of.

It turns out that the USB language IDs are the same as Microsoft LCID
codes (best I could tell without examining each and every one).
There’s a table matching those LCIDs with string language codes here:


I did a little regex work and turned that table into something usable.
The code is at:


beginning at line 806

I’d appreciate some review of it, as it seems like there’s plenty of
places where I could have made incorrect assumptions.

---

i18n problems

filelistview: transform filename to unicode

# HG changeset patch
# User André Sintzoff <andre.sintzoff@gmail.com>
# Date 1300821371 -3600
filelistview: transform filename to unicode

Since fd2662ea0068, changes from a file containing non ASCII characters in its name are no more displayed in fileview as the filename is incorrectly handled.

@@ -95,7 +95,7 @@
index = self.currentIndex()
data = self.model().dataFromIndex(index)
if data:
- self.fileSelected.emit(data['path'], data['status'])
+ self.fileSelected.emit(hglib.tounicode(data['path']), data['status'])
else:
    self.clearDisplay.emit()

guess: convert paths to unicode before display (fixes #410)

@@ -209,9 +209,10 @@
for index in self.matchtv.selectionModel().selectedRows():
    src, dest, percent = self.matchtv.model().getRow(index)
    if dest in remdests:
+    udest = hglib.tounicode(dest)
        QMessageBox.warning(self, _('Multiple sources chosen'),
            _('You have multiple renames selected for ')
            + _('destination file:
%s. Aborting!') % dest)
+    + _('destination file:
%s. Aborting!') % udest)
    return
remdests[dest] = src
for dest, src in remdests.iteritems():
@@ -233,7 +234,8 @@
date = hglib.displaytime(ctx.date())
difftext = mdiff.unidiff(rr, date, aa, date, src, dest, None)
if not difftext:
-    t = _('%s and %s have identical contents
\n\n') % (src, dest)
+    t = _('%s and %s have identical contents
\n\n') % 
+    (hglib.tounicode(src), hglib.tounicode(dest))
    hu.write(t, label='ui.error')
else:
    for t, l in patch.difflabel(difftext.splitlines, True):

Re: [PySide] I can not load files with non-utf8 names

de Benoît HERVIER <khertan@khertan.net>
heure de l’expéditeur Envoyé à 16:09 (GMT+01:00). Heure locale : 17:15.
à hipersayan x <hipersayan.x@gmail.com>
cc pyside@lists.pyside.org
date 23 mars 2011 16:09
objet Re: [PySide] I can not load files with non-utf8 names

Hi,

  # -*- coding: utf-8 -*-
  # ^ Declare that the file should be read by the python parser as utf-8
img = QtGui.QImage()
results = img.load(u'/home/MyHome/Imágenes/Vacación_En_España.png')
# ^ create the string as unicode stringe

Should do the trick, and save your py file in utf-8.

For more informations with utf-8 and python, i often suggest http://www.evanjones.ca/python-utf8.html

Regards,

2.17 Languages

2.17.1 programming languages

See Also:

• https://github.com/languages
• http://hyperpolyglot.org/

bash language

Advanced Bash-Scripting Guide

See Also:

• http://www.tldp.org/LDP/abs/html/

This tutorial assumes no previous knowledge of scripting or programming, but progresses rapidly toward an interme-
diate/advanced level of instruction . . . all the while sneaking in little nuggets of UNIX® wisdom and lore.

It serves as a textbook, a manual for self-study, and a reference and source of knowledge on shell scripting techniques.
The exercises and heavily-commented examples invite active reader participation, under the premise that the only way
to really learn scripting is to write scripts.

C language

See Also:

• https://fr.wikipedia.org/wiki/Langage_C
Introduction

Le C est un langage de programmation impératif conçu pour la programmation système. Inventé au début des années 1970 avec UNIX, C est devenu un des langages les plus utilisés. De nombreux langages plus modernes comme C++, Java et PHP reprennent des aspects de C.

c parser

See Also:
http://pypi.python.org/pypi/pycparser

dycparser is a complete parser of the C language, written in pure Python using the PLY parsing library. It parses C code into an AST and can serve as a front-end for C compilers or analysis tools.

C bindings

C and C bindings  See Also:

  • How to make a callback to Csharp from C Cplusplus
  • http://www.codeproject.com/Tips/318140/How-to-make-a-callback-to-Csharp-from-C-Cplusplus

C libraries  See Also:

  • https://fr.wikipedia.org/wiki/Langage_C

bionic  See Also:

  • https://fr.wikipedia.org/wiki/Langage_C

glibc  See Also:

  • https://fr.wikipedia.org/wiki/Langage_C

c language people

c language people Dennis Ritchie  See Also:

  • https://fr.wikipedia.org/wiki/Dennis_Ritchie
  • Fedora 16 Dedicated to Dennis Ritchie
Dennis MacAlistair Ritchie, né le 9 septembre 1941 à Bronxville dans l’État de New York et trouvé mort le 12 octobre 2011 à Berkeley Heights dans le New Jersey, est un des pionniers de l’informatique moderne, inventeur du langage C et co-développeur de Unix.

Il est parfois désigné par dmr, son adresse e-mail aux Laboratoires Bell.

Au début des années 1970, il travaille avec Ken Thompson en tant que programmeur dans les Laboratoires Bell sur le développement d’Unix.

Pour Unix, il s’avère nécessaire d’améliorer le langage B créé par Ken Thompson et c’est ainsi que Ritchie crée le langage C. Par la suite, aidé de Brian Kernighan, il promeut le langage et rédige notamment le livre de référence The C Programming Language.

Il reçoit conjointement avec Ken Thompson le prix Turing de l’ACM en 1983 pour leur travail sur le système Unix.

**Dennis Ritchie nous a quitté**

See Also:

https://www.april.org/dennis-ritchie-nous-quitte

Dennis M. Ritchie, l’un des plus grands contributeurs de l’informatique moderne nous a quitté.

Ce que Dennis Ritchie a apporté à l’informatique est difficile à quantifier. En effet, dans les années 70, alors qu’il est employé par les laboratoires Bell, il travaille avec Ken Thompson sur un projet de système d’exploitation dont les qualités devaient être la portabilité, notamment vers des machines de toute dimension. Pour cela, il conçoit un nouveau langage de programmation dérivé du langage B de Ken Thompson et du BCPL de Martin Richards.

Ce langage et ce système d’exploitation ne sont autre que le langage C et Unix. Ce langage se révèle complet et extrêmement performant et son usage dépassera rapidement la cadre de la programmation sous Unix.

Pour présenter son nouveau langage, Dennis M. Ritchie écrira avec Brian Kernighan, l’un des plus beaux ouvrage de programmation : *The C programming Language*.

Les auteurs y présentent toute la philosophie, les références du langage et des exercices, entraînant ainsi le lecteur vers la maîtrise progressive de ce langage. De ce livre, le K&R pour les informaticiens, on retiendra également le mythique premier exemple, qui est sans aucun doute la ligne de code la plus connue de tous les développeurs : `printf("hello, worldn");`

Le langage C est aujourd’hui toujours très utilisé pour ses performances et sa portabilité. Il est la souche de nombreux autres langages : le C++, l’Objective C cher à Apple, le C# de Microsoft, le Java et même le PHP en tirent leurs racines. L’immense majorité des projets libres sont basés sur ces langages. Du côté des systèmes d’exploitation, l’Unix de Ritchie et Thompson est la souche de nombreux systèmes libres tels que Minix, GNU/Linux, FreeBSD, NetBSD, OpenBSD.

Il est difficile d’imaginer ce que serait l’informatique d’aujourd’hui sans les travaux de Dennis Ritchie. Pour ne donner qu’un exemple, le système d’exploitation Android de Google, pour les smartphones et les tablettes, est basé sur un noyau Linux et sur le langage Java, dérivés respectivement d’Unix et du langage C.
Dennis M. Ritchie est décédé à l’âge de 70 ans. Homme d’une grande modestie, il constitue pour de nombreux informaticiens un exemple de technique et de partage du savoir au travers de l’un des plus beaux traité d’informatique. L’April, et plus généralement la communauté du Libre, lui rendent aujourd’hui hommage.

C standards

C11 standard (2011) See Also:

Contents
- C11 standard (2011)
  - Introduction
  - CLang, LLVL
  - GCC et C11
    * C11 n’est pas encore mort

Introduction
C11 (formerly C1X) is an informal name for ISO/IEC 9899:2011, the current standard for the C programming language.

It replaces the previous C standard, informally known as C99.

This new version mainly standardizes features that have already been supported by common contemporary compilers, and includes a detailed memory model to better support multiple threads of execution.

Due to delayed availability of conforming C99 implementations, C11 makes more features optional, to make it easier to comply with the core language standard.

CLang, LLVL See Also:
- Clang

GCC et C11 See Also:
- https://linuxfr.org/news/sortie-de-la-version-4-7-du-compilateur-gcc#toc_13
- GCC (GNU Compiler Collection)

Du côté du bon vieux langage C il y a là aussi des nouveautés puisque les développeurs de GCC travaillent sur la norme C11 (voir cette dépêche LinuxFr de GeneralZod pour un récapitulatif des nouveautés).

Quand on active l’option -std=c11 on retrouve donc la gestion des chaînes unicode et des macros __STDC_UTF_16__ et __STDC_UTF_32__.

GCC 4.7 intègre les fonctions d’alignement _Alignas et _Alignof, la gestion du mot-clé _Noreturn, la fonction __builtin_complex pour profiter des macros CMPLX, etc.

Maintenant que la norme C11 a été officiellement ratifié par l’ISO le 8 décembre 2011, il est probable que le travail d’intégration dans GCC va s’accélérer.
C11 n’est pas encore mort  See Also:

  * https://linuxfr.org/news/c11-nest-pas-encore-mort

La dernière norme du langage C a été publiée le 8 décembre 2011, la pré-version finale étant accepté le 10 octobre, peu avant le décès du regreté Denis Ritchie à l’origine du langage.

C11 intègre principalement la gestion du multithreading et rend optionnelles certaines fonctionnalités afin de faciliter la conformité des compilateurs vis-à-vis de la norme.

La précédente norme C99 a eu beaucoup de mal à s’imposer auprès de certains éditeurs, certains refusant même de l’impléanter (notamment un vendeur d’environnement Basic, situé à Redmond).

C99 standard (1999)  See Also:


Contents

- C99 standard (1999)
  - Introduction

Introduction  C99 is an informal name for ISO/IEC 9899:1999, a past version of the C programming language standard.

It extends the previous version (C90) with new linguistic and library features, and helps implementations make better use of available computer hardware, such as IEEE 754 arithmetic, and compiler technology.

C stdio functions

tmpfile()

c time functions

strftime c time function  See Also:

  * strftime c time function
    * http://manpagesfr.free.fr/man/man3/strftime.3.html

Contents

- strftime c time function
  - Signature
  - Tutorials
  - Afficher l’heure courante
    * Resultat

Signature

size_t strftime(char *s, size_t max, const char *format, const struct tm *tm);
Afficher l’heure courante  L’affichage de la date ou de l’heure de fait grâce à la fonction strftime() qui a été conçue pour ça.

Elle attend l’adresse d’une structure tm, qu’il va donc falloir mettre à jour au préalable. On utilise pour ça la valeur retournée par time(), qui donne la datation courante sous la forme d’un entier et une des fonction localtime() ou gmtime() selon que l’on veut convertir en heure locale ou en heure GMT (Greenwich).

En France, il y a une heure de différence.

L’usage précis de strftime() et de ses nombreux paramètres de formatage doit être étudié dans un document de référence.

```c
#include <stdio.h>
#include <time.h>

int main (void)
{
    /* lire l’heure courante */
    time_t now = time (NULL);

    /* la convertir en heure locale */
    struct tm tm_now = *localtime (&now);

    /* Creer une chaine JJ/MM/AAAA HH:MM:SS */
    char s_now[ sizeof "JJ/MM/AAAA HH:MM:SS" ];

    strftime (s_now, sizeof s_now, "%d/%m/%Y %H:%M:%S", &tm_now);

    /* afficher le resultat : */
    printf ("'%s'\n", s_now);
    return 0;
}
```

Résultat

’29/01/2009 02:34:00’
Process returned 0 (0x0)  execution time : 0.054 s
Press any key to continue.

strptime c time function  See Also:

- strptime c time function
- http://stackoverflow.com/questions/321849/strptime-equivalent-on-windows
Contents

• strptime c time function
  – Signature
  – Description
  – Exemple
  – Resultat

Signature

char *strptime(const char *s, const char *format, struct tm *tm);

Description  La fonction strptime() est complémentaire de la fonction strftime(3). Elle convertit la chaîne de caractères pointée par buf en une valeur qui est stockée dans la structure tm pointée par l’argument tm, la conversion étant réalisée en suivant les indications contenues dans la chaîne format.

Cette dernière contient des descripteurs de champs et du texte, rappelant scanf(3). Chaque descripteur consiste en un caractère % suivi d’un second caractère indiquant le champ à interpréter.

Tous les autres sont considérés comme du texte, qui doit être présent dans la chaîne fournie en entrée. Toutefois une espace blanche se trouvant dans la chaîne de format peut être mise en correspondance avec zéro ou plusieurs espaces. Il devrait toujours y avoir une espace ou un autre caractère alphanumérique entre deux descripteurs de champs.

La fonction strptime() traite la chaîne d’entrée de gauche à droite. Les trois types d’éléments d’entrée possibles (espace, caractère littéral, conversion) sont manipulés l’un après l’autre.

Si l’entrée ne peut pas être mise en correspondance avec la chaîne de format, la fonction s’arrête.

Le reste du format et de la chaîne d’entrée ne sont pas traités.

Exemple  L’exemple suivant montre l’utilisation de strptime() et strftime(3).

```c
#define _XOPEN_SOURCE
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void)
{
  struct tm tm;
  char buf[255];

  strftime("2001-11-12 18:31:01", "%Y-%m-%d %H:%M:%S", &tm);
  strftime(buf, sizeof(buf), "%d %b %Y %H:%M", &tm);
  puts(buf);
  exit(EXIT_SUCCESS);
}
```

Resultat

‘29/01/2009 02:34:00’
Process returned 0 (0x0)  execution time : 0.054 s
Press any key to continue.
C tutorials

basic-programs-in-c-explained-part-1  See Also:
  • http://www.scriptsnippets.com/basic-programs-in-c-explained-part-1/

css language

See Also:
  • http://fr.wikipedia.org/wiki/Feuilles_de_style_en_cascade

CSS (Cascading Style Sheets : feuilles de style en cascade) est un langage informatique qui sert à décrire la présentation des documents HTML et XML.

Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C).


css tutorials

See Also:
  • http://www.netmagazine.com/tag/css?ct=tutorial

http://www.codestyle.org/css/Glossary.shtml  See Also:
  • http://www.codestyle.org/css/Glossary.shtml
  • Web tutorials

instacss tutorial  See Also:
  • http://instacss.com/
  • https://github.com/rgarcia/instacss
  • http://instacss.com/#a
  • http://news.ycombinator.com/item?id=3233826

shayhowe tutorial A Beginners Guide to HTML & CSS  See Also:
  • http://learn.shayhowe.com/html-css/terminology-syntax-intro
  • Web tutorials

C++ language

See Also:
  • http://www.cplusplus.com/
C/C++ programming guides

See Also:

- http://www.cplusplus.com/

Czmq manual  See Also:
http://czmq.zeromq.org/manual:czmq

Design ideology The problem with C  C has the significant advantage of being a small language that, if we take a little care with formatting and naming, can be easily interchanged between developers.

Every C developer will use much the same 90% of the language.

Larger languages like C++ provide powerful abstractions like STL containers but at the cost of interchange.

The huge problem with C is that any realistic application needs packages of functionality to bring the language up to the levels we expect for the 21st century.

Much can be done by using external libraries but every additional library is a dependency that makes the resulting applications harder to build and port. While C itself is a highly portable language (and can be made more so by careful use of the preprocessor), most C libraries consider themselves part of the operating system, and as such do not attempt to be portable.

The answer to this, as we learned from building enterprise-level C applications at iMatix from 1995-2005, is to create our own fully portable, high-quality libraries of pre-packaged functionality, in C. Doing this right solves both the requirements of richness of the language, and of portability of the final applications.

Coding styles (mozilla)  See Also:
https://developer.mozilla.org/En/Developer_Guide/Coding_Style

https://developer.mozilla.org/index.php?title=en/C%2b%2b_Portability_Guide  What follows is a set of rules, guidelines, and tips that we have found to be useful in making C++ code portable across many machines and compilers.

This information is the result of porting large amounts of code across about 25 different machines, and at least a dozen different C++ compilers.

Some of these things will frustrate you and make you want to throw your hands up and say, “well, that’s just a stupid compiler if it doesn’t do <insert favorite C++ feature>.”

But this is the reality of portable code.

If you play by the rules, your code will seamlessly work on all of the Mozilla platforms and will be easy to port to newer machines.

opencv style guide  See Also:
http://opencv.willowgarage.com/wiki/CodingStyleGuide
C++ libraries

Boost C++ libraries  See Also:
  • http://www.boost.org/
  • http://www.boost.org/doc/libs/1_47_0/
  • http://en.highscore.de/cpp/boost/

Boost versions

Boost C++ versions

Boost C++ 1.4.9 beta 1  See Also:
http://sourceforge.net/projects/boost/files/boost/1.49.0.beta.1/

Boost C++ 1.4.8  See Also:
  • http://www.boost.org/users/history/version_1_48_0.html
  • http://sourceforge.net/blog/project-of-the-month-february-2012-boost/

New Libraries
  • Container: Standard library containers and extensions, from Ion Gaztañaga.
  • Locale: Provide localization and Unicode handling tools for C++, from Artyom Beilis.
  • Move: Portable move semantics for C++03 and C++11 compilers, from Ion Gaztañaga.

Boost C++ 1.4.7  See Also:
http://www.boost.org/users/history/version_1_47_0.html

New Libraries
  • Chrono: Useful time utilities, from Vicente J. Botet Escribá.
  • Geometry: Geometry Library, from Barend Gehrels, Bruno Lalande and Mateusz Loskot.
  • Phoenix: Define small unnamed function objects at the actual call site, and more, from Joel de Guzman, Dan Marsden and Thomas Heller.
  • Ratio: Compile time rational arithmetic, from Vicente J. Botet Escribá.

Documentation “The Boost C++ Libraries”  See Also:
http://en.highscore.de/cpp/boost/
Boost C++ libraries installation  See Also:
http://www.boostpro.com/download/

Installation on windows with mingw  Install in 2 steps when we use

1. go to C:boost_1_47_0toolsbuildv2 and type:
   ```
   bootstrap.bat
   ```

2. Open the cmd.exe windows and type:
   ```
   C:\boost_1_47_0>bjam.exe --toolset=gcc --build-type=complete
   ```

Installation on GNU/Linux  See Also:
http://www.boost.org/doc/libs/1_47_0/more/getting_started/unix-variants.html

```
/opt/boost/current$ su -c "./bootstrap.sh --prefix=/opt/boost/1.47.0/lib"
```

Building Boost.Build engine with toolset gcc... tools/build/v2/engine/bin.linuxx86/b2
Detecting Python version... 2.7
Detecting Python root... /usr
Unicode/ICU support for Boost.Regex?... not found.
Generating Boost.Build configuration in project-config.jam...
Bootstrapping is done. To build, run:

./b2
To adjust configuration, edit ‘project-config.jam’.
Further information:
- Command line help:
  ./b2 --help
- Getting started guide:
  http://www.boost.org/more/getting_started/unix-variants.html
- Boost.Build documentation:

The Boost C++ Libraries were successfully built!
The following directory should be added to compiler include paths:

/opt/boost/1.47.0

The following directory should be added to linker library paths:

/opt/boost/1.47.0/stage/lib

Boost asio C++ library  See Also:
- http://think-async.com/Asio
- http://sourceforge.net/projects/asio/files/asio/1.4.8%20%28Stable%29/

Introduction  Asio is a cross-platform C++ library for network and low-level I/O programming that provides developers with a consistent asynchronous model using a modern C++ approach.
Asio comes in two variants: (non-Boost) Asio and Boost.Asio.
The differences between the two are outlined below.

What are the differences in the source code?
— Asio is in a namespace called asio::, whereas Boost.Asio puts everything under boost::asio::.
— The main Asio header file is called asio.hpp. The corresponding header in Boost.Asio is boost/asio.hpp. All other headers are similarly changed.
— Any macros used by or defined in Asio are prefixed with ASIO. In Boost.Asio they are prefixed with BOOST_ASIO.
— Asio includes a class for launching threads: asio::thread. Boost.Asio does not include this class, to avoid overlap with the Boost.Thread library
— Boost.Asio uses the Boost.System library to provide support for error codes (boost::system::error_code and boost::system::system_error). Asio includes these under its own namespace (asio::error_code and asio::system_error). The Boost.System version of these classes currently supports better extensibility for user-defined error codes.
— Asio is header-file-only and for most uses does not require linking against any Boost library.

Boost.Asio always requires that you link against the Boost.System library, and also against Boost.Thread if you want to launch threads using boost::thread.

2.17. Languages
Should I use Asio or Boost.Asio? It depends. Here are some things to consider:

— If you prefer the convenience of header-file-only libraries then using Asio over Boost.Asio is suggested.

— If you must use a version of Boost older than 1.35 then Boost.Asio is not included. You can use Boost.Asio by copying it over the top of your Boost distribution (see above), but not everyone is comfortable doing this. In that case, using Asio over Boost.Asio is suggested.

— New versions of both the Asio and Boost.Asio packages will be created on a faster release cycle than that followed by Boost. If you want to use the latest features you can still use Boost.Asio as long as you are happy to copy it over the top of your Boost distribution. If you don’t want to do this, use Asio rather than Boost.Asio.

Can Asio and Boost.Asio coexist in the same program? Yes. Since they use different namespaces there should be no conflicts, although obviously the types themselves are not interchangeable. (In case you’re wondering why you might want to do this, consider a situation where a program is using third party libraries that are also using Asio internally.)

download See Also:

• http://www.boostpro.com/download/
• http://sourceforge.net/projects/asio/files/asio/1.4.8%20%28Stable%29/

Boost DateTime C++ library See Also:

• http://www.boost.org/doc/libs/1_49_0/doc/html/date_time.html

Boost filesystem C++ library See Also:

• http://www.boost.org/doc/libs/1_47_0/libs/filesystem/v3/doc/index.htm

The Boost.Filesystem library provides facilities to manipulate files and directories, and the paths that identify them.

Boost system C++ library See Also:

• http://www.boost.org/doc/libs/1_47_0/libs/system/doc/index.html

Error conditions originating from the operating system or other low-level application program interfaces (API’s) are typically reported via an integer representing an error code.

When these low-level API calls are wrapped in portable code, such as in a portable library, some users want to deal with the error codes in portable ways. Other users need to get at the system specific error codes, so they can deal with system specific needs.

The Boost System library provides simple, light-weight error_code objects that encapsulate system-specific error code values, yet also provide access to more abstract and portable error conditions via error_condition objects. Because error_code objects can represent errors from sources other than the operating system, including user-defined sources, each error_code and error_condition has an associated error_category.

An exception class, system_error, is provided. Derived from std::runtime_error, it captures the underlying error_code for the problem causing the exception so that this important information is not lost.

While exceptions are the preferred C++ default error code reporting mechanism, users of libraries dependent on low-level API’s often need overloads reporting error conditions via error code arguments and/or return values rather than via throwing exceptions. Otherwise, when errors are not exceptional occurrences and must be dealt with as they arise, programs become littered with try/catch blocks, unreadable, and very inefficient.

The Boost System library supports both error reporting by exception and by error code.
In addition to portable errors codes and conditions supported by the error_code.hpp header, system-specific headers support the Cygwin, Linux, and Windows platforms. These headers are effectively no-ops if included for platforms other than their intended target.

**Boost thread C++ library**  See Also:


Boost.Thread enables the use of multiple threads of execution with shared data in portable C++ code. It provides classes and functions for managing the threads themselves, along with others for synchronizing data between the threads or providing separate copies of data specific to individual threads.

The Boost.Thread library was originally written and designed by William E. Kempf.

This version is a major rewrite designed to closely follow the proposals presented to the C++ Standards Committee, in particular N2497, N2320, N2184, N2139, and N2094

In order to use the classes and functions described here, you can either include the specific headers specified by the descriptions of each class or function, or include the master thread library header:

```cpp
#include <boost/thread.hpp>
```

which includes all the other headers in turn.

**STL (Standard Template Library) C++ library**  See Also:


**C++ versions**

**C++11**  See Also:


**C language**

See Also:

- [https://secure.wikimedia.org/wikipedia/fr/wiki/C_sharp](https://secure.wikimedia.org/wikipedia/fr/wiki/C_sharp)

C (pronounced /si rp/ see sharp) is a multi-paradigm programming language encompassing imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines.
It was developed by Microsoft within the .NET initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270). C is one of the programming languages designed for the Common Language Infrastructure.

C is intended to be a simple, modern, general-purpose, object-oriented programming language. Its development team is led by Anders Hejlsberg. The most recent version is C 4.0, which was released on April 12, 2010.

Le nom “C sharp” n’est pas la correspondance du logo C. En effet, le caractère “#” est appelé “number sign” en américain alors que “sharp” est le nom du caractère “♯” utilisé en musique.

La normalisation du langage par l’ECMA indique bien ces deux aspects. La même confusion est souvent faite en français entre les noms croisillon (#) et dièse (♯).

De plus, “♯” correspond à la note de musique “Do”, “C sharp” signifie donc “Do dièse”.

Microsoft précise cependant que le terme sharp choisi fait bien référence à la notation musicale, et que ce n’est que par facilité d’écriture qu’en pratique le symbole # est utilisé.

C documentation

C doc See Also:
• http://www.dotnetperls.com/

C benchmarks See Also:
• http://www.dotnetperls.com/benchmark

You are wondering if the C programming language is fast. You question whether it is worth it to benchmark programming constructs. What else can you learn from careful benchmarks? Here’s a benchmarking overview using the C# language that touches on some important things about benchmarking.

We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. Donald Knuth

C benchmarks examples

C Stopwatch.StartNew() See Also:
http://www.dotnetperls.com/benchmark

using System;
using System.Diagnostics;

class Program
{
    static void Main()
    {
        const int m = 1000000;
        Stopwatch s1 = Stopwatch.StartNew();
        for (int i = 0; i < m; i++)
        {
        }
        s1.Stop();
        Stopwatch s2 = Stopwatch.StartNew();
        stopwatch s3 = Stopwatch.StartNew();
}
A static field identifies exactly one storage location to be shared by all instances of a given closed class type. Hejlsberg et al., p. 434

Example As an introduction, this program shows a static class, a static field, and also a static method. The Main method is a special case of a static method because it is invoked as the entry point when the program begins execution.

You can see that the static field is incremented and displayed.

using System;
static class Perls
{
    public static int _value = 5;
}

class Program
{
    static void Main()
    {
        Console.WriteLine(++Perls._value);
    
}
Every byte in your computer’s memory is part of a single array. Abstractions translate these bytes into objects and give them meaning. Similarly, arrays in the .NET Framework are a foundational type upon which more usable collections, such as List and Dictionary, are based. Arrays use a special syntax form in the C language.

Int array example As an introduction to the C# array type, let’s look at a simple example program that allocates and initializes an integer array of three elements.

Please notice how the elements can be assigned to or read from using the same syntax (values[int]). The array is zero-based; we also demonstrate the foreach loop upon it.

```csharp
using System;

class Program
{
    static void Main()
    {
        // Use an array.
        int[] values = new int[3];
        values[0] = 5;
        values[1] = values[0] * 2;

        foreach (int value in values)
        {
            Console.WriteLine(value);
        }
    }
}
```

Output

```
5
10
```
**Int array example 2**  There is another way to allocate an array in your C# program and fill it with values.

You can use the curly brackets { } to assign element values in one line.

The length of the array is automatically determined when you compile your program.

```csharp
using System;

class Program
{
    static void Main()
    {
        // Create an array of three ints.
        int[] array = { 10, 30, 50 };

        foreach (int value in array)
        {
            Console.WriteLine(value);
        }
    }
}
```

**Output**

```
10
30
50
```

c **dictionaries**

**c dictionary**  See Also:

- [http://www.dotnetperls.com/dictionary](http://www.dotnetperls.com/dictionary)

**Foreach**

```csharp
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // Example Dictionary again
        Dictionary<string, int> d = new Dictionary<string, int>()
        {
            {"cat", 2},
            {"dog", 1},
            {"llama", 0},
            {"iguana", -1}
        };
        // Loop over pairs with foreach
        foreach (KeyValuePair<string, int> pair in d)
        {
            Console.WriteLine("{0}, {1}",
```
pair.Key,
pair.Value);
}
// Use var keyword to enumerate dictionary
dictionary.TryGetValue(key, &value)
{
	Console.WriteLine("{0}, {1}",
		value.Key,
		value.Value);
}
}

c sorteddictionary  See Also:
• http://www.dotnetperls.com/sorteddictionary

e xample 1

using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // 1
        // New SortedDictionary
        SortedDictionary<string, int> sort = new SortedDictionary<string, int>();

        // 2
        // Add strings and int keys
        sort.Add("zebra", 5);
sort.Add("cat", 2);
sort.Add("dog", 9);
sort.Add("mouse", 4);
sort.Add("programmer", 100);

        // 3
        // Example: see if it doesn’t contain "dog"
        if (sort.ContainsKey("dog"))
        {
            Console.WriteLine(true);
        }

        // 4
        // Example: see if it contains "zebra"
        if (sort.ContainsKey("zebra"))
        {
            Console.WriteLine(true);
        }

        // 5
        // Example: see if it contains "ape"
        Console.WriteLine(sort.ContainsKey("ape"));
    }
}
// Example: see if it contains "programmer",
// and if so get the value for "programmer"
int v;
if (sort.TryGetValue("programmer", out v))
{
    Console.WriteLine(v);
}

// Example: print SortedDictionary in alphabetic order
foreach (KeyValuePair<string, int> p in sort)
{
    Console.WriteLine("{0} = {1}",
    p.Key,
    p.Value);
}

Output
True
True
False
100
cat = 2
dog = 9
mouse = 4
programmer = 100
zebra = 5

eexample 2

public class TupleCourantAntenne_Distance
{
    public double m_CourantAntenne; // en ampère-mètre crête
    public double m_DistanceElevation; // en cm

    /// <summary>
    /// constructor TupleCourantAntenne_Distance.
    /// </summary>
    public TupleCourantAntenne_Distance(double courantAntenne, double distanceElevation)
    {
        m_CourantAntenne = courantAntenne;
        m_DistanceElevation = distanceElevation;
    }
}

public partial class EstimateZ_Form : Form
{
    // Storing the list of data collected
    // A data is a tuple of 2 measures:
    // - antenna electric current
    // - elevation distance
public SortedDictionary<int, TupleCourantAntenne_Distance> m_DataCollected;
public TupleCourantAntenne_Distance this[int index]
{
    get { return m_DataCollected[index]; }
}

// Nombre de mesures collectées
// Doit être égal au nombre d'élévations
public int NbMesures
{
    get { return m_DataCollected.Count; }
}

    /// <summary>
    /// Initialisation du formulaire.
    /// </summary>
    void InitForm(ref FedmIscReader i_reader)
    {
        InitializeComponent();
        // ...
        m_DataCollected = new SortedDictionary<int, TupleCourantAntenne_Distance>();
        // ...
    }

    void CloseWindowZForm()
    {
        m_DataCollected.Clear();
        this.Close();
    }

public void test_algo_1()
{
    TupleCourantAntenne_Distance tuple_1 = new TupleCourantAntenne_Distance(1.05, 60);
    TupleCourantAntenne_Distance tuple_2 = new TupleCourantAntenne_Distance(1.45, 72.2);
    TupleCourantAntenne_Distance tuple_3 = new TupleCourantAntenne_Distance(1.85, 82.3);
    TupleCourantAntenne_Distance tuple_4 = new TupleCourantAntenne_Distance(2.02, 86.2);
    TupleCourantAntenne_Distance tuple_5 = new TupleCourantAntenne_Distance(2.13, 88.6);
    TupleCourantAntenne_Distance tuple_6 = new TupleCourantAntenne_Distance(2.52, 96);
    TupleCourantAntenne_Distance tuple_7 = new TupleCourantAntenne_Distance(2.64, 98.2);
    TupleCourantAntenne_Distance tuple_8 = new TupleCourantAntenne_Distance(3.00, 104.2);

    // à la première non détection, les valeurs de courant d'antenne et de
distance d'élévation sont acquises
    m_DataCollected[0] = tuple_1;
    m_DataCollected[1] = tuple_2;
    m_DataCollected[2] = tuple_3;
    m_DataCollected[3] = tuple_4;
    m_DataCollected[4] = tuple_5;
    m_DataCollected[5] = tuple_6;
    m_DataCollected[6] = tuple_7;
m_DataCollected[7] = tuple_8;
int ncz = 0;
m_algo_estimate_Z.calculer_profondeur(ref m_profestimee, ref m_nivhestimee, ref ncz);
}
}

3

example 3

public static string EN_US_CULTURE = "en-US";
public static string FR_FR_CULTURE = "fr-FR";
public static SortedDictionary<string, CultureInfo> cultures = new SortedDictionary<string, CultureInfo>()
{
    { EN_US_CULTURE, new CultureInfo(EN_US_CULTURE)}
    , {FR_FR_CULTURE, new CultureInfo(FR_FR_CULTURE)}
};

c Indexer See Also:
  • http://www.dotnetperls.com/indexer

Tuple

using System;

class Layout
{
    string[] _values = new string[100]; // Backing store

    public string this[int number]
    {
        get
        {
            // This is invoked when accessing Layout instances with the [ ].
            if (number >= 0 && number < _values.Length)
            {
                // Bounds were in range, so return the stored value.
                return _values[number];
            }
            // Return an error string.
            return "Error";
        }
    }

    set
    {
        // This is invoked when assigning to Layout instances with the [ ].
        if (number >= 0 && number < _values.Length)
        {
            // Assign to this element slot in the internal array.
            _values[number] = value;
        }
    }
}
Unlike arrays, the List collection type resizes dynamically: you do not need to manage its size on your own. The List type is ideal for linear collections not accessed by a key. It provides many methods and properties you can use in your C# program.

Example

```csharp
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        List<int> list = new List<int>();
        list.Add(2);
        list.Add(3);
        list.Add(5);
        list.Add(7);
    }
}
```

C Tuple  See Also:

- http://www.dotnetperls.com/tuple

New in version 4.0.

```csharp
using System;

class Program
{
    static void Main()
    {
        // Create three-item tuple.
        Tuple<int, string, bool> tuple = new Tuple<int, string, bool>(1, "cat", true);
        // Access tuple properties.
        if (tuple.Item1 == 1)
        {
            Console.WriteLine(tuple.Item1);
        }
        if (tuple.Item2 == "dog")
        {
            Console.WriteLine(tuple.Item2);
        }
        if (tuple.Item3)
        {
            Console.WriteLine(tuple.Item3);
        }
    }
}
```
example before tuple  If we have a version of C < 4.0 we have to use classes.

```csharp
public class TupleCourantAntenne_Distance
{
    public double m_CourantAntenne;  // en ampère-mètre crête
    public double m_DistanceElevation;  // en cm

    /// <summary>
    /// constructor TupleCourantAntenne_Distance.
    /// </summary>
    public TupleCourantAntenne_Distance(double courantAntenne, double distanceElevation)
    {
        m_CourantAntenne = courantAntenne;
        m_DistanceElevation = distanceElevation;
    }
}
```

C datetime  See Also:

- http://www.dotnetperls.com/datetime
- http://www.dotnetperls.com/datetime-format

You want to use the DateTime type in the C# programming language, and also need various methods to find important days, such as yesterday, tomorrow, the first of the year, and the last day. The DateTime type provides useful methods and properties for computing these values.

Key points: Use DateTime to compute relative dates.
Create new DateTimes with overloaded constructors.
Explore other DateTime properties.

C datetime examples

```csharp
using System;

class Program
{
    static void Main()
    {
        // This DateTime is constructed with an instance constructor.
        // ... We write it to the console.
        // ... If this is today, the second line will be "True".
        DateTime value = new DateTime(2010, 1, 18);
        Console.WriteLine(value);
        Console.WriteLine(value == DateTime.Today);
    }
}
```
C datetime interval

```csharp
DateTime nextDate = firstDate;
nextDate = videoCellTimers.AddIntervalTimeToDateTime(firstDate);

/// <summary>
/// Add interval time to the dateTime parameter.
/// </summary>
/// <returns>
/// DateTime.
/// </returns>
public DateTime AddIntervalTimeToDateTime(DateTime dateTime)
{
    DateTime newDate = dateTime.AddHours(interval.hours).AddMinutes(interval.minutes).AddSeconds(interval.seconds);
    return newDate;
}
```

C datetime milliseconds

```csharp
using System;

/// <summary>
/// Get the time with milliseconds.
/// </summary>
public static string GetTimeMilliseconds()
{
    string datenow_milli = DateTime.Now.ToString("yyyy-MM-dd_HH:mm:ss.fff");
    return datenow_milli;
}
```

C datetime microseconds

```csharp
using System;

/// <summary>
/// Get the time with microseconds.
/// </summary>
public static string GetTimeMicroseconds()
{
    string datenow_micro = DateTime.Now.ToString("yyyy-MM-dd_HH:mm:ss.ffffff");
    return datenow_micro;
}
```

C exceptions  See Also:
A powerful mechanism for catching errors in your code, exception handling is a major improvement over using result codes. Because it isolates error handling in a separate part of your program, it makes the control flow simpler.

Handling exceptions correctly is one of the most important parts of critical applications.

Exceptions in C# provide a structured, uniform, and type-safe way of handling both system-level and application-level error conditions. Hejlsberg et al., p. 599

**Examples** In the C# programming language, you can throw exceptions with a throw statement. Just as important, exceptions can be thrown automatically by the runtime because of the values of the variables in your code. In this program, we divide by zero; this results in a DivideByZeroException, which is reported.

**Program that throws an exception**

```csharp
using System;

class Program
{
    static void Main()
    {
        try
        {
            int value = 1 / int.Parse("0");
            Console.WriteLine(value);
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
}
```

**Parsing a double by using a culture** Use to parse the following numbers:

- 4.6 4.8

If the numbers have a different form (ex: 4,7) an exception will be raised.

```csharp
CultureInfo culture_en = new CultureInfo("en-Gb");

try
{
    String seuil = parser.GetSetting("conf", "seuil");
    IniFile.Seuil = Convert.ToDouble(seuil, culture_en);
}
catch (Exception ex)
{
    throw new Exception("seuil exception", ex);
}
```
C DriveInfo

Exemple

/**
 * @file DriveInfo.cs
 * @author $Author: pvergain $
 * @version $Revision: 505 $
 * @date $Date: 2012-02-06 16:24:18 +0100 (Mon, 06 Feb 2012) $
 * @brief Drive information management.
 * @details
 */
/** @addtogroup DriveInfo
 * @@
 */

using System;
using System.Collections.Generic;
using System.IO;
using System.Globalization;

namespace VideoCell{
    public class CDriveInfo{
        /// <summary>
        /// Returns the C DriveInfo.
        /// </summary>
        /// <remarks> see p.567 C#4.0 </remarks>
        public static DriveInfo GetDriveInfoC()
        {
            DriveInfo c = new DriveInfo("C");
            long totalSize = c.TotalSize;
            long freeBytes = c.TotalFreeSpace;
            long freeToMe = c.AvailableFreeSpace;
            return c;
        }

        /// <summary>
        /// Humanize the the C DriveInfo.
        /// http://en.wikipedia.org/wiki/Binary_prefix
        /// </summary>
        public static string humanize_size_kibi_IEC(long space_available)
        {
            string str_space_available = string.Empty;
            long reste;
            long quotient;
            if (space_available >= 1073741824) {
                }
```csharp
restes = (space_available % 1073741824);
quotient = (space_available / 1073741824);
str_space_available = string.Format("{0} GiB ", quotient);
if (restes > 0)
{
    str_space_available = str_space_available + "+ " + humanize_size_kibi_IEC(restes);
}
else if (space_available >= 1048576)
{
    reste = (space_available % 1048576);
    quotient = (space_available / 1048576);
    str_space_available = string.Format("{0} MiB ", quotient);
    if (reste > 0)
    {
        str_space_available = str_space_available + "+ " + humanize_size_kibi_IEC(reste);
    }
}
else if (space_available >= 1024)
{
    reste = (space_available % 1024);
    quotient = (space_available / 1024);
    str_space_available = string.Format("{0} KiB ", quotient);
    if (reste > 0)
    {
        str_space_available = str_space_available + "+ " + humanize_size_kibi_IEC(reste);
    }
}
else
{
    str_space_available = string.Format("{0} bytes", space_available);
}
return str_space_available;

/// <summary>
/// Humanize the C DriveInfo.
/// http://en.wikipedia.org/wiki/Binary_prefix
/// </summary>
public static string humanize_size_kb_SI(long space_available)
{
    string str_space_available = string.Empty;
    long reste;
    long quotient;

    if (space_available >= 1000000000)
    {
        reste = (space_available % 1000000000);
        quotient = (space_available / 1000000000);
        str_space_available = string.Format("{0} GB ", quotient);
        if (reste > 0)
        {
            str_space_available = str_space_available + "+ " + humanize_size_kb_SI(reste);
        }
    }
    else if (space_available >= 1000000)
    {  
```
```
{
    reste = (space_available % 1000000);
    quotient = (space_available / 1000000);
    str_space_available = string.Format("{0} MB ", quotient);
    if (reste > 0)
    {
        str_space_available = str_space_available + "+ " + humanize_size_kb_SI(reste);
    }
}
else if (space_available >= 1000)
{
    reste = (space_available % 1000);
    quotient = (space_available / 1000);
    str_space_available = string.Format("{0} KB ", quotient);
    if (reste > 0)
    {
        str_space_available = str_space_available + "+ " + humanize_size_kb_SI(reste);
    }
} else
{
    str_space_available = string.Format("{0} bytes", space_available);
}
return str_space_available;
}

/// <summary>
/// Log the the spaces on all the volumes.
/// </summary>
public static void LogFreeSpaces(VideoCellForm videoForm)
{
    DriveInfo[] allDrives = DriveInfo.GetDrives();
    foreach (DriveInfo d in allDrives)
    {
        if (d.IsReady)
        {
            LogAvailableSpace(videoForm, d.Name, d.AvailableFreeSpace, "available_free_space" );
            Log.Write(" ");
        }
    }
}

/// <summary>
/// Log the available space.
/// </summary>
public static void LogAvailableSpace(VideoCellForm videoForm, string drive_name, long available_space, string str_key)
{
    NumberFormatInfo nfi = videoForm.video_culture.Culture.NumberFormat;
    string message = string.Format("{0} {1} = {2} bytes ",
        videoForm.GetI18nString(str_key),
        drive_name,
        available_space.ToString("N", nfi));
    Log.Write(message);
}
```
message = string.Format("(0) {1} SI = {2} "
    , videoForm.GetI18nString(str_key)
    , drive_name
    , humanize_size_kb_SI(available_space)
  );
Log.Write(message);

message = string.Format("(0) {1} IEC = {2} "
    , videoForm.GetI18nString(str_key)
    , drive_name
    , humanize_size_kibi_IEC(available_space)
  );
Log.Write(message);
}

/// <summary>
/// Log the the available free space on the C drive
/// </summary>
public static void LogFreeSpace(VideoCellForm videoForm)
{
    DriveInfo c = GetDriveInfoC();

    LogAvailableSpace(videoForm, "C", c.AvailableFreeSpace, "available_free_space");
}

} // class CDriveInfo
} // VideoCell

/**
   * fin DriveInfo
 */

/*

c properties

“Auto-implemented properties” /”automatically implemented properties” / “Automatic properties” (C3)   See Also:


New in version 3: The automatic properties. Automatic properties are a simple way to write properties which just get and set their values directly from/to a backing variable.

For example, this code:

string name;
public string Name
{
    get { return name; }
    set { name = value; }
}
can be written in C# 3 like this:

    public string Name { get; set; }

You can specify separate access levels for the getter and the setter, just like in C2.

**c indexers**  See Also:

http://www.dotnetperls.com/indexer

**c threading**  See Also:

- http://www.albahari.com/threading/

**C threading backgroundworker**  See Also:

- http://webman.developpez.com/articles/dotnet/introbackgroundworker/
- http://www.dotnetperls.com/backgroundworker

**Notes**  La classe BackgroundWorker vous permet d’exécuter une opération sur un thread séparé, dédié.

Les opérations longues comme les téléchargements et les transactions de base de données peuvent vous donner l’impression que votre interface utilisateur a cessé de répondre au cours de leur exécution.

Si vous souhaitez une interface utilisateur réactive et êtes confronté à de longs délais associés à ce type d’opérations, la classe BackgroundWorker fournit une solution pratique.

Pour exécuter une longue opération en arrière-plan, créez un objet BackgroundWorker et écoutez les événements qui indiquent la progression de votre opération et lorsque celle-ci se termine.

**Note:**  Veillez à ne pas manipuler d’objets interface utilisateur dans votre gestionnaire d’événements DoWork. À la place, communiquez à l’interface utilisateur via les événements ProgressChanged et RunWorkerCompleted.

**Utilisation du background worker**  Le background worker est un composant que l’on peut utiliser indifféremment de manière graphique par glisser/déposer sur un formulaire Windows par exemple ou alors en le déclarant directement dans le code.

**Instanciation d’un Background worker**

```csharp
BackgroundWorker bgw1 = new BackgroundWorker();
bgw1.DoWork += new DoWorkEventHandler(bgw1_DoWork);
bgw1.ProgressChanged += new ProgressChangedEventHandler(bgw1_ProgressChanged);
bgw1.RunWorkerCompleted += new RunWorkerCompletedEventHandler(bgw1_RunWorkerCompleted);
```

Après avoir instancié le background worker il suffit d’appeler la méthode RunWorkerAsync pour lancer le traitement asynchrone, cela signifie que le code associé est exécuté dans un thread dédié et qu’il ne provoque donc pas un blocage ou un ralentissement de l’affichage et des traitements du thread principal.

La méthode RunWorkerAsync prend un paramètre qui est de type object, cela laisse donc le loisir de passer ce que l’on veut au thread qui exécutera le code.
Nous verrons plus tard dans ce tutoriel comment récupérer un résultat.
Le composant background worker possède 3 événements qui vont nous servir à placer le code que l’on souhaite exécuter, a détecter la progression ainsi que la fin de cette exécution.
Il faut s’abonner à ces 3 événements pour pouvoir tirer parti du composant.

backgroundworker Examples

C backgroundworker example    See Also:

• http://stackoverflow.com/questions/3384394/reportprogress-c-sharp-question

Implementation of the the background worker

/**
* @file MainForm_BackGroundWorker.cs
* @author $Author: pvergain $
* @version $Revision: 476 $
* @date $Date: 2012-01-25 13:56:47 +0100 (Wed, 25 Jan 2012) $
* @brief BackGroundWorker.
* @details
*
* @addtogroup MainForm_BackGroundWorker
* {@
*/

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Resources;
using System.Reflection; // Assembly
using System.Threading;

/** @addtogroup MainForm_BackGroundWorker
 * @{
 */

using VideoCell;

namespace VideoCell
{
    public partial class VideoCellForm : Form , IObservateurLog
    {

        private BackgroundWorker background_worker_acquisition;
        public BackgroundWorker BackgroundWorkerAcquisition
        {
            get { return background_worker_acquisition; }
            set { background_worker_acquisition = value; }
        }
    }
}
private void background_worker_acquisition_DoWork(object sender, DoWorkEventArgs e)
{
    Dictionary<string, string> dict_messages_i18n = e.Argument as Dictionary<string, string>;
    Dictionary<int, string> dict_messages = new Dictionary<int, string>();
    int i_date = 1;
    foreach (DateTime date_acquisition in AcquisitionScheduler.SecondaryAcquisitionDates.Keys)
    {
        if (BackgroundWorkerAcquisition.CancellationPending == true)
        {
            e.Cancel = true;
            break;
        }
        else
        {
            // Time consuming part
            string well_name = AcquisitionScheduler.SecondaryAcquisitionDates[date_acquisition];
            int percentComplete = (i_date * 100) / nb_dates;
            while (date_acquisition >= DateTime.Now)
            {
                dict_messages[0] = "WAITING";
                string message_acquisition = dict_messages_i18n["waiting_next_acquisition_date"]
                    .Format("{0} {1:d/M/yyyy HH:mm:ss} ", message_acquisition, date_acquisition);
                // give the message to display to the main thread
                BackgroundWorkerAcquisition.ReportProgress(percentComplete, dict_messages);
                // We are waiting until date_acquisition > DateTime.Now
                Thread.Sleep(200);
            }
            string message_start_acquisition = dict_messages_i18n["start_image_acquisition_for_well"]
                .Format("{0} {1} {2:d/M/yyyy HH:mm:ss} ({3}/{4}) ", message_start_acquisition,
                well_name, date_acquisition, i_date, nb_dates);
            dict_messages[0] = "ACQUISITION";
            dict_messages[1] = message_start_acquisition;
        }
    }
}

// Report progress as a percentage of the total task.
BackgroundWorkerAcquisition.ReportProgress(percentComplete, dict_messages);

i_date++;
Thread.Sleep(500);

private void background_worker_acquisition_DoWork(object sender, DoWorkEventArgs e)
{
    Dictionary<int, string> dict_messages = (Dictionary<int, string>)e.UserState;
    if (dict_messages[0] == "ACQUISITION")
    {
        label_message.Text = (e.ProgressPercentage.ToString() + "%");
        Log.Write(label_message.Text);
        Log.Write(dict_messages[1]);
    }
    else if (dict_messages[0] == "WAITING")
    {
        label_message.Text = dict_messages[1];
    }
}

/// </summary>
/// Initialize the backgroung worker.
/// </summary>
void InitializeBackgroundWorker()
{
    background_worker_acquisition = new BackgroundWorker();
    background_worker_acquisition.DoWork += new DoWorkEventHandler(background_worker_acquisition_DoWork);
    background_worker_acquisition.RunWorkerCompleted += new RunWorkerCompletedEventHandler(background_worker_acquisition_RunWorkerCompleted);
    background_worker_acquisition.ProgressChanged += new ProgressChangedEventHandler(background_worker_acquisition_ProgressChanged);
    background_worker_acquisition.WorkerReportsProgress = true;
    background_worker_acquisition.WorkerSupportsCancellation = true;
}

/// </summary>
/// The image acquisition is finished.
/// </summary>
void AcquisitionIsOver()
{
    acquisition_is_launched = false;
    // put back "image acquisition"
    this.btn_image_acquisition.Text = GetI18nString("lancer_acquisition_images");
    TestLaunchImageAcquisition();
}
private void background_worker_acquisition_RunWorkerCompleted(object sender, RunWorkerCompletedEventArgs e)
{
    AcquisitionIsOver();
    if (e.Cancelled == true)
    {
        label_message.Text = GetI18nString("acquisition_canceled");
    }
    else if (e.Error != null)
    {
    }
    else
    {
        label_message.Text = GetI18nString("fin_acquisition");
    }
    Log.Write(label_message.Text);
}

Calling the background worker

Dictionary<string, string> dict_messages_i18n = new Dictionary<string, string>();

string message_acquisition = GetI18nString("waiting_next_acquisition_date");
dict_messages_i18n["waiting_next_acquisition_date"] = message_acquisition;

string message_start_acquisition = GetI18nString("start_image_acquisition_for_well");
dict_messages_i18n["start_image_acquisition_for_well"] = message_start_acquisition;

string date_prevue_acquisition = GetI18nString("date_prevue_acquisition_image");
dict_messages_i18n["date_prevue_acquisition_image"] = message_start_acquisition;

// Start the images acquisition.
BackgroundWorkerAcquisition.RunWorkerAsync(dict_messages_i18n);

See Also:
You are developing a program in the C# language targeting the .NET Framework that must wait on an external process or procedure for some period of milliseconds to ensure correct execution of the application.

The Thread.Sleep method in the .NET Framework and the System.Threading namespace provides a way to suspend your program.

Here we look at how you can use Thread.Sleep.

Example

```csharp
using System;
using System.Diagnostics;
using System.Threading;

class Program
{
    static void Main()
    {
        // Demonstrates three different ways of calling Sleep.
        //
        var stopwatch = Stopwatch.StartNew();
        Thread.Sleep(0);
        stopwatch.Stop();
        Console.WriteLine(stopwatch.ElapsedMillseconds);
        Console.WriteLine(DateTime.Now.ToLongTimeString());

        stopwatch = Stopwatch.StartNew();
        Thread.Sleep(5000);
        stopwatch.Stop();
        Console.WriteLine(stopwatch.ElapsedMillseconds);
        Console.WriteLine(DateTime.Now.ToLongTimeString());

        stopwatch = Stopwatch.StartNew();
        System.Threading.Thread.Sleep(1000);
        stopwatch.Stop();
        Console.WriteLine(stopwatch.ElapsedMillseconds);

        // Bonus: shows SpinWait method.
        //
        stopwatch = Stopwatch.StartNew();
        Thread.SpinWait(100000 * 10000);
        stopwatch.Stop();
        Console.WriteLine(stopwatch.ElapsedMillseconds);
    }
}
```

Output

0 ElapsedMillseconds after Sleep(0)
8:14:43 AM Time after Sleep(0)
4999 ElapsedMillseconds after Sleep(5000)
8:14:48 AM Time after Sleep(5000)
999 ElapsedMillseconds after Sleep(1000)
3144 ElapsedMillseconds after SpinWait(Int32)
Task  See Also:


Appels asynchrones  See Also:
http://www.c2idotnet.com/articles/pattern-d-appel-asynchrone-dans-vos-services

c types

c bytearray type  See Also:
http://www.dotnetperls.com/convert-string-byte-array

You want to convert your string into a byte array, using the C# programming language.
Because strings in the C# language are stored with two bytes per character, and ASCII only allows one byte per character, this can cause data loss if the string is not actually ASCII-encoded. However, if the string is ASCII, you can store it in a byte array for reduced memory usage.
The `Encoding.ASCII.GetBytes()` method is demonstrated here.

Example  First off, we look at a program written in the C# language that uses a constant string literal and then converts that to a byte array.

Please note that the `GetBytes()` method will cause an incorrect conversion if the input string is not actually ASCII.

In the final loop, we examine each byte of the resulting byte array and see its numeric and also character-based representation; this demonstrates correctness.

```csharp
using System;
using System.Text;

class Program
{
    static void Main()
    {
        // Input string.
        const string input = "Dot Net Perls";

        // Invoke GetBytes method.
        // ... You can store this array as a field!
        byte[] array = Encoding.ASCII.GetBytes(input);

        // Loop through contents of the array.
        foreach (byte element in array)
        {
            Console.WriteLine("{0} = {1}", element, (char)element);
        }
    }
}
```
Output

68 = D
111 = o
116 = t
32 =
78 = N
101 = e
116 = t
32 =
80 = P
101 = e
114 = r
108 = l
115 = s

Examining the output. The character representations from the input string were first converted to fit in one byte elements each. Then, we see that the integer 68 corresponds to the uppercase letter ‘D’; this is the standard.

C Windows Forms  See Also:

C GUI

C Windows Forms  See Also:

• http://www.dotnetperls.com/picturebox

C versions

C versions  See Also:

• https://secure.wikimedia.org/wikipedia/en/wiki/C_Sharp_%28programming_language%29#Versions
• http://www.codeproject.com/Articles/327916/C-Language-Features-From-C-2-0-to-4-0

C version 5.0  See Also:

https://secure.wikimedia.org/wikipedia/en/wiki/C_Sharp_%28programming_language%29#Versions

Date  unknown

Frameworks

• unknown

IDEs

Summary of version

• Asynchronous methods
• Compiler as a service
C version 4.0  See Also:
https://secure.wikimedia.org/wikipedia/en/wiki/C_Sharp_%28programming_language%29#Versions

Date  April 2010

Frameworks
• Dot Net Framework 4.0

IDEs
• Visual Studio 2010
  • sharpdevelop 4.1

Summary of version
• Dynamic binding
• Named and optional arguments
• Generic co- and contravariance

C version 3.0  See Also:
https://secure.wikimedia.org/wikipedia/en/wiki/C_Sharp_%28programming_language%29#Versions

Date  November 2007

Frameworks
• Dot Net Frameworks 3.5
• Dot Net Frameworks 3.0
• Dot Net Frameworks 2.0

IDEs
• Visual Studio: 2008
• Visual Studio: 2010
• Monodevelop 2.8
• Monodevelop 2.6

Summary of version
• Implicitly typed local variables
• Object and collection initializers
• Auto-Implemented properties
• Anonymous types
• Extension methods
• Query expressions
• Lambda expressions
• Expression trees
C coding standards

C coding standards

C aviva coding standards  See Also:
http://csharpguidelines.codeplex.com/

1 Introduction  See Also:
• http://twitter.com/ddoomen
• dennis.doomen @ avivasolutions.nl
• http://www.csharpcodingguidelines.com

1.1 What is this?  This document attempts to provide useful and pragmatic guidelines for programming in C# 3.0 and C# 4.0 that we at Aviva Solutions already use in our day-to-day work.

Coding guidelines, or coding standards if you will, are documents consisting of rules and recommendations on the use of C# in enterprise systems.

They deal with code layout, naming guidelines, the proper use of the .NET Framework, tips on writing useful comments and XML documentation, and often also include guidance on proper object-oriented design.

Visual Studio’s Static Code Analysis (a.k.a. FxCop) and StyleCop can automatically verify a majority of those rules and recommendations by analyzing the compiled assemblies. You can configure to do that at compile time or as part of a continuous or daily build.

This document adds additional rules and recommendations and its companion site, http://www.csharpcodingguidelines.com provides a list of Code Analysis rules that are applicable for Line-of-Business applications and frameworks.

1.2 Why are guidelines necessary?  Because not every developer:

• is aware of the potential pitfalls of certain constructions in C#.
• is introduced into certain conventions when using the .NET Framework (e.g. IDisposable)
• is aware of the impact of using (or neglecting to use) particular solutions on aspects like security, performance, multi-language support, etc.
• knows that not every developer is as capable in understanding an elegant, but abstract, solution as the original developer.

Although complying with coding guidelines may seem to appear as undesired overhead or may limit creativity, this approach has already proven its value for many years. Also beware of the fact that not all coding guidelines have a clear rationale. Some of them are simply choices we made at Aviva Solutions.

1.3 Basic Principles  There are many unexpected things I run into during my work as a consultant, each deserving at least one guideline.

Unfortunately, I’m still trying to keep this document within a reasonable size.

But unlike to what some junior developers believe, that doesn’t mean that when something is not mentioned in this guidelines it must be okay.

2.17. Languages  505
In general, if I have a discussion with a colleague about a smell that this document does not provide absolution for, I’ll refer back to a set of basic principles that apply to all situations, regardless of context. These include:

- The Principle of Least Surprise (or Astonishment), which means that you should choose a solution that does include any things people might not understand, or put on the wrong track.
- Keep It Simple Stupid (a.k.a. KISS), a funny way of saying that the simplest solution is more than sufficient.
- You Ain’t Gonne Need It (a.k.a. YAGNI), which tells you to create a solution for the current problem rather than the ones you think will happen later on (since when can you predict the future?)
- Don’t Repeat Yourself (a.k.a. DRY), which requires you to rigorously remove duplication in your code base

Regardless of the elegance of somebody’s solution, if it’s too complex for the ordinary developer, or exposes unusual behavior, or tries to solve many possible future issues, it is very likely the wrong solution and needs redesign.

1.4 How do I get started?

- Ask all developers to carefully read this document at least once. This will give them a sense of the kind of guidelines the document contains.
- Make sure there are always a few hard copies of the Quick Reference close at hand.
- Include the most critical coding guidelines on your Project Checklist and verify the remainder as part of your Peer Review.
- Decide which CA rules are applicable for your project and write these down somewhere, such as your TFS team site, or create a custom Visual Studio 2010 Rule Set. The companion site offers Visual Studio 2010 rule sets for both Line-of-Business applications and more generic code like frameworks and class libraries.
- Add a custom Code Analysis Dictionary containing your domain- or company-specific terms, names and concepts. If you don’t, Static Analysis will report warnings for (parts of) phrases that are not part of its internal dictionary.
- Configure Visual Studio to verify the selected CA rules as part of the Release build. Then they won’t interfere with normal developing and debugging activities, but still can be run by switching to the Release configuration.
- Add the verification of the CA rules part of your Continuous or Daily Build.
- Add a rule to your team that the first one who comes in the office in the morning will check the build for CA warning and will make sure somebody (not necessarily himself) solves it as soon as possible.
- Add an item to your project checklist to make sure all new code is verified against CA violations, or use the corresponding Check-in Policy if you want to prevent any code from violating CA rules at all.
- ReSharper has an intelligent code inspection engine that, with some configuration, already supports many aspects of the Coding Guidelines. It will automatically highlight any code that does not match the rules for naming members (e.g. Pascal or Camel casing), detect dead code, and many other things. One click of the mouse button (or the corresponding keyboard shortcut) is usually enough to fix it.
- ReSharper also has a File Structure window that shows an overview of the members of your class or interface and allows you to easily rearrange them using a simple drag-and-drop action.
- Using GhostDoc you can generate XML comments for any member using a keyboard shortcut. The beauty of it, is that it closely follows the MSDN-style of documentation. However, you have to be careful not to misuse this tool, and use it as a starter only.
- Consider reading the book Clean Code: A Handbook of Agile Software Craftsmanship by Robert C. Martin. It provides excellent guidance on writing elegant and simple code that is easy to maintain and to extend. His ideas have evolved into a new quality standard maintained by many well-known community members, and had a lot of influence on this document.
1.5 Why did you create it? The idea started in 2002 when Vic Hartog (Philips Medical Systems) and I were assigned the task of writing up a coding standard for C# 1.0.

Since then, I’ve regularly added, removed and changed rules based on experiences, feedback from the community and new tooling support such as offered by Visual Studio 2010.

Additionally, after reading Robert C. Martin’s book Clean Code: A Handbook of Agile Software Craftsmanship, I became a big fan of his ideas and decided to include some of his smells and heuristics as guidelines.

Notice though that this document is in no way a replacement for his book. I sincerely recommend that you read his book to gain a solid understanding of the rationale behind his recommendations.

I’ve also decided to include some design guidelines in addition to simple coding guidelines. They are too important to ignore and have a big influence in reaching high quality code.

1.6 Is this a coding standard? The document does not state that projects must comply with these guidelines, neither does it say which guidelines are more important than others.

However, we encourage projects to decide themselves what guidelines are important, what deviations a project will use, who is the consultant in case doubts arise, and what kind of layout must be used for source code.

Obviously, you should make these decisions before starting the real coding work.

To help you in this decision, I’ve assigned a level of importance to each guideline:

- Guidelines that you should never skip and should be applicable to all situations
- Strongly recommended guidelines
- Recommended guidelines that may not be applicable in all situations

In general, generated code should not need to comply with coding guidelines.

However, if it is possible to modify the templates used for generation, try to make them generate code that complies as much as possible.

1.7 Feedback and disclaimer This document has been compiled using many contributions from (former) colleagues, sources from the Internet, and many years of developing in C#.

If you have questions, comments or suggestions, just let me know by sending me an email at dennis.doomen@avivasolutions.nl or tweet me at http://twitter.com/ddoomen.

I will try to revise and republish this document with new insights, experiences and remarks on a regular basis.

Notice though that it merely reflects my view on proper C# code so Aviva Solutions will not be liable for any direct or indirect damages caused by applying the guidelines of this document.

It is allowed to copy, adapt, and redistribute this document and its companion quick reference guide for non-commercial purposes or internal usage.

However, you may not republish this document, or publish or distribute any adaptation of this document for commercial use without first obtaining express written approval from Aviva Solutions.

C. index:: pair: C ; class design guidelines

2 Class Design Guidelines
**AV1000 A class or interface should have a single purpose**  A class or interface should have a **single purpose** within the system it participates in.

In general, a class is either representing a primitive type like an email or ISBN number, an abstraction of some business concept, a plain data structure or responsible for orchestrating the interaction between other classes. It is never a combination of those.

This rule is widely known as the **Single Responsibility Principle**, one of the **S.O.L.I.D.** principles.

**Tip**  A class with the word **And** in it is an obvious violation of this rule.

**Tip**  Use **Design Patterns** to communicate the intent of a class. If you can’t assign a single design pattern to a class, chances are that it is doing more than one thing.

**Note:**  If you create a class representing a primitive type you can greatly simplify it usage by making it immutable.

---

**AV1001 Only create a constructor that returns a useful object**  There should be no need to set additional properties before the object can be used for whatever purpose it was designed.

**AV1003 An interface should be small and focused**  Interfaces should have a clear name explaining the purpose or role of that interface within the system.

Do not combine many vaguely related members on the same interface, just because they were all on the same class. Separate the members based on the responsibility of those members so that callers only need to call or implement the interface related to a particular task.

This rule is more commonly known as the **Interface Segregation Principle**.

---

**AV1004 Use an interface to support multiple implementations, not a base class**  If you want to expose an extension point to your class, expose it as an interface rather than a base class.

It doesn’t force the users of that extension point to derive their implementations from a base-class that might have undesired behavior.

It improves testability and allows them to use their own implementation.

However, for their convenience you may implement an (abstract) default implementation that can serve as a starting point.

**AV1005 Use an interface to decouple classes from each other**  Interfaces are a very effective mechanism for decoupling classes from each other and:

- They can prevent bidirectional associations;
- They simplify the replacement of one implementation with another;
- They allow replacing an expensive external service or resource with a temporary stub for use in a non-production environment.
- They allow replacing the actual implementation with a dummy implementation in a unit test;
- Using a dependency injection framework you can centralize the choice which class is going to be used whenever a specific interface is requested.
AV1008 Mark classes that only contain static members as static  The advantage of using a static class is that the compiler can make sure that no instance members are accidentally defined.

The compiler will guarantee that instances of this class cannot be created and hence, relieves you of creating a private constructor such as was required in C# 1.0.

Use a static class to contain methods that are not associated with a particular instance. For example:

```csharp
public static class EuroConversion {
    public static Decimal FromUSD(Decimal inUsd) { ... }
    public static Decimal ToUSD(Decimal inEuro) { ... }
}
```

AV1010 Don’t hide inherited members with the new keyword  Not only does the new keyword break Polymorphism, one of the most essential object-orientation principles, it also makes subclasses more difficult to understand.

Consider the following two classes:

```csharp
public class Book {
    public virtual void Print() {
        Console.WriteLine("Printing Book");
    }
}

public class PocketBook : Book {
    public new void Print() {
        Console.WriteLine("Printing PocketBook");
    }
}
```

This will cause the following behavior which is not something you normally expect from class hierarchies:

```csharp
PocketBook pocketBook = new PocketBook();
pocketBook.Print(); // Will output "Printing PocketBook"
((Book)pocketBook).Print(); // Will output "Printing Book"
```

It should not make a difference whether you call Print through the base class or through the derived class.

AV1011 It should be possible to treat a derived object as if it were a base class object  In other words, it should be possible to use a reference to an object of a derived class wherever a reference to its base class object is used without knowing the specific derived class.

A very notorious example of a violation of this rule is throwing a NotImplementedException when overriding some of the base-class methods.

A less subtle example is not honoring the behavior expected by the base-class.

This rule is also known as the Liskov Substitution Principle, one of the S.O.L.I.D. principles.
AV1013 Don't refer to derived classes from the base class  Having dependencies between a base class and its derivatives goes against proper object oriented design and prevents other developers from adding new derived classes without having to modify the base class.

AV1014 Avoid exposing the objects an object depends on  If you find yourself writing code like this then you might be violating the *Law of Demeter*:

\[
\text{someObject.SomeProperty.GetChild().Foo()}
\]

An object should not expose any other classes it depends on because callers may misuse that exposed property or method to access the object behind it.

By doing so, you allow calling code to become coupled to the class you are using, and thereby limiting the chance you can easily replace it in a future stage.

**Note:** Using a class that is designed using the *Fluent Interface* pattern does seem to violate this rule, but is in reality something else. It simply returns itself so that method chaining is allowed.

**Exception** Inversion of Control or Dependency Injection frameworks often require you to expose a dependency as a public property. As long as this property is not used for anything else than dependency injection, then it is not considered a violation.

AV1020 Avoid bidirectional dependencies  Having bidirectional dependencies between classes means that two classes know about each other's public members or rely on each other's internal behavior.

Refactoring or replacing one of those two classes requires changes on both parties and may involve a lot of unexpected work.

The most obvious way of breaking that dependency is introducing an interface for one of the classes and using dependency injection.

**Exception** Domain models such as defined in Domain Driven Design tend to occasionally involve bidirectional associations that model real-life associations. In those cases, I would make sure they are really necessary, but if they are, keep them in.

AV1025 Classes should have state and behavior  The only exception to this rule are classes that are used to transfer data over a communication channel, also called Data Transfer Objects, or a class that wraps several parameters of a method.

In general, if you find a lot of data-only classes in your code base, you probably also have a few (static) classes with a lot of behavior.

Use the principles of object-orientation explained in this section and move the logic as close to the data it applies to.

3 Member Design Guidelines

AV1100 Allow properties to be set in any order  Properties should be stateless with respect to other properties, i.e. there should not be a difference between first setting property DataSource and then DataMember, and vice versa.
AV1105 Use a method instead of a property

- If the operation is orders of magnitude slower than setting a field value.
- If the operation is a conversion, such as the Object.ToString method.
- If the operation returns a different result each time it is called, even if the parameters didn’t change. For example, the NewGuid method returns a different value each time it is called.
- If the operation causes a side effect such as changing some internal state not directly related the property.

**Note:** populating an internal cache or implementing lazy loading is a good exception.

AV1110 Avoid mutual exclusive properties

Having properties that cannot be used at the same time typically signals a type that is representing two conflicting concepts.

Even though those concepts may share some of the behavior and state, they obviously have different rules that do not cooperate.

This violation is often seen in domain models and introduces all kinds of conditional logic related to those conflicting rules, causing a ripple effect that significantly worsens the maintenance burden.

AV1115 A method or property should do only one thing

Similarly to rule AV1000, a method or property should do exactly one thing, and *one thing only*.

AV1120 Use a public static readonly field to define predefined value objects

For example, consider a Color struct that stores a color internally as red, green, and blue components and this class has a constructor taking a numeric representation. This class may expose several predefined colors like this:

```csharp
public struct Color
{
    public static readonly Color Red = new Color(0xFF0000);
    public static readonly Color Black = new Color(0x000000);
    public static readonly Color White = new Color(0xFFFFFF);
    public Color(int redGreenBlue)
    {
        // implementation
    }
}
```

AV1125 Don’t expose stateful objects through static members

A stateful object is an object that contains many properties and lots of behavior behind that.

If you expose such an object through a static property or method of some other object, it will be very difficult to refactor or unit test a class that relies on such a stateful object.

In general, introducing a construction like that is a great example of violating many of the guidelines of this chapter.

A classic example of this is the HttpContext.Current property, part of ASP.NET.

It’s very difficult to unit test a class like that without using a very intelligent and intrusive mocking framework such as TypeMock Isolator.

Many see the HttpContext class as a source for a lot of ugly code. In fact, the testing guideline Isolate the Ugly Stuff often refers to this class.
AV1130 **Return an IEnumerable<T> or ICollection<T> instead of a concrete collection class**  In general, you don’t want callers to be able to change an internal collection, so don’t return arrays, lists or other collection classes directly.

Instead, return an IEnumerable<T>, or, if the caller must be able to determine the count, an ICollection<T>.

AV1135 **String, list and collection properties should never return a null reference**  Returning null can be unexpected by the caller. Always return an empty array and an empty string instead of a null reference.

This also prevents cluttering your code base with additional checks for null.

AV1140 **Consider replacing properties using primitive types to use rich value objects**  Instead of using strings, integers and decimals for representing domain specific types such as an ISBN number, an email address or amount of money, consider created dedicated value objects that wrap both the data and the validation rules that apply to it.

By doing this, you prevent ending up having multiple implementations of the same business rules, which both improves maintainability and prevents bugs.

### 4 Miscellaneous Design Guidelines

AV1200 **Throw exceptions rather than returning some kind of status value**  A code base that uses return values for reporting the success or failure tends to have nested if-statements sprinkled all over the code.

Quite often, a caller forgets to check the return value anyhow.

Structured exception handling has been introduced to allow you to throw exceptions and catch or replace exceptions at a higher layer.

In most systems it is quite common to throw exceptions whenever an unexpected situations occurs.

AV1202 **Provide a rich and meaningful exception message text**  The message should explain the cause of the exception and clearly describe what needs to be done to avoid the exception.

AV1205 **Throw the most specific exception that is appropriate**  For example, if a method receives a null argument, it should throw ArgumentNullException instead of its base type ArgumentException.

AV1210 **Don’t swallow errors by catching generic exceptions**  Avoid swallowing errors by catching non-specific exceptions, such as Exception, SystemException, and so on, in application code.

Only top-level code, such as a Last-Chance Exception Handler, should catch a non-specific exception for logging purposes and a graceful shutdown of the application.

AV1220 **Always check an event handler delegate for null**  An event that has no subscribers is null, so before invoking, always make sure that the delegate list represented by the event variable is not null.

Furthermore, to prevent conflicting changes from concurrent threads, use a temporary variable to prevent concurrent changes to the delegate:
event EventHandler<NotifyEventArgs> Notify;

void RaiseNotifyEvent (NotifyEventArgs args)
{
    EventHandler<NotifyEventArgs> handlers = Notify;
    if (handlers != null)
    {
        handlers(this, args);
    }
}

Tip You can prevent the delegate list from being empty altogether.
Simply assign an empty delegate like this:

event EventHandler<NotifyEventArgs> Notify = delegate {};

AV1225 Use a protected virtual method to raise each event  Complying with this guideline allows derived classes to handle a base class event by overriding the protected method. The name of the protected virtual method should be the same as the event name prefixed with On.

For example, the protected virtual method for an event named TimeChanged is named OnTimeChanged.

Important Derived classes that override the protected virtual method are not required to call the base class implementation.

The base class must continue to work correctly even if its implementation is not called.

AV1230 Consider providing property-changed events  Consider providing events that are raised when certain properties are changed. Such an event should be named PropertyChanged, where Property should be replaced with the name of the property with which this event is associated.

Note If your class has many properties that require corresponding events, consider implementing the INotifyPropertyChanged interface instead.

It is often used in the Presentation Model and Model-View-ViewModel patterns.

AV1235 Don’t pass null as the sender parameter when raising an event  Often, an event handler is used to handle similar events from multiple senders. The sender argument is then used to get to the source of the event. Always pass a reference to the source (typically this) when raising the event.

Furthermore don’t pass null as the event data parameter when raising an event. If there is no event data, pass EventArgs.Empty instead of null. Exception On static events, the sender parameter should be null.

AV1240 Use generic constraints if applicable  Instead of casting to and from the object type in generic types or methods, use where contraints or the as operator to specify the exact characteristics of the generic parameter. For example:

class SomeClass {}
// Don’t class MyClass<T>
{
    void SomeMethod(T t)
    {
        object temp = t; SomeClass obj = (SomeClass) temp;
    }
}
// Do class MyClass<T> where T : SomeClass
{
    void SomeMethod(T t)
    {
        SomeClass obj = t;
    }
}

AV1245 Don’t add extension methods to the same namespace as the extended class Even though it may seem convenient to add extension methods related to the String class in the System namespace, this may cause conflicts with future versions of the .NET Framework.

AV1250 Evaluate the result of a LINQ expression before returning it Consider the following code snippet:

```csharp
public IEnumerable<GoldMember> GetGoldMemberCustomers()
{
    const decimal GoldMemberThresholdInEuro = 1000000;
    var q = from customer in db.Customers
                where customer.Balance > GoldMemberThresholdInEuro
              select new GoldMember(customer.Name, customer.Balance);
    return q;
}
```

Since LINQ queries use deferred execution, returning q will actually return the expression tree representing the above query. Each time the caller evaluates this result using a foreach or something similar, the entire query is re-executed resulting in new instances of GoldMember every time.

Consequently, you cannot use the == operator to compare multiple GoldMember instances. Instead, always explicitly evaluate the result of a LINQ query using ToList(), ToArray() or similar methods.

5 Maintainability Guidelines

AV1500 Methods should not exceed 7 statements A method that requires more than 7 statements is doing too much, or has too many responsibilities.

It also requires the human mind to analyze the exact statements to understand what the code is doing.

Break it down in multiple small and focused methods with self-explaining names.

AV1501 Make all members private and types internal by default To make a more conscious decision on which members to make available to other classes, explicitly set the scope of all new members to private and that of a new type to internal.

Then carefully decide what to expose as a public member or type.

AV1502 Avoid conditions with double negatives Although a property like customer.HasNoOrders make sense, avoid using it in a negative condition like this:

```csharp
bool hasOrders = !customer.HasNoOrders;
```
Double negatives are more difficult to grasp than simple expressions, and people tend to read over the double negative easily.

**AV1505 Name assemblies after their contained namespace**  As an example, consider a group of classes organized under the namespace `AvivaSolutions.Web.Binding` exposed by a certain assembly.

According to this guideline, that assembly should be called `AvivaSolutions.Web.Binding.dll`.

All DLLs should be named according to the pattern `<Company>.<Component>.dll` where `<Company>` refers to your company’s name and `<Component>` contains one or more dot-separated clauses. For example `AvivaSolutions.Web.Controls.dll`.

Exception If you decide to combine classes from multiple unrelated namespaces into one assembly, consider post fixing the assembly with Core, but do not use that suffix in the namespaces. For instance, `AvivaSolutions.Consulting.Core.dll`.

**AV1506 Name a source file to the type it contains**  Also, use Pascal casing for naming the file and don’t use underscores.

**AV1507 Limit the contents of a source code file to one type**  Exception Nested types should, for obvious reasons, be part of the same file.

**AV1508 Name a source file to the logical function of the partial type**  When using partial types and allocating a part per file, name each file after the logical part that part plays. For example:

```csharp
// In MyClass.cs

public partial class MyClass {...}
```

**AV1510 Use using statements instead of fully qualified type names**  Limit usage of fully qualified type names to prevent name clashing.

For example:

Don’t:

```csharp
var list = new System.Collections.Generic.List<string>();
```

Do:

```csharp
using System.Collections.Generic;
var list = new List<string>();
```

If you do need to prevent name clashing, use a using directive to assign an alias:

```csharp
using Label = System.Web.UI.WebControls.Label;
```
AV1515 Don’t use “magic numbers”  Don’t use literal values, either numeric or strings, in your code other than to define symbolic constants. For example:

```csharp
public class Whatever {
    public static readonly Color PapayaWhip = new Color(0xFFEFD5);
    public const int MaxNumberOfWheels = 18;
}
```

Strings intended for logging or tracing are exempt from this rule. Literals are allowed when their meaning is clear from the context, and not subject to future changes. For example:

```csharp
mean = (a + b) / 2; // okay
WaitMilliseconds(waitTimeInSeconds * 1000); // clear enough
```

If the value of one constant depends on the value of another, do attempt to make this explicit in the code. For example, don’t:

```csharp
public class SomeSpecialContainer {
    public const int MaxItems = 32;
    public const int HighWaterMark = 24; // at 75%
}
```

but do:

```csharp
public class SomeSpecialContainer {
    public const int MaxItems = 32;
    public const int HighWaterMark = 3 * MaxItems / 4; // at 75%
}
```

Note An enumeration can often be used for certain types of symbolic constants.

AV1520 Only use var when the type is very obvious  Only use var as the result of a LINQ query, or if the type is very obvious from the same statement and using it would improve readability.

Don’t:

```csharp
var i = 3; // what type? int? uint? float?
var myfoo = MyFactoryMethod.Create("arg"); // Not obvious what base-class or // interface to expect. Also difficult // to refactor if you can’t search for // the class
```

Do:

```csharp
var q = from order in orders where order.Items > 10 and order.TotalValue > 1000;
var repository = new RepositoryFactory.Get<IOrderRepository>();
var list = new ReadOnlyCollection<string>();
```

In all of three above examples it is clear what type to expect. For a more detailed rationale about the advantages and disadvantages of using var, read Eric Lippert’s Uses and misuses of implicit typing.
AV1521 Initialize variables at the point of declaration Avoid the C and Visual Basic styles where all variables have to be defined at the beginning of a block, but rather define and initialize each variable at the point where it is needed.

AV1523 Favor Object and Collection Initializers over separate statements Instead of:

```csharp
var startInfo = new ProcessStartInfo("myapp.exe");
startInfo.StandardOutput = Console.Output;
startInfo.UseShellExecute = true;
```

Use Object Initializers:

```csharp
var startInfo = new ProcessStartInfo("myapp.exe")
    {
        StandardOutput = Console.Output,
        UseShellExecute = true
    };
```

Similarly, instead of:

```csharp
var countries = new List<string>();
countries.Add("Netherlands");
countries.Add("United States");
```

Use Collection Initializers:

```csharp
var countries = new List<string> { "Netherlands", "United States" };
```

AV1525 Don’t make explicit comparisons to true or false It is usually bad style to compare a bool-type expression to true or false. For example:

```csharp
while (condition == false) // wrong; bad style while (condition != true) // also wrong
while (((condition == true) == true) == true) // where do you stop? while (condition) // OK
```

AV1526 Use an enumeration instead of a list of strings if the list of values is finite If a variable can have a limited set of constant string values, use an enumeration for defining the valid values. Using the enumeration instead of a constant string allows compile-time checking and prevents typos.

AV1530 Don’t change a loop variable inside a for or foreach loop Updating the loop variable within the loop body is generally considered confusing, even more so if the loop variable is modified in more than one place. Although this rule also applies to foreach loops, an enumerator will typically detect changes to the collection the foreach loop is iteration over:

```csharp
for (int index = 0; index < 10; ++index) { if (some condition)
    index = 11; // Wrong! Use 'break' or 'continue' instead. }
```
AV1532 Don’t use nested loops in a method  A method that nests loops is more difficult to understand than one with only a single loop. In fact, in most cases having nested loops can be replaced with a much simpler LINQ query that uses the from keyword twice or more to join the data.

AV1535 Add a block after all flow control keywords, even if it is empty  Please note that this also avoids possible confusion in statements of the form:

```csharp
if (b1)
    if (b2)
        Foo();
else Bar(); // which ‘if’ goes with the ‘else’?
```

The right way:

```csharp
if (b1)
{
    if (b2)
    {
        Foo();
    } else
    {
        Bar();
    }
}
```

AV1536 Always add a default block after the last case in a switch statement  Add a descriptive comment if the default block is supposed to be empty.

Moreover, if that block is not supposed to be reached throw an InvalidOperationException to detect future changes that may fall through the existing cases.

This ensures better code, because all paths the code can travel has been thought about:

```csharp
void Foo(string answer)
{
    switch (answer)
    {
    case "no": Console.WriteLine("You answered with No"); break;
    case "yes": Console.WriteLine("You answered with Yes"); break;
    default: // Not supposed to end up here.
        throw new InvalidOperationException("Unexpected answer: " + answer);
    }
}
```

AV1537 Finish every if-else-if statement with an else-part  The intention of this rule, which applies to else-if constructs, is the same as the previous rule. For example:

```csharp
void Foo(string answer)
{
    if (answer == "no")
    {
        Console.WriteLine("You answered with No");
    }
```
} else if (answer == "yes") {
        Console.WriteLine("You answered with Yes");
} else {
        // What should happen when this point is reached? Ignored? If not, // throw an InvalidOperationException.
    }

AV1540 Be reluctant with multiple return statements  One entry, one exit is a sound principle and keeps control flow readable.

However, if the method is very small and complies with guideline AV1500 then multiple return statements may actually improve readability over some central Boolean flag that is updated at various points.

AV1545 Don’t use selection statements instead of a simple assignment or initialization  Express your intentions directly. For example, rather than:

bool pos;  
if (val > 0) { pos = true; } else { pos = false; }

write:

bool pos = (val > 0); // initialization

AV1546 Prefer conditional statements instead of simple if-else constructs  For example, instead of:

string result;  
if (someString != null) {
        result = someString;
    } else {
        result = "Unavailable";
    }  
return result;

write:

return someString ?? "Unavailable";

AV1547 Encapsulate complex expressions in a method or property  Consider the following example:

if (member.HidesBaseClassMember && (member.NodeType != NodeType.InstanceInitializer)) {
        // do something
    }
In order to understand what this expression is about, you need to analyze its exact details and all the possible outcomes. Obviously, you could add an explanatory comment on top of it, but it is much better to replace this complex expression with a clearly named method:

```csharp
if (NonConstructorMemberUsesNewKeyword(member))
{
    // do something
}
```

```csharp
private bool NonConstructorMemberUsesNewKeyword(Member member)
{
    return (member.HidesBaseClassMember
     && (member.NodeType != NodeType.InstanceInitializer)
    )
}
```

You still need to understand the expression if you are modifying it, but the calling code is now much easier to grasp.

**AV1551 Call the most overloaded method from other overloads**  This guideline only applies to overloads that are intended for providing optional arguments. Consider for example the following code snippet:

```csharp
public class MyString
{
    private string someText;
    public MyString(string text) { this.someText = text; }
    public int IndexOf(string phrase)
    {
        return IndexOf(phrase, 0, someText.Length);
    }

    public int IndexOf(string phrase, int startIndex)
    {
        return IndexOf(phrase, startIndex, someText.Length - startIndex);
    }

    public virtual int IndexOf(string phrase, int startIndex, int count)
    {
        return someText.IndexOf(phrase, startIndex, count);
    }
}
```

The class MyString provides three overloads for the IndexOf method, but two of them simply call the one with the most arguments. Notice that the same rule applies to class constructors; implement the most complete overload and call that one from the other overloads using the `this()` operator.

Also notice that the parameters with the same name should appear in the same position in all overloads.

Important If you also want to allow derived classes to override these methods, define the most complete overload as a protected virtual method that is called by all overloads.

**AV1553 Only use optional parameters to replace overloads**  The only valid reason for using C# 4.0’s optional parameters is to replace the example from rule AV1551 with a single method like::

```csharp
public virtual int IndexOf(string phrase, int startIndex = 0, int count = 0)
{
    return someText.IndexOf(phrase, startIndex, count);
}
```
If the optional parameter is a reference type then it can only have a default value of null.

But since strings, lists and collections should never be null according to rule AV1, you must use overloaded methods instead.

Note The default values of the optional parameters are stored at the caller side.

As such, changing the default value without recompiling the calling code will not apply the new default value properly.

**AV1555 Avoid using named parameters**   C# 4.0’s named parameters have been introduced to make it easier to call COM components that are known for offering tons of optional parameters.

If you need named parameters to improve the readability of the call to a method, that method is probably doing too much and should be refactored.

The only exception where named parameters improve readability is when a constructor that yields a valid object is called like this:

```csharp
Person person = new Person
{
    firstName: "John",
    lastName: "Smith",
    dateOfBirth: new DateTime(1970, 1, 1)
};
```

**AV1561 Avoid methods with more than three parameters**   If you end up with a method with more than three parameters, use a structure or class for passing multiple parameters such as explained in the Specification design pattern.

In general, the fewer the number of parameters, the easier it is to understand the method.

Additionally, unit testing a method with many parameters requires many scenarios to test.

**AV1562 Don’t use ref or out parameters**   Ref and out parameters make code less understandable and therefore may introduce bugs. Prefer returning compound objects instead.

**AV1564 Avoid methods that take a bool flag**   A flag parameter based on a bool is not self-explanatory. Consider the following method signature:

```csharp
public Customer CreateCustomer(bool platinumLevel) {}  
```

On first sight this signature seems perfectly fine, but when calling this method you will lose this purpose completely:

```csharp
Customer customer = CreateCustomer(true);
```

Often, a method taking such a flag is doing more than one thing and needs to be refactored into two or more methods. An alternative solution is to replace the flag with an enumeration.

**AV1668 Don’t use parameters as temporary variables**   Never use a parameter as a convenient variable for storing temporary state.

Even though the type of your temporary variable may be the same, the name usually does not reflect the purpose of the temporary variable.
AV1570 Always check the result of an as operation  If you use as to obtain a certain interface reference from an object, always ensure that this operation does not return null.
Failure to do so may cause a NullReferenceException at a much later stage if the object did not implement that interface.

AV1575 Don’t comment-out code  Never check-in code that is commented-out, but instead use a work item tracking system to keep track of some work to be done.
Nobody knows what to do when they encounter a block of commented-out code.
  - Was it temporarily disabled for testing purposes ?
  - Was it copied as an example ?
  - Should I delete it ?

AV1580 Consider abstracting an external dependency or 3rd party component  If your code relies on some kind of external class, service or UI control, consider wrapping that dependency in a lightweight wrapper that only exposes the members that are really used.
Such a wrapper smoothens the changes required when replacing that dependency with another, but can also be used to hide any undesired behavior or bugs that you don’t have influence on.

6 Naming Guidelines

AV1701 Use proper US-English  All identifiers should be named using words from the American English language.
  - Choose easily readable identifier names. For example, HorizontalAlignment is more readable than AlignmentHorizontal.
  - Favor readability over brevity. The property name CanScrollHorizontally is better than ScrollableX (an obscure reference to the X-axis).
  - Avoid using identifiers that conflict with keywords of widely used programming languages.
Exception In most projects, you will use words and phrases from your domain and names specific to your company. Visual Studio’s Static Code Analysis will perform a spelling check on all code, so you may need to add those terms to a Custom Code Analysis Dictionary.

AV1702 Use proper casing for members

AV1704 Don’t include numbers in identifiers  Numbers in names of fields, variables or members are very rarely needed. In fact, in most cases they are a lazy excuse for not defining a clear and intention-revealing name.

AV1705 Don’t prefix member fields  Don’t use a prefix for field names. For example, Don’t use g_ or s_ to distinguish static versus non-static fields.
In general, a method in which it is difficult to distinguish local variables from member fields, is too big.
Examples of incorrect identifier names are: _currentUser, mUserName, m_loginTime.
AV1706 Don’t use abbreviations or acronyms as parts of identifier names.

For example, use OnButtonClick rather than OnBtnClick. Avoid single character variable names, such as i or q. Use index or query instead.

Exceptions: Use well-known abbreviations that are widely accepted or well-known within the domain you work. For instance, use UI instead of UserInterface.

AV1707 Name an identifier according its meaning and not its type

- Use semantically interesting names rather than language-specific keywords for type names. For example, GetLength is a better name than GetInt.
- Don’t use terms like Enum, Class or Struct in a name.
- Identifiers that refer to an array or collection should have a plural name.

AV1708 Name types using nouns, noun phrases or adjective phrases

Bad examples include SearchExamination (a page for searching for examinations), Common (does not end with a noun, and does not explain its purpose) and SiteSecurity (although the name is technically okay, it does not say anything about its purpose).

Good examples include BusinessBinder, SmartTextBox, or EditableSingleCustomer.

Don’t include terms like Utility or Helper in classes. Classes with a name like that are usually static classes and are introduced without considering the object-oriented principles.

AV1709 Name generic type parameters with descriptive names

The following guidelines cover selecting the correct names for generic type parameters.

- Name generic type parameters with descriptive names, unless a single-letter name is completely self explanatory and a descriptive name would not add value. For example: IDictionary is an example of an interface that follows this guideline.
- Use the letter T as the type parameter name for types with one single-letter type parameter.
- Prefix descriptive type parameter names with the letter T.
- Consider indicating constraints placed on a type parameter in the name of parameter. For example, a parameter constrained to ISession may be called TSession.

AV1710 Don’t repeat the name of a class or enumeration in its members

class Employee
{
    // Wrong!
    static GetEmployee() {}
    DeleteEmployee() {}
    // Right
    static Get() {...}
    Delete() {...}
    // Also correct.
    AddNewJob() {...}
    RegisterForMeeting() {...}
}
**AV1711 Name members similarly to members of .NET Framework classes** Stay close to the naming philosophy of the .NET Framework.

Developers are already accustomed to the naming patterns .NET Framework classes use, so following this same pattern helps them find their way in your classes as well.

For instance, if you define a class that behaves like a collection, provide members like Add, Remove and Count instead of AddItem, Delete or NumberOfItems.

**AV1712 Avoid short names or names that can be mistaken with other names** Although technically allowed, the following statement is quite confusing.

```csharp
bool b001 = (lo == 10) ? (I1 == 11) : (lOl != 101);
```

**AV1715 Properly name properties**

- Do name properties with nouns, noun phrases, or occasionally adjective phrases.
- Do name boolean properties with an affirmative phrase. E.g. CanSeek instead of CantSeek.
- Consider prefixing Boolean properties with Is, Has, Can, Allows, or Supports.
- Consider giving a property the same name as its type. When you have a property that is strongly typed to an enumeration, the name of the property can be the same as the name of the enumeration. For example, if you have an enumeration named CacheLevel, a property that returns one of its values can also be named CacheLevel.

**AV1720 Name methods using verb-object pair** Name methods using a verb-object pair such as ShowDialog. A good name should give a hint on the what of a member, and if possible, the why. Also, don’t include And in the name of the method. It implies that the method is doing more than one thing, which violates the single responsibility principle.

```csharp
interface IEmployeeRepository
{
    Employee[] First() {} // Wrong: What does first mean? How many?
    Employee[] GetFirstFive() {} // Better
    Employee[] GetFiveMostRecent(){} // Best: self-describing
    void Add(Employee employee) {} // Although not using verb-object pair; // the type name is clear
}
```

**AV1725 Name namespaces according a well-defined pattern** All namespaces should be named according to the pattern:

```
<Company>.(<Product>|<Technology>)[.|<Feature>][.|<Subnamespace>]
```

For instance: Microsoft.WindowsMobile.DirectX.

**Note:** Don’t use the same name for a namespace and a type in that namespace. For example, don’t use Debug for a namespace name and also provide a class named Debug in the same namespace.
AV1735 Use a verb or verb phrase to name an event  Name events with a verb or a verb phrase. For example: Click, Deleted, Closing, Minimizing, and Arriving. For example, the declaration of the Search event may look like this:

    public event SearchEventHandler Search;

AV1737 Use -ing and -ed to express pre-events and post-events  Give event names a concept of before and after, using the present and past tense. For example, a close event that is raised before a window is closed would be called Closing and one that is raised after the window is closed would be called Closed.

Don’t use Before or After prefixes or suffixes to indicate pre and post events. Suppose you want to define events related to the deletion process of an object. Avoid defining the Deleting and Deleted events as BeginDelete and EndDelete.

Define those events as follows:

- Deleting: Occurs just before the object is getting deleted
- Delete: Occurs when the object needs to be deleted by the event handler.
- Deleted: Occurs when the object is already deleted.

AV1738 Prefix an event handler with On  It is good practice to prefix the method that handles an event with On. For example, a method that handles the Closing event could be named OnClosing.

AV1745 Group extension methods in a class suffixed with Extensions  If the name of an extension method conflicts with another member or extension method, you must prefix the call with the class name. Having them in a dedicated class with the Extensions suffix improves readability.

7 Performance Guidelines

AV1800 Consider using Any() to determine whether an IEnumerable<T> is empty  When a method or other member returns an IEnumerable<T> or other collection class that does not expose a Count property, use the Any() extension method rather than Count() to determine whether the collection contains items.

If you do use Count(), you risk that iterating over the entire collection might have a significant impact (such as when it really is an IQueryable<T> to a persistent store).

8 Framework Guidelines

AV2201 Use C# type aliases instead of the types from the System namespace  For instance, use object instead of Object, string instead of String, and int instead of Int32.

These aliases have been introduced to make the primitive types a first class citizen of the C# language, so use them accordingly.

Exception  When referring to static members of those types, it is custom to use the full CLS name, e.g. Int32.Parse() instead of int.Parse().
AV2205 Properly name identifiers referring to localized resources  The guidelines in this topic apply to localizable resources such as error messages and menu text.

- Use Pascal casing in resource keys.
- Provide descriptive rather than short identifiers. Keep them concise where possible, but don't sacrifice readability.
- Use only alphanumeric characters in naming resources.

AV2207 Don't hardcode strings that change based on the deployment  Examples include connection strings, server addresses, etc.

Use Resources, the ConnectionStrings property of the ConfigurationManager class, or the Settings class generated by Visual Studio.

AV2210 Build with the highest warning level  Configure the development environment to use Warning Level 4 for the C# compiler, and enable the option Treat warnings as errors.

This allows the compiler to enforce the highest possible code quality.

AV2211 Avoid suppressing specific compiler warnings

AV2215 Properly fill the attributes of the AssemblyInfo.cs file  Ensure that the attributes for the company name, description, copyright statement, version, etc. are filled.

One way to ensure that version and other fields that are common to all assemblies have the same values, is to move the corresponding attributes out of the AssemblyInfo.cs into a SolutionInfo.cs file that is shared by all projects within the solution.

AV2220 Avoid LINQ for simple expressions  Rather than:

```csharp
var query = from item in items where item.Length > 0;
```

prefer using the extension methods from the System.Linq namespace:

```csharp
var query = items.Where(i => i.Length > 0);
```

Since LINQ queries should be written out over multiple lines for readability, the second example is a bit more readable.

AV2221 Use Lambda expressions instead of delegates  Lambda expressions have been introduced in C# 3.0 and provide a much more elegant alternative for anonymous delegates.

So instead of:

```csharp
Customer c = Array.Find(customers,
                   delegate(Customer c) { return c.Name == "Tom";})
```

use an Lambda expression:

```csharp
Customer c = Array.Find(customers, c => c.Name == "Tom");
```
Or even better:

```csharp
var customer = customers.Where(c => c.Name == “Tom”);
```

**AV2230 Only use the dynamic keyword when talking to a dynamic object**  The dynamic keyword has been introduced for working with dynamic languages.

Using it introduces a serious performance bottleneck because the compiler has to generate some complex Reflection code.

Use it only for calling methods or members of a dynamically created instance (using the Activator) class as an alternative to `Type.GetProperty()` and `Type.GetMethod()`, or for working with COM Interop types.

### 9 Documentation Guidelines

**AV2301 Write comments and documentation in US English**

**AV2305 Use XML tags for documenting types and members**  Document all public types and members of types using the built-in XML comment functionality of Visual Studio.

Documenting your code allows Visual Studio to pop-up the documentation when your class is used somewhere else. Furthermore, by properly documenting your classes, tools can generate professionally looking class documentation.

**AV2306 Write XML documentation with the caller in mind**  Write the documentation of your class with the class user in mind. Assume the user will not have access to the source code and try to explain how to get the most out of the functionality of your class.

**AV2307 Write MSDN-style documentation**  Following the MSDN on-line help style and word choice helps the developer to find its way through your documentation more easily.

**Tip**  The tool GhostDoc can generate a starting point for documenting code with a shortcut key.

**AV2310 Avoid inline comments**  If you feel the need to explain a block of code using a comment, consider replacing that block with a method having a clear name.

**AV2315 Don’t use /* */ for comments**

**AV2316 Only write comments to explain complex algorithms or decisions**  Try to focus comments on the why and what of a code block and not the how.

Avoid explaining the statements in words, but instead help the reader understand why you chose a certain solution or algorithm and what you are trying to achieve.

If applicable, also mention that you chose an alternative solution because you ran into a problem with the obvious solution.
AV2318 Don’t use comments for tracking work to be done later Annotating a block of code or some work to be done using a TODO or similar comment may seem a reasonable way of tracking work-to-be-done. But in reality, nobody really searches for comments like that. Use a work item tracking system such as Team Foundation Server to keep track of left overs.

10 Layout Guidelines

AV2400 Use a common layout

- Keep the length of each line under 130 characters.
- Use an indentation of 4 whitespaces, and don’t use Tabs
- Keep one whitespace between keywords like if and the expression, but don’t add whitespaces after ( and before ) such as:
  ```csharp
  if (condition == null)
  ```
- Add a whitespace around operators, like +, -, ==, etc.
- Always succeed the keywords if, else, do, while, for and foreach, with opening and closing parentheses, even though the language does not require it.
- Always put opening and closing parentheses on a new line.
- Don’t indent object Initializers and initialize each property on a new line, so use a format like this:
  ```csharp
  var dto = new ConsumerDto()
  {
    Id = 123,
    Name = "Microsoft",
    PartnerShip = PartnerShip.Gold,
  }
  ```
- Don’t indent lambda statements and use a format like this:
  ```csharp
  methodThatTakesAnAction.Do(x =>
  {
    // do something like this
  })
  ```
- Put the entire LINQ statement on one line, or start each keyword at the same indentation, like this:
  ```csharp
  var query = from product in products where product.Price > 10 select product;
  ```
  ```csharp
  or:
  ```
  var query =
  from product in products
  where product.Price > 10
  select product;
  ```
- Add braces around every comparison condition, but don’t add braces around a singular condition. For example:
  ```csharp
  if (!string.IsNullOrEmpty(str) && (str != "new"))
  ```
- Add an empty line between multi-line statements, between members, after the closing parentheses, between unrelated code blocks, around the #region keyword, and between the using statements of different companies.
AV2402 Order and group namespaces according the company

// Microsoft namespaces are first
using System;
using System.Collections;
using System.Xml;
// Then any other namespaces in alphabetic order
using AvivaSolutions.Business;
using AvivaSolutions.Standard;
using Telerik.WebControls;
using Telerik.Ajax;

AV2406 Place members in a well-defined order   Maintaining a common order allows other team members to find their way in your code more easily.

In general, a source file should be readable from top to bottom, as if you are reading a book.
This prevents readers from having to browse up and down through the code file.

1. Private fields and constants (in a region)
2. Public constants
3. Public read-only static fields
4. Constructors and the Finalizer
5. Events
6. Properties
7. Other members grouped in a functional manner.
8. Private properties

Private and protected methods should be placed behind the public member in calling order.

AV2407 Be reluctant with #regions   Regions can be helpful, but can also hide the main purpose of a class.
Therefore, use #regions only for:

- Private fields and constants (preferably in a Private Definitions region).
- Nested classes
- Interface implementations (only if the interface is not the main purpose of that class)

11 Important Resources

11.1 The companion website   See Also:

This document is part of an effort to increase the consciousness with which C# developers do their daily job on a professional level.
Therefore I’ve started a dedicated CodePlex site that can be easily found using the URL http://www.csharpcodingguidelines.com.
In addition to the most up to date version of this document, you’ll find:
• A companion quick-reference sheet
• Visual Studio 2010 Rule Sets for different types of systems.
• ReSharper 5 layout configurations matching the rules in chapter 8.
• A place to start discussions on C# coding quality.

11.2 Useful links
In addition to the many links provided throughout this document, I’d like to recommend the following books, articles and sites for everyone interested in software quality.

**Code Complete: A Practical Handbook of Software Construction (Steve McConnel)  See Also:**
http://www.amazon.com/dp/0735619670/ref=rdr_ext_tmb
One of the best books I’ve ever read.
It deals with all aspects of software development, and even though the book was originally written in 2004, but you’ll be surprised when you see how accurate it still is. I wrote a review in 2009 if you want to get a sense of its contents.

**The Art of Agile Development (James Shore)  See Also:**
Another great all-encompassing trip through the many practices preached by processes like Scrum and Extreme Programming. If you’re looking for a quick introduction with a pragmatic touch, make sure you read James’ book.

**Applying Domain Driven-Design and Patterns: With Examples in C# and .NET (Jimmy Nilsson)  See Also:**
http://www.amazon.com/Applying-Domain-Driven-Design-Patterns/dp/0321268202
The book that started my interest for both Domain Driven Design and Test Driven Development. It’s one of those books that I wished I had read a few years earlier.
It would have saved me from many mistakes.

**Jeremy D. Miller’s Blog  See Also:**
http://codebetter.com/jeremymiller/
Although he is not that active anymore, in the last couple of years he has written some excellent blog posts on Test Driven Development, Design Patterns and design principles. I’ve learned a lot from his real-life and practical insights.

**LINQ Framework Design Guidelines  See Also:**
A set of rules and recommendations that you should adhere to when creating your own implementations of IQueryable<T>.

**C code project coding standards**
Naming Conventions  .NET standards do not recommend using Hungarian prefixes, so avoid them.

To refer to member field inside class member function, use this word `this.memberVariable`.

You use lower case for variables and function parameters, all the rest go with capitals:

```csharp
SomeClass { ... };  
SomeEnum { Item1, Item2, ... };  
SomeProperty { get; set }  

double someVariable;  
int someFunctionParameter;  
const char SomeConstantVariable;  
static readonly SomeReadOnlyStaticVariable;  
Exception SomeNameException;  
Attribute SomeNameAttribute;  
Button cancelButton;  
TextBox nameTextBox;  
PictureBox namePictureBox;  
```

Class Layout  It is recommended to declare class internals in that order:

- Fields
- Constructors
- Nested enums, structs, classes
- Properties
- Methods

Miscellaneous  One class per source file (SomeName class should be declared in somename.cs file).

Private members are private by default in class, avoid explicit private.

For a single statement get/set property use that construction:

```csharp
public int Foo  
{  
    get { return this.foo; }  
    set { this.foo = value; }  
}  
```

Use {} for if, else, for, while even if single lined:

```csharp
if (someVar == true)  
{  
    foo++;  
}  
```
Use @ instead of escape sequencies in strings - @"c:somepathfile.txt"

Use override instead of new:

```csharp
class SomeParent
{
    //...
    public int DoWork();
    //...
}
class SomeOverride
{
    //...
    public override int DoWork();
    //...
}
```

Initialize string to String.Empty rather than assigning it "".

Use StringBuilder for composing complex strings rather than using string class operators.

Document if the method returns copy of reference type.

Do not compare to true or false use if(condition).

Provide private constructor if only static fields are in the class or declare static class.

Const objects in class are static by default.

Struct may have constructor with parameters only.

**Documenting**  Consider documenting your code so you will not be trying to recall what the method is supposed to do after some days have gone:

```csharp
/// <summary>
/// Save trimmed text
/// </summary>
/// <param name="fileName">Trimmed text file name</param>
/// <returns>zero upon success</returns>
public int SaveText(String fileName)
{
    //...
}
```

**C monoproject coding standards**  See Also:

http://www.mono-project.com/Coding_Guideline

**C stylecop coding standards**  See Also:

http://stylecop.codeplex.com/
StyleCop Governance  Microsoft has turned over governance and coordination of the StyleCop project to the community.

The new Coordinator of this project is responsible for managing contributions.

Microsoft is not granting any IP rights to code or other material contributed to this project by third parties. The IP rights to each particular contribution are granted to you by the respective contributor under the Ms-PL license.

Project Description  StyleCop analyzes C# source code to enforce a set of style and consistency rules.

It can be run from inside of Visual Studio or integrated into an MSBuild project.

StyleCop has also been integrated into many third-party development tools (shardevelop).

C modules

C modules configuration

C ini-parser modules  See Also:
http://code.google.com/p/ini-parser/

This library allows to read or create INI data programatically.

An implementation for reading / writing INI data to and from streams, files and strings is included.

Really simple to use:

```csharp
// Load ini file
FileIniDataParser parser = new FileIniDataParser();

IniData data = parser.LoadFile("TestIniFile.ini");

// Retrieve value for key 'fullscreen' inside a config section
string useFullScreen? = data["ConfigSection"]['fullscreen'];

// Modify the value
data["ConfigSection"]['fullscreen'] = true;

// save new ini file
parser.SaveFile("NewTestIniFile.ini");
```

C mono

c mono  See Also:
- http://www.mono-project.com
- http://tirania.org/blog/
Release Notes_Mono_2.12  See Also:
http://www.mono-project.com/Release_Notes_Mono_2.12

Major Highlights
  • C# 5.0 support - the complete asynchronous support is now part of Mono
  • Portable Class Libraries
  • Updated Unicode tables, fixes long-standing 480178
  • System.Json is now available for everyone, even if originally limited to Silverlight
  • System.Threading.Tasks.Dataflow (aka TPL Dataflow) preview is now available
  • Alexander Chebaturkin has implemented initial version of static Code Contract analyser as part of SoC 2011.
  • Introducing the 4.5 preview API, a strict superset of the 4.0 API designed to better support asynchronous programming.

Mono applications

c mono applications

gnomesubtitles  See Also:
  • http://gnomesubtitles.org/

C people

C people

Jon skeet  See Also:
  • https://twitter.com/#!/jonskeet
  • http://csharpindepth.com/

C books

C books

C in depth  See Also:
  • http://www.twitter.com/jonskeet

About the Author  Jon Skeet is a Google software engineer working in London.
A C# MVP since 2003 and prominent C# community personality, Jon’s heart belongs to C#.
C# 4 is even more expressive and powerful than earlier versions. You can do amazing things with generics, lambda expressions, dynamic typing, LINQ, iterator blocks, and other features—but you first have to learn C# in depth.

C# in Depth, Second Edition is a thoroughly revised, up-to-date book that covers the new features of C# 4 as well as Code Contracts.

In it, you’ll see the subtleties of C# programming in action, learning how to work with high-value features that you’ll be glad to have in your toolkit.

The book helps readers avoid hidden pitfalls of C# programming by understanding “behind the scenes” issues.

**C bindings**

**c bindings**

**CXXI: Briding the C++ and C# worlds**

The Mono runtime engine has many language interoperability features but has never had a strong story to interop with C++.

Thanks to the work of Alex Corrado, Andreia Gaita and Zoltan Varga, this is about to change.

The short story is that the new CXXI technology allows C#/.NET developers to:

- Easily consume existing C++ classes from C# or any other .NET language
- Instantiate C++ objects from C#
- Invoke C++ methods in C++ classes from C# code
- Invoke C++ inline methods from C# code (provided your library is compiled with -fkeep-inline-functions or that you provide a surrogate library)
- Subclass C++ classes from C#
- Override C++ methods with C# methods
- Expose instances of C++ classes or mixed C++/C# classes to both C# code and C++ as if they were native code.

CXXI is the result of two summers of work from Google’s Summer of Code towards improving the interoperability of Mono with the C++ language.

**The Alternatives**

This section is merely a refresher of of the underlying technologies for interoperability supported by Mono and how developers coped with C++ and C# interoperability in the past. You can skip it if you want to get to how to get started with CXXI.

As a reminder, Mono provides a number of interoperability bridges, mostly inherited from the ECMA standard. These bridges include:

- The bi-directional “Platform Invoke” technology (P/Invoke) which allows managed code (C#) to call methods in native libraries as well as support for native libraries to call back into managed code.
• COM Interop which allows Mono code to transparently call C or C++ code defined in native libraries as long as the code in the native libraries follows a few COM conventions.

• A general interceptor technology that can be used to intercept method invocations on objects.

How CXXI Works  Accessing C++ methods poses several challenges. Here is a summary of the components that play a major role in CXXI:

• Object Layout: this is the binary layout of the object in memory. This will vary from platform to platform.

• VTable Layout: this is the binary layout that the C++ compiler will use for a given class based on the base classes and their virtual methods.

• Mangled names: non-virtual methods do not enter an object vtable, instead these methods are merely turned into regular C functions. The name of the C functions is computed based on the return type and the parameter types of the method. These vary from compiler to compiler.

For example, given this C++ class definition, with its corresponding implementation:

class Widget {
public:
    void SetVisible (bool visible);
    virtual void Layout (){
    virtual void Draw (){
};

class Label : public Widget {
public:
    void SetText (const char *text);
    const char *GetText (){
};

The C++ compiler on my system will generate the following mangled names for the SetVisible, Layout, Draw, SetText and GetText methods:

__ZN6Widget10SetVisibleEb
__ZN6Widget6LayoutEv
__ZN6Widget4DrawEv
__ZN5Label7SetTextEPKc
__ZN5Label7GetTextEv

The following C++ code:

Label *l = new Label ();
l->SetText ("foo");
l->Draw ();

Is roughly compiled into this (rendered as C code):

Label *l = (Label *) malloc (sizeof (Label));
ZN5Label1ClEv (l); // Mangled name for the Label’s constructor
__ZN5Label7SetTextEPKc (l, "foo");

// This one calls draw
(l->vtable [METHOD_PTR_SIZE*2])();

For CXXI to support these scenarios, it needs to know the exact layout for the vtable, to know where each method lives and it needs to know how to access a given method based on their mangled name.

The following chart explains shows how a native C++ library is exposed to C# or other .NET languages:

![Chart explaining how a native C++ library is exposed to C# or other .NET languages](image)

Your C++ source code is compiled twice. Once with the native C++ compiler to generate your native library, and once with the CXXI toolchain.

Technically, CXXI only needs the header files for your C++ project, and only the header files for the APIs that you are interested in wrapping. This means that you can create bindings for C++ libraries that you do not have the source code to, but have its header files.

The CXXI toolchain produces a .NET library that you can consume from C# or other .NET languages. This library exposes a C# class that has the following properties:

- When you instantiate the C# class, it actually instantiates the underlying C++ class.
- The resulting class can be used as the base class for other C# classes. Any methods flagged as virtual can be overwritten from C#.
- Supports C++ multiple inheritance: The generated C# classes expose a number of cast operators that you can use to access the different C++ base classes.
- Overwritten methods can call use the “base” C# keyword to invoke the base class implementation of the given method in C++.
- You can override any of the virtual methods from classes that support multiple inheritance.
A convenience constructor is also provided if you want to instantiate a C# peer for an existing C++ instance that you surfaced through some other mean.

This is pure gold  The CXXI pipeline in turn is made up of three components, as shown in the diagram on the right. The GCC-XML compiler is used to parse your source code and extract the vtable layout information. The generated XML information is then processed by the CXXI tooling to generate a set of partial C# classes that contain the bridge code to integrate with C++.

This is then combined with any customization code that you might want to add (for example, you can add some overloads to improve the API, add a ToString() implementation, add some async front-ends or dynamic helper methods). The result is the managed assembly that interfaces with the native static library.

It is important to note that the resulting assembly (Foo.dll) does not encode the actual in-memory layout of the fields in an object. Instead, the CXXI binder determines based on the ABI being used what the layout rules for the object are. This means that the Foo.dll is compiled only once and could be used across multiple platforms that have different rules for laying out the fields in memory.

Future Work  CXXI is not finished, but it is a strong foundation to drastically improve the interoperability between .NET managed languages and C++.

Currently CXXI achieves all of its work at runtime by using System.Reflection.Emit to generate the bridges on demand. This is useful as it can dynamically detect the ABI used by a C++ compiler.

One of the projects that we are interested in doing is to add support for static compilation, this would allow PS3 and iPhone users to use this technology. It would mean that the resulting library would be tied to the platform on which the CXXI tooling was used.

CXXI currently implements support for the GCC ABI, and has some early support for the MSVC ABI. Support for other compiler ABIs as well as for completing the MSVC ABI is something that we would like help with.

Currently CXXI only supports deleting objects that were instantiated from managed code. Other objects are assumed to be owned by the unmanaged world. Support for the delete operator is something that would be useful.

We also want to better document the pipeline, the runtime APIs and improve the binding.

C

How to make a callback to Csharp from C Cplusplus  See Also:

http://www.codeproject.com/Tips/318140/How-to-make-a-callback-to-Csharp-from-C-Cplusplus

coffeescript language

See Also:

http://jashkenas.github.com/coffee-script/
https://github.com/jashkenas/coffee-script/wiki
https://github.com/jashkenas/coffee-script/wiki/FAQ
http://stackoverflow.com/tags/coffeescript

javascript language
Figure 2.24: CXXI pipeline in turn is made up of three components
CoffeeScript is a little language that compiles into JavaScript. Underneath all of those embarrassing braces and semicolons, JavaScript has always had a gorgeous object model at its heart. CoffeeScript is an attempt to expose the good parts of JavaScript in a simple way.

The golden rule of CoffeeScript is: **It's just JavaScript**. The code compiles one-to-one into the equivalent JS, and there is no interpretation at runtime. You can use any existing JavaScript library seamlessly (and vice-versa).

The compiled output is readable and pretty-printed, passes through JavaScript Lint without warnings, will work in every JavaScript implementation, and tends to run as fast or faster than the equivalent handwritten JavaScript.

### Installation and Usage

The CoffeeScript compiler is itself written in CoffeeScript, using the Jison parser generator. The command-line version of coffee is available as a Node.js utility. The core compiler however, does not depend on Node, and can be run in any JavaScript environment, or in the browser (see “Try CoffeeScript”, above).

To install, first make sure you have a working copy of the latest stable version of Node.js, and npm (the Node Package Manager). You can then install CoffeeScript with npm:

```
npm install -g coffee-script
```

(Leave off the -g if you don’t wish to install globally.)

If you’d prefer to install the latest master version of CoffeeScript, you can clone the CoffeeScript source repository from GitHub, or download the source directly. To install the CoffeeScript compiler system-wide under /usr/local, open the directory and run:

```
sudo bin/cake install
```

If installing on Ubuntu or Debian, be careful not to use the existing out-of-date package. If installing on Windows, your best bet is probably to run Node.js under Cygwin. If you’d just like to experiment, you can try the CoffeeScript Compiler For Windows.

Once installed, you should have access to the coffee command, which can execute scripts, compile .coffee files into .js, and provide an interactive REPL.

### Language Reference

This reference is structured so that it can be read from top to bottom, if you like. Later sections use ideas and syntax previously introduced. Familiarity with JavaScript is assumed. In all of the following examples, the source CoffeeScript is provided on the left, and the direct compilation into JavaScript is on the right.

Many of the examples can be run (where it makes sense) by pressing the run button on the right, and can be loaded into the “Try CoffeeScript” console by pressing the load button on the left.

First, the basics: CoffeeScript uses significant whitespace to delimit blocks of code. You don’t need to use semicolons ; to terminate expressions, ending the line will do just as well (although semicolons can still be used to fit multiple expressions onto a single line). Instead of using curly braces { } to surround blocks of code in functions, if-statements, switch, and try/catch, use indentation.

### Books and Screencasts

There are a number of excellent books and screencasts to help you get started with CoffeeScript, some of which are freely available online.
• The Little Book on CoffeeScript is a brief 5-chapter introduction to CoffeeScript, written with great clarity and precision by Alex MacCaw.

• Smooth CoffeeScript is a reimagination of the excellent book Eloquent JavaScript, as if it had been written in CoffeeScript instead. Covers language features as well as the functional and object-oriented programming styles. By E. Hoigaard.

Compilers

See Also:
http://en.wikipedia.org/wiki/Compiler

A compiler is a computer program (or set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language, often having a binary form known as object code). The most common reason for wanting to transform source code is to create an executable program.

Clang

See Also:
  • http://fr.wikipedia.org/wiki/Clang
  • http://clang.llvm.org/

Clang est un compilateur pour les langages de programmation C, C++ et Objective-C. Son interface de bas niveau utilise les bibliothèques LLVM pour la compilation.

C’est un logiciel libre issu d’un projet de recherche universitaire et distribué selon les termes de la licence Open Source NCSA/Université de l’Illinois1.

Clang est aujourd’hui maintenu par une communauté autour de Chris Lattner chez Apple dans le cadre du projet LLVM.

GCC (GNU Compiler Collection)

Contents
  • GCC (GNU Compiler Collection)
    – Présentation
    – Versions
GNU Compiler Collection, abrégé en GCC, est un ensemble de compilateurs créés par le projet GNU. Les compilateurs sont des logiciels libres intégrés capables de compiler divers langages de programmation, dont C, C++, Objective-C, Java, Ada et Fortran.

GCC est utilisé pour le développement de la plupart des logiciels libres. Le noyau Linux dépend notamment étroitement des fonctionnalités de GCC.

Présentation GCC a été conçu pour remplacer le compilateur C fourni en standard sur le système d’exploitation Unix, qui s’appelle CC. GCC signifiait à l’origine GNU C Compiler, soit le compilateur C de GNU. Comme GCC est très extensible, le support de nombreux autres langages a été ajouté et le nom officiel a été changé en GNU Compiler Collection (note : sans s à Compiler).

En pratique, l’abréviation GCC est utilisée pour nommer trois entités légèrement différentes :

- la collection complète de compilateurs (le projet GCC) ;
- la partie commune à tous les compilateurs (GCC) ;
- le compilateur C lui-même (le frontend gcc, écrit en minuscule).

Pour faire référence précisément aux compilateurs de chaque langage, on parle de :

- gcc pour C ;
- G++ pour C++ ;
- Gobjc pour Objective C
- GCJ pour Java ;
- GNAT pour Ada ;
- Gfortran pour Fortran ;
- GPC pour le Pascal;
- GHDL pour le VHDL.
- GDC pour le D
**GNU Compiler Collection (GCC)**  The GNU Compiler Collection (usually shortened to GCC) is a compiler system produced by the GNU Project supporting various programming languages. GCC is a key component of the GNU toolchain. As well as being the official compiler of the GNU operating system, GCC has been adopted as the standard compiler by most other modern Unix-like computer operating systems, including GNU/Linux, the BSD family and Mac OS X. GCC has been ported to a wide variety of processor architectures, and is widely deployed as a tool in commercial, proprietary and closed source software development environments. GCC is also available for most embedded platforms, for example Symbian, AMCC and Freescale Power Architecture-based chips. The compiler can target a wide variety of platforms, including videogame consoles such as the PlayStation 2 and Dreamcast. Several companies make a business out of supplying and supporting gcc ports to various platforms, and chip manufacturers today consider a GCC port almost essential to the success of an architecture.

Originally named the GNU C Compiler, because it only handled the C programming language, GCC 1.0 was released in 1987, and the compiler was extended to compile C++ in December of that year.[1] Front ends were later developed for Fortran, Pascal, Objective-C, Java, and Ada, among others.[7]

The Free Software Foundation (FSF) distributes GCC under the GNU General Public License (GNU GPL). GCC is widely considered a strong example of free software.

See Also:

**Gcc version**

```
C:\MinGW\bin>gcc -v
Reading specs from ./../lib/gcc/mingw32/3.4.5/specs
Thread model: win32
gcc version 3.4.5 (mingw-vista special r3)
```

**GCC 64 bits**  The project’s goal is to deliver runtime, headers, and libs for developing 64 bit (x64), as well as 32 bit (x86), windows applications using gcc-4.4 or newer versions.

The current state: It is already possible to generate applications using these headers, libs and runtime together with gcc-4.4 and up :-) There might be bugs here and there but everything is mostly complete.

Please visit the project page for releases and more details! If you want to join us in chat, we are on the OFTC network in the channel #mingw-w64. For users that are unfamiliar with IRC, we have a web based client available.

See Also:
- [http://mingw-w64.sourceforge.net/](http://mingw-w64.sourceforge.net/)
- [https://sourceforge.net/apps/trac/mingw-w64/](https://sourceforge.net/apps/trac/mingw-w64/)

**Versions**

**GCC versions**
Today the GCC development team celebrates the 25th anniversary of the GNU Compiler Collection.

When Richard Stallman announced the first public release of GCC in 1987, few could have imagined the broad impact that it has had. It has prototyped many language features that later were adopted as part of their respective standards — everything from “long long” type to transactional memory. It deployed an architecture-neutral automatic vectorization facility, OpenMP, and Polyhedral loop nest optimization.

It has provided the toolchain infrastructure for the GNU/Linux ecosystem used everywhere from Google and Facebook to financial markets and stock exchanges. We salute and thank the hundreds of developers who have contributed over the years to make GCC one of the most long-lasting and successful free software projects in the history of this industry.

As a special present we have prepared the release of GCC 4.7.0 which continues the series of free software high-quality industry-standard compilers.

GCC 4.7.0 is a major release, containing substantial new functionality not available in GCC 4.6.x or previous GCC releases.

GCC 4.7 features support for software transactional memory on selected architectures. The C++ compiler supports a bigger subset of the new ISO C++11 standard such as support for atomics and the C++11 memory model, non-static data member initializers, user-defined literals, alias-declarations, delegating constructors, explicit override and extended friend syntax. The C compiler adds support for more features from the new ISO C11 standard. GCC now supports version 3.1 of the OpenMP specification for C, C++ and Fortran.

The link-time optimization (LTO) framework has seen improvements with regards to scalability, stability and resource needs. Inlining and interprocedural constant propagation have been improved.

GCC 4.7 now supports various new GNU extensions to the DWARF debugging information format, like entry value and call site information, a typed DWARF stack and a more compact macro representation.

Extending the widest support for hardware architectures in the industry, GCC 4.7 gains support for Adapteva’s Epiphany processor, National Semiconductor’s CR16, and TI’s C6X as well as Tilera’s TILE-Gx and TILEPro families of processors. The x86 family support has been extended by the Intel Haswell and AMD Piledriver architectures. ARM has gained support for the Cortex-A7 family.

See

GDB: The GNU Project Debugger

See Also:
What is GDB?

GDB, the GNU Project debugger, allows you to see what is going on inside another program while it executes – or what another program was doing at the moment it crashed.

GDB can do four main kinds of things (plus other things in support of these) to help you catch bugs in the act:

- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.
- Examine what has happened, when your program has stopped.
- Change things in your program, so you can experiment with correcting the effects of one bug and go on to learn about another.

The program being debugged can be written in Ada, C, C++, Objective-C, Pascal (and many other languages). Those programs might be executing on the same machine as GDB (native) or on another machine (remote).

GDB can run on most popular UNIX and Microsoft Windows variants.

GDB versions

GDB 7.4.0

Hello,

A quick message to announce that the GDB 7.4 branch has just been created.

If all goes well, we should publish the GDB 7.4 release at the start of next year.
Announce  See Also:

http://www.gnu.org/software/gdb/download/ANNOUNCEMENT

Release 7.4 of GDB, the GNU Debugger, is now available via anonymous FTP. GDB is a source-level debugger for Ada, C, C++, Objective-C, Pascal and many other languages. GDB can target (i.e., debug programs running on) more than a dozen different processor architectures, and GDB itself can run on most popular GNU/Linux, Unix and Microsoft Windows variants.

You can download GDB from the GNU FTP server in the directory:


The vital stats:

<table>
<thead>
<tr>
<th>Size</th>
<th>md5sum</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>20MB</td>
<td>95a9a8305fed4d30a30a6dc28ff9d060</td>
<td>gdb-7.4.tar.bz2</td>
</tr>
<tr>
<td>27MB</td>
<td>7877875c8af7c7ef7d06d329ac961d3f</td>
<td>gdb-7.4.tar.gz</td>
</tr>
</tbody>
</table>

There is a web page for GDB at:

http://www.gnu.org/software/gdb/

That page includes information about GDB mailing lists (an announcement mailing list, developers discussion lists, etc.), details on how to access GDB’s CVS repository, locations for development snapshots, preformatted documentation, and links to related information around the net. We will put errata notes and host-specific tips for this release on-line as any problems come up. All mailing lists archives are also browsable via the web.

GDB 7.4 brings new targets, features and improvements, including

New target: Texas Instruments TMS320C6x (tic6x-)

New simulator: Renesas RL78 (rl78–elf)

New configure option --with-iconv-bin

Python scripting improvements

- The Python commands and convenience functions located in the data directory are now automatically loaded on GDB start-up.
- New command “set python print-stack nonelfullmessage”, replacing “maint set python print-stack onloff”, which has been deprecated in GDB 7.5.
- The “gdb.breakpoint” function has been deprecated in favor of “gdb.breakpoints”.
- Type objects for struct and union types now allow access to the fields using standard Python dictionary (mapping) methods.
- Four new attributes in class Block.
- Class Symbols now provides a “type” attribute.
- A prompt substitution hook, and a new gdb.prompt module.
- A new class “gdb.FinishBreakpoint”.

546 Chapter 2. Development
• A new event “gdb.new_objfile”.
• A new function, “deep_items” has been added to module gdb.types.

Changes to existing commands
• libthread-db-search-path now supports two special values: $sdir and $pdir

New commands
• “skip file”, “skip function”: To skip uninteresting functions during debugging.
• watch EXPRESSION mask MASK_VALUE
• info auto-load-scripts [REGEXP]
• info macro [-all] [-] MACRO
• collect[/s] EXPRESSIONS
• tstart [NOTES]
• tstop [NOTES]
• "!" (alias of the “shell” command)

New options
• set extended-prompt show extended-prompt
• set print entry-values (both|compact|default|if-needed|no|only|preferred) show print entry-values
• set debug entry-values show debug entry-values
• set basenames-may-differ show basenames-may-differ
• set trace-user show trace-user
• set trace-notes show trace-notes
• set trace-stop-notes show trace-stop-notes
• GDB now handles ambiguous linespecs more consistently, and set a breakpoint on all matching locations. Locations will be added or removed according to inferior changes.
• Masked Watchpoint support on PowerPC BookE running a Linux kernel (version 2.6.34 or later).
• Ability to display function parameter values at the time the function gets called (only available with code compiled with GCC 4.7 or later). See new option “set print entry-values”.

Tracepoint improvements
• Ability to enable and disable tracepoints at any time after a trace experiment has been started.
• Fast tracepoints on 32-bit x86-architectures can now be placed at locations with 4-byte instructions (the minimum was previously 5 bytes).
Remote protocol changes

- New packets: QTEnable, QTDisable, QTNotes, qTP, qTMinFPI Len,
- New commands: “set dcacheline” and “set dcacheline-size”.
- New command “set remote hardware-watchpoint-length-limit”.

GDB/MI changes

- “stopped” events can report several new “reason”s.
- Breakpoint changes are now notified using new async records.
- New command: -ada-task-info.

New GDBserver operation: --once The gdbtui binary will be deprecated starting with GDB 7.5. Use “gdb -tui” instead.

For a complete list and more details on each item, please see the gdb/NEWS file.

GDB 7.3.0, July 26, 2011 July 26, 2011: GDB 7.3 Released!
The latest version of GDB, version 7.3, is available for download.
Changes in this release include:

- Initial OpenCL C support
- C++ support enhancements
- Improved Python support
- ia64 HP-UX (native) and Blackfin (target) support
- GDBserver support for PowerPC LynxOS, i686 LynxOS, and Blackfin Linux
- Support for reading and writing a new .gdb_index section, containing a fast index of DWARF debugging info

See the NEWS file for a more complete and detailed list of what this release includes.

LLVM (Low Level Virtual Machine) / CLang

Contents

- LLVM (Low Level Virtual Machine) / CLang
  - Introduction
  - Test

See Also:

- Clang

Introduction La Low Level Virtual Machine (LLVM, en français : « machine virtuelle de bas niveau ») est une infrastructure de compilateur conçue pour optimiser la compilation, l’édition de liens, l’exécution et les temps morts dans un programme écrit dans un langage quelconque.
Test

Contents

<table>
<thead>
<tr>
<th>LLVM (Low Level Virtual Machine) test</th>
</tr>
</thead>
<tbody>
<tr>
<td>• LLVM (Low Level Virtual Machine) test</td>
</tr>
<tr>
<td>• Introduction</td>
</tr>
</tbody>
</table>

See Also:

• http://llvm.org/docs/lnt/
• http://llvm.org/perf/

Introduction

LNT is LLVM’s performance tracking software.

This is the web page for the LNT software itself, the current online version of the server is available at http://llvm.org/perf/.

Python compilers

Python compiler (pycparser)  

See Also:

• http://pypi.python.org/pypi/pycparser/2.05
• http://code.google.com/p/pycparser/

pycparser is a complete parser of the C language, written in pure Python using the PLY parsing library. It parses C code into an AST and can serve as a front-end for C compilers or analysis tools.

pycparser is a complete parser for the C language, written in pure Python. It is a module designed to be easily integrated into applications that need to parse C source code.

The following are some uses for pycparser, taken from real user reports:

• C code obfuscator
• Front-end for various specialized C compilers
• Static code checker
• Automatic unit-test discovery
• Adding specialized extensions to the C language

pycparser is also referenced in a few academic theses and papers, mainly on the topic of parallel computing, where it’s being used to translate a set of parallel extensions for the C language, down to plain C.
DSL (Domain specific language)

See Also:

In software development and domain engineering, a domain-specific language (DSL) is a programming language or specification language dedicated to a particular problem domain, a particular problem representation technique, and/or a particular solution technique. The concept isn’t new—special-purpose programming languages and all kinds of modeling/specification languages have always existed, but the term has become more popular due to the rise of domain-specific modeling.

Examples of domain-specific languages include HTML, Logo for children, Verilog and VHSIC hardware description languages, Mata for matrix programming, Mathematica and Maxima for symbolic mathematics, spreadsheet formulas and macros, SQL for relational database queries, YACC grammars for creating parsers, regular expressions for specifying lexers, the Generic Eclipse Modeling System for creating diagramming languages, Csound for sound and music synthesis, and the input languages of GraphViz and GrGen, software packages used for graph layout and graph rewriting.

The opposite is:
- a general-purpose programming language, such as C, Java or Python,
- or a general-purpose modeling language such as the Unified Modeling Language (UML).

DSL (Domain specific language) with DotNet

DSL (Domain specific language) with Irony

Irony See Also:
- http://irony.codeplex.com/

Irony is a development kit for implementing languages on .NET platform. Unlike most existing yacc/lex-style solutions Irony does not employ any scanner or parser code generation from grammar specifications written in a specialized meta-language. In Irony the target language grammar is coded directly in C# using operator overloading to express grammar constructs.

Irony’s scanner and parser modules use the grammar encoded as C# class to control the parsing process. See the expression grammar sample for an example of grammar definition in C# class, and using it in a working parser.

Download See Also:
- http://irony.codeplex.com/SourceControl/list/changesets#

Introduction The goal of Irony, a new open source project, is to build a complete set of libraries and tools for language implementations in the .NET platform. It is currently in its first phase, which includes building the two compiler front-end modules - scanner and parser. This article provides an overview of the technology, and focuses on parser implementation with Irony. The project is hosted at CodePlex.

Irony brings several principal innovations into the field of compiler construction. Like many parser-building tools in use today, Irony produces a working parser from grammar specifications. However, unlike the existing parser builders, Irony does not use a separate meta-language for encoding the target grammar. In Irony, the grammar is encoded directly in C# using BNF-like expressions over grammatical elements represented by .NET objects.
Additionally, no code generation is employed - Irony’s universal LALR parser uses the information derived from C#-encoded grammar to control the parsing process.

**Irony samples**

**Mini-python**  See Also:  
http://irony.codeplex.com/SourceControl/changeset/view/6829bcd964f2#Irony.Samples%2fMiniPython%2fMiniPython.cs

```csharp
namespace Irony.Samples.MiniPython {
    // The grammar for a very small subset of Python. This is work in progress,
    // I will be adding more features as we go along, bringing it closer to real python.
    // Current version: expressions, assignments, indented code blocks, function def's, function calls
    // Full support for Python line joining rules: line continuation symbol "\", automatic line joining when
    // line ends in the middle of expression, with unbalanced parenthesis
    // Python is important test case for Irony as an indentation-sensitive language.
    [Language("MiniPython", "0.2", "Micro-subset of Python, work in progress")]
}
```

**Examples**  See Also:  
http://www.codeproject.com/KB/recipes/YourFirstDSL.aspx

**flash language**

See Also:  

Même si l’éditeur assure qu’il va continuer à supporter le framework, ainsi que la technologie sous-jacente Flash, il reconnaît aussi qu’à l’avenir les développeurs d’applications web vont plutôt utiliser HTML5 que Flash.

**A long terme, nous croyons que HTML5 sera la meilleure technologie pour concevoir des applications d’entreprise**, a indiqué la société dans un récent billet de blog.

Nous savons aussi qu’actuellement, Flex apporte des bénéfices sur les projets à grande échelle, typiquement ceux qui comportent une déclinaison sur les postes de travail.

**swiffy**

See Also:  

Swiffy converts Flash SWF files to HTML5, allowing you to reuse Flash content on devices without a Flash player (such as iPhones and iPads). Swiffy can’t yet convert DoubleClick Studio creatives.

Swiffy currently supports a subset of SWF 8 and ActionScript 2.0, and the output works in all Webkit browsers such as Chrome and Mobile Safari.
**go language**

See Also:

- [http://golang.org/](http://golang.org/)

The Go programming language is an open source project to make programmers more productive. Go is expressive, concise, clean, and efficient.

Its concurrency mechanisms make it easy to write programs that get the most out of multicore and networked machines, while its novel type system enables flexible and modular program construction.

Go compiles quickly to machine code yet has the convenience of garbage collection and the power of run-time reflection.

It’s a fast, statically typed, compiled language that feels like a dynamically typed, interpreted language.

**No exceptions**

See Also:

[http://golang.org/doc/go_faq.html#exceptions](http://golang.org/doc/go_faq.html#exceptions)

We believe that coupling exceptions to a control structure, as in the try-catch-finally idiom, results in convoluted code. It also tends to encourage programmers to label too many ordinary errors, such as failing to open a file, as exceptional.

Go takes a different approach.

For plain error handling, Go’s multi-value returns make it easy to report an error without overloading the return value. A canonical error type, coupled with Go’s other features, makes error handling pleasant but quite different from that in other languages.

Go also has a couple of built-in functions to signal and recover from truly exceptional conditions.

The recovery mechanism is executed only as part of a function’s state being torn down after an error, which is sufficient to handle catastrophe but requires no extra control structures and, when used well, can result in clean error-handling code.

See the [Defer, Panic, and Recover article](http://golang.org/doc/go_faq.html#exceptions) for details.

**Source**

From: Thiago Cangussu <cangussu@gmail.com>
Date: 2012/2/13
Subject: Re: [zeromq-dev] What do I need to do to create a subproject of the libzmq?
To: ZeroMQ development list <zeromq-dev@lists.zeromq.org>

> Show me at least one modern programming language that has no exceptions.

The Go language has no exceptions:

[http://golang.org/doc/go_faq.html#exceptions](http://golang.org/doc/go_faq.html#exceptions)
Groovy language

Groovy is an object-oriented programming language for the Java platform. It is a dynamic language with features similar to those of Python, Ruby, Perl, and Smalltalk. It can be used as a scripting language for the Java Platform.

Groovy uses a Java-like bracket syntax. It is dynamically compiled to Java Virtual Machine (JVM) bytecode and interoperates with other Java code and libraries. Most Java code is also syntactically valid Groovy.

Experience Groovy 1.8

Groovy 1.8 is the latest major and stable version of the popular dynamic language for the JVM. To learn more about the novelties, make sure to read the release notes.

In a nutshell, Groovy 1.8 provides new Domain-Specific Language authoring capabilities for more readability and expressivity of your business rules, runtime performance improvements, the bundling of the GPars parallel and concurrency library, built-in JSON support, new compile-time meta-programming features (several new useful AST transformations), new functional programming aspects for closures, and much more.

HTML5 language

HTML5 (HyperText Markup Language 5) est la prochaine révision majeure d’HTML (format de données conçu pour représenter les pages web). Cette version HTML5 est actuellement en développement.

HTML5 spécifie deux syntaxes d’un modèle abstrait défini en termes de DOM : HTML5 et XHTML5. Le langage comprend également une couche application avec de nombreuses API, ainsi qu’un algorithme afin de pouvoir traiter les documents à la syntaxe non conforme. Le travail a été repris par le W3C en mars 2007 après avoir été initié par le WHATWG. Les deux organisations travaillent en parallèle sur le même document afin de maintenir une version unique de la technologie. Le W3C vise la clôture des ajouts de fonctionnalités le 22 mai 2011 et une finalisation de la spécification en 2014, et encourage les développeurs Web à utiliser HTML 5 dès maintenant.
HTML5 and firefox

See Also:

Technologies often called part of HTML5 that aren’t

See Also:
- WebGL
- FileReader
- XMLHttpRequest
- querySelector(All)
- Geolocation
- ECMAScript5
- CSS3
- XBL2
- Web Workers
- Web Sockets
- Faster JavaScript

HTML5 tutorials

See Also:
- Web tutorials

Java language

See Also:

Java libraries

Make your target library available to your Java program. There are two ways to do this:
- The preferred method is to set the jna.library.path system property to the path to your target library. This property is similar to java.library.path, but only applies to libraries loaded by JNA.
- Change the appropriate library access environment variable before launching the VM. This is PATH on Windows, LD_LIBRARY_PATH on Linux, and DYLD_LIBRARY_PATH on OSX.
Java bindings

Java bindings  See Also:


Java Bridj  See Also:

- http://ochafik.com/blog/?p=674
- http://twitter.com/#!/ochafik
- http://stackoverflow.com/users/316785/zolive
- http://code.google.com/p/bridj/
- https://groups.google.com/forum/#!forum/nativelibs4java

It was inspired by JNA but it has:

- uncompromised speed (thanks to dyncall and assembler optimizations)
- more features : support for C++, COM, Objective-C...
- better usability : Java 1.5+ generics, annotations...
- a liberal BSD license (GPL-compatible)

Key features

- Dynamic C / C++ / COM interop : call C++ methods, create C++ objects (and subclass C++ classes from Java !)
- You never need to compile any native code : we deal with the cross-compilation hassle for you once and for all in BridJ ! (works on Windows, Linux, MacOS X, Solaris, Android...)
- Full JNAerator support : stay away from C / C++ headers !
- Small library size (~ 600 kB all included)
- Straightforward type mappings with good use of generics

C++  See Also:
http://code.google.com/p/bridj/wiki/CPlusPlus

Download  See Also:

- http://code.google.com/p/bridj/downloads/list

Build  See Also:
http://code.google.com/p/bridj/wiki/Build

2.17. Languages
The dyncall library encapsulates architecture-, OS- and compiler-specific function call semantics in a virtual “bind argument parameters from left to right and then call” interface allowing programmers to call C functions in a completely dynamic manner. In other words, instead of calling a function directly, the dyncall library provides a mechanism to push the function parameters manually and to issue the call afterwards. This means, that a program can determine at runtime what function to call, and what parameters to pass to it. The library is written in C and assembly and provides a very simple C interface to program against.

The library comes in very handy to power flexible message systems, dynamic function call dispatch mechanisms, closure implementations, to bridge different programming languages, or to simply wrap a “vararg” function.

When it comes to language bindings, the dyncall library provides a clean and portable C interface to dynamically issue calls to foreign code using small call kernels written in assembly. Instead of providing code for every bridged function call, which unnecessarily results in code bloat, only a couple of instructions are used to invoke every possible call.

Rough overview of platforms and features  The dyncall library runs on many different platforms and operating systems (including Windows, Linux, OpenBSD, FreeBSD, Darwin, DragonFlyBSD, NetBSD, Plan9, iOS, Haiku, Nintendo DS, Playstation Portable, Solaris, etc.) and processors (x86, x64, arm, mips, ppc32, etc.).

Most of the platforms’ C-calling conventions are supported - including “vararg” functions, as well as C++-“thiscalls” on some platforms, and the multitude of calling conventions on Windows (“fastcall”, “stdcall”, etc.). Most of C99’s types are supported for setting up a call, however, structure and union support is still missing (we are working on it, though).

Additionally, dyncall comes with dyncallback, a library for callback support (missing on some platforms), and dynload, to facilitate portable dynamic library symbol loading and access (only for platforms with dynamic library support).

Java CPP  See Also:

- http://code.google.com/p/javacpp/
- https://groups.google.com/forum/#!forum/javacpp-project
- http://www.cs.brown.edu/courses/csci1290/results/final/sbnguyen/
- http://www.ok.ctrl.titech.ac.jp/~saudet/
- http://stackoverflow.com/users/523744/samuel-audet
- http://code.google.com/p/javacv/
JavaCPP provides efficient access to native C++ inside Java, not unlike the way some C/C++ compilers interact with assembly language. No need to invent a whole new language, whatever Microsoft may opine about it. Under the hood, it uses JNI, so it works with all Java implementations, including Android. In contrast to other approaches (SWIG, JNIWrapper, Platform Invoke, JNI Direct, JNA, JniMarshall, J/Invoke, HawtJNI, BridJ, etc.), it supports naturally many features of the C++ language often considered problematic, including overloaded operators, template classes and functions, member function pointers, callback functions, nested struct definitions, variable length arguments, nested namespaces, large data structures containing arbitrary cycles, multiple inheritance, passing/returning by value/reference/vector, anonymous unions, bit fields, exceptions, destructors and garbage collection.

Copyright (C) 2011 Samuel Audet <samuel.audet at gmail.com> Personal home page: http://www.ok.ctrl.titech.ac.jp/~saudet/ Licensed under the GNU General Public License version 2 (GPLv2) with Classpath exception.

Why choosing javacpp instead swig ?  See Also:
http://www.cs.brown.edu/courses/csci1290/results/final/sbnguyen/

- JNI is a pain to work with
- Using SWIG to generate config file. Overhead of learning new things and headach when it breaks.
- Only 1 or 2 actual samples program and no instruction to create your own new projects.

After spending a few hours trying to get my SWIG config file to work, I scrapped the whole things. If it’s that hard just to get a sample program working then it’s not going to be much better than the N900.

You have to define your method 3 times: in Java, in C++ and in SWIG. That’s not fun.

See Also:
http://stackoverflow.com/questions/5201292/a-good-java-wrapper-for-tapi-2

0 down vote favorite share [fb] share [tw]

Does anyone know of a good JNI/Java wrapper for TAPI 2? ...

You could try to use one of the following tools, among others, to make the task more trivial.

- SWIG: http://www.swig.org/
- JNA: http://jna.java.net/
- JavaCPP: http://code.google.com/p/javacpp/

Being the author of the third there, I recommend that one :)

javacpp download  See Also:
http://code.google.com/p/javacpp/downloads/list

Put javacpp.jar into C:Program FilesJavajdk1.7.0jrelibext

Répertoire de C:\Program Files\Java\jdk1.7.0\jre\lib\ext
04/10/2011 13:15 <REP> ...
04/10/2011 13:15 <REP> ..
03/10/2011 15:52 8 934 dnsns.jar
04/10/2011 09:56 76 906 javacpp.jar
04/10/2011 12:45 253 160 junit-4.10.jar
03/10/2011 15:52 1 019 934 localedata.jar
03/10/2011 15:52 198 063 sunjce_provider.jar

2.17. Languages 557
JCC is a C++ code generator for producing the code necessary to call into Java classes from CPython via Java’s Native Invocation Interface (JNI).

JCC generates C++ wrapper classes that hide all the gory details of JNI access as well Java memory and object reference management.

JCC generates CPython types that make these C++ classes accessible from a Python interpreter.

JCC attempts to make these Python types pythonic by detecting iterators and property accessors.

Iterators and mappings may also be declared to JCC.

JCC has been built on Python 2.3, 2.4, 2.5, 2.6 and 2.7 and has been used with various Java Runtime Environments such as Sun Java 1.4, 1.5 and 1.6, Apple’s Java 1.4 and 1.5 on Mac OS X and open source Java OpenJDK 1.7 builds.

An experimental port to Python 3 is available from a branch: http://svn.apache.org/repos/asf/lucene/pylucene/branches/python_3/jcc/

JCC is supported on Mac OS X, Linux, Solaris and Windows.

JCC is written in C++ and Python. It uses Java’s reflection API to do its job and needs a Java Runtime Environment to be present to operate.

JCC is built with distutils or setuptools:

```
python setup.py build
sudo python setup.py install
```

JNA See Also:

http://code.google.com/p/bridj/wiki/FAQ#JNA
Java Native Access

See Also:

- http://code.google.com/p/bridj/wiki/FAQ#JNA
- http://code.google.com/p/jnaerator/
- Python ctype modules

Java Native Access provides Java programs easy access to native shared libraries without using the Java Native Interface. JNA’s design aims to provide native access in a natural way with a minimum of effort. No boilerplate or generated glue code is required.

History


JNA is a project with a long history (stretching back to 1999) but it was first released in November 2006. Since then it has been slowly gaining the attention of developers who need to integrate native C code into Java-based projects.

JNA has also made some waves among JRuby programmers because it can be used to work around one of JRuby’s pervasive problems, lack of support for POSIX calls. JNA has also been proposed as a solution for extending Ruby with low-level C code.

I enjoyed working with JNA and believe you will find it easier and safer to use than JNI for accessing native code. Needless to say, there is more to JNA than I could cover in a single article. See the Resources section to learn more about this open source Java project. Experiment with it and share your experiences in the discussion forum for this series. I’ll be back next month with another lesser known open source project that could benefit your everyday Java development.

Readme

https://github.com/twall/jna#readme

JNA provides Java programs easy access to native shared libraries (DLLs on Windows) without writing anything but Java code—no JNI or native code is required. This functionality is comparable to Windows’ Platform/Invoke and Python’s ctypes. Access is dynamic at runtime without code generation.

JNA allows you to call directly into native functions using natural Java method invocation. The Java call looks just like it does in native code. Most calls require no special handling or configuration; no boilerplate or generated code is required.
The **JNA** library uses a small native library stub to dynamically invoke native code. The developer uses a Java interface to describe functions and structures in the target native library. This makes it quite easy to take advantage of native platform features without incurring the high overhead of configuring and building JNI code for multiple platforms.

While some attention is paid to performance, correctness and ease of use take priority.

JNA includes a platform library with many native functions already mapped as well as a set of utility interfaces that simplify native access.

**Getting started**  

See Also:

https://github.com/twall/jna/blob/master/www/GettingStarted.md

Java Native Access (JNA) has a single component, jna.jar; the supporting native library (jnidispatch) is included in the jar file. JNA is capable of extracting and loading the native library on its own, so you don’t need additional configuration. JNA falls back to extraction if the native library is not already installed on the local system somewhere accessible to System.loadLibrary.

The native library is also available in platform-specific jar files for use with Java Web Start.

Begin by downloading the latest release of JNA and referencing jna.jar in your project’s CLASSPATH.

---

**Note**:  

with the Netbeans IDE simply add the jar files.

If the C header files for your library are available, you can auto-generate a library mapping by using Olivier Chafik’s excellent **JNAerator** utility. This is especially useful if your library uses long or complicated structures where translating by hand can be error-prone.

**Architecture**  

The JNA library uses a small native library called foreign function interface library (libffi) to dynamically invoke native code. The JNA library uses native functions allowing code to load a library by name and retrieve a pointer to a function within that library, and uses libffi library to invoke it, all without static bindings, header files, or any compile phase. The developer uses a Java interface to describe functions and structures in the target native library.

This makes it quite easy to take advantage of native platform features without incurring the high development overhead of configuring and building JNI code.

JNA is built and tested on Mac OS X, Microsoft Windows, FreeBSD / OpenBSD, Solaris, and Linux. It is also possible to tweak and recompile the native build configurations to make it work on other platforms. For example, it is known to work on Windows Mobile, even if it is not tested for this platform by the development team.

**Why to use JNA**  

See Also:


It’s difficult to answer such a generic question. I suppose the most obvious difference is that with JNI, the type conversion is implemented on the native side of the Java/native border, while with JNA, the type conversion is implemented in Java. If you already feel quite comfortable with programming in C and have to implement some native code yourself, I would assume that JNI won’t seem too complex. If you are a Java programmer and only need to invoke a third party native library, using JNA is probably the easiest path to avoid the perhaps not so obvious problems with JNI.

See Also:

http://stackoverflow.com/questions/3720563/access-c-shared-library-from-java-jni-jna-cni-or-swig

Which of the following (or other) method would you recommend for accessing a C++ shared library from Java and why?

- JNI: I hear this has a number of pitfalls and is quite the undertaking?
SWIG: Apparently this makes using JNI easier, but I’ve heard it has some problems too?

JNA: I could write a C interface and then use JNA with it instead, which is apparently significantly easier than JNI?

CNI: Apparently this is easier than JNI?

Another library: it can’t be commercial and I’d prefer something that is still well maintained (this would throw out JNEasy, JNative, and JACE - I think).

I’m trying to go for minimal development time and problems down the road. Any info on pros/cons of each of these would be appreciated.

For Java->C++, I’ve used JNI, JNA and played with SWIG.

**JNA is the easiest to use**, but as you note, requires hand-writing a C interface to the C++ API. **It can also be slower than JNI by an order of magnitude.** However, I measured individual calls at a few hundred nanoseconds on one machine, so that’s unlikely to matter except in a bottleneck.

JNA redundantly specifies C function signatures, in Java. JNI can redundantly specify Java function signatures, in C strings. Discrepancies in either can result in unexpected runtime behavior.

I personally would use JNA unless the interface is complex enough to make hand-writing the C interface cumbersome for you, or the individual calls are more than a few hundred nano seconds.

This week I’ve been faced with such an exception – a rich C++ interface with many classes and methods. I’ve started playing with SWIG, and it’s looking promising. It’s been fairly easy to use, and automatically generates the Java bindings and C implementation. Smart pointers did take a little extra work – you have to instruct SWIG to instantiate the templates.

EDIT (a year later):

SWIG is amazingly powerful. It can also be more complex to set up. For simple, thin interfaces, I’d probably consider JNA or JNI first. But SWIG is handy for thick interfaces.

I’m a little surprised that SWIG works, given the complexity of some C++ header files. But SWIG appears to have little difficulty.

I did have to write some SWIG typemaps and macros containing C++/JNI code. For example, passing std::strings by reference required a custom typemap. Transforming thrown C++ exceptions to thrown Java exceptions required a typemap and a macro.

No changes were needed to our header files except that SWIG fully instantiated a smart-pointer template, which had been parameterized with some classes that did not satisfy its expectation of a default constructor.

Solution: add a few default constructors.

**How does JNA performance compare to custom JNI?**

See Also:


JNA direct mapping can provide performance near that of custom JNI. Nearly all the type mapping features of interface mapping are available, although automatic type conversion will likely incur some overhead.

The calling overhead for a single native call using JNA interface mapping can be an order of magnitude (∼10X) greater time than equivalent custom JNI (whether it actually does in the context of your application is a different question).

In raw terms, the calling overhead is on the order of hundreds of microseconds instead of tens of microseconds. Note that that’s the call overhead, not the total call time. This magnitude is typical of the difference between systems using dynamically-maintained type information and systems where type information is statically compiled. JNI hard-codes type information in the method invocation, where JNA interface mapping dynamically determines type information at runtime.
You might expect a speedup of about an order of magnitude moving to JNA direct mapping, and a factor of two or three moving from there to custom JNI.

The actual difference will vary depending on usage and function signatures. As with any optimization process, you should determine first where you need a speed increase, and then see how much difference there is by performing targeted optimizations. The ease of programming everything in Java usually outweighs small performance gains when using custom JNI.

### JNA versus JNI

#### See Also:

http://mbaron.developpez.com/javase/jnijna/ (JNA versus JNI)

---

**jnaerator**

#### See Also:

- http://code.google.com/p/jnaerator/
- http://ochafik.com/blog
- *Java Native Access*

---

**Download**

#### See Also:

http://code.google.com/p/jnaerator/downloads/list

---

**Launch**

#### GUI

#### See Also:

http://jnaerator.sourceforge.net/webstart/JNAerator/JNAeratorStudio.jnlp

---

**Command line**

```
java -jar jnaerator.jar options headerFiles
```

See CommandLineOptionsAndEnvironmentVariables for more details.

---

**C to java transformation**

#### See Also:

http://code.google.com/p/jnaerator/wiki/CToJavaTransformation

---

**Credits and License**

JNAerator is written and maintained by Olivier Chafik and is distributed under the terms of the LGPL license, but it relies on many great thirdparty libraries.

---

**Java Native Access examples**
Java Native Access example 1  See Also:
http://www.scriptol.fr/programmation/jna.php

JNI (Java Native Interface)  See Also:
http://en.wikipedia.org/wiki/Java_Native_Interface
The Java Native Interface (JNI) is a programming framework that enables Java code running in a Java Virtual Machine (JVM) to call and to be called[1] by native applications (programs specific to a hardware and operating system platform) and libraries written in other languages such as C, C++ and assembly.

JNI examples

jzmq java binding  See Also:
• http://www.zeromq.org/bindings:java
• http://github.com/zeromq/jzmq

Scala  See Also:
http://www.theregister.co.uk/2009/04/01/twitter_on_scala/
“The biggest selling point for certain organizations is that Scala is 100 per cent compatible with your existing JVM code. There’s no wrappers. There’s no bullshit. You just load up your JVM code and call in to it, and there’s no performance penalty. If you have a big investment in Java infrastructure, Scala is the ideal language.”

Java swig  See Also:
• http://www.swig.org/tutorial.html
• http://en.wikipedia.org/wiki/SWIG
• http://www.dabeaz.com/cgi-bin/wiki.pl?SwigFaq/Java
• http://www.swig.org/projects.html
SWIG (Simplified Wrapper and Interface Generator) is an open source software tool used to connect computer programs or libraries written in C or C++ with scripting languages such as Lua, Perl, PHP, Python, R, Ruby, Tcl, and other languages like C#, Java, Modula-3, Objective Caml, Octave, and Scheme. Output can also be in the form of XML or Lisp S-expressions

Java swig projects

Quantlib  See Also:
• http://quantlib.org/extensions.shtml
QuantLib is available as a C#, Guile, Java, MzScheme, Perl, Python, and Ruby module by means of SWIG. Experimental bindings to GNU R and Objective Caml are also available; feedback is welcome.
Compilation of the Java module on Windows is performed by means of sKWash. A guide to the process can be found here. A guide to the compilation of the Java module on Linux is available here.

Java swig examples
Java swig example

See Also:

- http://www.swig.org/tutorial.html

Building a Java module  So you want to get going in a hurry? To illustrate the use of SWIG, suppose you have some C functions you want added to Tcl, Perl, Python, Java and C#.

Specifically, let’s say you have them in a file `example.c`:

```c
/* File : example.c */
#include <time.h>
double My_variable = 3.0;

int fact(int n) {
    if (n <= 1) return 1;
    else return n*fact(n-1);
}

int my_mod(int x, int y) {
    return (x%y);
}

char *get_time()
{
    time_t ltime;
    time(&ltime);
    return ctime(&ltime);
}
```

(swig) Interface file  Now, in order to add these files to your favorite language, you need to write an “interface file” which is the input to SWIG. An interface file for these C functions might look like this:

```c
/* example.i */
%module example
%
/* Put header files here or function declarations like below */
extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
%

extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
```

SWIG will also generate JNI code for accessing C/C++ code from Java. Here is an example building a Java module (shown for Cygwin, see the SWIG Wiki Shared Libraries page for help with other operating systems):

```
$ swig -java example.i
$ gcc -c example.c example_wrap.c -I/c/jdk1.3.1/include -I/c/jdk1.3.1/include/win32
$ gcc -shared example.o example_wrap.o -mno-cygwin -Wl,--add-stdcall-alias -o example.dll
$ cat main.java
```
public class main {
    public static void main(String argv[]) {
        System.loadLibrary("example");
        System.out.println(example.getMy_variable());
        System.out.println(example.fact(5));
        System.out.println(example.get_time());
    }
}

$ javac main.java
$ java main
3.0
120
Mon Mar 4 18:20:31 2002
$

Java implementations

Java 2 Standard Edition  See Also:


Contents

- Java 2 Standard Edition
  - Sommaire
  - Versions

**Sommaire**  J2SE (Java 2 Standard Edition) est l’une des éditions (ou distribution) de framework Java, destinée plus particulièrement aux applications pour poste de travail.

Cette plateforme contient outre toutes les API de base:

- toutes les API spécialisées dans le poste client (JFC et donc Swing, AWT et Java2D),
- ainsi que des API d’usage général comme JAXP (pour le parsing XML)
- et JDBC (pour la gestion des bases de données).

Il a été renommé Java SE par Sun Microsystems juste après son rachat par Oracle Corporation en 2009 pour 7 milliards de dollars.

À chaque version du JSE correspond notamment, comme toutes les éditions Java :

- les Java Specification Requests (JSR), constituant les spécifications de la version considérée
- un Java software development kits (JDK) contenant les librairies logicielles;
- un Java Runtime Environment (JRE) contenant le seul environnement d’exécution (compris de base dans le JDK)

**Versions**  

2.17. Languages
Java SE versions  See Also:


J2SE 1.7  See Also:

- http://www.oracle.com/technetwork/java/javase/7u-relnotes-515228.html

Contents

- J2SE 1.7
  - Java™ SE 7 Update 4
  - Highlights

Java™ SE 7 Update 4  The full internal version number for this update release is 1.7.0_04-b20 (where “b” means “build”) except for Windows on which it is 1.7.0_04-b22 and for Mac OS X on which it is 1.7.0_04-b21.

The external version number is 7u4.

Highlights  This update release contains the following enhancements:

- JDK Support for Mac OS X
- New JVM (Java HotSpot Virtual Machine, version 23)
- New Supported Garbage Collector: Garbage First (G1)
- JavaFX 2.1 Runtime co-installs with JRE 7 during auto-update
- JAXP upgraded to 1.4.6
- Java DB upgraded to 10.8.2.2
- SPARC T4 specific crypto optimizations in the security area
- New flag to unlock Commercial Features

Java translations

Java to python  See Also:

- https://github.com/natural/java2python/

Simple but effective tool to translate Java source code into Python.

The java2python package can translate any syntactically valid Java source code file.

The generated Python code is not guaranteed to run, nor is guaranteed to be syntactically valid Python.

However, java2python works well many cases, and in some of those, it creates perfectly usable and workable Python code.
Sharpen  See Also:

- https://github.com/xamarin/XobotOS/tree/master/sharpen
- http://blog.xamarin.com/2012/05/01/android-in-c-sharp/
- https://github.com/xamarin/XobotOS

This is a local copy of Sharpen (based on commit 2446d365 from lluis), with local additions and modifications.

We are only using sharpen.core, with a custom version of the builder and ui plugins.

**javascript language**

JavaScript est un langage de programmation de scripts principalement utilisé dans les pages web interactives mais aussi côté serveur. C’est un langage orienté objet à prototype, c’est-à-dire que les bases du langage et ses principales interfaces sont fournies par des objets qui ne sont pas des instances de classes, mais qui sont chacun équipés de constructeurs permettant de générer leurs propriétés, et notamment une propriété de prototypage qui permet d’en générer des objets héritiers personnalisés.

Le langage a été créé en 1995 par Brendan Eich pour le compte de Netscape Communications Corporation. Le langage, actuellement à la version 1.8.2 est une implémentation de la 3e version de la norme ECMA-262 qui intègre également des éléments inspirés du langage Python.

La version 1.8.5 du langage est prévue pour intégrer la 5e version du standard ECMA-2

**Javascript books**

**Eloquent javascript language**  See Also:


**Smooth coffescript**  See Also:


Smooth CoffeeScript is a reimagination of the excellent book Eloquent JavaScript, as if it had been written in CoffeeScript instead.

Covers language features as well a the functional and object oriented programming styles. By E. Hoigaard.

**Javascript patterns**  See Also:

- http://addyosmani.com/largescalejavascript/
javascript history

See Also:


As you may know, I wrote JavaScript in ten days. JS was born under the shadow of Java, and in spite of support by marca and Bill Joy, JS in 1995 was essentially a one-man show.

I had a bit of help, even at the start, that I’d like to acknowledge again. Ken Smith, a Netscape acquiree from Borland, ported JDK 1.0-era java.util.Date (we both just drafted off of the Java truck, per management orders; we did not demur from the Y2K bugs in that Java class). My thanks also to Netscape 2’s front-end hackers, chouck, atotic, and garrett for their support.

EDIT: can’t forget spence on the X front end!

That was 1995. Engine prototype took ten days in May. Bytecode compiler and interpreter from the start, because Netscape had a server-side JS product in the works. The rest of the year was browser integration, mainly what became known as “DOM level 0?”. Only now standardized in HTML 5 and Anne’s wg. Sentence fragments here show my PTSD from that sprint :-/.

In 1996 I finally received some needed assistance from RRJ, who helped port David M. Gay and Guy Steele’s dtoa.c and fix date/time bugs.

Also in summer 1996, nix interned at Netscape while a grad student at CMU, and wrote the first LiveConnect. I am still grateful for his generous contributions in wide-ranging design discussions and code-level interactions.

At some point in late summer or early fall 1996, it became clear to me that JS was going to be standardized. Bill Gates was bitching about us changing JS all the time (some truth to it; but hello! Pot meet Kettle…). We had a standards guru, Carl Cargill, who knew Jan van den Beld, then the Secretary-General of ECMA (now Ecma). Carl steered our standardization of JS to ECMA.

Joining ECMA and participating in the first JS standards meeting was an eye-opener. Microsoft sent a B-team, and Borland and a company called NOMBAS also attended. “Success has many fathers” was the theme. The NOMBAS founder greeted me by saying “oh, we’ve been doing JavaScript for years”. I did not see how that could be the case, unless JS meant any scripting language with C-based syntax. I had not heard of NOMBAS before then.

At that first meeting, I think I did well enough in meta-debate against the Microsoft team that they sent their A-team to the next meeting. This was all to the good, and Microsoft in full-blooded compete mode, but also with individual initiative beyond the call of corporate duty by Shon Katzenberger, materially helped create ES1. Sun contributed Guy Steele, who is composed of pure awesome. Guy even brought RPG for fun to a few meetings (Richard contributed ES1 Clause 4).

Meanwhile, in fall 1996, I was under some pressure from Netscape management to write a proto-spec for JS, but that was not something I could do while also maintaining the “Mocha” engine all by myself in both shipping and future Netscape releases, along with all of the DOM code.

This was a ton of work, and on top of it I had to pay off substantial technical debt that I had willingly taken on in the first year. So I actually stayed home for two weeks to rewrite Mocha as the codebase that became known as SpiderMonkey, mainly to get it done (no other way), also to go on a bit of a strike against the Netscape management team that was still underinvesting in JS. This entailed garbage collection and tagged values instead of slower reference-counting and fat discriminated union values.

Also in fall 1996, chouck decided to join me as the second full-time JS team-mate. He and I did some work targeting the (ultimately ill-fated) Netscape 4 release. This work was ahead of its time. We put the JS engine in a separate thread from the “main thread” in Netscape (still in Mozilla). This allowed us to better overlap JS and HTML/CSS/image computations, years ahead of multicore. You could run an iloop in JS and the “slow script dialog” seamlessly floated above it, allowing you to stop the loop or permit it to continue.

After summer 1996 and the start of ECMA-262 standardization, Netscape finally invested more in JS. Clayton Lewis joined as manager, and hired Norris Boyd, who ended up creating Rhino from SpiderMonkey’s DNA transcoded to
Java. This was ostensibly because Netscape was investing in Java on the server, in particular in an AppServer that wanted JS scripting.

I met shaver for the first time in October 1996 at Netscape’s NY-based Developer Conference, where he nimbly nerd-blocked some Netscape plugin API fanboys and saved me from having to digress from the main thing, which was increasingly JS.

Some time in 1997, shaver contributed “LiveConnect 2?, based on more mature Java reflection APIs not available to nix in 1996. Clayton hired shaver and the JS team grew large by end of 1997, when I decided to take a break from JavaScript (having delivered ES1 and ES2) and join the nascent mozilla.org.

I handed the keys to the JS kingdom to Waldemar Horwat, now of Google, in late 1997. Waldemar did much of the work on ES3, and threw his considerable intellect into JS2/ES4 afterwards, but without overcoming the market power and stalling tactics of Microsoft.

True story: Waldemar’s Microsoft nemesis on TC39 back then, at the time a static language fan who hated JS, has come around and now endorses JS and dynamic languages.

Throughout all of this, I maintained module ownership of SpiderMonkey.

Fast-forward to 2008. After a great (at the time) Firefox 3 release where @shaver and I donned the aging super-hero suits one more time to compete successfully on interpreter performance against JavaScriptCore in WebKit, Andreas Gal joined us for the summer in which we hacked TraceMonkey, which we launched ahead of Chrome and V8.

A note on V8: I’d learned of it in 2006, when I believe it was just starting. At that point there was talk about open-sourcing it, and I welcomed the idea, encouraging any of: hosting on code.google.com, hosting without any pressure to integrate into Firefox on mozilla.org (just like Rhino), or hosting with an integration plan to replace SpiderMonkey in Firefox. I had to disclose that another company was about to release their derived-from-JS engine to Mozilla, but my words included “the more the merrier”. It was early days as far as JS JITs were concerned.

V8 never open-sourced in 2006, and stealthed its way to release in September 2008. This may have been a prudent move by Google to avoid exciting Microsoft. Clearly, in 1995, the “Netscape + Java kills Windows” talk from Netscape antagonized Microsoft. I have it on good authority that a Microsoft board member wrote marca at the end of 1995 warning “you’ve waved the cape in the bull’s face — prepare to get the horns!” One could argue that Chrome in 2008 was the new red cape in the bull’s face, which begot IE9 and Chakra.

Whatever Google’s reasoning, keeping V8 closed-source for over two years hurt JS in this sense: it meant Apple and Mozilla had to climb the JIT learning curves on their own (at first; then finally with the benefit of being able to inspect V8 sources). Sure, the Anamorphic work on Self and Smalltalk was somewhat documented, and I had learned it in the ’90s, in part with a stint on loan from Netscape to Sun when they were doing due diligence in preparation for acquiring Anamorphic. But the opportunity to build on a common engine codebase was lost to path dependence.

On the upside, different competing open source engines have demonstrably explored a larger design space than one engine codebase could under consolidated management.

In any event, the roads not taken in JS’s past still give me pause, because similar roads lie ahead. But the past is done, and once we had launched TraceMonkey, and Apple had launched SquirrelFish Extreme, the world had multiple proofs along with the V8 release that JS was no longer consigned to be “slow” or “a toy”, as one referee dismissed it in rejecting a PLDI submission from Andreas in 2006.

You know the rest: JS performance has grown an order of magnitude over the last several years. Indeed, JS still has upside undreamed of in the Java world where 1% performance win is remarkable. And, we are still at an early stage in studying web workloads, in order to synthesize credible benchmarks. On top of all this, the web is still evolving rapidly, so there are no stable workloads as far as I can tell.

Around the time TraceMonkey launched, Mozilla was lucky enough to hire Dave Mandelin, fresh from PhD work at UCB under Ras Bodik.

The distributed, open source Mozilla JS team delivered the goods in Firefox 4, and credit goes to all the contributors. I single Dave out here because of his technical and personal leadership skills. Dave is even-tempered, super-smart, and a true empirical/skeptical scientist in the spirit of my hero, Richard Feynman.
So it is with gratitude and more than a bit of relief, after a very long 16 years in full, 13 years open source, that I’m announcing the transfer of SpiderMonkey’s module ownership to @dmandelin.

Hail to the king, baby!

Javascript libraries

classy  See Also:
http://www.pocoo.org/projects/classy/#classy

Classy is a library for JavaScript that provides Python inspired class support for JavaScript on top of the existing prototype based object system. It can be used with zero-footprint which allows its use in injected JavaScript code without breaking the existing website.

It supports the built-in instanceof operator, mixin classes, class and instance attributes as well as simplified calls into super methods.

It creates one global object called Class that can be renamed and transparently moved into an anonymous scope, leaving no trace behind on the global namespace.

javascript multitreading

See Also:

- http://david.blob.core.windows.net/html5/Introduction%20to%20the%20HTML5%20Web%20Workers%20the%20JavaScript%20multithreading%20approach.htm#firstwebworker

Introduction to the HTML5 Web Workers: the JavaScript multithreading approach  An HTML5 application is obviously written using JavaScript.

But compared to other kind of development environments (like native one), JavaScript historically suffers from an important limitation: all its execution process remains inside a unique thread. This could be pretty annoying with today multi-cores processors like the i5/i7 containing up to 8 logical CPUs and even with the latest ARM mobile processors being dual or even quad-cores.

Hopefully, we’re going to see that HTML5 offers to the web a way to better handle these new marvelous processors to help you embrace a new generation of web applications.

Before the workers  See Also:

Technologies often called part of HTML5 that aren’t

This JavaScript limitation implies that a long running processing will freeze the main window. We often say in our developers’ words that we’re blocking the “UI Thread”. This is the main thread in charge of handling all the visual elements and associated tasks: drawing, refreshing, animating, user inputs events, etc. We all know the bad consequences of overloading this thread: the page freezes and the user can’t interact anymore with your application.

The user experience is then of course very bad and the user will probably decide to kill the tab or the browser instance. This is probably not something you’d like to see while using your application!

To avoid that, the browsers have implemented a protection mechanism which alerts the user when a long-running suspect script occurs. Unfortunately, this protection mechanism can’t make the difference between a script not written correctly and a script which really needs more time to accomplish its work. Still, as it blocks the UI thread, it’s better
to tell you that something wrong is maybe currently occurring. Here are some messages examples (from Firefox 5 & IE9).

Javascript news

See Also:

Les applications natives seront-elles remplacées par les applications HTML 5 ?

Le HTML5, la prochaine révision majeure du HMTL dont les travaux de spécification sont encore en cours, fait couler beaucoup d’encre.

Le soutien et l’adoption du futur standard du Web par les géants comme Microsoft, Google ou encore Mozilla ont donné naissance à d’innombrables débats sur la pertinence des plateformes existantes comme Flash et Silverlight.

Certains estiment même que le HTML5 pourrait apporter une véritable mutation dans le monde des applications, avec une orientation des entreprises et développeurs vers les solutions HTML 5 au détriment des solutions natives.

C’est en tout cas le point de vue de Jon Lech Johansen, développeur Novergien, reconnu pour ses travaux de rétro-ingénierie sur la gestion des droits numériques, qui estime dans un tweet que les applications natives seront remplacées par les applications HTML5 l’an prochain.

La position actuelle des éditeurs donne du crédit à cette affirmation.

Microsoft a mis en avant dans son futur OS Windows 8, qui sera disponible l’année prochaine, la possibilité offerte aux développeurs de créer des applications Metro en utilisant uniquement le couple HTML5 + JavaScript.

Mozilla, de son côté, espère également publier avant la fin de 2012, son nouvel OS mobile Boot to Gecko, qui permettra de créer des applications HTML 5 pouvant supplanter des solutions natives.

À ces exemples, viennent s’ajouter les avis des éditeurs des magazines numériques. Rob Grimshaw, directeur général de la division numérique du Financial Times, lors du sommet Open Mobile a déclaré : Nous avons utilisé HTML5 pour sortir de iTunes et entrer à nouveau dans l’environnement du navigateur.

L’année prochaine pourrait être l’année du HTML 5 au détriment des applications natives ? Ou celles-ci ont encore des beaux jours devant elles ?

javascript tools

phantom js javascript language

See Also:
• http://code.google.com/p/phantomjs/
• http://phantomjs.org

PhantomJS is a headless WebKit with JavaScript API. It has fast and native support for various web standards: DOM handling, CSS selector, JSON, Canvas, SVG.

The official web site is phantomjs.org.

PhantomJS is based on Qt. There are two implementations, using C++ and Python.

The logo is from http://www.openclipart.org/detail/17847.
javascript tutorials

See Also:

perl language

See Also:
• http://fr.wikipedia.org/wiki/Perl_%28langage%29
• http://lhullier.developpez.com/tutoriels/perl/intro/

Le nom

Initialement, le concepteur de Perl, Larry Wall, avait prévu de nommer son langage pearl, d’après une parabole biblique se trouvant relatée dans l’évangile selon Matthieu (chapitre 13, versets 45 et 462).

Malheureusement pour Wall, il existait déjà un langage de programmation dénommé PEARL pour la programmation multitâche et temps réel. Finalement, Wall changea simplement l’orthographe pour arriver à « Perl ». Les significations diverses que l’on trouve de nos jours comme par exemple le Practical Extraction and Report Language sont simplement des rétro-acronymes (des acronymes forgés a posteriori).

Python

See Also:
• https://fr.wikipedia.org/wiki/Python_%28langage%29
• https://pythonconquerstheuniverse.wordpress.com/2011/11/06/posting-sourcecode-on-wordpress/
• http://www.haypocalc.com/wiki/Python#Apprendre_Python

Python conferences

Python us See Also:

PyCon is an activity of the Python Software Foundation, a 501c3 non-profit organization. To support future conferences, please donate to the Foundation at www.python.org/psf/donations. Video and audio material from PyCon are licensed under the Creative Commons CC-BY-NC-SA <creative_commons> license. This means you can incorporate excerpts or entire recordings in your own non-commercial projects, as long as you credit the speaker and you CC-license the finished project.

PyCon is the largest annual gathering for the community using and developing the open-source Python programming language.

PyCon is organized by the Python community for the community. We try to keep registration far cheaper than most comparable technology conferences, to keep PyCon accessible to the widest group possible.

PyCon is a diverse conference dedicated to providing an enjoyable experience to everyone. Our code of conduct is intended to help everyone maintain the PyCon spirit. We thank all attendees and staff for observing it.
Pycon US 2012  See Also:

- http://us.pycon.org/2012/
- http://us.pycon.org/2012/sponsors/

Mozilla sponsor  See Also:


Mozilla is now officially listed as a Pycon US sponsor.

This is great news for Python and for the Mozilla Community: more and more projects in Mozilla are using Python. It’s the mainstream language for all the work we’re doing in the Services team on server-side, and numerous projects like SUMO or AMO are powered by Python.

So, if you are involved in the Mozilla community and in Python, you should consider submitting a talk or a tutorial. Or just attend to the conference in Santa Clara next March.

Pycon US 2011  See Also:

- http://us.pycon.org/2011/home/

Pycon US 2010  See Also:

- http://linuxfr.org/news/actualit%C3%A9s-python-pypy-12-distutils2-vid%C3%A9os-de-pycon-2010

Teaching compilers with python  See Also:

http://www.matthieuamiguet.ch/media/documents/TeachingCompilersWithPython_Paper.pdf


Searching python code

http://nullege.com/codes  See Also:

- http://nullege.com/codes/

python doc

See Also:

- http://docs.python.org/index.html
- http://docs.python.org/glossary.html
- http://docs.python.org/py3k/glossary.html

python doc library  See Also:

http://docs.python.org/py3k/library/index.html

2.17. Languages
The argparse module makes it easy to write user-friendly command-line interfaces. The program defines what arguments it requires, and argparse will figure out how to parse those out of sys.argv. The argparse module also automatically generates help and usage messages and issues errors when users give the program invalid arguments.

Example The following code is a Python program that takes a list of integers and produces either the sum or the max:

```python
import argparse

parser = argparse.ArgumentParser(description='Process some integers.')
parser.add_argument('integers', metavar='N', type=int, nargs='+',
                    help='an integer for the accumulator')
parser.add_argument('--sum', dest='accumulate', action='store_const',
                    const=sum, default=max,
                    help='sum the integers (default: find the max)')

args = parser.parse_args()
print(args.accumulate(args.integers))
```

Assuming the Python code above is saved into a file called `prog.py`, it can be run at the command line and provides useful help messages:

```
$ prog.py -h
usage: prog.py [-h] [--sum] N [N ...]
Process some integers.

positional arguments:
 N  an integer for the accumulator

optional arguments:
 -h, --help  show this help message and exit
 --sum  sum the integers (default: find the max)
```

When run with the appropriate arguments, it prints either the sum or the max of the command-line integers:

```
$ prog.py 1 2 3 4
4

$ prog.py 1 2 3 4 --sum
10
```

If invalid arguments are passed in, it will issue an error:

```
$ prog.py a b c
usage: prog.py [-h] [--sum] N [N ...]
prog.py: error: argument N: invalid int value: ‘a’
```
The following sections walk you through this example.

**python bytes**  
See Also:
- [http://docs.python.org/py3k/library/stdtypes.html#sequence-types-str-bytes-bytearray-list-tuple-range](http://docs.python.org/py3k/library/stdtypes.html#sequence-types-str-bytes-bytearray-list-tuple-range)
- [http://docs.python.org/py3k/library/functions.html#bytes](http://docs.python.org/py3k/library/functions.html#bytes)

**Bytes literals**  
See Also:
- [http://docs.python.org/py3k/reference/lexical_analysis.html#strings](http://docs.python.org/py3k/reference/lexical_analysis.html#strings)

Bytes and bytearray objects contain single bytes – the former is immutable while the latter is a mutable sequence. Bytes objects can be constructed the constructor, `bytes()`, and from literals; use a b prefix with normal string syntax: `b'xyzzy'`. To construct byte arrays, use the `bytearray()` function.

**python contextmanager**  
See Also:
- [http://docs.python.org/py3k/library/stdtypes.html#context-manager-types](http://docs.python.org/py3k/library/stdtypes.html#context-manager-types)

Python’s with statement supports the concept of a runtime context defined by a context manager. This is implemented using a pair of methods that allow user-defined classes to define a runtime context that is entered before the statement body is executed and exited when the statement ends.

Python defines several context managers to support easy thread synchronisation, prompt closure of files or other objects, and simpler manipulation of the active decimal arithmetic context. The specific types are not treated specially beyond their implementation of the context management protocol.

See the `contextlib` module for some examples.

Python’s generators and the `contextlib.contextmanager` decorator provide a convenient way to implement these protocols. If a generator function is decorated with the `contextlib.contextmanager` decorator, it will return a context manager implementing the necessary `__enter__()` and `__exit__()` methods, rather than the iterator produced by an undecorated generator function.

**contextlib module**  
See Also:
[http://docs.python.org/py3k/library/contextlib.html#module-contextlib](http://docs.python.org/py3k/library/contextlib.html#module-contextlib)

**With Statement Context Managers**  
See Also:
[http://docs.python.org/py3k/reference/datamodel.html#context-managers](http://docs.python.org/py3k/reference/datamodel.html#context-managers)

The `with` statement  
See Also:
[http://docs.python.org/py3k/reference/compound_stmts.html#with](http://docs.python.org/py3k/reference/compound_stmts.html#with)

The `with` statement is used to wrap the execution of a block with methods defined by a context manager (see section With Statement Context Managers). This allows common try...except...finally usage patterns to be encapsulated for convenient reuse.

**python dict**  
See Also:
[http://docs.python.org/py3k/library/stdtypes.html#mapping-types-dict](http://docs.python.org/py3k/library/stdtypes.html#mapping-types-dict)
**python filesystem**  See Also:

- [http://docs.python.org/py3k/library/filesys.html](http://docs.python.org/py3k/library/filesys.html)

**python filesystem shutil**  See Also:

- [http://docs.python.org/py3k/library/shutil.html](http://docs.python.org/py3k/library/shutil.html)

**copytree example**  This example is the implementation of the `copytree()` function, described above, with the docstring omitted. It demonstrates many of the other functions provided by this module.

```python
def copytree(src, dst, symlinks=False):
    names = os.listdir(src)
    os.makedirs(dst)
    errors = []
    for name in names:
        srcname = os.path.join(src, name)
        dstname = os.path.join(dst, name)
        try:
            if symlinks and os.path.islink(srcname):
                linkto = os.readlink(srcname)
                os.symlink(linkto, dstname)
            elif os.path.isdir(srcname):
                copytree(srcname, dstname, symlinks)
            else:
                copy2(srcname, dstname)
        # XXX What about devices, sockets etc.?
        except (IOError, os.error) as why:
            errors.append((srcname, dstname, str(why)))
    # catch the Error from the recursive copytree so that we can
    # continue with other files
    except Error as err:
        errors.extend(err.args[0])
    try:
        copystat(src, dst)
    except WindowsError:
        # can’t copy file access times on Windows
    except OSError as why:
        errors.extend((src, dst, str(why)))
    if errors:
        raise Error(errors)
```

Another example that uses the `ignore_patterns()` helper:

```python
from shutil import copytree, ignore_patterns

copytree(source, destination, ignore=ignore_patterns('*.pyc', 'tmp*'))
```

This will copy everything except `.pyc` files and files or directories whose name starts with `tmp`.

**python memoryview**  See Also:

[http://docs.python.org/py3k/library/stdtypes.html#memoryview](http://docs.python.org/py3k/library/stdtypes.html#memoryview)
Memoryview objects allow Python code to access the internal data of an object that supports the buffer protocol without copying.

Memory is generally interpreted as simple bytes.

**python str**  
See Also:
- [http://docs.python.org/py3k/library/stdtypes.html#sequence-types-str-bytes-bytearray-list-tuple-range](http://docs.python.org/py3k/library/stdtypes.html#sequence-types-str-bytes-bytearray-list-tuple-range)
- [http://docs.python.org/py3k/library/functions.html#str](http://docs.python.org/py3k/library/functions.html#str)
- *Python parse module*

---

**What is a native string**  
See Also:
- [http://www.python.org/dev/peps/pep-3333/#a-note-on-string-types](http://www.python.org/dev/peps/pep-3333/#a-note-on-string-types)

If you don’t know what a native string is, then you need to study more to understand why Armin’s PEP exists and why it is useful. I suggest starting with PEP 3333 (the WSGI update to v1.0.1 that first clearly defined the concept of a native string: [http://www.python.org/dev/peps/pep-3333/#a-note-on-string-types](http://www.python.org/dev/peps/pep-3333/#a-note-on-string-types)).

There are concrete, practical reasons why the lack of Unicode literals in Python 3 makes porting harder than it needs to be. Are they insurmountable? No, of course not - there are plenty of successful ports already that demonstrate porting it quite feasible with existing tools. But the existing approaches require that, in order to be forward compatible with Python 3, a program must be made worse in Python 2 (i.e. harder to read and harder to write correctly for someone that hasn’t learned Python 3 yet). Restoring unicode literal support in 3.3 is a pragmatic step that allows a lot of code to just work on Python 3. Most 2.6+ code that still doesn’t work on Python 3 even after this change will be made better (or at least not made substantially worse) by the additional changes necessary for forward compatibility.

Unicode literals are somewhat unique in their impact on porting efforts, as they show up everywhere in Unicode correct code in Python 2. The diffs that will be needed to correctly tag bytestrings in such code under Python 2 are tiny compared to those that would be needed to strip the `u"""` prefixes.

Regards, Nick.

**String literals**  
See Also:
- [http://docs.python.org/py3k/reference/lexical_analysis.html#strings](http://docs.python.org/py3k/reference/lexical_analysis.html#strings)

**String methods**  
See Also:
- [http://docs.python.org/py3k/library/stdtypes.html#string-methods](http://docs.python.org/py3k/library/stdtypes.html#string-methods)

**String services**  
See Also:
- [http://docs.python.org/py3k/library/strings.html](http://docs.python.org/py3k/library/strings.html)
Certain objects available in Python wrap access to an underlying memory array or buffer. Such objects include the built-in bytes and bytearray, and some extension types like array.array.

Third-party libraries may define their own types for special purposes, such as image processing or numeric analysis.

---

**Buffer protocol**  See Also:

http://docs.python.org/py3k/c-api/buffer.html#bufferobjects

---

**python doc reference**  See Also:

http://docs.python.org/py3k/reference/index.html

---

**python compound statements**

---

**python decorator (function and class definitions)**  See Also:

• http://docs.python.org/py3k/glossary.html#term-decorator

---

**Class definitions**  See Also:

• http://docs.python.org/py3k/reference/compound_stmts.html#class

---

**Function definitions**  See Also:

• http://docs.python.org/py3k/reference/compound_stmts.html#function-definitions

• http://docs.python.org/py3k/reference/compound_stmts.html#the-with-statement

• http://effbot.org/zone/python-with-statement.htm

---

**python datamodel**

---

**python call (emulating callable objects)**  See Also:

• http://docs.python.org/py3k/reference/datamodel.html#emulating-callable-objects

• http://docs.python.org/py3k/reference/expressions.html#calls

• http://www.gabes.fr/jean/2010/02/15/partie-a-trois-python-__slots__-et-metaclass/

---

**python metaclass (customizing class creation)**  See Also:

• http://docs.python.org/reference/datamodel.html#customizing-class-creation

• http://docs.python.org/py3k/reference/datamodel.html#customizing-class-creation

• http://www.gabes.fr/jean/2010/02/15/partie-a-trois-python-__slots__-et-metaclass/

• http://www.gabes.fr/jean/tag/metaclass/

• http://www.afpy.org/Members/kerflyn/metaklasse
Introduction

This tutorial is a practical exploration of using Python coroutines (extended generators) for solving problems in data processing, event handling, and concurrent programming. The material starts off with generators and builds to writing a complete multitasking environment that can run thousands of concurrent tasks without using threads or using code based on event-driven callbacks (i.e., the “reactor” model).

python and FFI

python cffi  See Also:
http://cffi.readthedocs.org/

Contents

• python cffi
  - Version 0.1 (18 june 2012)
Version 0.1 (18 June 2012)  Hi all,

We (fijal and myself) finally released the beta-0.1 version of CFFI.

It is a(nother) simple Foreign Function Interface for Python calling C code. I talked about it with a few python core people during the PyCon sprint; now it’s done, with a pure Python part and a compact (but still 3000 lines) piece of C code.

The goal is for it to be simple yet as complete as possible; it can be used in places where ctypes (say) is not applicable or only with platform-specific difficulties, e.g. to rewrite a “_curses” module in pure Python, or access the X libraries, etc.

Of course I’m not going to suggest that it should be part of the standard library right now, but I do hope that over time, should it prove useful and used, I could come back and make such a suggestion.

In any case it looks like we are going to write native and JITted PyPy support for it, and change our pure Python “_ctypes” implementation to be based on “cffi”.

As it is much more compact to support than the full _ctypes, it is also good for Jython and IronPython.

A bientôt, Armin.

Python flavors

cpython

Contents

- cpython
  - Source code
  - Issue tracker for C python
  - C python
  - cpython sur android

Source code  See Also:

- http://blog.python.org/2012/06/mercurial-mirrors-provided-by-atlassian.html
- https://bitbucket.org/python_mirrors/cpython

Issue tracker for C python  See Also:

http://bugs.python.org

The Issue tracker for C python is roundup.

C python

cpython development
cython development with mercurial  See Also:

- http://potrou.net/hgdevguide/
- http://hg.python.org/cpythonextrahist/
- hg clone http://hg.python.org/cpython
- http://mercurial.selenic.com/wiki/HardlinkedClones
- http://www.selenic.com/mercurial/hgrc.5.html#paths
- http://selenic.com/repo/hg/help/urls

Github clone, https://github.com/schmir/python

de Ralf Schmitt <schmir@gmail.com>

I’m very happy to announce that the core Python repository switch to Mercurial is complete and the new repository at http://hg.python.org/cpython/ is now officially open for cloning, and for commits by those who had commit access to SVN.

I’ve setup a git mirror of that repository on github: https://github.com/schmir/python

Mercurial scripts for python

python mercurial patch

"Martin v. LÖwis" <martin@v.loewis.de>

I get "unknown revision" (listing the full expression text) when using Mercurial 1.6.3 (default version in Ubuntu 10.10).

Based on Baptiste’s approach, I propose the script below to compute a patch. Please report whether it works for you.

Regards, Martin

#!/bin/sh

base=`hg log --template {rev} -r’max(ancestors(default)-outgoing())’`
hg diff -r$base

Subversion to Mercurial transition News

2.17. Languages
Guido van Rossum <guido@python.org> wrote:
> What is "rebase"? Why does everyone want it and hate it at the same time?

It’s the same thing that happens when you do a “svn up” with local changes in your checkout. Logically, your patch gets modified so that it applies on a different (newer) version of the code.

If you don’t rebase then those modifications end up getting stored in the history as merge nodes. That’s quite messy in my opinion and generally not a good idea. Rebase is an important tool (which can at times be abused).

Regarding collapsing multiple comments (and rewriting history in general), I feel there are two main schools of thought. One school considers the development history of a change important and that it should be preserved: every step and misstep of development should end up in the public repository.

The other school, which I am a member of, considers a logical development sequence more important than actual development history. I like to see a feature or fix developed in smallish, logical steps and I’m willing to spend a lot of time to rewrite patches to make it happen. IMO, future maintainers will thank you for the effort.

Note though, when you are worked with a distributed system, you should not rebase commits that are in other people’s repositories. In practice this is generally not a problem. If you have a long lived branch that you are sharing with other people, you can have a agreement that everyone will destory their old copy when it is rebased. Alternatively, you just take care to only publicly push logical changes.

Regards,

Neil

Barry Warsaw <barry@python.org> wrote:
> I’m asking because I don’t know hg and git well enough to answer the
> question. In my own use of Bazaar over the last 4+ years, I’ve almost never
> rebased or even been asked to.

Maybe it depends on what kind of changes you commit. I consider future maintainers the most important “customer” of the repository history. As such, I try to make each commit a logical change that takes a working system and produces another working system. In that way, each change to be potentially reversed if later on if it found to cause problems. Also, ideally, each revision can be tested to narrow down the version where a bug was introduced.

I see a VCS system as having two related but different purposes. The first is to help keep track of changes when they are developed. This is messy. I don’t know about you but I make lots of mistakes: changes that don’t do what I want, crazy ideas that turn into dead ends, etc. I use a VCS to keep track of this experimentation and my incremental progress.

The second is to keep a record of the change history of a long lived piece of software. In that case, I like it that each change has a logical purpose. In the first case the “customer” is the developer. In the second it is the maintainer.
In this graph:

```
A --- B ------------.
/ \
... --- X --- C --- D --- E --- M
```

A and B do exist, but I shouldn’t care or notice them unless I explicitly drill down.

In my case, I usually have something like:

```
A --- B --- C
/
--- X --- F --- G --- H
```

A, B, and C are messy and not logical. Maybe I write them so I have two logical patches (assuming they are only in my private repo):

```
A' --- B'
/
--- X --- F --- G --- H
```

Next, putting the merge in public repository generally serves no purpose so I rebase on H:

```
--- X --- F --- G --- H --- A'' --- B''
```

This very much matches the result I would get if using CVS or Subversion. IMHO, the changes A, B, and C represent partially complete development and there is no purpose to put them in the public repo.

If you are able to directly commit A’ and B’ and your tool does a good job of hiding the logically unimportant merge then I guess you wouldn’t miss the ability to modify history.

---

Neil Schemenauer <nas@python.ca> writes:

> Regarding collapsing multiple comments (and rewriting history in general), I feel there are two main schools of thought. One school considers the development history of a change important and that it should be preserved: every step and misstep of development should end up in the public repository.

Yep, that’s the school I’m in. Other people don’t get to say what I would find useful, and the cost of having data there is very low compared to the inability to re-create it at the times when it’s needed.

> The other school, which I am a member of, considers a logical development sequence more important than actual development history.

That seems to be an artefact of VCS tools which force you to choose between those two. The reason I prefer Bazaar is that it gives me both without compromising either.
> I like to see a feature or fix developed in smallish, logical steps
> and I’m willing to spend a lot of time to rewrite patches to make it
> happen. IMO, future maintainers will thank you for the effort.

Right, and those logical steps are done as merges from the feature branch into the trunk (substitute those names as you
like). I consider the merging from one branch to another as the time to decide how to present my VCS work for others
to view.

I haven’t heard a useful case for rebase that I don’t get with Bazaar’s merging, default history presentation, and shelve
capability. And all of that without ever having to re-write history – nor even choose what valuable information to lose.

This philosophy is essentially what the “mq” extension to Mercurial tries to capture. In mq, you maintain a series of
patches “on top of” your repository, amending, refining and rebasing them as you wish until they are ready to commit,
at which time you take them off the patch queue and convert them into final commits in the repository.

The one downside of mq is that you do not get the usual benefits of distributed version control - local commits of your
work, branching to manage experiments, etc. This isn’t really surprising, as that sort of “messy” development doesn’t
really fit with a nice clean picture of logical and well-defined patches (at least, it doesn’t fit easily enough that it can
be automated :-)). There is a facility in mq to try to integrate the two, by versioning your patch queue, but that makes
my head hurt, so I can’t really comment on how useful that is...

For people in the “clean history” school, I’d recommend looking at mq for your personal use. But it’s definitely an
advanced feature of Mercurial, so it may be better to understand core Mercurial (and at least temporarily accept that
Mercurial is based on the “keep all history” school of thought, or you’ll struggle to match the assumptions of the
documentation to your thinking :-) before diving into mq.

PS You can do everything that mq provides using core Mercurial commands - and in theory, do it more safely - but it
won’t necessarily fit the way you think quite as well...
On Mon, Mar 28, 2011 at 8:13 PM, Paul Moore<p.f.moore@gmail.com> wrote:

For people in the "clean history" school, I’d recommend looking at mq for your personal use. But it’s definitely an advanced feature of Mercurial, so it may be better to understand core Mercurial (and at least temporarily accept that Mercurial is based on the "keep all history" school of thought, or you’ll struggle to match the assumptions of the documentation to your thinking :-) ) before diving into mq.

I’m seeing if I can get the best of both worlds by having a public sandbox repo where I work on things (which has the full messy history of development on its feature branches), and then just drop them into the main repo as coherent patches. Once I land a patch, I’ll close the original feature branch in the sandbox, so merge conflicts won’t be an issue.

Mercurial makes merging easy enough that I’m happy with the way that approach is working so far.

For any non-trivial work I think this is the best approach. You still get all the advantages of working with mercurial (able to commit frequently) without polluting the history of the core repository.

It has the major advantage of also being very simple to understand.

All the best,

Michael

CPython mercurial learning, march 23 march 2011

de  Dirkjan Ochtman <dirkjan@ochtman.nl>
heure de l’expéditeur  Envoyé à 08:51 (GMT+01:00). Heure locale : 09:54.
à  "Stephen J. Turnbull" <stephen@xemacs.org>
c  skip@pobox.com,
-python-dev@python.org
date  23 mars 2011 08:51
objet  Re: [Python-Dev] I am now lost - committed, pulled, merged, what is "collapse"?

On Wed, Mar 23, 2011 at 03:12, Stephen J. Turnbull <stephen@xemacs.org> wrote:
> No, software engineering scales up to a point, then it breaks and you
> need a serialization scheme. The problem is not the DVCS, it’s the
> demand for a *centralized*, authoritative, safe, stable version. DVCS
> can scale at least to Linux kernel pace if you only ask for
> centralized authoritative.<wink> DVCS is "No Silver Bullet", it only
> helps with some accidental costs of development. It doesn’t help with
> the costs of review and testing.

Yeah, Linux employs the other solution here (which Mercurial also uses, in fact, for development of Mercurial itself): only one person pushes to the central repository, that person pulls from other “staging” repositories as he is aware of changes being made.

> There are in fact problems with the current generation of DVCSes.
> Lack of plists on commits, for example. You’d like to have a "tested"
> property, in fact two of them (one for the committer, one for the
> buildbots). You’d like to have a "checkpoint" property, which commits
would by default be ignored by "log" and "bisect". You’d like to have
an "issues" property, a list of issues applicable to this commit. But
again, these would only reduce accidental costs.

Mercurial does in fact have a mechanism to attach arbitrary metadata to changesets (the extra dictionary), but there’s
no easy access from the command-line. One could also argue that this is basically just a special case of smart commit
message formatting, which python-dev already does, and could extend. (For example, Mozilla sometimes closes their
tree to general commits, but then has a CLOSED tag that can be put in a commit message to override that.)

Cheers,

Dirkjan

de Éric Araujo <merwok@netwok.org>
heure de l’expéditeur Envoyé à 04:28 (GMT+01:00). Heure locale : 10:27.
à "Stephen J. Turnbull" <stephen@xemacs.org>
cc Antoine Pitrou <solipsis@pitrou.net>,
python-dev@python.org
date 23 mars 2011 04:28
objet Re: [Python-Dev] Workflow proposal

So what you’re saying is that Mercurial by itself can’t support the recommended workflow, because any “col-
lapsing” of commits requires stripping, whether done by hg strip or implicitly by some other “non-average” hg
command.

Pretty average: http://mercurial.selenic.com/wiki/PruningDeadBranches

I don’t see the connection; mq supplies “qfinish” for the purpose of turning a patch into a commit. All I’m
suggesting is that “qrefresh” is a nicer way to handle both the collapsing process and the strip/ re-merge/recommit
process, although there is the problem of reverting the commit back to an mq patch, which AFAIK requires a “strip
–keep” followed by “qnew”.

I like mq as a power tool, but only for short-term work. Most of the time I don’t need it. Refreshing is painful
to me whereas merging is straightforward. Mercurial devs themselves advocate real branches (named, pbranch or
what-have-you) for long-term work.

Regards

de Éric Araujo <merwok@netwok.org>
heure de l’expéditeur Envoyé à 02:01 (GMT+01:00). Heure locale : 11:11.
à Glenn Linderman <v+python8@g.nevcal.com>
cc python-dev@python.org
date 23 mars 2011 02:01
objet Re: [Python-Dev] Hg: inter-branch workflow

I’m curious: what are the benefits of the Mercurial model?
Simplicity.
That’s an amusing response, after reading hundreds of emails on this
list

I think the great number of messages is caused by incomplete learning, confusion caused by familiarity with other
tools, FUD and anxiety.

Just in case my position is not clear: I have a lot of sympathy for people who struggle with the new tool and workflow
and do try to help.
Development tools, Release 2012.06.18

> about 5-12 step sequences of commands required to perform one operation.

Committing, merging and sharing are not one operation.

> So I must ask: where is the simplicity manifested?

I used simplicity in its strict (or scientific, if this speaks to you) meaning: not complex, not many-sided. Having one thing (a notion of an unnamed branch) instead of two things (a notion of trunk and a notion of side branch) is by definition simpler.

I often describe Mercurial’s simplicity (or chosen stupidity) as the reason for its robustness and consistency. (For example, the frowning upon history rewriting, or the absence of octopus merging.)

That said, I cannot learn Mercurial for you, not force you to read the devguide. I would advise you to learn Mercurial with one of the many available resources (http://hginit.com/, http://mercurial.aragost.com/kick-start/), proceed to the devguide, clone https://bitbucket.org/mirror/cpython and start fixing things in http.server (and thanks in advance for that work!).

Regards

Re: [Python-Dev] Workflow proposal

de Antoine Pitrou <solipsis@pitrou.net>

On Wed, 23 Mar 2011 12:30:17 +0900
"Stephen J. Turnbull" <stephen@xemacs.org> wrote:
> Antoine Pitrou writes:
> 
> > Now, “hg strip” should definitely be absent of any recommended or even
> > suggested workflow. It’s a power user tool for the experimented
> > developer/admin. Not the average hg command.
> >
> > So what you’re saying is that Mercurial by itself can’t support the
> > recommended workflow, because any “collapsing” of commits requires
> > stripping,

Not really. It requires that you either:

- work on your long-term features in a separate repo (and produce a diff at the end that you will apply to the main repo)
- use mq
- use a non-committing equivalent of mq (iterate on a patch which you periodically save with “hg di”, for example; that’s what I do for most patches)

Apparently some of you think “collapsing” should involve some specific hg command. It doesn’t. Perhaps the devguide should be rephrased there.

Regards
Antoine.
Re: [Python-Dev] Let's get PEP 380 into Python 3.3

I don’t have push rights so I can not push anything to <http://hg.python.org/>. However, I tried to click the “server side clone”, and to my surprise, it worked. I was expecting that some verification would be made, but there is now a new feature branch at: <http://hg.python.org/features/pep-380/>. Sorry to have done that, once again, I was not expecting to actually have the rights to cause any side-effect on <http://hg.python.org/>. I guess that it may not be desirable to allow anyone do such server-side clone.

Anyway, I can not push to this feature branch, so for now it’s just a clone of the current tip.

Done here: <https://bitbucket.org/rndblnch/cpython-pep380/>

The pep-380 changeset is pushed on top of current tip and is visible here: <https://bitbucket.org/rndblnch/cpython-pep380/changeset/40b19d1933ea>

renaud

Re: [Python-Dev] I am now lost - committed, pulled, merged, what is "collapse"?

Note that some of that isn’t unique to the Mercurial transition (specifically, the server side hook that complained about the whitespace change existed in SVN as well).

One key difference that we could historically ignore with SVN is that it would happily accept changes from an out-of-date working copy, so long as the committed changes didn’t alter any files that had also been updated on the head. In practice, this did mean a lot of commits did get rejected, since Misc/NEWS would often conflict. Mercurial’s changeset based view of the world means it wants to know how the changesets relate to each other even when they affect different files, unlike SVN which would happily do an implicit merge (creating a state in the central repo that no developer has ever tested locally).

So the full flow for a trunk commit in SVN was:
svn update # change stuff svn commit -m “Whatever” # get rejected by server side hooks (e.g. changed files, bad whitespace) make patchcheck # fix whitespace issues svn update # get updated files # Resolve any conflicts, rerun relevant tests svn commit -m “Whatever”

Mercurial isn’t really all that different, but it’s distributed nature means it want to keep track of even minor things like the local whitespace fixes and the merger of your changes with the other changes that were pushed since you started work. So the example above becomes something like:

hg pull -u #change stuff hg commit -m “Whatever” hg push # get rejected by server side hooks (e.g. changed files, bad whitespace) make patchcheck hg commit -m “Fix whitespace” hg pull -u hg merge # Resolve any conflicts, rerun relevant tests hg commit -m “Merge with remote” hg push

It definitely produces some additional noise on python-checkins, since things that would normally have been resolved privately by the developer before SVN accepted the commit are now being published to the mailing list. To help with that, I’ve personally split my python-checkins label into two: one which contains everything from the list, and another which filters out any messages with “merge” in the subject line.

You may also find “hg outgoing” useful, since that will tell you all the changesets that a call to “hg push” will publish, rather than just the latest changeset as shown by “hg heads.”

Cheers, Nick.

de Nick Coghlan <ncoghlan@gmail.com> heure de l’expéditeur Envoyé à 00:41 (GMT+10:00). Heure locale : 22:25. à Guido van Rossum <guido@python.org> cc Antoine Pitrou <solipsis@pitrou.net>, “Martin v. Löwis” <martin@v.loewis.de>, Raymond Hettinger <raymond.hettinger@gmail.com>, python-dev@python.org cci date 20 mars 2011 00:41 objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is “collapse”? liste de diffusion python-dev.python.org Filtrer les messages de cette liste de diffusion envoyé par python.org signé par python.org

Se désabonner Se désabonner de cette liste de diffusion Ce message est important de par son contenu, principalement.

masquer les détails 20 mars (Il y a 2 jours) On Sun, Mar 20, 2011 at 9:32 AM, Guido van Rossum <guido@python.org> wrote: > +1. Just as I hope for the Python 3-4 transition, I hope that whatever > comes along next has a better transition strategy. That would make it > really hot.

Given that the hardest part of any transition is updating developers’ brains, that would be some pretty damn neat neurohacking right there :

Cheers, Nick.

de Paul Boddie <paul@boddie.org.uk> heure de l’expéditeur Envoyé à 01:52 (GMT+01:00). Heure locale : 13:23. à python-dev@python.org date 20 mars 2011 01:52 objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is “collapse”? liste de diffusion python-dev.python.org Filtrer les messages de cette liste de diffusion

Skip wrote: > Antoine wrote: > > > 94 changesets? If you want to avoid risking conflicts, you should “hg > > pull” and “hg up” (or “hg pull -u”) before you start working on > > something (just like you “svn up”’ed before working on something). > > Sorry, this workflow is still new and hugely confusing to me. To make a > > simple edit to csv.rst I needed to do a couple pulls and merges, local > > commits, resolve the same conflict multiple times, get rebuffed when > I > first pushed because there was a tab in the file, and ask a question here. > > If this is the typical route necessary to make even the simplest changes > which would have been a single commit with svn, my already meager rate of > contributions is likely to wither away to nothing.

This is reminiscent of a message on the Mercurial mailing list recently, to which I responded with a question about people experiencing problems with Mercurial because they are used to file- or directory-specific operations in other systems, elicitng this reply:

“hg failed saying there were uncommitted changes (his source code changes in another part of the tree). He didn’t want to commit those changes yet, they weren’t finished. So he was stuck. To his mind, hg was being stupid because it was getting in his way, “unnecessarily” linking changes in the two sets of files together. The concept of a revision being a state of the whole repo was alien. For most people that concept came in with svn.”

2.17. Languages 589
I’ve spent some time trying to tidy up and improve a document on the Mercurial Wiki about CVS-like working practices with Mercurial, and this might explain the differences in operation between CVS/Subversion and Mercurial, although it doesn’t yet distinguish between the “narrow” file-specific commits you get in systems like CVS and Subversion, and the whole-project commits you get in systems like Mercurial:

http://mercurial.selenic.com/wiki/CvsLikePractice

(By the time you look at that page, I might have added something, though.)

I’m certainly no expert with Mercurial, and I’ve only been using it as a personal tool since the MoinMoin guys introduced me to it back at EuroPython 2006, so even now the “right way” or “best practice” when it comes to propagating fixes between branches/clones is something I’m still figuring out, but having lurked on the recent python-dev threads and having read various queries on the Mercurial list over the past year or so, I get the impression that reaching for things like rebase and mq as the first choice to solve various workflow problems would get some blunt criticism on the Mercurial list.

That said, I can’t readily find any good guides to managing a multi-branch project, but there seem to be some interesting techniques documented for some of the situations people are likely to encounter. For example:

http://mercurial.selenic.com/wiki/DaggyFixes describes pulling fixes selectively which I’ll probably have to try on some personal project at some point

http://hgitinit.com/05.html describes the presumably common way of propagating fixes from stable branches to development branches

Certainly, the Python devguide is a nice piece of documentation, but I feel that there’s an opportunity here to address some documentation issues that would also help others using and adopting Mercurial. For example, submitting “clean” changes to a project (that “collapse” thing) is a topic for a document that could usefully be written, containing some nice diagrams that explain the mechanism to newcomers, and it would surely benefit more than just the CPython development effort.

Paul

Jesus Cea writes:

> I think we are doing some antipatterns with our current approach, > battling the tools instead of “joining them”.

Yes. That is deliberate; see PEP 0374. I admit I personally didn’t foresee the issues Nick describes with the flow of patches from one branch to another. Also, the “collapse” stage was completely unexpected to me (and wasn’t stated as a requirement for PEP 0374), although in hindsight it seems natural for Python.

The problem with “joining” the tools is that is a euphemism for “RTFM, OK?” People who are already working with hg and similar tools in their daily lives can probably make that transition in Python’s workflow pretty easily, but they are not going to be very good at providing scripts for the DVCS newbies because they don’t have them; they Just Do It. The DVCS newbies, OTOH, were told that the workflow would change as little as possible, and we (the PEP 0374 authors) did our best to make that possible. Unfortunately, it hasn’t worked as well as hoped. I, for one, am sorry about that.

I’m coming to the conclusion that those who say that Mercurial documentation should be found at the Mercurial project are wrong. I think there’s a reasonably strong case (based on the explicit promise of PEP 0374 that workflows would change as little as possible) for a follow-on informational PEP providing verbal, and maybe automated, scripts for the various operations needed in the Python workflow.

OTOH, people who are having problems with the workflow imposed by Mercurial need to recognize that Subversion basically ripped a big hole in the QA aspect of Python’s workflow. As Nick points out, Subversion merges create new versions in the repository that never existed in any developer’s workspace and therefore was never tested before committing. This is somewhat mitigated by buildbot testing, but that is mostly unit testing and inherently is not very
good at catching problems due to interactions across modules. That is, it’s not that Subversion provided a simpler way of doing the work. Rather, it hid the fact that certain work was not being done at all. hg exposes this fact.

skip@pobox.com writes:

> Can you provide an example of this (real or hypothetical)?

What happened to you is a real example. With svn it would go through, and any problems would be caught later. With hg, it doesn’t.

This is basically the statistical idea of false positives vs. false negatives. svn tends to generate false negatives (tests don’t fail until after the merge), while hg tends to generate false positives (merges fail because they aren’t “fast-forwards”).

> It seems to me that if all you have is unit tests it matters > little, if at all, whether you catch test failures before a checkin > (in your local hg repo) or after (using svn + a buildbot run).

In the long run, that’s right. In the short run the choice is between imposing a substantially higher burden of merging and testing on individual developers in order to get a higher quality shared branches versus imposing a detectably higher burden on the project as whole in terms of red buildbots. It clearly matters to the individual developers, and core developers are likely to have different interests here from the occasional committers.

The burden on the project could get very high; that’s why the Linux kernel went to Bitkeeper, then git, rather than Subversion. It was necessary to impose as much burden on the branches as possible before merging to trunk. But then Python doesn’t have anywhere near the change rate that the kernel does (Lennart R’s woes notwithstanding).

I’m not going to take a position on what’s best for the Python project; I’ve drunk the DVCS Kool-Aid and am very much in Jesus’s “let’s work with the tools, not against them” camp. But there’s an uncertain amount of bias in that opinion, and I’m not a committer so can’t speak from experience with the Python workflow.

> The workflow of the version control system isn’t going to magically > expose semantic interactions across modules. It seems to me that > at best the new workflow means there is a better chance of catching > textual conflicts before checkin than before.

No, at best the DVCS workflow forces the developer on a branch to merge and test the revisions that will actually be added to the repository, and perhaps notice system-level anomalies before pushing. Whether that is significant in practice is as yet a matter of faith for me. But again that’s a personal opinion, not yet a recommendation for the project.
On Mon, 21 Mar 2011 14:07:46 +0900, “Stephen J. Turnbull” <stephen@xemacs.org> wrote: > No, at best the DVCS workflow forces the developer on a branch to merge and test the revisions that will actually be added to the repository, and perhaps notice system-level anomalies before pushing.

hg does not force the developer to test, it only forces the merge.

As far as I can see, the only difference between hg and svn in this regard is that svn merging was easier, because, as you say, it was done behind the scenes when one did a conflict-free commit. If there were conflicts, though, you had the same need to merge to tip as with hg, and the same lack of enforcing of the running of tests.

– R. David Murray http://www.bitdance.com

de Stephen J. Turnbull <stephen@xemacs.org> heure de l’expéditeur Envoyé à 10:33 (GMT+09:00). Heure locale : 19:12. à “R. David Murray” <rdmurray@bitdance.com> cc python-dev@python.org cci date 21 mars 2011 10:33 objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is “collapse”? liste de diffusion python-dev.python.org Filtrer les messages de cette liste de diffusion

18. David Murray writes:

> On Mon, 21 Mar 2011 14:07:46 +0900, “Stephen J. Turnbull” <stephen@xemacs.org> wrote: > > No, at best the DVCS workflow forces the developer on a branch to merge and test the revisions that will actually be added to the repository, and perhaps notice system-level anomalies before pushing. > > hg does not force the developer to test, it only forces the merge.

I didn’t say any VCS forces the test; I said that the workflow can (in the best case). That’s also inaccurate, of course. I should have said “require”, not “force”.

My understanding is that svn does not detect fast forwards, only lack of conflicts, and therefore in case of concurrent development it is possible that the repository contains a version that never existed in any developer’s workspace. If that is not true, I apologize for the misinformation.

If it is true, by definition developers cannot test or review what hasn’t existed in their workspace; that testing and review is therefore imposed on the project as a whole, and perhaps not done until more concurrent commits have been made. On the other hand, in a DVCS this can’t happen under normal circumstances.

> As far as I can see, the only difference between hg and svn in this regard is that svn merging was easier, because, as you say, it was done behind the scenes when one did a conflict-free commit.

That’s true from the point of view of the individual developer; the DVCS requires more effort of her. That is not true from the point of view of the whole project however.

It would be possible for the svn-based workflow to require that after testing in one’s workspace, one does an svn update, and if any changes are made to files in the workspace, the whole build and test procedure must be repeated. I don’t see that that has advantages over the hg workflow, though – it should cause an addition build-test cycle in exactly the same revision sequences that the hg workflow does.

de “Martin v. Löwis” <martin@v.loewis.de> heure de l’expéditeur Envoyé à 10:44 (GMT+01:00). Heure locale : 11:16. à “Stephen J. Turnbull” <stephen@xemacs.org> cc python-dev@python.org cci date 21 mars 2011 10:44 objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is “collapse”? liste de diffusion python-dev.python.org Filtrer les messages de cette liste de diffusion envoyé par python.org signé par python.org Se désabonner Se désabonner de cette liste de diffusion

Ce message est important de par son contenu, principalement.

masquer les détails 10:44 (Il y a 32 minutes) > My understanding is that svn does not detect fast forwards, only lack of conflicts, and therefore in case of concurrent development it is possible that the repository contains a version that never existed in any developer’s workspace.

I can’t understand how you draw this conclusion (“therefore”).

If you do an svn up, it merges local changes with remote changes; if that works without conflicts, it tells you what files it merged, but lets you commit.
Still, in this case, the merge result did exist in the sandbox of the developer performing the merge. Subversion never ever creates versions in the repository that didn’t before exist in some working copy. The notion that it may have done a server-side merge or some such is absurd.

> If it is true, by definition developers cannot test or review what hasn’t existed in their workspace; that testing and review is therefore imposed on the project as a whole, and perhaps not done until more concurrent commits have been made.

You make it sound as if you have never used subversion.

Regards

de Stephen J. Turnbull <stephen@xemacs.org> heure de l’expéditeur Envoyé à 13:34 (GMT+09:00). Heure locale : 22:18. à “Martin v. Löwis” <martin@v.loewis.de> cc python-dev@python.org cci date 21 mars 2011 13:34 objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is “collapse”? liste de diffusion python-dev.python.org Filtrer les messages de cette liste de diffusion envoyé par python.org signé par python.org Se désabonner Se désabonner de cette liste de diffusion

Ce message est important, en particulier parce que vous avez participé à la conversation.

masquer les détails 13:34 (Il y a 44 minutes) “Martin v. Löwis” writes:

>> My understanding is that svn does not detect fast forwards, only lack of conflicts, and therefore in case of concurrent development it is possible that the repository contains a version that never existed in any developer’s workspace. I can’t understand how you draw this conclusion (“therefore”).

A fast forward is a case where some ancestor of the workspace is the tip of the repository. When the tip is not an ancestor, it must contain changes not yet in the workspace. If a VCS does not check for fast-forward, then if those changes are in files not changed in the workspace, there will be no conflict, and in theory there could indeed be a silent server-side merge. QED, “therefore”.

This seems especially plausible for VCSes that allow only a subset of files to be committed/pushed.

>> Subversion never ever creates versions in the repository that didn’t before exist in some working copy.

John Arbash-Meinel disagrees with you, so I think I’ll go with his opinion absent a really convincing argument otherwise. No disrespect to you intended, but John is an expert I’ve known for years.

>> The notion that it may have done a server-side merge or some such is absurd.

False, quite possibly: I’m not an expert on Subversion internals. Absurd, definitely not. CVS does it (and much worse, but it certainly does this too).

>> You make it sound as if you have never used subversion.

These days, it’s awful hard to avoid using Subversion. However, I have no experience with committing in Python, and I don’t have that much experience that I can claim to be authoritative, nor have I managed a multiuser Subversion repository.

de Stephen J. Turnbull <stephen@xemacs.org> heure de l’expéditeur Envoyé à 15:55 (GMT+09:00). Heure locale : 00:46. à skip@pobox.com cc python-dev@python.org cci date 21 mars 2011 15:55 objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is "collapse"? liste de diffusion python-dev.python.org Filtrer les messages de cette liste de diffusion

masquer les détails 15:55 (Il y a 50 minutes) skip@pobox.com writes:
I believe it runs counter to the professed intention of the switch away from a centralized version control system, to make it easier for more people to contribute to Python. It certainly seems harder for this old dog.

Well, you may be an old dog, but you're also an early adopter. That means both that you get to pay for our mistakes (our = authors of PEPs 374 and 385), and that it's going to take a while to disentangle implementation issues from the real long-run costs and benefits.

Costs of transition were admitted up front. The professed intention was to make things harder in the short run (but as little as possible!), while making contribution to Python significantly more attractive (but not necessarily less work!) in the long run. I don't think anybody tried to hide the fact that changing habits would be required, or to claim that it would be costless. There were a few people with a Pollyanna “try it, you'll like it” attitude, but certainly those of us involved in PEP 374 knew better than that — we knew there were people like you whose patterns of contribution worked just fine with the svn-based workflow and didn't need or want to change. That's why PEP 374 was necessary!

Yes, based on the description you give of your principal contribution pattern, you take a complexity/effort hit in the transition. I think it can be alleviated quite a bit with the help of your reports, but that will take some time. All I can say about that time is “Sorry!” and “Thank you for trying the system while it's still in beta.”

I hope you will give it some more time to shake down.

de Hrvoje Niksic <hrvoje.niksic@avl.com>
heure de l’expéditeur Envoyé à 15:56 (GMT+01:00). Heure locale : 16:47.
à python-dev@python.org
date 21 mars 2011 15:56
objet Re: [Python-Dev] I am now lost - committed, pulled, merged, what is "collapse"?

On 03/21/2011 01:34 PM, Stephen J. Turnbull wrote: > Subversion never ever creates versions in the repository that didn't before exist in some working copy. > John Arbash-Meinel disagrees with you, so I think I'll go with his opinion

Besides, it's easy to confirm:

# create a repository and two checkouts:

```bash
[~/work]$ svnadmin create repo
[~/work]$ svn co file:///home/hniksic/work/repo checkout1
Checked out revision 0.
[~/work]$ svn co file:///home/hniksic/work/repo checkout2
Checked out revision 0.
```

# add a file to checkout 1:

```bash
[~/work]$ cd checkout1
[~/work/checkout1]$ touch a && svn add a && svn commit -m c1
A
Adding a
Transmitting file data.
Committed revision 1.
```

# now add a file to the second checkout without ever seeing # the new file added to the first one:

```bash
[~/work/checkout1]$ cd ../checkout2
[~/work/checkout2]$ touch b && svn add b && svn commit -m c2
A
Adding b
Transmitting file data.
Committed revision 2.
```
The second commit would be rejected by a DVCS on the grounds of a merge with revision “1” never having happened. What svn calls revision two is in reality based on revision 0, a fact the DVCS is aware of.

The message “committed revision 2”, while technically accurate, is misleading if you believe the revision numbers to apply to the entire tree (as the svn manual will happily point out). It doesn’t indicate that what you have in your tree when the message is displayed can be very different from the state of a freshly-checked-out revision 2. In this case, it’s missing the file “a”:

```
[~/work/checkout2]$ ls
b
```

This automatic merging often causes people who migrate to a DVCS to feel that they have to go through an unnecessary extra step in their workflows. But once you grasp the “hole” in the svn workflow, what svn does (and what one used to take for granted) tends to become unacceptable, to put it mildly.

de  Barry Warsaw <barry@python.org>

à  python-dev@python.org
date  21 mars 2011 17:20
objet  Re: [Python-Dev] Hg: inter-branch workflow
liste de diffusion  python-dev.python.org  Filtrer les messages de cette liste de diffusion

On Mar 20, 2011, at 04:39 PM, Georg Brandl wrote:

>On 20.03.2011 16:21, Guido van Rossum wrote: >> What is “rebase”? Why does everyone want it and hate it at the same time? >>Basically, rebase is a way to avoid having pointless merge commits on the >same branch.

There’s something I don’t understand about rebase. It seems like most git and hg users I hear from advocate rebase, while (ISTM) few Bazaar users do.

I’d like to understand whether that’s a cultural thing or whether it’s a byproduct of some aspect of the respective tools.

It could be cultural in that communities using git and hg don’t want those local commits to ever show up in their shared repository, even though they are mostly harmless. In this graph:

```
A — B ————. >/>... — X — C — D — E — M
```

A and B do exist, but I shouldn’t care or notice them unless I explicitly drill down. The mainline still goes from X to C to D to E to M, and looking at the differences between E and M should tell me all I need to know. I.e. specifically, I can ignore A and B for most purposes.

It could be that some aspect of the tools causes A and B to not be hidden as well as they should, so that when looking at the history for example, the fact that A and B exist is a jarring or annoying artifact that would be better if they didn’t exist.

I’m asking because I don’t know hg and git well enough to answer the question. In my own use of Bazaar over the last 4+ years, I’ve almost never rebased or even been asked to. Not that some Bazaar users don’t use rebase, but I just don’t think it’s that common (John can correct me if I’m wrong).

I’m not trolling, I really want to understand.

-Barry

CPython hg new committers, march 16 March 2011

Brian Curtin <brian.curtin@gmail.com>

heure de l’expéditeur  Envoyé à 17:10 (GMT-05:00). Heure locale : 11:54.
à  Python-Dev <python-dev@python.org>

date  16 mars 2011 17:10
Hi all,

As I’m sure you’re all aware, the PyCon sprints are going on right now and will run for two more days. As a result, you may have noticed an increased number of patches over the last few days – many of these were from first-time contributors.

The turnout for the CPython sprint has been huge and I wanted to introduce and thank a number of people for coming out.

- Scott Wilson
- Denver Coneybeare
- Jeff Ramnani
- Alicia Arlen
- Michael Henry
- Natalia Bidart
- Mattias Bordese
- Robbie Clemons
- Evan Dandrea
- Jonathan Hartley
- Piotr Kaspyzyk
- Brandon Craig Rhodes

There were a few others from which I didn’t get their name, but thanks to everyone for your contributions!

http://wiki.python.org/moin/Pycon2011Sprint has a listing of what the new contributors have been working on, including the issues that have been submitted.

Additionally, I did a small write-up (with a picture!) of what everyone has been up to: http://blog.briancurtin.com/2011/03/16/pycon-2011-cpython-sprint-newcomers/

CPython hg new committers, march 14 march 2011
As agreed to at the language summit, I have pinged the IronPython, Jython and PyPy teams for committers on their respective teams who (do/did not) have commit rights prior to PyCon. These people are:

- Jeff Hardy (IronPython)
- Alex Gaynor (PyPy)
- Carl Friedrich Bolz (PyPy)
- Maciej Fijalkowski (PyPy)
- Antonio Cuni (PyPy)

And one other whose CLA I have in my bag (still haven’t unpacked from PyCon). So if you see them committing, this is a heads up.

A side note: The Jython guys would really love to join us in our HG future-land. Frank Wierzbicki expressed a lot of interest in this.

Jesse

CPython hg transition complete, march march 2011

I’m very happy to announce that the core Python repository switch to Mercurial is complete and the new repository at http://hg.python.org/cpython/ is now officially open for cloning, and for commits by those who had commit access to SVN.

The developers’ guide at http://docs.python.org/devguide/ has been updated to talk about Mercurial and should be enough to get anyone started with a clone.
We’ll work on extracting active feature branches into separate clones next; please let us know which branches these are (we already know of py3k-cdecimal, pep-3151 and pep-382).

To make new feature “branches” (ie. clones) that are to be available at hg.python.org, best use the “server side clone” feature that is available at http://hg.python.org/cpython/ in order to create the new repository.

To look up SVN revisions, use hg.python.org/lookup/rXXXXX. The tracker has also been updated to link to hg.python.org for files and revisions. For the future, it will recognize hg changeset hashes as well (without brackets, see the recent thread).

The buildbots should also now be building from the hg repositories.

Please let me know if you notice any disruptions, or anything else that needs fixing, or any other question.

Also please redirect praise to Antoine Pitrou and Dirkjan Ochtman who did most of the actual work.

Georg

python dev with mercurial announce, 25 February 2011

The announce by Antoine Pitrou

To: python-dev@python.org
From: Antoine Pitrou <solipsis@pitrou.net>
Date: Fri, 25 Feb 2011 01:19:04 +0100
Lines: 95
Subject: [Python-Dev] Mercurial conversion repositories

Hello,

Georg and I have been working on converting the SVN repository to Mercurial. We can now present you a test repository (actually, two).

CPython repository  See Also:

http://hg.python.org/cpython/

This is the main conversion repository. It contains all history of trunk and py3k (since 1990!) up to now, including all maintenance branches starting from 2.0 up to 3.2.

For core developer  If you are a core developer, get your local clone of the repository using:

$ hg clone ssh://hg@hg.python.org/cpython

(this uses the same SSH key as your Subversion access; for more information about Mercurial and SSH keys, see the converted development FAQ: http://potrou.net/hgdevguide/faq.html#faq )

For non-core developer  If you are not a core developer:

$ hg clone http://hg.python.org/cpython

Your clone will contain the following branches:
$ hg branches

default 68026:f12ef116dd10
3.2 68025:ceef92e1a323
2.7 68010:8174d00d0797
3.1 67955:5be8b695ea86
2.6 67287:5e26a860edcd
2.5 65464:e4ecac76e499
trunk 62750:800f6c92c3ed
3.0 60075:1d05144224fe
2.4 58552:d72c3ac1899e
2.3 45731:a3d9a9730743
2.2 40444:d55ddc8c8501
2.1 30171:06fcc9eeca8
2.0 18214:dc0def9565cd

The branch “default” is the current py3k branch from SVN. The branch “trunk” represents SVN trunk history until 2.7 was branched for maintenance.

The full list of tags is too long to print here, but you can get it using:

$ hg tags

The size of the repository on-disk is (depending on your filesystem):

$ du -hs .hg 176M .hg

(the size of the network transfer is estimated to be around 80MB)

You can commit and even push to this repository (the latter if you are a core developer). Since this is a test repository, whatever you push will be discarded when we do the final conversion.

**CPython with extra history**  See Also:

http://hg.python.org/cpythonextrahist/

This repository is bigger, and has a much more complicated topology. It is a superset of the other repository, and contains the totality of the branches from the SVN repository (it has more than 450 repository heads, of which 87 non-closed). It also weighs quite a bit more:

$ du -hs .hg 248M .hg

This repository is unnecessary for development work, since even for history-digging purposes the normal repository has the necessary information. This repository is only to preserve historical record of some of the non-trunk development work from the SVN repository (such as orphaned/deleted feature branches).

**Development guide**  See Also:

http://potrou.net/hgdevguide/

This is the development guide adapted for Mercurial.

**Sources**  You can get its sources from the branch “hg_transition” in http://hg.python.org/devguide/.

Regards

Antoine.
Barry Warsaw question

Date: Fri, 25 Feb 2011 11:12:53 -0500
From: Barry Warsaw <barry@python.org>
To: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

On Feb 25, 2011, at 01:50 AM, Raymond Hettinger wrote:

> On Feb 25, 2011, at 12:09 AM, Martin v. Löwis wrote:
> >> I think I would have liked the strategy of the PEP better (i.e.
> >> create clones for feature branches, rather than putting all
> >> in a single repository).
> >In my brief tests, the single repository has been easy to work with.
> >> If they were separate, it would complicate backporting patches
> >> and merges. So, I’m happy with how George and Benjamin put this together.
>
> The way I work with the Subversion branches is to have all the active branches checked out into separate directories
> under a common parent, e.g:

~/projects/python/py26
~/projects/python/py27
~/projects/python/trunk
~/projects/python/py31
~/projects/python/py32
~/projects/python/py3k

This makes it very easy to just cd, svn up, make distclean, configure, make to test things.

How can I do this with the hg clone when all the branches are in the single repository, but more or less hidden? After
doing the ‘hg clone’ operation specified by Antoine, I’m left with a single cpython directory containing (iiuc) the
contents of the ‘default’ branch.

I’m sure I’m not the only one who works this way with Subversion. IWBN to cover this in the devguide (or is it there
and I missed it?).

Cheers, -Barry

Response from adrian@cadifra.com  See Also:

https://bitbucket.org/abuehl

Date: Fri, 25 Feb 2011 18:40:53 +0100
From: Adrian Buehlmann <adrian@cadifra.com>

I know (almost) nothing about developing Python (this is my first post to this list after lurking for quite a while now),
but as a regular Mercurial contributor, I think the following could be useful for you:

First, get an initial clone (let’s name it ‘master’) over the wire using: [1]

    $ hg clone -U ssh://hg@hg.python.org/cpython master

Then create a hardlinked clone [2] for working in each branch, specifying the branch to check out using option -u:
This will be fast and save space as these local ‘branch clones’ will share diskspace inside .hg/store by using hardlinks, and you need to do the initial slow clone over the wire only once.

Note that each of these branch clones will initially have your local master repo as the default path [3,4]. If you’d like to have the default push/pull path to point to ssh://hg@hg.python.org/cpython instead, you’d want to edit the [paths] section in the .hg/hgrc file in each of the branch repos. But of course you can also leave the default paths as they are and synchronize via the master repo (e.g. pull new changesets into master first, and then pull into the specific branch repo).

Barry Warsaw question

Date: Fri, 25 Feb 2011 14:43:15 -0500
From: Barry Warsaw <barry@python.org>

Thanks very much Adrian, this is exactly what I was looking for. It maps almost directly to my current mental model for working on Python in Subversion (and truth be told, also how I do/did it with Bazaar).

It does leave me with an empty ‘master’ directory that I basically won’t touch, though I suppose I could hide it in a dot-filename. And I have to remember to fiddle with .hg/hgrc when I clone a new branch working directory, but I guess that’s mostly a one-time cost.

I’ll have to remember that ‘hg pull’ does not update the working copy by default, and eventually I’ll figure out the whole merge thing.

One immediate thing that I’m missing from Bazaar is that ‘bzr commit’ invokes my editor and always shows me a ‘diff -u’ in the commit message buffer. This is incredibly handy because I don’t have to remember to do the diff in a different window, and I always have all the information I want right there to craft the commit message. It doesn’t look like this is possible with ‘hg commit’ though, right?

Cheers, -Barry

> >
> > I’ll have to remember that ‘hg pull’ does not update the working copy by
> > default, and eventually I’ll figure out the whole merge thing.
You can use "hg pull -u" to update (and "hg pull -uv" if you want to see the list of updated files).
Mercurial branches are not bazaar branches

Date: Sat, 26 Feb 2011 01:49:46 +0100
Organization: Eric Conspiracy Secret Labs
To: Barry Warsaw <barry@python.org>
Cc: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

Hi,

Le 25/02/2011 20:43, Barry Warsaw a écrit :
> On Feb 25, 2011, at 06:40 PM, Adrian Buehlmann wrote:
> >> Note that each of these branch clones will initially have your local
> >> master repo as the default path [3,4]. If you'd like to have the default
> >> push/pull path to point to ssh://hg@hg.python.org/cpython instead, you'd
> >> want to edit the [paths] section in the .hg/hgrc file in each of the
> >> branch repos.

I plan on having one clone per branch but pushing and pulling from only one repository, so I don’t need to edit hgrcs.

> > It does leave me with an empty ‘master’ directory that I basically won’t > > touch, though I suppose I could hide it in a dot-filename. Or have the master clone do double duty as the py3k clone. (IOW, I have 2.7 3.1 3.2 py3k, I pull/push in py3k and also do merges and new features in py3k).

> > And I have to remember to fiddle with .hg/hgrc when I clone a new branch
> > working directory, but I guess that’s mostly a one-time cost.

You don’t, see above. I’ve wanted to tell you something for a long time: Mercurial branches are not at all like Bazaar branches.

See:

- http://mercurial.selenic.com/wiki/Branch

Anecdote: I migrated from Subversion to Mercurial a few years ago, and found Mercurial branches very intuitive. When I saw mentions of Bazaar branches I was driven nuts by (what I saw as) the conflation between a branch and a clone. Later I understood how it made sense from the perspective of Bazaar, so I shifted my judgment from “insanely confusing” to “a particular choice that I don’t approve” <wink>.

Bazaar terminology does not map as-is to mercurial  What I’m saying is that a lot of Bazaar terminology using “branch” does not map as-is to Mercurial. We clone a repo, we pull from a repo/clone, we have named branches (e.g. 3.2) in a repo, we have unnamed branches on one named branch. I think you know that already, since you went from using Bazaar terms applied to Mercurial to mixing terms from both (“clone a new branch working directory” → “clone a repo”, probably). I salute your willingness to learn Mercurial, considering how fluent (and fluffly) you appear to be with Bazaar.

> > I’ll have to remember that ‘hg pull’ does not update the working copy by
> > default,
That pull does not update is a feature: The operation between a remote repo and the local repo (the .hg directory) is separate from the operation from the local repo to the working directory. When you know that you want pull + update (like when you have a clean working directory), you use “hg pull -u”. (I don’t like the fetch command.)

Some mercurial resources
>> and eventually I’ll figure out the whole merge thing.

Without anything specific, I’ll point to some resources:

Short tutorial http://hginit.com/04.html

Concepts and examples
• http://mercurial.selenic.com/wiki/Merge

Longer narratives http://hgbook.red-bean.com/read/a-tour-of-mercurial-merging-work.html

Commit messages
:: > > One immediate thing that I’m missing from Bazaar is that ‘bzr commit’ invokes > > my editor and always shows me a ‘diff -u’ in the commit message buffer. This > > is incredibly handy because I don’t have to remember to do the diff in a > > different window, and I always have all the information I want right there to > > craft the commit message.

You speak to my heart, sir. In your ~/.hgrc, under the section [ui], set “editor = path/to/mercurial/source/hgeditor” and enjoy your diffs.

I use it and love it.

If you want to commit a subset of your local changes, I use http://mercurial.selenic.com/wiki/CrecordExtension , a curses-based diff selection UI that works like a charm.

Kind regards, your friendly Mercurial whippersnapper

Response from Barry Warsaw (to Eric Araujo)
:: > >Le 25/02/2011 20:43, Barry Warsaw a écrit : >> >> On Feb 25, 2011, at 06:40 PM, Adrian Buehlmann wrote: >> >> [snip] >>> >>> Note that each of these branch clones will initially have your local >>> >>> master repo as the default path [3,4]. If you’d like to have the default >>> >>> push/pull path to point to ssh://hg@hg.python.org/cpython instead, you’d >>> >>> want to edit the [paths] section in the .hg/hgrc file in each of the >>> >>> branch repos. >> > > >I plan on having one clone per branch but pushing and pulling from only > > one repository, so I don’t need to edit hgrcs.

So let’s start from the basics. I want separate working directories for each active line-of-development (I’ll use that term instead of “branch”), e.g. working directories containing the tip of the 2.6 LoD, 2.7, 3.1, 3.2, and 3.3. I will not be doing feature or bug development in any of these directories. They are purely for local tracking of the remote masters. Thus, I want to be able to synchronize any one of those LoDs against the remote master with one command, like I would using Subversion’s ‘svn up’.

I clone the remote repository using the command in the devguide, so I now have a ‘cpython’ directory containing the history of all LoDs, but with a working directory that reflects the ‘default’ LoD. Now, in the parent of ‘cpython’, I do the following to get my separate-directory-LoDs:
$ hg clone -u 2.6 cpython py26 $ hg clone -u 2.7 cpython py27 # repeat and rinse for all other active LoDs

(Aside: that cpython directory still bugs me. It doesn’t naturally reflect a LoD, so why do I have to stare at it? Yes, I can make it play double duty by naming it “3.3” or whatever and updating it to the 3.3 LoD, but that feels artificial.)

Now I want to synchronize my local py27 directory with the state of that LoD on python.org. This is a two step process:

$ cd py27 # now I want to synchronize
$ (cd ../cpython && hg pull)
$ hg pull -u

Editing hgrc to point to hg.python.org means the sync-to-remote-master operation is one command.

I could do this:

$ cd py27 # now I want to synchronize
$ hg pull -u ssh://hg@hg.python.org/cpython

but I’m not going to remember that url every time. It wouldn’t be so bad if Mercurial remembered the pull URL for me, as (you guessed it :) Bazaar does.

>>Anecdote: I migrated from Subversion to Mercurial a few years ago, and
>>found Mercurial branches very intuitive. When I saw mentions of Bazaar
>>branches I was driven nuts by (what I saw as) the conflation between a
>>branch and a clone. Later I understood how it made sense from the
>>perspective of Bazaar, so I shifted my judgment from “insanely
>>confusing” to “a particular choice that I don’t approve” <wink>.

That’s funny because to my eyes, Mercurial conflates “branches” and “clones”. Why a clone operation should give me the history for all lines-of-development inside a working directory for one “arbitrary” one strikes me as bizarre (pardon the pun :). I get that for folks who like the “svn switch” method of working on multiple LoDs, this probably makes a lot of sense. I don’t personally like or trust that approach much though.

>>What I’m saying is that a lot of Bazaar terminology using “branch” does
>>not map as-is to Mercurial. We clone a repo, we pull from a repo/clone,
>>we have named branches (e.g. 3.2) in a repo, we have unnamed branches on
>>one named branch. I think you know that already, since you went from
>>using Bazaar terms applied to Mercurial to mixing terms from both
>> (“clone a new branch working directory” → “clone a repo”, probably). I
>>salute your willingness to learn Mercurial, considering how fluent (and
>>fluffly) you appear to be with Bazaar.

This is an inevitable problem with trying to converse fluently about three major dVCSs and at least one or two other non-dVCSs. They all use the same words to mean vaguely similar things, but quickly get bogged down in the implement-ation details assigned to those words. It all makes perfect sense when you’ve been immersed in those technologies, but it makes discussions and comparisons exceedingly difficult. I’ve no doubt it’s doubly painful to many people who have no prior experience with a dVCS.

(Aside: Bazaar could have potentially eased these folks transition because you can use Bazaar just like you would Subversion - as a centralized VCS – without stopping others from using it with full dVCS features on the same code base. I don’t know if Mercurial offers the same flexibility.)

It’s a little like trying to teach Python to a Java programmer. “Object”, “Class”, “Instance”, “Import” all mean roughly similar things, which lulls you into a false sense of understanding. We learn by holding up the new to the light of the familiar and looking for interference patterns. :)}
I’ll have to remember that ‘hg pull’ does not update the working copy by default. That pull does not update is a feature: The operation between a remote repo and the local repo (the .hg directory) is separate from the operation from the local repo to the working directory. When you know that you want pull + update (like when you have a clean working directory), you use “hg pull -u”. (I don’t like the fetch command.)

Sure, I get that. Again, this feature appears odd because I have the full history of all LoDs embedded in a working directory of a single LoD. With the arrangement I outlined above (which is independent of the dVCS backing it), it makes no sense for each LoD working directory to contain all the history of all the other LoDs.

In Bazaar, a “branch” is an independent LoD and it’s “repository” contains only its own history. Sure, it might have identical history with other LoDs up to the point of divergence, and I have ways to efficiently share that history across multiple LoD working directories, but they are still separate and I don’t need them if I don’t care about them. With Mercurial, all history for all LoDs in a repository always come along for the ride.

You speak to my heart, sir. In your ~/.hgrc, under the section [ui], set “editor = path/to/mercurial/source/hgeditor” and enjoy your diffs.

Great, I’ll try that, thanks. One thing Mercurial and Bazaar definitely share is a wealth of magical awesomeness hidden in manpages, wiki pages, mailing list posts, people’s heads, configuration files, and source code. :) If you want to commit a subset of your local changes, I use http://mercurial.selenic.com/wiki/CrecordExtension, a curses-based diff selection UI that works like a charm.

I very rarely want to do that. It’s more common (but still, IME not that common) to commit the changes to just a few files at a time. One thing Bazaar has that’s very nice is the ability to “shelve” some changes for a time. Let’s say I’m working on a bug and I want to merge your changes in or sync to the master. I can shelve some or all of my uncommitted changes, which saves them essentially out-of-the-way patch files, and then reverts my uncommitted changes. Unshelving then is the process of re-applying those patch files, and yes, resolving conflicts.

This is also a great way to work when you want to do test-driven-development but need to do some exploration first. You can hack around non-TDD until you’re confident of your approach, shelve all your changes, then incrementally apply them back as you write each test. I’m sure Mercurial has something equally awesome lurking about. :)

-Barry

Response from David Murray to Barry

$ cd py27 # now I want to synchronize
$ hg pull -u ssh://hg@hg.python.org/cpython

but I’m not going to remember that url every time. It wouldn’t be so bad if Mercurial remembered the pull URL for me, as (you guessed it :) Bazaar does.

How does setting it in the hgrc differ from “remebering” it? I’ve never been comfortable with the bzr –remember option because I’m never sure what it is remembering. Much easier for me to see it in a config file. But, then, that's how my brain works, and other people’s will work differently.

On Feb 26, 2011, at 01:49 AM, Éric Araujo wrote:
Anecdote: I migrated from Subversion to Mercurial a few years ago, and found Mercurial branches very intuitive. When I saw mentions of Bazaar branches I was driven nuts by (what I saw as) the conflation between a branch and a clone.
Later I understood how it made sense from the perspective of Bazaar, so I shifted my judgment from insanely confusing to a particular choice that I don’t approve <wink>. That's funny because to my eyes, Mercurial conflates “branches” and “clones”. Why a clone operation should give me the history for all lines-of-development inside a working directory for one “arbitrary” one strikes me as bizarre (pardon the pun :). I get that for folks who like the “svn switch” method of working on multiple LoDs, this probably makes a lot of sense. I don’t personally like or trust that approach much though.

I agree with you that I don’t trust the ‘svn switch’ style. I also find it slow. However....

That pull does not update is a feature: The operation between a remote and the local repo (the .hg directory) is separate from the operation from the local repo to the working directory. When you know which directory you want pull + update (like when you have a clean working directory), you use `hg pull -u`. (I don’t like the fetch command.)

Sure, I get that. Again, this feature appears odd because I have the full history of all LoDs embedded in a working directory of a single LoD. With the arrangement I outlined above (which is independent of the dVCS backing it), it makes no sense for each LoD working directory to contain all the history of all the other LoDs.

In Bazaar, a "branch" is an independent LoD and its "repository" contains only its own history. Sure, it might have identical history with other LoDs up to the point of divergence, and I have ways to efficiently share that history across multiple LoD working directories, but they are still separate and I don’t need them if I don’t care about them. With Mercurial, all history for all LoDs in a repository always come along for the ride.

I find bazaar’s model confusing, and hg’s intuitive, just like Eric. And consider that I learned bazaar before mercurial. To me, it makes perfect sense that in a DVCS the “unit” is a directory containing a repository and a working copy, and that the repository is the repository. That is, it has everything related to the project in it, just like the master SVN repository does (plus, since it is a DVCS, whatever I’ve committed locally but not pushed to the master). To have a repository that only has some of the stuff in it is, IMO, confusing. I advocated for having all the Python history in one repo partly for that reason.

I can intellectually see your point about not really needing the stuff for the LODs if you are only working on LOD X, but just like ‘svn switch’ makes me uncomfortable, I’m just not comfortable not having the whole repo in there :)

As an example, with mercurial, I feel comfortable moving directories around and renaming them (with the help of google it took me about 1 minute to figure out how to keep the association between the repository instances intact). With bazaar I got in trouble trying to do that, because the interrelationship between the working copy directories (and their subsets of the repo?) and the master repo(?) was not clear. I never did figure out how to make it work, I ended up starting over with a new clone.

Now, as I get farther into learning mercurial I may well find things that are just as confusing as I found that aspect of bazaar, but at least I am comfortable with the outermost layer: the repo is the repo is the repo.

18. David Murray www.bitdance.com

Branches and heads

Daniel Stutzbach <stutzbach8google.com> wrote:

> > On Sat, Feb 26, 2011 at 9:55 AM, Antoine Pitrou <solipsis@pitrou.net> wrote:
> >
>> > > There is no such thing as an "unnamed branch". What would "hg branches"
>> > > show? An empty space?
> >
>> > I understand now why I was confused. I had previously read the sentence
Ok, so beware, the term “branch” can conflate two concepts:

- a path in the topology (or line of development)
- a “named branch” in hg terminology

So, actually, hg promotes a slightly different terminology:

- a “head” is a changeset without a child in the topology
- a “branch” usually means a “named branch”: a set of changesets bearing the same label (e.g. “default”); that label is freely chosen by the committer at any point, and enforces no topological characteristic (even though in practice it will have, since it’s the whole point from the user’s perspective, and also because hg’s default behaviour and concept of a “current branch” encourages it)

A (named) branch can have zero, one, or several heads:

- zero head: if all branch-local heads have a child in another named branch (for example, “trunk” is linearly followed by “2.7”)
- several heads: if several lines of development were started in this branch without bothering to give them separate names

When you have several heads on a branch, you can merge them together if you want to reconcile the lines of development they represent.

When you have several branches with at least one head each, you can also merge them together: you must be careful to choose which named branch the merge changeset will be part of (for example, if you want to merge “3.1” into “3.2”, you will certainly want the merge changeset to be part of “3.2”, otherwise “3.1” will get a lot of unwanted features ;-)).

Note: a branch with zero head is marked “inactive” in “hg branches”. This basically means that it has already been merged in another branch. (of course, you can still develop in that branch, which will certainly create a new head as soon as you commit your first new changeset)

Regards
Antoine.

Response from Brett Cannon to Barry

From: Brett Cannon <brett@python.org>
Date: Sat, 26 Feb 2011 12:09:58 -0800
To: Barry Warsaw <barry@python.org>
Cc: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

> > On Feb 26, 2011, at 01:49 AM, Éric Araujo wrote:
> >
> >> Le 25/02/2011 20:43, Barry Warsaw a écrit :
> >>> On Feb 25, 2011, at 06:40 PM, Adrian Buehlmann wrote:
> >>> > >> [snip]
> >>>>> Note that each of these branch clones will initially have your local
> >>>>> master repo as the default path [3,4]. If you’d like to have the
I plan on having one clone per branch but pushing and pulling from only one repository, so I don’t need to edit hgrcs.

So let’s start from the basics. I want separate working directories for each active line-of-development (I’ll use that term instead of "branch"), e.g. working directories containing the tip of the 2.6 LoD, 2.7, 3.1, 3.2, and 3.3. I will not be doing feature or bug development in any of these directories. They are purely for local tracking of the remote masters.

Thus, I want to be able to synchronize any one of those LoDs against the remote master with one command, like I would using Subversion’s ‘svn up’.

For other people’s benefit, LoD == line of development (I think).

I clone the remote repository using the command in the devguide, so I now have a ‘cpython’ directory containing the history of all LoDs, but with a working directory that reflects the ‘default’ LoD. Now, in the parent of ‘cpython’, I do the following to get my separate-directory-LoDs:

$ hg clone -u 2.6 cpython py26
$ hg clone -u 2.7 cpython py27
# repeat and rinse for all other active LoDs

That’s one way of doing it.

(Aside: that cpython directory still bugs me. It doesn’t naturally reflect a LoD, so why do I have to stare at it? Yes, I can make it play double duty by naming it "3.3" or whatever and updating it to the 3.3 LoD, but that feels artificial.)

It’s a clone repository of CPython; the name makes perfect sense. You are focusing on what the repository happens to be updated to ATM, not the fact that the repository itself represents any and all LoDs.

Now I want to synchronize my local py27 directory with the state of that LoD on python.org. This is a two step process:

$ cd py27 # now I want to synchronize
$ (cd ../cpython && hg pull)
$ hg pull -u

Editing hgrc to point to hg.python.org means the sync-to-remote-master operation is one command.

I could do this:

$ cd py27 #: now I want to synchronize
$ hg pull -u ssh://hg@hg.python.org/cpython

but I'm not going to remember that url every time. It wouldn't be so bad if
if
Mercurial remembered the pull URL for me, as (you guessed it :) Bazaar
does.

So does Hg: simply edit your .hgrc to have it both pull and push to the same URL.

Clone the repository at a specific branch  Or you can save yourself some trouble, and simply clone the repository at a specific branch:

hg clone ssh://hg@hg.python.org/cpython#2.7

I believe that might even strip out other history outside the scope of the branch.

>> > > Anecdote: I migrated from Subversion to Mercurial a few years ago, and
>> > > found Mercurial branches very intuitive. When I saw mentions of Bazaar
>> > > branches I was driven nuts by (what I saw as) the conflation between a
>> > > branch and a clone. Later I understood how it made sense from the
>> > > perspective of Bazaar, so I shifted my judgment from "insanely
>> > > confusing" to "a particular choice that I don't approve" <wink>.
>> >
>> > That's funny because to my eyes, Mercurial conflates "branches" and
>> > "clones".
>> > Why a clone operation should give me the history for all
>> > lines-of-development
>> > inside a working directory for one "arbitrary" one strikes me as bizarre
>> > (pardon the pun :).

Because Hg views a clone as that: a clone of the entire repository.

A branch just happens to be a part of the repository.

And because it is the entire repository it has to have you point somewhere, so it goes with default since a lot of people never even work somewhere other than on default.

I get that for folks who like the "svn switch" method of
working on multiple LoDs, this probably makes a lot of sense. I don't
personally like or trust that approach much though.

Neither do I in an svn context and why Hg does let you check out just a branch and have all pulls and pushes only go to that branch.
What I’m saying is that a lot of Bazaar terminology using “branch” does not map as-is to Mercurial. We clone a repo, we pull from a repo/clone, we have named branches (e.g. 3.2) in a repo, we have unnamed branches on one named branch. I think you know that already, since you went from using Bazaar terms applied to Mercurial to mixing terms from both (“clone a new branch working directory” → “clone a repo”, probably). I salute your willingness to learn Mercurial, considering how fluent (and fluffy) you appear to be with Bazaar.

This is an inevitable problem with trying to converse fluently about three major dVCSs and at least one or two other non-dVCSs. They all use the same words to mean vaguely similar things, but quickly get bogged down in the implementation details assigned to those words. It all makes perfect sense when you’ve been immersed in those technologies, but it makes discussions and comparisons exceedingly difficult. I’ve no doubt it’s doubly painful to many people who have no prior experience with a dVCS.

(Aside: Bazaar could have potentially eased these folks transition because you can use Bazaar just like you would Subversion -- as a centralized VCS -- without stopping others from using it with full dVCS features on the same code base. I don’t know if Mercurial offers the same flexibility.)

It’s a little like trying to teach Python to a Java programmer. “Object”, “Class”, “Instance”, “Import” all mean roughly similar things, which lulls you into a false sense of understanding. We learn by holding up the new to the light of the familiar and looking for interference patterns. :)

Yes, fun isn’t it? Makes me that much more glad I don’t have to care about other DVCSs anymore; make the decision of which one to go through was painful partially for this reason.

That pull does not update is a feature: The operation between a remote repo and the local repo (the .hg directory) is separate from the operation from the local repo to the working directory. When you know that you want pull + update (like when you have a clean working directory), you use “hg pull -u”. (I don’t like the fetch command.)

Sure, I get that. Again, this feature appears odd because I have the full history of all LoDs embedded in a working directory of a single LoD.

Not single. _current_. I know you don’t like the whole svn switch approach that this is like, but Hg is very much about “don’t forget a thing”, which is why you need to view a clone as a clone repository that you are choosing to view in a certain way at any moment in time instead of as a single branch that just happens to be carrying around the weight of other branches. Totally different approaches to VCS.
As I said above, use the `<branch>` format and you skip this issue (I think).

**Hg mq extension**

>> You speak to my heart, sir. In your ~/.hgrc, under the section [ui],
>> set "editor = path/to/mercurial/source/hgeditor" and enjoy your diffs.
>> I use it and love it.
>>
>> Great, I’ll try that, thanks. One thing Mercurial and Bazaar definitely
>> share
>> is a wealth of magical awesomeness hidden in manpages, wiki pages, mailing
>> list posts, people’s heads, configuration files, and source code. :)
>>
>> If you want to commit a subset of your local changes, I use
>> http://mercurial.selenic.com/wiki/CrecordExtension , a curses-based diff
>> selection UI that works like a charm.
>>
>> I very rarely want to do that. It’s more common (but still, IME not *that*
>> common) to commit the changes to just a few files at a time. One thing
>> Bazaar
>> has that’s very nice is the ability to "shelve" some changes for a time.
>> Let’s say I’m working on a bug and I want to merge your changes in or sync
>> to
>> the master. I can shelve some or all of my uncommitted changes, which
>> saves
>> them essentially out-of-the-way patch files, and then reverts my
>> uncommitted
>> changes. Unshelving then is the process of re-applying those patch files,
>> and
>> yes, resolving conflicts.
>>

**Hg’s is the mq (Mercurial Queue) extension.**

>> This is also a great way to work when you want to do
test-driven-development
but need to do some exploration first. You can hack around non-TDD until
you’re confident of your approach, shelve all your changes, then
> > incrementally
> > apply them back as you write each test. I’m sure Mercurial has something
> > equally awesome lurking about. :)  

They all have the same history from the Linux kernel for the patch queue concept. I suspect it’s pretty universally implemented, just with different command names (of course as gods forbid it be consistent).

Response of Barry Warsaw to Eric Araujo

From: Barry Warsaw <barry@python.org>
To: = Araujo <merwok@netwok.org>
Mime-Version: 1.0
Cc: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

> > You speak to my heart, sir. In your ~/.hgrc, under the section [ui],
> > set “editor = path/to/mercurial/source/hgeditor” and enjoy your diffs.
> > I use it and love it.

Except it doesn’t quite work the way I want it to (hg 1.6.3). It opens your editor with two files, one is the commit message and the other is the diff. (The script itself is a bit buggy too. ;)

But it’s a good clue, and I’ve modified the default hgeditor script to get closer, and fix the bug I noticed. I basically append the diff to the temporary log message file. It’s still not right though because if the diff lines aren’t prepended with ‘HG:’, they end up in the commit message. Arg.

Oh well, I can clearly hack a more complicated script together. It’s such a blindingly obvious improvement, it’s too bad ‘hg commit’ doesn’t DTRT by default.

-Barry

Response of Barry Warsaw to David Murray

Date: Sat, 26 Feb 2011 16:06:45 -0500
From: Barry Warsaw <barry@python.org>
To: "R. David Murray" <rdmurray@bitdance.com>
Cc: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

> > On Sat, 26 Feb 2011 13:08:47 -0500, Barry Warsaw <barry@python.org> wrote:
>> $ cd py27 # now I want to synchronize
>> $ hg pull -u ssh://hg@hg.python.org/cpython
>>
>> but I’m not going to remember that url every time. It wouldn’t be so bad if
>> Mercurial remembered the pull URL for me, as (you guessed it :) Bazaar does.
>>
>> How does setting it in the hgrc differ from "remebering" it?

It’s different because you don’t use a familiar interface to set it (i.e hg). You have to know to hack a file to make it work.

That’s not awesome user interface. ;)

Chapter 2. Development
It’s easy to tell what it remembers because it’s exactly what you told it to remember :). But I guess you’re talking about push and pull automatically remembering the location when none was previously set. I love that feature.

And of course, bzr ‘remembers’ by setting a value in a config file, which of course you could hack if you wanted to. It’s just that you don’t normally have to open your editor and remember which value in which config file you have to manually modify to set the push and pull locations. I think that’s a win, but YMMV. :)

Oh, and ‘bzr info’ always tells you what the push and pull locations are.

I would feel better about Mercurial’s if the repo where not intimately tied with a default working tree (yes, I know -U). In a sense, that’s what Bazaar’s shared repositories are: a place where all your history goes. In Bazaar’s model though, it’s not tied to a specific working tree, and it’s hidden in a dot-directory.

It’s still kind of beside the point - this is the way Mercurial works, and I don’t really mean this thread to be an in-depth comparison between the two.

-Barry

Response of Antoine Pitrou to Barry Warsaw

Mercurial extensions  See Also:

http://mercurial.selenic.com/wiki/ShareExtension

> I find bazaar’s model confusing, and hg’s intuitive, just like Åhric.
> And consider that I learned bazaar before mercurial. To me, it makes
> perfect sense that in a DVCS the "unit" is a directory containing
> a repository and a working copy, and that the repository is *the*
> repository. That is, it has everything related to the project in it,
> just like the master SVN repository does (plus, since it is a DVCS,
> whatever I’ve committed locally but not pushed to the master). To have
> a repository that only has some of the stuff in it is, IMO, confusing.
> I advocated for having all the Python history in one repo partly for
> that reason.
> I would feel better about Mercurial’s if the repo where not intimately tied with a default working tree (yes, I know -U). In a sense, that’s what Bazaar’s shared repositories are: a place where all your history goes. In Bazaar’s model though, it’s not tied to a specific working tree, and it’s hidden in a dot-directory.
Often (but not always), when you’re wanting to do something, there’s an extension for Mercurial which can be enabled: \(^{1}\) http://mercurial.selenic.com/wiki/ShareExtension

Regards
Antoine.

Response of Barry Warsaw to Brett Cannon

> > For other people’s benefit, LoD == line of development (I think).

Yes. It’s just a word that isn’t intimately tied to the implementation details of a specific dVCS.

>> >> I clone the remote repository using the command in the devguide, so I now
>> >> have
>> >> a ‘cpython’ directory containing the history of all LoDs, but with a
>> >> working
>> >> directory that reflects the ‘default’ LoD. Now, in the parent of
>> >> ‘cpython’, I
>> >> do the following to get my separate-directory-LoDs:
>> >>
>> >> $ hg clone -u 2.6 cpython py26
>> >> $ hg clone -u 2.7 cpython py27
>> >> # repeat and rinse for all other active LoDs
>> >>
>> >> >That’s one way of doing it.

I’m sure there are many different ways of setting things up. I am totally in favor of the devguide documenting and encouraging one particular way, and I’m sure Mercurial will prove to be a flexible tool that can conform to user’s preferences rather than the other way ‘round.

>> >> (Aside: that cpython directory still bugs me. It doesn’t naturally reflect
>> >> a
>> >> LoD, so why do I have to stare at it? Yes, I can make it play double duty
>> >> by
>> >> naming it "3.3" or whatever and updating it to the 3.3 LoD, but that feels
>> >> artificial.)
>> >>
>> >> >It’s a clone repository of CPython; the name makes perfect sense. You are
>> >focusing on what the repository happens to be updated to ATM, not the fact
>> >that the repository itself represents any and all LoDs.

No, I get all that. I’m just not sure why I should care where all the history is stored. I’m not actually going to do any coding in the cpython directory, so it just seems like a wart I have to carry around. But as I said before, this is the Mercurial Way, so be it.

>> >> Now I want to synchronize my local py27 directory with the state of that
>> >> LoD
>> >> on python.org. This is a two step process:
>> >>
>> >> $ cd py27 # now I want to synchronize
>> >> $ (cd ../cpython && hg pull)
>> >> $ hg pull -u
Editing hgrc to point to hg.python.org means the sync-to-remote-master operation is one command.

I could do this:

```bash
$ cd py27 # now I want to synchronize
$ hg pull -u ssh://hg@hg.python.org/cpython
```

but I’m not going to remember that url every time. It wouldn’t be so bad if Mercurial remembered the pull URL for me, as (you guessed it :) Bazaar does.

So does Hg: simply edit your .hgrc to have it both pull and push to the same URL.

Right, see my follow up to RDM.

Or you can save yourself some trouble, and simply clone the repository at a specific branch:

```bash
hg clone ssh://hg@hg.python.org/cpython#2.7
```

That might be a better way if it doesn’t slurp down the entire repository history.

Anecdote: I migrated from Subversion to Mercurial a few years ago, and found Mercurial branches very intuitive. When I saw mentions of Bazaar branches I was driven nuts by (what I saw as) the conflation between a branch and a clone. Later I understood how it made sense from the perspective of Bazaar, so I shifted my judgment from “insanely confusing” to “a particular choice that I don’t approve” <wink>.

That’s funny because to my eyes, Mercurial conflates "branches" and "clones".

Why a clone operation should give me the history for all lines-of-development inside a working directory for one "arbitrary" one strikes me as bizarre. (pardon the pun :).

Because Hg views a clone as that: a clone of the entire repository. A branch just happens to be a part of the repository. And because it is the entire repository it has to have you point somewhere, so it goes with default since a lot of people never even work somewhere other than on default.

Yep, I get all that. It’s Mercurial’s model of the world.

Yes, fun isn’t it? Makes me that much more glad I don’t have to care about other DVCSs anymore; make the decision of which one to go through was painful partially for this reason.

Lucky you! I interact with tons of projects, so I still have to deal with everything from CVS to git. This will be my first serious foray into hg, and for that I’m glad.
And really, _any_ dVCS is better than a non-dVCS, so I’m really happy this is finally happening, despite any initial grumbling you’re reading into my posts. :) 

>>> >> >> I’ll have to remember that ‘hg pull’ does not update the working copy by default,
>>> >> >>
>>> >> > That pull does not update is a feature: The operation between a remote
>>> >> > repo and the local repo (the .hg directory) is separate from the
>>> >> > operation from the local repo to the working directory. When you know
>>> >> > that you want pull + update (like when you have a clean working
>>> >> > directory), you use "hg pull -u". (I don’t like the fetch command.)
>>> >>
>>> >> > Sure, I get that. Again, this feature appears odd because I have the full
>>> >> > history of all LoDs embedded in a working directory of a single LoD.
>>> >>
>>> >> > Not single, _current_. I know you don’t like the whole svn switch approach
>>> >> > that this is like, but Hg is very much about "don’t forget a thing", which
>>> >> > is why you need to view a clone as a clone repository that you are choosing
>>> >> > to view in a certain way at any moment in time instead of as a single branch
>>> >> > that just happens to be carrying around the weight of other branches.
>>> >> > Totally different approaches to VCS.

No really, I do get all that! I just don’t like it much. Maybe it’ll grow on me though.

**Mq, shelve, loom, pipeline extensions**

>> >> I very rarely want to do that. It’s more common (but still, IME not *that* common) to commit the changes to just a few files at a time. One thing
>> >> Bazaar
>> >> has that’s very nice is the ability to "shelve" some changes for a time.
>> >> Let’s say I’m working on a bug and I want to merge your changes in or sync to
>> >> the master. I can shelve some or all of my uncommitted changes, which
>> >> saves
>> >> them essentially out-of-the-way patch files, and then reverts my
>> >> uncommitted
>> >> changes. Unshelving then is the process of re-applying those patch files,
>> >> and
>> >> yes, resolving conflicts.
>> >>
>> >>
>> > Hg’s is the mq (Mercurial Queue) extension.

I think mq is more like quilt than shelve. The moral equivalents in Bazaar would probably be the loom and pipeline extensions.

**hg-git, bzr-hg**

>> >> This is also a great way to work when you want to do
>> >> test-driven-development
>> >> but need to do some exploration first. You can hack around non-TDD until
>> >> you’re confident of your approach, shelve all your changes, then
>> >> incrementally
>> >> apply them back as you write each test. I’m sure Mercurial has something
>> >> equally awesome lurking about. :)
> They all have the same history from the Linux kernel for the patch queue
> concept. I suspect it’s pretty universally implemented, just with different
> command names (of course as gods forbid it be consistent).

Truth to that.

I’ve often advocated for the big three to converge on repository format and wire protocol, and for them to innovate and differentiate on UI. The models might be different enough that you couldn’t do it 100%, but the existence of mapping extensions (e.g. hg-git, bzr-hg) seems to imply that they’re pretty darn close.

If we had this, then all the hand wringing about which dVCS to choose would be essentially moot. You’d just pick the cli and gui clients you preferred.

Really, sweating over the dVCS tool detracts from what you do care about, which is developing Python!

-Barry

Response from Dj Gilcrease to Brett Cannon (shelve extension)

From: Dj Gilcrease <digitalxero@gmail.com>
Date: Sat, 26 Feb 2011 16:48:05 -0500
To: Brett Cannon <brett@python.org>
Cc: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

Hg shelve extension

On Sat, Feb 26, 2011 at 3:09 PM, Brett Cannon <brett@python.org> wrote:
> Hg’s is the mq (Mercurial Queue) extension.

I prefer the hg shelve plugin (http://mercurial.selenic.com/wiki/ShelveExtension) for this, more intuitive to me.

Response of Adrian Buehlmann to Barry Warsaw

Date: Sat, 26 Feb 2011 23:45:24 +0100
From: Adrian Buehlmann <adrian@cadifra.com>

> On Feb 26, 2011, at 02:05 PM, R. David Murray wrote:
> 
> >> On Sat, 26 Feb 2011 13:08:47 -0500, Barry Warsaw <barry@python.org> wrote:
> >>> $ cd py27 # now I want to synchronize
> >>> $ hg pull -u ssh://hg@hg.python.org/cpython
> >>>
> >>> but I’m not going to remember that url every time. It wouldn’t be so bad if
> >>> Mercurial remembered the pull URL for me, as (you guessed it :) Bazaar does.
> >>>
> >> How does setting it in the hgrc differ from "remebering" it?
> >
> >> It’s different because you don’t use a familiar interface to set it (i.e hg).
> >> You have to know to hack a file to make it work. That’s not awesome user
> >> interface. ;)

2.17. Languages
.hg/hgrc files  You’d have to take this up with Mercurial’s BDFL Matt. He is a strong advocate for teaching users to learn edit their .hg/hgrc files.

And he’s quite firm on not wanting to have Mercurial touch .hg/hgrc – with the single exception being to write a initial .hg/hgrc on ‘hg clone’, containing the default path with the location from where the repo was cloned.

Regarding Bazaar: FWIW, I periodically retried the speed of ‘bzr check’ - and always gave up again looking at bzr due to the horrible slowness of that command. If I have to use a DVCS I want to be able to check the integrity of my clones in reasonable time. I do it with a cron job on our internal server here and I expect it to have finished checking all our repos when I get to my desk in the morning and look into my email inbox, reading the daily email with the result of the verify runs.

After all, we do have everything secured with hashes, so we can use them, don’t we?

hg paths

>> >> I’ve never been comfortable with the bzr --remember option because I’m never >> >> sure what it is remembering. Much easier for me to see it in a config file. >> >> But, then, that’s how my brain works, and other people’s will work >> >> differently. >> >> It’s easy to tell what it remembers because it’s exactly what you told it to >> remember ;). But I guess you’re talking about push and pull automatically >> remembering the location when none was previously set. I love that feature. >> >> And of course, bzr ‘remembers’ by setting a value in a config file, which of >> course you *could* hack if you wanted to. It’s just that you don’t normally >> have to open your editor and remember which value in which config file you >> have to manually modify to set the push and pull locations. I think that’s a >> win, but YMMV. :) >> >> Oh, and ‘bzr info’ always tells you what the push and pull locations are.

You can use ‘hg paths’ for that:

See [http://selenic.com/repo/hg/help/paths](http://selenic.com/repo/hg/help/paths) or ‘hg help paths’ on the command line

>> >> I find bazaar’s model confusing, and hg’s intuitive, just like Åkric. >> >> And consider that I learned bazaar before mercurial. To me, it makes >> >> perfect sense that in a DVCS the "unit" is a directory containing >> >> a repository and a working copy, and that the repository is *the* >> >> repository. That is, it has everything related to the project in it, >> >> just like the master SVN repository does (plus, since it is a DVCS, >> >> whatever I’ve committed locally but not pushed to the master). To have >> >> a repository that only has some of the stuff in it is, IMO, confusing. >> >> I advocated for having all the Python history in one repo partly for >> >> that reason. >> >> I would feel better about Mercurial’s if the repo where not intimately tied >> with a default working tree (yes, I know -U). In a sense, that’s what >> Bazaar’s shared repositories are: a place where all your history goes. In >> Bazaar’s model though, it’s not tied to a specific working tree, and it’s >> hidden in a dot-directory. >> >> It’s still kind of beside the point - this is the way Mercurial works, and I >> don’t really mean this thread to be an in-depth comparison between the two.

I’m quite surprised indeed to read that much about Bazaar in this thread here :)

Chapter 2. Development
python dev with mercurial announce, 27 february 2011

Response of Daniel Stutzbach to Brett Cannon

hg-git   See Also:

http://wiki.python.org/moin/Git

Date: Sat, 26 Feb 2011 15:08:19 -0800
From: Daniel Stutzbach <stutzbach@google.com>
To: Brett Cannon <brett@python.org>
Cc: python-dev@python.org
Subject: Re: [Python-Dev] Mercurial conversion repositories

> > There is hg-git, but that is hg on top of git.
> >

Actually, hg-git is bidirectional. The hg-git documentation is written from the perspective of an hg client talking to a git server, but for a DVCS “client” and “server” are a matter of perspective.

I spent some time on Friday setting up hg-git on my workstation and making a few test commits. It took me awhile to figure out how to get everything working, but it seems to work smoothly now.

At some point I’ll update http://wiki.python.org/moin/Git with instructions.

hgdevguide, http://potrou.net/hgdevguide/

From: Dj Gilcrease <digitalxero@gmail.com>
Date: Sat, 26 Feb 2011 18:13:15 -0500

So reading the thread about the conversion and the dev guide at http://potrou.net/hgdevguide/ there seems to not be a list of recommended extensions that the python devs should have and use, only a few examples of their use. so I figured I would build up a list for other people to add to / comment on

File Format Management

eol

http://mercurial.selenic.com/wiki/EolExtension required

flake8

http://pypi.python.org/pypiflake8/ recommended especially for new committers as it will validate pep8 compliance and check for common errors Maybe update it to do pep7 compliance checks on the c files as well?

Patch Management

mq http://mercurial.selenic.com/wiki/MqExtension This is the one the devguide uses in examples

rebase http://mercurial.selenic.com/wiki/RebaseExtension used with the --collapse option it is very easy to build up a patch patch with incremental commits, but discard the private history of the developer This one makes more sense to me personally than mq and fits how my standard workflow goes better

shelve http://mercurial.selenic.com/wiki/ShelveExtension Store un commited changes away so they dont affect generation of the patch

2.17. Languages
Development tools, Release 2012.06.18

- **transplant** [http://mercurial.selenic.com/wiki/TransplantExtension](http://mercurial.selenic.com/wiki/TransplantExtension) required to port patches between major versions

- **record** [http://mercurial.selenic.com/wiki/RecordExtension](http://mercurial.selenic.com/wiki/RecordExtension) Allows cherry picking of commits to build a patch. Also works with mq

- **Crecord** [http://mercurial.selenic.com/wiki/CrecordExtension](http://mercurial.selenic.com/wiki/CrecordExtension) appears to be a c optimized version of record

### Branch Management

- **bookmarks** [http://mercurial.selenic.com/wiki/BookmarksExtension](http://mercurial.selenic.com/wiki/BookmarksExtension) Great for tracking bug fix work without needing to create a separate working directory recommended that the central repo NOT have the extension enabled so as to ensure bookmarks are a local only tracking system

**Response of Daniel Stuzbach to Brett Cannon**

Date: Sun, 27 Feb 2011 00:41:05 +0100
From: Adrian Buehlmann <adrian@cadifra.com>

> > Branch Management
> > bookmarks
> > http://mercurial.selenic.com/wiki/BookmarksExtension

Bookmarks will be in Mercurial core for Mercurial 1.8, which will be released in a few days (March 1st). So, with 1.8 it’s no longer needed to enable this extension in the configuration – the feature will be built-in.

**Response of Antoine Pitrou to Dj Gilcrease**

From: Antoine Pitrou <solipsis@pitrou.net>
Date: Sun, 27 Feb 2011 01:59:48 +0100

- **eol**

  > > File Format Management
  > > eol
  > > http://mercurial.selenic.com/wiki/EolExtension
  > > required

Actually, it isn’t *required* on each developer’s setup, since we now have a hook that refuses bogus change groups (if needed, we can even refuse individual changesets). In most situations, even without the eol extension line endings won’t get modified anyway.

- **flake8**

  > > flake8
  > > http://pypi.python.org/pypi/flake8/
  > > recommended especially for new committers as it will validate pep8 compliance and check for common errors

IMHO, nothing replaces human reviews and communication for style and other likewise issues.
mq, rebase, shelve

All these depend on each developer’s taste, as long as only collapsed patches get submitted and committed.

transplant

Not really required, and actually controversial since it commits automatically (we would like people to commit and test before committing, otherwise buildbots can get bogus changesets and spurious failures).

bookmarks

Actually quite poor for tracking bug fix work (see my other messages in this thread :-)).

Regards
Antoine.

Response from Adrian Buehman to Barry Warsaw

Date: Sun, 27 Feb 2011 02:15:51 +0100
From: Adrian Buehlmann <adrian@cadifra.com>

> On Feb 26, 2011, at 11:45 PM, Adrian Buehlmann wrote:
> >
> >> You’d have to take this up with Mercurial’s BDFL Matt. He is a strong
> >> advocate for teaching users to learn edit their .hg/hgrc files.
> >
> >> Well, I guess it’s doubtful I’d change his mind then. :)

Yep.

hg verify

>> Regarding Bazaar: FWIW, I periodically retried the speed of ’bzr check’
>> - and always gave up again looking at bzr due to the horrible slowness
>> of that command. If I have to use a DVCS I want to be able to check the
>> integrity of my clones in reasonable time. I do it with a cron job on
>> our internal server here and I expect it to have finished checking all
Development tools, Release 2012.06.18

>> >> our repos when I get to my desk in the morning and look into my email
>> >> inbox, reading the daily email with the result of the verify runs.
>> >>
>> >> After all, we do have everything secured with hashes, so we can use
>> >> them, don’t we?
>> >
>> > Do you know how thorough ’bzr check’ is? I don’t, but then I’ve never used it
>> > or felt the need to. ;)

That’s quite amazing. If I talk with people about that, it often turns out that they don’t check the integrity of their repos.

Well, hg verify is through and fast enough. That’s good enough for me.

And being slow is not sufficient to earn my trust.

FWIW, be aware that Mercurial does not do integrity checks on normal operations, so chances are you will be able to use a repo that fails verify for quite a while – without even noticing it.

For example you can remove some file X inside .hg/store/data and continue to add history to that repo without any sign of errors, as long as the file X isn’t used during the operations you do.

Response from Adrian Buehlman to Greg Ewing

Date: Sun, 27 Feb 2011 11:42:21 +0100
From: Adrian Buehlmann <adrian@cadifra.com>

hg branch –force, graphlogextension

> > > From: Antoine Pitrou
> > > - a "branch" usually means a "named branch": a set of changesets
> > > bearing the same label (e.g. "default"); that label is freely chosen
> > > by the committer at any point, and enforces no topological
> > > characteristic
> > >
> > > There are *some* topological restrictions, because hg won’t
> > > let you assign a branch name that’s been used before to a node
> > > unless one of its parents has that name. So you can’t create
> > > two disconnected subgraphs whose nodes have the same branch
> > > name.

That’s not completely correct. You can do that.

Mercurial by default assumes you’re probably in error if you are trying to create such disconnected branch name subgraphs, but you can convince it that it’s really what you want by doing:

hg branch --force <existing branch name>

Example (glog command requires the graphlog extension enabled [1]):

$ hg init a
$ cd a
$ echo foo > bla
$ hg ci -Am1
adding bla
$ hg branch bl
marked working directory as branch b1
$ hg ci -m2
$ hg branch default
abort: a branch of the same name already exists (use ‘hg update’ to switch to it)
$ hg branch --force default
marked working directory as branch default
$ hg ci -m3
created new head
$ hg glog --template "{rev}, {branch}\n"
0 2, default
| o 1, b1
| o 0, default


Reponse of Antoine Pitrou to Martin von Lowis, changeset, tip

Date: Sun, 27 Feb 2011 16:08:21 +0100
From: Antoine Pitrou <solipsis@pitrou.net>

>>> >>> changeset: 72694:e65daae6cf44
>>> >>> user: Antoine Pitrou <solipsis@pitrou.net>
>>> >>> date: Mon Feb 21 21:30:55 2011 +0000
>>> >>> summary: Try s/UINT_MAX/INT_MAX/
>>> >>>
>>> It’s not on an unnamed branch, it’s on the "default" branch (which is
>>> omitted for concision by "hg log" and other commands with a similar
>>> output).
>>>>
>>> It’s probably a terminology issue, but: the changeset can’t be "on"
>>> the default "branch", because the head of the default branch (called
>>> "tip", IIUC) isn’t a descendant of that changeset.

Well, a branch (or named branch) in hg terminology can have several heads (see the other email about heads and branches).

Also, just so you know, “tip” is simply the latest (in pull or commit order) changeset in the local repository. It can be in any branch (for example, if you pull of bunch of changesets someone made in “3.2”, then your tip will be in branch “3.2”).

I hope it all starts to make sense ;)

Regards
Antoine.

Reponse of Nick Coghlan to Atoine Pitrou, eol extension

Date: Mon, 28 Feb 2011 01:23:29 +1000
From: Nick Coghlan <ncoghlan@gmail.com>
To: Antoine Pitrou <solipsis@pitrou.net>
Cc: python-dev@python.org
Subject: Re: [Python-Dev] hg extensions was Mercurial conversion repositories
On Sun, 27 Feb 2011 07:46:51 +0100
"Martin v. Löwis" <martin@v.loewis.de> wrote:

Actually, it isn't *required* on each developer's setup, since we
now have a hook that refuses bogus changegroups (if needed, we can even
refuse individual changesets). In most situations, even without the
eol extension line endings won't get modified anyway.

I think this is overly optimistic. Visual Studio will break all your
files if you don't use that extension (and you actually use it to
modify source code).

My assumption was that most developers don’t use MSVC, so most of them
do*n’t* risk breaking eols ;)

True, for Windows devs it might be necessary to promote it.

Windows devs were the original target audience, yes :)

Cheers, Nick.

---

**Announce of tortoise Hg 2.0 by Adrian Buehlman**

Date: Sun, 27 Feb 2011 17:01:37 +0100
From: Adrian Buehlmann <adrian@cadifra.com>

> On 2/27/2011 10:18 AM, Antoine Pitrou wrote:
> Well, chances are TortoiseHG comes with an UI to apply patches
> (TortoiseSVN had one), so the command-line instructions may be of
> little use to them.
> I don’t believe TortoiseHG has such a feature (or I can’t find it),
> although if you have TortoiseSVN, you can still use that as a patch tool.

TortoiseHg can import patches just fine.

FWIW, we are very close to releasing TortoiseHg 2.0 (due March 1st), which ported the current Gtk based TortoiseHg
to Qt (although, it was more like a rewrite :-). For the old Gtk TortoiseHg, see the online docs here:

http://tortoisehg.bitbucket.org/manual/1.1/patches.html#import-patches

Homepage for the Qt port  https://bitbucket.org/tortoisehg/thg/wiki/Home

For people on Windows, we have beta installers for the new Qt based TortoiseHg at:

https://bitbucket.org/tortoisehg/thg/downloads

Feedback is welcome on:

thg-dev@googlegroups.com or
tortoisehg-discuss@lists.sourceforge.net

(we moved the development list to google groups)
Annoucne of George Brandl for the hgeditor extension

To: python-dev@python.org
From: Georg Brandl <g.brandl@gmx.net>
Date: Sun, 27 Feb 2011 17:38:26 +0100

Martin v. Lowis, eol

>> > On Feb 26, 2011, at 01:49 AM, Éric Araujo wrote:
>> >
>> >> You speak to my heart, sir. In your ~/.hgrc, under the section [ui],
>> >> set "editor = path/to/mercurial/source/hgeditor" and enjoy your diffs.
>> >> I use it and love it.
>> >
>> > Except it doesn’t quite work the way I want it to (hg 1.6.3). It opens your
>> > editor with two files, one is the commit message and the other is the diff.
>> > (The script itself is a bit buggy too. ;)
>> >
>> > But it’s a good clue, and I’ve modified the default hgeditor script to get
>> > closer, and fix the bug I noticed. I basically append the diff to the
>> > temporary log message file. It’s still not right though because if the diff
>> > lines aren’t prepended with ‘HG:’, they end up in the commit message. Arg.
>> >
>> > Oh well, I can clearly hack a more complicated script together. It’s such a
>> > blindingly obvious improvement, it’s too bad ‘hg commit’ doesn’t DTRT by
>> > default.

While I understand the usefulness of the diff feature, it is not useful to everyone, e.g. those using almost exclusively
commit -m message.

Of course it would be nice if hg made it easier (a hgrc option, for example) to do this.

BTW, I had not heard of hgeditor before, and wrote a small hg extension to do what you want (with HG: prefix :) before I saw that others had already replied with hgeditor. The extension had 10 lines of code.

Georg

Cc: Antoine Pitrou <solipsis@pitrou.net>, python-dev@python.org
Subject: Re: [Python-Dev] hg extensions was Mercurial conversion repositories

>> I think this is overly optimistic. Visual Studio will break all your
>> files if you don’t use that extension (and you actually use it to
>> modify source code).
>>
>> My assumption was that most developers don’t use MSVC, so most of them
>> don’t risk breaking eols ;)
>> True, for Windows devs it might be necessary to promote it.

If I change code on Windows, I always use MSVC to edit it. It’s best integrated with the build process and the
dbugger. If I change Python code on Windows, I use vim or IDLE.

Different MSVC releases took different approaches wrt. LF-separated files. For a long time, new lines added would
be CRLF, whereas existing line endings would remain unchanged, resulting in a mix of line endings. It appears that
VS 2008 now uniformly converts the entire file to CRLF on first edit.

Regards, Martin
Python c implementation

python C code coverage  See Also:
http://coverage.livinglogic.de/

cpython versions  See Also:

• http://docs.python.org/dev/whatsnew/index.html

cpython 2.7  See Also:
http://www.python.org/download/releases/2.7.2/

cpython 3.3  See Also:

• http://www.python.org/download/releases/3.3.0/
• http://docs.python.org/3.3/whatsnew/3.3.html

Contents

• cpython 3.3
  – What’s new

What’s new  Python 3.3 includes a range of improvements of the 3.x series, as well as easier porting between 2.x and 3.x. Major new features and changes in the 3.3 release series are:

• PEP 380, syntax for delegating to a subgenerator (“yield from”)
• PEP 393, flexible string representation (doing away with the distinction between “wide” and “narrow” Unicode builds)
• A C implementation of the “decimal” module, with up to 80x speedup for decimal-heavy applications
• The import system (__import__) is based on importlib by default
• The new “packaging” module (also known as distutils2, and released standalone under this name), implementing the new packaging formats and deprecating “distutils”
• The new “lzma” module with LZMA/XZ support
• PEP 405, virtual environment support in core
• PEP 420, namespace package support
• PEP 3151, reworking the OS and IO exception hierarchy
• PEP 3155, qualified name for classes and functions
• PEP 409, suppressing exception context
• PEP 414, explicit Unicode literals to help with porting
• PEP 418, extended platform-independent clocks in the “time” module
• PEP 412, a new key-sharing dictionary implementation that significantly saves memory for object-oriented code
• The new “faulthandler” module that helps diagnosing crashes
• The new “unittest.mock” module
• The new “ipaddress” module
• A “collections.ChainMap” class for linking mappings to a single unit
• Wrappers for many more POSIX functions in the “os” and “signal” modules, as well as other useful functions such as “sendfile()”
• Hash randomization, introduced in earlier bugfix releases, is now switched on by default

cpython 3.2  See Also:
• http://www.python.org/download/releases/3.2.2/
• http://docs.python.org/py3k/

cpython 3.1  See Also:
• http://docs.python.org/dev/whatsnew/3.1.html

cpython 3.0  See Also:
• http://docs.python.org/dev/whatsnew/3.0.html

cpython sur android

Python for android  See Also:
• http://code.google.com/p/python-for-android/

This is Python built to run on Android devices. It is made to be used together with SL4A (Scripting Layer For Android).
Nearly all the actual non-python specific documentation can be found at android-scripting

cython

cython  See Also:
• http://cython.org/
• http://docs.cython.org/src/quickstart/overview.html
Cython is a programming language based on Python, with extra syntax allowing for optional static type declarations. It aims to become a superset of the Python language which gives it high-level, object-oriented, functional, and dynamic programming. The source code gets translated into optimized C/C++ code and compiled as Python extension modules. This allows for both very fast program execution and tight integration with external C libraries, while keeping up the high programmer productivity for which the Python language is well known.

The primary Python execution environment is commonly referred to as CPython, as it is written in C. Other major implementations use Java (Jython Jython), C# (IronPython IronPython) and Python itself (PyPy PyPy). Written in C, CPython has been conducive to wrapping many external libraries that interface through the C language. It has, however, remained non trivial to write the necessary glue code in C, especially for programmers who are more fluent in a high-level language like Python than in a close-to-the-metal language like C.

Originally based on the well-known Pyrex Pyrex, the Cython project has approached this problem by means of a source code compiler that translates Python code to equivalent C code. This code is executed within the CPython runtime environment, but at the speed of compiled C and with the ability to call directly into C libraries. At the same time, it keeps the original interface of the Python source code, which makes it directly usable from Python code. These two-fold characteristics enable Cython’s two major use cases: extending the CPython interpreter with fast binary modules, and interfacing Python code with external C libraries.

While Cython can compile (most) regular Python code, the generated C code usually gains major (and sometime impressive) speed improvements from optional static type declarations for both Python and C types. These allow Cython to assign C semantics to parts of the code, and to translate them into very efficient C code. Type declarations can therefore be used for two purposes: for moving code sections from dynamic Python semantics into static-and-fast C semantics, but also for directly manipulating types defined in external libraries. Cython thus merges the two worlds into a very broadly applicable programming language.

Wrapping C++ Classes in Cython  

This page aims to get you quickly up to speed so you can wrap C++ interfaces with a minimum of pain and ‘surprises’.

In the past, Pyrex (and later, Cython) only supported wrapping of C APIs, and not C++. To wrap C++, one had to write a pure-C shim, containing functions for constructors/destructors and method invocations. Object pointers were passed around as opaque void pointers, and cast to/from object pointers as needed.

This approach did work, but it got awfully messy and error-prone when trying to wrap APIs with large class hierarchies and lots of inheritance. Later versions of Pyrex and Cython simplified this somewhat by allowing textual definitions that would get generated literally into the output sources. This was still error prone but was at least capable of interfacing with a lot of existing code through a somewhat self-contained declaration.

Since then, a lot of work went into improving Cython and starting with version 0.13, it offers heavily extended support for C++ code in ‘real’ language syntax. The approach described in this document will help you wrap a lot of C++
code with only little effort. Due to the vast landscape that C++ considers valid syntax, there are still some limitations, which we will discuss at the end of the document.

python3

Python 3

Porting to python3 See Also:

Porting to python3

pypy

pypy See Also:

• http://pypy.org/
• http://fr.wikipedia.org/wiki/PyPy

PyPy est une mise en œuvre du langage Python écrite elle-même en Python, avec une architecture flexible

pypi speed See Also:

http://speed.pypy.org/

Sources

hg clone https://pvergain@bitbucket.org/pypy/pypy

Architecture pypi See Also:


PyPy is a Python implementation and a dynamic language implementation framework.
This chapter assumes familiarity with some basic interpreter and compiler concepts like bytecode and constant folding.
Little History  Python is a high-level, dynamic programming language. It was invented by the Dutch programmer Guido van Rossum in the late 1980s.

Guido’s original implementation is a traditional bytecode interpreter written in C, and consequently known as CPython.

There are now many other Python implementations.

Among the most notable are Jython, which is written in Java and allows for interfacing with Java code, IronPython, which is written in C# and interfaces with Microsoft’s .NET framework, and PyPy, the subject of this chapter.

CPython is still the most widely used implementation and currently the only one to support Python 3, the next generation of the Python language.

This chapter will explain the design decisions in PyPy that make it different from other Python implementations and indeed from any other dynamic language implementation.

Versions

pypy versions  See Also:

• http://pypy.org/
• http://fr.wikipedia.org/wiki/PyPy

pypy version 1.7

PyPy 1.7 - widening the sweet spot  We’re pleased to announce the 1.7 release of PyPy. As became a habit, this release brings a lot of bugfixes and performance improvements over the 1.6 release. However, unlike the previous releases, the focus has been on widening the “sweet spot” of PyPy. That is, classes of Python code that PyPy can greatly speed up should be vastly improved with this release. You can download the 1.7 release here:

http://pypy.org/download.html

What is PyPy?  PyPy is a very compliant Python interpreter, almost a drop-in replacement for CPython 2.7. It’s fast (pypy 1.7 and cpython 2.7.1 performance comparison) due to its integrated tracing JIT compiler.

This release supports x86 machines running Linux 32/64, Mac OS X 32/64 or Windows 32. Windows 64 work is ongoing, but not yet natively supported.

The main topic of this release is widening the range of code which PyPy can greatly speed up. On average on our benchmark suite, PyPy 1.7 is around 30% faster than PyPy 1.6 and up to 20 times faster on some benchmarks.

Highlights

• Numerous performance improvements. There are too many examples which python constructs now should behave faster to list them.
• Bugfixes and compatibility fixes with CPython.
• Windows fixes.
• PyPy now comes with stackless features enabled by default. However, any loop using stackless features will interrupt the JIT for now, so no real performance improvement for stackless-based programs. Contact pypy-dev for info how to help on removing this restriction.
• NumPy effort in PyPy was renamed numpypy. In order to try using it, simply
write:

```python
import numpypy as numpy
```

at the beginning of your program. There is a huge progress on numpy in PyPy since 1.6, the main feature being implementation of dtypes.

- JSON encoder (but not decoder) has been replaced with a new one. This one is written in pure Python, but is known to outperform CPython’s C extension up to 2 times in some cases. It’s about 20 times faster than the one that we had in 1.6.
- The memory footprint of some of our RPython modules has been drastically improved. This should impact any applications using for example cryptography, like tornado.
- There was some progress in exposing even more CPython C API via cpyext.

**Things that didn’t make it, expect in 1.8 soon** There is an ongoing work, which while didn’t make it to the release, is probably worth mentioning here. This is what you should probably expect in 1.8 some time soon:

- Specialized list implementation. There is a branch that implements lists of integers/floats/strings as compactly as array.array. This should drastically improve performance/memory impact of some applications
- NumPy effort is progressing forward, with multi-dimensional arrays coming soon.
- There are two brand new JIT assembler backends, notably for the PowerPC and ARM processors.

**Fundraising** It’s maybe worth mentioning that we’re running fundraising campaigns for NumPy effort in PyPy and for Python 3 in PyPy. In case you want to see any of those happen faster, we urge you to donate to numpy proposal or py3k proposal. In case you want PyPy to progress, but you trust us with the general direction, you can always donate to the general pot.

**ironpython**

Ironython  See Also:

http://ironpython.codeplex.com/

**Contents**

- Ironython
  - About
  - Sources on github
  - Ironython issue tracker
  - How to install ironpython
  - Ironpython tools for visual studio
    - An integrated environment for developing Python in VS2010
  - Ironpython versions
About  IronPython is an implementation of the Python programming language running under .NET/Mono and Silverlight/Moonlight.

- It supports an interactive console with fully dynamic compilation.
- It’s well integrated with the rest of the .NET Framework and makes all .NET libraries easily available to Python programmers, while maintaining compatibility with the Python language.
- There also is Visual Studio tooling integration.

IronPython is an open source project freely available under the Apache License v2.0. The sources are stored on GitHub as part of the IronLanguages project.

See Also:

- http://ironpython.net/documentation/
- http://ironpython.net/documentation/dotnet/

Sources on github  See Also:


git clone https://github.com/IronLanguages/main.git

Ironpython tools for visual studio  See Also:

http://pytools.codeplex.com/

An integrated environment for developing Python in VS2010

- Supports CPython and IronPython
- Python editor with advanced member and signature intellisense
- Code navigation “Find all refs”, goto definition, and object browser
- Local and remote debugging
- Profiling with multiple views
- Integrated REPL window
- Support for HPC clusters and MPI, including debugging & Profiling
- Interactive parallel computing via integrated IPython REPL
- Free & Open Source (Apache 2.0)

Ironpython versions
Ironpython versions

Ironpython 2.7.2

Ironpython 2.7.2.1

Release Notes  See Also:
http://ironpython.codeplex.com/releases/view/74478

On behalf of the IronPython team, I’m happy to announce the final release IronPython 2.7.2.

This release includes everything from IronPython 2.7 and 2.7.1 as well.

Like all IronPython 2.7-series releases, .NET 4 is required to install it. Installing this release will replace any existing IronPython 2.7-series installation.

Unlike previous releases, the assemblies for all supported platforms are included in the installer as well as the zip package, in the “Platforms” directory.

IronPython 2.7.2 includes support for .NET 3.5, .NET 4, Silverlight 4, and Silverlight 5, and preliminary releases for Mono for Android 4.0 and Windows Phone 7.5.

For 2.7.2, the mobile platforms are previews; they have known issues, and expect them to change for 2.7.3.

The biggest improvements in IronPython 2.7.2 are:

• support for loading libraries from zip archives (the zipimport module);
• the sqlite3 module;
• many improvements to the pyc.py compiler, such as the ability generate standalone executables by embedding the IronPython assemblies; and numerous other bug fixes.

Ironpython 2.7  See Also:
http://ironpython.codeplex.com/releases/view/71089

The first community release of IronPython is now available – IronPython 2.7 Beta 2  At long (long) last, the first community release of IronPython is now available – IronPython 2.7 Beta 2. The highlights of this release are the new zlib (which also enables gzip) and subprocess modules.

There have also been a number of bug fixes.

Since Beta 1, we’ve moved all development to the Github IronLanguages project, although the issue tracker is still on CodePlex. This meant learning the build system, learning how to package a release (something that needs some work), and various other odds ’n ends. There are definitely some stumbling blocks that new people might trip over, so those should be taken care of as soon as possible.

See Also:
• https://github.com/IronLanguages/main

This is the first release candidate of IronPython 2.7.1. Like IronPython 2.7, this release requires .NET 4 or Silverlight 4.

This release will replace any existing IronPython installation.
If there are no showstopping issues, this will be the only release candidate for 2.7.1, so please speak up if you run into any roadblocks.

The highlights of 2.7.1 are:

- Updated the standard library to match CPython 2.7.2.
- Add the ast, csv, and unicodedata modules.
- Fixed several bugs.

See Also:

http://ironpython.codeplex.com/releases/view/54498

On behalf of the IronPython team, I’m very pleased to announce the release of IronPython 2.7. This release contains all of the language features of Python 2.7, as well as several previously missing modules and numerous bug fixes.

IronPython 2.7 also includes built-in Visual Studio support through IronPython Tools for Visual Studio. IronPython 2.7 requires .NET 4.0 or Silverlight 4.


Any bugs should be reported at http://ironpython.codeplex.com/workitem/list/basic.

Python 2.7 includes a number of features backported from the Python 3.0 series. This release implements the new builtin _io module, includes dictionary and set comprehensions, set literals, supports multiple context managers in the with statement, and adds several new functions to the itertools methods, and auto indexing for the new string formatting. There are also numerous updates to the standard library such as ordered dictionaries and the new argparse module.

This release also includes a “IronPython Tools for Visual Studio” option within the IronPython installer. This enables one install to get both IronPython and IronPython Visual Studio support assuming you have an existing installation of Visual Studio 2010. This version of IronPython Tools includes a number of bug fixes as improved WPF designer support. The designer fully supports XAML and WPF including data binding to Python classes dynamically.

To improve interop with modern .NET code such as LINQ, support for extension methods has been added as the clr.ImportExtensions method.

We’ve also updated the IronPython installer to include documentation based upon the CPython documentation. This new .chm file includes documentation on the Python language and standard library. It’s been extended from the normal Python documentation to include IronPython specific topics such as the DLR hosting APIs and extending IronPython from statically typed .NET languages.

We flushed out more support for missing built-in modules which CPython includes. This release includes the mmap and signal modules bringing better support for interoperating with unmanaged code, the zlib and gzip modules for compression, and the subprocess and webbrowser modules for interacting with other programs.

...
Jython programs can seamlessly import and use any Java class. Except for some standard modules, Jython programs use Java classes instead of Python modules. Jython includes almost all of the modules in the standard Python programming language distribution, lacking only some of the modules implemented originally in C.

For example, a user interface in Jython could be written with Swing, AWT or SWT.

### Sources

svn co https://jython.svn.sourceforge.net/svnroot/jython jython

### GUI applications

See Also:

http://www.jython.org/jythonbook/en/1.0/GUIApplications.html

### Working with classpath

### Python on android

See Also:

Python for android

### Python on a Chip

#### Python-on-a-Chip (p14p) See Also:

http://code.google.com/p/python-on-a-chip

Welcome! Python-on-a-Chip (p14p) is a project to develop a reduced Python virtual machine (codenamed PyMite) that runs a significant subset of the Python language on microcontrollers without an OS.

The other parts of p14p are the device drivers, high-level libraries and other tools. Please join the python-on-a-chip google group to discuss this project.

nuitka
Nuitka  See Also:
http://www.nuitka.net/blog/nuitka-a-python-compiler/what-is-nuitka/

Right now Nuitka is a good replacement for the Python interpreter and compiles every construct that CPython 2.6
and 2.7 offer. It translates the Python into a C++ program that then uses “libpython” to execute in the same way as
CPython does, in a very compatible way.

It is somewhat faster than CPython already, but currently it doesn’t make all the optimizations possible, but a 258%
factor on pystone is a good start (number is from version 0.3.11).

python style guides

pep 0008 Style Guide for Python Code  See Also:
   - http://www.python.org/dev/peps/pep-0008/
   - pep 0008 Style Guide for Python Code

python header

```
# -*- coding: utf-8 -*-

"""
Acquisition des valeurs lues par le télemètre.
"""
```

python google style guide  See Also:
   - http://code.google.com/p/google-styleguide/
   - http://google-styleguide.googlecode.com/svn/trunk/pyguide.html

The SoC framework, and Melange web applications built upon it, are implemented in Python (it is one of the program-
ming language besides Java and Go which are currently supported by Google App Engine).

This style guide is a list of dos and don’ts for Python contributions to the Melange project.

The rules below are not guidelines or recommendations, but strict rules.

You may not disregard the rules we list below except as approved on a need-to-use basis.

But note also the advice on consistency at the end of the guide.

python google language rules

pychecker  See Also:
http://pychecker.sourceforge.net/

Run pychecker over your code.

Definition  PyChecker is a tool for finding bugs in Python source code. It finds problems that are typically caught by
a compiler for less dynamic languages like C and C++. It is similar to lint.

Because of the dynamic nature of Python, some warnings may be incorrect; however, spurious warnings should be
fairly infrequent.

Pros  Catches easy-to-miss errors like typos, use-vars-before-assignment, etc.

Cons  pychecker isn’t perfect. To take advantage of it, we’ll need to sometimes:
1. Write around it
2. Suppress its warnings
3. Improve it or
4. Ignore it.

**Decision** Make sure you run pychecker on your code.

For information on how to run pychecker, see the pychecker homepage.

To suppress warnings, you can set a module-level variable named `__pychecker__` to suppress appropriate warnings. For example:

```
__pychecker__ = 'no-callinit no-classattr'
```

Suppressing in this way has the advantage that we can easily search for suppressions and revisit them.

You can get a list of pychecker warnings by doing `pychecker --help`.

Unused argument warnings can be suppressed by using "_" as the identifier for the unused argument or prefixing the argument name with `unused_`.

In situations where changing the argument names is infeasible, you can mention them at the beginning of the function. For example:

```
def foo(a, unused_b, unused_c, d=None, e=None):
    (d, e) = (d, e)  # Silence pychecker
    return a
```

Ideally, pychecker would be extended to ensure that such “unused declarations” were true.

**Imports for packages and modules only** Use imports for packages and modules only.

**Definition** Reusability mechanism for sharing code from one module to another.

**Pros** The namespace management convention is simple. The source of each identifier is indicated in a consistent way; `x.Obj` says that object `Obj` is defined in module `x`.

**Cons** Module names can still collide. Some module names are inconveniently long.

**Decision** Use import `x` for importing packages and modules.

Use from `x` import `y` where `x` is the package prefix and `y` is the module name with no prefix.

Use from `x` import `y` as `z` if two modules named `y` are to be imported or if `y` is an inconveniently long name.

For example the module `sound.effects.echo` may be imported as follows:

```
from sound.effects import echo
...
echo.EchoFilter(input, output, delay=0.7, atten=4)
```

Do not use relative names in imports.

Even if the module is in the same package, use the **full package name**.

This helps prevent unintentionally importing a package twice.
**Packages** Import each module using the full pathname location of the module.

**Pros** Avoids conflicts in module names. Makes it easier to find modules.

**Cons** Makes it harder to deploy code because you have to replicate the package hierarchy.

**Decision** All new code should import each module by its full package name. Imports should be as follows:

```
# Reference in code with complete name.
import sound.effects.echo

# Reference in code with just module name (preferred).
from sound.effects import echo
```

**Exceptions are allowed but must be used carefully**

**Definition** Exceptions are a means of breaking out of the normal flow of control of a code block to handle errors or other exceptional conditions.

**Pros** The control flow of normal operation code is not cluttered by error-handling code. It also allows the control flow to skip multiple frames when a certain condition occurs, e.g., returning from N nested functions in one step instead of having to carry-through error codes.

**Cons** May cause the control flow to be confusing. Easy to miss error cases when making library calls.

**Decision** Exceptions must follow certain conditions:

Raise exceptions like this:

```
raise MyException("Error message") or raise MyException.
```

Do not use the two-argument form (raise MyException, “Error message”) or deprecated string-based exceptions (raise “Error message”).

Modules or packages should define their own domain-specific base exception class, which should inherit from the built-in Exception class.

The base exception for a module should be called Error::

```
class Error(Exception):
    pass
```

Never use catch-all except: statements, or catch Exception or StandardError, unless you are re-raising the exception or in the outermost block in your thread (and printing an error message).

Python is very tolerant in this regard and except: will really catch everything including Python syntax errors. It is easy to hide real bugs using except:.

Minimize the amount of code in a try/except block. The larger the body of the try, the more likely that an exception will be raised by a line of code that you didn’t expect to raise an exception. In those cases, the try/except block hides a real error.

Use the finally clause to execute code whether or not an exception is raised in the try block. This is often useful for cleanup, i.e., closing a file.
Avoid global variables

**Definition** Variables that are declared at the module level.

**Pros** Occasionally useful.

**Cons** Has the potential to change module behavior during the import, because assignments to module-level variables are done when the module is imported.

**Decision** Avoid global variables in favor of class variables.

Some exceptions are:

- Default options for scripts.
- Module-level constants. For example: PI = 3.14159. Constants should be named using all caps with underscores; see *Naming*.
- It is sometimes useful for globals to cache values needed or returned by functions.
- If needed, globals should be made internal to the module and accessed through public module level functions; see *Naming*.

**Nested/Local/Inner Classes and Functions are fine** Nested/local/inner classes and functions are fine.

**Definition** A class can be defined inside of a method, function, or class.

A function can be defined inside a method or function. Nested functions have read-only access to variables defined in enclosing scopes.

**Pros** Allows definition of utility classes and functions that are only used inside of a very limited scope. Very ADT-y.

**Cons** Instances of nested or local classes cannot be pickled.

**Decision** They are fine.

**List Comprehensions: okay to use for simple cases**.

**Definition** List comprehensions and generator expressions provide a concise and efficient way to create lists and iterators without resorting to the use of map(), filter(), or lambda.

**Pros** Simple list comprehensions can be clearer and simpler than other list creation techniques. Generator expressions can be very efficient, since they avoid the creation of a list entirely.

**Cons** Complicated list comprehensions or generator expressions can be hard to read.

**Decision** Okay to use for simple cases.

Each portion must fit on one line: mapping expression, for clause, filter expression.

Multiple for clauses or filter expressions are not permitted.

Use loops instead when things get more complicated.

No:
result = [(x, y) for x in range(10) for y in range(5) if x * y > 10]

return ((x, y, z) for x in xrange(5) for y in xrange(5) if x != y for z in xrange(5) if y != z)

Yes:

result = []
for x in range(10):
    for y in range(5):
        if x * y > 10:
            result.append((x, y))
for x in xrange(5):
    for y in xrange(5):
        if x != y:
            for z in xrange(5):
                if y != z:
                    yield (x, y, z)

return ((x, complicated_transform(x))
    for x in long_generator_function(parameter) if x is not None)
squares = [x * x for x in range(10)]
eat(jelly_bean for jelly_bean in jelly_beans
    if jelly_bean.color == 'black')

Default Iterators and Operators  Use default iterators and operators for types that support them, like lists, dictionaries, and files.

Definition  Container types, like dictionaries and lists, define default iterators and membership test operators ("in" and "not in").

Pros: The default iterators and operators are simple and efficient. They express the operation directly, without extra method calls. A function that uses default operators is generic. It can be used with any type that supports the operation.

Cons  You can’t tell the type of objects by reading the method names (e.g. has_key() means a dictionary). This is also an advantage.

Decision  Use default iterators and operators for types that support them, like lists, dictionaries, and files. The built-in types define iterator methods, too. Prefer these methods to methods that return lists, except that you should not mutate a container while iterating over it.

Yes:

for key in adict: ...
if key not in adict: ...
if obj in alist: ...
for line in afile: ...
for k, v in dict.iteritems(): ...

No:

for key in adict.keys(): ...
if not adict.has_key(key): ...
for line in afile.readlines(): ...

Use generators as needed

Definition A generator function returns an iterator that yields a value each time it executes a yield statement. After it yields a value, the runtime state of the generator function is suspended until the next value is needed.

Pros Simpler code, because the state of local variables and control flow are preserved for each call. A generator uses less memory than a function that creates an entire list of values at once.

Cons None.

Decision Fine. Use “Yields:” rather than “Returns:” in the doc string for generator functions.

Default Argument Values: okay in most cases

Definition You can specify values for variables at the end of a function’s parameter list, e.g., def foo(a, b=0):

If foo is called with only one argument, b is set to 0.

If it is called with two arguments, b has the value of the second argument.

Pros Often you have a function that uses lots of default values, but—rarely—you want to override the defaults. Default argument values provide an easy way to do this, without having to define lots of functions for the rare exceptions. Also, Python does not support overloaded methods/functions and default arguments are an easy way of “faking” the overloading behavior.

Cons Default arguments are evaluated once at module load time. This may cause problems if the argument is a mutable object such as a list or a dictionary. If the function modifies the object (e.g., by appending an item to a list), the default value is modified.

Decision Okay to use with the following caveats:

Do not use mutable objects as default values in the function or method definition.

Yes:

def foo(a, b=None):
    if b is None:
        b = []

No:

def foo(a, b=[]):
    ...

2.17. Languages
Calling code must use named values for arguments with a default value. This helps document the code somewhat and helps prevent and detect interface breakage when more arguments are added.

```python
def foo(a, b=1):
    ...

Yes:
foo(1)
foo(1, b=2)

No:
foo(1, 2)
```

**Properties**  Use properties for accessing or setting data where you would normally have used simple, lightweight accessor or setter methods.

**Definition** A way to wrap method calls for getting and setting an attribute as a standard attribute access when the computation is lightweight.

**Pros** Readability is increased by eliminating explicit get and set method calls for simple attribute access.
- Allows calculations to be lazy.
- Considered the Pythonic way to maintain the interface of a class. In terms of performance, allowing properties bypasses needing trivial accessor methods when a direct variable access is reasonable. This also allows accessor methods to be added in the future without breaking the interface.

**Cons**
- Properties are specified after the getter and setter methods are declared, requiring one to notice they are used for properties farther down in the code (except for readonly properties created with the `@property` decorator).
- Must inherit from object.
- Can hide side-effects much like operator overloading.
- Can be confusing for subclasses.

**Decision** Use properties in new code to access or set data where you would normally have used simple, lightweight accessor or setter methods.

**Read-only properties should be created with the `@property` decorator**

Inheritance with properties can be non-obvious if the property itself is not overridden.

Thus one must make sure that accessor methods are called indirectly to ensure methods overridden in subclasses are called by the property (using the Template Method DP).

Yes:
import math

class Square(object):
    """A square with two properties: a writable area and a read-only perimeter.

    To use:
    >>> sq = Square(3)
    >>> sq.area
    9
    >>> sq.perimeter
    12
    >>> sq.area = 16
    >>> sq.side
    4
    >>> sq.perimeter
    16
    """
    def __init__(self, side):
        self.side = side
    
def __get_area(self):
        """Calculates the 'area' property."""
        return self.side ** 2
    
def __get_area(self):
        """Indirect accessor for 'area' property."""
        return self.__get_area()
    
def __set_area(self, area):
        """Sets the 'area' property."""
        self.side = math.sqrt(area)
    
def __set_area(self, area):
        """Indirect setter for 'area' property.""
        self.__set_area(area)
    
    area = property(__get_area, __set_area,
                    doc="""Gets or sets the area of the square."""")

    @property
    def perimeter(self):
        return self.side * 4

True/False evaluations    Use the “implicit” false if at all possible.

Definition    Python evaluates certain values as false when in a boolean context. A quick “rule of thumb” is that all
    “empty” values are considered false so 0, None, [], {}, """ all evaluate as false in a boolean context.

Pros    Conditions using Python booleans are easier to read and less error-prone. In most cases, they’re also faster.

Cons    May look strange to C/C++ developers.

Decision    Use the “implicit” false if at all possible, e.g., if foo: rather than if foo != {}: There are a few caveats that
    you should keep in mind though:
    • Never use == or != to compare singletons like None. Use is or is not.
• Beware of writing if \( x \): when you really mean if \( x \) is not None:—e.g., when testing whether a variable or argument that defaults to None was set to some other value. The other value might be a value that’s false in a boolean context!

• Never compare a boolean variable to False using ==. Use if not \( x \): instead. If you need to distinguish False from None then chain the expressions, such as if not \( x \) and \( x \) is not None:.

• For sequences (strings, lists, tuples), use the fact that empty sequences are false, so if not seq: or if seq: is preferable to if len(seq): or if not len(seq):.

• When handling integers, implicit false may involve more risk than benefit (i.e., accidentally handling None as 0). You may compare a value which is known to be an integer (and is not the result of len()) against the integer 0.

Yes:

```python
if not users:
    print 'no users'
if foo == 0:
    self.handle_zero()
if i % 10 == 0:
    self.handle_multiple_of_ten()
```

No:

```python
if len(users) == 0:
    print 'no users'
if foo is not None and not foo:
    self.handle_zero()
if not i % 10:
    self.handle_multiple_of_ten()
```

Note that ‘0’ (i.e., 0 as string) evaluates to true.

### Deprecated Language Features

• Use string methods instead of the string module where possible.
• Use function call syntax instead of apply.
• Use list comprehensions and for loops instead of filter, map, and reduce.

### Definition

Current versions of Python provide alternative constructs that people find generally preferable.

### Decision

We do not use any Python version which does not support these features, so there is no reason not to use the new styles.

No:

```python
words = string.split(foo, ':')
map(lambda x: x[1], filter(lambda x: x[2] == 5, my_list))
apply(fn, args, kwargs)
```

Yes:
words = foo.split(' :')
fn(*args, **kwargs)

**Lexical Scoping : okay to use**

**Definition** A nested Python function can refer to variables defined in enclosing functions, but can not assign to them. Variable bindings are resolved using lexical scoping, that is, based on the static program text. Any assignment to a name in a block will cause Python to treat all references to that name as a local variable, even if the use precedes the assignment. If a global declaration occurs, the name is treated as a global variable.

An example of the use of this feature is:

```python
def get_adder(summand1):
    """Returns a function that adds numbers to a given number.""
    def adder(summand2):
        return summand1 + summand2
    return adder
```

**Pros** Often results in clearer, more elegant code. Especially comforting to experienced Lisp and Scheme (and Haskell and ML and ...) programmers.

**Cons** Can lead to confusing bugs. Such as this example based on PEP-0227:

```python
i = 4
def foo(x):
    def bar():
        print i,
        # ...
        # A bunch of code here
        # ...
        for i in x: # Ah, i *is* local to Foo, so this is what Bar sees
            print i,
        bar()
```

So foo([1, 2, 3]) will print 1 2 3 3, not 1 2 3 4.

**Decision** Okay to use.

**Function and Method Decorators: use judiciously** Use decorators judiciously when there is a clear advantage.

**Definition** Decorators for Functions and Methods (a.k.a “the @ notation”).

The most common decorators are `@classmethod` and `@staticmethod`, for converting ordinary methods to class or static methods.

However, the decorator syntax allows for user-defined decorators as well.

Specifically, for some function `my_decorator`, this:

```python
class C(object):
    @my_decorator
    def method(self):
        # method body ...
```
is equivalent to:

class C(object):
    def method(self):
        # method body ...
        method = my_decorator(method)

**Pros**  Elegantly specifies some transformation on a method; the transformation might eliminate some repetitive code, enforce invariants, etc.

**Cons**  Decorators can perform arbitrary operations on a function’s arguments or return values, resulting in surprising implicit behavior. Additionally, decorators execute at import time. Failures in decorator code are pretty much impossible to recover from.

**Decision**  Use decorators judiciously when there is a clear advantage.

Decorators should follow the same import and naming guidelines as functions.

Decorator pydoc should clearly state that the function is a decorator.

Write unit tests for decorators.

Avoid external dependencies in the decorator itself (e.g. don’t rely on files, sockets, database connections, etc.), since they might not be available when the decorator runs (at import time, perhaps from pychecker or other tools).

A decorator that is called with valid parameters should (as much as possible) be guaranteed to succeed in all cases.

Decorators are a special case of “top level code” - see *main* for more discussion.

**Lambda Functions : okay for one-liners**

**Definition**  Lambdas define anonymous functions in an expression, as opposed to a statement. They are often used to define callbacks or operators for higher-order functions like map() and filter().

**Pros**  Convenient.

**Cons**  Harder to read and debug than local functions. The lack of names means stack traces are more difficult to understand. Expressiveness is limited because the function may only contain an expression.

**Decision**  Okay to use them for one-liners. If the code inside the lambda function is any longer than 60–80 chars, it’s probably better to define it as a regular (nested) function.

For common operations like multiplication, use the functions from the operator module instead of lambda functions.

For example, prefer operator.mul to lambda x, y: x * y.

**Threading : Do not rely on the atomicity of built-in types**  Do not rely on the atomicity of built-in types.

While Python’s built-in data types such as dictionaries appear to have atomic operations, there are corner cases where they aren’t atomic (e.g. if __hash__ or __eq__ are implemented as Python methods) and their atomicity should not be relied upon.

Neither should you rely on atomic variable assignment (since this in turn depends on dictionaries).

Use the Queue module’s Queue data type as the preferred way to communicate data between threads.

Otherwise, use the threading module and its locking primitives. Learn about the proper use of condition variables so you can use threading.Condition instead of using lower-level locks.
Power Features: avoid these features

Definition Python is an extremely flexible language and gives you many fancy features such as metaclasses, access to bytecode, on-the-fly compilation, dynamic inheritance, object reparenting, import hacks, reflection, modification of system internals, etc.

Pros These are powerful language features. They can make your code more compact.

Cons It’s very tempting to use these “cool” features when they’re not absolutely necessary. It’s harder to read, understand, and debug code that’s using unusual features underneath. It doesn’t seem that way at first (to the original author), but when revisiting the code, it tends to be more difficult than code that is longer but is straightforward.

Decision Avoid these features in your code.

Avoid global variables

Definition Variables that are declared at the module level.

Pros Occasionally useful.

Cons Has the potential to change module behavior during the import, because assignments to module-level variables are done when the module is imported.

Decision Avoid global variables in favor of class variables.

Some exceptions are:

- Default options for scripts.
- Module-level constants. For example: PI = 3.14159. Constants should be named using all caps with underscores; see Naming.
- It is sometimes useful for globals to cache values needed or returned by functions.
- If needed, globals should be made internal to the module and accessed through public module level functions; see Naming.

python google syle rules

Line length: maximum line length is 80 characters

Exception lines importing modules may end up longer than 80 characters.

Make use of Python’s implicit line joining inside parentheses, brackets and braces.

If necessary, you can add an extra pair of parentheses around an expression.

Yes:

```python
fooBar(self, width, height, color='black', design=None, x='foo',
       emphasis=None, highlight=0)
if ((width == 0) and (height == 0)
    and (color == 'red') and (emphasis == 'strong')):
```

When a literal string won’t fit on a single line, use parentheses for implicit line joining:

```python
x = ('This will build a very long long long string')
```
Make note of the indentation of the elements in the line continuation examples above; see the indentation section for explanation.

**Semicolons** Do not terminate your lines with semi-colons and do not use semi-colons to put two commands on the same line.

**Use parentheses sparingly** Do not use them in return statements or conditional statements unless using parentheses for implied line continuation. (See above.)

It is however fine to use parentheses around tuples.

Yes:

```python
if foo:
    bar()
while x:
    x = bar()
if x and y:
    bar()
if not x:
    bar()
return foo
for (x, y) in dict.items(): ...
```

No:

```python
if (x):
    bar()
if not (x):
    bar()
return (foo)
```

**Indentation** Indent your code blocks with 4 spaces.

**Warning:** Never use tabs or mix tabs and spaces.

In cases of implied line continuation, you should align wrapped elements:

- either vertically, as per the examples in the line length section;
- or using a hanging indent of 4 spaces, in which case there should be no argument on the first line.

Yes:

```python
# Aligned with opening delimiter
foo = long_function_name(var_one, var_two,
                        var_three, var_four)

# 4-space hanging indent; nothing on first line
foo = long_function_name(
    var_one, var_two, var_three,
    var_four)
```

No:
# Stuff on first line forbidden
foo = long_function_name(var_one, var_two, var_three, var_four)

# 2-space hanging indent forbidden
foo = long_function_name(
    var_one, var_two, var_three, var_four)

**Blank Lines**  Two blank lines between top-level definitions, one blank line between method definitions.
Two blank lines between top-level definitions, be they function or class definitions.
One blank line between method definitions and between the class line and the first method.
Use single blank lines as you judge appropriate within functions or methods.

**Whitespace**  Follow standard typographic rules for the use of spaces around punctuation.
No whitespace inside parentheses, brackets or braces.
Yes:

```python
spam(ham[1], {eggs: 2}, [])
```

No:

```python
spam( ham[ 1 ], { eggs: 2 }, [ ] )
```

No whitespace before a comma, semicolon, or colon.
Do use whitespace after a comma, semicolon, or colon except at the end of the line.
Yes:

```python
if x == 4:
    print x, y
x, y = y, x
```

No:

```python
if x == 4 :
    print x , y
x , y = y , x
```

No whitespace before the open paren/bracket that starts an argument list, indexing or slicing.
Yes:

```python
spam(1)
```

No:

```python
spam (1)
```
Yes:

```python
dict[‘key’] = list[index]
```

No:

```python
dict [‘key’] = list [index]
```

Surround binary operators with a single space on either side for assignment (=), comparisons (==, <, >, !=, <=, >=, in, not in, is, is not), and Booleans (and, or, not).

Use your better judgment for the insertion of spaces around arithmetic operators but always be consistent about whitespace on either side of a binary operator.

Yes:

```python
x == 1
```

No:

```python
x<1
```

Don’t use spaces around the ‘=’ sign when used to indicate a keyword argument or a default parameter value.

Yes:

```python
def complex(real, imag=0.0): return magic(r=real, i=imag)
```

No:

```python
def complex(real, imag = 0.0): return magic(r = real, i = imag)
```

Don’t use spaces to vertically align tokens on consecutive lines, since it becomes a maintenance burden (applies to :, #, =, etc.).

Yes:

```python
foo = 1000  # comment
long_name = 2  # comment that should not be aligned
dictionary = {
    "foo": 1,
    "long_name": 2,
}
```

No:

```python
foo = 1000  # comment
long_name = 2  # comment that should not be aligned
dictionary = {
    "foo" : 1,
    "long_name": 2,
}
```
Shebang Line Most .py files do not need to start with a #! line.

Start the main file of a binary with #!/usr/bin/python.

This line is used by the kernel to find the Python interpreter, but is ignored by Python when importing modules.

It is only necessary on a file that will be executed directly.

python google comment rules

Doc Strings Python has a unique commenting style using doc strings.

A doc string is a string that is the first statement in a package, module, class or function.

These strings can be extracted automatically through the __doc__ member of the object and are used by pydoc. (Try running pydoc on your module to see how it looks.)

Our convention for doc strings is to use the three double-quote format for strings.

A doc string should be organized as a summary line (one physical line) terminated by a period, question mark, or exclamation point, followed by a blank line, followed by the rest of the doc string starting at the same cursor position as the first quote of the first line.

There are more formatting guidelines for doc strings below.

Modules comment Every file should contain the following items, in order:

- a copyright statement (for example, Copyright 2008 Google Inc.)
- a license boilerplate. Choose the appropriate boilerplate for the license used by the project (for example, Apache 2.0, BSD, LGPL, GPL)
- an author line to identify the original author of the file

Functions and Methods As used in this section “function” applies to methods, function, and generators.

A function must have a docstring, unless it meets all of the following criteria:

- not externally visible
- very short
- obvious

A docstring should give enough information to write a call to the function without reading the function’s code. A docstring should describe the function’s calling syntax and its semantics, not its implementation.

For tricky code, comments alongside the code are more appropriate than using docstrings.

Certain aspects of a function should be documented in special sections, listed below.

Each section begins with a heading line, which ends with a colon.

Sections should be indented two spaces, except for the heading.

Args List each parameter by name. A description should follow the name, and be separated by a colon and a space. If the description is too long to fit on a single 80-character line, use a hanging indent of 2 or 4 spaces (be consistent with the rest of the file).

The description should mention required type(s) and the meaning of the argument.

If a function accepts *foo (variable length argument lists) and/or **bar (arbitrary keyword arguments), they should be listed as *foo and **bar.
Returns (or Yields: for generators)

Describe the type and semantics of the return value. If the function only returns None, this section is not required.

Raises List all exceptions that are relevant to the interface.

def fetch_bigtable_rows(big_table, keys, other_silly_variable=None):
    """Fetches rows from a Bigtable.

    Retrieves rows pertaining to the given keys from the Table instance
    represented by big_table. Silly things may happen if
    other_silly_variable is not None.

    Args:
    big_table: An open Bigtable Table instance.
    keys: A sequence of strings representing the key of each table row
          to fetch.
    other_silly_variable: Another optional variable, that has a much
                          longer name than the other args, and which does nothing.

    Returns:
    A dict mapping keys to the corresponding table row data
    fetched. Each row is represented as a tuple of strings. For
    example:

    {'Serak': ('Rigel VII', 'Preparer'),
     'Zim': ('Irk', 'Invader'),
     'Lrrr': ('Omicron Persei 8', 'Emperor')}

    If a key from the keys argument is missing from the dictionary,
    then that row was not found in the table.

    Raises:
    IOError: An error occurred accessing the bigtable.Table object.
    ""
    pass

Classes should have a doc string Classes should have a doc string below the class definition describing the class.

If your class has public attributes, they should be documented here in an Attributes section and follow the same formatting as a function’s Args section.

class SampleClass(object):
    """Summary of class here.

    Longer class information.... Longer class information....

    Attributes: likes_spam: A boolean indicating if we like SPAM or not. eggs: An integer count of the eggs we
                have laid.

    ""

    def __init__(self, likes_spam=False):
        """Init SampleClass with blah.""
        self.likes_spam = likes_spam
        self.eggs = 0

    def public_method(self):
        """Performs operation blah.""
"""
**Block and Inline Comments**  The final place to have comments is in tricky parts of the code.

If you’re going to have to explain it at the next code review, you should comment it now.

Complicated operations get a few lines of comments before the operations commence. Non-obvious ones get comments at the end of the line.

```python
# We use a weighted dictionary search to find out where i is in
# the array. We extrapolate position based on the largest num
# in the array and the array size and then do binary search to
# get the exact number.

if i & (i-1) == 0:  # true iff i is a power of 2
    # BAD COMMENT: Now go through the b array and make sure whenever i occurs
    # the next element is i+1
```

To improve legibility, these comments should be at least 2 spaces away from the code.

On the other hand, never describe the code. Assume the person reading the code knows Python (though not what you’re trying to do) better than you do.

```python
# BAD COMMENT: Now go through the b array and make sure whenever i occurs
# the next element is i+1
```

**Strings: Use the % operator for formatting strings**  Use the % operator for formatting strings, even when the parameters are all strings.

Use your best judgement to decide between + and % though.

No:

```python
x = `%(a, b) % use + in this case
x = imperative + ', ' + expletive + '!

x = 'name: ' + name + '; score: ' + str(n)
```

Yes:

```python
x = a + b
x = `%(imperative, expletive)
x = 'name: %s; score: %d' % (name, n)
```

Avoid using the + and += operators to accumulate a string within a loop. Since strings are immutable, this creates unnecessary temporary objects and results in quadratic rather than linear running time.

Instead, add each substring to a list and `.join` the list after the loop terminates (or, write each substring to a cStringIO.StringIO buffer).

No:

```python
employee_table = `<table>`
for last_name, first_name in employee_list:
    employee_table += `<tr><td>%s, %s</td></tr>` % (last_name, first_name)
employee_table += `<table>`
```

Yes:
items = ['<table>']
for last_name, first_name in employee_list:
    items.append('<tr><td>%s, %s</td></tr>' % (last_name, first_name))
items.append('</table>')
employee_table = ''.join(items)

Use “""" for multi-line strings rather than """.

Note: however, that it is often cleaner to use implicit line joining since multi-line strings do not flow with the indentation of the rest of the program.

No:

print """"This is pretty ugly."

Don’t do this. """

Yes:

print ("This is much nicer.\n"Do it this way.\n")

TODO Comments Use TODO comments for code that is temporary, a short-term solution, or good-enough but not perfect.

TODOs should include the string TODO in all caps, followed by the name, e-mail address, or other identifier of the person who can best provide context about the problem referenced by the TODO, in parentheses.

A colon is optional. A comment explaining what there is to do is required.

The main purpose is to have a consistent TODO format that can be searched to find the person who can provide more details upon request.

A TODO is not a commitment that the person referenced will fix the problem. Thus when you create a TODO, it is almost always your name that is given.

# TODO(kl@gmail.com): Use a "*" here for string repetition.
# TODO(Zeke) Change this to use relations.

If your TODO is of the form “At a future date do something” make sure that you either include a very specific date (“Fix by November 2009”) or a very specific event (“Remove this code when all clients can handle XML responses.”).

Imports formatting Imports should be on separate lines.

E.g.:

Yes:

import os
import sys

No:
import os, sys

Imports are always put at the top of the file, just after any module comments and doc strings and before module globals and constants.

Imports should be grouped with the order being most generic to least generic:

• standard library imports
• third-party imports
• application-specific imports

Within each grouping, imports should be sorted lexicographically, ignoring case, according to each module’s full package path.

import foo
from foo import bar
from foo.bar import baz
from foo.bar import Quux
from Foob import ar

Classes inherit If a class inherits from no other base classes, explicitly inherit from object. This also applies to nested classes.

No:

class SampleClass:
    pass

class OuterClass:
    
    class InnerClass:
        pass

Yes:

class SampleClass(object):
    pass

class OuterClass(object):
    
    class InnerClass(object):
        pass

class ChildClass(ParentClass):
    """Explicitly inherits from another class already."""

Inheriting from object is needed to make properties work properly, and it will protect your code from one particular potential incompatibility with Python 3000.

It also defines special methods that implement the default semantics of objects including __new__, __init__, __delattr__, __getattr__, __setattr__, __hash__, __repr__, and __str__.
Statements: only one statement per line  Generally only one statement per line.
However, you may put the result of a test on the same line as the test only if the entire statement fits on one line. In particular, you can never do so with try/except since the try and except can’t both fit on the same line, and you can only do so with an if if there is no else.

Yes:

```python
if foo: bar(foo)
```

No:

```python
if foo: bar(foo)
else: baz(foo)
```

```python
try: bar(foo)
except ValueError: baz(foo)
```

```python
try:
    bar(foo)
except ValueError: baz(foo)
```

Access Control  If an accessor function would be trivial you should use public variables instead of accessor functions to avoid the extra cost of function calls in Python.

When more functionality is added you can use property to keep the syntax consistent.

On the other hand, if access is more complex, or the cost of accessing the variable is significant, you should use function calls (following the Naming guidelines) such as get_foo() and set_foo().

If the past behavior allowed access through a property, do not bind the new accessor functions to the property.

Any code still attempting to access the variable by the old method should break visibly so they are made aware of the change in complexity.

**Naming guideline**

- module_name
- package_name
- ClassName
- method_name
- ExceptionName
- function_name
- GLOBAL_CONSTANT_NAME
- global_var_name
- instance_var_name
- function_parameter_name
- local_var_name

**Names to Avoid**

- single character names except for counters or iterators
- dashes (-) in any package/module name
- __double_leading_and_trailing_underscore__ names (reserved by Python)
Naming Convention

• “Internal” means internal to a module or protected or private within a class.

• Prepending a single underscore (_) has some support for protecting module variables and functions (not included with import * from). Prepending a double underscore (__ ) to an instance variable or method effectively serves to make the variable or method private to its class (using name mangling).

• Place related classes and top-level functions together in a module. Unlike Java, there is no need to limit yourself to one class per module.

• Use CapWords for class names, but lower_with_under.py for module names. Although there are many existing modules named CapWords.py, this is now discouraged because it’s confusing when the module happens to be named after a class. (“wait – did I write import StringIO or from StringIO import StringIO?”)

<table>
<thead>
<tr>
<th>Type</th>
<th>Public</th>
<th>Internal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packages</td>
<td>lower_with_under</td>
<td>_lower_with_under</td>
</tr>
<tr>
<td>Modules</td>
<td>lower_with_under</td>
<td>_lower_with_under</td>
</tr>
<tr>
<td>Classes</td>
<td>CapWords</td>
<td>_CapWords</td>
</tr>
<tr>
<td>Exceptions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functions</td>
<td>lower_with_under()</td>
<td>_lower_with_under()</td>
</tr>
<tr>
<td>Global/Class Constants</td>
<td>CAPS_WITH_UNDER</td>
<td>_CAPS_WITH_UNDE</td>
</tr>
<tr>
<td>Global/Class Variables</td>
<td>lower_with_under</td>
<td>_lower_with_under</td>
</tr>
<tr>
<td>Instance Variables</td>
<td>lower_with_under</td>
<td>_lower_with_under (pro)</td>
</tr>
<tr>
<td>Method Names</td>
<td>lower_with_under()</td>
<td>_lower_with_under() (pro)</td>
</tr>
<tr>
<td>Function/Method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameters</td>
<td>lower_with_under</td>
<td></td>
</tr>
<tr>
<td>Local Variables</td>
<td>lower_with_under</td>
<td></td>
</tr>
</tbody>
</table>

Guidelines derived from Guido’s Recommendations

Main  Even a file meant to be used as a script should be importable and a mere import should not have the side effect of executing the script’s main functionality.

The main functionality should be in a main() function.

In Python, pychecker, pydoc, and unit tests require modules to be importable.

Your code should always check if __name__ == ‘__main__’ before executing your main program so that the main program is not executed when the module is imported.

```python
def main():
    ...

if __name__ == '__main__':
    main()
```

All code at the top level will be executed when the module is imported.

Be careful not to call functions, create objects, or perform other operations that should not be executed when the file is being pychecked or pydoced.

Parting Words  Warning:  BE CONSISTENT.

If you’re editing code, take a few minutes to look at the code around you and determine its style.

If they use spaces around all their arithmetic operators, you should too.
If their comments have little boxes of hash marks around them, make your comments have little boxes of hash marks around them too.

The point of having style guidelines is to have a common vocabulary of coding so people can concentrate on what you’re saying rather than on how you’re saying it.

We present global style rules here so people know the vocabulary, but local style is also important.
If code you add to a file looks drastically different from the existing code around it, it throws readers out of their rhythm when they go to read it. Avoid this.

python pocoo style guide   See Also:
   • http://www.pocoo.org//internal/styleguide/

Python installation

Bento, packaging   See Also:
   • http://cournape.github.com/Bento/

Contents
   • Bento, packaging
      – Introduction
      – Reaction de Tarek (21 juin 2012)
         • Declare stable, include in 3.3

Introduction   Bento is a packaging tool solution for python softwares, targeted as an alternative to distutils, setuptools, distribute, etc....

Bento philosophy is reproducibility, extensibility and simplicity (in that order).
Bento is under constant development. The main implemented features are

• Declarative description of your package. Metadata and package content are described in a simple format.
• Flexible installation scheme: you can install any file anywhere.
• Layered internal architecture: core functionalities, command line interface are clearly separated: bento is designed as a library from the ground-up.
• Hackable and extensible: bento is designed to be extensible: commands are not hardly coupled, and new ones can be inserted between existing ones without the need for monkey-patching.
• Scale down: a core principle of bento is to make software easier to package, and avoid too many choices for packages with simple needs.
• Scale up: on the other hand, bento is designed for complex packages. Bento is already capable to build numpy and scipy with less code than distutils.
• Distutils Compatibility Layer: bento can convert an existing distutils package into the bento format. Bento packages can be installed with tools which normally expect a setup.py (pip, easy_install, etc...).
Reaction de Tarek (21 juin 2012)

David Cournapeau’s Bento project takes the opposite approach, everything is explicit and without any magic.
http://cournape.github.com/Bento/
It had its 0.1.0 release a week ago.
Please, I don’t want to reopen any discussions about Bento here -- distutils2 vs. Bento discussions have been less than constructive in the past -- I just wanted to make sure everybody is aware that distutils2 isn’t the only horse in this race.
I don’t know if there are others too?

That’s exactly the kind of approach that has made me not want to continue.
People are too focused on implementations, and ‘how distutils sucks’ ‘how setuptools sucks’ etc ‘I’ll do better’ etc

Instead of having all the folks involved in packaging sit down together and try to fix the issues together by building PEPs describing what would be a common set of standards, they want to create their own tools from scratch.

That will not work. And I will say here again what I think we should do imho:

1. take all the packaging PEPs and rework them until everyone is happy compilation sucks in distutils ? write a PEP !!!

2. once we have a consensus, write as many tools as you want, if they rely on the same standards => interoperability => win.

But I must be naive because everytime I tried to reach people that were building their own tools to ask them to work with us on the PEPs, all I was getting was “distutils sucks!”

It worked with the OS packagers guys though, we have built a great data files managment system in packaging + the versions (386)

Declare stable, include in 3.3

On Wed, Jun 20, 2012 at 9:31 PM, Tarek Ziade<tarek@ziade.org> wrote:
   Yeah maybe this subset could be left in 3.3
   and we’d remove packaging-the-installer part (pysetup, commands, compilers)
   I think it’s a good idea !
   OK, to turn this into a concrete suggestion based on the packaging docs.
Declare stable, include in 3.3
------------------------------------------
packaging.version -- Version number classes
packaging.metadata -- Metadata handling
packaging.markers -- Environment markers
packaging.database -- Database of installed distributions

I think that’s a good subset.
+1 on all of the things you said after
If you succeed on getting the sci people working on “PEP: Distutils replacement: Compiling Extension Modules” it will be a big win.

Distribute, packaging  See Also:

- http://foobar.lu/wp/2012/05/13/a-comprehensive-step-through-python-packaging-a-k-a-setup-scripts/
Newbie hint on installing Python and it’s modules and packages  by Lennart Regebro on February 2, 2011

There is an issue that pops up from time to time, primarily with Python newbies on Stackoverflow.com and the like, but also with more experienced programmers.

And that is how to use package X that you installed for Python Y with Python Z. Sometimes it is a library like PIL or NumPy, but quite often it’s pip or virtualenv and it’s associated scripts.

**Warning:** And the answer is: You don’t!

You install each library/module/package separately, once for each version of Python. The best way to do it is like this:

1. You install each version of Python in a separate directory, like /opt/pythonXX, where XX is the version of Python, and /opt/pypy14 for PyPy, and so on.
2. You download the script to install Distribute and run it, once for each version of Python you have installed.
3. You will then get one easy_install script in each /opt/pythonXX/bin, so you run /opt/pythonXX/bin/easy_install virtualenv and optionally /opt/pythonXX/bin/easy_install pip for each version of Python.
4. And that’s it.

How do you then use this? Well:

1. Now when you have a project that needs NumPy you make a project directory with the virtualenv of the Python you want to use for that project. Like so: /opt/python26/bin/virtualenv /my/projects/coolmaths
2. Then you go to that directory: cd /my/projects/coolmaths
3. Now you can, if you like, activate that virtualenv, but I never do that. Instead I just call the correct scripts explicitly like so ./bin/easy_install numpy
4. Again, that’s it.

**It really is that simple**  You install every Python separately, you install every module and package separately in those Python install, and you use everything explicitly.

That solves all your problems.

You can install all pythons in the same directory, like /usr/local, or /opt/pythons.

This works well with python, and you then will have to chose which python by adding the version number to the executable: /usr/local/bin/python-2.4, for example.

Distribute will work well with this too, you get /usr/local/bin/easy_install-2.7 etc.

However, afaik can tell, pip does not always do this (maybe it’s a question of which pip version, or maybe it’s intelligent enough to know when it does have to? Not sure), so there you may have to make a copy of the pip script each time.

But in any case, it’s much harder to delete and re-install a Python if you install them together like this.

With separate directories you can, if your Python is fudged up, just remove the directory and do it again.

Also: Avoid the system Python on OS X or Linux distros for your projects.

You get stuck on a specific version, and you get conflicts with versions of packages, etc.

The OS needs to have it’s versions, and you often needs others, and using the same Python install makes everything very complicated.
So there. Now you know.

PEP 0394 The “python” command on Unix-Like Systems (march 2011) See Also:
http://www.python.org/dev/peps/pep-0394/

Working with virtualenv See Also:
http://blog.adamdklein.com/?p=416
You should have installed:

- virtualenv
- pip
- virtualenvwrapper

by now.
This section is not unique to Django projects. Any Python project can use a virtualenv like this.

Creating a virtualenv is easy ::

mkvirtualenv PROJECTNAME

Note: this will not create a project directory. You need to do this yourself:

mkdir -p PROJECTDIR

Very often, you will like to modify PYTHONPATH to include your project directory in it.
This depends on layout of your project of course. More on this later.

add2virtualenv $(pwd)/PROJECTDIR

Deactivating virtualenv To deactivate or log out of a virtualenv, simply type:

deactivate

Or switch to another project:

workon ANOTHERPROJECT
Installing dependencies  Once you have activated your virtualenv by mkvirtualenv above or manually by:

```bash
workon PROJECT
```

you can start installing dependencies with pip. Let’s install the Django trunk with the now very popular South:

```bash
pip install -e svn+http://code.djangoproject.com/svn/django/trunk#egg=django
pip install -e hg+http://bitbucket.org/andrewgodwin/south#egg=south
```

Make sure you check out the pip documentation for more. You may still want to and can install packages globally and the virtualenv will recognize it.

The following will install PIL for example:

```bash
apt-get install python-imaging
```

Django project layout  Much has been written on project layout in Django projects and many exist.

Here is one of them.:

```bash
mysite/  docs/ resources/ myproject/ setup.py INSTALL AUTHORS README
```

myproject is the top level project directory and can be the name of the virtualenv for clarity.

It contains the django project myproject and other files and directories which do not have anything to do with Django.

This extra directory (atop myproject) has been recommended by many and trust me, you will need it later as your project grows.

```bash
myproject/
  settings.py
  urls.py
  apps/
    myapp1/
    myapp2/
  libs/
  templates/
    myapp1/
    myapp2/
```

You will see why we are having this discussion in the virtualenv article at all. Referencing other modules

There is little confusion about how to refer to Django applications and Django parts of the code. Most projects do it like this:

```python
from django.contrib.auth.models import User
from django.contrib.contenttypes import generic
from django.db import connection
```

Given the layout above we can refer to myprojects’s objects in the same fashion:

```python
from myproject.apps.myapp1.models import Model1 from myproject.libs.mylib import myfunc
```

You can clearly separate objects from either one. Of course, this means PYTHONPATH contains the path to mysite like you should have added earlier:
add2virtualenv $(pwd)/mysite

**DJANGO_SETTINGS_MODULE**  With the layout from above, DJANGO_SETTINGS_MODULE has to be mysite.settings. You can have virtualenvwrapper set it for you each time you switch to your virtualenv:

cd virtualenv

```
  echo "export DJANGO_SETTINGS_MODULE=mysite.settings" > bin/postactivate
```

**Python modules**

See Also:

- [https://pythonpackages.com/](https://pythonpackages.com/)

Packaginator solves the problem in the Python community of being able to easily identify good apps, frameworks, and packages.

Ever want to know which is the most popular or well supported Python httplib replacement, web framework, or api tool?

Packaginator solves that problem for you!

**Python argument parsers**

**plac**  See Also:

- [http://pypi.python.org/pypi/plac](http://pypi.python.org/pypi/plac)
- pep 0389 - argcparse - New Command Line Parsing Module

There is no want of command line arguments parsers in the Python world. The standard library alone contains three different modules: getopt (from the stone age), optparse (from Python 2.3) and argparse (from Python 2.7). All of them are quite powerful and especially argparse is an industrial strength solution; unfortunately, all of them feature a non-zero learning curve and a certain verbosity. They do not scale down well, at least in my opinion.

It should not be necessary to stress the importance scaling down; nevertheless, a lot of people are obsessed with features and concerned with the possibility of scaling up, forgetting the equally important issue of scaling down.

This is an old meme in the computing world: programs should address the common cases simply and simple things should be kept simple, while at the same keeping difficult things possible. plac adhere as much as possible to this philosophy and it is designed to handle well the simple cases, while retaining the ability to handle complex cases by relying on the underlying power of argparse.

Technically plac is just a simple wrapper over argparse which hides most of its complexity by using a declarative interface: the argument parser is inferred rather than written down by imperatively. Still, plac is surprisingly scalable upwards, even without using the underlying argparse. I have been using Python for 8 years and in my experience it is extremely unlikely that you will ever need to go beyond the features provided by the declarative interface of plac: they should be more than enough for 99.9% of the use cases.
Introduction  Aaargh: an astonishingly awesome application argument helper.

Aaargh is a Python module that makes building friendly command line applications really easy.

Applications built with Aaargh provide a single executable with a subcommand for each exposed Python function. Each subcommand may have its own command line arguments. This is similar to the way version control systems provide many different commands using a single entry point. (Examples include bzr commit and git checkout).

Aaargh is named after one of the castles in the movie Monty Python and the Holy Grail. The acronym Aaargh expands to an astonishingly awesome application argument helper, but omits a few letters to make it triple A.

Aaargh works with both Python 2.6+ and Python 3.

Rationale  The Python standard library contains the optparse, getopt, and argparse modules, and out in the wild you will find many alternative command line interface libraries stacked on top of these, such as Cliff, Cement, opster, plac, and many others.

These libraries either separate the CLI part of your application from the actual code, force yet another API upon you, or even force you to hide your code in non-obvious framework constructs.

This makes you scream aaargh. And, lo and behold, here it is!

Usage  Aaargh delegates almost all of its work to the argparse module, which does a great job handling arguments and printing usage information.

However, argparse is a bit verbose and cumbersome for many simple applications, so Aaargh lets application authors minimize boilerplate code by wrapping commonly used argparse features in a few non-intrusive decorators.

Aaargh does not hide the argparse API, since the decorators have exactly the same API as their argparse counterparts. This is a deliberate design decision, and this is what makes Aaargh different from its many alternatives.

The docstrings in the aaargh.py file contain all information you need to use Aaargh.

Refer to the argparse documentation for information on specifying arguments, providing defaults, adding help texts, and so on.
**Python blargs module**  See Also:

- https://bitbucket.org/gyllstromk/blargs
- http://pypi.python.org/pypi/blargs

blargs provides easy command line parsing, as an alternative to argparse and optparse from Python’s standard library. The main distinctions are:

- Cleaner, more minimal, and possibly more *pythonic* syntax.
- Support for arbitrarily complex dependency relationships. For example, argument A might require arguments B and C, while conflicting with D; or requiring argument E in the case that A is less than 10 or B is equal to the string ‘wonderful!’.  
- Emphasis on *ease of use* over *configurability*.

Blargs has been tested on Python2.6 to Python3.2 and PyPy.

**Python clom module**  See Also:

- http://pypi.python.org/pypi/clom/

The easiest way to use the command line with Python. Command Line Object Mapper. A library for building POSIX command line arguments, commands, and parameters. Very useful for Fabric tasks.

**Python entrypoint module**  See Also:

- http://pypi.python.org/pypi/entrypoint/

## Contents

- Python entrypoint module
  - Introduction

### Introduction

This is a decorator library that helps one to write small scripts in Python. There are three main features that it provides:

- Automatically running a function when a script is called directly, but not when it is included as a module.
- Automatically generating argument parsers from function signatures and docstrings
- Automatically opening and closing files (with a codec or as binary) when a function is called and returns.

The raison d’être of this library is to be convenient, and it makes some sacrifices in other areas. Notably there are some stringent conditions on the order of application of decorators.
Python entrypoint2 module

See Also:
- http://pypi.python.org/pypi/entrypoint2/

Contents
- Python entrypoint2 module
  - Introduction

Introduction

entrypoint2 is an easy to use argparse based command-line interface for python modules, fork of entrypoint. It translates function signature and documentation to argparse configuration.

Python pbs module

See Also:
- https://github.com/amoffat/pbs
- Python sarge module

PBS is a unique subprocess wrapper that maps your system programs to Python functions dynamically.

PBS helps you write shell scripts in Python by giving you the good features of Bash (easy command calling, easy piping) with all the power and flexibility of Python.

Python plumbum module

See Also:

Contents
- Python plumbum module
  - About
  - Credits

About

The original purpose of Plumbum was to enable local and remote program execution with ease, assuming nothing fancier than good-old SSH.

On top of this, a file-system abstraction layer was devised, so that working with local and remote files would be seamless.

I’ve toyed with this idea for some time now, but it wasn’t until I had to write build scripts for a project I’ve been working on that I decided I’ve had it with shell scripts and it’s time to make it happen.

Plumbum was born from the scraps of the Path class, which I wrote for the aforementioned build system, and the SshContext and SshTunnel classes that I wrote for RPyC.

When I combined the two with shell combinators (because shell scripts do have an edge there) the magic happened and here we are.

Credits

The project has been inspired by PBS of Andrew Moffat, and has borrowed some of his ideas (namely treating programs like functions and the nice trick for importing commands).

However, I felt there was too much magic going on in PBS, and that the syntax wasn’t what I had in mind when I came to write shell-like programs.
I contacted Andrew about these issues, but he wanted to keep PBS this way.

Other than that, the two libraries go in different directions, where Plumbum attempts to provide a more wholesome approach.

Plumbum also pays tribute to Rotem Yaari who suggested a library code-named pyplatform for that very same purpose, but which had never materialized.

**Python pycli module**  
*See Also:*

- CLI and python
- [http://packages.python.org/pyCLI/](http://packages.python.org/pyCLI/)

The cli package is a framework for making simple, correct command line applications in Python.

With cli, you can quickly add standard command line

- parsing;
- logging;
- unit and functional testing;
- and profiling to your CLI apps.

To make it easier to do the right thing, cli wraps all of these tools into a single, consistent application interface

**Python sarge module**  
*See Also:*

- [http://pymolurus.blogspot.com/2012/02/working-with-subprocess.html](http://pymolurus.blogspot.com/2012/02/working-with-subprocess.html)
- **Python pbs module**

Welcome to the documentation for sarge, a wrapper for subprocess which aims to make life easier for anyone who needs to interact with external applications from their Python code.

**Python straight module**  
*See Also:*

- [http://techblog.ironfroggy.com/2012/05/ann-straightcommand-01a1-command.html](http://techblog.ironfroggy.com/2012/05/ann-straightcommand-01a1-command.html)

### Contents

- Python straight module
  - About

**About**  
I’d like to announce a new project, based on straight.plugin, a command framework that provides a declarative way to define command-line options, sub-commands, and allows plugins from third-parties to expand commands.

This is all very early, I’m calling this version 0.1a1 and lots of things are missing, but here is an example (which works) of a small todo application built with this.

**Python communication**
Development tools, Release 2012.06.18

python netzob  See Also:

• netzob (NETwork protocol modeliZatiOn By reverse engineering)
• https://code.google.com/p/netzob/

Dependencies

• tcpdump
• python
• python-ptrace
• python-nfqueue
• python-hachoir
• python-matplotlib
• strace
• lsof
• iptables

pyserial  See Also:

• http://pyserial.sourceforge.net

This module encapsulates the access for the serial port. It provides backends for Python running on Windows, Linux, BSD (possibly any POSIX compliant system), Jython and IronPython (.NET and Mono). The module named “serial” automatically selects the appropriate backend.

Opening serial ports  Open port 0 at “9600.8,N,1”, no timeout:

```python
>>> import serial
>>> ser = serial.Serial(0)  # open first serial port
>>> print ser.portstr  # check which port was really used
>>> ser.write("hello")  # write a string
>>> ser.close()  # close port
```

Open named port at “19200.8,N,1”, 1s timeout:

```python
>>> ser = serial.Serial('/dev/ttyS1', 19200, timeout=1)
>>> x = ser.read()  # read one byte
>>> s = ser.read(10)  # read up to ten bytes (timeout)
>>> line = ser.readline()  # read a ‘\n’ terminated line
>>> ser.close()
```
Open second port at “38400,8,E,1”, non blocking HW handshaking:

```python
>>> ser = serial.Serial(1, 38400, timeout=0,
...                       parity=serial.PARITY_EVEN, rtscts=1)
>>> s = ser.read(100)  # read up to one hundred bytes
...  # or as much is in the buffer
```

scapy  See Also:


python modules configuration

python infi.conf module  See Also:

- https://github.com/vmalloc/infi.conf
- https://pypi.python.org/pypi/infi.conf

infi.conf is a generic mechanism for storing, loading and manipulating configuration.

Le module concurrent configobj

- http://www.blog.pythonlibrary.org/2010/01/01/a-brief-configobj-tutorial/  Python comes with a handy module called ConfigParser. It’s good for creating and reading configuration files (aka INI files). However, Michael Foord (author of IronPython in Action) and Nicola Larosa decided to write their own configuration module called ConfigObj. In many ways, it is an improvement over the standard library’s module

python yay module  See Also:

- http://pypi.python.org/pypi/yay

yay ain’t YAML  yay is a configuration file format built on top of YAML based on our experience with the ConfigParser extensions buildout uses.  It adds overlays (one config file including and extending another) and variables.

Python ctypes modules  See Also:

- http://docs.python.org/library/ctypes.html
- ref: java_native_access

```python
import ctypes
ctypes.__doc__
```

'create and manipulate C data types in Python'
ctypes is a foreign function library for Python. It provides C compatible data types, and allows calling functions in DLLs or shared libraries. It can be used to wrap these libraries in pure Python.

À partir de Python 2.5, la bibliothèque standard inclut le module ctypes, une FFI (Foreign function interface) qui permet d’appeler directement une bibliothèque dynamique depuis du code python.

Python data management

Python bitstring module See Also:


bitstring is a pure Python module that makes the creation, manipulation and analysis of binary data as simple and natural as possible.

Bitstrings can be constructed from integers, floats, hex, octal, binary, bytes or files. They can also be created and interpreted using flexible format strings.

Bitstrings can be sliced, joined, reversed, inserted into, overwritten, etc. with simple methods or using slice notation. They can also be read from, searched and replaced, and navigated in, similar to a file or stream.

Internally the bit data is efficiently stored in byte arrays, the module has been optimized for speed, and excellent code coverage is given by over 400 unit tests.

The latest versions are available for Python 2.6 and later (including Python 3).

For Python 2.4 / 2.5 you can still use version 1.0, but you should refer to the manual that comes with the bitstring 1.0 download as some of the information on this website isn’t applicable to that version.

Python byteformat module See Also:

- [http://pypi.python.org/pypi/byteformat/](http://pypi.python.org/pypi/byteformat/)

byteformat is a Python library and command line script for displaying numbers of bytes as strings using standards-compliant human-readable units such as ‘23 KB’ or ‘1.25 terabytes’.

- Support for the two official and one de facto standards for bytes:
  - SI decimal units, e.g. 1000 bytes = 1 KB;
  - IEC binary units, e.g. 1024 bytes = 1 KiB;
  - Classic units, e.g. 1024 bytes = 1 KB.
- Supports the full set of decimal prefixes from kilo- to yotta- and the binary prefixes kibi- to yobi-.
- Generate strings using symbols (e.g. ‘KB’), abbreviated names (‘Kbyte’) or full names (‘kilobyte’).
- Uses correct plural terms when needed.
- Allows the caller to explicitly choose which unit to use
- Automatically selects the best unit for a given number of bytes.
- Easily customise the output without subclassing.
- Importable as a Python library module.
• Runs as a command line script.

google python data modules  See Also:
  • Google cli

gspread module  See Also:
  https://github.com/burnash/gspread
This is a simple Python library for accessing Google Spreadsheets.

Features
  • Open a spreadsheet by its title.
  • Extract entire row or column values.
  • No need to mess around with spreadsheets’ keys.
  • Independent of Google Data Python client library.

Python petl module Extract, Transform and Load (Tables of Data)  See Also:
  • http://packages.python.org/petl/0.2/index.html
petl is a tentative Python package for extracting, transforming and loading tables of data.
E.g., given the following data in a file at ‘example.csv’ in the current working directory:

```
foo,bar,baz
a,1,3.4
b,2,7.4
c,6,2.2
d,9,8.1
```

...the interactive session below illustrates some simple uses of this module:

```python
>>> from petl import *
>>> table1 = fromcsv('example.csv')
>>> look(table1)
+-------+-------+-------+
| 'foo' | 'bar' | 'baz' |
|-------+-------+-------|
| 'a' | '1' | '3.4' |
+-------+-------+-------+
| 'b' | '2' | '7.4' |
+-------+-------+-------+
| 'c' | '6' | '2.2' |
+-------+-------+-------+
| 'd' | '9' | '8.1' |
+-------+-------+-------+

>>> table2 = convert(table1, 'foo', 'upper')
>>> table3 = convert(table2, 'bar', int)
>>> table4 = convert(table3, 'baz', float)
>>> table5 = extend(table4, 'quux', expr('{bar} * {baz}'))
>>> table6 = cut(table5, 'foo', 'quux')
```
>>> table7 = selectgt(table6, 'quux', 10)
>>> table8 = sort(table7, 'quux')
>>> look(table8)
+-------+--------------------+
| 'foo' | 'quux' |
|-------+--------------------+
| 'C' | 13.200000000000001 |
|-------+--------------------+
| 'B' | 14.8 |
|-------+--------------------+
| 'D' | 72.89999999999999 |
|-------+--------------------+

>>> tocsv(table8, 'output.csv')

Python prettytable module  See Also:
- https://code.google.com/p/prettytable/

PrettyTable is a simple Python library designed to make it quick and easy to represent tabular data in visually appealing ASCII tables, like this.

Warning: the generated table is not a rest table.

Python pyvot module  See Also:
- PyPI: http://pypi.python.org/pypi/Pyvot
- http://packages.python.org/Pyvot/
- Python petl module Extract, Transform and Load (Tables of Data)

Pyvot connects familiar data-exploration and visualization tools in Excel with the powerful data analysis and transformation capabilities of Python, with an emphasis on tabular data. It provides a minimal and Pythonic interface to Excel, smoothing over the pain points in using the existing Excel object model as exposed via COM.

Python tablefactory module  See Also:
- http://kstrauser.github.com/tablefactory/
- git clone git://github.com/kstrauser/tablefactory

TableFactory is a simple API for creating tables in popular formats. It acts as a wrapper around other popular Python report generators and handles all the tedious, boilerplate problems of extracting columns from input data, creating the layout, applying formatting to cells, etc.

Why? I maintain an internal web application with many reports for our customers.

After writing the code to generate a report’s data, I’d have to make custom backends to write that data out in each of several formats that different customers might want.

If someone asked me to add a new column to a report, I’d have to add it separately to each of the output types. That got old quickly.
TableFactory is my solution to the problem. It lets me spend my efforts on making efficient, useful reports instead of on the boring, repetitive process of formatting them.

**Dependencies**  HTML tables are made using only standard Python modules.

Spreadsheets are made with xlwt (http://pypi.python.org/pypi/xlwt).

PDFs are made with ReportLab (http://www.reportlab.com/software/opensource/).

**License** TableFactory is available under the permissive MIT License.

**Python enums modules**

**flul.enum**  See Also:

- http://pypi.python.org/pypi/flul.enum/3.3.1
- http://packages.python.org/flul.enum

The **flul.enum** package provides yet another enumeration data type for Python. While this is similar intent to the reject ['PEP 354'](http://pypi.python.org/pypi/pep-354), this package defines an alternative syntax and semantics.

An enumeration is a set of symbolic names bound to unique, constant integer values. Within an enumeration, the values can be compared by identity, and the enumeration itself can be iterated over. Enumeration items can be converted to and from their integer equivalents, supporting use cases such as storing enumeration values in a database.

**Motivation**  [Lifted from PEP 354]

The properties of an enumeration are useful for defining an immutable, related set of constant values that have a defined sequence but no inherent semantic meaning. Classic examples are days of the week (Sunday through Saturday) and school assessment grades (‘A’ through ‘D’, and ‘F’). Other examples include error status values and states within a defined process.

It is possible to simply define a sequence of values of some other basic type, such as **int** or **str**, to represent discrete arbitrary values. However, an enumeration ensures that such values are distinct from any others, and that operations without meaning (“Wednesday times two”) are not defined for these values.

**Python hash management**

**Python humanhash module**  See Also:

- https://github.com/zacharyvoase/humanhash

humanhash provides human-readable representations of digests.

Example:

```python
>>> import humanhash
>>> digest = '7528880a986c40e78c38115e640da2a1'
>>> humanhash.humanize(digest)
'three-georgia-xray-jig'
>>> humanhash.humanize(digest, words=6)
'high-mango-white-oregon-purple-charlie'
>>> humanhash.uuid()
('potato-oranges-william-friend', '9d2278759ae24698b1345525bd53358b')
```
Caveats  Don’t store the humanhash output, as its statistical uniqueness is only around 1 in 4.3 billion. Its intended use is as a human-readable (and, most importantly, memorable) representation of a longer digest, unique enough for display in a user interface, where a user may need to remember or verbally communicate the identity of a hash, without having to remember a 40-character hexadecimal sequence. Nevertheless, you should keep original digests around, then pass them through humanize() only as you’re displaying them.

How It Works  The procedure for generating a humanhash involves compressing the input to a fixed length (default: 4 bytes), then mapping each of these bytes to a word in a pre-defined wordlist (a default wordlist is supplied with the library). This algorithm is consistent, so the same input, given the same wordlist, will always give the same output. You can also use your own wordlist, and specify a different number of words for output.

Inspiration  
• Chroma-Hash: a human-viewable representation of a hash (albeit not one that can be output on a terminal, or shouted down a hallway).
• The NATO Phonetic Alphabet: A great example of the trade-off between clarity of human communication and byte-wise efficiency of representation.

Python crc module  See Also:  
• http://www.tty1.net/pycrc/index_en.html  
• http://www.tty1.net/pycrc/faq_en.html
pycrc is a free, easy to use Cyclic Redundancy Check (CRC) calculator and source code generator.

Description  pycrc provides CRC reference implementations in Python and a source code generator for C. The used CRC variant can be chosen from a fast but space-consuming implementation to slower but smaller versions especially suitable for embedded applications. The models can be freely chosen, but a comprehensive collection of CRC models is available by name. The following functions are implemented:
• calculate the checksum of a string or a file.
• generate the source files for a “C” implementation.

The following variants of the CRC algorithm are supported:
• bit-by-bit: the basic algorithm which operates individually on every bit of the augmented message (i.e. the input data with width 0-bits attached to the end). This algorithm is the easiest one to understand, because it’s a direct implementation of the basic polynomial division, but it is also the slowest among all possible variants.
• bit-by-bit-fast: a variation of the simple bit-by-bit algorithm, which doesn’t need the augmented message. This algorithm might be a good choice for embedded platforms, where code space is a major concern.
• table-driven: the standard table driven algorithm. This is the fastest variant, because it operates on bytes as opposed to bits, and uses a look-up table of 256 elements, which might not be feasible for small embedded systems, though. Anyway, the number of elements in the look-up table can be reduced by means of the –table_idx_with command line switch. By using 4 bits (16 elements in the look-up table) the code is still very fast (roughly half the speed of a 8-bit table-driven code) but much more compact.

pycrc is released under the terms of the MIT licence.
**Python GUI**

**easygui  See Also:**


EasyGUI is a module for very simple, very easy GUI programming in Python.

Experienced Pythonistas need to be able to make simple GUI interfaces quickly and easily.

New Python programmers, students, and sysadmins need GUI capabilities that don’t require knowledge of Tkinter, frames, widgets, callbacks or lambda.

This is what EasyGUI provides. Using EasyGUI, all GUI interactions are invoked by simple function calls.

EasyGUI is different from other GUIs in that EasyGUI is NOT event-driven.

It allows you to program in a traditional linear fashion, and to put up dialogs for simple input and output when you need to. If you have not yet learned the event-driven paradigm for GUI programming, EasyGUI will allow you to be productive with very basic tasks immediately.

Later, if you wish to make the transition to an event-driven GUI paradigm, you can do so with a more powerful GUI package such as anygui, PythonCard, Tkinter, wxPython, etc.

**Python pyQt  See Also:**


**pyqt examples  See Also:**


**pyQt libs**

[http://code.google.com/p/formlayout/](http://code.google.com/p/formlayout/)  formlayout is a tiny Python module for creating form dialogs/widgets to edit various type of parameters with having to write any GUI code (it requires Python 2.x and PyQt4 4.y, with x >= 5 and y > 3).

**Installation**  formlayout requires Python 2.x and PyQt4 4.y, with x >= 5 and y > 3:

- All supported platforms (GNU/Linux, MacOS and Microsoft Windows XP/Vista): install Python and PyQt4
- Microsoft Windows XP/Vista: note that Python(x,y) (scientific-oriented Python distribution: [http://www.pythonxy.com](http://www.pythonxy.com)) includes formlayout as a standard plugin

**Installation instructions:**

- Python(x,y): nothing to do, it’s already included
- Without setuptools: download the .tar.gz file, extract files and type python setup.py install
- With setuptools: simply type easy_install formlayout
**Simple example**  The main feature of formlayout is to provide the fedit function which transforms a list of parameters into a GUI-based form (you may also create a list of parameter lists - see formlayout.py for more advanced examples):

```python
from formlayout import fedit
data = [('Name', 'Paul'), ('Age', 30), ('Sex', [0, 'Male', 'Female']), ('Size', 12.1), ('Eyes', 'green'), ('Married', True), ]
print "result:", fedit(data, title="Describe yourself", comment="This is just an example.")
```

**Python pySide**  See Also:
- [http://qt.gitorious.org/pyside](http://qt.gitorious.org/pyside)

**Python pySide news**


---

Sivan Greenberg wrote:

> We should probably suppor it, there are already packages for PySide > no? Who’s responsible for them? If they are > in universe I can tale > care of them and introduce the new version. >> Sivan

Hi,

AFAIK, there is nobody (yet) on the Canonical side taking care of Ubuntu packages.

On the Debian side, I’m working on inclusion of the whole chain (from apiextractor up to pyside) into Debian. apiextractor and generatorrunner are now in unstable and shiboken is waiting on license/distributability review in NEW, aimed for experimental. As soon as shiboken lands in experimental, I’ll release a new update of it to unstable and pyside will follow shortly after. I don’t intend to work specially on Ubuntu packages, because that’d just be diversion of my energy (the packages will land “as is” anyway).

Pyside should normally make it into Squeeze and by such, will be in the _next_ Ubuntu version.

Otherwise, the Pyside team is providing a PPA: [http://ppa.launchpad.net/pyside/ppa/ubuntu](http://ppa.launchpad.net/pyside/ppa/ubuntu)

---

[http://dot.kde.org/2009/08/24/pyside-interview](http://dot.kde.org/2009/08/24/pyside-interview)  ... We wanted to enable the community to be able to develop Qt software on the Maemo platform with Python and not limit their licensing options. Since this was not possible using the existing Python bindings for Qt, we decided to roll our own. Naturally, it was clear to us from day one that the project could only fly if it was available not only for Maemo but other mainstream platforms as well, and was open-sourced as soon as possible. That’s where we are at the moment.
We naturally contacted Riverbank, the creators of the current, GPL licensed Python bindings, during the planning phase as well as later on to see if there would be a possibility to find a common solution to the licensing issue. However, these were private meetings and I hope you understand that we can’t go into more details publicly.

Who are working on PySide?

So far the the code has been written by the OpenBossa developers working at INdT and funded by Nokia. The team will keep on working on the project but would love to see a genuine community project evolve around our contributions.

http://www.pyside.org/docs/pseps/psep-0101.html PyQt4 provides two different (and incompatible) APIs. API 1 is the original API and the one supported by PySide. API 2 is a new Python 3-specific API that is much more Pythonic. This PSEP proposes that PySide adopt PyQt4’s API 2 for PySide.

Rationale PyQt4 provides two different (and incompatible) APIs [1]. API 1 is useful for those using PyQt4 to prototype C++/Qt programs since it is very close to the C++/Qt API. However, for those who want to create Python Qt programs in their own right, and especially for existing Python programmers, API 1 is cumbersome and un-Pythonic when it comes to handling QStrings and QVariants, both of which are used a lot.

If PySide were to support API 2, it will make PySide much more attractive to Python 3 programmers.

Also, supporting API 2 will mean that existing Python 3/PyQt4 programs that use API 2 by default will be able to switch to PySide.

One key purpose of API 2 is to avoid the need for Python programmers to have to worry or even know about QString or QVariant. For Python programmers str is the string they are used to so translation to/from QString should be transparent and automatic. Similarly, why should Python programmers have to know about QVariant when that is a C++ workaround for C++’s lack of support for duck typing etc.? Again, translation to/from QVariant should be transparent and automatic. API 2 achieves this and eliminates QString and (to some extent) QVariant.

See Also:

vsgui http://pypi.python.org/pypi/vsgui

python i18n modules

unicode python See Also:


de Victor Stinner victor.stinner@haypocalc.com
to python-dev@python.org
date 24 juin 2011 13:18
objet Re: [Python-Dev] EuroPython Language Summit report

Le vendredi 24 juin 2011 à 10:52 +0200, Mark Dickinson a écrit :
> - [Armin Ronacher] Python 3’s Unicode support still has some dark areas.

What ? Unicode support is perfect in Python 3!

> One example: when opening a text file for reading and writing, the default
> encoding used depends on the platform and on various environment variables.
... oh, I agree.

This choice is a big portability issue. Mac OS X, most Linux distro, BSD systems use UTF-8 local encoding, whereas Windows use legacy code pages like cp1252 (something like ISO-8859-1) or cp952 (shift jis).

But sometimes, the locale is “C” (e.g. on our buildbots) and programs start to fail with Unicode errors...

I see two options to improve the situation

1) hard way  change open() API to make encoding a mandatory argument.

   Problem   it breaks compatibility with Python 3.0, 3.1 and 3.2 (ooops!); the encoding argument is the 4th argument, you have to use a keyword or choose a value for the buffering argument.

   I proposed to change open() API in Python 3.1 to make all arguments -except the filename and the mode- keyword-only arguments, but Guido rejected my idea:

   "Remember, for 3.0 we’re trying to get a release out of the door, not cram in new features, no matter how small."

   http://bugs.python.org/issue4121

2) soft way   add a warning if the encoding is implicit (locale encoding).

   I don’t know what is the best warning type, and if it should be always displayed, only once, or not by default.

   Even if it is hidden by default, a careful developer will be able to use -Werror to fix bugs...

   I suspect that most tests fail if open() raises an exception if the encoding is not specified (e.g. see distutils/packaging issues about the file encoding).

   Victor

   Python tools   See Also:

   http://pypi.python.org/pypi/Unidecode

   python web modules

   python internet rss modules

   python rss2email module   See Also:

   •  http://www.allthingsrss.com/rss2email/download/
   •  http://philippe.scoffoni.net/rss2email-lecture-rss-thunderbird/
   •  http://blog.tomamplius.net/index.php?post/2010/12/08/webrss2email

   a free, open-source tool for Windows and UNIX for getting news from RSS feeds in email
python scapy module  See Also:

  - http://www.secdev.org/projects/scapy/

Scapy is a packet manipulation tool for computer networks, written in Python by Philippe Biondi. It can forge or decode packets, send them on the wire, capture them, and match requests and replies. It can also handle tasks like scanning, tracerouting, probing, unit tests, attacks, and network discovery.

Scapy provides a Python interface into libpcap, (WinPCap on Windows), in a similary way to that in which Wireshark provides a view and capture GUI. It can interface with a number of other programs to provide visualisation including Wireshark for decoding packets, GnuPlot for providing graphs, graphviz or VPython for visualisation, etc.

Python multimedia management

Python PIL module  See Also:

  - http://effbot.org/zone/pil-index.htm

The Python Imaging Library (PIL) adds image processing capabilities to your Python interpreter. This library supports many file formats, and provides powerful image processing and graphics capabilities.

Python simplecv module  See Also:


SimpleCV is an interface for Open Source machine vision libraries in Python. It provides a consise, readable interface for cameras, image manipulation, feature extraction, and format conversion.

Our mission is to give casual users a comprehensive interface for basic machine vision functions and an elegant programming interface for advanced users.

Logging

Le module de logging standard

  - http://docs.python.org/library/logging.html

C’est le module que j’ai utilisé, le module Logbook n’existant pas quand j’ai commencé.

voir la défense du module standard par son créateur:
Some inaccuracies and misunderstandings about how stdlib logging works have been expressed in the documentation and marketing presentations of a suggested alternative. On closer examination, certain statements which imply a lack of functionality or other shortcomings in stdlib logging have been shown to be inaccurate. You should feel confident that in using the stdlib logging package you are very unlikely to find it wanting, and that, if it seems too hard or not possible to achieve some result that you want to achieve, you should raise the issue on comp.lang.python or bugs.python.org and be assured of prompt responses and resolutions. Now, you can read on if you want more details :-).

Recently Armin Ronacher, developer of Werkzeug and Jinja among other things, released a library for logging which, in his opinion, is preferable to the logging package provided as part of Python since the 2.3 release. Now preferences are a very personal thing, and since Armin has made worthy software contributions to the Python community, there will no doubt be many people who trust his judgement and follow where he leads. In general, choice in software is good, and people should be free to make up their own minds about the software they want to use. However, in order to make an informed decision, people need accurate information on which to base that decision. For example, I chose to use argparse over the stdlib’s optparse even before argparse landed in the stdlib; in making that choice, I looked at Steven Bethard’s rationale as described here and was fortunate to have the time to be able to examine and evaluate each of his points for myself.

Le tout récent module concurrent Logbook

Logbook is a logging system for Python that replaces the standard library’s logging module. It was designed with both complex and simple applications and mind and the idea to make logging fun.

module py

This document is an attempt to briefly state the actual specifcation of the py.log module. It was written by Francois Pinard and also contains some ideas for enhancing the py.log facilities.

NOTE that py.log is subject to refactorings, it may change with the next release.

This document is meant to trigger or facilitate discussions. It shamelessly steals from the Agile Testing comments, and from other sources as well, without really trying to sort them out.

python installation modules

python bundle module

A bundle consists of several packages, and can be used as a shortcut in applications and libraries requirements lists. Bundles are used to follow a common group of packages, or a package with an optional extension feature. This distribution lets you create bundles, and may also be able to manage installed bundles in the future.

Python distribute

Python distribute news
At least one package management tool for 2.7

from anatoly techtonik <techtonik@gmail.com>
to Python-Dev <python-dev@python.org>
date Wed, Mar 24, 2010 at 10:59 AM
subject [Python-Dev] At least one package management tool for 2.7
mailing list python-dev.python.org Filter messages from this mailing list
signed-by python.org
unsubscribe Unsubscribe from this mailing-list

hide details 10:59 AM (2 hours ago)

I wonder if there are many people here who don’t use some kind of "easy_install" package for package management in their Python / virtualenv installations? I propose to include at least one such package that is capable to auto-update itself in Python 2.7

C:\~env\Python27>python.exe -m easy_install
C:\~env\Python27\python.exe: No module named easy_install

C:\~env\Python27>python.exe -m pip
C:\~env\Python27\python.exe: No module named pip

It bugs me when I have to troubleshoot things on yet another machine that doesn’t have some kind of ’setuptools’ installed. Or when I have to test some bug in my package on different Python version with a clean install and need some dependencies.

from Tarek Ziadé <ziade.tarek@gmail.com>
to anatoly techtonik <techtonik@gmail.com>
cc distutils-sig <distutils-sig@python.org>, Python-Dev <python-dev@python.org>
date Wed, Mar 24, 2010 at 11:26 AM
subject Re: [Python-Dev] At least one package management tool for 2.7
mailing list python-dev.python.org Filter messages from this mailing list

We are working on distutils2 right now to improve the situation, and Ian has proposed to work on the possible inclusion of virtualenv in the stdlib as well.

I’ll talk for distutils2 :

The plan is to provide a distutils2 standalone version that can be installed from 2.4 to 3.x, and that will provide a basic installer/uninstaller via -m.

Distutils2 is planned to be reintegrated in the stdlib in Python 3.3, and my goal is to release it when Python 2.7 final is released.

The open question is: do we want to include a full installer that takes care of installing / removing dependencies as well ?

I think not. Pip already provides this feature on the top of distutils (and distutils2 later I guess) and is not hard to install on the top of Python.
But the "auto-update" story seems interesting, can you expand on this?

Tarek

from anatoly techtonik <techtonik@gmail.com>
to Tarek Ziadé <ziade.tarek@gmail.com>
cc distutils-sig <distutils-sig@python.org>, Python-Dev <python-dev@python.org>
date Wed, Mar 24, 2010 at 12:20 PM
subject Re: [Python-Dev] At least one package management tool for 2.7
mailing list python-dev.python.org Filter messages from this mailing list

On Wed, Mar 24, 2010 at 12:26 PM, Tarek Ziadé <ziade.tarek@gmail.com> wrote:
> Distutils2 is planned to be reintegrated in the stdlib in Python 3.3,
> and my goal is to release it when Python 2.7 final is released.

Does that mean "after" Python 2.7, because I meant it to be "before" or at least "with"?

> The open question is: do we want to include a full installer that
> takes care of installing / removing dependencies as well?

If there is a risk to get nothing at all in 2.7 distribution, because it just won’t be ready/accepted by that time, then I it is certainly optional.

> But the "auto-update" story seems interesting, can you expand on this?

Sure. Package management tool should have an ability to update itself when required regardless of Python release. For example:

    python.exe -m easy_install setuptools

This will get you new version of 'setuptools' and 'easy_install' module from it automatically. You do not need to install new version of 'setuptools' manually or copy files from SVN if you want to see fixes before next Python release. The stuff you would likely need to do with distutils bugs, which I personally find awkward.

Package management is orthogonal to Python releases, and it is more oriented at Python users who don't like to wait or follow PEPs. That’s why package management tool such as 'easy_install' has shorter development cycle, and it should faster react to user feedback. What can be one of the reasons that no package management tool is included with Python.

In various README you may often see "requires setuptools > 0.6c9" or similar. I can’t see why package management tool can not detect this dependency and propose to update itself.

If it is impossible to ship the whole package management system then at least Python distribution may carry small bootstrap script for it. When user tries to execute package management tools, it warns him that these are not installed and gives a hint where to get them

> python -m easy_install bla-bla-bla
Error: easy_install module is not shipped with this Python release.
Please execute the following command to install the latest version.

```bash
python -m easy_bootstrap
```

cpython packaging module

cpython packaging module news  See Also:


cpython packaging module news, march 29 2011

de Tarek Ziadé <ziade.tarek@gmail.com>

heure de l’expéditeur  Envoyé à 10:13 (GMT+02:00). Heure locale : 14:02.

répondre à  the-fellowship-of-the-packaging@googlegroups.com

à  the-fellowship-of-the-packaging@googlegroups.com

cc  Doug Hellmann <doug.hellmann@gmail.com>

date  29 mars 2011 10:13

objet  Re: Trying out “Documentation Driven Development” for pysetup

liste de diffusion  <the-fellowship-of-the-packaging@googlegroups.com> Filtre les messages de cette
envoyé par  googlegroups.com

On Tue, Mar 29, 2011 at 3:09 AM, Kelsey Hightower
<kelsey.hightower@gmail.com> wrote:

> I am trying out the "Documentation Driven Development" process while
> working on the pysetup script.
>
> My goal is to document how pysetup is supposed to work in the form of
> a tutorial. This effort will hopefully help add focus and a clear
> scope while we work on the pysetup script. The concept seems pretty
> cool, and we may end up with a good tutorial for pysetup.
>
>
> Awesome !

> I am hosting the tutorial on bitbucket:
> https://bitbucket.org/khightower/pysetup-tutorial

that leads me to another task we need to do: merge back the docs/ from distutils in cpython

so I’ll go ahead and create cpython/Doc/packaging, and I think you should add your stuff there directly

Cheers

de Tarek Ziadé <ziade.tarek@gmail.com>

répondre à  the-fellowship-of-the-packaging@googlegroups.com

à  The fellowship of the packaging <the-fellowship-of-the-packaging@googlegroups.com>

cc  Doug Hellmann <doug.hellmann@gmail.com>,

Georg Brandl <g.brandl@gmx.net>

date  29 mars 2011 10:37
Hey,

I’ve pushed back the Distutils2 doc into Doc/packaging in the cpython trunk at https://bitbucket.org/tarek/cpython/, and what we need to do next is:

1. remove all distutils2 occurrences
2. start to re-work the doc that lives outside the packaging folder, that refers to installing/building stuff
3. reorganize the doc into sections:
   - for end-user
   - for developers
   - for package managers
4. see if we want to change the Python build itself so it uses packaging instead of distutils1

This is a big work – I am wondering if we couldn’t do a doc sprint for this.

Thoughts?

Cheers Tarek

cpython packaging module news, march 30 2011

de  Kelsey Hightower <kelsey.hightower@gmail.com>
heure de l’expéditeur  Envoyé à 03:38 (GMT-04:00). Heure locale : 03:42.
répondre à  the-fellowship-of-the-packaging@googlegroups.com
à  the-fellowship-of-the-packaging@googlegroups.com
cc  Mathieu Leduc-Hamel <marrakis@gmail.com>,
Yannick Gingras <ygingras@ygingras.net>,
Tarek Ziade <ziade.tarek@gmail.com>,
Doug Hellmann <doug.hellmann@gmail.com>,
Georg Brandl <g.brandl@gmx.net>
date 30 mars 2011 03:38

The goal of this sprint is to get the “packaging” docs in working order. I’m going to setup a sprint page and update the list with the details.

I am thinking we should focus on the following tasks:

1. remove all distutils2 occurrences and replace them with packaging.
2. start to re-work the doc that lives outside the packaging folder, that refers to installing/building stuff
3. reorganize the doc into sections:
   - for end-user
   - for developers
   - for package managers

I think we should focus on these 3 objectives, and if time permits work on item number 4 (use packaging to build Python in-place of distutils)

de  Kelsey Hightower <kelsey.hightower@gmail.com>
répondre à the-fellowship-of-the-packaging@googlegroups.com
à The fellowship of the packaging <the-fellowship-of-the-packaging@googlegroups.com>
date 30 mars 2011 05:03
objet Re: Documenting Packaging
liste de diffusion <the-fellowship-of-the-packaging@googlegroups.com> Filtrer les messages de cette
I have setup a sprint page here:

http://wiki.python.org/moin/Packaging/Sprints/DocumentationSprint

---

## Contents

- Python snippets
  - renumber_sorted_exportdef.py
  - Password

---

### renumber_sorted_exportdef.py

```python
""
Python2.6 script to renumber and sort the functions from a visual C++ export file.
""

list_functions=[]

# read the list of functions
with open('export.def', 'r') as f:
    for line in f.readlines():
        line2 = line.split('@')
        if (len(line2)>=2):
            function=line2[0].strip()
            list_functions += [function]

# to suppress the possible duplicates
set_functions=set(list_functions)

# sorting the functions
list_functions_sorted=sorted(set_functions,reverse=False)

# write the sorted functions
with open('renumber_sorted_export.def', 'w') as f:
    for i, function in enumerate(list_functions_sorted):
        s="	{0:65}@{1}
        .format(function,i+1)
        f.write(s)
```

---

### renumber_sorted_exportdef.py

```python
# -*- coding: ISO-8859-1 -*-
#
# $Id: build_CL1356T_SDK_C_Doc_msi.py 2991 2010-04-30 14:03:23Z pvergain $

from __future__ import print_function

import subprocess
from subprocess import Popen as subprocess_Popen

def os_system(command):
    """Execute a command""
    proc = subprocess_Popen(command, shell=True, stdout=subprocess.PIPE,)
    stdout_value = proc.communicate()[0]
    _logger.info(stdout_value)
```

---

2.17. Languages 685
def build_msi(wsiname = 'CL1356T_SDK_C_Doc_Installer.wsi',msiname='CL1356T_SDK_C_Doc_Installer.msi'):
    cmd = 'WFWI.EXE %s /c /o "%s" /s' % (wsiname, msiname)
    os_system(cmd)
    #_logger.info('Building of %s successful' % app.msiname)

build_msi()

print ("OK build_msi")

Password

Python password generator  See Also:
http://code.activestate.com/recipes/578169-extremely-strong-password-generator/

This is an improved version of the generator written by Isendrak Skatasmid - simple-password-generator. I have introduced few changes in the algorithm.

Also, the entire set of special characters is not being used. Sometimes passwords that start with special-characters or numbers are generated. I am not sure whether that is a bad thing.

A few links that I had used to test the password strength are:

- How Secure Is My Password
- Password Meter
- Password Strength Checker

You are welcome to comment/criticize/suggest improvements for the implementation.

Télécharger le fichier password_generator.py

```python
import urandom

char_set = {'small': 'abcdefghijklmnopqrstuvwxyz',
            'nums': '0123456789',
            'big': 'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
            'special': '^!@#$%^&*()=?{[\]}+~#-_.:,;<>|'}

def generate_pass(length=16):
    """Function to generate a password""
    password = ''
    current_char_set = ''

    while len(password) < length:
        for i in char_set.iterkeys():
            current_char_set = char_set[i]
            a_char = urandom(1)
            if a_char in current_char_set and a_char not in password:
                if check_prev_char(password, current_char_set):
                    continue
                else:
                    password += a_char
            return password
```
def check_prev_char(password, char_set):
    
    Function to ensure that there are no consecutive 
    UPPERCASE/lowercase/numbers/special-characters.
    
    index = len(password)
    if index == 0:
        return False
    else:
        prev_char = password[index - 1]
        if prev_char in char_set:
            return True
        else:
            return False

if __name__ == '__main__':
    print generate_pass()

Analyse de code python

Analyse de code avec pylint Je l’utilise récemment. Indispensable :)

Pylint is a Python source code analyzer which looks for programming errors, helps enforcing a coding standard and
sniffs for some code smells (as defined in Martin Fowler’s Refactoring book).

Pylint can be seen as another PyChecker since nearly all tests you can do with PyChecker can also be done with Pylint.

However, Pylint offers some more features, like checking length of lines of code, checking if variable names are well-
formed according to your coding standard, or checking if declared interfaces are truly implemented, and much more.
Additionally, it is possible to write plugins to add your own checks.

Python string modules

Python parse module See Also:

• http://pypi.python.org/pypi/parse/

• http://docs.python.org/py3k/library/stdtypes.html#string-methods

Parse strings using a specification based on the Python format() syntax.

parse() is the opposite of format()

Basic usage:

>>> from parse import * # only exports parse() and compile()
>>> parse("It’s {}, I love it!", "It’s spam, I love it!")
<Result (’spam’,) ()>
>>> p = compile("It’s {}, I love it!")
>>> print p
<Parser "It’s {}, I love it!">
>>> p.parse("It’s spam, I love it!")
<Result (’spam’,) ()>
Python system modules

**psutil module**  See Also:


**Summary**  psutil is a module providing an interface for retrieving information on all running processes and system utilization (CPU, disk, memory) in a portable way by using Python, implementing many functionalities offered by command line tools such as:

- ps
- top
- df
- kill
- free
- lsof
- netstat
- ifconfig
- nice
- ionice
- iostat
- iotop
- uptime
- tty

It currently supports Linux, Windows, OSX and FreeBSD both 32-bit and 64-bit with Python versions from 2.4 to 3.3 by using a single code base

**pyrasite module**  See Also:

- [http://pypi.python.org/pypi/pyrasite-gui](http://pypi.python.org/pypi/pyrasite-gui)

**Summary**  A graphical interface for interacting with running Python processes

Python tests modules

**Python texttests**  See Also:


TextTest is an open source tool for text-based functional testing. This means running a program in lots of different ways from the command line, and using the text output produced as a means of controlling the behaviour of that application.

Also on this site are StoryText and CaptureMock, which aid in GUI testing and mocking respectively. They do not depend on TextTest in any way but are well suited to being used together with it.
Python time modules

Python sanetime  See Also:
  • http://readthedocs.org/docs/sanetime/en/latest/
  • http://pypi.python.org/pypi/sanetime/4.0.9

Python timelines  timespan and scheduling helpers for Python
See Also:
  • http://pypi.python.org/pypi/timelines/0.2

```python
>>> import datetime
>>> from timelines import timespan, timelayer
```

A timespan object has a start time and an end time. It can be created either by specifying a start time and an elapsed time, or by specifying both start and end times:

```python
>>> span1 = timespan(datetime.datetime(1984, 11, 26), datetime.timedelta(1))
```

```python
>>> span2 = timespan(datetime.datetime(1984, 11, 26) + datetime.timedelta(2), datetime.datetime(1984, 11, 26) + datetime.timedelta(2, 50))
```

```python
>>> span1.start
datetime.datetime(1984, 11, 26, 0, 0)
```

```python
>>> span1.elapsed
datetime.timedelta(1)
```

python translate to other languages  See Also:

python to javascript  See Also:

pyjaco  See Also:
  • http://pyjaco.org/about
  • http://pyjaco.org/download

About the project  Pyjaco is a Python-to-Javascript compiler. It enables compilation of standard Python code into a javascript equivalent.

This javascript code can then be run in a browser, or anywhere else that javascript is used.
Why not just write javascript? Using Python through Pyjaco, rather than writing javascript directly, has several advantages:

You don’t have to learn a new language The semantic differences between Python and Javascript are huge. To name a few differences, javascript has no real classes, support prototypical rather than classical inheritance, does not support any hooks like __getattr__ and the like, and considers an empty list to be True, rather than False.

Using pyjaco, you only need to know the Python semantics, which you already do! Pyjaco translates your python code into javascript code that performs the same operations. And the classes are mapped too, even for numbers, strings and lists.

Spend less time debugging

Javascript, JScript or ECMAScript? Version 1.5 or 1.8? Or something else entirely? Using javascript can be quite complicated in itself, and it is not helped by the fact that each browser supports different features.

Because pyjaco relies only on a minimal subset of javascript, it can be guaranteed to run on each major web platform. The extensive test suite has nearly-full code coverage, and has achieved a success rate of 100% on Chrome, Firefox and Opera. We are still working on Internet Explorer, since it is broken in so many ways. When Internet Explorer support is finished, we have done the hard work so you do not have to.

Be more productive

Let’s face it, there is a reason people love Python. It is immensily powerful, and has an excellent standard library. Javascript lacks almost all the features people love in Python, and has a much less comprehensive standard library. With pyjaco, you don’t have to worry about that.

Just write your code using Python and the Python standard library, and let pyjaco do the hard work for you.

python VCS modules Python modules for Version Control Systems.

github2 module See Also:

- [http://packages.python.org/github2/](http://packages.python.org/github2/)
- [github2 using sphinx](http://packages.python.org/github2/)

This is a Python library implementing all of the features available in version 2 of the Github API.

You should read the developer documentation for the Github API first.

**Git repository** [https://github.com/ask/python-github2/](https://github.com/ask/python-github2/)

**Issue tracker** [https://github.com/ask/python-github2/issues/](https://github.com/ask/python-github2/issues/)

**Contributors** [https://github.com/ask/python-github2/contributors/](https://github.com/ask/python-github2/contributors/)

Quickstart Once github2 is installed we can open an interactive Python session and perform some basic tasks to familiarise ourselves with the package.

Create an unauthenticated client object:

```python
>>> github = Github()
```

**Note:** Creating an unauthenticated client object means we can play with the API without fear of creating or deleting data on our account.
See how many followers the **github2** project has:

```python
>>> len(github.repos.watchers("ask/python-github2"))
129
```

Read the description of the **python-github2** project:

```python
>>> repo = github.repos.show("ask/python-github2")
>>> repo.description
```

We can take advantage of Python’s `dir()` to explore the package a little more:

```python
>>> filter(lambda s: not s.startswith("_") , dir(github.users))
['domain', 'follow', 'followers', 'following', 'get_value', 'get_values',
 'make_request', 'request', 'search', 'search_by_email', 'show',
 'unfollow']
```

For more information on specific functionality see api/index’, and the copious examples contained within.

- pair: Python VCS; github3

**github3 module**  See Also:

https://github.com/devjones/py_github3

This is ultimately intended to be a python library for the Github v3 API.

Currently, the library only implements commits to Github from local files.

However, as time permits I may implement a complete implementation of the new Github v3 API for python as there doesn’t seem to be one available.

**python web modules**

**python web modules**  See Also:

https://fedorahosted.org/suds/

Suds is a lightweight SOAP python client for consuming Web Services.

**Python on windows**  See Also:

Introduction  Windows Management Instrumentation (WMI) is Microsoft’s implementation of Web-Based Enterprise Management (WBEM), an industry initiative to provide a Common Information Model (CIM) for pretty much any information about a computer system.

The Python WMI module is a lightweight wrapper on top of the pywin32 extensions, and hides some of the messy plumbing needed to get Python to talk to the WMI API. It’s pure Python and has been tested against all versions of Python from 2.4 to 3.2. It should work with any recent version of pywin32.

See Also:

- http://timgolden.me.uk/python/wmi/index.html
- http://timgolden.me.uk/python/wmi/tutorial.html
- http://timgolden.me.uk/python/wmi/cookbook.html

python wmi registry  See Also:

- http://timgolden.me.uk/python/wmi/cookbook.html#list-registry-keys
- http://timgolden.me.uk/python-on-windows/programming-areas/registry.html
- http://code.google.com/p/python-on-windows-docs/source/browse/#svn/trunk/programming-areas/registry

delete the registry  The requirement: To delete one or more keys from the registry, including all their values.

The standard registry API, exposed in Python by the _winreg module, only allows keys to be deleted which have no subkeys. The Windows shell API does offer a recursive key delete but that function isn’t exposed neither by the Python stdlib nor by the pywin32 extensions.

The first example below uses the _winreg module to delete keys one at a time. For simplicity, this example knows the layout of subtrees it needs to delete. It would be possible to adapt the registry walker example to go depth first and delete subtrees from the bottom up. The second example uses ctypes to invoke the shell API function which will delete an entire tree in one go.

See Also:

http://timgolden.me.uk/python-on-windows/programming-areas/registry/delete-the-registry.html

Deleting with _winreg
import _winreg

root = _winreg.HKEY_CURRENT_USER
keypaths = [
    r"Software\PySoft\PyApp",
    r"Software\PySoft\PyThing",
    r"Software\PySoft\PyTool",
    r"Software\PySoft",
]

for keypath in keypaths:
    keys = keypath.split (r\\)
    mainkey, subkey = r\\.join (keys[:-1]), keys[-1]
    key = _winreg.OpenKey (root, mainkey, 0, _winreg.KEY_ALL_ACCESS)
    _winreg.DeleteKey (key, subkey)
    print "Deleted", keypath

Deleting with Shell API

import _winreg
import ctypes
from ctypes import wintypes
import win32api
import win32con

shlwapi = ctypes.windll.shlwapi
SHDeleteKey = shlwapi.SHDeleteKeyW

root, keypath = wintypes.HKEY (_winreg.HKEY_CURRENT_USER), ur"Software\PySoft"
result = SHDeleteKey (root, keypath)
if result:
    raise RuntimeError (
        win32api.FormatMessageW (win32con.FORMAT_MESSAGE_FROM_SYSTEM, None, result, 0, [])
    )

serial ports   See Also:
http://www.activexperts.com/admin/scripts/wmi/python/0358/

Python registry on windows

Recipe 476229: YARW - Yet Another Registry Wrapper (Python)

pywin32   See Also:

- http://sourceforge.net/projects/pywin32/
- http://sourceforge.net/users/mhammond

python wrappers modules   See Also:

python c modules   See Also:
Development tools, Release 2012.06.18

**python nb_inline module**  See Also:

http://pages.cs.wisc.edu/~johnl/np_inline/doc-0.3.html

**Overview**  The np_inline module was written as a simplified replacement for scipy.weave.inline for numeric computations on numpy arrays. The module is implemented in a single file containing less than 500 lines of code, including the C-file template, docstrings, comments and white space.

One advantage np_inline has over weave.inline is that it works properly in a multiprocessing environment because it uses a multiprocessing.Lock object to protect the module production code.

**python weave module**  See Also:

- http://www.scipy.org/Weave

**Overview**  The weave package allows the inclusion of C/C++ within Python code and is useful in accelerating Python code.

Weave is a subpackage of scipy. (I.e. you have it already if you installed SciPy) Alternatively, you can check-out and install weave separately using

```
svn co http://svn.scipy.org/svn/scipy/trunk/scipy/weave weave
cd weave
sudo python setup.py install
```

- Current documentation (which is still being updated to reflect the move to NumPy) can be seen here.
- PerformancePython: A comparison of various ways to improve the performance of Python code using Numeric, weave, Pyrex, Psyco and Fortran (f2py) for solving Laplace’s equation. These are compared with code written in C++.

**python fortran modules**  See Also:

**python f2pypy module**  See Also:


The above is talk and hand-waving. Code’s also good. There was a PyPy sprint this week and I decided to join in for a few days and prototype an idea I’ve been thinking about: f2pypy. It’s a variation of f2py which generates Python ctypes bindings which PyPy could use to talk with shared libraries implemented in Fortran.

At the end of several days of work, I got f2pypy to generate a Python module based on the “fblas.pyf” code from SciPy. I could import that library in CPython and (for the few functions I tested) get answers which matched the fblas module in SciPy. I could also use pypy to call some of the functions, but PyPy’s “numpy” implementation is not mature enough. Its array objects don’t support the ctypes interface, so I was unable to call out to the shared library. I could only call the scalar-based functions.

The code is definitely incomplete. Even my CPython-based tests fail some of the the “test_blas.py” from SciPy (I don’t implement “cblas” and I think one of the tests depends on Fortran order instead of C order.) It’s a proof-of-concept which shows that this approach is definitely viable, and it shows some of the difficulties in the approach.

My point though is that it opens new possibilities which aren’t available in NumPy. For example, suppose you want to use one of the BLAS functions in your code. Every Mac includes a copy of BLAS as a built-in library. Instead of making people install SciPy, what about shipping the ctypes module description instead, and using that interface? You can ship pure Python code and still take advantage of platform-optimized libraries!
I earlier highlighted the performance problems in CPython’s ctypes interface. But this is PyPy. They already have cross-module optimizations for Python calling Python. There’s no reason why those can’t apply to ctypes-based functions. (Or perhaps it’s already there? I’ve not tested that.)

**Python people**

**Guido Van Rossum**  See Also:

- [http://www.python.org/~guido/](http://www.python.org/~guido/)
- [http://neopythonic.blogspot.com/](http://neopythonic.blogspot.com/)
- [https://plus.google.com/115212051037621986145](https://plus.google.com/115212051037621986145)

Guido van Rossum est un développeur néerlandais, connu pour être le créateur et leader du projet du langage de programmation Python. Au sein de la communauté pythonienne, il est considéré comme un Benevolent Dictator for Life (« dictateur bienveillant à vie »), ce qui signifie qu’il continue à suivre le développement de Python et qu’il prend des décisions lorsque c’est nécessaire.

**Georg Brandl**  See Also:

- [http://www.pocoo.org/team/#georg-brandl](http://www.pocoo.org/team/#georg-brandl)
- [https://twitter.com/#!/birkenfeld](https://twitter.com/#!/birkenfeld)

Georg Brandl is a Python core developer since 2005, and cares for its documentation at docs.python.org. He is blogging on pythonic.pocoo.org.

His IRC nickname is birkenfeld, and you can contact him via email at georg@python.org.

Follow Georg on twitter: @birkenfeld

**python peps**

See Also:

- [http://www.python.org/dev/peps/](http://www.python.org/dev/peps/)

**pep 0008 Style Guide for Python Code**  See Also:

- [http://www.python.org/dev/peps/pep-0008/](http://www.python.org/dev/peps/pep-0008/)

This document gives coding conventions for the Python code comprising the standard library in the main Python distribution.

Please see the companion informational PEP describing style guidelines for the C code in the C implementation of Python.

This document was adapted from Guido’s original Python Style Guide essay[, with some additions from Barry’s style guide.

Where there’s conflict, Guido’s style rules for the purposes of this PEP.

This PEP may still be incomplete (in fact, it may never be finished <wink>).
pep 0370 Per user site-packages directory See Also:

- http://www.python.org/dev/peps/pep-0370/

Abstract This PEP proposes a new a per user site-packages directory to allow users the local installation of Python packages in their home directory.

Rationale Current Python versions don’t have a unified way to install packages into the home directory of a user (except for Mac Framework builds). Users are either forced to ask the system administrator to install or update a package for them or to use one of the many workarounds like Virtual Python [1], Working Env [2] or Virtual Env [3]. It’s not the goal of the PEP to replace the tools or to implement isolated installations of Python. It only implements the most common use case of an additional site-packages directory for each user.

The feature can’t be implemented using the environment variable PYTHONPATH.

The env var just inserts a new directory to the beginning of sys.path but it doesn’t parse the pth files in the directory.

A full blown site-packages path is required for several applications and Python eggs.

pep 0389 argparse - New Command Line Parsing Module See Also:

- http://www.python.org/dev/peps/pep-0389/
- python argparse
- plac

Acceptance This PEP was approved by Guido on python-dev on February 21, 2010.

Abstract This PEP proposes inclusion of the argparse module in the Python standard library in Python 2.7 and 3.2.

Motivation The argparse module is a command line parsing library which provides more functionality than the existing command line parsing modules in the standard library, getopt and optparse. It includes support for positional arguments (not just options), subcommands, required options, options syntaxes like “/f” and “+rgb”, zero-or-more and one-or-more style arguments, and many other features the other two lack.

The argparse module is also already a popular third-party replacement for these modules. It is used in projects like IPython (the Scipy Python shell) , is included in Debian testing and unstable, and since 2007 has had various requests for its inclusion in the standard library .

This popularity suggests it may be a valuable addition to the Python libraries.

pep 0394 The “python” Command on Unix-Like Systems See Also:

- http://www.python.org/dev/peps/pep-0394/
- python argparse
- plac
Abstract  This PEP provides a convention to ensure that Python scripts can continue to be portable across nix systems, regardless of the default version of the Python interpreter (i.e. the version invoked by the python command).

- python2 will refer to some version of Python 2.x
- python3 will refer to some version of Python 3.x
- python should refer to the same target as python2 but may refer to python3 on some bleeding edge distributions

Recommendation  Unix-like software distributions (including systems like Mac OS X and Cygwin) should install the python2 command into the default path whenever a version of the Python 2 interpreter is installed, and the same for python3 and the Python 3 interpreter.

When invoked, python2 should run some version of the Python 2 interpreter, and python3 should run some version of the Python 3 interpreter.

Similarly, the more general python command should be installed whenever any version of Python is installed and should invoke the same version of Python as either python2 or python3.

For the time being, it is recommended that python should refer to python2 (however, some distributions have already chosen otherwise; see the Rationale and Migration Notes below).

The Python 2.x idle, pydoc, and python-config commands should likewise be available as idle2, pydoc2, and python2-config, with the original commands invoking these versions by default, but possibly invoking the Python 3.x versions instead if configured to do so by the system administrator.

In order to tolerate differences across platforms, all new code that needs to invoke the Python interpreter should not specify python, but rather should specify either python2 or python3 (or the more specific python2.x and python3.x versions; see the Migration Notes).

This distinction should be made in shebangs, when invoking from a shell script, when invoking via the system() call, or when invoking in any other context.

One exception to this is scripts that are deliberately written to be source compatible with both Python 2.x and 3.x. Such scripts may continue to use python on their shebang line without affecting their portability.

When reinvoking the interpreter from a Python script, querying sys.executable to avoid hardcoded assumptions regarding the interpreter location remains the preferred approach.

These recommendations are the outcome of the relevant python-dev discussions in March and July 2011 and February 2012.

Impact on PYTHON* Environment Variables  The choice of target for the python command implicitly affects a distribution's expected interpretation of the various Python related environment variables.

The use of .pth files in the relevant site-packages folder, the “per-user site packages” feature (see python -m site) or more flexible tools such as virtualenv are all more tolerant of the presence of multiple versions of Python on a system than the direct use of PYTHONPATH.
Abstract  This PEP proposes to add to Python a mechanism for lightweight “virtual environments” with their own site directories, optionally isolated from system site directories.

Each virtual environment has its own Python binary (allowing creation of environments with various Python versions) and can have its own independent set of installed Python packages in its site directories, but shares the standard library with the base installed Python.

Abstract  This document proposes the reintegration of an explicit unicode literal from Python 2.x to the Python 3.x language specification, in order to reduce the volume of changes needed when porting Unicode-aware Python 2 applications to Python 3.

Abstract  This PEP proposes to add time.get_clock_info(name), time.monotonic(), time.perf_counter() and time.process_time() functions to Python 3.3.

Abstract  Namespace packages are a mechanism for splitting a single Python package across multiple directories on disk. In current Python versions, an algorithm to compute the packages __path__ must be formulated.

With the enhancement proposed here, the import machinery itself will construct the list of directories that make up the package.

This PEP builds upon previous work, documented in PEP 382 and PEP 402.

Those PEPs have since been rejected in favor of this one. An implementation of this PEP is at
PEP 3148 – futures - execute computations asynchronously  See Also:

- http://www.python.org/dev/peps/pep-3148/
- http://www.dalkescientific.com/writings/diary/archive/2012/01/19/concurrent.futures.html
- python Multithreading libraries

Abstract  This PEP proposes a design for a package that facilitates the evaluation of callables using threads and processes.

Motivation  Python currently has powerful primitives to construct multi-threaded and multi-process applications but parallelizing simple operations requires a lot of work i.e. explicitly launching processes/threads, constructing a work/results queue, and waiting for completion or some other termination condition (e.g. failure, timeout).

It is also difficult to design an application with a global process/thread limit when each component invents its own parallel execution strategy.

pep 3151 Reworking the OS and IO exception hierarchy  See Also:

- http://www.python.org/dev/peps/pep-3151/

Abstract  The standard exception hierarchy is an important part of the Python language.

It has two defining qualities: it is both generic and selective.

Generic in that the same exception type can be raised - and handled - regardless of the context (for example, whether you are trying to add something to an integer, to call a string method, or to write an object on a socket, a TypeError will be raised for bad argument types).

Selective in that it allows the user to easily handle (silence, examine, process, store or encapsulate...) specific kinds of error conditions while letting other errors bubble up to higher calling contexts. For example, you can choose to catch ZeroDivisionErrors without affecting the default handling of other ArithmeticErrors (such as OverflowErrors).

This PEP proposes changes to a part of the exception hierarchy in order to better embody the qualities mentioned above: the errors related to operating system calls (OSError, IOError, mmap.error, select.error, and all their sub-classes).

pep 3155 Qualified name for classes and functions  See Also:

- http://www.python.org/dev/peps/pep-3155/

Python’s introspection facilities have long had poor support for nested classes.

Given a class object, it is impossible to know whether it was defined inside another class or at module top-level; and, if the former, it is also impossible to know in which class it was defined. While use of nested classes is often considered poor style, the only reason for them to have second class introspection support is a lousy pun.

Porting to python3

See Also:

- https://bitbucket.org/loewis/django-3k/
- http://python3porting.com/
Pep 0414  See Also:

http://www.python.org/dev/peps/pep-0414/

That’s all PEP 414 is about - lowering the friction of porting to Python 3. Is it necessary? No, there are already enough successful ports to prove that, if sufficiently motivated, porting to Python 3 is feasible with the current toolset. However, that’s the wrong question. The right question is “Does PEP 414 make porting substantially easier, by significantly reducing the volume of code that needs to change in order to attain Python 3 compatibility?”. And the answer to that question is “Absolutely.” Porting the web frameworks themselves to Python 3 is only the first step in migrating those ecosystems to Python 3, and because the web APIs exposed by those frameworks are so heavily Unicode based this is an issue that will hit pretty much every Python web app and library on the planet.

Python 3 on ubuntu  See Also:

https://wiki.ubuntu.com/Python/3

Python

Modifying Grammar/grammar and other foul acts  See Also:

http://writeonly.wordpress.com/2010/04/01/whython-python-for-people-who-hate-whitespace/

from    Gregg Lind <gregg.lind@gmail.com>
to
cc      python-dev@python.org
date    Sat, Mar 6, 2010 at 7:26 PM
subject  Re: [Python-Dev] Modifying Grammar/grammar and other foul acts
mailing list python-dev.python.org Filter messages from this mailing list

Sorry, re: question one, forgive the ill-formed question. I meant more, are the parser rules applied “first matching”. Essentially trying to confirm that the parser is “top down” or “bottom up” or whether or not it even matters.

Thanks for the tip – it seems to be exactly what I want. To make it explicit, this seems to be fuller (unix) recipe for how to make this style of debugging happen:

$ ./configure --with-pydebug
$ make
$ set PYTHONDEBUG=1
$ ./python -d  # then this shows the parsing info
Guido van Rossum <guido@python.org>
to Gregg Lind <gregg.lind@gmail.com>
cce python-dev@python.org
date Sat, Mar 6, 2010 at 10:41 PM
subject Re: [Python-Dev] Modifying Grammar/grammar and other foul acts
mailing list python-dev@python.org Filter messages from this mailing list

On Sat, Mar 6, 2010 at 10:26 AM, Gregg Lind <gregg.lind@gmail.com> wrote: > Sorry, re: question one, forgive the ill-formed question. I meant more, are > the parser rules applied “first matching”. Essentially trying to confirm > that the parser is “top down” or “bottom up” or whether or not it even > matters.

That’s not how it works at all. I can’t explain it in a few words – but any text on LL(1) parsing should clarify this. The parser uses no backtracking and a 1-token lookahead. The only unusual thing is that individual rules use a regex-like notation, but that is all converted to a DFA. If one token is not enough to know which path to take through the DFA (this may invoke another rule – but you always know which one) you’re hosed.

I suspect you’ve introduced ambiguities, though I don’t immediately see where (they could be in the combination of different rules).

Another possibility is that you may be running into problems where the parser expects a newline at the end of a suite. (FWIW since you’re not proposing a language change, this is technically off-topic for python-dev. :-)

from Gregg Lind <gregg.lind@gmail.com>
to python-dev@python.org
date Thu, Apr 1, 2010 at 5:38 AM
subject Re: [Python-Dev] Modifying Grammar/grammar and other foul acts
mailing list python-dev@python.org Filter messages from this mailing list

Thank you for the advice everyone. This seed has finally born (rotten) fruit at:

http://writeonly.wordpress.com/2010/04/01/whython-python-for-people-who-hate-whitespace/
http://bitbucket.org/gregglind/python-whython3k/

http://writeonly.wordpress.com/2010/04/01/whython-python-for-people-who-hate-whitespace/ More seriously:

- reading the Dragon Book Aho86 gives a person dangerous ideas
- good excuse to deep dive into the python interpreter source code and the AST, dis modules
- finally wanted to learn GDB and python -d debug mode
- humming trolls is fun
- for education, the whitespace thing really can cause problems. When copying code out of books into IDLE or IPython, there are corner cases when it terminates blocks “too early”, confusing new learners.
- preparation for the “Python Spring Cleaning” project, to see how hard it is to get and modify source, write a PEP, raise bug ideas, talk in irc, etc.
- since this is unlikely to ever be adopted by Python (I hope!), it will remain a useful exercise, unlike other “bugs” which get fixed once and for all

PySide for Python 3?

from Mark Summerfield <list@qtrac.plus.com>
reply-to Mark Summerfield <mark@qtrac.eu>
to pyside@lists.openbossa.org
date Fri, Apr 9, 2010 at 4:19 PM
Hi,

I am hoping that at some point this year there will be a PySide for Python 3?

If there is, will it support PyQt’s API 2 (which IMO is nicer than the API 1 provided by PySide and PyQt for Python 2)?

I’d really like to convert all my PyQt apps to Python 3, and ideally I’d like them all to work with both PyQt and PySide with as few changes as possible to account for the differences (i.e., just having an import that tries for one and falls back to the other).

– Mark Summerfield, Qtrac Ltd, www.qtrac.eu C++, Python, Qt, PyQt - training and consultancy “Advanced Qt Programming” - ISBN 0321635906

That’s definitely on our roadmap, but with the Shiboken schedules already slipping, it’s a bit premature to say when the core team will be able to allocate time for it.

Adding Python 3 CPython API code generation support to Shiboken sounds like a perfect project for some enterprising Python hacker. ;-):

If there is, will it support PyQt’s API 2 (which IMO is nicer than the API 1 provided by PySide and PyQt for Python 2)?

I have to admit I’ve had very limited experience on the PyQt’s API v2 other than reading through the relevant reference pages. They’d generally seem like the right thing (apart from the implementation-specific way of selecting the API version) but since these changes would imply making slightly incompatible API modifications between PySide versions, I’d really prefer making an reasoned decision using the PSEP process.

Also, with my very limited experience on the API changes, I don’t feel comfortable about being the champion for that PSEP. Anyone willing to take the ball? It shouldn’t be too much work to write the proposal:

I’d really like to convert all my PyQt apps to Python 3, and ideally I’d like them all to work with both PyQt and PySide with as few changes as possible to account for the differences (i.e., just having an import that tries for one and falls back to the other).

Yep, that’d definitely be a preferable goal.

Cheers,

PyQt’s API v2  See Also:

http://www.qtrac.eu/pyqtbook.html#pyqtapis

About PyQt’s APIs—From PyQt4.6, PyQt has two APIs, API#1 (the original), and API#2 (new). API#2 is more Pythonic and eliminates QString and QVariant, and is a bit nicer to use. API#1 remains best for those using PyQt to prototype C++/Qt applications. API#1 is the default for PyQt4.x with Python 2.x, and for PyQt4.0-4.5 with Python 3.x, and is the API used by PySide. API#2 is the default for PyQt4.6+ with Python 3.x.
The book, and all its examples use API#1, so they don’t work with PyQt4.6+ with Python 3.x—but they do work with PyQt4.x with Python 2.x, and for PyQt4.0-4.5 with Python 3.x.

**Warning:** Although I personally prefer API#2, I am not planning to port the examples to use it, since it would make the examples so far out of sync with the book as to be confusing.

---

**[Python-Dev] [RELEASED] 2.7 beta 1**

from Guido van Rossum <guido@python.org>

from Guido van Rossum <guido@python.org>
to Antoine Pitrou <solipsis@pitrou.net>
cc python-dev@python.org
date Sun, Apr 11, 2010 at 9:36 PM
subject Re: [Python-Dev] [RELEASED] 2.7 beta 1

On Sat, Apr 10, 2010 at 11:57 AM, Antoine Pitrou <solipsis@pitrou.net> wrote: > Benjamin Peterson <benjamin <att> python.org> writes: >>> On behalf of the Python development team, I’m merry to announce the first beta >> release of Python 2.7. >>> Congratulations, and thanks for your patience :)

Congratulations indeed!

Let me use this occasion to point out what an awesome community python-dev is. Not only do we have an great release manager (kudos to Benjamin for taking on this thankless job of herding cats), but the whole developer community has been contributing to some excellent releases. This makes me very happy and confident that Python will continue to thrive for many, many years, with or without my direct involvement in the details. While rumors of my retirement are greatly exaggerated, I am very glad to be able to leave the details to the community.

Keep them coming, folks!

––Guido van Rossum (python.org/~guido)

**Python stand-alone**

**PyInstaller**   See Also:

• [http://www.pyinstaller.org/](http://www.pyinstaller.org/)

PyInstaller is a program that converts (packages) Python programs into stand-alone executables, under Windows, Linux, Mac OS X, Solaris and AIX. Its main advantages over similar tools are that PyInstaller works with any version of Python since 2.2, it builds smaller executables thanks to transparent compression, it is fully multi-platform, and use the OS support to load the dynamic libraries, thus ensuring full compatibility.

**python to**

**python to C++**   See Also:

[https://github.com/serge-sans-paille/pythran](https://github.com/serge-sans-paille/pythran)
What is it? Pythran is a python to C++ compiler for a subset of the python language. It takes a python module annotated with a few interface description and turns it into a native python module with the same interface, but (hopefully) faster.

python tools

See Also:

- http://www.python.org/dev/peps/

ipython See Also:

- http://ipython.org/index.html

ipython versions

ipython 0.1.2 (december 2011) See Also:

- http://ipython.org/ipython-doc/rel-0.12/whatsnew/version0.12.html

We’re pleased to announce the release of IPython 0.12, a major new version.

The highlight of this release is the IPython Notebook, an interactive Python interface running in the browser.

Download it now, or read more about what’s new.

IPython 0.12 contains several major new features, as well as a large amount of bug and regression fixes. The 0.11 release brought with it a lot of new functionality and major refactorings of the codebase; by and large this has proven to be a success as the number of contributions to the project has increased dramatically, proving that the code is now much more approachable. But in the refactoring inevitably some bugs were introduced, and we have also squashed many of those as well as recovered some functionality that had been temporarily disabled due to the API changes.

The following major new features appear in this version.

An interactive browser-based Notebook with rich media support A powerful new interface puts IPython in your browser. You can start it with the command ipython notebook:

This new interface maintains all the features of IPython you are used to, as it is a new client that communicates with the same IPython kernels used by the terminal and Qt console. But the web notebook provides for a different workflow where you can integrate, along with code execution, also text, mathematical expressions, graphics, video, and virtually any content that a modern browser is capable of displaying.

You can save your work sessions as documents that retain all these elements and which can be version controlled, emailed to colleagues or saved as HTML or PDF files for printing or publishing statically on the web. The internal storage format is a JSON file that can be easily manipulated for manual exporting to other formats.
This Notebook is a major milestone for IPython, as for years we have tried to build this kind of system. We were inspired originally by the excellent implementation in Mathematica, we made a number of attempts using older technologies in earlier Summer of Code projects in 2005 (both students and Robert Kern developed early prototypes), and in recent years we have seen the excellent implementation offered by the Sage <http://sagemath.org> system.

But we continued to work on something that would be consistent with the rest of IPython’s design, and it is clear now that the effort was worth it: based on the ZeroMQ communications architecture introduced in version 0.11, the notebook can now retain 100% of the features of the real IPython.

But it can also provide the rich media support and high quality Javascript libraries that were not available in browsers even one or two years ago (such as high-quality mathematical rendering or built-in video).

The notebook has too many useful and important features to describe in these release notes; our documentation now contains a directory called examples/notebooks with several notebooks that illustrate various aspects of the system. You should start by reading those named 00_notebook_tour.ipynb and 01_notebook_introduction.ipynb first, and then can proceed to read the others in any order you want.

To start the notebook server, go to a directory containing the notebooks you want to open (or where you want to create new ones) and type:

```
ipython notebook
```

You can see all the relevant options with:

```
ipython notebook --help
ipython notebook --help-all    # even more
```

and just like the Qt console, you can start the notebook server with pylab support by using:
ipython notebook --pylab

for floating matplotlib windows or:

ipython notebook --pylab inline

for plotting support with automatically inlined figures. Note that it is now possible also to activate pylab support at runtime via %pylab, so you do not need to make this decision when starting the server.

**Two-process terminal console**  Based on the same architecture as the notebook and the Qt console, we also have now a terminal-based console that can connect to an external IPython kernel (the same kernels used by the Qt console or the notebook, in fact). While this client behaves almost identically to the usual IPython terminal application, this capability can be very useful to attach an interactive console to an existing kernel that was started externally. It lets you use the interactive %debug facilities in a notebook, for example (the web browser can’t interact directly with the debugger) or debug a third-party code where you may have embedded an IPython kernel.

This is also something that we have wanted for a long time, and which is a culmination (as a team effort) of the work started last year during the 2010 Google Summer of Code project.

**Tabbed QtConsole**  The QtConsole now supports starting multiple kernels in tabs, and has a menubar, so it looks and behaves more like a real application. Keyboard enthusiasts can disable the menubar with ctrl-shift-M.

![Figure 2.29: The improved Qt console for IPython, now with tabs to control multiple kernels and full menu support.](image-url)
Full Python 3 compatibility  IPython can now be installed from a single codebase on Python 2 and Python 3. The installation process for Python 3 automatically runs 2to3. The same ‘default’ profile is now used for Python 2 and 3 (the previous version had a separate ‘python3’ profile).

Standalone Kernel  The ipython kernel subcommand has been added, to allow starting a standalone kernel, that can be used with various frontends. You can then later connect a Qt console or a terminal console to this kernel by typing e.g.:

ipython qtconsole --existing

if it’s the only one running, or by passing explicitly the connection parameters (printed by the kernel at startup).

PyPy support  The terminal interface to IPython now runs under PyPy. We will continue to monitor PyPy’s progress, and hopefully before long at least we’ll be able to also run the notebook. The Qt console may take longer, as Qt is a very complex set of bindings to a huge C++ library, and that is currently the area where PyPy still lags most behind. But for everyday interactive use at the terminal, with this release and PyPy 1.7, things seem to work quite well from our admittedly limited testing.

Other important new features

• SSH Tunnels: In 0.11, the IPython.parallel Client could tunnel its connections to the Controller via ssh. Now, the QtConsole ssh tunneling, as do parallel engines.

• relaxed command-line parsing: 0.11 was released with overly-strict command-line parsing, preventing the ability to specify arguments with spaces, e.g. ipython --pylab qt or ipython --c "print ’hi’". This has been fixed, by using argparse. The new parsing is a strict superset of 0.11, so any commands in 0.11 should still work in 0.12.

• HistoryAccessor: The HistoryManager class for interacting with your IPython SQLite history database has been split, adding a parent HistoryAccessor class, so that users can write code to access and search their IPython history without being in an IPython session.

• kernel %gui and %pylab: The %gui and %pylab magics have been restored to the IPython kernel (e.g. in the qtconsole or notebook). This allows activation of pylab-mode, or eventloop integration after starting the kernel, which was unavailable in 0.11. Unlike in the terminal, this can be set only once, and cannot be changed.

• %config: A new %config magic has been added, giving easy access to the IPython configuration system at runtime.

• Multiline History: Multiline readline history has been restored to the Terminal frontend by default.

• %store: The %store magic from earlier versions has been updated and re-enabled. To autorestore stored variables on startup, specify c.StoreMagic.autorestore = True in ipython_config.py.

virtualenv  See Also:

• http://pypi.python.org/pypi/virtualenv

What It Does  virtualenv is a tool to create isolated Python environments.

The basic problem being addressed is one of dependencies and versions, and indirectly permissions.

Imagine you have an application that needs version 1 of LibFoo, but another application requires version 2. How can you use both these applications? If you install everything into /usr/lib/python2.7/site-packages (or whatever your
platform’s standard location is), it’s easy to end up in a situation where you unintentionally upgrade an application that shouldn’t be upgraded.

Or more generally, what if you want to install an application and leave it be? If an application works, any change in its libraries or the versions of those libraries can break the application.

Also, what if you can’t install packages into the global site-packages directory? For instance, on a shared host.

In all these cases, virtualenv can help you. It creates an environment that has its own installation directories, that doesn’t share libraries with other virtualenv environments (and optionally doesn’t access the globally installed libraries either).

virtualenvwrapper

**Contents**

virtualenvwrapper
  - virtualenvwrapper
    - On GNU/Linux
    - On windows

On GNU/Linux  See Also:


*virtualenvwrapper* is a set of extensions to Ian Bicking’s *virtualenv* tool for creating isolated Python development environments.

The extensions include wrappers for creating and deleting virtual environments and otherwise managing your development workflow, making it easier to work on more than one project at a time without introducing conflicts in their dependencies.

On windows  See Also:


This is a port of Doug Hellmann’s *virtualenvwrapper* to Windows batch scripts.

The idea behind virtualenvwrapper is to ease usage of Ian Bicking’s *virtualenv*, a tool for creating isolated Python virtual environments, each with their own libraries and site-packages.

These should work on any version of Windows (Windows XP, Windows Vista, Windows 7). They do not require Powershell.

ruby language

See Also:

Scala est un langage de programmation multi-paradigme conçu à l’École polytechnique fédérale de Lausanne (EPFL) pour exprimer les modèles de programmation courants dans une forme concise et élégante. Son nom vient de l’anglais Scalable language qui signifie à peu près *langage adaptable* ou *langage qui peut être mis à l'échelle*. Il peut en effet être vu comme un métalangage.

Scala intègre les paradigmes de programmation orientée objet et de programmation fonctionnelle, avec un typage statique. Il concilie ainsi ces deux paradigmes habituellement opposés (à de rares exceptions près, telle que le langage OCaml) et offre au développeur la possibilité de choisir le paradigme le plus approprié à son problème.

Il est prévu pour être compilé en bytecode Java (exécutable sur la JVM), ou .Net. Ces deux plateformes sont supportées officiellement par l’EPFL, mais d’autres plateformes pourront potentiellement être supportées dans le futur.

**See Also:**

- http://www.scala-lang.org/
- http://typesafe.com/
- http://www.scala-lang.org/faq
- http://www.scala-lang.org/node/1658 (Scala in the enterprise)
- http://neopythonic.blogspot.com/2008/11/scala.html (Guido van Rossum)
- http://prezi.com/fb0vqbuqk33a/why-scala/
Akka

See Also:

- http://www.typesafe.com/technology/akka
- http://akka.io/

Akka is the platform for the next generation event-driven, scalable and fault-tolerant architectures on the JVM.

We believe that writing correct concurrent, fault-tolerant and scalable applications is too hard. Most of the time it’s because we are using the wrong tools and the wrong level of abstraction.

Akka is here to change that.

Using the Actor Model together with Software Transactional Memory we raise the abstraction level and provide a better platform to build correct concurrent and scalable applications.

For fault-tolerance we adopt the “Let it crash” / “Embrace failure” model which have been used with great success in the telecom industry to build applications that self-heal, systems that never stop.

Actors also provides the abstraction for transparent distribution and the basis for truly scalable and fault-tolerant applications.

Akka is Open Source and available under the Apache 2 License.

Akka and 0mq  See Also:

https://github.com/gruggiero/zeromq-scala-examples is a Scala port of ZeroMQ guide samples.

https://github.com/zcox/akka-zeromq-java  Examples of using Akka and 0MQ in Java, separately and together.

Vala language

See Also:

- https://live.gnome.org/Vala

Vala - Compiler for the GObject type system

Introduction

Vala is a new programming language that aims to bring modern programming language features to GNOME developers without imposing any additional runtime requirements and without using a different ABI compared to applications and libraries written in C.
Visual languages

Blockly

See Also:

http://code.google.com/p/google-blockly/

Contents

• Blockly
  – Introduction

Introduction  Blockly is a web-based, graphical programming language. Users can drag blocks together to build an application.

No typing required.

Blockly is currently a technology preview. We want developers to be able to play with Blockly, give feedback, and think of novel uses for it.

All the code is free and open source. Join the mailing list and let us know what you think.

2.18 Licences Libres

2.18.1 Licences Libres

See Also:

• https://secure.wikimedia.org/wikipedia/en/wiki/Software_license
• http://www.april.org/liberez-vos-oeuvres-appel-publier-sous-licence-libre
• http://fr.wikipedia.org/wiki/Copyleft
• http://fr.creativecommons.org
Licences Libres

Open attribute

See Also:

- http://openattribute.com/
- Open attribute extension

give credit where credit is due.

The problem: Creative Commons licensed content is awesome, but attributing it properly can be difficult and confusing.

The first rule for re-using openly licensed content is that you have to properly attribute the creator. There are specific requirements for what needs to go into that attribution, but those requirements can be confusing and hard to find.

The solution: A simple tool everyone can use to do the right thing with the click of a button.

That’s why we’re building Open Attribute, a suite of tools that makes it ridiculously simple for anyone to copy and paste the correct attribution for any CC licensed work. These tools will query the metadata around a CC-licensed object and produce a properly formatted attribution that users can copy and paste wherever they need to.

Open Attribute is a Mozilla Drumbeat project born at the “Learning, Freedom and the Web” Festival in Barcelona. A team of volunteers from all over the world has been collaborating to design, build and now distribute Open Attribute.

Special Thanks to those who have worked so hard to make this a reality!

Wordpress extension


Copyleft

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Copyleft
Le C “inversé” est le symbole du copyleft. En 2009, il n’est pas reconnu comme symbole légal. L’« opposé » est le symbole copyright.

Le copyleft est la possibilité donnée par l’auteur d’un travail soumis au droit d’auteur (œuvre d’art, texte, programme informatique, etc.) de copier, d’utiliser, d’étudier, de modifier et/ou de distribuer son œuvre dans la mesure où ces possibilités restent préservées.

L’auteur n’autorise donc pas que son travail puisse évoluer avec une restriction de ce droit à la copie, ce qui fait que le contributeur apportant une modification (une correction, l’ajout d’une fonctionnalité, une réutilisation d’une œuvre d’art, etc.) est contraint de ne redistribuer ses propres contributions qu’avec les mêmes conditions d’utilisation.

Autrement dit, les créations réalisées à partir d’éléments sous copyleft héritent de cette caractéristique.

Licence Art libre

See Also:

- http://artlibre.org/
- http://valentin.villenave.info/Quelques-questions-sur-le-Libre-et

Licence Art Libre 1.3 (LAL 1.3)

Préambule  Avec la Licence Art Libre, l’autorisation est donnée de copier, de diffuser et de transformer librement les œuvres dans le respect des droits de l’auteur.

Loi d’ignorer ces droits, la Licence Art Libre les reconnaît et les protège.

Elle en reformule l’exercice en permettant à tout chacun de faire un usage créatif des productions de l’esprit quels que soient leur genre et leur forme d’expression.

Si, en règle générale, l’application du droit d’auteur conduit à restreindre l’accès aux œuvres de l’esprit, la Licence Art Libre, au contraire, le favorise.

L’intention est d’autoriser l’utilisation des ressources d’une œuvre ; créer de nouvelles conditions de création pour amplifier les possibilités de création.

La Licence Art Libre permet d’avoir jouissance des œuvres tout en reconnaissant les droits et les responsabilités de chacun.

GFDL (Licence de documentation libre GNU)

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/GFDL
La licence de documentation libre GNU (de l’anglais GNU Free Documentation License), abrégée en GFDL, est une licence relevant du droit d’auteur produite par la Free Software Foundation.

Elle a pour but de protéger la diffusion de contenu libre et peut être utilisée par chacun afin de déterminer le mode de diffusion de son œuvre.

Compatibilité entre GFDL et CC-BY-SA  
Bien que les deux licences soient très proches, la GFDL n’est pas compatible avec CC-BY-SA (Creative Commons - paternité - partage à l’identique).

Cependant, le paragraphe 11 de la version 1.3 de la GFDL permet la migration depuis la GFDL 1.3 vers CC-BY-SA 3.0 sous certaines conditions :

- Elle ne concerne que les sites collaboratifs massivement multi-auteur (Massive Multiauthor Collaboration Site, MMC), dont les wikis sont le meilleur exemple.
- De plus, la possibilité d’effectuer la transition est temporaire et limitée dans le temps, elle ne concerne que les éléments sous GFDL avant le 1er novembre 2008 et la possibilité de changement a expiré le 1er août 2009.

Creative Commons

See Also:
- http://fgallaire.flext.net/tag/creative-commons/
- http://fr.creativecommons.org

CC-BY-NC-SA 3.0 (Paternité - Pas d’Utilisation Commerciale - Partage des Conditions Initiales à l’Identique 3.0 Unported)

See Also:
- http://creativecommons.org/licenses/by-nc-sa/3.0/deed.fr
- http://fgallaire.flext.net/

Paternité - Pas d’Utilisation Commerciale - Partage des Conditions Initiales à l’Identique 3.0 Unported (CC BY-NC-SA 3.0)

Vous êtes libres
- de reproduire, distribuer et communiquer cette création au public
- de modifier cette création
Selon les conditions suivantes  Paternité (by)

Vous devez citer le nom de l’auteur original de la manière indiquée par l’auteur de l’œuvre ou le titulaire des droits qui vous confère cette autorisation (mais pas d’une manière qui suggérerait qu’ils vous soutiennent ou approuvent votre utilisation de l’œuvre).

Pas d’Utilisation Commerciale (nc)

Vous n’avez pas le droit d’utiliser cette création à des fins commerciales.

Partage des Conditions Initiales à l’Identique (sa)

Si vous modifiez, transformez ou adaptez cette création, vous n’avez le droit de distribuer la création qui en résulte que sous un contrat identique à celui-ci.

CC-BY-NC-SA 2.0  Paternité - Pas d’Utilisation Commerciale - Partage des Conditions Initiales à l’Identique
2.0 France (CC BY-NC-SA 2.0)  See Also:
- http://creativecommons.org/licenses/by-nc-sa/2.0/fr/
- http://fr.issuepedia.org/Creative_commons

Vous êtes libres

- de reproduire, distribuer et communiquer cette création au public
- de modifier cette création

Selon les conditions suivantes  Paternité (by)

Vous devez citer le nom de l’auteur original de la manière indiquée par l’auteur de l’œuvre ou le titulaire des droits qui vous confère cette autorisation (mais pas d’une manière qui suggérerait qu’ils vous soutiennent ou approuvent votre utilisation de l’œuvre).

Pas d’Utilisation Commerciale (nc)
Vous n’avez pas le droit d’utiliser cette création à des fins commerciales.

Partage des Conditions Initiales à l’Identique (sa)

Si vous modifiez, transformez ou adaptez cette création, vous n’avez le droit de distribuer la création qui en résulte que sous un contrat identique à celui-ci.

CC-BY-SA 2.0 (Paternité - Partage des Conditions Initiales à l’Identique 2.0)  See Also:

- http://creativecommons.org/licenses/by-sa/2.0/fr/
- http://fr.issuepedia.org/Creative_commons

Vous êtes libres

- de reproduire, distribuer et communiquer cette création au public
- de modifier cette création

Selon les conditions suivantes  Paternité (by)

Vous devez citer le nom de l’auteur original de la manière indiquée par l’auteur de l’œuvre ou le titulaire des droits qui vous confère cette autorisation (mais pas d’une manière qui suggérerait qu’ils vous soutiennent ou approuvent votre utilisation de l’œuvre).

Partage des Conditions Initiales à l’Identique (sa)

Si vous modifiez, transformez ou adaptez cette création, vous n’avez le droit de distribuer la création qui en résulte que sous un contrat identique à celui-ci.

Free Software licenses

A software license (or software licence in commonwealth usage) is a legal instrument (usually by way of contract law) governing the usage or redistribution of software. All software is copyright protected, except material in the public domain.
Contractual confidentiality is another way of protecting software. A typical software license grants an end-user permission to use one or more copies of software in ways where such a use would otherwise constitute copyright infringement of the software owner’s exclusive rights under copyright law.

**Ethique des logiciels libres**  
See Also:
- [http://fgallaire.flext.net/](http://fgallaire.flext.net/)

**Python licenses**

**PSF license**  
See Also:
- [http://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq](http://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq)

Michael Foord <fuzzyman@voidspace.org.uk>

heure de l’expéditeur  
Envoyé à 00:27 (UTC). Heure locale : 07:51.

à the-fellowship-of-the-packaging@googlegroups.com

cc VanL <van@python.org>

date 16 février 2011 00:27

objet Re: [Heads Up] Contributors to Distutils2

liste de diffusion <the-fellowship-of-the-packaging.googlegroups.com>

On 15 February 2011 23:11, Michael Foord <fuzzyman@voidspace.org.uk> wrote:

On 15 February 2011 22:10, Fred Drake <fdrake@acm.org> wrote:

> On Tue, Feb 15, 2011 at 4:58 PM, Tarek Ziadé <ziade.tarek@gmail.com> wrote:
> > So are you implying that the Distutils2 project cannot be licensed 
> > under PSF ? We will do standalone releases.
> 
> distutils2 can be publicly offered under the PSF license.

But shouldn’t. :-} (Right, Van?)

I’ve had a reply from Van Lindberg, and I’m posting it here as it may not get through if he isn’t subscribed to the list. My advice is slightly incorrect: it turns out that the PSF license (different from the Python license) is perfectly suitable for use by distutils2 (so my apologies):

There is a difference between the Python License Stack (run away!) and the PSF license. The PSF license is the one at the top of the stack

The PSF license can be used in the same way that the BSD license is used - just remove “Python Software Foundation” and put in “Michael Foord” or [whomever]. The PSF license isn’t bad to use - in fact, it can be a good one to use if the intent is for the end product to be integrated with Python.

On choosing a license (and on my decision to use BSD for unittest2) Van says:
This is also ok, and does the same thing as what I just described. It is probably preferable if the intent isn’t solely to integrate with Python, but instead to be a standalone project.
Licenses form communities, so pick a license that will put you into (or let you into) the community you want to be associated with.
Thanks,
Van

Licencing your Python apps for dummies

Mathieu Leduc-Hamel <marrakis@gmail.com>

> oh my... I thought I was that super cool Python guy by using the PSF licence.
> We need a "Licencing your Python apps for dummies" talk at Pycon :D

I can do it, or maybe a tutorial might be more appropriate...

Then it seems we can stick PSF License but there’s another requirement inside the agreement, saying that we should:
“Contributor shall identify each Contribution by placing the following notice in its source code adjacent to Contributor’s valid copyright notice”

Is it ok from your understanding, if we just extract from bitbucket all the contributions of the people of Montreal-Python and asking them to review, i don’t wanna ask every person to identify, personally all possible lines of code on their side, nobody would do it...

protect the author from getting sued by user

Yannick Gingras <ygingras@ygingras.net>

> Also, BSD-Like licenses gives credits to the authors I think. And I
> want to give that recognition to all d2 contributors, because that’s
> all they get in return besides experience ;)

There is that and there is the fact that the almost all free/open licences protect the author from getting sued by user is the code does not work.

Apache License v2.0
>My problem is in python-cerealizer:
>
>| # Cerealizer
>| # Copyright (C) 2005-2008 Jean-Baptiste LAMY
>| # Copyright (C) 2008 Peter Eckersley
>| #
>| # This program is free software.
>| # It is available under the Python licence.
>|`----
>
>Upstream licensed its software using Python license to allow the
>inclusion into Python.

I think this is essentially meaningless. ;) The “Python license” is an imprecise way to refer to the Python Software Foundation License that Python itself is released under. It is not correct to use that license unaltered in third party libraries, even to allow its inclusion in Python.

For that, the PSF only accepts contributions licensed under the Academic Free License v2.1, or the Apache License v2.0. Together with a signed contributor agreement, the code would then be relicensed under the PSF license for release with Python.

All the gory details are in the following links:

- http://www.python.org/psf/contrib/
- http://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq

Cheers, -Barry

Books

Books on Free Software licenses

Option libre See Also:


People

Free Software “People”

Benjamin Jean Benjamin Jean est un juriste spécialisé en propriété intellectuelle.

Il travaille sur les nouveaux usages et collaborations qui se développent autour de la création et de l’innovation. Il a notamment œuvré pour leur prise en compte au sein des gouvernances de multiples acteurs privés et publics, et formé de nombreuses équipes aux aspects juridiques afférents.

Actif depuis près de dix ans dans ce domaine, il enseigne la propriété intellectuelle dans plusieurs Masters, intervient comme consultant au sein du cabinet Gilles Vercken (Paris) et achève une thèse sur les systèmes collaboratifs.

À l’échelle européenne, il organise les conférences annuelles EOLE (European Open Source & Free Software Law Event) et il est membre de l’European Legal Network (FSF Europe).

Très présent dans les communautés du logiciel libre, il est cofondateur de Veni Vidi Libri et de la SARD (Société d’Acceptation et de Répartition des Dons).

En 2011, il a créé sa propre société, Inno, qui accompagne les entreprises et acteurs publics dans l’ouverture de leur politique d’innovation en faveur de processus partagés et collaboratifs.

2.19 Localization (l10n)

2.19.1 Localization (l10n)

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/R%C3%A9gionalisation_de_logiciel

La régionalisation de logiciel

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/R%C3%A9gionalisation_de_logiciel

https://secure.wikimedia.org/wikipedia/fr/wiki/R%C3%A9gionalisation_de_logiciel

La régionalisation de logiciel concerne le processus de traduction de l’interface utilisateur d’un logiciel d’une langue vers une autre et en l’adaptant à la culture locale. On utilise parfois le terme localization, qui est une transposition du mot anglais localization (faux ami).

On écrit parfois l10n car le mot localization est composé de dix lettres encadrées par un l et un n.

Avant qu’un logiciel ne soit régionalisé, il faut au préalable qu’il ait été internationalisé, c’est-à-dire qu’il ait été écrit pour pouvoir être traduit dans différentes langues et cultures.

Ce processus demandant beaucoup de travail et d’efforts de la part des équipes de développement, des outils ont été créés pour simplifier le processus de régionalisation.

Un grand nombre de ces projets sont sous-traités à des entreprises spécialisées afin de réduire les coûts.

La régionalisation d’un logiciel concerne également le fait de l’adapter à une culture. Les paramètres régionaux sont généralement partagés par les différentes applications.

Certaines applications spécialisées nécessitent leurs propres paramètres.

Le consortium Unicode travaille sur une normalisation de ces paramètres régionaux, le projet Common Locale Data Repository.

Outils d’aide à la régionalisation informatique

Sous Linux, le format po est utilisé pour permettre la réalisation de programmes informatiques multilingues.

Un fichier po est un fichier texte comportant la version anglaise des messages systèmes avec leurs équivalences traduites les remplaçant.
• Lingobit Localizer, outil de localisation pour .Net, MFC et Java
• Gettext, qui est utilisé par la plupart des logiciels libres tournant sous Linux ;
• Codito (projets Delphi) ;
• Multilizer (projets Delphi).
• Visual Localize (AIT), pour les projets Visual Studio et .NET de Microsoft Windows
• Loc@le™ d’Accent Software, pour Microsoft Windows (apparemment plus distribué)
• Qt Linguist, logiciel facilitant la traduction de programmes réalisés avec la bibliothèque Qt.
• Lokalize, l’outil de localisation de KDE.

2.20 Multimedia

2.20.1 Multimédia

Multimedia Audio

See Also:

• http://orthophonielibre.wordpress.com/2011/04/13/groupe-de-travail-accessibilite-et-logiciels-libres-de-lapril-projet-audition-et-logiciels-libres/

Multimedia Audio tools

Python Audio tools  See Also:
http://audiotools.sourceforge.net/

Python Audio Tools are a collection of audio handling programs which work from the command line.
These include programs for CD extraction, track conversion from one audio format to another, track renaming and retagging, track identification, CD burning from tracks, and more.
Supports internationalized track filenames and metadata using Unicode.
Works with high-definition, multi-channel audio as well as CD-quality.
Track conversion uses multiple CPUs or CPU cores if available to greatly speed the transcoding process.
Track metadata can be retrieved from FreeDB, MusicBrainz or compatible servers.
This project aims to show why free and open is the better way to go for audio, radio and journalism, and to set the yardsticks first—by tinkering and exploring on the edge of the possible. The term “Hyper Audio” draws inspiration from Tristan Nitot who, after seeing the first popcorn.js demo by Brett Gaylor and developers from Seneca College, coined the term Hyper Video.

Since this is an offspring of Mozilla’s existing HTML5 video efforts, it seemed appropriate to run with it.

The first demo with Tristan Nitot (the test page)  See Also:

http://yoyodyne.cc/h/

• This HTML5 audio demo is brought to you by Mozilla Drumbeat.

• It utilizes the Popcorn.js framework to dynamically display source data annotations as the audio plays.

• Player powered by jPlayer.org.

• Follow @moltke to be the first to find out about more demos like this.

On-demand speech radio: disaggregation and transcripts  See Also:

http://james.cridland.net/blog/on-demand-speech-radio-disaggregation-and-transcripts/

I’m very excited to see a test page from Henrik Moltke which takes this idea one step further, using the audio from DR’s Hard Drive radio programme. (You’ll need a decent HTML5 browser, like Firefox v4, or Chrome).

This offers disaggregated audio and a transcript. It offers translation into English from the slightly less accessible Danish.

Using HTML5, it also shows you the translation alongside the original language as it plays. Highlight a part of the transcript, and you can tweet that part automatically.

It’s a very impressive and strong example of what’s possible with new technologies.

Later: Here’s how it was built ; this is a Mozilla project using popcorn.js.

Audio libraries

ffmpeg audio library  See Also:

• ffmpeg video library

• http://ffmpeg.org/index.html

• Extract sound from a video with ffmpeg

Project Description  FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video. It includes libavcodec - the leading audio/video codec library.
color

Argyll CMS

See Also:

http://www.argyllcms.com/

ArgyllCMS is an ICC compatible color management system, available as Open Source under the AGPL. It supports
accurate ICC profile creation for scanners, cameras and film recorders, and calibration and profiling of displays and
RGB & CMYK printers. Spectral sample data is supported, allowing a selection of illuminants observer types, and
paper fluorescent whitener additive compensation.

Profiles can also incorporate source specific gamut mappings for perceptual and saturation intents. Gamut mapping
and profile linking uses the CIECAM02 appearance model, a unique gamut mapping algorithm, and a wide selection
of rendering intents. Device Link can be created with a wide variety of advanced options. It also includes code for
the fastest portable 8 bit raster color conversion engine available anywhere, as well as support for fast, fully accurate
16 bit conversion. Device color gamuts can also be viewed and compared using a VRML viewer. Comprehensive
documentation is provided for each utility, and a general guide to using the tools for typical color management tasks
is also available. A mailing list provides support for more advanced usage

colorhug

See Also:

http://www.hughski.com/

Un colorimètre open-source pour Linux


Richard Hughes a développé un colorimètre open-source appelé ColorHug. Le dispositif mesure les couleurs affichées
à l’écran et crée un profil de calibration. Alors que le matériel existant sur le marché est 100% propriétaire et fermé,
ColorHug repose sur un bootloader et un firmware sous licence GPL.

L’auteur annonce un gain de rapidité et un coût moindre par rapport aux matériels vendus sur le marché. Vendu 60€,
la production d’une première série devrait débutée en janvier 2012.

Plus d’informations sur : www.hughski.com

frederic.mantegazza@gbiloba.org via guilde.asso.fr Intéressant !

Par contre, je ne vois pas l’intérêt d’avoir développé un nouveau soft, vu qu’il existe déjà l’excellent ArgyllCMS.
J’imagine que ce dernier supportera sans problème ce nouveau matériel, vu qu’il y a les specs (enfin, les sources).
Je vais regarder le hard, mais ça a l’air assez bestial. C’est d’ailleurs quasi le même chip que dans celui de HCFR, qui
est supporté par Argyll.
À suivre.

See Also:

http://www.homecinema-fr.com/colorimetre

2.20. Multimedia


Drawing

Raster drawing

GIMP See Also:

- http://www.gimp.org/

GIMP (GNU Image Manipulation Program) is a free software raster graphics editor.

It is primarily employed as an image retouching and editing tool and is freely available in versions tailored for most popular operating systems including Microsoft Windows, Apple Mac OS X, and Linux.

In addition to detailed image retouching and free-form drawing, GIMP can accomplish essential image editing tasks such as resizing, editing, and cropping photos, photomontages combining multiple images, and converting between different image formats.

GIMP can also be used to create animated images in many formats such as GIF and MPEG through the Animation Plugin.

Automation, scripts and plug-ins GIMP has approximately 150 standard effects and filters, including Drop Shadow, Blur, Motion Blur and Noise.

GIMP operations can be automated with scripting languages. The Script-Fu is a Scheme based extension language implemented using TinyScheme.

GIMP can also be scripted in Perl, Python (Python-fu), or Tcl. New features can be added to GIMP not only by changing program code (GIMP core), but also by creating plug-ins.

These are external programs that are executed and controlled by the main GIMP program. MathMap is an example of a plug-in written in C.

GIMP plugins See Also:

http://registry.gimp.org/

Vector drawing

inkscape See Also:


inkscape extensions

inkscape sozi extension See Also:

- http://sozi.baierouge.fr/wiki/sozi
- http://senshu.baierouge.fr/
- https://identi.ca/senshu
- https://twitter.com/#!/senshuapash
Sozi est un petit programme qui permet de jouer des présentations animées.

Contrairement aux outils de présentation classiques, un document Sozi n’est pas organisé comme un diaporama.

Il s’agit plutôt d’un poster où la disposition du contenu est entièrement libre. Le déroulement de la présentation consiste en une succession de translations, zooms et rotations permettant de focaliser l’attention sur les éléments que vous voulez montrer.

Sozi repose sur des standards ouverts. C’est un logiciel libre sous licence GPL 3.0.

**Créer** Une présentation Sozi est un document au format SVG dans lequel une séquence de “vues” a été définie. Ces vues sont délimitées par des rectangles auxquels sont associées des informations sur la présentation et l’animation, comme par exemple la durée des transitions.

L’éditeur de présentations est une extension pour le logiciel de dessin vectoriel *Inkscape*. Il est ainsi possible de construire l’intégralité d’une présentation en restant dans le même environnement.

- Installer Sozi.

**Contribuer** Le dépôt officiel du code source, chez GitHub. Signaler un problème ou proposer une fonctionnalité.

**Documentation pour les développeurs** Installer Sozi à partir des fichiers sources. Les éléments et attributs XML spécifiques à Sozi. Liste des contributeurs.

**Presentations** See Also:

http://sozi.wikidot.com/presentations

**Voir Aussi sozigen** See Also:

https://pypi.python.org/pypi/sozigen/0.1.0

This project’s purpose is to automate building of the hierachical slideshows. It is useful to show monitoring graphs on large TV displays and projectors.

With SoziGen, you describe graphs layout in python and it generates SVG file to be opened in any modern web browser.

**Multimedia Video**

**Video libraries**

ffmpeg video library See Also:

• ffmpeg audio library
Development tools, Release 2012.06.18

- http://ffmpeg.org/index.html
- http://ffmpeg.org/about.html
- http://www.commandlinefu.com/commands/tagged/76/ffmpeg
- http://pypi.python.org/pypi/wffmpeg/0.1.1
- libav
- Video ffmpeg scripts

Project Description  FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video. It includes:

- libavcodec, the leading open source codec library
- libavformat (implementing muxers and demuxers),
- libswscale (for very fast video scaling),
- libavfilter (an advanced video filter system supporting arbitrary filter graphs),
- libavutil (a utility library intended to supplement libc).

A simple multimedia player is included too. An experimental streaming server for live broadcasts is also included.

FFmpeg is free software licensed under the LGPL or GPL depending on your choice of configuration options. If you use FFmpeg or its constituent libraries, you must adhere to the terms of the license in question. You can find basic compliance information and get licensing help on our license and legal considerations page.

The project is made of several components:

- ffmpeg is a command line tool to convert multimedia files between formats.
- ffserver is a multimedia streaming server for live broadcasts.
- ffplay is a simple media player based on SDL and the FFmpeg libraries.
- ffprobe is a is a simple multimedia stream analyzer.
- libavutil is a library containing functions for simplifying programming, including random number generators, data structures, mathematics routines, core multimedia utilities, and much more.
- libavcodec is a library containing decoders and encoders for audio/video codecs.
- libavformat is a library containing demuxers and muxers for multimedia container formats.
- libavdevice is a library containing input and output devices for grabbing from and rendering to many common multimedia input/output software frameworks, including Video4Linux, Video4Linux2, VfW, and ALSA.
- libavfilter is a library containing media filters.
- libswscale is a library performing highly optimized image scaling and color space/pixel format conversion operations.
Commandlinefu  See Also:
http://www.commandlinefu.com/commands/tagged/76/ffmpeg

FFmpeg Git  FFmpeg is developed with Git. Given the decentralized nature of Git, multiple repositories from developers and groups of developers are available.

- git://git.videolan.org/ffmpeg.git

download  See Also:
http://freecode.com/projects/ffmpeg#release_338118

About FFmpeg  The project is made of several components:

- ffmpeg is a command line tool to convert multimedia files between formats.
- ffserver is a multimedia streaming server for live broadcasts.
- ffplay is a simple media player based on SDL and the FFmpeg libraries.
- ffprobe is a simple multimedia stream analyzer.
- libavutil is a library containing functions for simplifying programming, including random number generators, data structures, mathematics routines, core multimedia utilities, and much more.
- libavcodec is a library containing decoders and encoders for audio/video codecs.
- libavformat is a library containing demuxers and muxers for multimedia container formats.
- libavdevice is a library containing input and output devices for grabbing from and rendering to many common multimedia input/output software frameworks, including Video4Linux, Video4Linux2, VfW, and ALSA.
- libavfilter is a library containing media filters.
- libswscale is a library performing highly optimized image scaling and color space/pixel format conversion operations.

ffmpeg history  See Also:

libav History

May 3, 2011  FFmpeg now accesses x264 presets via libx264.

This extends functionality by introducing several new libx264 options including -preset, -tune, and -profile.

You can read more detailed information about these options with “x264 --fullhelp”.

The syntax has changed so be sure to update your commands.

Example:

ffmpeg -i input -vcodec libx264 -preset fast -tune film -profile main -crf 22 -threads 0 output

Versions

ffmpeg versions

2.20. Multimedia
ffmpeg 0.9.0 “Harmony”  See Also:
http://ffmpeg.org/download.html#release_0.9

0.9 was released on 2011-12-11.

It is the latest stable FFmpeg release from the 0.9 release branch, which was cut from master on 2011-12-11.

Amongst lots of other changes, it includes all changes from:

- ffmpeg-mt,
- libav master of 2011-12-11
- libav 0.7.2 as of 2011-12-11

We have made a new major release (0.9) It contains all features and bugfixes of the git master branch. A partial list of new stuff is below:

- native dirac decoder
- mmsh seeking
- more accurate rgb->rgb in swscale
- MPO file format reading support
- mandelbrot fraktal video source
- libass filter
- export quarter_sample & divx_packed from decoders
- VBLE decoder
- libopenjpeg encoder
- alpha opaqueness fixes in many codecs
- 8bit palette dynamic range fixes in many codecs
- AVIOInterruptCB
- OS/2 threads support
- cbr mp3 muxing fix
- sample rate change support in flv (nellymoser decoder)
- mov/mp4 chunking support (equivalent to mp4boxs -inter)
- mov/mp4 fragment support (equivalent to mp4boxs -frag)
- rgba tiffs
- x264rgb bugfix
- cljrencoder with dither
- escape130 decoder
- many new ARM optimizations
- report
- Dxtory capture format decoder
- life video source
- wtv, sox, utvideo and many other new regression tests
- gcc coverage support
- callauto video source
- planar rgb input support in sws
- libmodplug & bintext output
- g723.1 encoder
- g723.1 muxer
- random() function for the expression evaluator
- persistent variables for the expression evaluator
- pulseaudio input support
- h264 422 inter decoding support
- prores encoder
- native utvideo decoder
- libutvideo support
- deshake filter
- aevalsrc filter
- segment muxer
- mkv timecode v2 muxer
- cache urlprotocol
- libaacplus support
- ACT/BIT demuxers
- AMV video encoder
- g729 decoder
- stdin control of drawtext
- 2bpp, 4bpp png support
- interlaced 1bpp and PAETH png fixes
- libspeex encoding support
- hardened h264 decoder that wont overread the bitstream
- wtv muxer
- H/W Accelerated H.264 Decoding on Android
- stereo3d filter from libmpcodecs works now
- an experimental jpeg2000 encoder
- many bugfixes
- libswresample

Wrappers

ffmpeg wrappers
python ffmpegwrapper  See Also:

- http://pypi.python.org/pypi/ffmpegwrapper/
- http://github.com/interrupted/ffmpegwrapper

A simple wrapper for ffmpeg-cli

FFmpegWrapper is a small wrapper for the ffmpeg encoder.

You can append Codec, Filters and other OptionStores to the FFmpeg class and then run the resulting command:

```python
>>> from ffmpegwrapper import FFmpeg, Input, Output, VideoCodec, VideoFilter

>>> codec = VideoCodec('webm')

>>> input_video = Input('old')

>>> output_video = Output('new', videofilter, codec)

>>> FFmpeg('ffmpeg', input_video, output_video)

<FFmpeg ['ffmpeg', '-i', 'old', '-vcodec', 'webm', 'new']>
```

libav  See Also:

- http://libav.org/

Libav is a complete, cross-platform solution to record, convert and stream audio and video. It includes libavcodec - the leading audio/video codec library.

See the documentation for a complete feature list and the Changelog for recent changes.

Features  We strive to implement and provide useful and innovative features in a timely fashion, provide stable release and interact to the best of our abilities with our downstream such as distributors and end users. For instance, most of the ffmpeg-nt work has been merged into Libav for quite some time.

This work is still ongoing, we are still working hard to fix the remaining issues such as with multi-threaded h264 decoding.

We are not afraid to overhaul radically our code in order to provide a better foundation for future works and we prefer to spend effort to get clean code right instead of piling up special cases and heuristics.

We are patient enough to track bugs and corner cases until they are completely solved

With the help of our users we try to improve the experience of using both the libraries and the tools

We try to focus both on real world issues as well on experiments that might lead to new interesting outcomes.

We are happy to learn that the now arisen competition has finally lead FFmpeg to merge important and long requested features such as frame based multi-threaded decoding based on ffmpeg-nt, something the project leader strongly refused to merge during our attempts to reconcile with him.

libav History

March 13, 2011  We, as a group of FFmpeg developers, have decided to continue developing FFmpeg under the name Libav. All existing infrastructure will be transferred to the libav.org domain.

<table>
<thead>
<tr>
<th>Website</th>
<th><a href="http://www.libav.org/">http://www.libav.org/</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Git</td>
<td>git://git.libav.org/libav.git (see <a href="http://libav.org/download.html">http://libav.org/download.html</a>)</td>
</tr>
<tr>
<td>FATE</td>
<td><a href="http://fate.libav.org/">http://fate.libav.org/</a></td>
</tr>
<tr>
<td>Roundup</td>
<td><a href="https://roundup.libav.org/">https://roundup.libav.org/</a></td>
</tr>
<tr>
<td>Patchwork</td>
<td><a href="http://patches.libav.org/">http://patches.libav.org/</a></td>
</tr>
<tr>
<td>Mailinglists</td>
<td><a href="http://lists.libav.org/">http://lists.libav.org/</a> (developer list: <a href="mailto:libav-devel@libav.org">libav-devel@libav.org</a>)</td>
</tr>
</tbody>
</table>
You can update your git repository using the following command:

```
git remote set-url origin 'git://git.libav.org/libav'
```

For now we are still reachable over FFmpeg’s mailing lists and IRC channels but we will migrate to libav.org counterparts. For a transition period both the website and source might still contain references to FFmpeg. These will disappear over time, except where historically relevant.

**gstreamer multimedia framework**  
See Also:


GStreamer is a pipeline-based multimedia framework written in the C programming language with the type system based on GObject.

GStreamer allows a programmer to create a variety of media-handling components, including simple audio playback, audio and video playback, recording, streaming and editing. The pipeline design serves as a base to create many types of multimedia applications such as video editors, streaming media broadcasters, and media players.

Designed to be cross-platform, it is known to work on Linux (x86, PowerPC and ARM), Solaris (Intel and SPARC) and OpenSolaris, FreeBSD, OpenBSD, NetBSD, Mac OS X, Microsoft Windows and OS/400. GStreamer has bindings for programming-languages like Python, Vala, C++, Perl, GNU Guile and Ruby. GStreamer is licensed under the GNU Lesser General Public License.

See Also:


See Also:


**gstreamer multimedia framework**  
See Also:

- [http://gstconf.ubicast.tv/](http://gstconf.ubicast.tv/)

**gstreamer people**

**wim-taymans**  
See Also:

Wim Taymans has a computer science degree from Katholieke Universiteit in Leuven, Belgium and decades of software development experience. He co-founded the GStreamer multimedia framework in 1999, and co-founded Collabora in 2007.

Today Wim focuses on maintaining and extending the core of GStreamer, both for clients and as general research and development.

For those who don’t know you, can you tell us a little about yourself? Wim Taymans: I’m a Belgian living in Barcelona with my awesomest wife Maria and my cute newborn son Ilian. I’ve been programming since an early age, first I learned BASIC on ZX80, then later I wrote demos and games for Commodore 64 and Amiga. We recently managed to release a new Commodore 64 game that I wrote with a friend back in 1990, called Jim Slim.

I got involved with Linux at around 1998 when I cofounded a project to write a video editor. We soon figured out that we needed a solid multimedia framework for our needs. We started looking for available options and that’s how I became involved with GStreamer at around 1999.

For the last 3 years I have been working together with many of the other core GStreamer developers at Collabora, providing development and design work for customers using GStreamer all over the world.

The GStreamer 0.10.x series has been around since December 2005, which means GStreamer has been able to stay API stable for about 6 years, what were the major lessons learned from GStreamer 0.10?

Wim Taymans: It’s indeed quite impressive how we managed to keep the API stable that long without hitting a wall. A lot of thought went into the 0.10 API and we managed to assemble some of the brightest minds in one place to develop the 0.10 version.

gstreamer tutorials

gstreamer nicolargo tutorial See Also:
• http://blog.nicolargo.com/gstreamer

Mise à jour de la présentation de GStreamer

Il y a quelques mois, j’avais publié une présentation (au format “Powerpoint”) du framework multimédia GStreamer. A l’occasion d’une présentation de cette technologie à la commission open-source de la Telecom Valley, j’ai mis à jour cette présentation en l’illustrant avec des exemples de pipelines que je lançais au fur et à mesure de mon exposé. La nouvelle version (1.2) de cette présentation est disponible au téléchargement aux formats PDF et ODP. Elle est diffusé sous licence Creative Common BY v3.0 comme la totalité des billets de ce blog. Vous pouvez également télécharger les scripts shells contenant les pipelines utilisés lors de la présentation.

• http://blog.nicolargo.com/wp-content/plugins/download-monitor/download.php?id=8

Note: l’archive des scripts shells contient également une musique (Carl Phaser - “Domination” sous licence CC BY-NC-SA) et une vidéo (Justin Cone - “Building on the Past” sous licence CC BY-NC 1.0) que j’utilise dans ces pipelines.

Video tools

See Also:
http://fr.wikipedia.org/wiki/Montage_vid%C3%A9o
FF diaporama  See Also:

## Contents
- FF diaporama
  - Introduction

### Introduction
ffDiaporama est une application de création de séquences vidéos constituées :
- de titres, fixes ou animées.
- d’images ou de photographies, fixes ou animées.
- de clips vidéo
- de musiques

### Video ffmpeg

#### Video ffmpeg scripts  See Also:
- ffmpeg video library
- http://www.commandlinefu.com/commands/tagged/76/ffmpeg

#### Video ffmpeg images to video  See Also:
- http://www.freesoftwaremagazine.com/articles/assembling_video_png_stream_ffmpeg

Here is the command that takes the stream of PNGs and strips off the letterboxing, then packs it up as raw video, 24fps:

```bash
ffmpeg -i 1080/sintel_trailer_2k_%04d.png -vf crop=1920:816:0:132 -vcodec rawvideo -r 24 -pix_fmt yuv444p sintel_trailer.y4m
```

From there its easy to convert into theora:

```bash
ffmpeg2theora -v 10 -K 24 sintel_trailer.y4m
```

Personally I’ve never gotten comfortable with specifying the video quality, I always like to try out different bit-rates and then look at different results to figure out what is “good enough” (to just try it on part of the video without encoding the whole thing use -ss to specify a staring point and -t to specify a duration). Without the video quality paramater (which I don’t think ffmpeg supports), I can do it all at once:

```bash
ffmpeg -i 1080/sintel_trailer_2k_%04d.png -vf crop=1920:816:0:132 -vcodec libtheora -b 9000k -r 24 -pix_fmt yuv444p sintel_trailer.ogv
```

You could also go to the VP8 codec, and knock about 55% off the file size with about the same quality:

```bash
ffmpeg -i 1080/sintel_trailer_2k_%04d.png -vf crop=1920:816:0:132 -vcodec libvpx -b 4000k -r 24 -pix_fmt yuv444p sintel_trailer.ogv
```

Or if you really want quality, you could do either of those with 2-pass:

```bash
ffmpeg -i 1080/sintel_trailer_2k_%04d.png -vf crop=1920:816:0:132 -vcodec libtheora -b 9000k -r 24 -pix_fmt yuv444p -pass 1
```

```bash
ffmpeg -i 1080/sintel_trailer_2k_%04d.png -vf crop=1920:816:0:132 -vcodec libtheora -b 9000k -r 24 -pix_fmt yuv444p -pass 2
```
ffmeg -i 1080/sintel_trailer_2k_%04d.png -pass 1 -vf crop=1920:816:0:132 -vcodec libtheora -b 600k -bt 600k -r 24 -pix_fmt yuv444p -f rawvideo -y /dev/null
ffmeg -i 1080/sintel_trailer_2k_%04d.png -pass 2 -vf crop=1920:816:0:132 -vcodec libtheora -b 600k -bt 600k -r 24 -pix_fmt yuv444p

ffmeg -i 1080/sintel_trailer_2k_%04d.png -pass 1 -vf crop=1920:816:0:132 -vcodec libvpx -b 300k -bt 300k -r 24 -pix_fmt yuv444p -f rawvideo -y /dev/null
ffmeg -i 1080/sintel_trailer_2k_%04d.png -pass 2 -vf crop=1920:816:0:132 -vcodec libvpx -b 300k -bt 300k -r 24 -pix_fmt yuv444p

If you really want the letterboxing, (sigh) you can add:

-padtop 131 -padbottom 131 -padcolor 000000 to the ffmpeg commands.

To see the results for the entire video quality demo I did for my class, check out this site: http://mediaintro.teeks99.com/Video/Video-3-Quality.html

FF Multi Converter  See Also:


FF Multi Converter is a GUI application that converts multiple file formats to different extensions, using and combining other programs.

The application supports Audio, Video, Image and Document file formats.

It uses ffmpeg for audio/video files, unoconv for document files (which uses the OpenOffice’s UNO bindings) and PIL library for image file conversions. It also offers recursively conversions (same type or extension).

The application is available for Linux platforms only.

Supported formats: https://github.com/Ilias95/FF-Multi-Converter/wiki/Supported-formats

Requirements and Dependencies To run this program you need: python 2, PyQt4

Dependencies: Python Imaging Library (PIL), ffmpeg, unoconv, Open/Libre office suite

The program does NOT require all dependencies to run. E.g. you can run the application even if you don’t have PIL installed, but you will be able to convert any other types except image files.

In an Ubuntu system you can install all requirements and dependencies with the command:

-sudo apt-get install python-qt4 ffmpeg unoconv

Python, PIL and OpenOffice are already installed.

Download and install Download: http://pypi.python.org/pypi/ffmulticonverter/1.1.0

You don’t have to compile the program. From application’s directory just run:

-sudo ./setup.py install

You will find it at Applications menu in Gnome/Unity or you can execute from terminal by typing ffmulticonverter.
Free Software

Copyright (C) 2011 Ilias Stamatis < stamatis.iliass@gmail.com >
License: GPL 3
Source code: https://github.com/Ilias95/FF-Multi-Converter

Current Version 1.1.0

Video openshot editor  See Also:
- http://fr.wikipedia.org/wiki/Montage_vid%C3%A9o
- https://launchpad.net/openshot

OpenShot Video Editor est un éditeur vidéo non-linéaire libre pour Linux, construit avec Python, GTK+, et le MLT Framework.
Il intègre le moteur Blender pour créer des titres et effets (feux d’artifices, jeux de lumières, neige, mappemonde animée) évolués. Le projet, initié par Jonathan Thomas, a démarré en août 2008, avec pour objectif de fournir un éditeur vidéo stable, libre, agréable, performant et d’utilisation facile.

watchvideo  See Also:
- http://pypi.python.org/pypi/WatchVideo

A small application to play or download videos from various YouTube-like sites.
WatchVideo is an application to watch videos from many popular Flash based sites using an external player like VLC.

For both GUI and CLI interface
- Python 2.5/2.6/2.7 (tested with Python 2.6 and 2.7)
- GetMediumURL 0.0a2 or later
- simplejson if using Python 2.5 (python-simplejson)

The installation script will automatically install GetMediumURL if it is not available.
It requires lxml which should be installed from distribution package before (in Debian-based GNU/Linux distributions it is named python-lxml).

For GUI only
- PyQt4 (python-qt4) or PySide
- libVLC version 1.1 (optional, required for built-in player)
- python-notify (optional) – uses the system’s default mechanism to show notification messages
- FFmpeg (optional) – needed for all operations related with video conversion, ripping or the built-in player.
- FFmpeg2theora (optional) – needed to convert videos to Ogg Vorbis or Theora, but not for ripping audio

For installation  Distribute (python-setuptools)
If you downloaded through Subversion

- pyqt4-dev-tools (pylupdate4, pyrcc4, pyuic4)
- libqt4-dev (lrelease)
- Inkscape
- OptiPNG (optional)

PyQt4 is used by default. When it is not found, PySide is used but there are known bugs when using it.

Set the environment variable WATCHVIDEO_QT to a space-separated sequence of package names tried to use a different package (e.g. WATCHVIDEO_QT=PySide to use PySide on systems where PyQt4 is installed).

UMplayer  See Also:

http://sourceforge.net/projects/umplayer/

UMPlayer is a cross-platform multimedia player supporting many audio and video codecs and file formats as well as DVDs, VCDs, and various streaming protocols.

It boasts many advanced features such as built-in subtitle search and YouTube player.

Cross-Platform  UMPlayer is written under the Qt platform and is available for Microsoft Windows, Apple Mac OS/X, and GNU/Linux operating systems.

Source code

svn co https://umplayer.svn.sourceforge.net/svnroot/umplayer umplayer

VLC media player (VLC)  See Also:

- http://www.commandlinefu.com/commands/tagged/1117/vlc

VLC media player (VLC) (à l’origine VideoLAN Client) est un lecteur multimédia libre issu du projet VideoLAN. Ce logiciel est multiplateforme puisqu’il fonctionne sous Windows, toutes les tendances GNU/Linux, BSD, Mac OS X, BeOS, Solaris, QNX et même Pocket PC soit en tout près de 20 plateformes.

Il est distribué sous licence GNU/GPL.

Un des grands atouts de VLC est qu’il intègre les codecs nécessaires à la lecture de la plupart des formats audio et vidéo. De plus, le lecteur est capable de lire un grand nombre de flux réseaux. Il se montre par ailleurs très tolérant avec les flux légèrement endommagés, allant même jusqu’à les réparer de son mieux.

Historique  VLC media player était au début le client pour le projet VideoLAN qui avait pour but la création d’un logiciel permettant la diffusion de vidéos à travers un réseau informatique. Il était à l’origine développé par les étudiants de l’école centrale Paris et a été diffusé pour la première fois le 1er février 2001 sous licence GNU GPL. Il est aujourd’hui développé par des contributeurs du monde entier. Le cœur du développement de VLC n’est toutefois assuré que par une dizaine de personnes, dont seulement quatre développeurs très actifs.
Fonctions de VLC  Article détaillé : VideoLAN pour des informations sur le streaming.

VLC est très populaire pour sa capacité à jouer des vidéos incomplètes ou endommagées. Ceci permet de prévisualiser des vidéos avant de les avoir téléchargées entièrement. Exemple : avec eMule, il permet de vérifier la qualité du fichier téléchargé et de vérifier si on a affaire à un faux fichier (fake) ou pas.

Une autre caractéristique intéressante est la capacité de VLC à utiliser la bibliothèque libcdio pour lire les fichiers .iso pour que l’utilisateur puisse jouer les fichiers sur une image de disque, même si le système d'exploitation de l’utilisateur n’a pas la capacité de travailler directement avec les fichiers .iso.

VLC supporte tous les codecs et tous les formats de fichier supportés par FFmpeg.

Le lecteur multimédia VLC a des filtres de distorsion, de rotation, d’inversion, de désentrelacement, d’ajustement, de duplication, d’agrandissement ou de redimensionnement.

Sous Windows, VLC supporte également DirectShow.

PiTiVi  See Also:

- https://fr.wikipedia.org/wiki/PiTiVi

PiTiVi est un logiciel libre de montage vidéo non linéaire conçu pour s’intégrer au bureau GNOME et distribué selon les termes de la licence GNU GPL.

Il est en phase de développement intensif. Il a été intégré dans l’installation par défaut de Ubuntu depuis la version 10.04 (Lucid Lynx) jusqu’à la version 11.04 (Natty Narwal) inclus.

Historique du développement  Edward Hervey1 a initié le projet en 2004 dans le cadre de son projet de fin d’études à l’EPITECH. L’année suivante Edward rejoint Fluendo où il travaille sur GStreamer et PiTiVi. En 2007, il poursuit son travail au sein de Collabora.


Parallèlement au développement de PiTiVi, les développeurs enrichissent GStreamer d’outils facilitant la réalisation ultérieure d’autres logiciels de montage non linéaires : GNonLin, puis GStreamer Editing Services (GES)5 (avec le soutien financier de Nokia pour ce dernier, dans le cadre du projet MeeGo).

PiTiVi a également reçu du soutien dans le cadre du Google Summer of Code (GSoC) (Brandon Lewis et Thibault Saunier notamment ont commencé à travailler sur le projet dans ce cadre).

People

Thibault Saunier  See Also:

http://thiblahute.blogspot.com/
You may have seen that Brett Gaylor is joining Mozilla (see also Mark Surman’s post). For those who don’t know Brett, he’s particularly famous for his “Open Source documentary” Rip! A remix Manifesto [1].

One may ask why Mozilla has hired a film director[2], but it actually makes a lot of sense thanks to Mozilla Drumbeat, as Brett is working on a Drumbeat project called Web made movies.

Now Brett has been a Mozilla community member for quite some time, contributing with the good folks at CDOT / Seneca College to create popcorn.js, “a JavaScript library for merging video with semantic data”.

I understand that this is a bit of a mouthful, but don’t close your browser window just yet! Popcorn.js is what I would describe as “hyper-video” (“hyper” as in “hypertext”): the ability to leverage data from the video and link to it, Web style. Such data include:

- location. Where on earth was this video sequence made? Then display it on an interactive map
- subtitles. What is being said on the soundtrack. Display it as text, and offer to translate it into the foreign language of your choice using an online translation service
- license. Under which license is this video sequence made available? (Copyright, Creative Commons, etc.)
- person. Who’s on the screen? If we know, then link to his/her Twitter and Flickr streams in real time
- topic. What is being discussed? Then link to the corresponding article in Wikipedia and in the news.

Go and see for yourself the PopCorn.js demo (in case you’re stuck with an older browser that is not capable of running the demo, here is a video of the demo).

I think this is a very significant step further for video on the Web, which was until now a very TV-like, passive and linear approach, now merged with the hypertext nature of the Web (its ability to link to things in other places), so that users can click on links in order to learn more. Of course, this is just a demo.

Tons of things need to be done, but I see this as a very cool way to show what HTML5 and its video element, combined with the power of JavaScript and mash-ups.
This little pearl of wisdom was delivered to me by my fellow Mozillian Chris Blizzard while I was interviewing him for a video.

The recent adoption of our popcorn.js library by Arte, one of Europe’s most respected broadcasters, is making me agree.

Popcorn.js is a project of Web Made Movies, Mozilla’s open video lab. It came out of a frustration with the way filmmakers and media publishers were hosting their videos on the web.

In most cases, we were seeing “TV in a webpage”, a block of proprietary code that was built on technologies that sidestepped the web’s “view source” ethic and didn’t actually interact with the rest of the page. While sites like YouTube have seen a fantastic surge of users and content over the last several years, the video itself has been a “black box” that the rest of the web page couldn’t understand. As a result, video and the rest of the web have never truly merged.

Enter HTML5 and the video tag, and suddenly we can apply web technologies like CSS and Javascript to videos.

And with popcorn.js, we can share data ABOUT the video with web services like twitter, flickr and wikipedia, allowing us to create mashups of web services and moving images.

When Arte began noticing popcorn.js, they wondered if this work could be a good fit for some of their programming. They turned to the seasoned developers at Upian, who were able to pull our open source code from Github, and today we’re immensely proud to announce that the web versions of Notre Poison Quotidien have been augmented and remixed using popcorn.js.

As you can see on the beautiful pages they have put together, this content is an ideal use case for popcorn.js. Producers are often at a loss for how to deliver the reams of associated material that surround their content – well, thanks to the open technologies that make the web, now we can do that!

My colleague David Humphrey, whose students at Seneca College’s Center for Development of Open Technology are in fact among the core authors of the technology, expressed to me how happy he is to see the technology being adopted “in the wild”. “This is how we’ll make popcorn even better. Let’s see how people use it and learn from that”, he remarked in a recent discussion online.

In collaboration with great partners like Arte, our growing community at Web Made Movies is changing the world of web video, a little bit of software at a time.

Posted on February 27, 2011 by brett

http://butterapp.org/ The Popcorn.js Authoring Tool.

Please consider this source release to be an early development preview. This software is not stable, nor does it claim to be stable and is not recommended for production use.

All aspects of this software are subject to be changed without notice or regard for previous revision compatibility.

This code has NOT been fully optimized and should not be treated as though it has.
Event emitters within the application are subject to change.

**Video recording tools**

**Freeseer video recording tools**  See Also:

https://github.com/fosslic/freeseer

The Freeseer project is a powerful software suite for capturing or streaming video. It enables you to capture great presentations, demos, training material, and other videos. It handles desktop screen-casting with ease.

It is one of a few such tools that can also record vga output or video from external sources such as firewire, usb, s-video, or rca.

It is particularly good at handling very large conferences with hundreds of talks and speakers using varied hardware and operating systems.

Freeseer itself can run on commodity hardware such as a laptop or desktop. It is supported on Windows, and Linux, It will support MacOS soon.

Freeseer is written in Python, and uses Qt4 for its GUI.

It also uses Gstreamer for video/audio processing.

**Multimedia images**

**images libraries**

**Free image library**  See Also:


**Contents**

- Free image library
  - Introduction
  - Links
  - Sourcecode
  - Versions

**Introduction**  FreeImage is an Open Source library project for developers who would like to support popular graphics image formats like PNG, BMP, JPEG, TIFF and others as needed by today’s multimedia applications.

FreeImage is easy to use, fast, multithreading safe, compatible with all 32-bit or 64-bit versions of Windows, and cross-platform (works both with Linux and Mac OS X).
Thanks to its ANSI C interface, FreeImage is usable in many languages including C, C++, VB, C#, Delphi, Java and also in common scripting languages such as Perl, Python, PHP, TCL or Ruby.

The library comes in two versions: a binary DLL distribution that can be linked against any WIN32 C/C++ compiler and a source distribution.

Workspace files for Microsoft VS.Net 2003, VS.Net 2005 and VS.Net 2008 are provided, as well as makefiles for Linux, MinGW and Mac OS X.

From January 2000 to July 2002, FreeImage was designed and mainly developed by Floris van den Berg. FreeImage is now maintained by Hervé Drolon.

Links
See Also:
http://freeimage.sourceforge.net/links.html

Sourcecode
See Also:
http://freeimage.sourceforge.net/sourcecode.html

Versions

Freeimage versions

Freeimage 3.15.2

February 20th, 2012 – 3.15.2

! FreeImage now uses LibRaw 0.14.5
! FreeImage now uses LibPNG 1.5.8
! FreeImage now uses LibJPEG 8d
! FreeImage now uses ZLib 1.2.6
! FreeImage now uses OpenJPEG 1.5.0 (released version)
! FreeImage now uses LibTIFF 4.0.0
- [Herve Drolon] removed dependency on LibMNG 1.0.10 (MNG and JNG files are now handled internally)
+ [Herve Drolon] replaced the MNG plugin with a new MNG internal FreeImage plugin (with read support)
+ [Herve Drolon] added a new JNG internal FreeImage plugin (with read/write support)
+ [Christian Heimes] added write support to the TIFF plugin for EXIF_MAIN tags
+ [Herve Drolon] added new Exif maker note tags
+ [Herve Drolon] added TAG_COMPRESSION conversion to FreeImage_TagToString
* [Mylek Grey] enabled the use of multi-component transforms (MCT) in J2K and JP2 saving
* [Herve Drolon] refactored PluginICO in order to correctly support Windows Vista 256x256 icons
* [Herve Drolon] added minor speed improvements to FreeImage_Rescale
* [Herve Drolon] fixed dib allocation failing with very large images (i.e. more than 4GB)
* [Herve Drolon] fixed FreeImage_CloneTag behavior with ASCII data handling
* [Herve Drolon] improved JPEG plugin behavior with very big images
* [Herve Drolon] improved JPEG plugin behavior with C++ exceptions
* [Herve Drolon] fixed loading of palettized PNG with more that 256 palette entries
* [Herve Drolon] fixed a bug inside IFF plugin occuring when loading a 24-bit dib with a palette
* [Herve Drolon] fixed a bug with loading of PNG images containing a cHRM chunk (regression introduced by LibPNG 1.5.4 and fixed by LibPNG 1.5.5)
* [Herve Drolon] allowed loading of PNG with benign errors (such as images with too many IDATs)
* [Mihail Naydenov] fixed some incorrect MIME types returned by FreeImage_GetFIFMimeType
* [Herve Drolon] fixed loading of Exif with bad thumbnail data or with a bad first offset size

**pystacia Python image library**  
See Also:

- [imagemagick image library](http://www.imagemagick.org/script/index.php)

---

Pystacia is a new image manipulation library born out of practical needs. It’s simple, it’s cross-platform, runs on Python 2.5+, 3.x, PyPy and IronPython.

It’s compact but still appropriate for most of your day to day image handling tasks.

Pystacia leverages powerful *ImageMagick library* as a back-end exposing easily comprehensible Pythonic API.

Here is one of the simplest code snippets showing what you can do with it:

```python
import pystacia
image = pystacia.read('example.png')
image.rescale(320, 240)
image.rotate(30)
image.show()
image.write('output.jpeg')
# free acquired resources
image.close()
```

When saved to simple.py, the above script can be run via:

```
$ pip install pystacia
$ python simple.py
```

**Motivations**  
See Also:


**imagemagick image library**  
See Also:

ImageMagick est un logiciel libre, comprenant une bibliothèque, ainsi qu'un ensemble d'utilitaires en ligne de commande, permettant de créer, de convertir, de modifier et d’afficher des images dans un très grand nombre de formats.

Les images peuvent être découpées, les couleurs peuvent être modifiées, différents effets peuvent être appliqués aux images, les images peuvent subir des rotations, il est possible d’y inclure du texte, des segments, des polygones, des ellipses et des courbes de Bézier, etc.

ImageMagick est un logiciel libre : sa licence est compatible avec la licence GNU GPL. Il est disponible sur la plupart des plates-formes.

La plupart des fonctionnalités d’ImageMagick peuvent être utilisées en ligne de commande, mais souvent, toutefois, ImageMagick est combiné avec d’autres programmes écrits dans des langages comme Perl, C, C++, Python, Ruby, PHP, OCaml ou Java, pour lesquels des interfaces prêtes à l’emploi (PerlMagick, Magick++, pystacia PythonMagick, RMagick, MagickWand pour PHP et JMagick) sont disponibles. Cela permet de manipuler des images de façon plus automatisée.

Types de transformation  See Also:

- http://www.imagemagick.org/Usage

This web pages presents a set of examples using ImageMagick from the command line. However they are also examples of what can be done using the ImageMagick Application Programming Interface (API). As such these pages should be the first stop for IM users after reading the terse user manuals.

Les différentes options d’ImageMagick et leur syntaxe  See Also:

http://www.imagemagick.org/script/command-line-options.php

Les différents outils d’ImageMagick  Cette suite d’outils inclut en réalité 11 commandes distinctes, chacune dédiée à un certain type d’opérations.

animate  Permet d’animer une séquence d’images.

compare  Permet d’établir les différences d’un point de vue mathématique et visuel entre une image et sa reproduction.

composite  Permet de superposer 2 images.

conjure  Est une commande capable d’interpéter et d’exécuter des scripts MSL (Magick Scripting Language)

convert  Est l’une des commandes les plus utiles : elle permet de réaliser toutes sortes de transformations sur une image (conversion de format, redimensionnement, découpage, effets de flou, retournement, etc.); à noter qu’un nouveau fichier image est créé, l’original demeure intact.

display  Permet d’afficher une image ou une séquence d’images.

identify  Précise le format et les caractéristiques d’un ou plusieurs fichiers images.

import  est dédiée à la capture d’écran : elle permet de capturer une fenêtre, l’ensemble de l’écran ou bien n’importe quelle zone à l’écran (à sélectionner à la souris).
mogrify est similaire à la commande convert, à ceci près que l’image originale est cette fois-ci écrasée par sa nouvelle version.

See Also:

Resizing image files

montage permet de créer une image composite à partir de plusieurs images (à la manière d’une mosaïque)

stream est destinée au streaming d’un ou plusieurs pixels d’une image, ou d’une portion de l’image, dans le format qui vous convient.

libpng image library

See Also:

- http://libpng.git.sourceforge.net/git/gitweb.cgi?p=libpng/libpng;a=summary
- http://libpng.git.sourceforge.net/git/gitweb.cgi?p=libpng/libpng
- http://code.google.com/p/skia/source/checkout

The PNG module has been designed to handle multiple sessions at one time, to be easily modifiable, to be portable to the vast majority of machines (ANSI, K&R, 16-, 32-, and 64-bit) available, and to be easy to use. The ultimate goal of libpng is to promote the acceptance of the PNG file format in whatever way possible. While there is still work to be done (see the TODO file), libpng should cover the majority of the needs of its users.

Libpng uses zlib for its compression and decompression of PNG files. Further information about zlib, and the latest version of zlib, can be found at the zlib home page, http://www.info-zip.org/pub/infozip/zlib/. The zlib compression utility is a general purpose utility that is useful for more than PNG files, and can be used without libpng. See the documentation delivered with zlib for more details. You can usually find the source files for the zlib utility wherever you find the libpng source files.

About PNG, and the latest version of libpng, can be found at the PNG home page, <http://www.libpng.org/pub/png/>.
pngstatus

See Also:

http://www.libpng.org/pub/png/libpng.html

libpng is the official PNG reference library. It supports almost all PNG features, is extensible, and has been extensively
tested for over 14 years. The home site for development versions (i.e., may be buggy or subject to change or include
experimental features) is http://libpng.sourceforge.net/ and the place to go for questions about the library is the png-
mng-implement mailing list.

libpng is available as ANSI C (C89) source code and requires zlib 1.0.4 or later (1.2.3 recommended for performance
and security reasons).

PNG links

See Also:

• http://www.libpng.org/pub/png/pngfaq.htm

libpng test suite

See Also:

http://www.schaik.com/pngsuite/

opencv image library

See Also:

opencv cheatsheet

OpenCV Overview: > 500 functions

goopy

See Also:
• http://sourceforge.net/projects/opencvlibrary/
• http://opencv.willowgarage.com/wiki/
• http://opencv.willowgarage.com/documentation/cpp/index.html (doc C++)
• http://opencv.willowgarage.com/documentation/python/cookbook.html (doc python cookbook)
• http://opencv.willowgarage.com/documentation/c/index.html (doc c)

OpenCV tree sources

| CMakeLists.txt
| cmake_uninstall.cmake.in
| cvconfig.h.cmake
| index.rst
| OpenCV.mk.in
| opencv.pc.cmake.in
| OpenCVConfig.cmake.in
| OpenCVFindIPP.cmake
| OpenCVFindLATEX.cmake
| OpenCVFindOpenEXR.cmake
| OpenCVFindOpenNI.cmake
| OpenCVFindPkgConfig.cmake
| OpenCVModule.cmake
| OpenCVPCHSupport.cmake
| Package.cmake.in
| README

|--3rdparty
| | CMakeLists.txt
| | readme.txt
| |
| |--ffmpeg
| | | CMakeLists.txt
| | | ffopencv.c
| | | make.bat
| | | opencv_ffmpeg.dll
| | | opencv_ffmpeg_64.dll
| | | readme.txt
| |
| |--ilmimf
| | | LICENSE
| | | README
| |
|--include
| | | msc_inttypes.h
| | | mscstdint.h
| | |
| | |--dshow
| | | | amvideo.h
| | | | audevcod.h
| | | | bdatypes.h
| | | | control.h
| | | | ddraw.h
| | | | dshow.h
| | | | dsound.h
| | | | dvdmedia.h
| | | | errors.h
Development tools, Release 2012.06.18

| evcode.h |
| ksuuids.h |
| strmif.h |
| uuids.h |
| _mingw_dxhelper.h |
| _mingw_unicode.h |
\---ffmpeg_
+---libavcodec
| avcodec.h |
| avfft.h |
| dxva2.h |
| opt.h |
| vaapi.h |
| vdpau.h |
| xvmc.h |
+---libavdevice
| avdevice.h |
+---libavformat
| avformat.h |
| avio.h |
+---libavutil
| adler32.h |
| attributes.h |
| avconfig.h |
| avstring.h |
| avutil.h |
| base64.h |
| common.h |
| crc.h |
| error.h |
| fifo.h |
| intfloat_readwrite.h |
| log.h |
| lzo.h |
| mathematics.h |
| md5.h |
| mem.h |
| pixdesc.h |
| pixfmt.h |
| rational.h |
| shal.h |
\---libswscale
| swscale.h |
+---lib
| libavcodec.a |
| libavcodec64.a |
| libavcore64.a |
| libavdevice.a |
| libavdevice64.a |
| libavformat.a |
| libavformat64.a |
| libavutil.a |

2.20. Multimedia 747
Development tools, Release 2012.06.18

| libavutil64.a |
| libcdl64.a |
| libgcc64.a |
| libgcc_.a |
| libmingw64.a |
| libmingw_.a |
| libnative_camera_r2.2.2.so |
| libnative_camera_r2.3.3.so |
| libswscale.a |
| libswscale64.a |
| libwsock3264.a |
| libwsock32_.a |

---libjasper

| CMakeLists.txt |
| jas_cm.c |
| jas_debug.c |
| jas_getopt.c |
| jas_icc.c |
| jas_iccdata.c |
| jas_image.c |
| jas_init.c |
| jas_malloc.c |
| jas_seq.c |
| jas_stream.c |
| jas_string.c |
| jas_tmr.c |
| jas_tvp.c |
| jas_version.c |
| jp2_cod.c |
| jp2_cod.h |
| jp2_dec.c |
| jp2_dec.h |
| jp2_enc.c |
| jpc_bs.c |
| jpc_bs.h |
| jpc_cod.h |
| jpc_cs.c |
| jpc_cs.h |
| jpc_dec.c |
| jpc_dec.h |
| jpc_enc.c |
| jpc_enc.h |
| jpc_fix.c |
| jpc_flt.h |
| jpc_math.c |
| jpc_math.h |
| jpc_mct.c |
| jpc_mct.h |
| jpc_mqcod.c |
| jpc_mqcod.h |
| jpc_mqdec.c |
| jpc_mqdec.h |
| jpc_mqenc.c |
| jpc_mqenc.h |
| jpc_qmfb.c |
| jpc_qmfb.h |
| jpc_tlcod.c |
Development tools, Release 2012.06.18

jcphuff.c
jcprepct.c
jcsample.c
jctrans.c
jdapimin.c
jdapistd.c
jdatadst.c
jdatasrc.c
jdcoeftct.c
jdcolor.c
jdct.h
jddctmgr.c
jdhuff.c
jdhuff.h
jinput.c
jmaininct.c
jmarker.c
jmaster.c
jmerge.c
jpdpuff.c
jdpstc.tct.c
jdsample.c
jdtrans.c
jerror.c
jerror.h
jfdctflt.c
jfdctfst.c
jfdctint.c
jjdctflt.c
jjdctfst.c
jjdctint.c
jjdctred.c
jinclude.h
jmemansi.c
jmemmgr.c
jmemsys.h
jmorecfg.h
jpegint.h
jpeglib.h
jquant1.c
jquant2.c
jutils.c
jversion.h
README
transupp.c
transupp.h

---libpng
CMakeLists.txt
png.c
png.h
pngconf.h
pngerror.c
pngget.c
pngmem.c
pngpriv.h
pngread.c

Chapter 2. Development
Development tools, Release 2012.06.18

|   |   | pngrio.c   |
|   |   | pngrtran.c  |
|   |   | pngrutil.c  |
|   |   | pngset.c   |
|   |   | pngtest.c   |
|   |   | pngtest.png |
|   |   | pngtrans.c  |
|   |   | pngwio.c    |
|   |   | pngwrite.c  |
|   |   | pngwtran.c  |
|   |   | pngwutil.c  |
|   |   | README      |
|   |   |   | ---libtiff |
|   |   |   |   | CMakeLists.txt |
|   |   |   |   | t4.h    |
|   |   |   |   | tiff.h   |
|   |   |   |   | tiffconf.h |
|   |   |   |   | tiffio.h  |
|   |   |   |   | tiffio.hxx |
|   |   |   |   | tiffiop.h |
|   |   |   |   | tiffvers.h |
|   |   |   |   | tif_apple.c |
|   |   |   |   | tif_aux.c  |
|   |   |   |   | tif_close.c |
|   |   |   |   | tif_codec.c |
|   |   |   |   | tif_color.c |
|   |   |   |   | tif_compress.c |
|   |   |   |   | tif_config.h |
|   |   |   |   | tif_dir.c   |
|   |   |   |   | tif_dir.h   |
|   |   |   |   | tif_dirinfo.c |
|   |   |   |   | tif_dirread.c |
|   |   |   |   | tif_dirwrite.c |
|   |   |   |   | tif_dumpmode.c |
|   |   |   |   | tif_error.c |
|   |   |   |   | tif_extension.c |
|   |   |   |   | tif_fax3.c   |
|   |   |   |   | tif_fax3.h   |
|   |   |   |   | tif_fax3sm.c |
|   |   |   |   | tif_flush.c  |
|   |   |   |   | tif_getimage.c |
|   |   |   |   | tif_jbig.c   |
|   |   |   |   | tif_jpeg.c   |
|   |   |   |   | tif_luv.c    |
|   |   |   |   | tif_lzw.c    |
|   |   |   |   | tif_next.c   |
|   |   |   |   | tif_ojpeg.c  |
|   |   |   |   | tif_open.c   |
|   |   |   |   | tif_packbits.c|
|   |   |   |   | tif_pixarlog.c|
|   |   |   |   | tif_predict.c |
|   |   |   |   | tif_predict.h |
|   |   |   |   | tif_print.c  |
|   |   |   |   | tif_read.c   |
|   |   |   |   | tif_stream.cxx|
|   |   |   |   | tif_strip.c   |
|   |   |   |   | tif_swab.c   |

2.20. Multimedia
Chapter 2. Development
android-cv-typemaps.i
android-cv.i
buffers.i
Calibration.cpp
Calibration.i
CMakeLists.txt
cv.i
glcamera.i
gl_code.cpp
image_pool.cpp
image_pool.i
nocopy.i
yuv2rgb16tab.c
yuv2rgb_neon.c
yuv420rgb888.s
yuv420rgb888c.c
yuv420sp2rgb.c

include
android_logger.h
Calibration.h
glcamera.h
image_pool.h
yuv2rgb.h
yuv420sp2rgb.h

res
drawable-mdpi
cameraback.jpg

layout
calibrationviewer.xml
camera.xml
camerasettings.xml
chesssizer.xml

values
attrs.xml
chessnumbers.xml
settingnumbers.xml
strings.xml

src
calom
opencv
OpenCV.java

calibration
CalibrationViewer.java
Calibrator.java
ChessBoardChooser.java

services
CalibrationService.java

camera
CameraActivity.java
CameraButtonsHandler.java
2.20. Multimedia

| sample.local.env.mk |
| +---jni |
| | Android.mk |
| | Application.mk |
| | foobar.i |
| | TestBar.cpp |
| | TestBar.h |
| +---res |
| | +---drawable-hdpi |
| | | icon.png |
| | +---drawable-ldpi |
| | | icon.png |
| | +---drawable-mdpi |
| | | icon.png |
| | \---values |
| | | strings.xml |
| \---src |
| \---com |
| \---foo |
| \---bar |
| | FooBar.java |
| +---CVCamera |
| | AndroidManifest.xml |
| | CMakeLists.txt |
| | default.properties |
| | project_create.sh |
| | README.txt |
| | uninstall.phone.sh |
| +---jni |
| | CMakeLists.txt |
| | cvcamera.i |
| | Processor.cpp |
| | Processor.h |
| | Processor.i |
| +---res |
| | +---drawable-hdpi |
| | | icon.png |
| | +---drawable-ldpi |
| | | icon.png |
| | +---drawable-mdpi |
| | | icon.png |
| | +---layout |
| | | main.xml |
| | \---values |
| | | strings.xml |
Development tools, Release 2012.06.18

| |--src
  |   |--com
  |   |   |--theveganrobot
  |   |   |   |--cvcamera
  |   |   |   |   CVCamera.java
  |   +---HelloAndroid
  |       |   CMakeLists.txt
  |       |   cmake_android.cmd
  |       |   cmake_android.sh
  |       |   main.cpp
  |       |   run.cmd
  |       |   run.sh
  |   \---OpenCV_SAMPLE
  |       |   AndroidManifest.xml
  |       |   CMakeLists.txt
  |       |   cmake_android.cmd
  |       |   cmake_android_neon.sh
  |       |   default.properties
  |       |   project_create.sh
  |       +---jni
  |           |   CMakeLists.txt
  |           |   cvsample.cpp
  |           |   cvsample.h
  |           |   OpenCV_SAMPLE.i
  |       +---res
  |           +---drawable-hdpi
  |               |   icon.png
  |           +---drawable-ldpi
  |               |   icon.png
  |           +---drawable-mdpi
  |               |   icon.png
  |           +---layout
  |               |   main.xml
  |           +---menu
  |               |   sample_menu.xml
  |           \---values
  |               |   strings.xml
  \---scripts
     build.cmd
     cmake_android.cmd
     cmake_android.sh
     cmake_android_armeabi.sh
Development tools, Release 2012.06.18

```
cmake_android_neon.sh
wincfg.cmd.tmpl
```

```
+---data
| | CMakeLists.txt
| | readme.txt
| |
| +---haarcascades
| | haarcascade_eye.xml
| | haarcascade_eye_tree_eyeglasses.xml
| | haarcascade_frontalface_alt.xml
| | haarcascade_frontalface_alt2.xml
| | haarcascade_frontalface_alt_tree.xml
| | haarcascade_frontalface_default.xml
| | haarcascade_fullbody.xml
| | haarcascade_lefteye_2splits.xml
| | haarcascade_lowerbody.xml
| | haarcascade_mcs_eyepair_big.xml
| | haarcascade_mcs_eyepair_small.xml
| | haarcascade_mcs_lefteye.xml
| | haarcascade_mcs_mouth.xml
| | haarcascade_mcs_nose.xml
| | haarcascade_mcs_righteye.xml
| | haarcascade_mcs_upperbody.xml
| | haarcascade_profileface.xml
| | haarcascade_righteye_2splits.xml
| | haarcascade_upperbody.xml
| |
| \---lbpcascades
| | lbpcascade_frontalface.xml
| |
| +---doc
| | acircles_pattern.png
| | check_docs.py
| | check_docs_whitelist.txt
| | CMakeLists.txt
| | conf.py
| | Doxyfile.in
| | haartraining.htm
| | license.txt
| | mymath.sty
| | ocv.py
| | opencv-logo.png
| | opencv-logo2.png
| | opencv.bib
| | opencv.ico
| | opencv.jpg
| | opencv2refman.pdf
| | opencv_cheatsheet.pdf
| | opencv_cheatsheet.tex
| | opencv_tutorials.pdf
| | opencv_user.pdf
| | packaging.txt
| | pattern.png
| | reformat.py
| |
| +---pattern_tools
| | gen_pattern.py
```

2.20. Multimedia
Development tools, Release 2012.06.18

2.20. Multimedia

noContent.rst

README.txt

features2d

\-table_of_content_features2d

\-table_of_content_features2d.rst

general

\-table_of_content_general

\-table_of_content_general.rst

gpu

\-table_of_content_gpu

\-table_of_content_gpu.rst

highgui

\-table_of_content_highgui

\-table_of_content_highgui.rst

\-images

Adding_Trackbars_Tutorial_Cover.png

\-trackbar

\-trackbar.rst

\-images

Adding_Trackbars_Tutorial_Cover.png

Adding_Trackbars_Tutorial_Result_0.png

Adding_Trackbars_Tutorial_Result_1.png

Adding_Trackbars_Tutorial_Result_1a.png

Adding_Trackbars_Tutorial_Result_1b.png

Adding_Trackbars_Tutorial_Trackbar.png

\-images

calib3d.png

core.png

feature2D.png

general.png

gpu.png

highgui.png

imgproc.png

introduction.png

ml.png

objdetect.png

video.png

\-imgproc

\-erosion_dilatation

\-erosion_dilatation.rst

\-images

Morphology_1_Tutorial_Cover.png

Morphology_1_Tutorial_Dilation_Result.png

Morphology_1_Tutorial_Erosion_Result.png

Morphology_1_Tutorial_Original_Image.png

Morphology_1_Tutorial_Theory_Dilation.png

Morphology_1_Tutorial_Theory_Erosion.png

Morphology_1_Tutorial_Theory_Original_Image.png

---multimedia

images
Development tools, Release 2012.06.18

2.20. Multimedia
Development tools, Release 2012.06.18

| | | solvepnp.cpp
| | | stereobm.cpp
| | | stereogc.cpp
| | | stereosgbm.cpp
| | | triangulate.cpp
| | | _modelest.h
| |
| \---test
| | test_affine3d_estimator.cpp
| | test_cameralibration.cpp
| | test_cameralibration_artificial.cpp
| | test_cameralibration_badarg.cpp
| | test_chessboardgenerator.cpp
| | test_chessboardgenerator.hpp
| | test_chesscorners.cpp
| | test_chesscorners_badarg.cpp
| | test_chesscorners_timing.cpp
| | test_compose_rt.cpp
| | test_cornerssubpix.cpp
| | test_fundam.cpp
| | test_main.cpp
| | test_posit.cpp
| | test_precomp.cpp
| | test_precomp.hpp
| | test_reproject_image_to_3d.cpp
| | test_sovepnp_ransac.cpp
| | test_stereomatching.cpp
| | test_undistort.cpp
| | test_undistortBadarg.cpp
| |
| +++-contrib
| | | CMakeLists.txt
| |
| | +++-include
| | | \---opencv2
| | | \---contrib
| | | | contrib.hpp
| |
| | \---src
| | | adaptiveskindetector.cpp
| | | ba.cpp
| | | chamfermatching.cpp
| | | fuzzymeanhifttracker.cpp
| | | octree.cpp
| | | polyfit.cpp
| | | precomp.cpp
| | | precomp.hpp
| | | selfsimilarity.cpp
| | | spinimages.cpp
| | | stereovar.cpp
| |
| +++-core
| | | CMakeLists.txt
| |
| | +++-doc
| | | | basic_structures.rst
| | | | clustering.rst
| | | | core.rst

2.20. Multimedia
Development tools, Release 2012.06.18

| | test_math.cpp
| | test_operations.cpp
| | test_precomp.cpp
| | test_precomp.hpp
| | test_rand.cpp
| |
| +---features2d
| | CMakeLists.txt
| |
| +---doc
| | common_interfaces_of_descriptor_extractors.rst
| | common_interfaces_of_descriptor Matchers.rst
| | common_interfaces_of_feature Detectors.rst
| | common_interfaces_of_generic_descriptorMatchers.rst
| | drawing_function_of_keypoints_and_matches.rst
| | features2d.rst
| | feature_detection_and_description.rst
| | object_categorization.rst
| |
| +---include
| | \---opencv2
| | \---features2d
| | features2d.hpp
| |
| +---src
| | bagofwords.cpp
| | blobdetector.cpp
| | brief.cpp
| | calondere.cpp
| | descriptors.cpp
| | detectors.cpp
| | draw.cpp
| | dynamic.cpp
| | evaluation.cpp
| | fast.cpp
| | generated_16.i
| | generated_32.i
| | generated_64.i
| | keypoint.cpp
| | matchers.cpp
| | mser.cpp
| | oneway.cpp
| | orb.cpp
| | orb_pattern.hpp
| | planardetect.cpp
| | precomp.cpp
| | precomp.hpp
| | sift.cpp
| | stardetector.cpp
| | surf.cpp
| | test_pairs.txt
| |
| \---test
| | test_bruteforcematcher.cpp
| | test_detectors.cpp
| | test_fast.cpp
| | test_features2d.cpp
| | test_main.cpp

2.20. Multimedia 767
Development tools, Release 2012.06.18

```plaintext
| test_mser.cpp  
| test_nearestneighbors.cpp  
| test_precomp.cpp  
| test_precomp.hpp  

---flann
  | CMakeLists.txt  
+---include  
    
    \---opencv2  
      \---flann  
        | allocator.h  
        | all_indices.h  
        | autotuned_index.h  
        | composite_index.h  
        | dist.h  
        | flann.hpp  
        | flann_base.hpp  
        | general.h  
        | ground_truth.h  
        | hdf5.h  
        | heap.h  
        | index_testing.h  
        | kdtree_index.h  
        | kmeans_index.h  
        | linear_index.h  
        | logger.h  
        | matrix.h  
        | nn_index.h  
        | object_factory.h  
        | random.h  
        | result_set.h  
        | sampling.h  
        | saving.h  
        | simplex_downhill.h  
        | timer.h  
    
\---src  
  | flann.cpp  
  | precomp.cpp  
  | precomp.hpp  

---gpu  
  | CMakeLists.txt  
+---doc  
    | camera_calibration_and_3d_reconstruction.rst  
    | data_structures.rst  
    | feature_detection_and_description.rst  
    | gpu.rst  
    | image_filtering.rst  
    | image_processing.rst  
    | initialization_and_information.rst  
    | introduction.rst  
    | matrix_reductions.rst  
    | object_detection.rst  
    | operations_on_matrices.rst  
    | per_element_operations.rst
```
+++-include
| | \+-opencv2
| | \+-gpu
| | devmem2d.hpp
| | gpu.hpp
| | matrix_operations.hpp
| | stream_accessor.hpp

+++-src
| | | arithm.cpp
| | | bilateral_filter.cpp
| | | blend.cpp
| | | brute_force_matcher.cpp
| | | calib3d.cpp
| | | cascadeclassifier.cpp
| | | color.cpp
| | | cudastream.cpp
| | | element_operations.cpp
| | | error.cpp
| | | filtering.cpp
| | | graphcuts.cpp
| | | hog.cpp
| | | imgproc_gpu.cpp
| | | initialization.cpp
| | | match_template.cpp
| | | matrix_operations.cpp
| | | matrix_reductions.cpp
| | | mssegmentation.cpp
| | | precomp.cpp
| | | precomp.hpp
| | | speckle_filtering.cpp
| | | split_merge.cpp
| | | stereobm.cpp
| | | stereobp.cpp
| | | stereocsbp.cpp
| | | surf.cpp

+++-cuda
| | | blend.cu
| | | brute_force_matcher.cu
| | | calib3d.cu
| | | color.cu
| | | element_operations.cu
| | | filters.cu
| | | hog.cu
| | | imgproc.cu
| | | internal_shared.hpp
| | | match_template.cu
| | | mathfunc.cu
| | | matrix_operations.cu
| | | matrix_reductions.cu
| | | safe_call.hpp
| | | split_merge.cu
| | | stereobm.cu
| | | stereobp.cu
| | | stereocsbp.cu
| | | surf.cu

2.20. Multimedia
Chapter 2. Development
2.20. Multimedia

Development tools, Release 2012.06.18

|-- files_Qt
  |   stylestheet_trackbar.qss
  |   Milky
  |   README.txt
  +--48
     |   1.png
     |   10.png
     |   100.png
     |   101.png
     |   102.png
     |   103.png
     |   104.png
     |   105.png
     |   106.png
     |   107.png
     |   108.png
     |   109.png
     |   11.png
     |   110.png
     |   111.png
     |   112.png
     |   113.png
     |   114.png
     |   115.png
     |   116.png
     |   117.png
     |   118.png
     |   119.png
     |   12.png
     |   120.png
     |   121.png
     |   122.png
     |   123.png
     |   124.png
     |   125.png
     |   126.png
     |   127.png
     |   128.png
     |   129.png
     |   13.png
     |   130.png
     |   131.png
     |   14.png
     |   15.png
     |   16.png
     |   17.png
     |   18.png
     |   19.png
     |   2.png
     |   20.png
     |   21.png
     |   22.png
     |   23.png
     |   24.png
     |   25.png
| | | 129.png |
| | | 13.png |
| | | 130.png |
| | | 131.png |
| | | 14.png |
| | | 15.png |
| | | 16.png |
| | | 17.png |
| | | 18.png |
| | | 19.png |
| | | 2.png |
| | | 20.png |
| | | 21.png |
| | | 22.png |
| | | 23.png |
| | | 24.png |
| | | 25.png |
| | | 26.png |
| | | 27.png |
| | | 28.png |
| | | 29.png |
| | | 3.png |
| | | 30.png |
| | | 31.png |
| | | 32.png |
| | | 33.png |
| | | 34.png |
| | | 35.png |
| | | 36.png |
| | | 37.png |
| | | 38.png |
| | | 39.png |
| | | 4.png |
| | | 40.png |
| | | 41.png |
| | | 42.png |
| | | 43.png |
| | | 44.png |
| | | 45.png |
| | | 46.png |
| | | 47.png |
| | | 48.png |
| | | 49.png |
| | | 5.png |
| | | 50.png |
| | | 51.png |
| | | 52.png |
| | | 53.png |
| | | 54.png |
| | | 55.png |
| | | 56.png |
| | | 57.png |
| | | 58.png |
| | | 59.png |
| | | 6.png |
| | | 60.png |
| | | 61.png |
| | | 62.png |
2.20. Multimedia

Development tools, Release 2012.06.18
Development tools, Release 2012.06.18

| | | | | imgproc.rst
| | | | | miscellaneous_transformations.rst
| | | | | motion_analysis_and_object_tracking.rst
| | | | | object_detection.rst
| | | | | planar_subdivisions.rst
| | | | | structural_analysis_and_shape_descriptors.rst
| |
| | | | 
| | | | \---pics
| | | | backprojectpatch.png
| | | | bayer.png
| | | | boundingrect.png
| | | | building.jpg
| | | | contoursecarea.png
| | | | contoursubpix.png
| | | | defects.png
| | | | houghp.png
| | | | integral.png
| | | | inv_logpolar.jpg
| | | | logpolar.jpg
| | | | minareabox.png
| | | | pointpolygon.png
| | | | quadedge.png
| | | | subdiv.png
| | | | threshold.png
| |
| | | | 
| | | | +++-include
| | | | | \---opencv2
| | | | | | \---imgproc
| | | | | | | imgproc.hpp
| | | | | | | imgproc_c.h
| | | | | | | types_c.h
| | | | | 
| | | | | +++-src
| | | | | | accum.cpp
| | | | | | approx.cpp
| | | | | | canny.cpp
| | | | | | color.cpp
| | | | | | contours.cpp
| | | | | | convhull.cpp
| | | | | | corner.cpp
| | | | | | cornersubpix.cpp
| | | | | | deriv.cpp
| | | | | | distransform.cpp
| | | | | | emd.cpp
| | | | | | featureselect.cpp
| | | | | | featuretree.cpp
| | | | | | filter.cpp
| | | | | | floodfill.cpp
| | | | | | gcgraph.hpp
| | | | | | geometry.cpp
| | | | | | grabcut.cpp
| | | | | | histogram.cpp
| | | | | | hough.cpp
| | | | | | imgwarp.cpp
| | | | | | inpaint.cpp
| | | | | | kdtree.cpp
| | | | | | linefit.cpp
| | | | | | lsh.cpp

778 Chapter 2. Development
Development tools, Release 2012.06.18

- estimate.cpp
- gbt.cpp
- inner_functions.cpp
- knearest.cpp
- nbayes.cpp
- precomp.cpp
- precomp.hpp
- rtree.cpp
- svm.cpp
- testset.cpp
- tree.cpp

|--test
  |--test_emknearestkmeans.cpp
  |--test_gbttest.cpp
  |--test_main.cpp
  |--test_mltests.cpp
  |--test_mltests2.cpp
  |--test_precomp.cpp
  |--test_precomp.hpp
  |--test_save_load.cpp

|--objdetect
  |--CMakeLists.txt
  ||--doc
  |  ||--cascade_classification.rst
  |  ||--objdetect.rst
  |  ||--pics
  |  |  ||--haarfeatures.png
  ||--include
  |  ||--opencv2
  |  |  ||--objdetect
  |  |  |  ||--objdetect.hpp
  ||--src
  |  ||--cascadedetect.cpp
  |  ||--datamatrix.cpp
  |  ||--distancetransform.cpp
  |  ||--featurepyramid.cpp
  |  ||--fft.cpp
  |  ||--haar.cpp
  |  ||--hog.cpp
  |  ||--latentsvm.cpp
  |  ||--latentsvmdetector.cpp
  |  ||--lsvmparser.cpp
  |  ||--lsvm_tbbversion.cpp
  |  ||--matching.cpp
  |  ||--planardetect.cpp
  |  ||--precomp.cpp
  |  ||--precomp.hpp
  |  ||--resizeimg.cpp
  |  ||--routine.cpp
  |  ||--_latentsvm.h
  |  ||--_lsvmparser.h
  |  ||--_lsvm_distancetransform.h
Development tools, Release 2012.06.18

```
| | | _lsvm_error.h
| | | _lsvm_fft.h
| | | _lsvm_matching.h
| | | _lsvm_resizeing.h
| | | _lsvm_routine.h
| | | _lsvm_tbbversion.h
| | | _lsvm_types.h
| |
\---test
| | test_cascadeandhog.cpp
| | test_latentsvmdetector.cpp
| | test_main.cpp
| | test_precomp.cpp
| | test_precomp.hpp
|
| | | ---python
| | | | CMakeLists.txt
| | | | |
| | | | | |--src1
| | | | | | api
| | | | | | cv.cpp
| | | | | | defs
| | | | | | gen.py
| | | | |
| | | | | |--src2
| | | | | | cv2.cpp
| | | | | | gen2.py
| | | | | | hdr_parser.py
| | | | | | opencv_extra_api.hpp
| | | |
| | | \---test
| | | | calchist.py
| | | | camera_calibration.py
| | | | findstereocorrespondence.py
| | | | goodfeatures.py
| | | | leak1.py
| | | | leak2.py
| | | | leak3.py
| | | | leak4.py
| | | | precornerdetect.py
| | | | test.py
| | | | tickets.py
| | | | ticket_6.py
| | | | transformations.py
| | | |
| | | | | | |--stitching
| | | | | | | autocalib.cpp
| | | | | | | autocalib.hpp
| | | | | | | blenders.cpp
| | | | | | | blenders.hpp
| | | | | | | CMakeLists.txt
| | | | | | | exposure_compensate.cpp
| | | | | | | exposure_compensate.hpp
| | | | | | | main.cpp
| | | | | | | matchers.cpp
| | | | | | | matchers.hpp
| | | | | | | motion_estimators.cpp
| | | | | | | motion_estimators.hpp
```

2.20. Multimedia
Development tools, Release 2012.06.18

| | precomp.cpp
| | precomp.hpp
| | seam_finders.cpp
| | seam_finders.hpp
| | util.cpp
| | util.hpp
| | util_inl.hpp
| | warpers.cpp
| | warpers.hpp
| | warpers_inl.hpp
| |
+---traincascade
  | | boost.cpp
  | | boost.h
  | | cascadeclassifier.cpp
  | | cascadeclassifier.h
  | | CMakeLists.txt
  | | features.cpp
  | | haarfeatures.cpp
  | | haarfeatures.h
  | | imagestorage.cpp
  | | imagestorage.h
  | | lbpfeatures.cpp
  | | lbpfeatures.h
  | | traincascade.cpp
  | | traincascade_features.h
|
+---ts
  | | CMakeLists.txt
  | |
  | +---include
  | | \---opencv2
  | | \---ts
  | |  ts.hpp
  | |  ts/gtest.h
  | |
  | \---src
  | | precomp.cpp
  | | precomp.hpp
  | | ts.cpp
  | | ts_arrtest.cpp
  | | ts_func.cpp
  | | ts/gtest.cpp
  |
\---video
  | CMakeLists.txt
  |
  | +---doc
  | | motion_analysis_and_object_tracking.rst
  | | video.rst
  |
  | +---include
  | | \---opencv2
  | | \---video
  | |  background_segm.hpp
  | |  tracking.hpp
  |
  | +---src
| | bgfg_acmmm2003.cpp
| | bgfg_codebook.cpp
| | bgfg_common.cpp
| | bgfg_gaussmix.cpp
| | bgfg_gaussmix2.cpp
| | camshift.cpp
| | kalman.cpp
| | lkpyramid.cpp
| | motempl.cpp
| | optflowbm.cpp
| | optflowgf.cpp
| | optflowhs.cpp
| | optflowlk.cpp
| | precomp.cpp
| | precomp.hpp
| |
\---test
| test_accum.cpp
| test_camshift.cpp
| test_estimaterigid.cpp
| test_kalman.cpp
| test_main.cpp
| test_motiontemplates.cpp
| test_optflow.cpp
| test_optflowpyrlk.cpp
| test_precomp.cpp
| test_precomp.hpp
|
\---samples
| CMakeLists.txt
|
++-c
| | adaptiveskindetector.cpp
| | agaricus-lepiota.data
| | airplane.jpg
| | baboon.jpg
| | baboon200.jpg
| | baboon200_rotated.jpg
| | bgfg_codebook.cpp
| | blobtrack_sample.cpp
| | box.png
| | box_in_scene.png
| | build_all.sh
| | calonder_params.xml
| | cat.jpg
| | cat.xml
| | CMakeLists.txt
| | contours.c
| | convert_cascade.c
| | cvsample.dsp
| | delaunay.c
| | facedetect.cmd
| | facedetect.cpp
| | fback_c.c
| | find_obj.cpp
| | find_obj_calonder.cpp
| | find_obj_ferns.cpp
| | fruits.jpg

2.20. Multimedia
Development tools, Release 2012.06.18

| intrinsics.yml |
| JCB.png |
| latentsvmdetect.cpp |
| lena.jpg |
| morphology.c |
| motempl.c |
| mser_sample.cpp |
| mushroom.cpp |
| one_way_sample.cpp |
| one_way_train_0000.jpg |
| one_way_train_0001.jpg |
| one_way_train_images.txt |
| polar_transforms.c |
| puzzle.png |
| pyramid_segmentation.c |
| scene_l.bmp |
| scene_r.bmp |
| stuff.jpg |
| tree.avi |
| tree_engine.cpp |
| waveform.data |
| \---example_cmake |
| CMakeLists.txt |
| minarea.c |
| README.txt |
| \+++cpp |
| 3calibration.cpp |
| baboon.jpg |
| bagofwords_classification.cpp |
| bgfg_segm.cpp |
| brief_match_test.cpp |
| build3dmodel.cpp |
| building.jpg |
| calibration.cpp |
| calibration_artificial.cpp |
| camshiftdemo.cpp |
| chamfer.cpp |
| CMakeLists.txt |
| connected_components.cpp |
| contours2.cpp |
| convexhull.cpp |
| cout_mat.cpp |
| demhist.cpp |
| descriptor_extractor_matcher.cpp |
| detector_descriptor_evaluation.cpp |
| dft.cpp |
| distrans.cpp |
| drawing.cpp |
| edge.cpp |
| em.cpp |
| fback.cpp |
| fern_params.xml |
| ffilldemo.cpp |
| filestorage.cpp |
| fitellipse.cpp |
| fruits.jpg |

Chapter 2. Development
development tools, release 2012.06.18

|     | generic_descriptor_match.cpp
|     | grabcut.cpp
|     | houghlines.cpp
|     | image.cpp
|     | imagelist_creator.cpp
|     | inpaint.cpp
|     | kalman.cpp
|     | kinect_maps.cpp
|     | kmeans.cpp
|     | laplace.cpp
|     | left01.jpg
|     | left02.jpg
|     | left03.jpg
|     | left04.jpg
|     | left05.jpg
|     | left06.jpg
|     | left07.jpg
|     | left08.jpg
|     | left09.jpg
|     | left11.jpg
|     | left12.jpg
|     | left13.jpg
|     | left14.jpg
|     | lena.jpg
|     | letter-recognition.data
|     | letter_recog.cpp
|     | lkdemo.cpp
|     | logo.png
|     | logo_in_clutter.png
|     | matcher_simple.cpp
|     | matching_to_many_images.cpp
|     | meanshift_segmentation.cpp
|     | minarea.cpp
|     | morphology2.cpp
|     | multicascadeclassifier.cpp
|     | peopledetect.cpp
|     | pic1.png
|     | pic2.png
|     | pic3.png
|     | pic4.png
|     | pic5.png
|     | pic6.png
|     | points_classifier.cpp
|     | right01.jpg
|     | right02.jpg
|     | right03.jpg
|     | right04.jpg
|     | right05.jpg
|     | right06.jpg
|     | right07.jpg
|     | right08.jpg
|     | right09.jpg
|     | right11.jpg
|     | right12.jpg
|     | right13.jpg
|     | right14.jpg
|     | segment_objects.cpp
|     | select3dobj.cpp
Development tools, Release 2012.06.18

| squares.cpp
| starter_imagelist.cpp
| starter_video.cpp
| stereo_calib.cpp
| stereo_calib.xml
| stereo_match.cpp
| stuff.jpg
| tsukuba_l.png
| tsukuba_r.png
| video_dmtx.cpp
| video_homography.cpp
| watershed.cpp

---matching_to_many_images
| query.png

---train
| 1.png
| 2.png
| 3.png
| trainImages.txt

---Qt_sample
| CMakeLists.txt
| cube4.avi
| main.cpp

---tutorial_code
| Basic
| AddingImages.cpp
| AddingImagesTrackbar.cpp
| BasicLinearTransforms.cpp
| BasicLinearTransformsTrackbar.cpp
| DisplayImage.cpp
| Drawing_1.cpp
| Drawing_2.cpp
| LoadSaveImage.cpp

| images
| cat.jpg
| HappyFish.jpg
| lena.png
| LinuxLogo.jpg
| WindowsLogo.jpg

---Image_Processing
| Morphology_1.cpp
| Smoothing.cpp

---gpu
| aloeL.jpg
| aloeR.jpg
| cascadeclassifier.cpp
| cascadeclassifier_nvidia_api.cpp
| CMakeLists.txt
| driver_api_multi.cpp
| driver_api_stereo_multi.cpp
| hog.cpp

Chapter 2. Development
Development tools, Release 2012.06.18

| squares.py |
| watershed.py |
| \---python2 |
| browse.py |
| calibrate.py |
| coherence.py |
| color_histogram.py |
| common.py |
| edge.py |
| gaussian_mix.py |
| letter_recog.py |
| obj_detect.py |
| video.py |
| _coverage.py |

visit image library  See Also:

- https://wci.llnl.gov/codes/visit/about.html

About VisIt  VisIt is a free interactive parallel visualization and graphical analysis tool for viewing scientific data on Unix and PC platforms.

Users can quickly generate visualizations from their data, animate them through time, manipulate them, and save the resulting images for presentations.

VisIt contains a rich set of visualization features so that you can view your data in a variety of ways. It can be used to visualize scalar and vector fields defined on two- and three-dimensional (2D and 3D) structured and unstructured meshes.

VisIt was designed to handle very large data set sizes in the terascale range and yet can also handle small data sets in the kilobyte range.

Multimedia images tools

images 2mandvd tool  See Also:

- http://2mandvd.tuxfamily.org/
- http://git.tuxfamily.org/2mandvd/develcode.git

Bienvenue sur le site internet du logiciel 2ManDVD.

2ManDVD est un outils développé en c++ avec la bibliothèque QT4. Avec 2ManDVD vous allez pouvoir créer simplement vos DVD vidéos, vos diaporamas et ce de manière totalement gratuite. Cet outils fonctionne sur plateforme Linux et Mac en 32 et en 64 bits.

Developppement  Vous pouvez aussi récupérer la dernière version (attention ceci est une version de développement) en tapant dans une console :

git clone git://git.tuxfamily.org/gitroot/2mandvd/develcode.git
cd develcode
qmake 2ManDVD.pro
make

Et hous voila en possession de la dernière version en cours de développement.
images image movie tool  See Also:

- http://imagemovie.sourceforge.net/
- http://www.youtube.com/watch?v=jpFU9Q8Ee-k
- https://sourceforge.net/projects/imagemovie/

With ImagEmovie you will be able to create a video from your pictures and furthermore to include one or more movies inside your video.

Many effects are possible, such as transitions between pictures (77 available) and text animations (20 available), 5 different kind of slide motions (Ken Burns effect) and some other effects are possible.

The audio file with theirs own sound effects can be attached to each slide.

Finally, the slideshow created can be exported in VOB (to be burn on a DVD), in OGV, FLV or MP4 (format compatible with the iPod).

Irfanview  See Also:

http://www.irfanview.com/

With Irfanview, we can visualize a great number of image formats:

- raw_format
- wsq format
- bmp format

Raw format  To visualize a raw format we have to input the size of the raw image.

images openshot tool  See Also:

- Video openshot editor

Il est possible de faire des diaporamas avec Openshot. Voir Linux Pratique numéro 68 (novembre/décembre 2011) p.15. par Yann Morère.

Dans un premier temps, on sélectionne toutes les photos que l’on désire voir apparaître dans le diaporama et on les ajoute au projet par un simple glisser-déposer. Ensuite on les sélectionne toutes puis à l’aide du clique droit, on choisit l’option ajouter à la timeline; une fenêtre de configuration s’ouvre nous permettant de choisir l’emplacement dans la timeline, les fondus et les transitions. Une fois le bouton Ajouter cliqué, le montage des images est réalisé automatiquement.

pencil  See Also:


Pencil can run on both GNU/Linux and Windows as a Firefox addon or a standalone application using XULRunner.

2.21 Open data

2.21.1 Open data

See Also:
Ifrap

- http://www.ifrap.org/Open-Data-Donnees-publiques,11925.html

Transparence France

e lobbying fait beaucoup parler, mais qu’en sait-on réellement ? Y a-t-il ou non une volonté de pluralisme dans les arguments écoutés ? Le lobbying visant à influencer les décisions publiques, mérite d’être mieux documenté et analysé pour faire avancer son débat. Une plus grande transparence et un cadre adapté aux relations entre les parlementaires et les acteurs rencontrés doivent être recherchés dès lors qu’un mandat électif est un contrat passé avec les Français, exercé sur fonds publics.

Transparence International France et Regards Citoyens ont convenu en juillet 2010 d’un partenariat original, pour livrer ensemble au débat public et aux Français, des informations nouvelles sur les rencontres connues entre députés et acteurs publics et privés. Si ces informations sont toutes publiques, elles sont présentées ici de façon inédite afin d’être accessibles à tous. Elles ont vocation à être affinées, dans un processus collaboratif, avec le concours souhaité de toutes les personnes intéressées par le fonctionnement de la vie publique.

See Also:

http://www.transparence-france.org/

Regards citoyens

- http://www.regardscitoyens.org/nosdeputes-fr/

Biens communs


Ekopedia

- http://fr.ekopedia.org/Accueil

Intercoop


Gnomunisme

- http://fr.issuepedia.org/Gnomunisme
Open data tools

Open spending  See Also:


OpenSpending is a project to make government finances easier to explore and understand.

It started out as “Where does my money go”, a platform to visualize the United Kingdom’s state finance, but has been renamed and restructured to allow arbitrary financial data to be loaded and displayed.

The key use case for the software is the site openspending.org which aims to tracker government finance around the world.

Links


Open source ecology

See Also:

- http://openfarmtech.org/wiki/Open_Source_Fab_Lab
- http://openfarmtech.org/wiki/Main_Page-FR

Open Source Ecology est un mouvement dédié à l’élaboration conjointe de technologies reproductibles, “open-source” et modernes pour des communautés villageoises résilientes.

En utilisant tout à la fois la permaculture et les ateliers de conception numérique pour la satisfaction des besoins de base, selon une méthodologie “open source” favorisant la reproduction à bas coût de l’ensemble des opérations, nous souhaitons aider chaque personne qui le souhaite à dépasser le stade de la survie et à évoluer vers la liberté.

Dans notre analyse, la plupart des technologies nécessaires à un mode de vie durable et plaisant peuvent se réduire au coût de la ferraille et du travail. Il y a un potentiel immense de transformation sociale dès lors que ces technologies seront pleinement développées de manière à construire des communautés auto-suffisantes reliées entre elles. Nous serons alors libérés des contraintes matérielles et aptes à nous réaliser nous-mêmes.

Bien sûr, il s’agit d’une tâche ambitieuse, mais nous avons déjà accompli beaucoup et nos progrès sont rapides. Nous mettons la théorie en œuvre à Factor e Farm, notre installation à la campagne. Nos moyens d’atteindre ces objectifs sont minutieusement détaillés dans le “Global Village Construction Set” ainsi que dans nos “Propositions pour une écologie open-source” (OSE Proposal).

Open ODF

See Also:

The Open Document Format for Office Applications (also known as OpenDocument or ODF) is an XML-based file format for representing electronic documents such as spreadsheets, charts, presentations and word processing documents.

While the specifications were originally developed by Sun Microsystems, the standard was developed by the OASIS Open Document Format for Office Applications (OpenDocument) TC - OASIS ODF TC,[2] committee of the Organization for the Advancement of Structured Information Standards (OASIS) consortium and based on the XML format originally created and implemented by the OpenOffice.org office suite (see OpenOffice.org XML).

In addition to being an OASIS standard, it is published (in one of its version 1.0 manifestations) as an ISO/IEC international standard, ISO/IEC 26300:2006 Open Document Format for Office Applications (OpenDocument) v1.0.[3]

The most common filename extensions used for OpenDocument documents are:

- .odt for word processing (text) documents
- .ods for spreadsheets
- .odp for presentations
- .odb for databases
- .odg for graphics
- .odf for formulae, mathematical equations

LibreOffice

See Also:


LibreOffice is an office suite that is compatible with other major office suites, and available on a variety of platforms. Its goal is to produce a vendor-independent office suite with ODF support and without any copyright assignment requirements.

The name is a hybrid word with the first part Libre, which means free (as in freedom) in both Spanish and French and the English word Office.
As free software, users are free to download, modify, use and distribute LibreOffice.

**Odfpy**  See Also:

http://odfpy.forge.osor.eu/

Odfpy is a library to read and write OpenDocument v. 1.1 files. The main focus has been to prevent the programmer from creating invalid documents. It has checks that raise an exception if the programmer adds an invalid element, adds an attribute unknown to the grammar, forgets to add a required attribute or adds text to an element that doesn’t allow it.

These checks and the API itself were generated from the RelaxNG schema, and then hand-edited. Therefore the API is complete and can handle all ODF constructions, but could be improved in its understanding of data types.

**WebODF**

See Also:

- http://webodf.org/
- http://gitorious.org/odfkit/webodf
- https://demo.webodf.org/demo/

WebODF is a JavaScript library that makes it easy to add Open Document Format (ODF) support to your website and to your mobile or desktop application.

It uses HTML and CSS to display ODF documents.

**Nos données (fr)**

See Also:

- http://www.nosdonnees.fr/
NosDonnees.fr vise à apporter plus de visibilité aux données publiques librement accessibles à chacun.

Le site est constitué de deux outils :

- un registre libre de jeux de données : reposant sur le logiciel CKAN, développé par l’Open Knowledge Foundation, cet outil simplifie l’accès, le partage et la réutilisation de données et contenus, notamment sous des formes lisibles par les machines ;
- une plateforme collaborative de chasse aux trésors : prenant la forme d’un wiki, cet outil permet de coordonner le double effort de référencement sur CKAN des données publiques librement accessibles sur Internet, et de libération et publication des données pas encore accessibles.

Convaincus des fantastiques apports à l’innovation sociale et économique permis par l’ouverture des données publiques, par exemple pour valoriser auprès des citoyens le travail des administrations publiques, nous espérons participer avec NosDonnees.fr à l’émergence d’une communauté Open Data en France.

David Larlet  See Also:

- http://packages.python.org/SuRF/

Open knowledge

See Also:

- http://okfn.org/

We’re a not-for-profit organization promoting open knowledge: that’s any kind of information – sonnets to statistics, genes to geodata – that can be freely used, reused, and redistributed. We promote open knowledge because of its potential to deliver far-reaching societal benefits.

We organize events like OKCon, run projects like Open Shakespeare, and develop tools like CKAN and Knowledge-Forge to help people create, find and share open material. See our full list of events » and projects ».

ORDF  ORDF is the Open Knowledge Foundation’s library of support infrastructure for RDF. It is available from

- PyPi: http://pypi.python.org/pypi/ordf/
- Mercurial: http://ordf.org/src/

It is based on RDFLib and contains an object-description mapper, support for multiple back-end indices, message passing, revision history and provenance, a namespace library and a variety of helper functions and modules to ease integration with the Pylons framework.
Free pdf readers

See Also:

- http://www.pdfreaders.org/

Le Format de Document Portable (Portable Document Format - PDF) est un format populaire pour publier textes et documents formatés.

Il en existe de nombreuses versions, certaines peuvent être qualifiées de Standards Ouverts, d’autres sont normalisées par l’ISO et d’autres enfin sont grevées par des brevets logiciels.

Vous devriez privilégier et promouvoir les versions basées sur des Standards Ouverts parce que seuls les Standards Ouverts garantissent l’interopérabilité, la concurrence et la liberté de choix. Pour en savoir plus...

Il existe différents programmes pour lire et écrire des documents PDF.
La liste suivante de lecteurs de PDF est indépendante de tous revendeurs.
Tous sont des Logiciels Libres qui respectent vos quatre libertés fondamentales

- d’utiliser
- d’étudier
- de partager
- et d’améliorer ces logiciels.

Ceux-ci vous assurent un parfait contrôle de votre ordinateur ainsi que la protection de votre vie privée. Pour en savoir plus...

Evince (on windows) See Also:
http://live.gnome.org/Evince/Downloads

Okular (on GNU/Linux) See Also:
http://windows.kde.org/download.php

Google public data

See Also:

- http://www.google.com/publicdata/directory

DSPL: Dataset Publishing Language  DSPL is the Dataset Publishing Language, a representation language for the data and metadata of datasets. Datasets described in this format can be processed by Google and visualized in the Google Public Data Explorer.

Tutorial


2.21. Open data
gapminder

See Also:

- http://www.gapminder.org/data/

Videos


Building a fact-based world view  Gapminder is a non-profit foundation based in Stockholm. Our goal is to replace devastating myths with a fact-based world view.

Our method is to make data easy to understand. We are dedicated to innovate and spread new methods to make global development understandable, free of charge, without advertising.

We want to let teachers, journalists and everyone else continue to freely use our tools, videos and presentations.

2.22 Operating systems

2.22.1 Operating systems

Contents

- Operating systems
  - Android/replicant operating systems
  - Boot to gecko
  - FreeBSD
  - iOS operating systems
  - Linux operating systems
  - Windows operating systems
  - MacOSX operating system
  - bootloaders
  - Virtualization

Android/replicant operating systems

Android Operating system

See Also:

- http://blog.xamarin.com/2012/05/01/android-in-c-sharp/
- http://xamarin.com/monoforandroid

- Replicant Operating system
- Dalvik Java Virtual Machine

D’autres types d’appareils possédant ce système d’exploitation existent, par exemple des téléviseurs et des tablettes.

Android installation  See Also:

- http://www.buildroid.org/blog/

Android on virtualbox  See Also:

- http://www.buildroid.org/blog/?page_id=121

Android versions

Android 4.0 Ice Cream Sandwich  See Also:


News  See Also:

https://groups.google.com/forum/#!msg/android-building/T4XZJCZnqF8/WkJWGYb4MAJ

Hi! We just released a bit of code we thought this group might be interested in. Over at our Android Open-Source Project git servers, the source code for Android version 4.0 (Ice Cream Sandwich) is now available.

Here’s how to get it: Follow the instructions at http://source.android.com/source/downloading.html Check out the ‘ics-release’ branch: repo init -u https://android.googlesource.com/platform/manifest -b android-4.0.1_r1 That’s it!

However since this is a large push, please be aware that it will take some time to complete. If you sync before it's done, you’ll get an incomplete copy that you won’t be able to use, so please wait for us to give the all-clear before you sync.

This is actually the source code for version 4.0.1 of Android, which is the specific version that will ship on the Galaxy Nexus, the first Android 4.0 device. In the source tree, you will find a device build target named “full_maguro” that you can use to build a system image for Galaxy Nexus. Build configurations for other devices will come later.

Unfortunately we still don’t have our Gerrit code review servers back online. That remains our top priority though, and we hope to have them back soon.

This release includes the full history of the Android source code tree, which naturally includes all the source code for the Honeycomb releases. However, since Honeycomb was a little incomplete, we want everyone to focus on Ice Cream Sandwich. So, we haven’t created any tags that correspond to the Honeycomb releases (even though the changes are present in the history.)

JBQ, on behalf of the AOSP team. – Jean-Baptiste M. “JBQ” Queru Software Engineer, Android Open-Source Project, Google.
Oracle and Google are currently in a $1 billion wrestling match over Google’s use of Java in Android.

But Java is not the only way to build native apps on Android.

In fact, it’s not even the best way: we have been offering C# to Android developers as a high-performance, low-battery consuming alternative to Java.

Our platform, Mono, is an open source implementation of the .NET framework that allows developers to write their code using C# while running on top of the Java-powered operating system, and then share that same code with iOS and Windows Phone.

Android python

Android pyside See Also:


Replicant Operating system

See Also:

- Android Operating system

« Android représente une étape majeure vers un téléphone portable libre qui soit contrôlé par l’utilisateur, mais il y a encore beaucoup de chemin à parcourir. Les hackers travaillent sur Replicant, mais c’est un gros travail de rendre un nouveau modèle compatible, et il reste encore le problème du micrologiciel. Même si les téléphones Android d’aujourd’hui sont considérablement moins mauvais que les smartphones d’Apple ou de Windows, on ne peut pas dire qu’ils respectent vos libertés. »
Ainsi se termine l’article de Richard Stallman consacré à Android et la liberté des utilisateurs. On y évoque donc le projet Replicant qui vise à produire un dérivé entièrement libre du système d’exploitation de Google.

Or il se trouve que nous avons la chance d’avoir l’un des développeurs du projet parmi les membres traducteurs de Framalang. Il eut été alors dommage de ne pas en profiter pour en savoir plus et comprendre la difficulté qu’il y a à tenter de libérer totalement un téléphone portable.

On notera que ce développeur est encore lycéen, ce qui fournit au passage un excellent témoignage d’une jeunesse qui, ayant découvert le logiciel libre, n’a plus forcément envie de passer tout son temps sur Facebook :-) 

Replicant is a distribution of Android that is 100% Free Software.

Most of Android is licensed freely under the Apache License 2.0. The Linux core is mostly Free Software under the GPLv2. However, there are numerous components of the default software stack on the devices that are proprietary software. Most notably, nearly any component that touches the hardware directly is proprietary software.

We are not experts in embedded devices; we are just enthusiastic hackers that are giving a try.

Currently we have this blog and a Trac instance, both graciously donated by OSU-OSL.

Replicant was founded by four people: Bradley M. Kuhn, Aaron Williamson, Graziano Sorbaioli, Denis ‘GNUtoo’ Carikli.

**Boot to gecko**

**See Also:**
- http://www.mozilla.org/en-US/b2g/

**Architecture**

**See Also:**
- https://groups.google.com/forum/?fromgroups#!forum/mozilla.dev.b2g

**FreeBSD**

**FreeBSD Operating system**

**See Also:**
**iOS operating systems**

**iOS**

See Also:


**iOS versions**

**iOS 5.0**

**Linux operating systems**

**Linux operating system**

![The linux logo](image)

See Also:


**Mailing list** [http://vger.kernel.org/vger-lists.html](http://vger.kernel.org/vger-lists.html)

**Comment about GNU/Linux** From Roland Mas.

La partie strictement GNU a tendance à devenir un poil minoritaire, même si elle reste relativement centrale (entre le compilateur, la libc et les coreutils), d’autres éléments « non GNU » sont tout aussi centraux (je pense à udev, dbus, et tous les policykit, consolekit, et autres kit de chez [http://freedesktop.org](http://freedesktop.org), par exemple).

Si on veut être correct, l’OS complet, c’est GNU/Mozilla/Xorg/Openoffice/Gnome/Freedesktop/Apache/Perl/Python/OpenBSD (pour ssh/etc./Linux. Pour faire court, on dit juste Debian (et en prime comme ça on inclut même un noyau FreeBSD).

(Ceci n’est qu’à moitié un troll. Je suis conscient de la différence entre le noyau et l’OS, mais faudrait voir à pas oublier les autres gens que ceux du projet GNU qui font aussi un boulot formidable.)

**The “kernel”**

**linux kernel** See Also:


**GPL licenses** The following license idents are currently accepted as indicating free software modules.

<table>
<thead>
<tr>
<th>License ident</th>
<th>License</th>
</tr>
</thead>
<tbody>
<tr>
<td>“GPL”</td>
<td>[GNU Public License v2 or later]</td>
</tr>
<tr>
<td>“GPL v2”</td>
<td>[GNU Public License v2]</td>
</tr>
<tr>
<td>“GPL and additional rights”</td>
<td>[GNU Public License v2 rights and more]</td>
</tr>
<tr>
<td>“Dual BSD/GPL”</td>
<td>[GNU Public License v2 or BSD license choice]</td>
</tr>
<tr>
<td>“Dual MIT/GPL”</td>
<td>[GNU Public License v2 or MIT license choice]</td>
</tr>
<tr>
<td>“Dual MPL/GPL”</td>
<td>[GNU Public License v2 or Mozilla license choice]</td>
</tr>
</tbody>
</table>
The different GPL licenses for USB drivers  Since the linux 2.6.25 (February 2008), if we build the certisV2.ko kernel module with this line:

```c
MODULE_LICENSE("Proprietary");
```

we obtain the following errors:

```
Building modules, stage 2.
MODPOST 1 modules
FATAL: modpost: GPL-incompatible module certisV2.ko uses GPL-only symbol 'usb_buffer_alloc'
```

In order to link with the USB kernel functions we have to put this line at the end of the certis_driver.c file:

```c
MODULE_LICENSE("GPL");
```

**Warning:** This means that we must deliver the source code which is not the case.

See Also:

- http://linuxgazette.net/issue32/rubini.html
- http://www.gnu.org/licenses/gpl-faq.html
USB driver API moves to EXPORT_SYMBOL_GPL (February 2008)  See Also:

http://archive.netbsd.se/?ml=&a=2009-01&t=6220842

Subject: [PATCH] USB: mark USB drivers as being GPL only
From: Greg KH
Date: 2008-01-25 19:02:32

FYI, this is a patch that will be sent out in the next round to Linus for inclusion in 2.6.25.
If anyone has any objections about it, please let me know.
thanks,
greg k-h
--------
From: Greg Kroah-Hartman
Subject: USB: mark USB drivers as being GPL only

No way to create a USB kernel driver that was not under the GPL

Warning: Over two years ago, the Linux USB developers stated that they believed there was no way to create a USB kernel driver that was not under the GPL. This patch moves the USB apis to enforce that decision.

There are no known closed source USB drivers in the wild, so this patch should cause no problems.

Signed-off-by: Greg Kroah-Hartman

-What: USB driver API moves to EXPORT_SYMBOL_GPL
-When: February 2008
-Files: include/linux/usb.h, drivers/usb/core/driver.c

Warning: The USB subsystem has changed a lot over time, and it has been possible to create userspace USB drivers using usbfs/libusb/gadgetfs that operate as fast as the USB bus allows. Because of this, the USB subsystem will not be allowing closed source kernel drivers to register with it, after this grace period is over. If anyone needs any help in converting their closed source drivers over to use the userspace filesystems, please contact the linux-usb-<email removed> mailing list, and the developers there will be glad to help you out. Greg Kroah-Hartman

See Also:

- libusb 1.0
- http://git.libusb.org/?p=libusb.git;a=tree
- libfprint linux library

From: Greg KH
Date: 2008-02-02 20:19:30

On Sat, Feb 02, 2008 at 12:37:10PM +0100, Christer Weinigel wrote:
> On Fri, 25 Jan 2008 10:02:32 -0800
> Greg KH wrote:
> >
> > FYI, this is a patch that will be sent out in the next round to Linus
> > for inclusion in 2.6.25.
> >
> > If anyone has any objections about it, please let me know.
Yes, I have objections and I’ve told you before.

You sent one message on this topic to me, back in Feb of 2007, disagreeing that you could write a userspace USB driver running at full speed in a non-racey manner.

Unfortunately, many other userspace USB drivers seem to disprove your statement, including a number of vision systems running in military applications (tanks running Linux!). Perhaps this is just a matter of using the api properly ;)

I do know that the current usbfs interface is a major pain, hence the work to create usbfs2. I know those developers could use the help in getting that cleaned up and into the kernel tree.

Also see the rapid development these days in wrappers around usbfs. There is competing projects right now with OpenUSB and the revitalization of the old libusb project. I know those developers are looking for examples where their new frameworks do not meet the needs of developers for stuff exactly like you describe (lots of threads, async callbacks, high throughput, cross-platform portability, etc.)

> For some of these drivers, being in kernel space is very important > since they transfer large amounts of data with very tight latency > requirements. It may, in theory, be possible to do the same thing in > userspace with multiple cooperative threads and libusb, but it would be > much more complex and much more error prone (it’s hard to do control > loops where you need about 40 us turnaround time).

See statement above about vision systems in tanks, it can, and is done all the time...

If a company wants to keep a driver closed, then use another operating system, it’s not like there isn’t other options out there. I hear the BSDs and Microsoft are quite comfortable with things like that. :)

From: Christer Weinigel
Date: 2008-02-03 12:48:49

On Sat, 2 Feb 2008 11:19:30 -0800
Greg KH wrote:

> If a company wants to keep a driver closed, then use another operating > system, it’s not like there isn’t other options out there. I hear the > BSDs and Microsoft are quite comfortable with things like that. :)

So in other words you want to crack down on GPL violations, and you’re going to ignore anyone who does have a proprietary driver as "not relevant" or "it can be done with usbfs" (maybe). So why even ask on the mailing list? Just do it.

Saying "use BSD" instead isn’t a good answer for me since I don’t know BSD well enough. And personally, I want to see Linux everywhere; I think it’s a lot better to have Linux + a proprietary driver in an embedded system than BSD or Windows CE. It means that the customers get used to Linux, and if I can get them to at least contribute back a
bit (any improvements to the core kernel for example), to me that is a lot better than giving a lot more money to BillG.

Later, when I can show them how much easier everything gets if they use open drivers (I’d never have managed to get my latest Samsung platform up and running as quickly as I did without the patches I got from Sandeep Patil, and by posting my patches to his patches I got some feedback that helped me fix a bunch of bugs). But it usually takes some time to convince a company that the things they get back is more valuable than keeping things proprietary. So I think Linux as a whole gains a lot more by being a bit lenient about proprietary drivers. That is why I’m opposed this change of yours.

See Also:

http://kerneltrap.org/node/2991

In 2001 during the 2.4 kernel development cycle, a MODULE_LICENSE macro was introduced which allows a module to explicitly declare how it is licensed.

Currently (2004) there are five supported types of free software modules:

- "GPL",
- "GPL v2"
- "GPL and additional rights"
- "Dual BSD/GPL",
- "Dual MPL/GPL"

otherwise the kernel is considered "tainted". The include/linux/module.h header file lists three reasons for this macro: to allow users to review their license info to verify that they have a free setup, so the development community can ignore bug reports that include proprietary modules which don’t release their source code, and so that vendors can do as is defined by their own policies. Further information is available in the lkml FAQ, as well as this earlier thread.

From Linus Torvalds: RE: Linux GPL and binary module exception clause?  
See Also:

http://lkml.org/lkml/2003/12/5/13

Date Thu, 4 Dec 2003 22:58:09 -0800 (PST)
From Linus Torvalds <>
Subject RE: Linux GPL and binary module exception clause?

On Thu, 4 Dec 2003, David Schwartz wrote:
>
> The GPL gives you the unrestricted right to *use* the original work.
> This implicitly includes the right to perform any step necessary to use
> the work.

No it doesn’t.

Your logic is fundamentally flawed, and/or your reading skills are deficient.

The GPL expressly states that the license does not restrict the act of "running the Program" in any way, and yes, in that sense you may "use" the
program in whatever way you want.

But that "use" is clearly limited to running the resultant program. It very much does NOT say that you can "use the header files in any way you want, including building non-GPL’d programs with them".

In fact, it very much says the reverse. If you use the source code to build a new program, the GPL _explicitly_ says that that new program has to be GPL’d too.

> Please tell me how you use a kernel header file, other than by including it in a code file, compiling that code file, and executing the result.

You are a weasel, and you are trying to make the world look the way you want it to, rather than the way it _is_.

You use the word "use" in a sense that is not compatible with the GPL. You claim that the GPL says that you can "use the program any way you want", but that is simply not accurate or even _close_ to accurate. Go back and read the GPL again. It says:

"The act of running the Program is not restricted"

and it very much does NOT say

"The act of using parts of the source code of the Program is not restricted"

In short: you do _NOT_ have the right to use a kernel header file (or any other part of the kernel sources), unless that use results in a GPL’d program.

What you _do_ have the right is to _run_ the kernel any way you please (this is the part you would like to redefine as "use the source code", but that definition simply isn’t allowed by the license, however much you protest to the contrary).

So you can run the kernel and create non-GPL’d programs while running it to your hearts content. You can use it to control a nuclear submarine, and that’s totally outside the scope of the license (but if you do, please note that the license does not imply any kind of warranty or similar).

BUT YOU CAN NOT USE THE KERNEL HEADER FILES TO CREATE NON-GPL’D BINARIES.

Comprende?

Linus

**module.h**  This is the linux module.h header file.

```c
/*
 * The following license idents are currently accepted as indicating free
 * software modules
 *
 * "GPL" [GNU Public License v2 or later]
 * "GPL v2" [GNU Public License v2]
 */
```
Other articles about the GPL license

Being honest with MODULE_LICENSE: http://lwn.net/Articles/82305/  MODULE_LICENSE() is a macro which allows loadable kernel modules to declare their license to the world. Its purpose is to let the kernel developers know when a non-free module has been inserted into a given kernel.

If you submit an oops report showing a “tainted” kernel, chances you will be asked to reproduce the problem without the proprietary module loaded, or to talk to that module’s vendor about the problem. In general, the kernel hackers want to hear about problems, but their interest drops remarkably when they cannot get at the source to diagnose or fix the problem.

The declared module license is also used to decide whether a given module can have access to the small number of “GPL-only” symbols in the kernel.

**Warning:** There is no central authority which checks license declarations; it is assumed that module authors will not want to lie about the license they are using.

**Certis_bio taint the kernel**  When a certis device is inserted we have the following messages.

dmesg

Certis Bio code detected
certis_bio: module license ‘Proprietary’ taints kernel.
Disabling lock debugging due to kernel taint
Begin init_module(void)
Begin certis_GetExchangeObject()
End certis_GetExchangeObject().
Begin certis_bio_init(STCOM * pstCOM) with the shared memory.
End certis_bio_init()
End init_module(void)
The certis_bio.ko module is loaded: 0
The plugged certis device is now attached to the USB minor:<181>
End certis_probe().

root@portuxg20:/mnt/portuxg20# lsmod
Module Size Used by Tainted: P
certis_bio 54880 0
CERTISV2 35968 1 certis_bio

root@portuxg20:/mnt/portuxg20# lsmod

linux kernel versions

linux kernel 3.2.0 (5 janvier 2012) See Also:

• http://linuxfr.org/users/barmic/journaux/linux-32-en-route
• http://www.h-online.com/open/features/Kernel-Log-Linux-3-2-main-development-phase-complete-1375016.html

La sortie de la version stable 3.2 du noyau Linux vient d’être annoncée par Linus Torvalds sur la liste de diffusion et sur Google+.

Le nouveau noyau est, comme d’habitude, téléchargeable sur les serveurs du site kernel.org.

Le détail des évolutions, nouveautés et prévisions est dans la seconde partie de la dépêche.

P.S. : Merci à toutes les personnes qui ont aidé à traduire les courriels de RC quand cette dépêche était dans l’espace de rédaction. Merci également à Laurent Wandrebeck (low) pour sa contribution sur la brève concernant DVFS.

Linux 3.2 includes support for Hexagon CPUs, enhancements to the TCP stack, an upbeat SHA1 implementation and dozens of new and revised drivers. And Linux now has a Google+ page.

linux kernel 3.1.0 See Also:

• http://linuxfr.org/news/sortie-du-noyau-linux%C2%A031
• http://article.gmane.org/gmane.linux.kernel/1206934

Le commit marquant la sortie de la version stable 3.1 du noyau Linux vient d’être effectué par Linus Torvalds lors du sommet de Prague.

Les sources de ce nouveau noyau sont téléchargeables sur les serveurs du site kernel.org et le message d’annonce de Linus est lisible ici.

linux kernel 3.0.0 See Also:


La sortie de la version stable 3.0 du noyau Linux vient d’être annoncée par Linus Torvalds.

Le nouveau noyau est, comme d’habitude, téléchargeable sur les serveurs du site kernel.org.

Ce changement de numérotation du noyau est l’occasion de tirer un coup de chapeau aux 176 extralucides du sondage LinuxFr de janvier 2010 qui avaient deviné que ce noyau 3.0 sortirait cette année. Bravo à eux!

Le détail des évolutions et des nouveautés se trouve dans la seconde partie de la dépêche.
An anonymous reader writes “Linus just released the first -rc of the next kernel series, but rather than continuing development as the Linux 2.6.40 kernel, he has renamed it to be the Linux 3.0 kernel.”

And he’s tacked on a second dot and another zero (3.0.0), at least for now, because many scripts expect and rely on a three-part kernel version.

Many programming books begin with a “hello world” example as a way of showing the simplest possible program. This book deals in kernel modules rather than programs; so, for the impatient reader, the following code is a complete “hello world” module:
This module defines two functions, one to be invoked when the module is loaded into the kernel (hello_init) and one for when the module is removed (hello_exit). The module_init and module_exit lines use special kernel macros to indicate the role of these two functions. Another special macro (MODULE_LICENSE) is used to tell the kernel that this module bears a free license; without such a declaration, the kernel complains when the module is loaded.

### Loading and Unloading Modules

After the module is built, the next step is loading it into the kernel. As we’ve already pointed out, insmod does the job for you. The program loads the module code and data into the kernel, which, in turn, performs a function similar to that of ld, in that it links any unresolved symbol in the module to the symbol table of the kernel. Unlike the linker, however, the kernel doesn’t modify the module’s disk file, but rather an in-memory copy. insmod accepts a number of command-line options (for details, see the manpage), and it can assign values to parameters in your module before linking to the current kernel. Thus, if a module is correctly designed, it can be configured at load time; load-time configuration gives the user more flexibility than compile-time configuration, which is still used sometimes. Load-time configuration is explained in the section “Module Parameters,” later in this chapter.

Interested readers may want to look at how the kernel supports insmod it relies on a system call defined in kernel/module.c. The function sys_init_module allocates kernel memory to hold a module (this memory is allocated with vmalloc; see the section “vmalloc and Friends” in Chapter 8); it then copies the module text into that memory region, resolves kernel references in the module via the kernel symbol table, and calls the module’s initialization function to get everything going.

If you actually look in the kernel source, you’ll find that the names of the system calls are prefixed with sys. This is true for all system calls and no other functions; it’s useful to keep this in mind when grepping for the system calls in the sources.

The modprobe utility is worth a quick mention. modprobe, like insmod, loads a module into the kernel. It differs in that it will look at the module to be loaded to see whether it references any symbols that are not currently defined in the kernel. If any such references are found, modprobe looks for other modules in the current module search path that define the relevant symbols. When modprobe finds those modules (which are needed by the module being loaded), it loads them into the kernel as well. If you use insmod in this situation instead, the command fails with an “unresolved symbols” message left in the system logfile.

As mentioned before, modules may be removed from the kernel with the rmmod utility. Note that module removal fails if the kernel believes that the module is still in use (e.g., a program still has an open file for a device exported by the modules), or if the kernel has been configured to disallow module removal. It is possible to configure the kernel to allow “forced” removal of modules, even when they appear to be busy. If you reach a point where you are considering using this option, however, things are likely to have gone wrong badly enough that a reboot may well be the better course of action.
lsmod

The lsmod program produces a list of the modules currently loaded in the kernel. Some other information, such as any other modules making use of a specific module, is also provided. lsmod works by reading the /proc/modules virtual file. Information on currently loaded modules can also be found in the sysfs virtual filesystem under /sys/module.

depmod

Linux kernel modules can provide services (called “symbols”) for other modules to use (using EXPORT_SYMBOL in the code). If a second module uses this symbol, that second module clearly depends on the first module. These dependencies can get quite complex.

depmod creates a list of module dependencies, by reading each module under /lib/modules/version and determining what symbols it exports, and what symbols it needs. By default this list is written to modules.dep in the same directory. If filenames are given on the command line, only those modules are examined (which is rarely useful, unless all modules are listed).

root@portuxg20:/mnt/portuxg20# cat /lib/modules/2.6.31/modules.dep

/lib/modules/2.6.31/certisV2.ko:
/lib/modules/2.6.31/kernel/drivers/usb/gadget/gadgetfs.ko: /lib/modules/2.6.31/kernel/drivers/usb/gadget/g_file_storage.ko: /lib/modules/2.6.31/kernel/drivers/usb/gadget/g_serial.ko: /lib/modules/2.6.31/kernel/drivers/usb/gadget/at91_udc.ko
/lib/modules/2.6.31/kernel/drivers/usb/gadget/g_file_storage.ko: /lib/modules/2.6.31/kernel/drivers/usb/gadget/g_serial.ko: /lib/modules/2.6.31/kernel/drivers/usb/gadget/at91_udc.ko
/lib/modules/2.6.31/kernel/drivers/usb/gadget/g_serial.ko: /lib/modules/2.6.31/kernel/drivers/usb/gadget/at91_udc.ko
/lib/modules/2.6.31/certis_bio.ko: /lib/modules/2.6.31/certisV2.ko

modutils

See Also:
http://www.kernel.org/pub/linux/utils/kernel/modutils/

modprobe

udev

See Also:
- http://git.kernel.org/?p=linux/hotplug/udev.git;a=summary

udev is the device manager for the Linux kernel. Primarily, it manages device nodes in /dev. It is the successor of devfs and hotplug, which means that it handles the /dev directory and all user space actions when adding/removing devices, including firmware load.
Manual installation

- download the archive under /tmp
- cd /tmp; unzip the archive
- ./autogen.sh

Architecture  The system is divided into three parts:

- The library libudev, that allows access to device information.
- The daemon udevd, in user space, that manages the virtual /dev.
- The administrative command udevadm for diagnostics.

The system gets calls from the kernel via netlink socket.

Earlier versions used hotplug, adding a link to themselves in /etc/hotplug.d/default with this purpose.

See Also:

http://git.kernel.org/?p=linux/hotplug/udev.git;a=tree

Its goals are the following:

- Run in userspace
- Create a dynamic /dev.
- Provide consistent device naming, if wanted.
- Provide a userspace API to access info about current system devices.

The first item, “run in userspace,” is easily done by harnessing the fact that /sbin/hotplug generates an event for every device that is added or removed from the system, combined with the ability of sysfs to show all needed information about all devices.

The rest of the goals enable the udev project to be split into three separate subsystems:

1. namedev – handles all device naming
2. libsysfs – a standard library for accessing device information on the system.
3. udev – dynamic replacement for /dev

The udev program will be responsible for talking to both the namedev and libsysfs libraries to accomplish the device naming policy that has been specified. The udev program is run whenever /sbin/hotplug is called by the kernel. It does this by adding a symlink to itself in the /etc/hotplug.d/default directory, which is searched by the /sbin/hotplug multiplexer script

udev links

udev README  Source: http://git.kernel.org/?p=linux/hotplug/udev.git;a=blob;f=README;h=6e09b3a52da14540adf82a03a650a3df

udev - Linux userspace device management  Integrating udev in the system has complex dependencies and may differ from distribution to distribution. A system may not be able to boot up or work reliably without a properly installed udev version. The upstream udev project does not recommend to replace a distro’s udev installation with the upstream version.

The upstream udev project’s set of default rules may require a most recent kernel release to work properly.
Tools and rules shipped by udev are not public API and may change at any time. Never call any private tool in /lib/udev from any external application, it might just go away in the next release. Access to udev information is only offered by udevadm and libudev. Tools and rules in /lib/udev, and the entire content of the /dev/.udev directory is private to udev and does change whenever needed.

Requirements

- Version 2.6.27 of the Linux kernel with sysfs, procfs, signalfd, inotify, unix domain sockets, networking and hotplug enabled:
  
  CONFIG_HOTPLUG=y
  CONFIG_UEVENT_HELPER_PATH=""
  CONFIG_NET=y
  CONFIG_UNIX=y
  CONFIG_SYSFS=y
  CONFIG_SYSFS_DEPRECATED*=n
  CONFIG_PROC_FS=y
  CONFIG_TMPFS=y
  CONFIG_INOTIFY_USER=y
  CONFIG_SIGNALFD=y
  CONFIG_TMPFS_POSIX_ACL=y (user ACLs for device nodes)
  CONFIG_BLK_DEV_BSG=y (SCSI devices)

- Udev will not work with the CONFIG_SYSFS_DEPRECATED* option.

- Unix domain sockets (CONFIG_UNIX) as a loadable kernel module may work, but it is not supported.

- The deprecated hotplug helper /sbin/hotplug should be disabled in the kernel configuration, it is not needed today, and may render the system unusable because the kernel may create too many processes in parallel so that the system runs out-of-memory.

- The proc filesystem must be mounted on /proc, the sysfs filesystem must be mounted at /sys. No other locations are supported by a standard udev installation.

- The system must have the following group names resolvable at udev startup: disk, cdrom, floppy, tape, audio, video, lp, tty, dialout, kmem. Especially in LDAP setups, it is required, that getgrnam() is able to resolve these group names with only the rootfs mounted, and while no network is available.

- To build all ‘udev extras’, libacl, libglib2, libusb, usbutils, pciutils, gperf are needed. These dependencies can be disabled with the disable-extras configure option.

Setup

- At bootup, the /dev directory should get the ‘devtmpfs’ filesystem mounted. Udev will manage permissions and ownership of the kernel-created device nodes, and possibly create additional symlinks. If needed, udev also works on an empty ‘tmpfs’ filesystem, but some static device nodes like /dev/null, /dev/console, /dev/kmsg are needed to be able to start udev itself.

- The content of /lib/udev/devices directory which contains static content like symlinks and directories, which are always expected to be in /dev, should be copied over to the mounted /dev directory:

  cp -axT --remove-destination /lib/udev/devices /dev

- The udev daemon should be started to handle device events sent by the kernel. During bootup, the kernel can be asked to send events for all already existing devices, to apply the configuration to these devices. This is usually done by:

  /sbin/udevadm trigger --type=subsystems
  /sbin/udevadm trigger --type=devices
- Restarting the daemon does never apply any rules to existing devices.
- New/changed rule files are picked up automatically, there is no daemon restart or signal needed.

Operation

- Udev creates/removes device nodes in /dev, based on events the kernel sends out on device creation/removal.
- All kernel events are matched against a set of specified rules, which possibly hook into the event processing and load required kernel modules to setup devices. For all devices the kernel exports a major/minor number, if needed, udev will create a device node with the default kernel name. If specified, udev applies permissions/ownership to the device node, creates additional symlinks pointing to the node, and executes programs to handle the device.
- The events udev handles, and the information udev merges into its device database, can be accessed with libudev:

For more details about udev and udev rules see the udev(7) man page.

Please direct any comment/question to the linux-hotplug mailing list at: linux-hotplug@vger.kernel.org

udev and devfs - The final word  See Also:

- http://www.kroah.com/linux/
- http://www.kroah.com/log/

December 30, 2003
Greg Kroah-Hartman
<greg@kroah.com>

Executive summary for those too lazy to read this whole thing: I don’t care about devfs, and I don’t want to talk about it at all anymore. If you love devfs, fine, I’m not trying to tell anyone what to do. But you really should be looking into using udev instead. All further email messages sent to me about devfs will be gladly ignored.

First off, some background. For a description of udev, and what its original design goals were, please see the OLS 2003 paper on udev, available at:


and the slides for the talk, available at:


The OLS paper can also be found in the docs/ directory of the udev tarball, available on kernel.org in the /pub/linux/utils/kernel/hotplug/ directory.

In that OLS paper, I described the current situation of a static /dev and the current problems that a number of people have with it. I also detailed how devfs tries to solve a number of these problems. In hindsight, I should have never mentioned the word “devfs” when talking about udev. I did so only because it seemed like a good place to start with. Most people understood what devfs is, and what it does. To compare udev against it, showing how udev was more powerful, and a more complete solution to the problems people were having, seemed like a natural comparison to me.

But no more. I hereby never want to compare devfs and udev again. With the exception of this message...
The Problems

1. A static /dev is unwieldy and big. It would be nice to only show the /dev entries for the devices we actually have running in the system.

2. We are (well, were) running out of major and minor numbers for devices.

3. Users want a way to name devices in a persistent fashion (i.e. “This disk here, must _always_ be called “boot_disk” no matter where in the scsi tree I put it”, or “This USB camera must always be called “camera” no matter if I have other USB scsi devices plugged in or not.”)

4. Userspace programs want to know when devices are created or removed, and what /dev entry is associated with them.

The constraints

1. No policy in the kernel!

2. Follow standards (like the LSB)

3. must be small so embedded devices will use it.

So, how does devfs stack up to the above problems and constraints

Problems

1. devfs only shows the dev entries for the devices in the system.

2. devfs does not handle the need for dynamic major/minor numbers

3. devfs does not provide a way to name devices in a persistent fashion.

4. devfs does provide a daemon that userspace programs can hook into to listen to see what devices are being created or removed.

Constraints

1. devfs forces the devfs naming policy into the kernel. If you don’t like this naming scheme, tough.

2. devfs does not follow the LSB device naming standard.

3. devfs is small, and embedded devices use it. However it is implemented in non-pagable memory.

Oh yeah, and there are the insolvable race conditions with the devfs implementation in the kernel, but I’m not going to talk about them right now, sorry. See the linux-kernel archives if you care about them (and if you use devfs, you should care...)

So devfs is 2 for 7, ignoring the kernel races.

And now for udev

Problems

1. using udev, the /dev tree only is populated for the devices that are currently present in the system.

2. udev does not care about the major/minor number schemes. If the kernel tomorrow switches to randomly assign major and minor numbers to different devices, it would work just fine (this is exactly what I am proposing to do in 2.7...)

3. This is the main reason udev is around. It provides the ability to name devices in a persistent manner. More on that below.
4. udev emits D-BUS messages so that any other userspace program (like HAL) can listen to see what devices are created or removed. It also allows userspace programs to query its database to see what devices are present and what they are currently named as (providing a pointer into the sysfs tree for that specific device node.)

Constraints

1. udev moves _all_ naming policies out of the kernel and into userspace.
2. udev defaults to using the LSB device naming standard. If users want to deviate away from this standard (for example when naming some devices in a persistent manner), it is easily possible to do so.
3. udev is small and is entirely in userspace, which is swapable, and doesn’t have to be running at all times.

Nice, 7 out of 7 for udev. Makes you think the problems and constraints were picked by a udev developer, right? No, the problems and constraints are ones I’ve seen over the years and so udev, along with the kernel driver model and sysfs, were created to solve these real problems. I also have had the luxury to see the problems that the current devfs implementation has, and have taken the time to work out something that does not have those same problems.

So by just looking at the above descriptions, everyone should instantly realize that udev is far better than devfs and start helping out udev development, right? Oh, you want more info, ok...

Back in May 2003 I released a very tiny version of udev that implemented everything that devfs currently does, in about 6Kb of userspace code:

- [http://marc.theaimsgroup.com/?l=linux-kernel&m=105003185331553](http://marc.theaimsgroup.com/?l=linux-kernel&m=105003185331553)

Yes, that’s right, 6Kb. So, you are asking, why are you still working on udev if it did everything devfs did back in May 2003? That’s because just managing static device nodes based on what the kernel calls the devices is _not_ the primary goal of udev. It’s just a tiny side effect of its primary goal, the ability to never worry about major/minor number assignments and provide the ability to achieve persistent device names if wanted.

All the people wanting to bring up the udev vs. devfs argument go back and read the previous paragraph. Yes, all Gentoo users who keep filling up my inbox with smoking emails, I mean you.

So, how well does udev solve its goals:

- Prevent users from ever worrying about major/minor numbers And here you were, not knowing you ever needed to worry about major/minor numbers in the first place, right? Ah, I see you haven’t plugged in 2 USB printers and tried to figure out which printer was which /dev entry? Or plugged in 4000 SCSI disks and tried to figure out how to access that 3642nd disk and what it was called in /dev. Or plugged in a USB camera and a USB flash drive and then tried to download the pictures off of the flash drive by accident?

As the above scenarios show, both desktop users and big iron users both need to not worry about which device is assigned to what major/minor device.

udev doesn’t care what major/minor number is assigned to a device. It merely takes the numbers that the kernel says it assigned to the device and creates a device node based on it, which the user can then use (if you don’t understand the whole major/minor to device node issue, or even what a device node is, trust me, you don’t really want to, go install udev and don’t worry about it...) As stated above, if the kernel decides to start randomly assigning major numbers to all devices, then udev will still work just fine.

- Provide a persistent device naming solution: Lots of people want to assign a specific name that they can talk to a device to, no matter where it is in the system, or what order they plugged the device in. USB printers, SCSI disks, PCI sound cards, Firewire disks, USB mice, and lots of other devices all need to be assigned a name in a consistent manner (udev doesn’t handle network devices, naming them is already a solved solution, using nameif). udev allows users to create simple rules to describe what device to name. If users want to call a program running a large database half-way around the world, asking it what to name this device, it can. We don’t put the naming database into the kernel (like other Unix variants have), everything is in userspace, and easily accessible. You can even run a perl script to name your device if you are that crazy...
For more information on how to create udev rules to name devices, please see the udev man page, and look at the example udev rules that ship with the tarball.

So, convinced already why you should use udev instead of devfs? No.

Ok, fine, I’m not forcing you to abandon your bloated, stifling policy, nonextensible, end of life feature if you don’t want to. But please don’t bother me about it either, I don’t care about devfs, only about udev.

This is my last posting about this topic, all further emails sent to me about why devfs is wonderful, and why are you making fun of this wonderful, stable gift from the gods, will be gleefully ignored and possibly posted in a public place where others can see.

thanks,
greg k-h

Links

D-BUS project  See Also:
• http://www.freedesktop.org/software/dbus/
• http://cgit.freedesktop.org/dbus/dbus/tree/

Linux Filesystem Hierarchy Standard LFHS  See Also:
http://www.pathname.com/fhs/2.2

The Linux Assigned Names And Numbers Authority  See Also:
http://www.lanana.org/

Linux Weekly News

udev rules  See Also:
• http://doc.ubuntu-fr.org/udev (in french)
• http://www.reactivated.net/writing_udev_rules.html
• http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev_vs_devfs

Introduction  udev rules are flexible and very powerful. Here are some of the things you can use rules to achieve:

• Rename a device node from the default name to something else.
• Provide an alternative/persistent name for a device node by creating a symbolic link to the default device node.
• Name a device node based on the output of a program.
• Change permissions and ownership of a device node
• Launch a script when a device node is created or deleted (typically when a device is attached or unplugged)
• Rename network interfaces

Writing rules is not a workaround for the problem where no device nodes for your particular device exist. Even if there are no matching rules, udev will create the device node with the default name supplied by the kernel.
History  udev is targeted at Linux kernels 2.6 and beyond to provide a userspace solution for a dynamic /dev directory, with persistent device naming. The previous /dev implementation, devfs, is now deprecated, and udev is seen as the successor. udev vs devfs is a sensitive area of conversation.

Over the years, the things that you might use udev rules for has changed, as well as the flexibility of rules themselves. On a modern system, udev provides persistent naming for some device types out-of-the-box, eliminating the need for custom rules for those devices.

Terminology: devfs, sysfs, nodes, etc.  A basic introduction only, might not be totally accurate.

On typical Linux-based systems, the /dev directory is used to store file-like device nodes which refer to certain devices in the system. Each node points to a part of the system (a device), which might or might not exist.

Userspace applications can use these device nodes to interface with the systems hardware, for example, the X server will “listen to” /dev/input/mice so that it can relate the user’s mouse movements to moving the visual mouse pointer.

The original /dev directories were just populated with every device that might possibly appear in the system. /dev directories were typically very large because of this. devfs came along to provide a more manageable approach (noticeably, it only populated /dev with hardware that is plugged into the system), as well as some other functionality, but the system proved to have problems which could not be easily fixed.

udev is the “new” way of managing /dev directories, designed to clear up some issues with previous /dev implementations, and provide a robust path forward. In order to create and name /dev device nodes corresponding to devices that are present in the system, udev relies on matching information provided by sysfs with rules provided by the user. This documentation aims to detail the process of rule-writing, one of the only udev-related tasks that must (optionally) be performed by the user.

sysfs is a new filesystem to the 2.6 kernels. It is managed by the kernel, and exports basic information about the devices currently plugged into your system. udev can use this information to create device nodes corresponding to your hardware. sysfs is mounted at /sys and is browseable. You may wish to investigate some of the files stored there before getting to grips with udev.

Why using udev rules?  udev rules are flexible and very powerful. Here are some of the things you can use rules to achieve:

• Rename a device node from the default name to something else
• Provide an alternative/persistent name for a device node by creating a symbolic link to the default device node
• Name a device node based on the output of a program
• Change permissions and ownership of a device node
• Launch a script when a device node is created or deleted (typically when a device is attached or unplugged)
• Rename network interfaces

Writing rules is not a workaround for the problem where no device nodes for your particular device exist. Even if there are no matching rules, udev will create the device node with the default name supplied by the kernel.

Create a plugdev group  See Also:

http://www.linux-usb.org/usb.ids

Before creating the rules, you have to create a ‘‘plugdev’’ group.

Before creating the udev rule we must:

• create a plugdev group (if it does not already exist):
groupadd plugdev

- add the users who will use the CL1356A+ smartcard reader to this group:
  
sudo gpasswd -a $USERNAME plugdev

> groups $USERNAME

user : user adm dialout cdrom plugdev fuse lpadmin netdev admin sambashare

**For “udev” systems**  In `/etc/udev/rules.d/41-flip.rules` I have (I am part of the group “flip”):

```bash
#add support AT89C5132 AT89C51SND1 AT89C51SND2 AT89C51SND2 AT89C52TARGET
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2fff", GROUP="flip", MODE="0660"

#add support AT89C5130 AT89C5131
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ffd", GROUP="flip", MODE="0660"

#add support AT90USB1286 AT90USB1287
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ffb", GROUP="flip", MODE="0660"

#add support AT90USB162
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ffa", GROUP="flip", MODE="0660"

#add support AT90USB647 AT90USB646
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ff9", GROUP="flip", MODE="0660"

#add support AT32UC3A0128 AT32UC3A0256 AT32UC3A0512 AT32UC3A1128 AT32UC3A1256 AT32UC3A1512
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ff8", GROUP="flip", MODE="0660"

#add support AT90USB82
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ff7", GROUP="flip", MODE="0660"

#add support AT32UC3B0128 AT32UC3B0256 AT32UC3B064 AT32UC3B1128 AT32UC3B1256 AT32UC3B164
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="03eb", ATTRS{idProduct}=="2ff6", GROUP="flip", MODE="0660"
```

Remember to run `/sbin/udevstart` after adding the file. Also add group ‘flip’ and add necessary users.

There is a file that came with batchisp and it reads The Linux platform used different systems for device drivers. Older systems uses hotplug while newer systems uses udev.

**Most current solution: use udev**  The latest and greatest way of managing hotplugged (and cold-plugged) devices under Linux is called `udev <udev_readme>`. This is a development of the older “hotplug” system (see below).

When a device is connected, the kernel will call the program `/sbin/udev` in order to create a device node in the `/dev` file hierarchy. It will also remove devices from this hierarchy when they are unplugged.

Traditionally, all devices plugged into a Linux system are expected to have a kernel device driver, or to load one on-the-fly when a new device is connected.

Libusb cannot use these device drivers, instead it attempts to access the raw device nodes from user mode, not as a kernel module.

In order for libusb to find the device node, it needs to locate it in the `/dev` filesystem.
The recommended way to let udev create nodes in the /dev filesystem is to add a udev rule like the following into some foo.rules file inside the /etc/udev/rules.d/ directory:

```bash
# usbfs-like devices SUBSYSTEM=="usb", PROGRAM="/bin/sh -c 'K=%k; K=$${K#usbdev}; printf bus/usb/%03i/%03i $$K%%%%.*} $$K#*.'}', NAME="%c"
```

This layout is used by for example the Debian distribution and Fedora Core. This rule creates a device tree identical to the earlier /proc/bus/usb/ tree, but under /dev/bus/usb/ instead. If this device tree exists, libusb will default to use it. It will look like this:

```
/dev
/bus
/usb
/001
/002
/003
/001
/002
/003
...
```

However notice that the permissions on the nodes will be default permissions: often this means they are only accessible for writing by the root user, whereas non-root users often can access it read-only.

The way of controlling access to a device node differs between systems, but a typical way of complementing udev rules with appropriate permissions is to use PAM (pluggable Authentication Modules), with some sort of configuration under /etc/security/ (For details on this, see below.)

The use of /dev nodes is also different from the old usbfs solution in that it enables the use of ACL:s (Access Control Lists) to control access for the USB device nodes.

**Warning:** ACL:s could not be used on the /proc filesystem.

A less good alternative that may however be useful for debugging would be to supply the argument MODE="666" to the above udev rule, or, slightly better, to tag on: MODE="660", GROUP="foo"

where “foo” is a group of users (e.g. desktop users) that need to access the device in read/write mode.

If libusb cannot find a device hierarchy below /dev/bus/usb/ (as is the case if you are not using udev, or not using it with the above rule), it will fall back on using /proc/bus/usb/ instead.

Additionally, you may want to trigger unique actions for your device at the same time. To do this, create a rules file /etc/udev/rules.d/bar.rules with these lines:

```
SUBSYSTEM=="usb", ACTION=="add", ATTRS{idVendor}=="1234", ATTRS{idProduct}=="4321"
```

At the end of this line you can then tag on any device-specific actions for device 1234/4321, for example:

```
MODE="660", GROUP="baz" to set mode and group
RUN="/usr/local/bin/baz" to run a script on plug-in
SYMLINK="foo" to create a symlink device node with this name in /dev
```

You can read more about udev in its own documentation.

**Permissions setting with PAM** In addition to the udev rule for creating the device node you will want to change the permissions on the new node, unless it defaults to something that is globally writeable and readable.

---

**2.22. Operating systems**
Making anything that is plugged in on the USB bus writeable and readable by ALL users is typically a bad idea, because what you most typically want to do is to make it writeable and readable for the console user, i.e. the person that happens to sit behind the screen and keyboard of this very computer.

Managing this by groups is a bit kludgy: it means you set up a group for all console users and add all users that may use the console to this group. This also means that one user that is a member of this group could be at the console plugging his USB keydrive in, while another user of the same group is logged in remotely, and making a blank copy of the same keydrive at the same time, for example.

Since Linux is used in a strict multi-user context, this has to be solved: **give permissions to hotplugged devices only to the console user.**

Fedora Core 5 and later does this by using PAM. Whenever something happens in udev, PAM is called to modify the permissions on anything that appeared in the file system in accordance to a set of security rules.

The trick is to create a symbolic link for your new device, then let PAM match the name of this link and change the permissions of it.

For example, in `/etc/udev/rules.d/foo.rules` you write:

```bash
SUBSYSTEM=="usb", ACTION=="add", ATTRS{idVendor}=="1234", \ ATTRS{idProduct}=="4321", SYMLINK+="foo-%k"
```

This will create a symlink named “/dev/foo-nn” where nn is some unique number for each added device matching this VID and PID.

You then set up PAM console rules in accordance, by adding a `/etc/security/console.perms.d/foo.perms` containing:

```bash
<foo>=/dev/foo*
<console> 0600 <foo> 0600 root
```

This instructs PAM to give the console user (and root) read and write permissions to the new symlink, whenever it appears. The permission change on the symlink will then fall through to the new device node.

**Previous solution: use hotplug** Before udev another system, generally considered less elegant, known simply as “hotplug” was used. In this case the program `/sbin/hotplug` would be called whenever devices were connected or removed from the system, and the corresponding configuration lives in `/etc/hotplug/`.

With hotplug not using udev at the same time, all devices are accessed using the usbfs hierarchy below `/proc/bus/usb/`. Again, this http://mihd.net/y7w20q will be used by libusb, since libusb does not use any device drivers. The hierarchy will look like this:

```
/proc
/bus
/usb
/001
/001
/002
/002
/001
/002
... 
```

When USB devices are plugged in, their corresponding device node is created in `/proc/bus/usb/` by the kernel, without any external program intervention (as is the case with udev).
However, to correct the permissions on these device nodes, if your device requires anything else than read access, you need to supply a script in /etc/hotplug/usb/ that detects your device and change its permissions, for example this /etc/hotplug/usb/foo.usermap:

```
# Foo device with VID=1234 and PID=4321
bar 0x0003 0x1234 0x4321 0x0000 0x0000 0x00 0x00 0x00 0x00 0x00 0x00 0x00000000
```

(All this need to be in one line.)

The first string “bar” points out the name of a script placed in /etc/hotplug/usb/bar, with for example the following contents:

```
#!/bin/bash
if [ "${ACTION}" = "add" ] && [ -f "${DEVICE}" ] then chgrp baz "${DEVICE}"
chmod 660 "${DEVICE}"
fi
```

to let users in the group “baz” access the device for reading and writing. There exist solutions similar to the PAM permission change for hotplug, but they are all kind of hackish.

You can read more about hotplug and its usermaps in the hotplug documentation. Maybe someone with more Linux experience can understand this better.

**For “hotplug” systems**

- The files libavrtools and libavrtools.usermap should be placed in /etc/hotplug/usb.
- The group avrtools should be created, containing the users which are allowed to use the JTAGICE mkII devices.

**udev rules**

See Also:

- [http://www.reactivated.net/writing_udev_rules.html](http://www.reactivated.net/writing_udev_rules.html)

**Preambule**

udev is targeted at Linux kernels 2.6 and beyond to provide a userspace solution for a dynamic /dev directory, with persistent device naming.

The previous /dev implementation, devfs, is now deprecated, and udev is seen as the successor. udev vs devfs is a sensitive area of conversation

Over the years, the things that you might use udev rules for has changed, as well as the flexibility of rules themselves. On a modern system, udev provides persistent naming for some device types out-of-the-box, eliminating the need for custom rules for those devices.

**Terminology: devfs, sysfs, nodes, etc.**

A basic introduction only, might not be totally accurate.

On typical Linux-based systems, the /dev directory is used to store file-like device nodes which refer to certain devices in the system. Each node points to a part of the system (a device), which might or might not exist.

Userspace applications can use these device nodes to interface with the systems hardware, for example, the X server will “listen to” /dev/input/mice so that it can relate the user’s mouse movements to moving the visual mouse pointer.

The original /dev directories were just populated with every device that might possibly appear in the system. /dev directories were typically very large because of this. devfs came along to provide a more manageable approach (noticeably, it only populated /dev with hardware that is plugged into the system), as well as some other functionality, but the system proved to have problems which could not be easily fixed.
udev is the “new” way of managing /dev directories, designed to clear up some issues with previous /dev implementations, and provide a robust path forward. In order to create and name /dev device nodes corresponding to devices that are present in the system, udev relies on matching information provided by sysfs with rules provided by the user. This documentation aims to detail the process of rule-writing, one of the only udev-related tasks that must (optionally) be performed by the user.

sysfs is a new filesystem to the 2.6 kernels. It is managed by the kernel, and exports basic information about the devices currently plugged into your system. udev can use this information to create device nodes corresponding to your hardware. sysfs is mounted at /sys and is browseable. You may wish to investigate some of the files stored there before getting to grips with udev.

**Why using udev rules?** udev rules are flexible and very powerful. Here are some of the things you can use rules to achieve:

- Rename a device node from the default name to something else
- Provide an alternative/persistent name for a device node by creating a symbolic link to the default device node
- Name a device node based on the output of a program
- **Change permissions and ownership of a device node**
- Launch a script when a device node is created or deleted (typically when a device is attached or unplugged)
- Rename network interfaces

Writing rules is not a workaround for the problem where no device nodes for your particular device exist. Even if there are no matching rules, udev will create the device node with the default name supplied by the kernel.

In our specific case, we want that any user in the plugdev group can use the id3 certis2 fingerprint reader.

### 99-id3-fingerprint-reader.rules

Create a plugdev group Before creating the udev rule we must:

- create a plugdev group (if it does not already exist):
  ```
groupadd plugdev
  ```
- add the users who will use the Certis2 reader to this group:
  ```
gpasswd -a $USERNAME plugdev
  ```

> groups $USERNAME

```
user : user adm dialout cdrom plugdev fuse lpadmin netdev admin sambashare
```

Create the udev rule Any user in the plugdev group will be able to use the Certis2.

> cd /etc/udev/rules.d
> udevadm control --reload-rules
> more 99-id3-fingerprint-reader.rules

```
ATTRS{idVendor}=="0b81", ATTRS{idProduct}=="0103", SUBSYSTEMS=="usb", ACTION=="add", MODE="0666", GROUP="plugdev"
```
Test  If you type:

<pre>
> lsusb
</pre>

You can see the Certis2 reader which is not the case if the user is not root or is not in the plugdev group:

<pre>
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 011: ID 0b81:0103 id3 Semiconductors Certis V2 fingerprint reader
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 003: ID 0bda:0111 Realtek Semiconductor Corp. Card Reader
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
</pre>

**udev rules examples**  See Also:  

*udev rules*

**Smartcard udev rules to access to the smartcard reader**  See Also:  

*udev rules*

In our specific case, we want that any user in the plugdev group can use the id3 smartcard reader.

Create the udev rule file

Prerequisite  See Also:  

Create a plugdev group

You are a ‘root’ user.

Any user in the plugdev group will be able to use the CL1356A+ smartcard reader (VendorId=0x0b81, ProductId=0x0200).

<pre>
> cd /etc/udev/rules.d
> udevadm control --reload-rules
> more 99-id3.rules
</pre>

```
ATTRS{idVendor}=="0b81", ATTRS{idProduct}=="0200", SUBSYSTEMS=="usb", \ ACTION=="add", MODE="0666", GROUP="plugdev"
```

/etc/udev/rules.d directory

```bash
drwxr-xr-x 3 root root 4096 2011-04-11 17:06 ./
drwxr-xr-x 3 root root 4096 2011-01-10 15:35 ../
-rw-r--r-- 1 root root  406 2011-01-10 16:03 70-persistent-cd.rules
-rw-r--r-- 1 root root  595 2011-01-10 10:46 70-persistent-net.rules
-rw-r--r-- 1 root root  115 2010-04-19 11:30 README
```
Test: plugging the CL1356A+ smartcard reader  If you type:

> lsusb

You can see the CL1356A+ reader which is not the case if the user is not root or is not in the plugdev group:

```
Bus 006 Device 002: ID 0b81:0200 id3 Semiconductors CL1356T / CL1356A / CL1356T5 smartcard readers
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 002: ID 0bda:0111 Realtek Semiconductor Corp. Card Reader
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Serial udev rules

Create the udev rule file

**Prerequisite**  You are a ‘root’ user.

```
pvergain@houx: ls -l /dev/ttyACM0
```

```
crw-rw---- 1 root dialout 166, 0 27 sept. 11:24 /dev/ttyACM0
```

Only the users who belong to the root or dialout groups can access to the device.

So we have to add the users who will use serial device to this group:

```
gpasswd -a $USERNAME dialout
```

Example:

```
gpasswd -a pvergain dialout
```

Depending on your OS, you have to reboot (fedora 15 by example).

```
dmesg
```

[ 1414.179055] usb 6-2: new full speed USB device using ohci_hcd and address 2
[ 1414.336096] usb 6-2: New USB device found, idVendor=0b81, idProduct=8004
[ 1414.336107] usb 6-2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 1414.336114] usb 6-2: Product: MEABOARD
[ 1414.336119] usb 6-2: Manufacturer: id3 semiconductors
[ 1414.336124] usb 6-2: SerialNumber: 00000000
[ 1414.661889] cdc_acm 6-2:1.0: This device cannot do calls on its own. It is not a modem.
[ 1414.662253] cdc_acm 6-2:1.0: ttyACM0: USB ACM device
[ 1414.666421] cdc_acm: v0.26:USB Abstract Control Model driver for USB modems and ISDN adapters
> cd /etc/udev/rules.d
> udevadm control --reload-rules
> more 99-id3_serial.rules

ATTRS{idVendor}=="0b81", ATTRS{idProduct}=="8004", SUBSYSTEMS=="tty", \nATTRS{SerialNumber}=="00000000", SYMLINK += "cle_rfid"

Python reading test  See Also:

Serial Python Communication

#!/usr/bin/python2

import serial

device_port = serial.Serial("/dev/serial/by-id/usb-id3_semiconductors_MEABOARD_00000000-if00", 115200)
while 1:
    answer = device_port.read(6)
    print answer
device_port.close()

Test result

Msg000
Msg001
Msg002
Msg003
Msg004
Msg005
Msg006
Msg007
Msg008
Msg009
Msg010
Msg011
Msg012
Msg013
Msg014
Msg015
Msg016
Msg017

fingerprint udev rules  See Also:

udev rules

In our specific case, we want that any user in the plugdev group can use the id3 certis2 fingerprint reader.

Create the udev rule  Any user in the plugdev group will be able to use the Certis2.
> cd /etc/udev/rules.d
> udevadm control --reload-rules
> more 99-id3-fingerprint-reader.rules

ATTRS{idVendor}=="0b81", ATTRS{idProduct}=="0103", SUBSYSTEMS=="usb", \ ACTION=="add", MODE="0666", GROUP="plugdev"

Test  If you type:

> lsusb

You can see the Certis2 reader which is not the case if the user is not root or is not in the plugdev group:

Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 011: ID 0b81:0103 id3 Semiconductors Certis V2 fingerprint reader
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 003: ID 0bda:0111 Realtek Semiconductor Corp. Card Reader
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

udev tools

udevdiscover  See Also:

http://fontanon.org/udevdiscover/download-and-install/

Installation tips  Udev-discover GUI has been developed with Gtk3. It needs Gobject Introspection Repository (gir) bindings for python.

Dependencies:
  • python >= 2.6
  • python-gobject
  • python-gudev
  - gir1.2-gtk-3.0 - gir1.2-gconf-2.0 - gir1.2-gdkpixbuf-2.0

systemd  See Also:

  • http://www.freedesktop.org/wiki/Software/systemd
  • http://fedoraproject.org/wiki/Features/systemd
  • http://0pointer.de/blog/projects/socket-activation.html

systemd is a system and service manager for Linux, compatible with SysV and LSB init scripts. systemd provides aggressive parallelization capabilities, uses socket and D-Bus activation for starting services, offers on-demand starting of daemons, keeps track of processes using Linux cgroups, supports snapshotting and restoring of the system state, maintains mount and automount points and implements an elaborate transactional dependency-based service control logic. It can work as a drop-in replacement for sysvinit.
See Lennart’s blog story for a longer introduction, and the two status updates since then. Also see the Wikipedia article.

Socket activation  See Also:
http://0pointer.de/blog/projects/socket-activation.html

In the original blog story about systemd I tried to explain why socket activation is a wonderful technology to spawn services. Let’s reiterate the background here a bit.

systemd use cases

systemd and pcscd  See Also:

License: by-nc-sa Creative Commons Licence This blog by Ludovic Rousseau is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License.

Tuesday, November 15, 2011 pcscd auto start using systemd  In pcsc-lite version 1.6.0 (5 May 2010) I introduced a mechanism to start the pcscd daemon only when an application called SCardEstablishContext() to get a PC/SC context.

Read Configuring your system for pcscd auto-start for more information.

Old way: start by the libpcsclite library  One of the problem is that pcscd is then started as the user running the application. The pcscd process has to be setgid to get special privileges to get access to the USB devices (the smart card readers). Some people (and I understand) do not like files with special access right (like setgid).

New way: systemd  Around the same time (23 Aug 2010) the first article about systemd was posted by systemd author: Lennart Poettering

Socket activation  I will not describe systemd in details. But one of the services of systemd is to start a process when an application tries to communicate to a socket. The good news is that libpcsclite and pcscd are communicating through a UNIX (local) socket. So systemd provides a way to start pcscd when libpcsclite starts the communication.

The good news is that the changes are really minimal. libpcsclite has not changed and only the initialisation of pcscd has been updated a bit to use systemd services.

systemd configuration  pcsc-lite now use 2 systemd configuration files: pcscd.service and pcscd.socket

File pcscd.service

[Unit]
Description=PC/SC Smart Card Daemon
Requires=pcscd.socket

[Service]
ExecStart=/usr/sbin/pcscd --foreground --auto-exit
ExecReload=/usr/sbin/pcscd --hotplug
StandardOutput=syslog

[Install]
Also=pcscd.socket

File pcscd.socket

[Unit]
Description=PC/SC Smart Card Daemon Activation Socket

[Socket]
ListenStream=/var/run/pcscd/pcscd.comm

[Install]
WantedBy=sockets.target

These two files are installed in /lib/systemd/system/ on my Debian (testing) system.

Status

$ systemctl status pcscd.service
pcscd.service - PC/SC Smart Card Daemon
 Loaded: loaded (/lib/systemd/system/pcscd.service)
 Active: inactive (dead) since Mon, 14 Nov 2011 16:33:40 +0100; 17h ago
 Process: 15489 ExecStart=/usr/sbin/pcscd --foreground --auto-exit (code=exited, status=0/SUCCESS)
 CGroup: name=systemd:/system/pcscd.service

$ systemctl status pcscd.socket
pcscd.socket - PC/SC Smart Card Daemon Activation Socket
 Loaded: loaded (/lib/systemd/system/pcscd.socket)
 Active: active (listening) since Mon, 14 Nov 2011 14:49:45 +0100; 19h ago
 CGroup: name=systemd:/system/pcscd.socket

The pcscd socket is configured and active but the pcscd process is not running. After starting a PC/SC application we get:

$ systemctl status pcscd.service
pcscd.service - PC/SC Smart Card Daemon
 Loaded: loaded (/lib/systemd/system/pcscd.service)
 Active: active (running) since Tue, 15 Nov 2011 10:27:39 +0100; 9s ago
 Main PID: 26929 (pcscd)
 CGroup: name=systemd:/system/pcscd.service
 26929 /usr/sbin/pcscd --foreground --auto-exit

Stop It is possible to stop the running pcscd process.

$ sudo systemctl stop pcscd.service
$ systemctl status pcscd.service
pcscd.service - PC/SC Smart Card Daemon
 Loaded: loaded (/lib/systemd/system/pcscd.service)
But in general I just kill(1) the pcscd process.

Start If you start pcscd by hand, for example in debug and foreground mode, the daemon will remove the socket /var/run/pcscd/pcscd.comm on exit. This socket is not recreated automatically by systemd. You need to stop and start the pcscd.socket, using just start is not enough.

    $ ls /var/run/pcscd/pcscd.comm
    ls: cannot access /var/run/pcscd/pcscd.comm: No such file or directory
    $ sudo systemctl stop pcscd.socket
    $ sudo systemctl start pcscd.socket
    $ ls /var/run/pcscd/pcscd.comm
       /var/run/pcscd/pcscd.comm

The pcscd process is not started but the socket is now listening.

Migration I removed the old autostart code in revision 6105. If you have systemd installed on your system I recommend using it to start pcscd. If you do not have systemd installed (maybe you do not use a Linux kernel) then you have to start pcscd at boot (as before version 1.6.0).

    systemd availability systemd is only available with a Linux kernel. systemd is now installed by default in Fedora 14. systemd is provided by Debian but Debian is not just limited to a Linux kernel. Debian also provides Hurd and FreeBSD kernel based Debian systems and systemd is not (yet) available for these kernels.

Major GNU/Linux distributions should provide systemd now. If your distribution do not have systemd you can still use the old way of starting pcscd at boot.

    For non-Linux systems I have no auto-start solution. Just start pcscd at boot.

Conclusion systemd is a nice new system to replace init and a lot more. It is a good piece of code to implement auto start for pcsc-lite.

Thanks to Kalev Lember for pushing the systemd patches.

GNU/linux kernel tools

buildroot linux tool (Making Embedded Linux easy) See Also:  
   - http://buildroot.uclibc.org/ 
   - Making Embedded Linux easy 

Buildroot is a set of Makefiles and patches that makes it easy to generate a complete embedded Linux system.

Buildroot can generate any or all of a cross-compilation toolchain, a root filesystem, a kernel image and a bootloader image.

Buildroot is useful mainly for people working with small or embedded systems, using various CPU architectures (x86, ARM, MIPS, PowerPC, etc.) : it automates the building process of your embedded system and eases the cross-compilation process.

The major Buildroot features are:
• Can handle everything in your embedded system development project: cross-compiling toolchain, root filesystem generation, kernel image compilation and bootloader compilation. Buildroot is also sufficiently flexible that it can also be used for only one or several of these steps.

• Is very easy to set up, thanks to its menuconfig, gconfig and xconfig configuration interfaces, familiar to all embedded Linux developers. Building a basic embedded Linux system with Buildroot typically takes 15-30 minutes.

• Supports several hundreds of packages for userspace applications and libraries: X.org stack, Gtk2, Qt, DirectFB, SDL, GStreamer and a large number of network-related and system-related utilities and libraries are supported.

• Supports multiple filesystem types for the root filesystem image: JFFS2, UBIFS, tarballs, romfs, cramfs, squashfs and more.

• Can generate an uClibc cross-compilation toolchain, or re-use your existing glibc, eglibc or uClibc cross-compilation toolchain

• Has a simple structure that makes it easy to understand and extend. It relies only on the well-known Makefile language.

Buildroot is maintained by Peter Korsgaard, and licensed under the GNU GENERAL PUBLIC LICENSE V2 (Or later).

Stable releases are delivered every three months.

Versions

buildroot versions

buildroot 2011.11  See Also:
http://git.buildroot.net/buildroot/plain/CHANGES?id=2011.11

2011.11, Released November 30th, 2011  Fixes all over the tree.
Bump kernel headers / default Linux version to 3.1.4.
Updated/fixed packages: ruby

2011.11-rc3, Released November 26th, 2011  Fixes all over the tree.
Toolchain: Fix gdb dependencies for external toolchains, adjust uClibc patches so they don’t confuse modern versions of patch, bump crosstool-ng, kernel headers and linux versions.
Updated/fixed packages: busybox, freetype, mplayer, opencv, php, rsyslog, ruby, thttpd, xapp_xf86dga
Issues resolved (http://bugs.uclibc.org):

#4357: Prevent patch commands from accessing source control #4369: Fix permissions on untarred lsof archive

2011.11-rc2, Released November 18th, 2011  Fixes all over the tree and new features.

Updated asciidoc documentation

Toolchain: Bumped 3.x stable kernel headers, use wget in crosstool-ng as well, bump crosstool-ng version, gdb fixes, uClibc sparc fix.

Updated/fixed packages: distcc, file, gst-plugins-bad, libxcb, mplayer, newt, qt, rpm, rrdtool, tar, tftpd

Issues resolved (http://bugs.uclibc.org):

#3355: mplayer fails to build #4021: uClibc: undefined reference to ‘__GI___errno_location’ #4297: Qt’s qmake uses wrong pkg-config

2011.11-rc1, Released November 11th, 2011  Fixes all over the tree and new features.

Moved misc scripts and support stuff to support/. Renamed patch-kernel.sh to support/scripts/apply-patches.sh.

Documentation: Moved to asciidoc format, make targets to generate text/html/pdf/epub output added.

Defconfigs: Qemu configs updated to 3.1 kernel and readmes added.

Bootloaders: Add support for custom git tree / tarballs for barebox, similar to how it’s handled for u-boot. Clean up menuconfig options.

Toolchain: Update external codesourcery toolchain download URLs after Codesourcery got bought by Mentor, add x86 toolchain, update toolchain versions and optimize toolchain sysroot copying. Fix uClibc 0.9.32 builds for e500 PPC, updated GDB versions / download URLs. Binutils libbfd/libopcodes static/dynamic linking fix. GCC 4.6.2 added, use ctng-1.13.0.

Package infrastructure: Support for local packages / overrides, package dir / name arguments dropped from {GEN,AUTO,CMAKE}TARGETS.

Linux: Kernel extensions infrastructure support, Xenomai + RTAI support.

Updated/fixed packages: acpid, bind, busybox, dash, dbus, dbus-glib, directfb, dnsmasq, drystone, e2fsprogs, ethtool, fakeroot, fbgdump, file, freetype, fuse, gamin, gmp, gmpc, gnutls, gob2, gstreamer, hostapd, ifplugd, imagemagick, intltool, ipsec-tools, ipset, iptables, iw, jpeg, kexec, leafpad, less, libargtable2, libao, libconfuse, libcfile, libiconv, liblua, libevent, libglib2, libiconv, libmpd, libping, libplaygen, libxml2, libxmlreq, libxslt, libxt, linux-fusion, lite, lrzsz, lsif, lzo, lzop, makedevs, mcookie, mpg123, mpd, mpfr, mtd, musepack, mutt, mysql_client, neovim, netcat, netsnmp, nftables, ntp, openntpd, openssh, openvpn, opencv, oracle, pciconf, pdb, psmisc, python, qt, radvd, rpm, rsync, samba, sawdust, sdSL_sound, smartmontools, slocate, squid, stunnel, sudo, sylpheed, syslog, taglib, tar, tcpreplay, tsklib, util-linux, valgrind, wget, whetstone, which, wpapy-patch, xdata_xcursor-themes, xmlstarlet, xterm

New packages: bluez-utils, cifs-utils, fftw, fluxbox, json-c, libev, libfreetype, libgeotiff, libmodbus, libplayer, live555, ngrep, noip, opencv, opencvcore, picocom, picoaudio, pulseaudio, pv, rtai, vala, xenomai.

Removed packages: liboil, sfdisk, swfdec, webif

Issues resolved (http://bugs.uclibc.org):

#505: live555: new package #507: Enable live and tv options in MPlayer-1.0rc2 #531: let e2fsprogs package to export headers to staging dir if needed #1171: Linuxthreads new cannot find sysdep.h #1357: Add bluez to buildroot system #2107: New package: input-event-daemon #2599: New package: orc (Oil Runtime Compiler) #2605: gstreamer: Update to 0.10.30 #2677: introducing util-linux-ng for replaces util-linux #2917: Qt: Add declarative module #3145: jffs2 image generation fails #3271: netperf-2.4.5 fails to compile #3331: xdata_xcursor-themes depends on xcursorgen #3343: Add file:/// download SITE_METHOD #3391: Add support for specifying an external kernel tree

2.22. Operating systems 833
#3631: Error while compiling with Xorg  
#3709: oprofile doesn’t build for mipsel  
#3925: midori not getting compile  
#4045: Add support for downloading i386 toolchains from codesourcery  
#4165: lrzsz-fix-symlink-at-rebuild.patch  
#4171: makedevs-unused-but-set-variable.patch  
#4183: Codesourcery toolchain download site has changed  
#4231: libneon.so: undefined reference to ‘SSL_SESSION_cmp’  
#4381: Add option to lighttpd to enable Lua support  
#4387: Make sure that dest dir exists before installing mtd files

People

buildroot people

Thomas Petazzoni  See Also:

- http://thomas.enix.org
- http://free-electrons.com/company/staff/thomas-petazzoni/
- http://git.buildroot.net/buildroot/log/?qt=author&q=thomas.petazzoni
- Thomas Petazzoni

http://identi.ca/tpetazzoni

Figure 2.33: Thomas Petazzoni

Quatre questions à Thomas Petazzoni  See Also:

http://www.toolinux.com/Quatre-questions-a-Thomas


**Que fais-tu dans la vie ?** Je suis ingénieur chez Free Electrons, où je donne des formations et fait du développement et du conseil autour de Linux embarqué. D’ailleurs, les supports de formation de la société sont tous disponibles sous licence libre (CC-BY-SA).

En dehors de ça, je suis l’un des fondateurs et animateurs de Toulibre groupe d’utilisateurs de Logiciels Libres de Toulouse, je travaille également sur le projet Buildroot, un outil de génération de systèmes Linux embarqué) et sur MapOSMatic, un service Web de génération de plans de ville à partir des données OpenStreetMap).


L’objectif du groupe est donc de gérer l’organisation du Libre en Fête, en : Incitant un maximum d’Espaces Publics Numériques à participer.

Pour cela, l’April a mis en place depuis quelques années un partenariat avec la Délégation aux Usages de l’Internet, et plus récemment un partenariat avec le réseau Cyberbase.

**Comment as-tu commencé à participer au groupe ?**

J’ai commencé à participer à l’organisation du Libre en Fête en 2007 en proposant d’utiliser l’Agenda du Libre (dont je suis l’un des créateurs et animateurs) comme support pour le référencement des événements.


Source : April.org

**Support commercial Buildroot**  
[See Also:](http://thomas.enix.org/Blog-20110222201711-Technologie)

Je contribue maintenant depuis deux ans de manière très active au projet Buildroot, un outil de construction de systèmes Linux embarqué.

Je suis le premier contributeur en nombre de patches après le mainteneur du projet, et suis responsable d’une bonne partie des grosses réorganisations/améliorations du code qui ont eu lieu ces deux dernières années.

Du coup, à Free Electrons nous venons de formaliser une offre de support commercial pour Buildroot.

Avec ma forte implication dans le projet, je suis assez bien placé pour aider les entreprises qui veulent mettre en œuvre Buildroot pour leurs projets Linux embarqué. N’hésitez pas à me contacter!

**Séminaire gratuit « Introduction à Linux Embarqué » ce jeudi 15 décembre 2011, près de Lyon**  
[See Also:](http://thomas.enix.org/Blog-20111212194459-Technologie)

Comme je l’ai fait en mars à Montpellier, puis à Limoges et Clermont-Ferrand au mois de juin, puis à Marseille en octobre, je serai ce jeudi 15 décembre de 14h à 18h à Villefontaine près de Lyon pour donner une nouvelle édition du séminaire « Introduction à Linux embarqué ».

L’accès est gratuit après inscription.
Buildroot 2010.05  See Also:
http://thomas.enix.org/Blog-20100602095823-Technologie

Buildroot 2010.05 est sorti juste à temps, le 30 mai. Je continue à contribuer de manière assez active au projet, et c’est assez plaisant: j’ai enfin trouvé un projet sur lequel j’arrive à m’investir sur une durée suffisamment longue pour contribuer de manière significative, être en mesure de répondre aux retours des utilisateurs, relire les patches pour les valider, etc.

Sur cette release, j’ai notamment réalisé la ré-écriture complète du code de génération des systèmes de fichier racine, le support des toolchains externes multilib et pas mal de cleanup de code.

Voir le billet sur le blog de Free Electrons pour tous les détails.

Présentation de Buildroot aux RMLL 2009 par Thomas Petazzoni  See Also:
• http://free-electrons.com/fr/blog/buildroot-rmll2009/

La semaine prochaine, du 7 au 11 juillet, se tiennent à Nantes les Rencontres Mondiales du Logiciel Libre, le rendez-vous majeur de l’année pour la communauté du Logiciel Libre en France.

Un thème Systèmes embarqués et matériel libre proposera un certain nombre de conférences sur nos sujets préférés. J’y ferai une présentation de Buildroot, l’outil de génération de systèmes Linux embarqué, auquel je contribue par ailleurs. Elle aura lieu le jeudi 9 juillet à 10h50.

linux File system hierarchy  See Also:
http://proton.pathname.com/fhs/

Introducing /run  See Also:
• http://linuxfr.org/news/run-or-not-run
• http://lwn.net/Articles/436012/

From: Lennart Poettering <mzerqung-AT-0pointer.de>
To: Fedora Development ML <devel-AT-lists.fedoraproject.org>
Subject: What’s this /run directory doing on my system and where does it come from?
Date: Wed, 30 Mar 2011 13:54:30 +0200

Heya,

I just uploaded a new version of systemd into F15, which establishes a directory /run in the root directory. Most likely you’ll sooner or later stumble over it, so here’s an explanation what this is and why this is.

It’s a fairly minor technical change, though presumably people consider this a bigger political change, so I guess this deserves an explanation:

For quite a while programs involved with early boot used to place runtime data in /dev under numerous hidden dot directories. /dev/.udev was the first one, but over time this grew to at least /dev/.mdadm, /dev/.systemd, /dev/.mount, dracut, initscripts and more tools. (Other distros have even more) The reason they used directories there is that /dev was known to be a tmpfs and available from the first instant the machine was booted. /var/run otoh is only available very late at boot, since /var might reside on a separate file system.
However, /dev always has been an inappropriate and ugly place for runtime data: runtime data is not a device node, and thus simply does not belong there. Also, hiding the existence of directories from the administrator is a bad idea. Then, the fact that some runtime data was placed in /var/run/xxx, and other in /dev/yyyy is often not understandable to the user, and especially when tools originally intended to be used only after boot are needed during early boot a complicated move between these directories needed to take place.

Over time different distributions experimented with different broken solutions for the early-runtime-dir problem: on Debian /lib/init/rw was introduced, a tmpfs fs mounted during early boot. On Ubuntu /var/run was mounted as tmpfs even before /var itself was mounted, with some really ugly bind mount magic. Most software however just sidestepped the issue and used /dev.xxx.

In the past weeks key people from the Debian, Suse, Ubuntu and Fedora camps (and others, too) discussed the whole issue forth and back, to find a solution to stop the misuse of /dev before it becomes even more widespread. Various solutions have been suggested, but in the end it all boiled down to the fact that /var/run does not belong beneath /var and what we really want is a top-level directory /run, and that that is the only really clean solution. The only reason why nobody dared to actually implement such a directory was unwillingness to deal with the political backlash, especially messy discussions on mailing lists like this one.

Understanding this, we came to the conclusion that we should rather implement what everybody thinks is the right technical solution, instead of evading the political backlash for it. And so we implemented this.

With this upload Fedora and Suse have already adopted /run now. Debian folks will suggest this for their coming release. Ubuntu has agreed with introducing /run as well.

**Dracut, udev and systemd have already been updated upstream to make use of /run.**

We expect mdadm and mount to follow suit quickly.

A few years back Debian folks already suggested introduction of /run, and even pinged LSB folks about this, and back then there even was a vaguely positive response from them.

So, what is implemented in F15 precisely?

/run is now a tmpfs, and /var/run is bind mounted to it. /var/lock is bind mounted to /run/lock. Applications can use /run the same way as /var/run. Since the latter is FHS/LSB most apps should just use the latter, only early boot stuff should use /run, for now. The folks who have packages where this applies already have been informed. If you haven’t heard from any of us, then this doesn’t apply to you.

So, what’s the benefit of this again?

- There’s only one tmpfs used, backing /run, /var/lock and /var/run, reducing a bit the ever increasing amount of tmpfs’ used on a default system.

- All runtime data at the same place. systemd’s, udev’s, dracut’s data are all beneath /run and /var/run now. Easily discoverable to the admin. For the first time you can see the data all these important tools used on your system store just like any other by doing “ls /var/run”.

- Nothing is hidden anymore. The admin can see everything beneath /var/run and /run, no hidden dot-files anymore.

- We have standardized the early-runtime-dir solution across all major distributions

- The people involved feel much better since they don’t have to misuse /dev anymore

- The lifecycle properties of directories are clear from the top-level directory name. Lifecycle properties do no longer change the further you go down your tree, i.e. /var is “persistence runtime data” and /run is “volatile runtime data”, and /etc is “persistence system config data”, and so on. The ugliness that /var/run abd /var/lock had completely different lifetime guarantees than /var where they both reside in is gone.

So, this is what is implemented for F15 now. For F16 we will make a minor change on top of this: /var/run and /var/lock will become symlinks to /run (resp /run/lock), so that we don’t have to use bind mounts anymore which are

---

**2.22. Operating systems** 837
not the most beautiful thing to use by default, and confusing to the admin. Due to the implications of symlinks and RPM we didn’t want to make that change in F15.

The actual code changes we needed to implement this scheme were trivial (basically, just bind mount /var/run and /var/lock instead of mounting two new tmpfs’ to them,), which is why we opted to do this so late in the F15 cycle. However, the political implications are much bigger I guess, so let’s see what a fantastic flamewar we can start with this on fedora-devel now. Flame away!

Lennart Poettering - Red Hat, Inc.

**Environment variables**

**GNU/Linux environment**

**The Environment Modules package**

See Also:


The Environment Modules package provides for the dynamic modification of a user’s environment via modulefiles.

Each modulefile contains the information needed to configure the shell for an application.

Once the Modules package is initialized, the environment can be modified on a per-module basis using the module command which interprets modulefiles.

Typically modulefiles instruct the module command to alter or set shell environment variables such as PATH, MAN-PATH, etc. modulefiles may be shared by many users on a system and users may have their own collection to supplement or replace the shared modulefiles.

Modules can be loaded and unloaded dynamically and atomically, in an clean fashion. All popular shells are supported, including bash, ksh, zsh, sh, csh, tcsh, as well as some scripting languages such as perl.

Modules are useful in managing different versions of applications. Modules can also be bundled into metamodules that will load an entire suite of different applications.

See Also:

*Fabric*

**Installation**

**Installation GNU/Linux**

See Also:


Et maintenant, tu indiques que tu n’as qu’une partition qui occupe tout le disque ?

Pour Linux, ton disque c: s’appelle sda1 (1ère partition du 1er disque)

Tu dois donc le réduire avec Gparted pour y mettre les partitions que Ubuntu a besoin !

Pour installer Ubuntu, tu as besoin au minimum d’une partition ext3 pour le système -> 8 Go mini, mais prévoir les programmes à installer et une pour le swap -> mini 256 Mo, mais il vaut mieux indiquer entre 512Mo à 1Go.

Ensuite rien ne t’empêche de créer plusieurs partitions de données.

Un exemple de 4 partitions principales pour ton disque dur de 160 Go :

- Ubuntu : Partition dev/sda1 -> ext 3 - Point de montage / - 10 Go
- Swap : dev/sda2 -> linux-swap - 1 Go

---

838 Chapter 2. Development
Données : Partition dev/sda3 -> ext 3 - Point de montage /Home - 100 Go

Si tu veux pouvoir avoir accès à des données sous Linux et Windows tu peux créer une partition en Fat32 :
• Partition dev/sda4 -> Fat32 - Point de montage media/documents

Exemple2

Bonjour,
Je m’apprete à créer une machine avec un dual boot : Windows 7 et Linux (Ubuntu probablement). Linux me servira à faire de bureautique, de la musique, un peu de vidéo. Windows 7 sera pour les jeux :sarcastic:

J’envisage de partitionner mon disque en 4 :
1) Partition Linux système et programmes Linux.
2) Système Windows 7 et programmes Windows.
3) Mes documents (à voir pour utiliser cette partition à la fois par Linux et 7).
4) Swap Linux, 4go.

Sachant que le DD fait 640 Go, quelle proportions entre les 3 partitions ?? Voilà mon idée :
1) Linux : 100Go.
2) Windows 7 : 200Go
3) Mes documents : 337Go.
4) Swap 4go.

Config hardware :
Asus PQ5-E
Go DDR2
Intel E8400
HD 4870 Sapphire Vapor-x 1Go
WD Black caviar 640Go.

Estimes-tu vraiment avoir besoin d’une partition de swap ? Les tutos que tu trouveras et qui justifient d’usage d’une partition de swap égale au double de la mémoire datent un peu. Un système Linux fonctionne d’autant mieux qu’il dispose de mémoire, et le rapport efficience/mémoire est linéaire. Alors au tant en profiter. Mais ajouter du swap a un système déjà largement doté de mémoire vive est inutile. Sauf pour certains modes de veille. Là, ça se justifie, donc.

Concernant la partition partagée, et Windows ne connaissant que son nombril, ce sera à Linux de faire l’effort de lire et d’écrire dans le format de l’autre.

Mais si permettre à Linux d’accéder aux fichiers de Windows n’est pas idiot, je ne pense pas que faire d’une partition NTFS et d’une arborescence Windows le point de montage de /HOME soit une excellente idée.

Autre chose. 100 Go pour les programmes Linux ! Mais qu’est ce que tu vas installer ? Réserve plutôt 10 Go pour ton système libre, et 90 Go pour /HOME.

Après, pour Windows, c’est à toi de voir.

Issue tracker

linux issue tracker      See Also:
https://bugzilla.kernel.org/

This is the Kernel Tracker system (based on Bugzilla) for posting bugs against the mainline Linux kernels(not distribution kernels).

If you have problems or questions related to the Kernel Tracker itself, please contact the bugme admin or submit a bug report against it.

2.22. Operating systems 839
You can find the answer to some of your questions in the FAQ page too.

All new categories are created owned by “virtual users”.

You may also want to read the Kernel Bug Tracker User’s Guide to find out more about Kernel Bug Tracker and how to use it.


Thomas Petazzoni est un des principaux développeurs de Buildroot, un outil de construction de systèmes Linux embarqué, et utilisateur/développeur sous système Linux depuis plus de dix ans.
Embedded operating systems

Linux embedded videos

video Linux conferences embedded (2011) See Also:

Vidéos de l'ELCE 2009 See Also:
- http://linuxfr.org/2010/03/24/26636.html
- http://embeddedlinuxconference.com/elc_europe09/

Videos from the Embedded Linux Conference Europe, Grenoble, October 2009

Just a few weeks before the next edition of the Embedded Linux Conference in San Francisco, here are the videos from the previous edition in Europe a few months ago.

These videos were shot by Satoru Ueda and Tim Bird (Sony), Ruud Derwig (NXP) and by Thomas Petazzoni and Michael Opdenacker (Free Electrons). As usual, they are released under the terms of the Creative Commons Attribution – ShareAlike Licence version 3.0.

Ruud Derwig

If you have never been to an Embedded Linux Conference yet, these videos should show you how useful this conference is for embedded Linux system developers. This is the place where you can discover new

Figure 2.34: Thomas Pettazzoni

Figure 2.35: Linux embedded video 2009 Grenoble
development tools and technologies that will change your working life, benefit from the experience from your peers, get the opportunity to talk to the fantastic people who implement the Free and Open Source software that makes your system run, and win cool penguin goodies. So, don’t miss next next edition in San Francisco. It’s still time to register.

Les vidéos de l’édition 2009 de l’Embedded Linux Conference Europe (ELCE), qui s’est tenue pour la première fois en France à Grenoble en octobre dernier, sont enfin disponibles en licence libre sur le site de Free Electrons.

Si vous êtes intéressés par le thème des logiciels libres et de l’embarqué, vous trouverez un large spectre de sujets couverts par des intervenants issus du monde entier.

Cela peut également vous permettre de faire connaissance avec des acteurs français dans ce domaine, venus en nombre cette année:

- Grégory Clément (Adeneo),
- Florian Fainelli (OpenWRT),
- Philippe Gerum (Sourcetrek),
- Nicolas Palix (Université de Copenhague),
- Pascal Pellet (e2v),
- Cédric Hombourger (Montavista),
- Pierre Ficheux (OS4i),
- Jean-Pierre André (Tuxera),
- Yann E. Morin (Crosstool-ng),
- Jean-Marc Temmos (Visteon Software),
- Samuel Ortiz (Intel)
- Michael Opdenacker (Free Electrons).

Free Electrons sera également présent à l’édition nord-américaine de cette conférence, qui aura lieu le mois prochain à San Francisco, et nous filmerons également les interventions. N’hésitez pas à nous faire signe si vous aussi avez la chance d’y participer !

- [en] ELCE 2009 à Grenoble (136 clics)
- [en] ELC 2010 à San Francisco (68 clics)

**Linux conferences**

**Linux conferences embedded**

**Linux conferences embedded (2011)** See Also:

http://free-electrons.com/blog/elce-2011-videos/

**Introduction à Linux embarqué (Villefontaine (38) le 15 décembre 2011** See Also:

http://free-electrons.com/company/staff/thomas-petazzoni/
http://linuxfr.org/news/s%C3%A9minaire-gratuit-linux-embarqu%C3%A9-jeudi-15-d%C3%A9cembre-%E8s-de-lyon
Depuis une dizaine d’années maintenant, le système d’exploitation Linux et les logiciels open-source offrent de nouvelles possibilités pour le développement des systèmes embarqués.

Avec un faible coût, un code source disponible et modifiable, des offres de support commercial ou communautaire, des possibilités de réutilisation importantes permettant de se focaliser sur la valeur ajoutée de son système embarqué, la solution Linux embarquée a de nombreux atouts en comparaison de systèmes propriétaires traditionnels ou de solutions spécifiques.

Ce séminaire se propose de faire le point sur les possibilités offertes par l’utilisation de Linux dans l’embarqué, en couvrant les sujets énoncés dans le programme ci-dessous.

**PROGRAMME**

14h - Introduction  Bref historique de Linux et du monde open-source.
Les avantages de Linux et de l’open-source dans l’embarqué

Matériel nécessaire pour utiliser Linux embarqué  Quelques exemples de plateformes répandues.

L’offre du monde open-source pour les systèmes embarqués

- La chaîne de compilation Les chargeurs de démarrage
- Le noyau Linux et les pilotes de périphériques
Les bases du système Les bibliothèques graphiques et multimédia
Les composants orientés réseau
Les extensions temps-réel

Le système Android, comparaison avec Linux embarqué

Le processus de développement d’un système Linux embarqué
• Les licences open-source
• Construction du système Linux
• Développement et mise au point des applications

Le support commercial et le support communautaire.

Conclusion – Débat – Questions/réponses
Des démonstrations illustrant l’utilisation pratique de Linux embarqué sur une carte ARM seront réalisées au fil de la présentation.

Linux based systems

Linux based Operating systems

Linux desktop operating systems

Fedora

See Also:
• https://secure.wikimedia.org/wikipedia/fr/wiki/Fedora_%28Linux%29
• http://www.fedora-fr.org/
• https://twitter.com/#!/fedorafr

Fedora, anciennement Fedora Core, est une distribution GNU/Linux basée sur le système RPM, développée par le Projet Fedora et soutenue par la société Red Hat.

Cette distribution se veut être un système d’exploitation complet et généraliste, composé uniquement de logiciels libres.

Fedora dérive donc de la distribution Red Hat Linux, et est destinée à la remplacer pour les utilisateurs finaux (utilisation non commerciale). Le maintien de Fedora provient en grande partie de sa communauté d’utilisateurs.

Bien que Red Hat emploie de nombreux développeurs pour Fedora, l’entreprise ne fournit pas d’assistance officielle pour les utilisateurs du grand public.

Le rythme de sortie des nouvelles versions est basé sur celui de GNOME, c’est-à-dire 6 mois.
Fedora 16  See Also:

Fedora 16 Dedicated to Dennis Ritchie  See Also:
  • c language people Dennis Ritchie
During the preparation of Fedora 16, the computing world lost one of its great contributors: Dennis Ritchie. Ritchie co-invented Unix and the C language.
He also co-authored “The C Programming Language”, a book that taught many programmers just at the time personal computing was exploding. Without Ritchie computing would be nothing like it is today.
A humble man, not well-known outside his field, Dennis will always be remembered by those of us who practice the craft. Thank you Dennis.

Ubuntu

Ubuntu development  See Also:
  • https://wiki.ubuntu.com/AppDevelopers
  • http://developer.ubuntu.com/
  • https://twitter.com/#!/ubuntuappdev
In order to go from 20 million to 200 million users, we need to get more & better apps on to Ubuntu. We want Ubuntu to have a full suite of first class applications and to be a thriving market place for free and commercial applications.
We believe a crucial element of this is to make the lives of application developers more rich and satisfying.
This page is your starting point to find out about the work that’s going on to make this happen.
It’s a bit rough and is definitely incomplete. You are welcome to join us in improving it, or to follow along as we do so.

How to join in
  • #ubuntu-app-devel on Freenode
  • ubuntu-app-devel@lists.ubuntu.com
  • Answer questions on Ask Ubuntu
  • Follow @ubuntuappdev on Twitter
  • Follow the App Developer blog on http://developer.ubuntu.com

Ubuntu 11.10

Ubuntu desktop
Ubuntu 11.10 clic lanceur


Je vous présentais il y a quelques jours la possibilité de créer lanceurs et raccourcis sur votre bureau à l’aide du terminal; je vous propose aujourd’hui un script permettant de réaliser la même opération à l’aide d’un simple clic droit sur le bureau.

C’est chez Ubuntu Up que j’ai découvert ce script que j’ai du modifier pour qu’il fonctionne sur le « Bureau » français plutôt que sur le « Desktop ».

Dans un premier temps, nous allons ouvrir un terminal afin de télécharger ce script et de le placer dans notre dossier « Scripts » à l’aide de la commande:

```
wget http://www.clapico.com/telechargement/Create-Launcher
cp Create-Launcher ~/.gnome2/nautilus-scripts
chmod +x ~/.gnome2/nautilus-scripts/Create-Launcher
mv ~/.gnome2/nautilus-scripts/Create-Launcher ~/.gnome2/nautilus-scripts/"Créer lanceur ou raccourci"
```

**Gnome classic**  Après avoir mis à jour son environnement Gnome, on veut retrouver son environnement gnome:

```
sudo aptitude install gnome-session-fallback
```

```
sudo aptitude install gnome-shell
```

ensuite, ouvrir une nouvelle session et dans l’interface de connexion, cliquer sur l’icône en forme de roue dentée à coté du login et choisir Gnome classic.

**Tails**  See Also:

- [http://tails.boum.org/about/index.fr.html](http://tails.boum.org/about/index.fr.html)

Tails est un système ‘live’ : un système d’exploitation complet destiné à être utilisé sur un CD ou une clef USB indépendamment du système installé sur l’ordinateur. C’est un logiciel libre basé sur Debian GNU/Linux.

Tails est livré avec de nombreuses applications, configurées avec une attention particulière accordée à la sécurité : navigateur internet, client de messagerie instantanée, client email, suite bureautique, éditeur d’image et de son, etc.

**Windows operating systems**

**Windows Operating systems**

**Windows operating systems**
Windows 8  See Also:


![Windows 8 Logo](image)

Figure 2.37: *Le logo de windows 8*

### Contents

- Windows 8
  - Introduction
  - Polémiques
  - Interface
  - News

### Introduction


Pour Microsoft, Windows 8 est devenu une priorité en juillet 2010.

### Polémiques

Cette nouvelle version de Windows pourrait imposer l’utilisation d’un BIOS de confiance, appelé EFI, qui est plus sécurisé, mais rend quasi impossible l’installation d’autres système d’exploitation à coté de Windows 8.

Microsoft a signalé que cette option sera facilement désactivable, mais cela dépend des OEM.

En outre Microsoft impose aux OEM que les machines ARM soient bloquées sur les systèmes d’exploitation de Microsoft pour pouvoir faire tourner Windows 8.

### Interface

Une nouvelle interface sera mise d’avant par les développeurs de Microsoft.


Le Menu Démarrer a été changé complètement en Start Screen, ou écran de démarrage, similaire à celui de Windows Phone 7.

### News

Windows 8 news

Windows 8 2012 news  See Also:


So did T.S. Eliot. And, more recently, Steve Jobs.

Let’s face it: If something makes sense and succeeds, it gets imitated.

Though Windows 8 and Linux distributions differ greatly from each other in design, ideology and – last but not least – their primary audience, they’re all built on the same basic principles of OS design so there’s bound to be some overlap.

And while Microsoft has long been accused of stealing from the open source community, according to some Linux fans, it’s getting to the point where Microsoft simply appropriates good Linux features.

Native support for USB 3.0 In their very first blog post, the Building 8 folks explained their new native USB 3.0 stack and, of course, that news was greeted with comments of the “Linux has been doing that for three years” variety.

The Microsoft twist: Move along. Nothing to see here. USB 3.0 devices work pretty well with Windows 7 already since hardware manufacturers provide their own drivers. Microsoft just finally implemented an industry

Windows 7 (6.1) See Also:


<table>
<thead>
<tr>
<th>Platform support</th>
<th>Version</th>
<th>Current version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IA-32, x86-64</td>
<td>6.1</td>
<td>6.1 (build 7600.16385.090713-1255) October 22, 2009</td>
</tr>
</tbody>
</table>

See Also:

Innosetup example (.iss file)

Win32, Win64

File System Redirector See Also:


The %windir%System32 directory is reserved for 64-bit applications. Most DLL file names were not changed when 64-bit versions of the DLLs were created, so 32-bit versions of the DLLs are stored in a different directory. WOW64 hides this difference using a file system redirector.

In most cases, whenever a 32-bit application attempts to access %windir%System32, the access is redirected to %windir%SysWOW64. Access to %windir%lastgoodsystem32 is redirected to %windir%lastgoodSysWOW64. Access to %windir%regedit.exe is redirected to %windir%SysWOW64regedit.exe.

Certain subdirectories are exempt from redirection. Access to these subdirectories is not redirected to %windir%SysWOW64:

- %windir%system32catroot
- %windir%system32catroot2
- %windir%system32driversstore
- %windir%system32driversetc
- %windir%system32logfiles
• `%windir%\system32\spool`

If the access causes the system to display the UAC prompt, redirection does not occur. Instead, the 64-bit version of the requested file is launched. To prevent this problem, either specify the SysWOW64 directory to avoid redirection and ensure access to the 32-bit version of the file, or run the 32-bit application with administrator privileges so the UAC prompt is not displayed.

**Windows Phone 7**  See Also:


Windows Phone 7 est un système d’exploitation mobile développé par Microsoft pour succéder à Windows Mobile, sa précédente plateforme logicielle qui a été renommée pour l’occasion en Windows Phone Classic.

Contrairement au système qu’il remplace, Windows Phone 7 est principalement destiné au grand public plutôt qu’au marché des entreprises.

Il a été lancé le 21 octobre 2010 en Europe, à Singapour, en Australie et en Nouvelle-Zélande, le 8 novembre 2010 aux États-Unis et au Canada, puis le 24 novembre 2010 au Mexique ; son lancement en Asie est prévu pour le courant 2011.

**Windows Vista 32 (6.0)**  See Also:

http://en.wikipedia.org/wiki/Windows_Vista

<table>
<thead>
<tr>
<th>Platform support</th>
<th>Version</th>
<th>Current version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IA-32, x86-64</td>
<td>6.0</td>
<td>(6002.18005.090410-1830) Service Pack 2 (SP2) (Build 6002) April 28, 2009</td>
</tr>
</tbody>
</table>

See Also:

*Innosetup example (.iss file)*

**Windows XP**

**Windows XP 32 bits (5.1)**  See Also:


<table>
<thead>
<tr>
<th>Platform support</th>
<th>Version</th>
<th>Current version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IA-32, x86-64, IA-64</td>
<td>5.1</td>
<td>5.1.2600.5512 Service Pack 3 (x86 SP3) 21 April 2008</td>
</tr>
</tbody>
</table>

See Also:

*Innosetup example (.iss file)*

**Windows XP 64 (5.2)**  See Also:


Windows XP Professional x64 Edition uses version 5.2.3790.1830 of core operating system binaries, the same version used by Windows Server 2003 SP1 as they were the latest versions during the operating system’s development. Even service packs and updates for Windows XP x64 and Windows Server 2003 x64 are distributed in unified packages, much in the manner as Windows 2000 Professional and Server editions for x86.

**Windows backend on XP x64**
On Thu, Apr 15, 2010 at 6:45 PM, Pete Batard <pbatard@gmail.com> wrote: > I know that there is no way to have WinUSB on that OS, > Is that true? WinUSB is available for XP 32, and we have the 64 bit > drivers for Vista and later, which I would assume are compatible with XP > 64. Is that not the case?

It is true that WinUSB is not officially supported by Microchip on Windows Server 2003 and XP64.


Just take note XP64 is not XP actually.

**It is Win 2003 in terms of the kernel**

...

http://en.wikipedia.org/wiki/Microsoft_Windows XP 64 is the same as Windows server 2003 (NT5.2). So you need to detect that as well.

I think that’s the source of the problem. The following page list Windows versions :


And as Xiaofan Chen mentionned, Windows XP x64 has version 5.2. But the libusb code looks like that.

```
windows_version = WINDOWS_UNSUPPORTED;
if ((GetVersionEx(&os_version) != 0) && (os_version.dwPlatformId == VER_PLATFORM_WIN32_NT)) {
    if ((os_version.dwMajorVersion == 5) && (os_version.dwMinorVersion == 1)) {
        windows_version = WINDOWS_XP;
    } else if (os_version.dwMajorVersion >= 6) {
        windows_version = WINDOWS_VISTA_AND_LATER;
    }
}
```

The third line should be changed to

```
if ((os_version.dwMajorVersion == 5) && (os_version.dwMinorVersion >= 1)) {
```

That should fix the issue for both XP x64 and Server 2003 versions of Windows.

**Windows server 2003 (5.2)** See Also:

Windows 2000 (5.0)  See Also:


Windows 2000 is a line of operating systems produced by Microsoft for use on business desktops, notebook computers, and servers. Released on February 17, 2000, it was the successor to Windows NT 4.0, and is the final release of Microsoft Windows to display the “Windows NT” designation. It was succeeded by Windows XP for desktop systems in October 2001 and Windows Server 2003 for servers in April 2003.

<table>
<thead>
<tr>
<th>Platform support</th>
<th>Version</th>
<th>Current version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IA-32, Itanium</td>
<td>5.0</td>
<td>5.0 SP4 Rollup 1 v2 (5.0.3700.6690) September 13, 2005</td>
</tr>
</tbody>
</table>

Windows tools  See Also:

Python cli management

Devcon based tools

devcon (nt)  L’utilitaire DevCon est un utilitaire de ligne de commande qui peut être utilisé à la place du Gestionnaire de périphériques. Il vous permet d’activer, de désactiver, de redémarrer, de mettre à jour, de supprimer et d’interroger des périphériques individuels ou des groupes de périphériques.

Il fournit également des informations utiles aux développeurs de pilotes, mais non disponibles dans le Gestionnaire de périphériques.

See Also:

• http://support.microsoft.com/kb/311272

Usage

Device Console Help:

devcon [-r] [-m:\\<machine>] <command> [arg]...
-r if specified will reboot machine after command is complete, if needed.
-m\<machine> is name of target machine.
<command> is command to perform (see below).
<arg>... is one or more arguments if required by command.

For help on a specific command, type: devcon help <command>
classfilter Allows modification of class filters.
classes List all device setup classes.
disable Disable devices that match the specific hardware or instance ID.
driverfiles List driver files installed for devices.
drivenodes Lists all the driver nodes of devices.
enable Enable devices that match the specific hardware or instance ID.
find Find devices that match the specific hardware or instance ID.
findall Find devices including those that are not present.
help Display this information.
hwids Lists hardware ID’s of devices.
install Manually install a device.
listclass List all devices for a setup class.
reboot Reboot local machine.
remove Remove devices that match the specific hardware or instance ID.
rescan Scan for new hardware.
resources Lists hardware resources of devices.
restart Restart devices that match the specific hardware or instance ID.
sethwid Modify Hardware ID’s of listed root-enumerated devices.
stack Lists expected driver stack of devices.
status List running status of devices.
update Manually update a device.
updateni Manually update a device (non interactive).

Exemples de commandes DevCon

devcon find usbvid*

C:\Tmp>devcon find usb\vid\*

USB\VID_0B81&PID_0200\5&344BCE15&0&1 : USB Smart Card reader

Trouve tous les périphériques USB commençant par ‘vid’

devcon -m:\test findpci*

devcon -m:\\test findpci\\*

Dresse la liste de tous les périphériques PCI connus présents sur l’ordinateur test (l’argument -m permet de spécifier un ordinateur cible ; vous devez utiliser la communication entre processus (IPC) pour pouvoir accéder à cet ordinateur).

devcon -r install %WINDIR%\Inf\Netloop.inf MSLOOP

devcon -r install %WINDIR%\Inf\Netloop.inf *MSLOOP

Installe une nouvelle instance de la carte de bouclage Microsoft. Cette commande crée un nouveau nœud pour les périphériques énumérés à la racine, avec lequel vous pouvez installer un « périphérique virtuel », tel qu’une carte de bouclage. Elle redémarre également l’ordinateur en mode silencieux, si nécessaire.

devcon classes

devcon classes

Dresse la liste de toutes les classes de configuration connues. La sortie contient le nom court non traduit (« USB », par exemple) et le nom descriptif (« contrôleurs Universal Serial Bus », par exemple).

devcon classfilter upper !filter1 !filter2

devcon classfilter upper !filter1 !filter2

Supprime les deux filtres spécifiés.

devcon classfilter lower !badfilter +goodfilter

devcon classfilter lower !badfilter +goodfilter

Remplacez « badfilter » par « goodfilter ». 
devcon driverfiles =ports

devcon driverfiles =ports

Dresse la liste des fichiers associés à chaque périphérique de la classe de configuration ports.

**devcon disable MSLOOP**  Désactive tous les périphériques dont l’ID de matériel se termine par « MSLOOP » (y compris “*MSLOOP”).

**devcon drivernodes @ROOT\PCI_HAL\PNP0A03**

devcon drivernodes @ROOT\PCI_HAL\PNP0A03

Dresse la liste de tous les pilotes compatibles avec le périphérique ROOTPCI_HALPNP0A03. Cette commande peut servir à déterminer la raison pour laquelle un fichier .inf (Integral Device Information) a été préféré à un fichier .inf tiers.

**devcon enable MSLOOP**

devcon enable \*MSLOOP

Active tous les périphériques dont l’ID de matériel est “*MSLOOP”. Le guillemet simple indique qu’il faut considérer l’ID de matériel comme une expression littérale (en d’autres termes, l’astérisque [« * »] représente bien un astérisque ; il ne s’agit pas d’un caractère générique).

**devcon find ***

devcon find *

Dresse la liste de toutes les instances des périphériques présents sur l’ordinateur local.

**devcon find pci**

devcon find pci\*

Dresse la liste de tous les périphériques PCI (Peripheral Component Interconnect) connus présents sur l’ordinateur local (cette commande suppose qu’un périphérique est de type PCI si son ID de matériel commence par « PCI »).

**devcon find =ports pnp**

devcon find =ports *pnp*

Dresse la liste des périphériques présents appartenant à la classe de configuration ports et dont l’ID de matériel contient « PNP ».
devcon find =ports @root*

devcon find =ports @root\*

Dresse la liste des périphériques présents appartenant à la classe de configuration ports et figurant à la racine de l’arborescence d’énumération (l’ID d’instance commence par « root»). Ne faites aucune supposition quant à la façon dont les ID d’instance sont mises en forme dans le code. Pour déterminer les périphériques à la racine, vous pouvez examiner les bits d’état des périphériques. Cette fonction est incluse dans DevCon pour faciliter le débogage.

devcon findall =ports

devcon findall =ports

Dresse la liste de tous les périphériques absents et présents appartenant à la classe ports. Sont inclus les périphériques supprimés, les périphériques transférés dans un autre emplacement et, dans certains cas, les périphériques énumérés de façon différente en raison d’une modification du BIOS.

devcon listclass usb 1394

devcon listclass usb 1394

Dresse la liste de tous les périphériques présents pour chaque classe indiquée (dans cet exemple, USB et 1394).

devcon remove @usb*

devcon remove @usb\*

Supprime tous les périphériques USB. Les périphériques supprimés sont accompagnés de leur état de suppression.

devcon rescan

devcon rescan

Effectue une nouvelle recherche sur les périphériques Plug and Play.

devcon resources =ports

devcon resources =ports

Dresse la liste de toutes les ressources utilisées par tous les périphériques de la classe de configuration ports.

devcon restart =net @’ROOT*MSLOOP0000

devcon restart =net @’ROOT\*MSLOOP\0000

Redémarre la carte de bouclage ROOT*MSLOOP0000. Le guillemet simple dans la commande indique qu’il faut considérer l’ID de matériel comme une expression littérale.
devcon hwids=mouse
devcon hwids=mouse

Dresse la liste de tous les ID du matériel des périphériques de classe souris sur le système.

devcon sethwid @ROOT\LEGACY_BEEP\0000 := beep
devcon sethwid @ROOT\\LEGACY_BEEP\0000 := beep

Attribue l’ID de matériel, beep, au périphérique hérité beep.

devcon stack =ports
devcon stack =ports

Indique la pile de pilotes attendue pour le périphérique. Elle inclut les filtres inférieurs/supérieurs de classes et de périphériques, ainsi que le service de contrôle.

devcon status @pci*
devcon status @pci\*

Indique l’état de chaque périphérique présent dont l’ID d’instance commence par « pci».

devcon status @ACPI\PNP05011
devcon status @ACPI\PNP0501\1

Indique l’état d’une instance de périphérique particulière ; dans cet exemple, il s’agit d’un port série énuméré par l’interface ACPI (Advanced Configuration and Power Interface).

devcon status @root\rdp_mou0000
devcon status @root\rdp_mou\0000

Indique l’état du pilote de souris de Microsoft Terminal Server ou des services Terminal Server.

devcon status PNP05
devcon status *PNP05*

Indique l’état de tous les ports COM.
devcon update mydev.inf pnp0501

Met à jour tous les périphériques correspondant exactement à l’ID de matériel pnp0501 pour qu’ils utilisent le meilleur pilote possible dans Mydev.inf associé à l’ID de matériel pnp0501.

Note: Cette mise à jour force tous les périphériques à utiliser le pilote figurant dans Mydev.inf, même s’il existe déjà un meilleur choix sur le système.

Cela peut être utile lorsque vous souhaitez installer de nouvelles versions de pilotes au cours d’une phase de développement, avant d’avoir obtenu une signature. La mise à jour ne concerne que les périphériques correspondant à l’ID de matériel spécifié, mais pas les périphériques enfants. Si le fichier .inf spécifié ne comporte pas de signature, il se peut que Windows affiche une boîte de dialogue vous invitant à confirmer s’il faut installer le pilote. S’il est nécessaire de redémarrer l’ordinateur, vous en êtes averti et DevCon renvoie une erreur de niveau 1. Si vous spécifiez l’argument -r, le redémarrage se fait automatiquement, le cas échéant.

Microsoft CCID driver manager

CCID driver manager (id3 Semiconductors)  See Also:

• devcon (mt)
• Windows usbccid driver

This is a proprietary software from id3 Semiconductors.
Performs EscapeCommandEnable management, registry keys cleanup and smartcard reader restart.

Usage

CcidDrvMgr.exe -help -pause

*****************************************************************************
*** Microsoft CCID Driver Manager ***
*** 1.0.1811 ***
*** @2010 id3 Semiconductors ***
*****************************************************************************

<> Displays CCID Driver Escape Commands status <>
Syntax : CcidDrvMgr -status

<> Enables CCID Driver to pass Escape Commands to reader <>
Syntax : CcidDrvMgr -enaec -pid=PID -reenum
Development tools, Release 2012.06.18

where params are:
PID : the USB Product ID to filter.
   Optional : if omitted, Escape commands are enabled for all products IDs
restart : performs an USB re-enumeration of smartcard readers to force drive r to read new registry settings
<> Disables CCID Driver to pass Escape Commands to reader <>
Syntax : CcidDrvMgr -disec -pid=PID -reenum
where params are:
PID : the USB Product ID to filter.
   Optional : if omitted, Escape commands are enabled for all products IDs
restart : performs an USB re-enumeration of smartcard readers to force driver to read new registry settings
<> Removes Readers <>
Syntax : CcidDrvMgr -remove -pid=PID
where params are:
PID : the USB Product ID for which operation is performed.
   Optional : if omitted, all readers are removed
restart : performs an USB re-enumeration to force new device enumeration
<> Others switches common to all modes <>
-help : displays this text
-quiet : no information are displayed. Ignored if present with -dispkeys switch
-pause : Keyboard key must be pressed before the manager can exit. Ignore if quiet mode

When you plug a smartcard reader you have to enable the proprietary command.

CDM_Status.bat

CcidDrvMgr.exe -pause

Enable the proprietary commands  To enable the CCID proprietary commands:

2.22. Operating systems  857
x64 CcidDrvMgr version  x64 CcidDrvMgr.exe version

x86 CcidDrvMgr version  x86 CcidDrvMgr.exe version

Manifest tool (mt)  The Mt.exe file is a tool that generates signed files and catalogs. It is available in the Microsoft Windows Software Development Kit (SDK). Mt.exe requires that the file referenced in the manifest be present in the same directory as the manifest.

Mt.exe generates hashes using the CryptoAPI implementation of the Secure Hash Algorithm (SHA-1). For more information about hash algorithms, see Hash and Signature Algorithms. Hashes are inserted as a hexadecimal string into the file tags in the manifest.

The tool currently only generates SHA-1 hashes, although files in manifests may use other hashing schemes.

Mt.exe uses Makecat.exe to generate catalog files (.cat) from catalog definition files (.cdf). This tool fills out a standard template CDF with the name and location of your manifest. You can use this with Makecat.exe to generate the assembly catalog.

The version of Mt.exe provided in recent versions of the Windows SDK can also be used to generate manifests for managed assemblies and unmanaged side-by-side assemblies.
Usage

mt.exe
[ -manifest <manifest1 name> <manifest2 name> ... ]
[ -identity:<identity string> ]
[ -managedassemblyname:<managed assembly> [ -nodependency ] [ -category ] ]
[ -out:<output manifest name> ]
[ -inputresource:<file>[;[#]<resource_id>] ]
[ -outputresource:<file>[;[#]<resource_id>] ]
[ -updateresource:<file>[;[#]<resource_id>] ]
[ -hashupdate:[<path to the files>] ]
[ -makecdfs ]
[ -validate_manifest ]
[ -validate_file_hashes:<path to the files> ]
[ -canonicalize ]
[ -check_for_duplicates ]
[ -nologo ]

Options

• manifest Used to specify manifests that need to be processed. At least one manifest name should follow this option. NOTE: There is no colon(;) after -manifest:

<manifest1 name> <manifest2 name> ...
Names of manifests to be processed and/or merged.
Required if the -manifest option is used.
NOTE: More than one manifest automatically indicates a manifest "merge" operation. In that case, an output specified by one of -out / -outputresource / -updateresource is mandatory.

• identity:<identity string> The identity string contains the attributes of the assemblyIdentity element. The identity string is a set of comma separated name=value pairs starting with the “name” attribute’s value. e.g.: “Microsoft.Windows.Common-Controls, processorArchitecture=x86, version=6.0.0.0, type=win32, publicKeyToken=6595b64144ccf1df”. NOTE: Only the “name” attribute is not of the form “name=value” and it should be the first attribute in the identity string.

• replacements:<.XML filename> Specifies the file that contains values for replaceable strings in the RGS file.

• managedassemblyname:<managed assembly> [ -nodependency ] [ -category ] Generates a manifest from a managed assembly.

• nodependency suppresses the generation of dependency elements in the final manifest.

See Also:
• http://msdn.microsoft.com/en-us/library/ms235591%28VS.80%29.aspx (How to: Embed a Manifest Inside a C/C++ Application)

Microsoft (R) Manifest Tool version 5.2.3790.2075 Copyright (c) Microsoft Corporation 2005. All rights reserved.

2.22. Operating systems 859
• category causes the category tags to be generated.

• out:<Output manifest name> Name of the output manifest. If this is skipped and only one manifest is being operated upon by the tool, that manifest is modified in place.

• inputresource:<file>[:[#]<resource_id>] Input the manifest from a resource of type RT_MANIFEST with the specified id. resource_id is restricted to be a non-negative, 16 bit number. resource_id is optional and defaults to CREATEPROCESS_MANIFEST_RESOURCE_ID (winuser.h).

• outputresource:<file>[:[#]<resource_id>] Output the manifest to a resource of type RT_MANIFEST with the specified id. resource_id is restricted to be a non-negative, 16 bit number. resource_id is optional and defaults to CREATEPROCESS_MANIFEST_RESOURCE_ID (winuser.h).

• updateresource:<file>[:[#]<resource_id>] Equivalent to specifying both -inputresource and

• outputresource with identical parameters. resource_id is restricted to be a non-negative, 16 bit number.

• hashupdate:<path to the files> Computes the hash of files specified in the file elements and updates the hash attribute with this value. The searchpath for the actual files specified in the file elements is specified explicitly. If <path to the files> is not specified, the searchpath defaults to the location of the output manifest.

• makecdfs Generates Catalog Definition Files (.cdf) - used to make catalogs.

• validate_manifest Validates to check syntactic correctness of a manifest and its conformance to the manifest schema.

• validate_file_hashes:<path to the files> Validates the hash values of all the file elements.

• canonicalize Does a C14N canonicalization of the output manifest contents.

• check_for_duplicates Performs a check to see if the final manifest contains duplicate elements.

• nologo Runs without displaying standard Microsoft copyright data. This may be used to suppress unwanted output in log files when running mt.exe as part of a build process or from a build environment.

Samples   To update the hash of an XML manifest:

mt.exe -manifest 1.manifest -hashupdate -out:updated.manifest

To update the hash of an XML manifest while simultaneously producing the .cdf file:

mt.exe -manifest 1.manifest -hashupdate -makecdfs -out:updated.manifest

To merge two manifests:

mt.exe -manifest 1.manifest 2.manifest -out:merged.manifest

To merge two manifests and finally update the hash to produce the final merged manifest.

Note:   The searchpath for the actual files specified in the file elements is specified explicitly.

mt.exe -manifest 1.manifest 2.manifest -hashupdate:d:\filerpository -out:merged.manifest

To generate a manifest from an RGS and/or TLB file:

To generate an XML manifest from a managed assembly:

```
mt.exe -managedassemblyname:managed.dll -out:out.manifest
```

To suppress dependencies:

```
mt.exe -managedassemblyname:managed.dll -nodependency -out:out.manifest
```

To generate `<category>` elements:

```
mt.exe -managedassemblyname:managed.dll -category -out:out.manifest
```

To extract manifest out of a dll:

```
mt.exe -inputresource:dll_with_manifest.dll;#1 -out:extracted.manifest
```

To merge two manifests, one of them embedded in a dll, and embedding final merged manifest into another dll’s resource:

```
mt.exe -inputresource:dll_with_manifest.dll;#1 -manifest 2.manifest -outputresource:dll_with_merged_manifest.dll;#3
```

To update the manifest in a PE’s resource (by updating the hashes of the file elements):

```
mt.exe -updateresource:dll_with_manifest.dll;#1 -hashupdate:f:\files
```

To validate the hash values of all the file elements:

```
mt.exe -manifest 1.manifest -validate_file_hashes:"c:\files"
```

To validate a manifest (i.e., to see if it conforms to the manifest schema):

```
mt.exe -manifest 1.manifest -validate_manifest
```

To do a C14N canonicalization of a manifest (in order to get rid of spurious namespace prefixes (like “dsig”)):

```
mt.exe -manifest 1.manifest -canonicalize
```

To check for duplicate elements in a manifest:

```
mt.exe -manifest 1.manifest -check_for_duplicates
```

CL1356 sample

```
1.manifest Le panneau de configuration peut ajouter la clé de registre ‘EscapeCommandEnable’ comme il le fait sous XP, à condition qu’il soit lancé en tant qu’administrateur. Pour se faire, il faut incorporer un manifest à l’exécutable. L’incorporation du manifest est réalisée par l’outil mt.exe:

mt.exe -manifest MyApp.exe.manifest -outputresource:MyApp.exe;1
```

2.22. Operating systems
msysgit on windows  See Also:
https://code.google.com/p/msysgit/downloads/list

Path Editor  See Also:
http://www.redfernplace.com/software-projects/patheditor/

Introduction  If you have ever had need to alter, manage, or otherwise interact with the System or User Path on your PC then you know how painful it is do with the built in tools.

Path Editor is a small utility that makes path management very straightforward with it’s intuitive user interface and drag-and-drop simplicity. Path Editor can clean your path of missing and duplicate entries with a single click of the mouse.

proxycfg tool  See Also:
• http://www.processlibrary.com/fr/directory/files/proxycfg/28503/

This topic explains the use of the Microsoft Windows HTTP Services (WinHTTP) proxy configuration tool, “ProxyCfg.exe”.

There are two ways to access HTTP and Secure Hypertext Transfer Protocol (HTTPS) servers through a proxy using Microsoft Windows HTTP Services (WinHTTP).

First, you can specify proxy settings from within your WinHTTP application.

Second, you can specify default proxy settings from outside your application using the proxy configuration utility located in the %windir%\system32 directory.

C:\>proxycfg  -u

Outil de configuration du proxy par défaut WinHTTP Microsoft (R)
Copyright (c) Microsoft Corporation. Tous droits réservés.

Paramètres proxy mis à jour
Paramètres proxy WinHTTP en cours sous :
  HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\InternetSetting \Connections\WinHttpSettings :
  Serveurs proxy : http=gateway:3128
  Liste d’exception : <local>
**Rapid Environment Editor**  See Also:

http://www.rapidee.com/en/about

Rapid Environment Editor (RapidEE) is an environment variables editor. It includes easy to use GUI and replaces small and inconvenient Windows edit box. RapidEE 6.x supports Windows NT, 2000, 2003, XP, Vista, 2008 & Windows 7 (including 64 bit versions).

If you still use Windows 9x or ME, then use version 2.1.

**Snarl**  See Also:

- http://sourceforge.net/projects/snarlwin/files/Snarl/

![Figure 2.38: Snarl logo](image)

**What is Snarl?**  Snarl is an application that allows other applications to alert the user to an event in a structured, unified way through the creation of on-screen notifications.

Ultimately, the end-user has final say over which events an application notifies, and how they are presented based on the user’s presence and whether they want to be disturbed or not.

**Getting Started**

- The Snarl User Guide is a good place to start...
- The Rationale page explains more about the Snarl ethos
- Documentation for the bundled Snarl Apps

**General Topics**  New Features Developer Guide

**APIs**

- Generic API reference
- Windows API specifics
- Snarl Network Protocol
- SNP 2.0 API specifics
Development tools, Release 2012.06.18

• SNP 3.0 API (Beta)
• libSnarl reference

SDKs
• WebForward SDK

Standards
• Notification Appearance and Layout Guidelines
• Creating dedicated Snarl Extensions
• GNTP Support (details coming soon)

More
• Read about Ongoing Development
• Forthcoming Centralised Snarl Administration

WMI (Windows Management Instrumentation)  See Also:
• Python wmi
• windows_powershell
• http://en.wikipedia.org/wiki/Windows_Management_Instrumentation
• http://quux.wiki.zoho.com/PowerShell-Snippets.html
• http://quux.wiki.zoho.com/WMIC-Snippets.html (old page before powershell)
• http://support.microsoft.com/servicedeskss/webcasts/wc072402/listofsampleusage.asp
• http://laurent-dardenne.developpez.com/articles/wmi-p1/

Contents
• WMI (Windows Management Instrumentation)
  – Introduction
  – List of Win32 classes
    * Win32_Desktop
    * win32_serialport
    * Win32_PnP_SignedDriver
    * Win32_Processor
    * Win32_OperatingSystem
    * Win32_ComputerSystem

Introduction  Windows Management Instrumentation (WMI) is a set of extensions to the Windows Driver Model that provides an operating system interface through which instrumented components provide information and notification. WMI is Microsoft’s implementation of the Web-Based Enterprise Management (WBEM) and Common Information Model (CIM) standards from the Distributed Management Task Force (DMTF).

WMI allows scripting languages like VBScript or Windows PowerShell to manage Microsoft Windows personal computers and servers, both locally and remotely. WMI is preinstalled in Windows 2000 and newer OSs.
It is available as a download for Windows NT, Windows 95 and Windows 98.

Microsoft also provides a command line interface to WMI called Windows Management Instrumentation Command-line (WMIC)

**List of Win32 classes**  
See Also:


To display the classe’s informations type the following command:

```bash
wmic path Win32_OperatingSystem get /ALL /FORMAT:list
```

```bash
wmic path Win32_PnPEntity get /?
```

<table>
<thead>
<tr>
<th>Propriété</th>
<th>Type</th>
<th>Opération</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>Caption</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ClassGuid</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ConfigManagerErrorCode</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>ConfigManagerUserConfig</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>CreationClassName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Description</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>DeviceID</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ErrorCleared</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>ErrorDescription</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>InstallDate</td>
<td>datetime</td>
<td>Read</td>
</tr>
<tr>
<td>LastErrorCode</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Name</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>PNPDeviceID</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>PowerManagementCapabilities</td>
<td>array of uint16</td>
<td>Read</td>
</tr>
<tr>
<td>PowerManagementSupported</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>Service</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Status</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>StatusInfo</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>SystemCreationClassName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>SystemName</td>
<td>string</td>
<td>Read</td>
</tr>
</tbody>
</table>

**Win32_Desktop**

```bash
wmic path Win32_Desktop get /?
```

<table>
<thead>
<tr>
<th>Propriété</th>
<th>Type</th>
<th>Opération</th>
</tr>
</thead>
<tbody>
<tr>
<td>BorderWidth</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>Caption</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>CoolSwitch</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>CursorBlinkRate</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>Description</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>DragFullWindows</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>GridGranularity</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>IconSpacing</td>
<td>uint32</td>
<td>Read</td>
</tr>
</tbody>
</table>

2.22. Operating systems
IconTitleFaceName | string | Read
IconTitleSize   | uint32 | Read
IconTitleWrap   | boolean| Read
Name            | string | Read
Pattern         | string | Read
ScreenSaverActive | boolean | Read
ScreenSaverExecutable | string | Read
ScreenSaverSecure | boolean | Read
ScreenSaverTimeout | uint32 | Read
SettingID       | string | Read
Wallpaper       | string | Read
WallpaperStretched | boolean | Read
WallpaperTiled  | boolean | Read

**win32_serialport**

wmic path win32_serialport get /?

<table>
<thead>
<tr>
<th>Propriété</th>
<th>Type</th>
<th>Opération</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>Binary</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>Capabilities</td>
<td>array of uint16</td>
<td>Read</td>
</tr>
<tr>
<td>CapabilityDescriptions</td>
<td>array of string</td>
<td>Read</td>
</tr>
<tr>
<td>Caption</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ConfigManagerErrorCode</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>ConfigManagerUserConfig</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>CreationClassName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Description</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>DeviceID</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ErrorCleared</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>ErrorDescription</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>InstallDate</td>
<td>datetime</td>
<td>Read</td>
</tr>
<tr>
<td>LastErrorCode</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>MaxBaudRate</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>MaxNumberControlled</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>MaximumInputBufferSize</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>MaximumOutputBufferSize</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>Name</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>OSAutoDiscovered</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>PNPDeviceID</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>PowerManagementCapabilities</td>
<td>array of uint16</td>
<td>Read</td>
</tr>
<tr>
<td>PowerManagementSupported</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>ProtocolSupported</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>ProviderType</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>SettableBaudRate</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SettableDataBits</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SettableFlowControl</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SettableParity</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SettableParityCheck</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SettableRLSD</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SettableStopBits</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>Status</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>StatusInfo</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>Supports16BitMode</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>SupportsDTRDSR</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>Propriété</td>
<td>Type</td>
<td>Opération</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------</td>
<td>-----------</td>
</tr>
<tr>
<td>Caption</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ClassGuid</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>CompatID</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>CreationClassName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Description</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DevLoader</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DeviceClass</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DeviceID</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DeviceName</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DriverDate</td>
<td>datetime</td>
<td>Read</td>
</tr>
<tr>
<td>DriverName</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DriverProviderName</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>DriverVersion</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>FriendlyName</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>HardWareID</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>InfName</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>InstallDate</td>
<td>datetime</td>
<td>Read</td>
</tr>
<tr>
<td>IsSigned</td>
<td>Boolean</td>
<td>N/A</td>
</tr>
<tr>
<td>Location</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>Name</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>PDO</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>Signer</td>
<td>String</td>
<td>Read</td>
</tr>
<tr>
<td>StartMode</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Started</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>Status</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>SystemCreationClassName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>SystemName</td>
<td>string</td>
<td>Read</td>
</tr>
</tbody>
</table>

**Win32_PnPSignedDriver**

wmic path Win32_PnPSignedDriver get /?

<table>
<thead>
<tr>
<th>Propriété</th>
<th>Type</th>
<th>Opération</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddressWidth</td>
<td>uint16</td>
<td>Read</td>
</tr>
</tbody>
</table>
Architecture uint16 Read
Availability uint16 Read
Caption string Read
ConfigManagerErrorCode uint32 Read
ConfigManagerUserConfig boolean Read
CpuStatus uint16 Read
CreationClassName string Read
CurrentClockSpeed uint32 Read
CurrentVoltage uint16 Read
DataWidth uint16 Read
Description string Read
DeviceID string Read
ErrorCleared boolean Read
ErrorDescription string Read
ExtClock uint32 Read
Family uint16 Read
InstallDate datetime Read
L2CacheSize uint32 Read
L2CacheSpeed uint32 Read
LastErrorCode uint32 Read
Level uint16 Read
LoadPercentage uint16 Read
Manufacturer string Read
MaxClockSpeed uint32 Read
Name string Read
NumberOfCores uint32 Read
NumberOfLogicalProcessors uint32 Read
OtherFamilyDescription string Read
PNPDeviceID string Read
PowerManagementCapabilities array of uint16 Read
PowerManagementSupported boolean Read
ProcessorId string Read
ProcessorType uint16 Read
Revision uint16 Read
Role string Read
SocketDesignation string Read
Status string Read
StatusInfo uint16 Read
Stepping string Read
SystemCreationClassName string Read
SystemName string Read
UniqueID string Read
UpgradeMethod uint16 Read
Version string Read
VoltageCaps uint32 Read

Win32_OperatingSystem

wmic path Win32_OperatingSystem get /?

Propriété Type Opération
======== ==== =========
BootDevice string Read
BuildNumber string Read
BuildType string Read
CSCreationClassName string Read
CSDVersion  string  Read
CSName  string  Read
Caption  string  Read
CodeSet  string  Read
CountryCode  string  Read
CreationClassName  string  Read
CurrentTimeZone  sint16  Read/Write
DataExecutionPrevention_32BitApplications  boolean  Read
DataExecutionPrevention_Available  boolean  Read
DataExecutionPrevention_Drivers  boolean  Read
DataExecutionPrevention_SupportPolicy  uint8  Read
Debug  boolean  Read
Description  string  Read/Write
Distributed  boolean  Read
EncryptionLevel  uint32  Read
ForegroundApplicationBoost  uint8  Read/Write
FreePhysicalMemory  uint64  Read
FreeSpaceInPagingFiles  uint64  Read
FreeVirtualMemory  uint64  Read
InstallDate  datetime  Read
LargeSystemCache  uint32  Read
LastBootUpTime  datetime  Read
LocalDateTime  datetime  Read
Locale  string  Read
Manufacturer  string  Read
MaxNumberOfProcesses  uint32  Read
MaxProcessMemorySize  uint64  Read
Name  string  Read
NumberOfLicensedUsers  uint32  Read
NumberOfProcesses  uint32  Read
NumberOfUsers  uint32  Read
OSLanguage  uint32  Read
OSProductSuite  uint32  Read
OSType  uint16  Read
Organization  string  Read
OtherTypeDescription  string  Read
PlusProductID  string  Read
PlusVersionNumber  string  Read
Primary  boolean  Read
ProductType  uint32  Read
QuantumLength  uint8  Read/Write
QuantumType  uint8  Read/Write
RegisteredUser  string  Read
SerialNumber  string  Read
ServicePackMajorVersion  uint16  Read
ServicePackMinorVersion  uint16  Read
SizeStoredInPagingFiles  uint64  Read
Status  string  Read
SuiteMask  uint32  Read
SystemDevice  string  Read
SystemDirectory  string  Read
SystemDrive  string  Read
TotalSwapSpaceSize  uint64  Read
TotalVirtualMemorySize  uint64  Read
TotalVisibleMemorySize  uint64  Read
Version  string  Read
WindowsDirectory  string  Read

2.22. Operating systems
Win32_ComputerSystem

wmic path Win32_ComputerSystem get /?

<table>
<thead>
<tr>
<th>Propriété</th>
<th>Type</th>
<th>Opération</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdminPasswordStatus</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>AutomaticResetBootOption</td>
<td>boolean</td>
<td>Read/Write</td>
</tr>
<tr>
<td>AutomaticResetCapability</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>BootOptionOnLimit</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>BootOptionOnWatchDog</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>BootROMSupported</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>BootupState</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Caption</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ChassisBootupState</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>CreationClassName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>CurrentTimeZone</td>
<td>sint16</td>
<td>Read/Write</td>
</tr>
<tr>
<td>DaylightInEffect</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>Description</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Domain</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>DomainRole</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>EnableDaylightSavingsTime</td>
<td>boolean</td>
<td>Write</td>
</tr>
<tr>
<td>FrontPanelResetStatus</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>InfraredSupported</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>InitialLoadInfo</td>
<td>array of string</td>
<td>Read</td>
</tr>
<tr>
<td>InstallDate</td>
<td>datetime</td>
<td>Read</td>
</tr>
<tr>
<td>KeyboardPasswordStatus</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>LastLoadInfo</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Model</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>Name</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>NameFormat</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>NetworkServerModeEnabled</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>NumberOfLogicalProcessors</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>NumberOfProcessors</td>
<td>uint32</td>
<td>Read</td>
</tr>
<tr>
<td>OEMLogoBitmap</td>
<td>array of uint8</td>
<td>Read</td>
</tr>
<tr>
<td>OEMStringArray</td>
<td>array of string</td>
<td>Read</td>
</tr>
<tr>
<td>PartOfDomain</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>PauseAfterReset</td>
<td>sint64</td>
<td>Read</td>
</tr>
<tr>
<td>PowerManagementCapabilities</td>
<td>array of uint16</td>
<td>Read</td>
</tr>
<tr>
<td>PowerManagementSupported</td>
<td>boolean</td>
<td>Read</td>
</tr>
<tr>
<td>PowerOnPasswordStatus</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>PowerSupplyState</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>PrimaryOwnerContact</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>PrimaryOwnerName</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ResetCapability</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>ResetCount</td>
<td>sint16</td>
<td>Read</td>
</tr>
<tr>
<td>ResetLimit</td>
<td>sint16</td>
<td>Read</td>
</tr>
<tr>
<td>Roles</td>
<td>array of string</td>
<td>Read/Write</td>
</tr>
<tr>
<td>Status</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>SupportContactDescription</td>
<td>array of string</td>
<td>Read</td>
</tr>
<tr>
<td>SystemStartupDelay</td>
<td>uint16</td>
<td>Read/Write</td>
</tr>
<tr>
<td>SystemStartupOptions</td>
<td>array of string</td>
<td>Read/Write</td>
</tr>
<tr>
<td>SystemStartupSetting</td>
<td>uint8</td>
<td>Read/Write</td>
</tr>
<tr>
<td>SystemType</td>
<td>string</td>
<td>Read</td>
</tr>
<tr>
<td>ThermalState</td>
<td>uint16</td>
<td>Read</td>
</tr>
<tr>
<td>TotalPhysicalMemory</td>
<td>uint64</td>
<td>Read</td>
</tr>
</tbody>
</table>
Use WMI in C/C++  
See Also:


Contents

- Use WMI in C/C++
  - Using Disphelper in a Qt project with MINGW
    - The windows source files
    - Link with WMI (lole32 -loleaut32 -luuid)

Using Disphelper in a Qt project with MINGW  
We have to link with the

The windows source files

```
win32 {
 SOURCES += \n   ../disphelper/samples_cpp/wmi_test_drive.cpp \n   ../disphelper/source/dh_invoke.c \n   ../disphelper/source/dh_init.c \n   ../disphelper/source/dh_extra.c \n   ../disphelper/source/dh_exceptions.c \n   ../disphelper/source/dh_enum.c \n   ../disphelper/source/dh_create.c \n   ../disphelper/source/dh_core.c \n   ../disphelper/source/convert.c

 HEADERS += \n   ../disphelper/source/convert.h \n   ../disphelper/source/disphelper.h
}
```

Link with WMI (-lole32 -loleaut32 -luuid)

```
win32 {
 DEFINES += WIN32
 DEFINES += _WIN32_WINNT=0x0501
 INCLUDEPATH += "C:/boost_1_47_0"
 # see C:/boost_1_47_0/stage/lib
 LIBS += -LZ:/PDEV1V160_CodesRousseau/Soft/PC/test_boost/lib/windows
 #static
 # see https://svn.boost.org/trac/boost/ticket/4614
 #  DEFINES += BOOST_THREAD_USE_LIB # pour l’édition de liens statique
 #  LIBS += -static -lpthread # for static linking
 #  LIBS += libboost_system-mgw44-mt-s-l_47
```
# LIBS += libboost_thread-mgw44-mt-s-1_47
#dynamic
LIBS += libboost_system-mgw44-mt-1_47
LIBS += libboost_thread-mgw44-mt-1_47 # pour l’exemple 3_async
LIBS += -lws2_32 # WSAStartUp@8  WSACleanup@0
LIBS += -lole32 -loleaut32 -luuid #WMI

# Future use
# INCLUDEPATH += "C:/zeromq-2.1.9/include"
# LIBS += -LC:/zeromq-2.1.9/lib libzmq

## MacOSX operating system

### Mac OS/X Operating system

### See Also:

http://fr.wikipedia.org/wiki/Macosx

Mac OS X est une ligne de systèmes d’exploitation propriétaires développés et commercialisés par Apple, dont la version la plus récente (Mac OS X v10.6, dit Snow Leopard, depuis 28 août 2009) est installée sur tous les Macintosh vendus actuellement.

### Versions

### Mac OS/X versions

**Mac OS/X 10.8 Mountain Lion**  See Also:

- [http://techland.time.com/2012/02/16/apples-os-x-10-8-mountain-lion-the-mac-gets-even-more-ipad-like/](http://techland.time.com/2012/02/16/apples-os-x-10-8-mountain-lion-the-mac-gets-even-more-ipad-like/)

## bootloaders

### bootloaders

Un chargeur d’amorçage (ou bootloader) est un logiciel permettant de lancer un ou plusieurs systèmes d’exploitation (multi-boot), c’est-à-dire qu’il permet d’utiliser plusieurs systèmes, à des moments différents, sur la même machine.

**Linux embedded bootloaders**  The role of the bootloader is to initialize some basic hardware peripherals, load the Linux kernel image and run it.

The boot process of most recent embedded processors is the following:

1. The processor **executes code in ROM**, to load a first-stage bootloader from NAND, SPI flash, serial port or SD card
2. The **first stage bootloader** initializes the memory controller and a few other peripherals, and loads a second stage bootloader. No interaction is possible with this first stage bootloader, and it is typically provided by the CPU vendor.
3. The **second stage bootloader** offers more features: usually a shell, with commands. It allows to manipulate the storage devices, the network, configure the boot process, etc. This bootloader is typically generic and open-source.

**Open-source bootloaders**

**U-Boot**  
See Also:
- [http://www.denx.de/wiki/U-Boot](http://www.denx.de/wiki/U-Boot)

![U-Boot principle](image)

Figure 2.39: *U-boot principle*

U-Boot is the de-facto standard in open-source bootloaders.

Available on ARM, PowerPC, MIPS, m68k, Microblaze, x86, NIOS, SuperH, Sparc.

Huge hardware support available, large number of features (networking, USB, SD, etc.)

**Wiki area**  
See Also:
- [http://www.denx.de/wiki/U-Boot](http://www.denx.de/wiki/U-Boot)

Welcome to the Wiki area for cooperating on U-Boot development.

**Coding style**  
See Also:
- [http://www.denx.de/wiki/U-Boot/CodingStyle](http://www.denx.de/wiki/U-Boot/CodingStyle)

The following Coding Style requirements shall be mandatory for all code contributed to the U-Boot project.

Exceptions are only allowed if code from other projects is integrated with no or only minimal changes.

**Versions**
Development tools, Release 2012.06.18

**U-Boot versions**  See Also:


Like with many other things, U-Boot development follows the Linux model of a release cycle. That means:

- We will have U-Boot releases at a fixed interval.
- The release interval shall be (approximately) 3 months.
- Immediately following each release, there will be a “merge window” of normally 2 weeks. While this merge window is open, new features can be added to the U-Boot source tree. Linus Torvalds explains here and there what the term “merge window” is supposed to mean.
- After the merge window closes, no new features may be added to allow for a release candidate phase which is intended to fix bugs and regressions.

**Warning:** even though we follow Linux ways in may respects, there are differences in the actual procedures, which are documented in section Development Process.

**Version Numbers**  Starting with the release in October 2008, the names of the releases were changed from numerical release numbers without deeper meaning into a time stamp based numbering.

Regular releases are now identified by names consisting of the calendar year and month of the release date.

Additional fields (if present) indicate release candidates or bug fix releases in “stable” maintenance trees.

Examples:

U-Boot v2009.11 - Release November 2009
U-Boot v2009.11.1 - Release 1 in version November 2009 stable tree
U-Boot v2010.09-rc1 - Release candidate 1 for September 2010 release

**U-boot versions**

**U-Boot v2011.09**  See Also:


**Barebox**  See Also:

- [http://www.barebox.org](http://www.barebox.org)
- [git://git.pengutronix.de/git/barebox.git](git://git.pengutronix.de/git/barebox.git)

Barebox, a newer open-source bootloader, with a cleaner design than U-Boot, but less hardware support for the moment.

Barebox (formerly known as u-boot-v2) is a bootloader that inherits the best of U-Boot and the Linux kernel: The size and look-and-feel of u-boot, with driver model and lots of design concepts from the kernel.

Note also the slides from the ELCE09 talk by Sascha Hauer, creator of barebox.
Barebox is a bootloader that initializes a hardware and boots Linux and maybe other operating systems or bare metal code on a variety of processors.

It was initially derived from U-Boot and captures up with several of it’s ideas, so users being familiar with U-Boot should come into production quickly with barebox.

However, as the barebox developers are highly addicted to the Linux kernel, its coding style and code quality, we try to stick as closely as possible to the methodologies and techniques developed in Linux.

In addition we have a strong background in POSIX, so you’ll find several good old Unix traditions realized in Barebox as well.

PC bootloaders

Unified Extensible Firmware Interface  See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Extensible_Firmware_Interface

Micrologiciel EFI à la place du BIOS  Sur certains PC actuels, c’est le micrologiciel EFI (et non pas le BIOS) qui est utilisé pour lancer le chargeur d’amorçage : l’EFI lit la GPT du disque (GUID Partition Table, voir (en) GPT) pour déterminer l’emplacement de la routine d’amorçage.
The Unified Extensible Firmware Interface (UEFI) is a specification that defines a software interface between an operating system and platform firmware. UEFI is a more secure replacement for the older BIOS firmware interface, present in all IBM PC-compatible personal computers, which is vulnerable to bootkit malware.[1][2]

The original EFI (Extensible Firmware Interface) specification was developed by Intel. In 2005, development of the EFI specification ceased in favour of UEFI, which had evolved from EFI 1.10. The UEFI specification is being developed by the industry-wide organization Unified EFI Forum.

UEFI is not restricted to any specific processor architecture and can run on top of, or instead of, older BIOS implementations.

Grandes lignes Une des fonctions principales d’UEFI est l’amorçage d’un système d’exploitation. Celui-ci se fait par l’intermédiaire d’un programme d’amorçage, qui est un cas particulier d’application UEFI. Le secteur de démarrage du BIOS n’est plus utilisé.

UEFI gère pour les disques, outre le partitionnement classique par MBR (limité à 2,2 To), un nouveau système de partitionnement nommé GPT (globally unique identifier partition table). Le GPT permet 128 partitions principales sur un support de capacité allant jusqu’à 9.4 Zo (milliards de téraoctets).

UEFI permet ainsi le démarrage sur des disques de 2,2 To et plus.

L’UEFI permet également :

• grâce à une pile de protocoles réseau, les mises à jour automatique du micrologiciel depuis le site du constructeur, à la manière des mises à jour d’OS ;

• grâce à la gestion bas niveau des disques, le clonage de disques sans passer par l’OS, ce qui facilite les copies de disques hébergeant plusieurs systèmes d’exploitation.

Micrologiciel BIOS Dans le cas le plus simple, il n’y a qu’une seule partition du disque de boot : le micrologiciel BIOS charge les 512 premiers octets de ce disque, ces 512 octets constituant le MBR.

À partir des informations du MBR, il détermine l’emplacement du chargeur d’amorçage.

Si le disque de boot a plusieurs partitions, le micrologiciel BIOS lit le MBR du disque, puis le VBR de la partition (Volume Boot Record, voir (en) VBR).

À partir de ces informations, il peut déterminer l’emplacement du chargeur d’amorçage et le lancer.

Si le support de boot est une disquette, c’est le VBR de cette disquette qui est utilisé pour déterminer l’emplacement du chargeur d’amorçage.

Open-source bootloaders

GRUB See Also:

• http://www.gnu.org/software/grub/

• http://fr.wikipedia.org/wiki/GRUB

GNU GRUB (acronyme signifiant en anglais « GRand Unified Bootloader ») est un programme de multiboot, libre, au même titre que LILO (Linux loader), qui permet de choisir au démarrage de son ordinateur entre plusieurs systèmes d’exploitation.

Ses avantages sont notamment la gestion d’autres systèmes que Linux et Windows (utile pour Hurd, Solaris, FreeBSD et OpenBSD), la lecture de la configuration au démarrage (pas besoin de réinstaller GRUB dans le secteur d’amorçage après un changement de configuration, contrairement à LILO), une ligne de commande permettant de changer la configuration au démarrage et surtout la reconnaissance en natif de divers systèmes de fichiers existants. Il possède
également une sorte de langage de commande simple permettant de « rattraper » un amorçage qui se serait mal passé, suite au mauvais adressage d’une partition, par exemple.

Grub doit être capable de reconnaître tous les systèmes de fichiers sur lesquels il pourrait être amené à démarrer. Il est pour cette raison beaucoup plus volumineux que LILO.

**Introduction** GNU GRUB is a Multiboot boot loader. It was derived from GRUB, the GRand Unified Bootloader, which was originally designed and implemented by Erich Stefan Boleyn.

Briefly, a boot loader is the first software program that runs when a computer starts. It is responsible for loading and transferring control to the operating system kernel software (such as the Hurd or Linux).

The kernel, in turn, initializes the rest of the operating system (e.g. GNU).

**GRUB Development** GRUB 2 has replaced what was formerly known as GRUB (i.e. version 0.9x), which has, in turn, become GRUB Legacy. Enhancements to GRUB are still being made, but the current released versions are quite usable for normal operation.

GRUB Legacy is no longer being developed. For the differences between GRUB Legacy and GRUB, see the Grub Legacy Documentation.

**Virtualization**

**Virtualization boxes** See Also:
- [http://git.gnome.org/browse/gnome-boxes](http://git.gnome.org/browse/gnome-boxes)
How to help? Seems many people are already very excited about this little project of ours and I’m hoping this blog entry will attract more contributors so I wanted to point out some things we need help with. If you are interested in UI work, Marc-Andre has written down a TODO for Boxes that you can pick some tasks from. Other than that, we still need a lot of help with two of our main dependencies:

- **libosinfo**: This library is our store for information on operating systems and means to detect operating systems from installation media.
- **libvirt-glib**: libvirt-glib wraps libvirt to provide a high-level object-oriented API better suited for glib-based applications.

While Boxes is written in Vala, these libraries are completely written in C so if you are a C hacker and want to contribute, these would be good places to start with. While most of the work needed on libosinfo is that of populating its database with information on all kinds of operating systems out there, libvirt-glib still lacks a lot of needed API. One particular part of libvirt-glib that needs the most work and is of highest priority to us is its API to deal with libvirt’s configuration XML.

Virtualbox See Also:

- https://fr.wikipedia.org/wiki/Oracle_VM_VirtualBox
- https://secure.wikimedia.org/wikipedia/en/wiki/VirtualBox
- http://virtualboxes.org/images/
- http://www.commentcamarche.net/download/telecharger-3673479-virtualbox

Oracle VM VirtualBox (anciennement VirtualBox) est un logiciel de virtualisation créé par InnoTek.
Il est disponible en tant qu’hôte sur les systèmes d’exploitation:
  
  • Windows,
  • Linux 32 et 64 bits,
  • FreeBSD 32 et 64 bits
  • et Mac OS X.

Il supporte en tant qu’invité:
  
  • Windows (dont Vista et 7),
  • Linux 2.x, OS/2 Warp,
  • OpenBSD
  • et FreeBSD entre autres.

See Also:

VirtualBox permet d’émuler complètement un PC, comme si vous aviez un second ordinateur dans une simple fenêtre. C’est utile pour tester d’autres systèmes d’exploitation sans repartitionner son disque dur, pour naviguer en toute sécurité ou pour tester un logiciel sans risque de rendre son système d’exploitation instable.

Vous pouvez créer autant de machines virtuelles et installer tous les systèmes d’exploitation que vous le souhaitez. Il est possible de définir, pour chaque machine virtuelle, sa mémoire RAM, son espace disque, si elle aura accès aux ports USB, au réseau, à la carte son, etc.

VirtualBox contient un gestionnaire de disques qui vous permet de créer des disques virtuels sous forme de fichiers .vdi qui apparaîtront comme de vrais disques dans les machines virtuelles.

Cela vous permet donc de “créer” à volonté des disques, et cela sans jamais avoir à repartitionner votre disque dur.

Vous pouvez également utiliser directement des images ISO de CD et DVD, ce qui permet de tester des distributions Linux sans avoir à les graver.

Enfin, VirtualBox possède un serveur RDP intégré, ce qui permet de démarrer une machine virtuelle sur un ordinateur, et utiliser cette machine virtuelle à partir d’un autre ordinateur.

Note: Pour les machines sous un autre OS que Windows, veuillez choisir une autre version parmi celles-ci.

Installation des add-ons   See Also:
  
  • http://www.commentcamarche.net/faq/7576-virtualbox-installer-les-additions-client-dans-ubuntu

Virtualbox and Linux

Virtualbox and ubuntu   See Also:
http://virtualboxes.org/images/ubuntu/

Virtualbox tools
Fabtest (test Fabric scripts on VirtualBox VMs)  See Also:
- http://pypi.python.org/pypi/fabtest/
- https://bitbucket.org/kmike/fabtest/src
- Fabric

Fabtest is a set of utilities and base TestCases that aid testing Fabric scripts against VirtualBox VMs. License is MIT.
VM is rolled back to initial state before each test so tests can do anything with target system; Fabric commands can be run from Python.

VMs  In order to run tests you’ll need VirtualBox 4.x and an OS image.

Image should have ssh server installed

Example VMs:
- Lenny.ova (312M)
- Squeeze.ova (436M)
- Ubuntu-10.10.ova (277M)

Virtualbox versions

Virtualbox 4.1.18  VirtualBox 4.1.8 (released 2011-12-19)

This is a maintenance release. The following items were fixed and/or added:

- VMM: fixed VERR_MAP_FAILED during savestate under certain circumstances (bug #7929)
- GUI: stop updating the VM status icons when the VM is paused (bug #8368)
- VBoxManage: fixed wrong return code after startvm (bug #9642)
- BIOS: fixed hang at launch of DOS applications generated by Clipper 5.3 (note that hardware virtualization may be required)
- USB: fixed OS/2 boot hang when using recent USB drivers
- NAT: increase maximum number of parallel connections making connections with port forwarding more robust (#8471)
- Metrics: fixed potential problem with invalid access in guest metrics collection upon VM destruction
- Main: don’t crash if a medium is ejected twice (bug #9922)
- VBoxSVC: fixed crash under rare circumstances (e.g. client crash)
- VRDP: fixed screen freeze (bug #9620)
- OVF/OVA: fixed broken disk images on import under rare circumstances
- OVF/OVA: better error message when importing corrupted appliances
- VMDK/VHD: fixed a possible corruption with host cache disabled when using snapshots under rare circumstances (bug #9749)
- 3D Support: Fixed fullscreen mode issues for ATI graphics (bug #9775), Windows Media Player rendering for XPDM-based Direct3D support (bug #8341). Multiple fixes to XPDM and WDDM - based 3D support for Windows Guests and for 3D support in general
- Linux hosts: fixes for Fedoras Linux 2.6.41 (bug #9948)
• Linux hosts/guests: fixes for Linux 3.2 (bug #9743)

• Solaris Additions: various shared folder fixes (bugs #9856, #9862, #9917)

• Windows Additions: various fixes for Direct3D support (un)installation, added detection of missing or wrong Direct3D system files

Virtualbox et windows

Physical access

Virtualbox et ‘vrai’ Windows    See Also:

• http://www.guilde.asso.fr/lurker/thread/20120125.074153.4498f5de.en.html

• http://www.sysprobs.com/access-physical-disk-virtualbox-desktop-virtualization-software

-------- Message original --------
Sujet: Re: Virtualbox et ‘vrai’ Windows
Date de renvoi : Sat, 28 Jan 2012 14:39:40 +0100
De (renvoi) : guilde@guilde.asso.fr
Date : Sat, 28 Jan 2012 14:39:30 +0100
De : Olivier Allard-Jacquin <olivieraj@free.fr>
Pour : guilde@guilde.asso.fr

Bonjour,

Le 24/01/2012 14:11, Frédéric a écrit :
> Le mardi 24 janvier 2012, sly (sylvain letuffe) a écrit :
> 
> >> Les deux sont possibles, il y a cependant quelques trucs sioux à gérer
> >> pour convertir en disque virtuel, mieux vaut refaire l’installation est
> >> activer le windows avec la clef de licence.
> >
> > La solution d’utiliser directement l’installation d’origine est donc
> > possible est plus simple ? Parfait :o)
> >
> > Tu sais où c’est documenté ? Je ne trouve pas, dans la doc officielle...

Ce que tu veux faire, c’est de l’accès physique depuis virtualbox. Je l’ai déjà pratiqué par le passé, et l’explication est ici :

• http://www.sysprobs.com/access-physical-disk-virtualbox-desktop-virtualization-software

VBoxManage internalcommands createrawvmdk -filename mydrive.vmdk -rawdisk /dev/sda

Mais attention, il y a plusieurs raisons pour lesquelles cela peut ne pas marcher, et quelques précautions à prendre.

• D’abord, sache que la licence de Windonws Seven “starter” n’autorise pas une utilisation dans une machine virtuelle. Seul les licences plus chères l’autorisent.

• Si le Windows utilise un tatouage placé dans le BIOS, il ne pourra pas le retrouver dans le BIOS de la virtualbox. Donc, il refusera de booter

• Si le Windows utilise un tatouage placé dans le MBR (512 premiers octets du disque), ton LILO/GRUB risque de l’écraser. Il vaut mieux alors laisser le boot loader de Windows se charger du boot, puis de “rebondir” sur le Linux http://doc.ubuntu-fr.org/tutoriel/comment_amorcer_ubuntu_avec_bootmgr

2.22. Operating systems 881
Si le Windows utilise un tatouage placé dans la “piste 0”, il est préférable de faire un backup de cette piste, avec un bon vieux “dd”. En effet, GRUB pourra écraser le tatouage, lorsqu’il s’installera dans la “piste 0”:

- Boot le portable sur un liveCD/liveUSB
- Utilise “fdisk -lu /dev/sda” pour repérer la taille de la piste 0 (généralement des blocs 0 à 62)
- Et créé ton backup : dd if=/dev/sda of=ma_piste0.bin bs=512 count=63

Pour essayer de détecter le type de tatouage utilisé, tu peux utiliser un “hexedit /dev/sda” ou “hexdump -C /dev/sdalless”, afin d’étudier le contenu du début du disque. L’idée est alors de trouver des trucs “louches” par rapport à une installation non-tatouée.

Sache enfin que le mécanisme de restauration du Windows pourra ne plus fonctionner, simplement parce que la taille de la partition Windows a changé suite à l’installation du Linux (j’ai déjà vu ce type de vérification stupide de la part d’un mécanisme de restauration choisi par un construteur). Un conseil : Utilise donc un bon vieux http://clonezilla.org/ afin de créer ton propre mécanisme de restauration.

Une fois le Linux installé, utilise http://www.sysprobs.com/access-physical-disk-virtualbox-desktop-virtualization-software afin de créer ton “disque virtuel”. Ce ne sera en fait qu’un petit fichier texte de quelques octets.

Le problème, c’est que ton utilisateur Linux (“frederic”) a besoin d’un accès direct à /dev/sda . Soit tu modifies udev pour le permettre, soit tu t’écrits un petit programme qui fait un chown (via sudo). L’avantage, c’est que quelques secondes plus tard, Linux remettra les droits normaux de /dev/sda , ce qui empêchera d’autres programmes que virtualbox d’avoir un accès direct au disque

```
#!/bin/bash -norc
sudo chown frederic /dev/sda
exec virtualbox --startvm "le nom de ton virtualbox"
```

e et dans ton /etc/sudoers.d/virtualbox (pour une Debian):

```
User_Alias PRIVILEGE_USERS = frederic
Cmd_Alias PRIVILEGE = \\
    chown frederic /dev/sda
PRIVILEGE_USERS ALL = NOPASSWD: PRIVILEGE
```

Enfin, dans cette configuration tu peux être confronté à des lenteurs tout bonnement inacceptables, avec un temps de boot de Windows de plusieurs minutes (juste pour voir apparaître la fenêtre de login). La cause de ce problème vient d’une configuration APIC+Windows+Virtualbox. Je ne sais pas si c’est toujours d’actualité, notamment avec un Windows Seven (j’ai eu le problème il y a quelques années avec un Windows XP). La solution est là :

- https://www.google.com/#q=virtualbox+apic+windows

Notes que dans une telle configuration, ton Windows a accès lui aussi au /dev/sda, donc potentiellement il peut mettre à mal ton Linux

De plus, tu pourrais re-lancer ton Linux depuis le virtualbox, lui-même exécuté par le Linux. Et alors là, bonjour les dégâts !!!
Afin de limiter les risques, je te suggère donc de laisser le Windows en boot par défaut, ou de ne pas permettre à un OS de booter automatiquement au bout d’un timeout de grub ou du boot loader de Windows.

Je pense que cette réponse devrait couvrir tout les problèmes que tu devrais rencontrer.

Cordialement, Olivier

Windows 7  See Also:

• http://www.technologie-college.com/Installation-de-Windows-7-avec.html
• http://forum-vista-seven.1fr1.net/t1215-tutoriel-installer-windows-7-sur-une-machine-virtuelle-virtualbox

2.23 Privacy

2.23.1 Privacy

See Also:

• http://guilde.asso.fr/rencontres/20111013/ (Identité numérique et anonymat)
• http://www.netpublic.fr/2012/05/reseaux-sociaux-decouverte-interet-enjeux-dangers/  
• http://www.google.com/intl/fr/goodtoknow/

Privacy (from Latin: privatus “separated from the rest, deprived of something, esp. office, participation in the government”, from privo “to deprive”) is the ability of an individual or group to seclude themselves or information about themselves and thereby reveal themselves selectively. The boundaries and content of what is considered private differ among cultures and individuals, but share basic common themes. Privacy is sometimes related to anonymity, the wish to remain unnoticed or unidentified in the public realm. When something is private to a person, it usually means there is something within them that is considered inherently special or personally sensitive.

The degree to which private information is exposed therefore depends on how the public will receive this information, which differs between places and over time. Privacy is broader than security and includes the concepts of appropriate use and protection of information.
List of privacy softwares

Duckduckgo

- https://duckduckgo.com/privacy.html
- http://donttrack.us/

Ghostery

See Also:
http://www.ghostery.com/about

Ghostery tracks the trackers and gives you a roll-call of the ad networks, behavioral data providers, web publishers, and other companies interested in your activity.

EFF

See Also:
- http://www.eff.org/
- http://twitter.com/#!/EFF
- http://www.eff.org/https-everywhere

HTTPS Everywhere is a Firefox extension produced as a collaboration between The Tor Project and the Electronic Frontier Foundation.

It encrypts your communications with a number of major websites

Anonymat

See Also:
http://guilde.asso.fr/rencontres/20111013/anonymat.pdf

Outils libres pour l’anonymat.

TorProject

See Also:
- https://www.torproject.org/
- https://www.torproject.org/torbutton/

Other links on twitter

- http://twitter.com/#!/privacyint
Mozilla, privacy box, guide boum

Mozilla Do Not Track

As many people know, Mozilla jumped into Do Not Track (DNT) in a big way earlier this year by providing Firefox users on desktop and mobile with a simple way to tell companies to stop tracking them online. We did this before knowing exactly how sites and advertisers would respond. We believed we had to do something to advance the debate and we counted on developers seeing the technical advantages to our approach over current proposals and practices.

34 extensions Firefox pour vous protéger et tester la sécurité de vos sites  See Also:
http://korben.info/34-extensions-firefox-pour-vous-proteger-et-tester-la-securite-de-vos-sites.html

Privacy box, Rebellyon

See Also:
http://rebellyon.info/Un-formulaire-securise-a-la.html
Un formulaire sécurisé à la Wikileaks pour envoyer des infos et des documents à Rebellyon
Afin de permettre à tout le monde de nous envoyer des informations et des documents de manière sécurisée, et éventuellement anonyme, le collectif d’animation de Rebellyon met en place une Privacy Box.

Détails pratiques  En gros il s’agit d’un bête formulaire d’envoi d’email, sauf que celui-ci sera ensuite crypté ainsi que les documents que vous pourrez joindre à votre message (plus précisément, ils seront chiffrés avec la technique de chiffrage asymétrique).
L’ensemble du service mis en place par la fondation allemande Pryvacybox.de permet de garantir votre anonymat.
Lancé par la Privacy Foundation allemande, une ONG de défense de la vie privée et de la liberté d’expression, privacybox.de fait encore mieux, dans la mesure où elle permet à tous ceux qui ne peuvent ou ne veulent pas utiliser GPG ou PGP d’écrire de façon confidentielle, anonyme et sécurisée, à tout journaliste, blogueur ou internaute qui, utilisateur de GPG ou de PGP, s’y est inscrit (et c’est gratuit, forcément, et puis facile, aussi).
Si vous en avez marre de l’omerta de la presse commerciale lyonnaise sur toute une série de sujets, si vous désirez rendre public des documents sans risquer que la DCRI vous retrouve, en chopant par exemple les fadettes
de votre interlocuteur, ou en remontant des envois de mails... envoyez-nous par ce formulaire vos infos ! : 
https://privacybox.de/rebellyon.msg

La fiabilité des documents et informations sera ensuite vérifiée avant publication. Si vous souhaitez que nous puisions vous contacterer, n’oubliez pas d’indiquer un moyen de vous joindre (ça peut être via une autre privacybox par exemple).

Pour renforcer votre anonymat Anonymiser votre connexion internet à la Privacybox :

Il est possible de se connecter de deux manières qui permettent d’invisibiliser complètement à quel site vous vous connectez, Tor ou I2P.

Ces deux manières de se connecter nécessitent d’avoir certains logiciels installés. Les adresses de la privacybox de Rebellyon sont alors :

- si TOR est activé sur votre ordinateur : http://c4wcxidkfhmzw6.onion/rebellyon.msg
- via I2P, le projet Internet furtif : http://privacybox.i2p/rebellyon.msg

Effacer les traces informatiques vous identifiant des documents que vous nous propozez:

- pour les documents texte : il est conseillé de les transformer en pdf. Les documents word peuvent en effet contenir des informations sur votre ordinateur si vous avez édité le document. Si vous ne savez pas le faire, nous le ferons dès réception et détruirons le fichier contenant des informations.


Si vous n’êtes pas sûrs de l’ordinateur que vous utilisez (risque de mouchard logiciel, maintenant prévu dans la loi) nous vous conseillons d’utiliser un live CD Linux sécurisé (qui utilise Tor par défaut) : par exemple Tails. Ou si vous vous sentez particulièrement surveillés, utilisez un autre ordinateur avec ce live CD sécurisé.

Enfin, vous pouvez déposer des lettres dans notre boîte située à La Gryffe, 5 rue Sébastien Gryphe, Lyon 7e ou nous les faire parvenir par voie postale de la même manière, mais là, malgré toute l’estime que nous portons à nos camarades facteurs et factrices, on ne garantie rien.

**Tails  See Also:**

http://tails.boum.org/

Tails est un live CD ou live USB dont le but est de préserver votre vie privée et anonymat.

Il vous permet de :

- utiliser Internet de manière anonyme (presque) partout et avec n’importe quel ordinateur; toutes les connexions sortantes vers Internet sont obligées de passer par le réseau Tor ;
- ne pas laisser de traces sur l’ordinateur utilisé, à moins que vous ne le demandiez explicitement.

**Notes**

[1] Pour une introduction plus détaillée de ce service, on pourra lire PrivacyBox : permettre l’envoi d’emails chiffrés aux personnes qui n’ont pas (encore) de clés PGP.


[3] Dont une présentation assez complète a été publiée sur Rebellyon : L’anonymat sur Internet grâce à la technique du routage en oignon.
ExifToolGUI ne se suffit pas à lui-même, c’est une interface graphique sur un programme en ligne de commande. Il faut :

- aller le télécharger là (Windows Executable) : http://www.sno.phy.queensu.ca/phil/exiftool
- extraire le contenu du .zip le fichier
- renommer le fichier exiftool(-k).exe en exiftool.exe ou exiftool(-k) en exiftool
- copier ou déplacer le fichier renommé.

Guide auto-défense numérique


See Also:

- http://guide.boum.org/tomes/1_hors_connexions/pdf/
- http://guide.boum.org/prochains_tomes/

Parution de la seconde édition du Guide d’autodéfense numérique

Il y a un an, paraissait le premier tome du Guide - premiers pas sur la place publique de cette aventure souhaitant diffuser savoirs et savoirs-faire, face aux impacts du monde numérique sur nos vies.

Alors que le deuxième tome se fait attendre, ces derniers mois ont été émaillés par la sortie de quelques lois, la publication de recherches et de nouvelles versions des systèmes Debian et Tails.

Ces changements se devaient d’être reflétés dans une édition révisée du premier tome. La voici aujourd’hui : hors connexions toujours, mais corrigée et augmentée.

Le travail sur le deuxième tome, dédié aux enjeux de l’utilisation des réseaux et d’Internet, continue pendant ce temps. La tâche est ardue, les problèmes complexes, la cible mouvante, mais il finira bien par arriver.

2.24 Programming

2.24.1 Programming

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Cat%C3%A9gorie:Programmation_informatique

Application programming interface (API)

See Also:

API advices

Contents

• API advices
  – api-design-is-ui-for-developers
  – Easy Access
  – Example Code
  – Documentation
  – Full Enumeration of Responses
  – Variety of Response Language
  – Human Readable Response
  – Human Readable Requests
  – Unchanging
  – Simplicity
  – Libraries
  – Detailed Errors
  – Feedback
  – So What?
  – Further Reading

api-design-is-ui-for-developers  See Also:

• http://shkspr.mobi/blog/index.php/2012/03/api-design-is-ui-for-developers/

So, here are my hastily scribbled thoughts on what an API needs at a minimum to entice the busy developer.

I don’t think any of these are Earth-shattering, but it’s amazing how many APIs fail to meet even these basic requirements.

Easy Access

• Don’t make me register, set up an account, or fill in a load of forms – I just want to see what I can do with you before I make a commitment.

• Don’t give me absurd key signing requirement – I should be able to call the API from a web browser just by typing stuff in.

• Do make it as easy as possible for me to get started – get your CEO (or similar) and see how long it takes her to set up an account & launch her first call. If it’s more than 5 minute, go back to the drawing board.

Example Code

Documentation

I need to know what I can do, how I can do it, why I should do it a certain way, and what response to expect.

Full Enumeration of Responses  A good example is Twitter’s “retweet_count”, the documentation used to imply that the response would always be an integer.

However, it would occasionally respond with a string of “100+”. Naturally, this meant developers would write code expecting ints which would fail whenever a string was encountered. If you can’t tell me what responses to expect – how can I code something which handles those responses correctly?
Variety of Response Language  The customer is always right. If the customer (developer) wants JSON, XML, PHPObject, or just plain text – you should give it to them.

It’s the API designer’s job to make life easy for developers – so reply in whatever formats the developer wants.

Human Readable Response  Developers are humans! Yes! It’s true! And they can’t all pretty-print JSON in their heads. Give them something they can read without resorting to external tools.

Unless you have a very good reason not to – all your responses should be pretty-printed. It helps with debugging and makes life just that little bit easier for a struggling developer.

Human Readable Requests  Developers are humans! Yes! It’s true! And they can’t all remember every little acronym in their heads. Give them something they can read without resorting to documentation.

Unchanging  Consistency is a virtue. If you have two similar APIs (say, search & read) they should take the same parameters and produce identically formatted responses.

If you have to change the way your API responds, or the way it accepts request, that’s fine – but use versioning so that developers don’t have to update their code if they don’t want to.

Never deprecate anything! Once an embedded device has had its firmware burned, it’s unlikely to ever be updated. If people or services rely on you, it’s simply unacceptable to kill something off.

Remember, not every developer or end user can update their software.

Simplicity  Try to explain what you’re doing in a single sentence. If you can’t – it’s probably too complicated.

Libraries  I’m sure you’re very proud that your community has created some libraries – but they’re not good enough.

Detailed Errors  Do you really need to throw an error? Can your API take a “best” guess at what the user was trying to do? Be generous in what you accept.

Feedback  Provide a mechanism where people can feed back what they think is broken, poorly implemented, missing, or just plain confusing.

So What?  There is nothing new in this post to the seasoned developer.

But as Matt Gemmell reminded me recently – some people just don’t know the basics.

If you’re interested in making your API useful for developers, you have to treat it like any other product.

You have to consider HCI factors, you have to do product testing, design, and planning.

Your API is a product.

Treat your developers as you would your most profitable users.

Further Reading  There are many books on this subject – the two I recommend are:

• Basics of the Unix Philosophy (free).

• The Design of Everyday Things (paper or ebook).
Parse Designing great api docs

See Also:

- http://blog.parse.com/2012/01/11/designing-great-api-docs/
- http://www.parse.com/docs/ios_guide

Qt API Design Principles

See Also:

http://wiki.qt-project.org/API_Design_Principles

Zeromq (API)

See Also:

- http://api.zeromq.org/
- http://czmq.zeromq.org/manual:czmq

Advanced Message Queuing Protocol (AMQP) Programming

See Also:


How to explain ØMQ?

See Also:

- http://www.zeromq.org/
- http://twitter.com/#!/hintjens
- http://zguide.zeromq.org/
- http://lwn.net/Articles/370307/
- http://www.coastrd.com/zeromq-messaging
- http://nichol.as/zeromq-an-introduction

Contents

- How to explain ØMQ?
  - Physics of software
  - Some Assumptions
  - zeromq Architecture
  - Links
  - Versions
  - Projects using Ømq
  - Bindings
  - Use cases
Some of us start by saying all the wonderful things it does:

- It’s sockets on steroids.
- It’s like mailboxes with routing.
- It’s fast!

Others try to share their moment of enlightenment, that zap-pow-kaboom satori paradigm-shift moment when it all became obvious. Things just become simpler. Complexity goes away. It opens the mind.

Others try to explain by comparison. It’s smaller, simpler, but still looks familiar. Personally, I like to remember why we made ØMQ at all, because that’s most likely where you, the reader, still are today.

**Physics of software**  Programming is a science dressed up as art, because most of us don’t understand the physics of software, and it’s rarely if ever taught.

The physics of software is not algorithms, data structures, languages and abstractions. These are just tools we make, use, throw away.

The real physics of software is the physics of people.

Specifically, our limitations when it comes to complexity, and our desire to work together to solve large problems in pieces. This is the science of programming: make building blocks that people can understand and use easily, and people will work together to solve the very largest problems.

We live in a connected world, and modern software has to navigate this world. So the building blocks for tomorrow’s very largest solutions are connected and massively parallel. It’s not enough for code to be “strong and silent” any more.

Code has to talk to code. Code has to be chatty, sociable, well-connected. Code has to run like the human brain, trillions of individual neurons firing off messages to each other, a massively parallel network with no central control, no single point of failure, yet able to solve immensely difficult problems.

And it’s no accident that the future of code looks like the human brain, because the endpoints of every network are, at some level, human brains. If you’ve done any work with threads, protocols, or networks, you’ll realize this is pretty much impossible. It’s a dream. Even connecting a few programs across a few sockets is plain nasty, when you start to handle real life situations. Trillions? The cost would be unimaginable. Connecting computers is so difficult that software and services to do this is a multi-billion dollar business.

So we live in a world where the wiring is years ahead of our ability to use it. We had a software crisis in the 1980s, when people like Fred Brooks believed there was no “Silver Bullet”. Free and open source software solved that crisis, enabling us to share knowledge efficiently. Today we face another software crisis, but it’s one we don’t talk about much. Only the largest, richest firms can afford to create connected applications. There is a cloud, but it’s proprietary. Our data, our knowledge is disappearing from our personal computers into clouds that we cannot access, cannot compete with. Who owns our social networks? It is like the mainframe-PC revolution in reverse.

We can leave the political philosophy for another book. The point is that while the Internet offers the potential of massively connected code, the reality is that this is out of reach for most of us, and so, large interesting problems (in health, education, economics, transport, and so on) remain unsolved because there is no way to connect the code, and thus no way to connect the brains that could work together to solve these problems. There have been many attempts to solve the challenge of connected software. There are thousands of IETF specifications, each solving part of the puzzle. For application developers, HTTP is perhaps the one solution to have been simple enough to work, but it arguably makes the problem worse, by encouraging developers and architects to think in terms of big servers and thin, stupid clients.

So today people are still connecting applications using raw UDP and TCP, proprietary protocols, HTTP, WebSockets. It remains painful, slow, hard to scale, and essentially centralized. Distributed p2p architectures are mostly for play, not work. How many applications use Skype or Bittorrent to exchange data? Which brings us back to the science of programming. To fix the world, we needed to do two things. One, to solve the general problem of “how to connect
any code to any code, anywhere”. Two, to wrap that up in the simplest possible building blocks that people could understand and use easily.

It sounds ridiculously simple. And maybe it is. That’s kind of the whole point.

ØMQ (ZeroMQ, 0MQ, zmq) looks like an embeddable networking library but acts like a concurrency framework. It gives you sockets that carry whole messages across various transports like in-process, inter-process, TCP, and multicast.

You can connect sockets Nto-N with patterns like fanout, pub-sub, task distribution, and request-reply. It’s fast enough to be the fabric for clustered products.

Its asynchronous I/O model gives you scalable multicore applications, built as asynchronous message-processing tasks. It has a score of language APIs and runs on most operating systems. ØMQ is from iMatix and is LGPL open source.

**Some Assumptions** We assume you are using the latest stable release of ØMQ. We assume you are using a Linux box or something similar. We assume you can read C code, more or less, that’s the default language for the examples. We assume that when we write constants like PUSH or SUBSCRIBE you can imagine they are really called ZMQ_PUSH or ZMQ_SUBSCRIBE if the programming language needs it.

**zeromq Architecture** See Also:


ØMQ is a messaging system, or “message-oriented middleware”, if you will.

It’s used in environments as diverse as financial services, game development, embedded systems, academic research and aerospace.

**Links** See Also:

http://lwn.net/Articles/370307/

**Versions**

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>zmq releases</td>
</tr>
<tr>
<td>• zmq releases</td>
</tr>
<tr>
<td>– Introduction</td>
</tr>
<tr>
<td>– Releases</td>
</tr>
</tbody>
</table>

**Introduction** We maintain four branches of ZeroMQ at present:

• 2.0.x is the deprecated 2.x release, which receives user-contributed bug fixes only.

• 2.1.x is the stable 2.x release, which receives downstreamed bug fixes only.

• 2.2.x is the 2.x ongoing release, which receives downstreamed bug fixes, new functionality, and additional layers, e.g. czmq. This is mainly a placeholder for continuing 2.x maintenance after 2.1 is frozen. One goal with 2.2.x is to expand the ZeroMQ distribution package to include essential core language bindings.

• 3.1.0 is the new 3.x unstable release.

2.0 and 3.1 implements the same wire level protocol, ZMTP/1.0. 3.0 (deprecated) uses an incompatible version of ZMTP.
Releases

**zmq 3.1**  libzmq development happens on the Source git repository at [http://github.com/zeromq/libzmq](http://github.com/zeromq/libzmq).
This git also holds the project issue tracker.
The master branch holds the latest unstable version of libzmq (version 3.1 at present).
Source code documentation is at [http://travlr.github.com/libzmq](http://travlr.github.com/libzmq).
Man pages for current versions are at [http://api.zeromq.org](http://api.zeromq.org).

Projects using ØMQ?

**circus**  See Also:
- [https://github.com/mozilla-services/circus](https://github.com/mozilla-services/circus)

Contents

- circus
  - Introduction
  - Circus CLI

Introduction  Circus is a program that will let you run and watch multiple processes.
Circus can be driven through a command-line interface, or programmatically through its APIs.
It shares some of the goals of Supervisord, BluePill and Daemontools.

Circus CLI  See Also:
For each command below, we provide a usage example with circusctl but also the input / output zmq messages.

Cubic web  See Also:
- [http://www.cubicweb.org/blogentry/2219569](http://www.cubicweb.org/blogentry/2219569)
There has been a growing interest in ZMQ in the past months, due to its ability to efficiently deal with message passing, while being light and robust.
We have worked on introducing ZMQ in the CubicWeb framework for various uses:
- As a replacement/alternative to the Pyro source, that is used to connect to distant instances. ZMQ may be used as a lighter and more efficient alternative to Pyro. The main idea here is to use the send_pyobj/recv_pyobj API of PyZMQ (python wrapper of ZMQ) to execute methods on the distant Repository in a totally transparent way for CubicWeb.
- As a JSONServer. Indeed, ZMQ could be used to share data between a server and any requests done through ZMQ. The request is just a string of RQL, and the response is the result set formatted in Json.

2.24. Programming 893
• As the building block for a simple notification (publish/subscribe) system between CubicWeb instances. A component can register its interest in a particular topic, and receive a callback whenever a corresponding message is received. At this point, this mechanism is used in CubicWeb to notify other instances that they should invalidate their caches when an entity is deleted.

glimpse  See Also:

• http://packages.python.org/glimpse/index.html
• http://packages.python.org/glimpse/manual/pools/gearman_pool.html

This worker pool uses a Gearman job server to manage the list of tasks (i.e., function invocations), and a small number of ZeroMQ channels for logging and command communication. To create a new Gearman cluster pool, use the MakePool function as:

```python
>>> config_file = "config.ini"
>>> pool = glimpse.pools.gearman_cluster.MakePool(config_file)
>>> pool.map(hex, [1, 2, 3])
['0x1', '0x2', '0x3']
```

Powerhose  See Also:

• https://github.com/mozilla-services/powerhose
• http://tarekziade.wordpress.com/2012/02/06/scaling-crypto-work-in-python/

scaling-crypto-work-in-python  See Also:

• http://tarekziade.wordpress.com/2012/02/06/scaling-crypto-work-in-python/

It turns out zeromq is perfect for this job – there are clients in Python and C++, and defining a protocol to exchange data from the Python web server to the crypto workers is quite simple.

In fact, this can be done as a reusable library that takes care of passing messages to workers and getting back results. It has been done hundreds of times, there are many examples in the zmq website, but I have failed to find any Python packaged library that would let me push some work to workers transparently, via a simple execute() call — if you know one tell me!

So I am building one since it’s quite short and simple –

The project is called PowerHose and is located here:

• https://github.com/mozilla-services/powerhose.

Here’s an example of using Powerhose:

• The Server – https://github.com/mozilla-services/powerhose/blob/master/examples/square_master.py
• The Python worker – https://github.com/mozilla-services/powerhose/blob/master/examples/square_worker.py
• The C++ worker (don’t look at the code :) – https://github.com/mozilla-services/powerhose/blob/master/examples/square_worker.cpp

For the Token server, we’ll have:

• A JobRunner in our Cornice application
• A C++ worker that uses Crypto++
The first benches look fantastic — probably faster that anything I’d have implemented myself using plain sockets :)

**I'll try to package Powerhose so other projects at Mozilla can use it.**

I am wondering if this could be useful to more people, since I failed to find that kind of tool.

How do you scale your CPU-bound web apps?

`pyscale`  **See Also:**

- [http://pypi.python.org/pypi/pyscale/0.1.1](http://pypi.python.org/pypi/pyscale/0.1.1)
- [https://github.com/alexcepoi/pyscale](https://github.com/alexcepoi/pyscale)

General purpose Python framework for writing highly scalable applications.

**About**  A typical application consists of several modules. Each module has its own process, stores a pidd file in `tmp/pids`, and has a logfile in `logs`.

A RPC protocol is implemented on top of zeromq in order to allow for inter-module communication.

Modules have an auto-adjustable number of workers in order to cope with a high number of requests.

These rpc requests will block until that module becomes available.

Read more about zeromq at [http://zguide.zeromq.org/](http://zguide.zeromq.org/)

Each module consists of several gevent greenlets. A basic module will already contain a few greenlets that handle incoming rpc requests.

You can spawn additional greenlets for your own needs. Read more about gevent at [http://www.gevent.org/](http://www.gevent.org/)

`zmqc`  **See Also:**

- [http://pypi.python.org/pypi/zmqc/0.0.1](http://pypi.python.org/pypi/zmqc/0.0.1)
- [http://github.com/zacharyvoase/zmqc](http://github.com/zacharyvoase/zmqc)

zmqc is a small but powerful command-line interface to ØMQ.

It allows you to create a socket of a given type, bind or connect it to multiple addresses, set options on it, and receive or send messages over it using standard I/O, in the shell or in scripts.

It’s useful for debugging and experimenting with most possible network topologies.

**Examples**

`zmqc -rc SUB 'tcp://127.0.0.1:5000'`

Subscribe to tcp://127.0.0.1:5000, reading messages from it and printing them to the console. This will subscribe to all messages by default (you don’t need to set an empty SUBSCRIBE option). Alternatively:

`zmqc -rc SUB -o SUBSCRIBE='com.organization.' 'tcp://127.0.0.1:5000'`

This will subscribe to all messages starting with com.organization..

`ls | zmqc -wb PUSH 'tcp://*:4000'`
Send the name of every file in the current directory as a message from a PUSH socket bound to port 4000 on all interfaces.

Don’t forget to quote the address to avoid glob expansion.

Bindings

**zmq bindings**

*C czmq library*  
See Also:  
- [http://twitter.com/#!/hintjens](http://twitter.com/#!/hintjens)  
- [http://zguide.zeromq.org/](http://zguide.zeromq.org/)  

**Code**

```
git clone git://github.com/zeromq/czmq.git
```

**Name**

czmq - high-level C binding for ØMQ

**Synopsis**

```
#include <czmq.h>

c `[flags]` `files` -lzmq -lczmq `[libraries]`
```

**Description**

**Scope and goals**  
czmq has these goals:  
- To wrap the ØMQ core API in semantics that are natural and lead to shorter, more readable applications.  
- To hide the differences between versions of ØMQ.  
- To provide a space for development of more sophisticated API semantics.
Introduction  nzmqt, a lightweight C++ Qt binding for ZeroMQ

nzmqt is a re-implementation of the approach taken by the zeromqt library.

The idea is to integrate ZeroMQ into the Qt event loop, mapping ZeroMQ message events onto Qt signals.

The original implementation also provides a Qt-like API which allows to represent messages as QByteArrayList instances.

While I took this idea and the original implementation as a source of information, I’ve done a completely new implementation. Not only in order to get rid of some short comings, but also because I wanted to be sure I can use the code in my projects without problems, because until now zeromqt’s author hasn’t officially released his work under a certain (open source) license. Consequently, nzmqt is released to the public under the simplified BSD license.

So you can use the code in your projects without any fear.

While zeromqt works fine for non-multi-part messages, it doesn’t support multi-part messages yet. Also, a lot of code duplicates code of ZeroMQ’s standard C++ binding. But this requires to take care of both implementations. So in contrast to the original implementation, nzmqt reuses as much code of ZeroMQ’s original C++ binding as possible by using inheritance.
Additionally, several things have been changed from a user’s perspective. In summary, nzmqt contains the following changes compared to zeromqt:

- The implementation is a complete re-write in the sense that it doesn’t duplicate code of ZeroMQ’s official C++ binding anymore. Instead, it builds upon existing code through inheritance and, hence, it will likely benefit from future bugfixes and enhancements targeted at ZeroMQ’s C++ binding.
- All classes are placed into a separate namespace ‘nzmqt’.
- This version now also supports ZeroMQ’s multi-part messages.
- The initial support for using Qt’s way of handling errors using error codes has been dropped. Instead, this code only throws exception originally thrown by ZeroMQ’s official C++ API. Note that although it looks like ‘ZMQException’ is a new custom exception class there is no custom exception class, but only a simple typedef which places the original exception class into the new namespace giving it a new name.
- As with ZeroMQ’s C++ binding all classes are contained within a single header file which makes integrating this Qt binding very easy.
- There is no ‘ZmqContext’ singleton anymore. Instead you can create your own ‘ZMQContext’ instance yourself.
- The socket class ‘ZMQSocket’ now also inherits from QObject, so you can add it as a child to any QObject parent as you know it from Qt.
- The code is officially licensed under the simplified BSD license.
- Not only PUB-SUB, but also REQ-REP and PUSH-PULL are supported. (Note that other socket combinations might work, but they haven’t been tested yet.)

Status  There are no known bugs.

Usage  As ZeroMQ’s C++ binding this Qt binding only consists of a single C++ header file which you need to include in your project.

Consequently, using ‘nzmqt’ in a Qt project is as simple as adding that single header file to your project’s .pro file as follows (assumed you use QT Creator):

HEADERS += nzmqt/nzmqt.hpp

If not already done, you also need to link against ZeroMQ library:

LIBS += -lzmq

Of course, you need to make sure the header file as well as the ZeroMQ library can be found by your compiler/linker. As nzmqt uses C++ exceptions for error handling so you will need to catch them by overriding QCoreApplication::notify() method. The included sample will show you how this can be done.

Included Samples  Currently, there are samples showing PUB-SUB, REQ-REP and PUSH-PULL protocol with multi-part messages in action. They also show how to deal with exceptions in Qt.

Java czmq library  See Also:

- http://twitter.com/#!/hintjens
- http://zguide.zeromq.org/
- http://www.zeromq.org/bindings:java
• http://www.zeromq.org/area:download

Code

http://github.com/zeromq/jzmq

C++11 zmq++ library

See Also:
• C++11
• http://www.250bpm.com/blog:4

Contents

• C++11 zmq++ library
  – History
  – Why should I have written ZeroMQ in C, not C++

History

From: niXman <i.nixman@gmail.com>
Date: 2012/2/13
Subject: [zeromq-dev] What do I need to do to create a subproject of the libzmq?
To: ZeroMQ development list <zeromq-dev@lists.zeromq.org>

Hello ØMQ dev-list!

I want to create a subproject of the ØMQ the main direction of which will be:

1. full implementation of the C++ API, not as cppzmq project, which is in fact a wrapper over C API.
2. use of exceptions.
3. the use of opportunities of C++11 such as: r-value references and move semantic.
4. the use of C++11 Thread support library: http://en.cppreference.com/w/cpp/thread
5. implementation of IOCP.
6. implementation of asynchronous handlers.
7. implementation of memory-pool.

I’m going to call this project: zmq++

Why should I have written ZeroMQ in C, not C++

See Also:
• http://www.250bpm.com/blog:4

Just to be clear from the very beginning: This is not going to be a Torvalds-ish rant against C++ from the point of view of die-hard C programmer.

I’ve been using C++ whole my professional career and it’s still my language of choice when doing most projects.

Naturally, when I started ZeroMQ project back in 2007, I’ve opted for C++. The main reasons were:

• Library of data structures and algorithms (STL) is part of the language. With C I would have to either depend on a 3rd party library or had to write basic algorithms of my own in 1970’s manner.

2.24. Programming
• C++ enforces some basic consistency in the coding style. For example, having the implicit ‘this’ parameter doesn’t allow to pass pointer to the object being worked on using several disparate mechanisms as it often happens to be the case with C projects. Same applies to explicit marking of member variables as private and many other features of the language.

• This point is actually a subset of the previous one, but it’s worth of explicit mention: Implementing virtual functions in C is pretty complex and tends to be slightly different for each class which makes understanding and managing the code a pain.

• And finally: Everybody loves destructors being invoked automatically at the end of the block.

Now, almost 5 years later, I would like to publicly admit that using C++ was a poor choice and explain why I believe it is so.

First, it’s important to take into account that ZeroMQ was intended to be a piece of infrastructure with continuous uptime.

If should never fail and never exhibit undefined behaviour. Thus, the error handling was of utmost importance. It had to be very explicit and unforgiving.

C++ exceptions just didn’t fill the bill. They are great for guaranteeing that program doesn’t fail — just wrap the main function in try/catch block and you can handle all the errors in a single place.

However, what’s great for avoiding straightforward failures becomes a nightmare when your goal is to guarantee that no undefined behaviour happens.

The decoupling between raising of the exception and handling it, that makes avoiding failures so easy in C++, makes it virtually impossible to guarantee that the program never runs into undefined behaviour.

Use cases

ØMQ? use cases

ØMQ? use case 1  Hi all,

I recently encountered 0MQ and am very intrigued by its capabilities. I have what I think is an unusual use-case and am hoping someone might have suggestions on how 0MQ could be part of a solution, or where I might run into trouble. (Who knows, maybe it will be a useful use case to think about for development purposes too?)

I have a number of hardware devices (up to perhaps 300) that sample and spit out data. Such data will come in on many different channels (e.g., TCP, UDP, bluetooth, WiFi, 802.11.4 radio, USB, A/V, maybe even RS232).

Maximally simplified, I need to (1) capture this data, (2) distribute the data to subscribed clients as rapidly as possible, and (3) allow clients to send command messages back to individual devices. At any given time, there may be up to 50 or so clients, ranging from being link-local (which should have essentially real-time performance) to being on the other side of the planet (where some delays are expected). The capture step will be handled in python, perhaps using a Twisted Python reactor setup.

The requirements/restrictions are as follows:

1. Robustness in the face of link failure or slow links (i.e., delay-tolerant networking, tear-away networking)
2. Quality of service capabilities (or at least an ability to easily prioritize, and re-prioritize, queued messages for sending)
3. Devices and clients may blink in and out of existence, as hardware is turned on/off; this shouldn’t affect the system at all
4. A few “key” clients—at least one per tear-away subnet—will require reliable receipt of device data. Such data will be stored in a local database. These clients are expected to remain up in the three-9s sense, but should also be easy to duplicate/replace. Loss of such a key client should not prevent the system from continuing to function.

5. Key clients will need reliable comm with other key clients, store-and-forward style, for database sync/aggregation.

6. Minimal (zero?) maintenance is important (i.e., auto-discovery of client type–key vs. non-key; zero-config service discovery on devices, self-monitoring daemons, etc) are strongly preferred as this will be deployed in a setting with minimal IT-resources.

7. Variable data rates and payload sizes (ranging from a single integer each hour to 1080p video)

#3 suggests a pub/sub design, while #4-5 suggest store-and-forward. #3 plus #6 make store-and-forward more difficult, if clients will appear/disappear and also need to be discovered and configured with minimal IT intervention.

I don’t think 0MQ (or, in fact, anything) can do all of this, but maybe it can do a lot. I’ve considered lower level comm (e.g., pub/sub via multicast), but that requires too much wheel-reinvention. I’ve also considered RabbitMQ, but that appears less friendly on config and expects links will be more up/stable than I am promised.

I briefly looked at bigger DDS options (openDDS, OpenSplice, RTI) but they seemed a combination of complete overkill in some areas and insufficient in others. Any suggestions on how to best utilize 0MQ in such a setting (with or without other packages)? Any (even random) thoughts are welcome. :-)

--best

Gary

Response from Steven McCoy

de Steven McCoy steven.mccoy@miru.hk

heure de l’expéditeur Envoyé à 05:12 (GMT-04:00). Heure locale : 03:34.
répondre à ZeroMQ development list <zeromq-dev@lists.zeromq.org>
à ZeroMQ development list <zeromq-dev@lists.zeromq.org>
date 21 juin 2011 05:12
objet Re: [zeromq-dev] Use case suggestions?

Note that The Guide™ has most of this covered already, certainly queuing and queue functionality is above the remit of basic ØMQ socket implementation. This is also all certainly possible with products like IBM WebSphere MQ, TIBCO Rendezvous, etc, but there are obviously downsides with each product to be considered, from expense to complexity. It is definitely a good idea to head towards a common middleware system that is flexible enough to be adapted to each situation.

There are a few things not listed, such as disconnected proxies, high data-loss aware transports, multi-pathing, and possibly congestion control, rate limiting not explicitly mentioned but implied that certainly limit your choices a bit. Note that you can integrate solutions like both ØMQ and RabbitMQ which may make disconnected operation significantly easier: https://github.com/rabbitmq/rmq-0mq

Also, ØMQ is already integrated with OpenPGM so you get a great TCP and multicast interchangeable pub/sub system for scaling to a significant number of recipients with options to target a variety of different deployments.

The only real question that springs to mind is in regard to performance requirements and pricing limitations, it is clear some queuing and synchronous storage looks required but it’s not obvious how far you wish to push beyond a basic queue system like SMTP or UUCP.

– Steve-o
Concurrent Programming

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Cat%C3%A9gorie:Programmation_concurrente
- http://zguide.zeromq.org/

Gevent

See Also:

- http://sdiehl.github.com/gevent-tutorial/
- http://software.schmorp.de/pkg/libev.html

Contents

- Gevent
  - Description
  - Tutorials

Description  gevent is a concurrency library based around libev. It provides a clean API for a variety of concurrency and network related tasks.

Tutorials  See Also:

- http://sdiehl.github.com/gevent-tutorial/

Interprocess communication
D-BUS Interprocess communication  See Also:


D-Bus is a message bus system, a simple way for applications to talk to one another.

In addition to interprocess communication, D-Bus helps coordinate process lifecycle; it makes it simple and reliable to code a “single instance” application or daemon, and to launch applications and daemons on demand when their services are needed.

D-Bus supplies both a system daemon (for events such as “new hardware device added” or “printer queue changed”) and a per-user-login-session daemon (for general IPC needs among user applications). Also, the message bus is built on top of a general one-to-one message passing framework, which can be used by any two apps to communicate directly (without going through the message bus daemon).

Currently the communicating applications are on one computer, or through unencrypted TCP/IP suitable for use behind a firewall with shared NFS home directories. (Help wanted with better remote transports - the transport mechanism is well-abstracted and extensible.)

**DBUS coding examples**

**python DBUS examples**  See Also:

- [QT DBUS examples](http://www.freedesktop.org/wiki/Software/dbus)

2.24. Programming 903
You can use D-Bus bindings and listen to DeviceAdded and DeviceRemoved signals. You will have to check the capabilities of the Added device in order to select the storage devices only.

Here is a small example, you can remove the comments and try it:

```python
import dbus
import gobject

class DeviceAddedListener:
    def __init__(self):
        self.bus = dbus.SystemBus()
        self.hal_manager_obj = self.bus.get_object(
            "org.freedesktop.Hal",
            "/org/freedesktop/Hal/Manager")
        self.hal_manager = dbus.Interface(self.hal_manager_obj, 
            "org.freedesktop.Hal.Manager")

        self.hal_manager.connect_to_signal("DeviceAdded", self._filter)

    def _filter(self, udi):
        device_obj = self.bus.get_object("org.freedesktop.Hal", udi)
        device = dbus.Interface(device_obj, "org.freedesktop.Hal.Device")
        if device.QueryCapability("volume"):
            return self.do_something(device)

    def do_something(self, volume):
        device_file = volume.GetProperty("block.device")
        label = volume.GetProperty("volume.label")
        fstype = volume.GetProperty("volume.fstype")
        mounted = volume.GetProperty("volume.is_mounted")
        mount_point = volume.GetProperty("volume.mount_point")

        try:
            size = volume.GetProperty("volume.size")
        except:
            size = 0

        print "New storage device detected:",
        print " device_file: $s" % device_file
        print " label: $s" % label
        print " fstype: $s" % fstype
        if mounted:
            print " mount_point: $s" % mount_point
        else:
```

print " not mounted"

print " size: \$s (\%.2fGB)" % (size, float(size) / 1024**3)

if __name__ == '__main__':
    from dbus.mainloop.glib import DBusGMainLoop
    DBusGMainLoop(set_as_default=True)
    loop = gobject.MainLoop()
    DeviceAddedListener()
    loop.run()


import pyudev
context = pyudev.Context()
monitor = pyudev.Monitor.from_netlink(context)
observer = pyudev.pygtk.GUDevMonitorObserver(monitor)
observer.connect('device-added', device_added_callback)
observer.connect('device-changed', device_changed_callback)
monitor.enable_receiving()
mainloop = gobject.MainLoop()
mainloop.run()

This is what I use to list already plugged flash sticks. You can modify script to your needs:

import dbus
bus = dbus.SystemBus()

proxy = bus.get_object("org.freedesktop.Hal", "/org/freedesktop/Hal/Manager")
iface = dbus.Interface(proxy, "org.freedesktop.Hal.Manager")

devices = iface.GetAllDevices()

for device in devices:
    try:
        proxy1 = bus.get_object("org.freedesktop.Hal", device)
        iface1 = dbus.Interface(proxy1, "org.freedesktop.Hal.Device")
        props = iface1.GetAllProperties()

        removable = iface1.GetProperty("storage.removable")
        usb = iface1.GetProperty("storage.bus")

        if usb == "usb":
            print \n".join({"%s: %s" % (k, props[k]) for k in props}) # shows available properties
    except:
        pass

And this is what I use to see if any usb plugged:

#!/usr/bin/python
import dbus
import gobject

class DeviceAddedListener:
    def __init__(self):

2.24. Programming 905
self.bus = dbus.SystemBus()
self.hal_manager_obj = self.bus.get_object("org.freedesktop.Hal", "/org/freedesktop/Hal/Manager")
self.hal_manager = dbus.Interface(self.hal_manager_obj, "org.freedesktop.Hal.Manager")
self.hal_manager.connect_to_signal("DeviceAdded", self._filter)

def _filter(self, udi):
    device_obj = self.bus.get_object("org.freedesktop.Hal", udi)
    device = dbus.Interface(device_obj, "org.freedesktop.Hal.Device")
    self.do_something(device)

def do_something(self, device):
    try:
        usb = device.GetProperty("storage.bus")
        info = device.GetProperty("info.product")
        removable = device.GetProperty("storage.removable")
        print info
    except:
        pass
        # blah blah

if __name__ == '__main__':
    from dbus.mainloop.glib import DBusGMainLoop
    DBusGMainLoop(set_as_default=True)
    loop = gobject.MainLoop()
    DeviceAddedListener()
    loop.run()

QT DBUS examples  See Also:

• python DBUS examples

http://www.qtcentre.org/threads/18311-Hot-plug-and-button-event  See Also:

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent), ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    if (!QDBusConnection::systemBus().isConnected()) {
        qDebug() << "Cannot connect to system bus";
    }

    bool connected = QDBusConnection::systemBus().connect(
        "org.freedesktop.Hal",
        "/org/freedesktop/Hal/Manager",
        "org.freedesktop.Hal.Manager",
        "DeviceAdded",
        this, SLOT(deviceAdded()));
}

void MainWindow::deviceAdded()
{
    qDebug() << "Device added";
}
QDBusConnection::systemBus().connect("org.freedesktop.Hal",
"/org/freedesktop/Hal/Manager",
"org.freedesktop.Hal.Manager",
"DeviceAdded",
this, SLOT(deviceAdded()));

QDBusConnection::systemBus().connect("org.freedesktop.Hal",
"/org/freedesktop/Hal/Manager",
"org.freedesktop.Hal.Manager",
"DeviceRemoved",
this, SLOT(deviceRemoved()));

void deviceAdded(){
    qDebug() << "device added!";
}

void deviceRemoved(){
    qDebug() << "device removed!";
}

DBUS tools  See Also:

• http://blog.fpmurphy.com/2011/08/introduction-to-udisks.html

dbus-monitor

dbus-monitor --system

signal sender="org.freedesktop.DBus" -> dest=:1.58 serial=2 path="/org/freedesktop/DBus"; interface="org.freedesktop.DBus"
string ":1.58"
signal sender=1.7 -> dest=(null destination) serial=9 path="/org/freedesktop/Hal/Manager"; interface="org.freedesktop.Hal.Manager"
string "/org/freedesktop/Hal/devices/usb_device_b81_200_01699449"
signal sender=1.7 -> dest=(null destination) serial=10 path="/org/freedesktop/Hal/Manager"; interface="org.freedesktop.Hal.Manager"
string "/org/freedesktop/Hal/devices/usb_device_b81_200_01699449_if0"
signal sender=1.7 -> dest=(null destination) serial=11 path="/org/freedesktop/Hal/Manager"; interface="org.freedesktop.Hal.Manager"
string "/org/freedesktop/Hal/devices/usb_device_b81_200_01699449_if0"
signal sender=1.7 -> dest=(null destination) serial=12 path="/org/freedesktop/Hal/Manager"; interface="org.freedesktop.Hal.Manager"
string "/org/freedesktop/Hal/devices/usb_device_b81_200_01699449"

dbus-monitor --session

dbus-send

D-Feet graphical D-Bus debugging tool  See Also:

• http://live.gnome.org/DFeet/

Description  D-Feet is a D-Bus debugger written in PyGtk+ by John Palmieri.
Design  D-Feet needs your help. The current design was a quick one off that doesn’t really fit any model other than showing off the internals of a D-Bus hierarchy. It is not optimised for actual debugging workflows. While it is a useful tool it can be much better. I am requesting that anyone who has a D-Bus development workflow to write up their day to day usage of D-Bus debugging tools.

Features  Current Features

- View names on any bus
- View exported objects, interfaces, methods and signals
- View the full command line of services on the bus
- Execute methods with parameters on the bus and see their return values

udisks (1)  See Also:

- http://www.freedesktop.org/wiki/Software/udisks
- udisks-tcp-bridge
- http://library.gnome.org/users/palimpsest/

Udisks (formerly called DeviceKit-disks) is a replacement for part of the functionality which used be provided by the now deprecated HAL (Hardware Abstraction Layer). Essentially it is an abstraction for enumerating disk and storage devices and performing operations on them.

Udisks provides:

- A daemon (udisks-daemon) that implements well-defined D-Bus interfaces that can be used to query and manipulate disk and storage devices.
- A command-line tool (udisks), that can be used to query and use the daemon.

```
ps -ef | grep udisks

udisks --help

udisks --enumerate

udisks --monitor
```

udisks2  See Also:

http://people.freedesktop.org/~david/udisks2-20110628/index.html

DBUS_SESSION_BUS_ADDRESS

```
$ env | grep DBUS_SESSION_BUS_ADDRESS
```

DBUS and smartcard
Hello everyone,

a while ago, there was a discussion in xdg [1] about DBus support for smartcard activity. I’ve hacked together a bash script for that purpose:

```
#!/bin/bash
pcsc_scan -n | { while read RESULT; do
    ATR=$(echo $RESULT | perl grep "ATR: ..35m(\.*\)\e")
    [ "$ATR" ] && dbus-send --session --type=signal /org/debian/alioth/pcsclite/reader0/slot0 org.debian.alioth.pcsclite.CardPresence string:"$ATR"
    RMV=$(echo $RESULT | grep "removed")
    done }
```

However, I think it would be a bit smoother to directly integrate this into pcsc-lite. The best to add support would be in eventhandler.c, correct?

Florian


Concurrent multithread Programming

See Also:
- https://secure.wikimedia.org/wikipedia/fr/wiki/Thread_%28informatique%29

C++ thread libraries

Using QThread  See Also:
- Qt C++ nzmqt library

We use IRC extensively to communicate amongst ourselves as well as with the community. I hang out on the #qt channel on the Freenode network and help people with questions when I can.

A common question that I see (and that makes me cringe at the same time) has to do with understanding threading with Qt and how to make some code they’ve written work.

People show their code, or examples based on their code, and I often end up thinking:
You’re doing it wrong

I know this is a bit of a bold thing to say, perhaps a bit provocative, but at the same time, I can’t help but think that the (hypothetical) class below is an incorrect application of object-oriented principles as well as incorrect usage of Qt.

class MyThread : public QThread
{
public:
    MyThread()
    {
        moveToThread(this);
    }  
    void run();

signals:
    void progress(int);
    void dataReady(QByteArray);

public slots:
    void doWork();
    void timeoutHandler();
};

My #1 biggest gripe with this code is moveToThread(this); I see so many people using this without understanding what it does.

What does it do, you ask? The moveToThread() function tells Qt to ensure that event handlers, and by extension signals and slots, are called from the specified thread context. QThread is the thread interface, so we’re telling the thread to run code “in itself”.

We’re also doing this before the thread is running as well. Even though this seems to work, it’s confusing, and not how QThread was designed to be used (all of the functions in QThread were written and intended to be called from the creating thread, not the thread that QThread starts).

My impression is that moveToThread(this); creeps into people’s code because they saw some blog somewhere that used it. A quick web search turns up several of these blogs, all of which follow the pattern in the class above:

• subclass QThread
• add signals and slots to do work
• test code, see that the slots aren’t called “from the right thread”
• ask google, find moveToThread(this); and comments that “it seems to work when I add this”

In my opinion, the problems started at step 1. QThread was designed and is intended to be used as an interface or a control point to an operating system thread, not as a place to put code that you want to run in a thread.

We object-oriented programmers subclass because we want to extend or specialize the base class functionality. The only valid reasons I can think of for subclassing QThread is to add functionality that QThread doesn’t have, e.g. perhaps providing a pointer to memory to use as the thread’s stack, or possibly adding real-time interfaces/support. Code to download a file, or to query a database, or to do any other kind of processing should not be added to a subclass of QThread; it should be encapsulated in an object of it’s own.

Usually, this means simply changing your class to inherit from QObject instead of QThread and, possibly, changing the class name. QThread has a started() signal that you can connect to when you need to perform some initialization.

To actually have your code run in the new thread context, you need to instantiate a QThread and assign your object to that thread using the moveToThread() function. Even though you are still using moveToThread() to tell Qt to run your
code in a specific thread context, we are keeping the thread interface separate. If necessary, it is now possible to have multiple instances of your class assigned to a single thread, or multiple instances of many different classes assigned to a single thread. In other words, it’s unnecessary to tie a single instance of a class to a single thread.

I take much of the blame for the confusion that comes with writing threaded Qt code. The original QThread class was abstract, so subclassing was necessary. It wasn’t until Qt 4.4 that QThread::run() gained a default implementation.

Previously, the only way to use QThread was to subclass. With the addition of thread affinity and support for signal and slot connections between objects of different affinity, suddenly we have a convenient way of working with threads.

We like convenience, we want to use it. Unfortunately, I realized to late that forcing people to subclass QThread actually made it harder than it needed to be.

I also take the blame for not getting up-to-date examples and documentation made to show people how to get the convenience with the minimum amount of headaches.

For now, the best resource I can point at is a blog I wrote several years ago.

Disclaimer: everything you see above is of course opinion. I’ve worked a lot on these classes, and have a fairly clear idea of how to use them and how not to use them.

**Threading without the headache with QThread**

**See Also:**

- [Qt C++ nzmqt library](http://labs.trolltech.com/blogs/2006/12/04/threading-without-the-headache/)

A couple of weeks ago, I started trying to find out if a pure virtual function can be made impure without breaking binary compatibility.

“Why?” you ask? Because I want to make QThread::run() call QThread::exec() by default. We all know that threading is difficult to do, mostly because of the need to lock data, synchronize threads (with a QWaitCondition or QSemaphore, for example). However, it doesn’t have to be.

Consider the following code snippet:

```cpp
// create the producer and consumer and plug them together
Producer producer;
Consumer consumer;
producer.connect(&consumer, SIGNAL(consumed()), SLOT(produce()));
consumer.connect(&producer, SIGNAL(produced(QByteArray *)), SLOT(consume(QByteArray *)));

// they both get their own thread
QThread producerThread;
producer.moveToThread(&producerThread);
QThread consumerThread;
consumer.moveToThread(&consumerThread);

// go!
producerThread.start();
consumerThread.start();
```

Wouldn’t life be wonderful if it were that easy? The good news is, it already is, if you do just a small amount of work: subclass QThread and reimplement run() to simply call exec(). The code snippet above comes from an example [1] where I’ve done this.

The end result is a solution to the Producer/Consumer problem without the need for a lock, a condition variable, or a semaphore. Hell, I don’t even have to write a thread. I can do everything in a nice object oriented way; the Producer code goes in one place, the Consumer code in another, and then I move these into the wonderful black boxed thread object that does what I expect.
So what’s the point of all this? I hope to be able to answer my original question: can a pure virtual function be made impure without breaking binary compatibility? If so, I hope to make QThread::run() call exec() in Qt 4.3.

- [http://chaos.troll.no/~bhughes/producerconsumer.tar.gz](http://chaos.troll.no/~bhughes/producerconsumer.tar.gz)

**Python Multithreading libraries**

**Python concurrent.futures multithreading library**  
See Also:

- [http://www.dalkescientific.com/writings/diary/archive/2012/01/19/concurrent.futures.html](http://www.dalkescientific.com/writings/diary/archive/2012/01/19/concurrent.futures.html)
- [PEP 3148 – futures - execute computations asynchronously](http://www.dalkescientific.com/writings/diary/archive/2012/01/19/concurrent.futures.html)

**Design patterns**

**Active Object Design pattern**

See Also:

- [http://sites.google.com/site/kjellhedstrom2/active-object-with-cpp0x](http://sites.google.com/site/kjellhedstrom2/active-object-with-cpp0x)

**Javascript Design patterns**

**Freebook**  
See Also:


In this book we will explore applying both classical and modern design patterns to the JavaScript programming language.

**Observer Design pattern**

**C Observer Design pattern**

**Example_1 C Observer Design pattern**

**The subject**

```csharp
using System;
using System.IO;
using System.Collections.Generic;

namespace VideoCell
{
    public interface IObservateurLog
    {
        void UpdateLog(string text);
    }

    public static class Log
    {
        static IList<IObservateurLog> observers = new List<IObservateurLog>();
    }
}
```
public static void Add(IObservateurLog observer) {
    observers.Add(observer);
}

public static void Remove(IObservateurLog observer) {
    observers.Remove(observer);
}

static void Notifie(string text) {
    foreach (IObservateurLog observer in observers) {
        observer.UpdateLog(text);
    }
}

static bool isOpen = false;
static StreamWriter writerLog;

/// <summary>
/// Open a log file.
/// </summary>
public static void Open() {
    if (isOpen) {
        // already open
        return;
    }
    string filename = "C:\VideoCell.log";
    try {
        // false => pas de concaténation
        writerLog = new StreamWriter(filename, false);
        isOpen = true;
    }
    catch (Exception ex) {
        isOpen = false;
        throw new Exception("Cant open log file", ex);
    }
}

/// <summary>
/// Close a log file.
/// </summary>
public static void Close() {
    if (isOpen) {
        Write("close Log()");
        writerLog.Flush();
        writerLog.Close();
        isOpen = false;
    }
}
public static void Flush()
{
    if (!isOpen)
    {
        return;
    }
    string datenow_milli = DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss.fff");
    return datenow_milli;
}

public static string GetTimeMilliseconds()
{
    string datenow_milli = DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss.fff");
    return datenow_milli;
}

public static string GetTimeMicroseconds()
{
    string datenow_micro = DateTime.Now.ToString("yyyy-MM-dd HH:mm:ss.ffffff");
    return datenow_micro;
}

public static void Write(string text)
{
    if (!isOpen)
    {
        return;
    }
    string datenow = GetTimeMicroseconds();
    string str_message = String.Format("{0} {1}\n", datenow, text);
    writerLog.Write(str_message);
    writerLog.Flush();
}

The observer
namespace VideoCell
{
    public partial class VideoCellForm : Form, IObservateurLog
    {
        ...

        public void UpdateLog(string str_message)
        {
            textBox_logs.AppendText(str_message);
        }

        ...

        void OpenApplication()
        {
            Log.Open();
            Log.Add(this);
            Log.Write("Open VideoCellApplication");
        }

        ...
    }
} // namespace VideoCell

Errors

Contents

- Errors
  - Go errors treatment

Go errors treatment

See Also:

Parser

See Also:
- http://www.matthieuamiguet.ch/pages/compilateurs

In computing, a parser is one of the components in an interpreter or compiler, which checks for correct syntax and builds a data structure (often some kind of parse tree, abstract syntax tree or other hierarchical structure) implicit in the input tokens. The parser often uses a separate lexical analyser to create tokens from the sequence of input characters. Parsers may be programmed by hand or may be (semi-)automatically generated (in some programming languages) by a tool.
Python parsers

Python PLY Parser  See Also:
- http://www.dabeaz.com/ply/
- http://www.matthieuamiguet.ch/pages/compilateurs

Welcome to the PLY homepage. PLY is an implementation of lex and yacc parsing tools for Python. If you don’t have the slightest idea what that means, you’re probably in the wrong place. Otherwise, keep reading.

Cours de Mathieu Miguet  See Also:
http://www.matthieuamiguet.ch/pages/compilateurs

La contribution la plus originale de cours consiste en une série de tutoriaux sur la conception d’un compilateur avec le package python PLY. Ce tutoriel peut probablement être utile hors du cadre de ces cours, puisqu’il s’agit de l’une des rares sources d’information en français sur l’utilisation de PLY.

Sagradaproject  See Also:

The bottom line is that web services can be described in a Domain Specific Language (DSL) that can be used to generate the documentation automatically (and it stays accurate and up to date) but also to set up in the server things we usually do on the code side: the request dispatching.

And if we hide the DSL behind a nice user interface developers can use to build their apps, that’s even better. That’s what’s happening in the screencast I’ve recorded. The forms generate portions of DSL.

A web service DSL  Back to our DSL. When you write a web service, it’s always the same story no matter what framework you’re using, you’re basically doing these steps (I am over-simplying for now):

Define a route for your service on the server
Build the response
Send back the response

These steps can be described in a simple DSL.

Here’s a basic example:

```
path hello_world {
  description "Simplest application: Hello World!",
  method GET,
  url /hello,
  use python:somemodule.hello
};
```

With a function hello located in somemodule that can look like this:

```
def hello(request):
    return ’Hello World’
```

The application in that case is composed of:

- a DSL file
• a Python file with a single function

It’s easy from there with the proper DSL parser to:
• deploy those two files in our infrastructure
• run the app
• provide auto-generated documentation for the service

Architecture

The prototype I’ve written for the demo does the following:
• The tool is a web application that provides forms to create Service Containers
• Each Service Container has a unique root on the server
• In each container you can add web services.
• For each container, the DSL is built on the fly, then the corresponding AST is kept in memory. The web UI allow users to modify it on the fly
• When a request comes in, it’s rooted to the right Service Container depending on the beginning of the path, then the AST is used to find out what function(s) should be used. The function is then executed in a sandbox.

The prototype also provides a command-line tool to start a server by loading an arbitrary DSL file.

The DSL Parser  I will not go in to great details here, you can look at my previous posts mentioning RedBarrel.
The current DSL is implemented with PLY and can be found here.
download parser.py

Regular expressions

See Also:
• http://fr.wikipedia.org/wiki/Expressions_r%C3%A9guli%C3%A8res
• http://en.wikipedia.org/wiki/Regular_expression

Une expression rationnelle ou expression régulière1 est en informatique une chaîne de caractères que l’on appelle parfois un motif et qui décrit un ensemble de chaînes de caractères possibles selon une syntaxe précise.
Les expressions rationnelles sont issues des théories mathématiques des langages formels des années 1940. Leur puissance à décrire des ensembles réguliers explique qu’elles se retrouvent dans plusieurs domaines scientifiques dans les années d’après-guerre et justifie leur adoption en informatique.
Les expressions rationnelles sont aujourd’hui utilisées par les informaticiens dans l’édition et le contrôle de texte ainsi que dans la manipulation des langues formelles que sont les langages de l’informatique.

Exemples

Regular expressions examples
parses Apache logs  See Also:


How much do I love Python? Consider this little snippet that parses Apache logs.

```python
import re
from collections import defaultdict, namedtuple

format_pat= re.compile(r"(?P<host>\[[d\.\]+\])\s" r"(?P<identity>\S*)\s" r"(?P<user>\S*)\s" r"\[(?P<time>.*?)\]\s" r""(?P<request>.*?)^\s" r"(?P<status>\d+)\s" r"(?P<bytes>\S*)\s" r""(?P<referer>.*?)^\s" # [SIC]
 r""(?P<user_agent>.*?)^\s"")

Access = namedtuple('Access', ['host', 'identity', 'user', 'time', 'request', 'status', 'bytes', 'referer', 'user_agent'] )

def access_iter( source_iter ):
    for log in source_iter:
        for line in (l.rstrip() for l in log):
            match= format_pat.match(line)
            if match:
                yield Access( **match.groupdict() )
```

That’s about it. The access log rows are now first-class Access-class objects that can be processed pleasantly by high-level Python applications.

**Cool things**

- The adjacent string concatenation means that the regular expression can be broken up into bits to make it readable.
- When the named tuple attributes match the regular expression names, we can trivially turn the match.groupdict() into a named tuple.
- By using a generator, the other parts of the application can simply loop through the results without tying up memory to create vast intermediate structures.

A couple of years back, a sysadmin was trying to justify spending money on a log analyzer product. I suggested they (at the very least) get an open source log analyzer.

I also suggested that they learn Python and save themselves the pain of working with a (potentially) complex tool. Given this as a common library module, log analysis applications are remarkably easy to write.

**Software bus**

See Also:

Un bus logiciel est un composant de connexion logicielle, utilisé par des Middlewares comme ORB.

Aux bus orientés Données, d’architecture et conception plus classique utilisés pour transmettre directement des données, inspirés des Bus informatiques hardware, se sont ajoutés les bus orientés Service, mis en œuvre par l’Architecture orientée services, dans lequel transitent des objets plus complexes.

dbus

See Also:
- *D-BUS Interprocess communication*
- [http://www.freedesktop.org/wiki/IntroductionToDBus](http://www.freedesktop.org/wiki/IntroductionToDBus)

En informatique, D-Bus est un système de communication inter-processus simple permettant aux logiciels de communiquer entre eux.

Hautement influencé par le système DCOP implémenté dans KDE 2 et KDE 3, il l’a remplacé dans KDE 4.

Red Hat est le développeur principal de D-Bus, en tant qu’élément du projet freedesktop.org. [Freedesktop.org](http://freedesktop.org) diffuse D-Bus sous les termes de la licence publique générale GNU et la Licence Académique Libre en tant que logiciel libre.

**DBUS bindings**  D-Bus bindings are available for an increasing number of languages. There is a low-level C binding, but that is probably too detailed and cumbersome for anything but writing other bindings. A more practical C binding is based on [GLib](http://www.gtk.org). There are also Java, Perl and Python bindings (about we also have notes and examples), and so on.

**API for using D-BUS with GLib**  See Also:  [http://dbus.freedesktop.org/doc/dbus-glib/index.html](http://dbus.freedesktop.org/doc/dbus-glib/index.html)

libdbus proper is a low-level API, these GLib bindings wrap libdbus with a much higher-level approach. The higher level approach is possible because GLib defines a main loop, an object/type system, and an out-of-memory handling policy (it exits the program). See [http://www.gtk.org](http://www.gtk.org) for GLib information.

ivy

See Also:  [http://www2.tls.cena.fr/products/ivy/](http://www2.tls.cena.fr/products/ivy/)

Ivy is a simple protocol and a set of open-source (LGPL) libraries and programs that allows applications to broadcast information through text messages, with a subscription mechanism based on regular expressions.

Ivy libraries are available in C, C++, Java, Python and Perl, on Windows and Unix boxes and on Macs. Several Ivy utilities and hardware drivers are available too.

Ivy is currently used in research projects in the air traffic control and human-computer interaction research communities as well as in commercial products. It is also taught to CS students.
A software bus See Also:
http://www2.tls.cena.fr/products/ivy/about.shtml

A software bus is a system that allows software applications to exchange information with the illusion of broadcasting that information, selection being performed by the receiving applications based on the contents of the messages. Other denominations are sometimes publish-subscribe notification services or message-oriented middleware.

Software buses federate pieces of software written in different programming languages on different platforms.

The main features of Ivy Ivy is not based on a centralised server. Actually, Ivy is mostly a communication convention, implemented through a collection of libraries for various languages and platforms. The current version of the Ivy protocol is version 3, which has been stable for the last 3 years. Language bindings are available in C (Unix and Windows), C++ (Mac, Unix, Windows), Java and Perl. There have been successful uses through the C library.

Messages are formatted in text, and subscriptions are based on regular expressions. Plans to move to an XML-based subscription language are on their way.

From the programmer’s point of view, Ivy is an information broadcasting channel. The main functions are:

- connecting to a bus. Example: IvyInit (b, “192.126:2011”)
- sending a message. Example: IvySend (b, “HELLO %s”, world)
- binding a message pattern to a callback function. Example: IvyBind (b, “HELLO (.*)”, cb)
- the main loop. Example : IvyLoop ()

Subscriptions are managed on the emitter’s side, which limits the actual network traffic. Direct point-to-point messages are also available.

Ivy was designed by a research group in Human-Computer Interaction, with the goals of connecting applications written on different toolkits/languages/platforms (such as an OpenGL application on a SGI connected to a PerlTk application on a Linux box), while keeping it simple: no server to be launched and supervised, a simplistic API, and a communication model compatible with classical event-based GUI programming. We think we have somewhat reached our goal.

Latest news

ejanuary 17th, 2011
- ivy-c (and other libraries based on it) is now compatible with IPv6
- ivy-c++ is now compatible with Qt mainloop
- Inventor node for coin3d 3D rendering library

Ivy Applications See Also:
http://www2.tls.cena.fr/products/ivy/download/applications.html

ivy libraries See Also:
http://www2.tls.cena.fr/products/ivy/documentation/
ivy C library  See Also:
http://www2.tls.cena.fr/products/ivy/documentation/ivy-c/index.html

The Ivy C library (aka Ivy-C or ivy-c) is a C library that allows you to connect applications to an Ivy bus. You can use it to write applications in C or any other language that supports C extensions.

You can also use it to integrate an application that already has a main loop (such as a GUI application) within an Ivy bus. This guide is here to help you do that.

Basic functions  See Also:
http://www2.tls.cena.fr/products/ivy/documentation/ivy-c/x76.html

Initialization and main loop (IvyMainLoop)  The Ivy C library provides its own main loop: IvyMainLoop. You should use it unless you already use a toolkit that provides its own main loop and you want to use that one. If it is the case, please refer to section XX. Otherwise, just call IvyMainLoop. From within the main loop, you can call IvyStop to exit the loop.

Using Ivy with another main loop  See Also:
http://www2.tls.cena.fr/products/ivy/documentation/ivy-c/x200.html

The ivyprobe source code holds examples of use of Ivy within other main loops, namely Xt and Gtk.

Adding Ivy to another main loop  You can decide to use the main loop from another toolkit than the X Toolkit or the Tk toolkit. If you do that, you’ll have to define four functions that Ivy will use to get its own channels managed by the other toolkit. You should link ivy with your new module instead of the ivy(xxx)loop module.

These functions are declared in ivychannel.h:
- IvyChannelInit
- IvyChannelStop
- IvyChannelAdd
- IvyChannelRemove

ivy C++ library  See Also:

- http://www2.tls.cena.fr/products/ivy/download/packages/IvyProbe.msi

ivy Java library  See Also:

- http://www2.tls.cena.fr/products/ivy/documentation
- http://www2.tls.cena.fr/products/ivy/download/packages/IvyGUIMonitor.jar

Ivy and swing GUI  See Also:

http://www2.tls.cena.fr/products/ivy/download/packages/IvyGUIMonitor.jar

Swing requires the code to run in the main swing thread. In order to avoid problems, be sure to use the SwingUtilities.invokeLater() or SwingUtilities.invokeLaterAndWait() methods if you Ivy callbacks impact swing components.
Java sources

svn co http://svn.tls.cena.fr/svn/ivy/ivy-java/trunk

ivy Python library See Also:

- http://www2.tls.cena.fr/products/ivy/documentation
- http://www2.tls.cena.fr/products/ivy/documentation/ivy-python/2.1/
- http://svn.tls.cena.fr/wsvn/ivy/?sc=0

Introduction Ivy is a lightweight software bus for quick-prototyping protocols. It allows applications to broadcast information through text messages, with a subscription mechanism based on regular expressions.

If you’re used to the standard Ivy API, you probably want to look at the std_api module.

The Ivy software bus was designed at the French Centre d’Etudes de la Navigation Aerienne (CENA). The original work: software, documentation, papers, credits can be found on the Ivy Home Page; it contains all the necessary materials needed to understand Ivy.

This package is the Python library; Ivy libraries are also available for numerous languages, among wich: C, C#, C++, Java, Perl –see the Ivy Download Page for details.

This python library is a full rewrite of the original software, and it is written in pure python. Note that another python implementation is available at the Ivy downloads page, which is built by SWIG on top of the Ivy C library.

state machine

Qt state machine

See Also:


Qt Quick transition state machine

Intro to Qt Quick - Part II See Also:


This webcast presents an introduction into the specific functions of Qt Quick.

You will learn how to manage input events, control the various states and transitions between states, and how to do animations and arrangements of images and graphical elements using the Qt Quick Framework.

Python fsm state machine

See Also:

- https://code.google.com/p/python-fsm/
Programming tools

kdiff3 tool

See Also:

http://kdiff3.sourceforge.net

Source

svn co https://kdiff3.svn.sourceforge.net/svnroot/kdiff3 kdiff3

Time

See Also:

•

Time applications

Django 1.4  See Also:

• https://docs.djangoproject.com/en/dev/topics/i18n/timezones/#time-zones-faq

The biggest new feature in Django 1.4 is support for time zones when handling date/times. When enabled, this Django will store date/times in UTC, use timezone-aware objects internally, and translate them to users’ local timezones for display.

If you’re upgrading an existing project to Django 1.4, switching to the time-zone aware mode may take some care: the new mode disallows some rather sloppy behavior that used to be accepted.

We encourage anyone who’s upgrading to check out the timezone migration guide and the timezone FAQ for useful pointers.

Support for time zones  See Also:

http://fr.wikipedia.org/wiki/UTC

In previous versions, Django used “naive” date/times (that is, date/times without an associated time zone), leaving it up to each developer to interpret what a given date/time “really means”.

This can cause all sorts of subtle timezone-related bugs.

In Django 1.4, you can now switch Django into a more correct, time-zone aware mode. In this mode, Django stores date and time information in UTC in the database, uses time-zone-aware datetime objects internally and translates them to the end user’s time zone in templates and forms.

Reasons for using this feature include:

• Customizing date and time display for users around the world.

• Storing datetimes in UTC for database portability and interoperability. (This argument doesn’t apply to PostgreSQL, because it already stores timestamps with time zone information in Django 1.3.)

• Avoiding data corruption problems around DST transitions.
Time zone support is enabled by default in new projects created with startproject.
If you want to use this feature in an existing project, read the migration guide. If you encounter problems, there’s a helpful FAQ.

**taskwarrior**  See Also:

- [http://taskwarrior.org/projects/show/taskwarrior](http://taskwarrior.org/projects/show/taskwarrior)
- [http://pypi.python.org/pypi/bugwarrior/0.3.1](http://pypi.python.org/pypi/bugwarrior/0.3.1)

Taskwarrior is a command-line todo list manager.
It maintains a list of tasks that you want to do, allowing you to add/remove, and otherwise manipulate them.
Task has a rich list of subcommands that allow you to do sophisticated things with it.
You’ll find it has customizable reports, charts, GTD features, Lua extensions, device synching and more.
Taskwarrior is a very active project involving people around the globe - check often for updates.

**Time libraries**

**Time Period library for NET**  See Also:


**Python Time**

See Also:

- [http://www.python.org/dev/peps/pep-0418/](http://www.python.org/dev/peps/pep-0418/)

**Contents**

- Python Time
  - Abstract
  - Use cases
  - Definitions
    * Monotonic

**Abstract**  Add time.monotonic() and time.get_clock_info(name) functions to Python 3.3.

**Use cases**

- Display the current time to a human (e.g. display a calendar or draw a wall clock): use system clock, i.e. time.time() or datetime.datetime.now().
- Event scheduler, timeout: time.monotonic().
- Benchmark, profiling: time.clock() on Windows, time.monotonic(), or fallback to time.time()
Monotonic  A monotonic clock cannot go backward. It may give the same value for two close reads depending on the clock resolution.

On Linux, CLOCK_MONOTONIC is a monotonic clock but its rate is adjusted by NTP.

Time theory

See Also:

- http://cr.yp.to/libtai/tai64.html#tai64n

Contents

- Time theory
  - UTC
    * Description
  - Semaines

UTC  See Also:


Le Temps universel coordonné (UTC) est une échelle de temps adoptée comme base du temps civil international par la majorité des pays du globe.

Description  See Also:


UTC est une échelle de temps comprise entre le Temps atomique international (TAI ; stable mais déconnecté de la rotation de la Terre) et le Temps universel (TU), directement lié à la rotation de la Terre et donc lentement variable.

Le terme « coordonné » indique que le Temps universel coordonné est en fait identique au Temps atomique international (il en a la stabilité et l’exactitude) à un nombre entier de secondes près, ce qui lui permet de coller au Temps universel à moins de 0,9 s près.

« Coordinated universal time » a été abrégé en « UTC », au lieu de « CUT » correspondant à l’acronyme en anglais ou de « TUC » correspondant à l’acronyme en français. En effet, si les experts de l’UIT étaient d’accord pour définir une abréviation commune à toutes les langues, ils étaient dividés sur le choix de la langue.

Finalement, c’est le compromis UTC, nécessitant un effort des deux parties, qui fut choisi. C’est cette notation qui est utilisée par la norme ISO 8601.

Semaines  See Also:

http://fr.wikipedia.org/wiki/Num%C3%A9rotation_ISO_des_semaines

2.25 Security

2.25.1 Computer security

See Also:
Computer security is a branch of computer technology known as information security as applied to computers and networks. The objective of computer security includes protection of information and property from theft, corruption, or natural disaster, while allowing the information and property to remain accessible and productive to its intended users. The term computer system security means the collective processes and mechanisms by which sensitive and valuable information and services are protected from publication, tampering or collapse by unauthorized activities or untrustworthy individuals and unplanned events respectively. The strategies and methodologies of computer security often differ from most other computer technologies because of its somewhat eluding objective of preventing unwanted computer behavior instead of enabling wanted computer behavior.

**Antivirus software**

See Also:

Antivirus or anti-virus software is used to prevent, detect, and remove malware, including but not limited to computer viruses, computer worm, trojan horses, spyware and adware.

This page talks about the software used for the prevention and removal of such threats, rather than computer security implemented by software methods.

**Windows Antivirus software**

See Also:

**avira_antivir_personal**  See Also:
- [http://www.free-av.com/fr/products/1/avira_antivir_personal__free_antivirus.html](http://www.free-av.com/fr/products/1/avira_antivir_personal__free_antivirus.html)

**Authentication**

See Also:
- [http://fr.wikipedia.org/wiki/S%C3%A9curit%C3%A9_des_syst%C3%A8mes_d%27information](http://fr.wikipedia.org/wiki/S%C3%A9curit%C3%A9_des_syst%C3%A8mes_d%27information)
Authentication (from Greek: ἀθέτης; real or genuine, from ἀθέτη authentes; author) is the act of confirming the truth of an attribute of a datum or entity.

This might involve confirming the identity of a person, tracing the origins of an artifact, ensuring that a product is what its packaging and labeling claims to be, or assuring that a computer program is a trusted one.

**Biometric authentication**

See Also:


Biometrics (or biometric authentication) consists of methods for uniquely recognizing humans based upon one or more intrinsic physical or behavioral traits. In computer science, in particular, biometrics is used as a form of identity access management and access control.

It is also used to identify individuals in groups that are under surveillance.

**Biometrics fingerjet**

See Also:


**FingerJetFX fingerprint extraction function open sourced**

DigitalPersona also announced that a FingerJetFX version of FingerJet OEM’s central feature extractor function (see above) is now available as a free, open source FingerJetFX, Open Source Edition (OSE) release under the Lesser GNU Public License (LGPL). FingerJetFX OSE is similarly compliant with the MINEX Outgoing Test, and compatible with the same operating systems, including Linux, Android, Windows, and Windows CE.

According to DigitalPersona, FingerJetFX OSE can extract fingerprint minutiae from an image in 10-25 milliseconds on an Intel Core i7. The company is also offering a commercially-licensed FingerJetFX version of the extactor that is said to offer fingerprint matching for both identification and verification while still running in a small amount of memory.

Stated Jim Fulton, vice president of DigitalPersona, “Hardware and software developers have historically had to trade off size, speed, reliability and support for standards when choosing fingerprint recognition software to embed into their products. FingerJet OEM frees device manufacturers and system integrators from having to make compromises.”

**README**
FingerJetFX OSE -- Fingerprint Feature Extractor, Open Source Edition

Copyright (c) 2011 by DigitalPersona, Inc. All rights reserved.

DigitalPersona, FingerJet, and FingerJetFX are registered trademarks or trademarks of DigitalPersona, Inc. in the United States and other countries.

FingerJetFX OSE is open source software that you may modify and/or redistribute under the terms of the GNU Lesser General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version, provided that the conditions specified in the COPYRIGHT.txt file provided with this software are met.

For more information, please visit digitalpersona.com/fingerjetfx.

FingerJetFX OSE
Fingerprint feature extractor library, initial contribution by DigitalPersona, Inc. digitalpersona.com/fingerjetfx

Overview
To use this software, application takes the fingerprint image, and makes one call into the FingerJetFX library to obtain fingerprint minutiae data, which can be used with industry-standard fingerprint recognition engines.

Platforms
The FingerJetFX OSE fingerprint feature extractor is platform-independent and can be built for, with appropriate changes to the make files, and run in environments with or without operating systems, including
- Linux
- Android
- Windows
- Windows CE
- various RTOSs

FingerJetFX OSE can run on
- general purpose 32-bit and 64-bit CPUs
- 32-bit microcontrollers (ARM, MIPS and others)

In addition, FingerJetFX OSE
- does not require DSPs or co-processors
- does not use floating point

The library has a very small memory footprint:
- ~128K of codespace (can run from ROM)
- ~128K of RAM (actual RAM requirement is the size of the image buffer + 32K)

Performance and Recognition Accuracy
The initial contribution by DigitalPersona, Inc. (www.digitalpersona.com) has met the required PIV performance thresholds for fingerprint minutiae generation in MINEX test (SDK 3F).


On an ARM Cortex-M3 150MHz processor, feature extraction takes 0.5 to 1.25 sec. On an Intel i7 PC, feature extraction takes 10 to 25 mS on a single CPU core.

The code is provided with extensive set of unit tests to ensure accurate and reliable operation in the target environments.

The Sample Application

The sample application bin/fjfxSample is a command-line tool that does feature extraction on 500DPI fingerprint image files in PGM (portable graymap) format and saves the fingerprint minutiae data in ISO/IEC 19794-2:2005 format.

Required Libraries

FingerJetFX OSE requires
- CxxTest - open source test framework
- STLport - portable implementation of Standard Template Library (STL)
These third party components are included in the folder src/ExternalDependencies for your convenience to ensure that the library builds correctly.

Standards and Compatibility

The output from FingerJetFX OSE is formatted according to the ANSI INSITS 378-2004 or ISO/IEC 19794-2:2005 specifications for fingerprint minutiae data.

Building FingerJetFX OSE

Run ‘make’ to build the library and the sample application.
- lib/libfjfx.so
- bin/fjfxSample

Run ‘make check’ to build the library, sample application and the unit tests and to test the libraries using the unit tests and the sample application.

Run ‘make debug’ to build debug version of the library and sample application:
- lib/debug/libfjfx.so
- bin/debug/fjfxSample
Note that the debug version is significantly slower than the release version.

Run ‘make checkdebug’ to build the debug version of library, sample application and the unit tests and to test the libraries using the unit tests and the sample application.

Contents

README.txt This file

2.25. Security
Questions

This software is provided AS-IS, however we’ll do our best to answer questions. Please see digitalpersona.com/fingerjetfx for assistance.

Computer viruses

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Ver_informatique
- https://secure.wikimedia.org/wiki/Computer_worm

A computer worm is a self-replicating malware computer program, which uses a computer network to send copies of itself to other nodes (computers on the network) and it may do so without any user intervention. This is due to security shortcomings on the target computer. Unlike a computer virus, it does not need to attach itself to an existing program. Worms almost always cause at least some harm to the network, even if only by consuming bandwidth, whereas viruses almost always corrupt or modify files on a targeted computer.

stuxnet worm

See Also:

- https://secure.wikimedia.org/wiki/Stuxnet

Stuxnet is a Windows computer worm discovered in July 2010 that targets industrial software and equipment. While it is not the first time that hackers have targeted industrial systems, it is the first discovered malware that spies on and subverts industrial systems, and the first to include a programmable logic controller (PLC) rootkit.

The worm initially spreads indiscriminately, but includes a highly specialized malware payload that is designed to target only Siemens Supervisory Control And Data Acquisition (SCADA) systems that are configured to control and monitor specific industrial processes.

Stuxnet infects PLCs by subverting the Step-7 software application that is used to reprogram these devices

Computer viruses

See Also:

- https://secure.wikimedia.org/wiki/fr/wiki/Virus_informatique
Program protection and reverse engineering are dualisms of good and evil. Beneficial uses of reverse engineering abound: malicious software needs to be analyzed and understood in order to prevent their spread and to assess their functional footprint; owners of intellectual property (IP) at times need to recover lost or unmaintained designs. Conversely, malicious reverse engineering allows illegal copying and subversion and designers can employ obfuscation and tamper-proofing on IP to target various attack vectors. In this sense, protecting IP and protecting malware from detection and analysis is a double-edged sword: depending on the context, the same techniques are either beneficial or harmful. Likewise, tools that deobfuscate malware in good contexts become analysis methods that support reverse engineering for illegal activity.

### cryptography

See Also:

- [http://fr.wikipedia.org/wiki/Portail:S%C3%A9curit%C3%A9_informatique](http://fr.wikipedia.org/wiki/Portail:S%C3%A9curit%C3%A9_informatique)
- [http://directory.fedoraproject.org/wiki/CoolKey](http://directory.fedoraproject.org/wiki/CoolKey)
- [http://docs.python.org/library/crypto.html](http://docs.python.org/library/crypto.html)

### Hash functions

See Also:

A hash function is any algorithm or subroutine that maps large data sets of variable length, called keys, to smaller data sets of a fixed length.

For example, a person’s name, having a variable length, could be hashed to a single integer.

The values returned by a hash function are called hash values, hash codes, hash sums, checksums or simply hashes.

Français  On nomme fonction de hachage une fonction particulière qui, à partir d’une donnée fournie en entrée, calcule une empreinte servant à identifier rapidement, bien qu’incomplètement, la donnée initiale.

Les fonctions de hachage sont utilisées en informatique et en cryptographie.

Terminologie  Le résultat d’une fonction de hachage peut être appelé selon le contexte:

- somme de contrôle,
- empreinte,
- hash,
- résumé de message,
- condensé
- condensat
- ou encore empreinte cryptographique lorsque l’on utilise une fonction de hachage cryptographique.

Il ne faut toutefois pas confondre une fonction de hachage avec une signature numérique qui fournit, en plus du code de hachage, une information sur l’identité de l’émetteur du message.

SHA-256  See Also:

- hashlib python module

Definitions  SHA-256 (Secure Hash Algorithm) est une fonction de hachage cryptographique conçue par la National Security Agency des États-Unis (NSA), et publiée en 2000.

Elle produit un résultat (appelé « hash » ou condensat) de 256 bits et dérive du SHA-1

Les fonctions de hachage sont utilisées en informatique et en cryptographie.
Fonctionnement  Très proche du SHA-1 dont il est issu, SHA-256 en diffère par la longueur du résultat obtenu, mais aussi par certaines opérations, fonctions et constantes spécifiques qui amènent ce résultat.

Celles-ci sont structurées sur le même modèle que SHA-1 (prétraitement et calcul), mais le travail s’effectue en 64 rondes avec 8 variables initialisées avec des constantes spécifiques. Les fonctions booléennes utilisées sont différentes de SHA-1.

À la fin des 64 tours les huit variables concaténées forment le résultat.

Introduction à la cryptographie

See Also:

http://www.ellendhel.net/article.php?ref=2011+10+12-0

Ayant eu l’honneur de présenter mon introduction à la cryptographie lors des dernières RMLL à Strasbourg j’avais préparé pour l’occasion une version légèrement différente des précédentes, plus courte (mais cela n’a pas été suffisant, bavard que je suis) et réactualisée sur quelques points.


cryptography libraries

See Also:

http://docs.python.org/library/crypto.html

Cryptographic algorithms

polarssl  See Also:

• http://en.wikipedia.org/wiki/PolarSSL
• http://polarssl.org/

PolarSSL is an SSL library written in ANSI C. PolarSSL makes it easy for developers to include cryptographic and SSL/TLS capabilities in their (embedded) products with as little hassle as possible.

It is designed to be readable, documented, tested, loosely coupled and portable.

Hash functions libraries

m2crypto  See Also:

• http://chandlerproject.org/bin/view/Projects/MeTooCrypto
• http://packages.python.org/m2crypto/
• http://svn.osafoundation.org/m2crypto/trunk/doc/howto.ssl.html
Contents

- m2crypto
  - Presentation
  - Tutorial
  - Subversion source repository
  - Bugzilla database

Presentation M2Crypto is the most complete Python wrapper for OpenSSL featuring RSA, DSA, DH, HMACs, message digests, symmetric ciphers (including AES); SSL functionality to implement clients and servers; HTTPS extensions to Python’s httplib, urllib, and xmlrpclib; unforgeable HMAC’ing AuthCookies for web session management; FTP/TLS client and server; S/MIME; ZServerSSL: A HTTPS server for Zope and ZSmime: An S/MIME messenger for Zope.

M2Crypto can also be used to provide SSL for Twisted.

Tutorial See Also:

http://svn.osafoundation.org/m2crypto/trunk/doc/howto.ssl.html

Subversion source repository Anonymous access to the repository:

svn co http://svn.osafoundation.org/m2crypto/trunk m2crypto

Subversion over SSH (with write access) to the repository:

svn co svn+ssh://svn.osafoundation.org/svn/m2crypto/trunk m2crypto

Browse the Subversion repository:

http://svn.osafoundation.org/m2crypto/
viewvc
websvn (check the RSS feeds for changes)

Bugzilla database Please note that OSAF’s Bugzilla installation has several products, and M2Crypto is but one of them.

Take that into account when you search for bugs and file new bugs.

List all open M2crypto bugs.

File an M2Crypto bug (advanced).

See Also:

https://bugzilla.osafoundation.org/
Passlib is a password hashing library for Python 2 & 3, which provides cross-platform implementations of over 30 password hashing algorithms, as well as a framework for managing existing password hashes. It’s designed to be useful for a wide range of tasks, from verifying a hash found in `/etc/shadow`, to providing full-strength password hashing for multi-user application.

Get a local copy of the passlib repository with this command:

```
hg clone https://code.google.com/p/passlib/
```

As a quick sample, the following code hashes and then verifies a password using the SHA256-Crypt algorithm:

```
>>> # import the hash algorithm
>>> from passlib.hash import sha256_crypt

>>> # generate new salt, and hash a password
>>> hash = sha256_crypt.encrypt("toomanysecrets")

>>> hash
'$5$rounds=80000$zvpXD3gCkrt7tw.1$QqeTS01NhEfgryc5oMgq1o8qCEAcmye3FoMSuvgToC'

>>> # verifying the password
>>> sha256_crypt.verify("toomanysecrets", hash)
True
>>> sha256_crypt.verify("joshua", hash)
False
```
Development tools, Release 2012.06.18

hashlib python module

See Also:

• http://docs.python.org/library/hashlib.html#module-hashlib
Contents
• hashlib python module
– Presentation
– Example

Presentation This module implements a common interface to many different secure hash and message
digest algorithms.
Included are the FIPS secure hash algorithms SHA1, SHA224, SHA256, SHA384, and SHA512 (defined in FIPS
180-2) as well as RSA’s MD5 algorithm (defined in Internet RFC 1321).
The terms secure hash and message digest are interchangeable.
Older algorithms were called message digests. The modern term is secure hash.
Example For example,
repetition’:

to obtain the digest of the string ’Nobody inspects the spammish

>>> import hashlib
>>> m = hashlib.md5()
>>> m.update("Nobody inspects")
>>> m.update(" the spammish repetition")
>>> m.digest()
’\xbbd\x9c\x83\xdd\x1e\xa5\xc9\xd9\xde\xc9\xa1\x8d\xf0\xff\xe9’
>>> m.digest_size
16
>>> m.block_size
64

More condensed:
>>> hashlib.sha256("Nobody inspects the spammish repetition").hexdigest()
’a4337bc45a8fc544c03f52dc550cd6e1e87021bc896588bd79e901e2’

A generic new() constructor that takes the string name of the desired algorithm as its first parameter also exists to
allow access to the above listed hashes as well as any other algorithms that your OpenSSL library may offer. The
named constructors are much faster than new() and should be preferred.
Using new() with an algorithm provided by OpenSSL:
>>> h = hashlib.new(’ripemd160’)
>>> h.update("Nobody inspects the spammish repetition")
>>> h.hexdigest()
’cc4a5ce1b3df48aec5d22d1f16b894a0b894eccc’

936

Chapter 2. Development


**PERSEUS library**

See Also:

- http://code.google.com/p/libperseus/
- http://code.google.com/p/andromeda/downloads/list

PERSEUS library is an open source technology whose aim is to secure any kind of communication streams against illegitimate or abusive eavesdropping except for Nation State Security offices, provided that a suitable, huge computing power is used (from tens of hours with a supercomputer).

PERSEUS enables to provide at the same time users’ needs for privacy and confidentiality while preserving the ability of security agencies (police, defense, national security...) to eavesdrop communications of really bad actors (terrorists, child pornographs...).

The PERSEUS technology can be very useful in different cases:

- To protect against illegal or abusive eavesdropping activity by malware on a large scale (e.g. botnets which are listening on non standard TCP ports rather than hooking keyboards for example; hooking techniques are indeed supposed to be detected by antivirus software).

- To protect against private stream eavesdropping by private intelligence companies (consuming behaviour analysis, economic intelligence, non legal eavesdropping...). 

- To protect communications from “non democratic” states towards democratic states (e.g: journalists).

- To protect professionnal of business communications in countries where cryptography use is limited or forbidden.

The interest of the PERSEUS technology lies also in the fact that TRANSEC aspect is also provided (protecting the communication channel).

In other words, any data stream protected with PERSEUS looks like a lot of legitimate, unencrypted data streams.

Moreover, its statistical profile is constantly changing (statistical mutation).

PERSEUS technology has been developed by Eric Filiol. Current applications and implementations of PERSEUS library are:

- Eddy Deligne has implemented the first application dedicated to the HTTP stream through a Linux/Windows Firefox module (the sever is also available). See the Google code repository.

- Fabien Jobin has designed the Andromede library embedding the PERSEUS library to protect torrent protocols (soon available).

PERSEUS technology is an open source technology under the triple GPL/LGPL/MPL licence.

The industrial support is provided by DFT-Technology.

**Cryptography people**

Dan Boneh

See Also:

- http://crypto.stanford.edu/~dabo/
- http://www.crypto-class.org/
Professor of Computer Science and Electrical Engineering, Stanford University.

Professor Dan Boneh heads the applied cryptography group at the Computer Science department at Stanford University. Professor Boneh's research focuses on applications of cryptography to computer security. His work includes cryptosystems with novel properties, web security, security for mobile devices, digital copyright protection, and cryptanalysis. He is the author of over a hundred publications in the field and a recipient of the Packard Award, the Alfred P. Sloan Award, and the RSA award in mathematics.

Last year Dr. Boneh received the Ishii award for industry education innovation.

Professor Boneh received his Ph.D from Princeton University and joined Stanford in 1997.

cryptography projects

crypto  See Also:

- https://crypto.is/
- https://crypto.is/docs/reading/
- https://crypto.is/guides/

The Crypto Project is an organization designed to assist and encourage anonymity and encryption research, development, and use. As part of this goal, we seek to revitalize the Cypherpunk movement and provide better software, security, and anonymity to individuals worldwide.

cryptographic tools

GNU Privacy Guard  See Also:

- http://www.gnupg.org/

GNU Privacy Guard (GnuPG or GPG) is a GPL Licensed alternative to the PGP suite of cryptographic software. GnuPG is compliant with RFC 4880, which is the current IETF standards track specification of OpenPGP.
Current versions of PGP (and Veridis’ Filecrypt) are interoperable with GnuPG and other OpenPGP-compliant systems.

GNU Privacy Guard versions

GNU Privacy Guard 2.0.18  
See Also:  

Hello!  
We are pleased to announce the availability of a new stable GnuPG-2 release: Version 2.0.18.

The GNU Privacy Guard (GnuPG) is GNU’s tool for secure communication and data storage. It can be used to encrypt data, create digital signatures, help authenticating using Secure Shell and to provide a framework for public key cryptography. It includes an advanced key management facility and is compliant with the OpenPGP and S/MIME standards.

GnuPG-2 has a different architecture than GnuPG-1 (e.g. 1.4.11) in that it splits up functionality into several modules.

However, both versions may be installed alongside without any conflict.

In fact, the gpg version from GnuPG-1 is able to make use of the gpg-agent as included in GnuPG-2 and allows for seamless passphrase caching.

The advantage of GnuPG-1 is its smaller size and the lack of dependency on other modules at run and build time.

We will keep maintaining GnuPG-1 versions because they are very useful for small systems and for server based applications requiring only OpenPGP support.

GnuPG is distributed under the terms of the GNU General Public License (GPLv3+). GnuPG-2 works best on GNU/Linux and *BSD systems but is also available for other Unices, Microsoft Windows and Mac OS X.

What’s New

- Bug fix for newer versions of Libgcrypt.
- Support the SSH confirm flag and show SSH fingerprints in ssh related pinentries.
- Improved dirmngr/gpgsm interaction for OCSP.
- Allow generation of card keys up to 4096 bit.
GPG, logiciel libre de cryptographie forte

GPG est un logiciel de chiffrement compatible avec PGP® et peut être utilisé pour un usage privé ou commercial (licence GNU GPL).

L'utilisation, la diffusion et l'exportation de GPG ont été autorisées en France pour les particuliers et les entreprises, conformément aux obligations imposées par les lois françaises limitant la libre diffusion de logiciels informatiques de cryptologie.

Cette autorisation administrative franco-française ne concerne pas la sécurité de GPG, sécurité qui reste garantie par la communauté internationale des utilisateurs de GPG, notamment la Free Software Foundation, l’“OpenPGP working group” de l’IETF, et les différentes associations de défense des libertés civiles internationales, européennes et françaises.

En clair, halte à la paranoïa : ce n’est pas parce qu’il a été “autorisé”, que GPG est un logiciel piégé par le gouvernement français!

TrueCrypt

TrueCrypt est un logiciel gratuit, open source mais non libre de chiffrement à la volée, fonctionnant sur Microsoft Windows XP/2000/2003/Vista (32-bit et 64-bit)/7, Mac OS X et Linux.

Il permet de créer un disque virtuel chiffré (volume TrueCrypt) contenu dans un fichier et de le monter comme un disque physique réel.

TrueCrypt peut aussi chiffrer une partition entière ou un périphérique, comme une disquette ou une clé USB.

Le chiffrement est automatique, en temps réel et transparent.

Tout ce qui sera stocké dans un volume TrueCrypt sera entièrement chiffré (i.e. incluant les noms des fichiers et les répertoires). Les volumes TrueCrypt se comportent (une fois montés) comme des disques durs physiques.

Il est ainsi possible, par exemple, d’en réparer le système de fichiers avec chkdsk, ou de défragmenter les volumes créés par TrueCrypt une fois montés, etc.

La version 6.0a a reçu un Certificat de Sécurité de Premier Niveau (CSPN) par l’Agence Nationale de Sécurité des Systèmes d’Information (ANSSI)
TrueCrypt versions

TrueCrypt 7.1 (September 1, 2011)

New features
- Full compatibility with 64-bit and 32-bit Mac OS X 10.7 Lion

Improvements and bug fixes
- Minor improvements and bug fixes (Windows, Mac OS X, and Linux)

Firewall softwares

See Also:

A firewall is a device or set of devices designed to permit or deny network transmissions based upon a set of rules and is frequently used to protect networks from unauthorized access while permitting legitimate communications to pass.

Windows firewall softwares

See Also:
- http://www.secuser.com/telechargement/index.htm#parefeux

ZoneAlarm  See Also:
- http://www.zonealarm.com/
- http://download.zonealarm.com/bin/free/1001_za/zaSetupWeb_101_065_000.exe

Identity cards

Belgian eID

See Also:
- http://eid.belgium.be/fr/

Contents
- Belgian eID
  - Site officiel
  - Articles
  - Développement
NIST Security Publications

NIST Computer Security Division Released Special Publication 800-125

de NIST Security Publications <csrc.nist@service.govdelivery.com>
to pvergain@gmail.com
date 3 février 2011 21:54
objet NIST Computer Security Division Released Special Publication 800-125
envoyé par service.govdelivery.com

NIST Special Publication (SP) 800-125, Guide to Security for Full Virtualization Technologies has been released. NIST Special Publication 800-125 discusses security concerns associated with full virtualization technologies for server and desktop systems, and gives recommendations for addressing these concerns.

Exploit

See Also:

• http://en.wikipedia.org/wiki/Category:Computer_security_exploits
• http://en.wikipedia.org/wiki/Exploit_%28computer_security%29

An exploit (from the same word in the French language, meaning “achievement”, or “accomplishment”) is a piece of software, a chunk of data, or sequence of commands that takes advantage of a bug, glitch or vulnerability in order to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic (usually computerised).
This frequently includes such things as gaining control of a computer system or allowing privilege escalation or a denial of service attack.

**Return of Programming Exploit**

**See Also:**

Return-oriented programming is a computer security exploit technique in which the attacker leverages control of the call stack to indirectly execute cherry-picked machine instructions or groups of machine instructions immediately prior to the return instruction in subroutines within the existing program code, in a way similar to the execution of a threaded code interpreter.

**Nodep  See Also:**
- [http://code.google.com/p/nodep](http://code.google.com/p/nodep)

A python proof of concept to show how to build a “return into libc” shellcode automatically.

With this little script, you can automatically convert a classic shellcode into a “ret into libc” shellcode. It leverages common tools such as objdump and ldd.

**metasploit project**

**See Also:**
- [http://en.wikipedia.org/wiki/Metasploit_Project](http://en.wikipedia.org/wiki/Metasploit_Project)

The Metasploit Project is an open-source computer security project which provides information about security vulnerabilities and aids in penetration testing and IDS signature development.

Its most well-known sub-project is the Metasploit Framework, a tool for developing and executing exploit code against a remote target machine.

Other important sub-projects include:
- the Opcode Database,
- shellcode archive,
- and security research

**GUI for metasploit project**

**A GUI for metasploit: Armitage  See Also:**

Armitage is a graphical cyber attack management tool for Metasploit that visualizes your targets, recommends exploits, and exposes the advanced capabilities of the framework.

Advanced users will find Armitage valuable for managing remote Metasploit instances and collaboration. Armitage’s red team collaboration features allow your team to use the same sessions, share data, and communicate through one Metasploit instance.
Armitage makes Metasploit usable for security practitioners who understand hacking but don’t use Metasploit every day. If you want to learn Metasploit and grow into the advanced features, Armitage can help you.

**Necessary Vocabulary**  To use Armitage, it helps to understand Metasploit. Here are a few things you absolutely must know before continuing:

Metasploit is a console driven application. Anything you do in Armitage is translated into a command Metasploit understands. You can bypass Armitage and type commands yourself (covered later). If you’re ever lost in a console, type help and hit enter.

Metasploit presents its capabilities as modules. Every scanner, exploit, and even payload is available as a module. If you’re scanning a host, you use an auxiliary module. Before launching a module, you must set one or more variables to configure the module. The exploit process is similar. To launch an exploit, you must choose an exploit module, set one or more variables, and launch it.

Armitage aims to make this process easier for you.

If you successfully exploit a host, you will have a session on that host. Armitage knows how to interact with shell and Windows meterpreter sessions.

Meterpreter is an advanced agent that makes a lot of post-exploitation functionality available to you. Armitage is built to take advantage of Meterpreter. Working with Meterpreter is covered later.

The Metasploit Unleashed course maintained by the Offensive Security folks is excellent. I recommend reading it before going further.

**smartcards**

See Also:

- *Personal Computer/Smart Card (PC/SC)*

## Glossaire

**Smartcard Glossary**  See Also:

- [Development Glossary](http://www.smartcardalliance.org/pages/smart-cards-intro-glossary)
- *CCID Chip/Smart Card Interface Devices Glossary*

See Also:

http://www.smartcardalliance.org/pages/smart-cards-intro-glossary
AFI  Application Family Identifier (see Parameters description: Type B AFI).

APDU  Application Protocol Data Unit.

APDU Command Header  The four byte sequence that begins an APDU; CLA INS P1 P2 (ISO/IEC 7816-4 § 5.3.1)

ATR  Answer To Reset. The first response of a card inserted in a smartcard reader is called the ATR (Answer To Reset). The main goal for the ATR is to describe the communication parameters supported to establish a dialogue with the smartcard reader. The ATR is a series of bytes standardized by the ISO7816-3 standard. The first bytes describe the conventions of communication with the card, the available interfaces and their parameters. These bytes are followed by a series of bytes called “historical bytes”, for which no standard exists and which are used to transmit information such as the type of card, the embedded software or the status of the card. Finally, these historical bytes may be followed by a byte verification, checksum of the previous bytes.

ATTRIB  PICC selection command

Carrier  Carrier: Base signal (e.g. 13.56Mhz). Reader modifies this signal to send data.

See Also:

Subcarrier

CCID  Chip/Smart Card Interface Devices. The USB CCID specification from the USB working group aims to normalize USB smartcard readers, in order to have a single driver (supplied once for all with the operating system) for virtually any reader from any manufacturer.

CID  (smart)card identifier.

CRC  Cyclic Redundancy check

CSC  A Country Signing Certificate (CSC) is needed to verify the authenticity of the Document Signing Certificate (DSC). Some countries have made their CSC publicly available. The list below contains those CSCs that we could find with Google. The list has links to government Web and LDAP servers where we found the certificates. (But please, don’t trust us, go to your government’s server and download the certificate yourself!) see http://jmrtd.org/csca.shtml


See Also:

• http://www.openpcd.org/DFU
• http://www.openpcd.org/Sam7dfu

DSFID  Data Storage Format IDentifier

EOF  End of frame

ePassport  A travel document that contains an integrated circuit chip based on international standard ISO/IEC 14443 and that can securely store and communicate the ePassport holder’s personal information to authorized reading devices.

GNU/Linux

See Also:

http://en.wikipedia.org/wiki/Linux

GNU/Linux is a generic term referring to Unix-like computer operating systems based on the Linux kernel. Their development is one of the most prominent examples of free and open source software collaboration; typically all the underlying source code can be used, freely modified, and redistributed, both commercially and non-commercially, by anyone under licenses such as the GNU General Public License. The name “Linux” comes from the Linux kernel, originally written in 1991 by Linus Torvalds.
Note: The contribution of a supporting Userland in the form of system tools and libraries from the GNU Project (announced in 1983 by Richard Stallman) is the basis for the Free Software Foundation’s preferred name GNU/Linux.

Supported platforms: IA-32, MIPS, x86-64, SPARC, DEC Alpha, Itanium, PowerPC, ARM, m68k, PA-RISC, s390, SuperH, M32R and more Latest stable release Kernel 2.6.32.8

High frequency (HF) Radio frequencies (RF) in the range of 3 MHz to 30 MHz. When used in an RF-based identification system, the high frequency used is typically 13.56 MHz. International Civil Aviation Organization (ICAO) MRTD

International Civil Aviation Organization (ICAO) MRTD International Civil Aviation Organization Machine Readable Travel Documents. ICAO establishes international standards for travel documents. An MRTD is an international travel document (e.g., a passport or visa) containing eye- and machine-readable data. ICAO Document 9303 is the international standard for MRTDs.

ICC Integrated circuit card. ICC typically refers to a plastic (or other material) card containing an integrated circuit which is compatible to ISO/IEC 7816.

Identity document A piece of documentation designed to verify aspects of a person’s identity. (See also breeder document.)

ISO/IEC 14443 ISO14443 is a series of international, vendor-independent standards for proximity RFID. It operates on 13.56MHz and uses magnetic coupling between the reader (PCD) and transponder (PICC).

See Also:
- http://www.iso.ch/
- http://www.waazaa.org/14443/
- http://www.openpcd.org/ISO14443
- ISO/IEC 14443 protocol

ISO/IEC 14443 consists of four parts and describes two types of cards: Type A and Type B, both of which communicate via radio at 13.56 MHz. The main differences between these types concern modulation methods, coding schemes (Part 2) and protocol initialization procedures (Part 3). Both Type A and Type B cards use the same transmission protocol described in Part 4. The transmission protocol specifies data block exchange and related mechanisms:

1. data block chaining
2. waiting time extension
3. multi-activation

An attempt was made to include additional legacy systems as appendixes – Type C (Sony/Japan), Type D (OTI/Israel), Type E (Cubic/USA), and Type F (Legic/Switzerland) and Type G (China) – but they were not finally accepted as ISO-standard.

ISO 14443 is supported by most contactless smart card providers to one level or another and is usually specified in the different RFP’s who are looking for contactless smart cards. VISA and MasterCard have both announced that they are supporting ISO14443 in their relevant contactless specifications.

ISO/IEC 14443 Type B The standard of contactless Smart card developed by Motorola. It is used in the Basic Resident Register Card. Type-B cards are mainly used in France and francophone countries.

ISO/IEC 14443 Type A Mifare versus ISO 14443 MIFARE and ISO 14443 Type A are not the same. While MIFARE is often viewed as equivalent to or subset of ISO 14443 Type A, it is a proprietary encryption/conditional
access protocol owned and licensed by Philips Semiconductors to multiple vendors of card ICs and reader ICs. All MIFARE readers must make use of a Philips special chip that handles these special security features. Because MIFARE has been so predominantly used with products employing ISO 14443 Type A technology, it has mistakenly become synonymous with the standard. However ISO 14443 Type A is an open standard and does not require the use of this MIFARE encryption/conditional access scheme.

ISO 14443-4 This part describes an optional transport layer protocol. This protocol is often also referred-to as “T=CL”. This is a name derived from the commonly-used contact based smart card protocols T=0 and T=1. “CL” means “contact less”.

ISO/IEC 15693 ISO15693 is a series of international, vendor-independent standards for vicinity RFID.

It operates on 13.56MHz and uses magnetic coupling between the reader (VCD) and transponder (VICC).

See Also:

- http://www.openpcd.org/ISO15693
- http://www.waazaa.org/15693/


Mifare classic This is a proprietary Philips protocol which runs on top of 14443-1,2,3 (Type A). It does not implement 14443-4. Since Mifare Classic includes some proprietary CRYPTO1 algorithm, you can only do Mifare if you have a Philips reader ASIC (such as the RC632).

NAD node address (NAD) used in ISO/IEC 14443

OpenPCD OpenPCD is a free hardware design for Proximity Coupling Devices (PCD) based on 13.56MHz communication. This device is able to screen informations from Proximity Integrated Circuit Cards (PICC) conforming to vendor-independent standards such as ISO 14443, ISO 15693 as well as proprietary protocols such as Mifare Classic. Contactless cards like these are for example used in the new electronic passports.

See Also:

http://www.openpcd.org/

PCSC, PC/SC Personal Computer/Smart Card. The PC/SC specification defines how to integrate smart card readers and smart cards with the computing environment and how to allow multiple applications to share smart card devices

PCSC Lite Personal Computer/Smart Card Lite. PCSC Lite is open source software that implements the PC/SC specification for Linux.

See Also:

libpcsclite

PCD Proximity Coupling Device: it refers to the smartcard reader (reader).

See Also:

- VCD
  - http://www.openpcd.org/

PICC Proximity Integrated Circuit Card (ICC): it refers to the card (smartcard, memory card, ...) Also known as tag.

PPS Protocol Parameter Selection

See Also:
Proximity cards: A generic name for contactless integrated circuit devices typically used for security access or payment systems. It can refer to 125 kHz RFID devices or 13.56 MHz contactless smart cards. (See ISO/IEC 14443.).

Raw mode  En français: mode transparent.

Reader  Any device that communicates information or assists in communications from a card, token or other identity document and transmits the information to a host system, such as a control panel/processor or database for further action.

Radio frequency identification (RFID): Technology that is used to transmit information about objects wirelessly, using radio waves. RFID technology is composed of 2 main pieces: the device that contains the data and the reader that captures such data. The device has a silicon chip and an antenna and the reader also has an antenna. The device is activated when put within range of the reader. The term RFID has been most commonly associated with tags used in supply chain applications in the manufacturing and retail industries.

See Also:
- http://www.rfidhandbook.blogspot.com/
- https://secure.wikimedia.org/wikipedia/fr/wiki/RFID

RC632  The RC632 is a multi-protocol 13.56MHz RFID reader ASIC produced by Philips/NXP.

See Also:
- http://www.openpcd.org/RC632

Reserved for Future Use

Smart card  A device that includes an embedded integrated circuit that can be either a secure microcontroller or equivalent intelligence with internal memory or a memory chip alone. The card connects to a reader with direct physical contact or with a remote contactless radio frequency interface. With an embedded microcontroller, smart cards have the unique ability to store large amounts of data, carry out their own on-card functions (e.g., encryption and mutual authentication) and interact intelligently with a smart card reader. Smart card technology conforms to international standards (ISO/IEC 7816 and ISO/IEC 14443) and is available in a variety of form factors, including plastic cards, subscriber identification modules used in GSM mobile phones, and USB-based tokens.

SOF  Start of Frame

Subcarrier  Signal on top of the base signal. Tag generates this signal to send data.

See Also:

Type A card  Type A smartcard = ISO1443A.

UID  Unique Identifier (UID): All ISO-compliant smart cards are provided with a UID number. For interoperability purposes, a card’s UID is open and available to be read by all compliant readers. Since this unique number is not secured by keys, reading the UID of a smart card is comparable to reading a proximity card, mag stripe card or other technology that utilizes open, unsecured numbers.

USB  Universal Serial Bus. A serial bus standard to interface devices.
What is a smart card?  A smart card is a device that includes an embedded integrated circuit that can be either a secure microcontroller or equivalent intelligence with internal memory or a memory chip alone. The card connects to a reader with direct physical contact or with a remote contactless radio frequency interface.

With an embedded microcontroller, smart cards have the unique ability to store large amounts of data, carry out their own on-card functions (e.g., encryption and mutual authentication) and interact intelligently with a smart card reader. Smart card technology conforms to international standards (ISO/IEC 7816 and ISO/IEC 14443) and is available in a variety of form factors, including plastic cards, key fobs, watches, subscriber identification modules used in GSM mobile phones, and USB-based tokens.

For the purposes of this FAQ, “card” is used as the generic term to describe any device in which smart card technology is used.

What are the ISO/IEC 14443 and ISO/IEC 7816 standards?  ISO/IEC 14443 is the international standard for contactless smart chips and cards that operate (i.e., can be read from or written to) at a distance of less than 10 centimeters (4 inches). This standard operates at 13.56 MHz and includes specifications for the physical characteristics, radio frequency power and signal interface, initialization and anticollision protocols and transmission protocol.

ISO/IEC 7816 is the international standard for contact smart cards. ISO/IEC 7816 Parts 4 and above are used by both contact and contactless smart card applications for security operations and commands for interchange.

What is a contactless smart card?  A contactless smart card includes an embedded smart card secure microcontroller or equivalent intelligence, internal memory and a small antenna and communicates with a reader through a contactless radio frequency (RF) interface. Contactless smart card technology is used in applications that need to protect personal information and/or deliver fast, secure transactions, such as transit fare payment cards, government and corporate identification cards, documents such as electronic passports and visas, and financial payment cards.

Contactless smart cards have the ability to securely manage, store and provide access to data on the card, perform on-card functions (e.g., encryption and mutual authentication) and interact intelligently with a contactless smart card reader. Contactless smart card technology and applications conform to international standards (ISO/IEC 14443 and ISO/IEC 7816).

Contactless smart card technology is available in a variety of forms - in plastic cards, watches, key fobs, documents and other handheld devices (e.g., built into mobile phones).

How do contactless smart cards work?  Contactless smart card systems are closely related to contact smart card systems. Like contact smart card systems, information is stored on a chip embedded within the contactless smart card. However, unlike the contact smart card, the power supplied to the card as well as the data exchanged between the card and the reader are achieved without the use of contacts, using magnetic or electromagnetic fields to both power the card as well as to exchange data with the reader.

Source: http://www.smartcardalliance.org/pages/smart-cards-faq
The contactless smart card contains an antenna embedded within the plastic body of the card (or within a key fob, watch or other document). When the card is brought into the electromagnetic field of the reader, the chip in the card is powered on. Once the chip is powered on, a wireless communication protocol is initiated and established between the card and the reader for data transfer.

The following four functions describe at a high level the sequence of events that happen when a contactless smart card is brought near a card reader:

- Energy transfer to the card for powering the integrated circuit (chip)
- Clock signal transfer
- Data transfer to the contactless smart card
- Data transfer from the contactless smart card

Hence, once the card is brought within range of an electromagnetic field of the required frequency, the card will be powered up, ready to communicate with the reader. Since the contactless smart cards described in this FAQ are based on the ISO/IEC 14443 standard, this frequency is 13.56 MHz and a reader that complies with the standard would have an activation field (range) of about 4 inches (approximately 10 centimeters). In other words, the card needs to be within 10 centimeters of a reader for it to be effectively powered; however, the effective range for communications for the card to be read will depend on a number of factors like the power of the reader, the antenna of the reader and the antenna of the card.

Is contactless smart card technology the same as RFID technology? No. There is significant confusion in discussions of RF-enabled applications, with contactless smart card technology often incorrectly categorized as ‘RFID.’ There are a wide range of RF technologies used for a variety of applications – each with different operational parameters, frequencies, read ranges and capabilities to support security and privacy features. For example, the RFID technologies that are used to add value in manufacturing, shipping and object-related tracking operate over long ranges (e.g., 25 feet), were designed for that purpose alone and have minimal built-in support for security and privacy.

Contactless smart cards, on the other hand, use RF technology, but, by design, operate at a short range (less than 4 inches) and can support the equivalent security capabilities of a contact smart card chip.

What security capabilities do contactless smart cards support? Contactless smart cards use RF technology, but, by design, operate at a short range (less than 4 inches) and can support the equivalent security capabilities of a contact smart card chip (see below). Contactless smart cards and readers conform to international standards, ISO/IEC 14443 and ISO/IEC 7816, and can implement a variety of industry-standard cryptographic protocols (e.g., AES, 3DES, RSA, ECC).

The contactless smart chip includes a smart card secure microcontroller and internal memory and has unique attributes RFID tags lack – i.e., the ability to securely manage, store and provide access to data on the card, perform complex functions (for example, encryption and mutual authentication) and interact intelligently via RF with a contactless reader. Applications using contactless smart cards support many security features that ensure the integrity, confidentiality and privacy of information stored or transmitted, including the following:

- Mutual authentication. For applications requiring secure card access, the contactless smart card-based device can verify that the reader is authentic and can prove its own authenticity to the reader before starting a secure transaction.
- Strong information security. For applications requiring complete data protection, information stored on cards or documents using contactless smart card technology can be encrypted and communication between the contactless smart card-based device and the reader can be encrypted to prevent eavesdropping. Hashes and/or digital signatures can be used to ensure data integrity and to authenticate the card and the credentials it contains.

Cryptographically strong random number generators can be used to enable dynamic cryptographic keys, preventing replay attacks:
• Strong contactless device security. Like contact smart cards, contactless smart card technology is extremely difficult to duplicate or forge and has built-in tamper-resistance. Smart card chips include a variety of hardware and software capabilities that detect and react to tampering attempts and help counter possible attacks. For example, the chips are manufactured with features such as extra metal layers, sensors to detect thermal and UV light attacks, and additional software and hardware circuitry to thwart differential power analysis.

• Authenticated and authorized information access. The contactless smart card’s ability to process information and react to its environment allows it to uniquely provide authenticated information access and protect the privacy of personal information. The contactless smart card can verify the authority of the information requestor and then allow access only to the information required. Access to stored information can also be further protected by a personal identification number (PIN) or biometric to protect privacy and counter unauthorized access.

• Support for biometric authentication. For human identification systems that require the highest degree of security and privacy, smart cards can be implemented in combination with biometric technology. Biometrics are measurable physical characteristics or personal behavioral traits that can be used to recognize the identity or verify the claimed identity of an individual. Smart cards and biometrics are a natural fit to provide two- or multi-factor authentication. A smart card is the logical secure storage medium for biometric information. During the enrollment process, the biometric template can be stored on the smart card chip for later verification. Only the authorized user with a biometric matching the stored enrollment template receives access and privileges.

• Strong support for information privacy. The use of smart card technology strengthens the ability of a system to protect individual privacy. Unlike other technologies, smart card-based devices can implement a personal firewall for an individual, releasing only the information required and only when it is required. The ability to support authenticated and authorized information access and the strong contactless device and data security make contactless smart cards excellent guardians of personal information and individual privacy.

It is important to note that information privacy and security must be designed into an application at the system level by the organization issuing the contactless device, card or document. It is critical that issuing organizations have the appropriate policies in place to support the security and privacy requirements of the application being deployed and then implement the appropriate technology that delivers those features. The ability of contactless smart card technology to support a wide array of security features provides organizations with the flexibility to implement the level of security that is commensurate with the risk expected in the application.

contact iso7816 protocols

ISO 7816 protocol (with contact)  See Also:
  • http://en.wikipedia.org/wiki/ISO_7816

ISO/IEC 7816 is an international standard related to electronic identification cards with contacts, especially smart cards, managed jointly by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC).

APDU  See Also:
  • http://cheef.ru/docs/HowTo/APDU.table

contactless protocols

smartcard contactless protocols

ISO/IEC 14443 protocol  See Also:
  • http://www.waazaa.org/14443/
Introduction  ISO 14443 is an international standard for «Proximty Cards», i.e. contactless cards operating at the 13.56 MHz frequency, with a maximum operating distance of 10 centimetres.

The International Organization for Standardization, or ISO, issued its ISO 14443 standards in 2008 to harmonize the use of proximity or contactless smart cards and their communication protocol. Contactless cards can retrieve the identity of a person from a distance.

The speed and convenience of using contactless smart cards has opened the doors to a large amount of applications, such as transactions with banks, mass transit or gas stations.

The 14443 guidelines promote good design and business practices to ensure that the smart card performs as expected, is highly reliable and works with many different technology platforms.

Scope  ISO 14443 specifically addresses the physical and programming aspects of contactless cards.

These new devices do not make contact with a card reader. They contain a wireless communications system that transmits identification information in proximity to a reader. These smart devices store 64 kilobytes of information and contain strong security features.

Most contact cards, such as credit cards, will eventually migrate to the contactless technology to gain access to greater data storage, better encryption and less wear and tear.

First Application  ISO 14443 has been tested and applied to passenger identification in transportation systems in Asia and Europe. For instance, South Korea implemented it in its T-money system that facilitates payment in buses, on the subway or with taxis.

Similarly, Hong Kong leveraged the guidance of the standards when it launched its Octopus card for transportation payments.

Requirements  ISO 14443 represents four sets of standards that cover the physical card requirements, the radio frequency power and signal spectrum interferences, initialization aspects and anticollision and finally communications protocols.

In addition, ISO 14443 comes in two flavors:

- Type A represents most contactless card applications.
- Type B is reserved for banking systems that carry larger amount of data.
Specifications ISO 14443 specifies the thickness—0.03 inches, or 0.84 mm—and the size of the card, which is about the size of a credit card. The radio frequency required is 13.56 megahertz and must provide a read range of 4 inches, or 10 centimeters.

The speed of communications must operate at 106 kilobits per second. Anticollision methods must be implemented to manage the read of multiple cards passing an area all at the same time, such as in mass transit transportation. ISO 1443 describes the various options for the language of communication and the protocol instructions that must be implemented.

Exclusions The standards do not cover the reading system and operating system embedded in the cards. These belong to each vendor who has proprietary rights to them. However, ISO 1443 paints the characteristics of an “envelope protocol” to guarantee an error-free communication.

Openpcd, 14443 See Also:
http://www.openpcd.org/ISO14443

Other parts

ISO/IEC 14443 anticollision

Radio interface - Type A

PCD to PICC Carrier: 13.56Mhz Modulation ASK 100% Coding: Modified Miller Bandwidth: 106kbit/s -> 847kbit/s

PICC to PCD Load modulation Subcarrier: fc=16 -> fc=128 Coding: OOK, Manchester Bandwidth: 106kbit/s -> 847kbit/s

Radio interface - Type B


Protocol issues

Protocol issue CL1356A, 14443 / 15693 protocol

Question from jncabarro@ateis-eu.com

2.25. Security
Bonjour M. Bourgault,

APDUResponse apduResp;

CardNative iCard = new CardNative();

string[] readers = iCard.ListReaders();

iCard.Connect(readers[0], SHARE.Exclusive, PROTOCOL.T1);
Console.WriteLine("Connects card on reader: " + readers[0]);

APDUParam apduParam = new APDUParam();

apduParam.Data = new byte[] { 0xFF, 0xCA, 0x00, 0x00, 0x00, 0x00 };  
apduSelectFile.Update(apduParam);  // {apduParam : Class=A0 Ins=A4 P1=00 P2=00 P3=05 Data=FFCA000000}

apduResp = iCard.Transmit(apduSelectFile);  

Merci pour votre aide

Response from PB

Vous passez un APDU "Select File" adressable uniquement à une carte supportant le T=CL (level 14443-4) alors que votre carte est une 15693.

Vous devez passer un APDU comme suit :

APDUCommand apduGetData_ID = new APDUCommand(0xFF , 0xCA, 0, 0, null, 0)

apduResp = iCard.Transmit(apduGetData);

ISO 15693 protocol  See Also:

• http://en.wikipedia.org/wiki/ISO_15693
• http://www.waazaa.org/15693/

ISO 15693 is an international standard for “Vicinity Cards”, i.e. contactless cards operating at the 13.56 MHz frequency, with a maximum operating distance of 1 to 1.5 metres.

iso18092 Near Field Communication  See Also:

• http://en.wikipedia.org/wiki/Near_Field_Communication

Near Field Communication or NFC, is a short-range high frequency wireless communication technology which enables the exchange of data between devices over about a 10 centimetre (around 4 inches) distance.[1] The technology is a simple extension of the ISO/IEC 14443 proximity-card standard (contactless card, RFID) that combines the interface of a smartcard and a reader into a single device. An NFC device can communicate with both existing ISO/IEC 14443 smartcards and readers, as well as with other NFC devices, and is thereby compatible with existing contactless infrastructure already in use for public transportation and payment. NFC is primarily aimed at usage in mobile phones.
Near Field Communication (NFC)  See Also:

- https://secure.wikimedia.org/wikipedia/en/wiki/Near_field_communication

Near field communication, or NFC, is a set of short-range wireless technologies, typically requiring a distance of 4cm or less. NFC operates at 13.56 MHz and at rates ranging from 106 kbit/s to 848 kbit/s. NFC communication always involves an initiator and a target: the initiator actively generates an RF field that can power a passive target. This enables NFC targets to take very simple form factors such as tags, stickers, key fobs, or cards that do not require batteries.

NFC peer-to-peer communication is also possible, where both devices are powered.

Security aspects  Although the communication range of NFC is limited to a few centimeters, NFC alone does not ensure secure communications.

In 2006, Ernst Haselsteiner and Klemens Breitfuß described different possible types of attacks, and detail how to leverage NFC’s resistance to Man-in-the-middle attacks to establish a specific key.[11]

Unfortunately, as this technique is not part of the ISO standard, NFC offers no protection against eavesdropping and can be vulnerable to data modifications.

Applications may use higher-layer cryptographic protocols (e.g., SSL) to establish a secure channel.

NFC and Qt  See Also:


Contents

- NFC and Qt
  - What is NFC ?
  - First, what can Qt do with NFC ?
  - What are we demonstrating ?

What is NFC ?  For those that don’t know, NFC is a hot new technology that will make mobile devices that much more powerful and important in our lives. It gives users the ability to interact with other devices that are nearby. Think about buying a soda from a vending machine simply by waving your phone at it, or printing pictures from your mobile by setting it down on top of your printer.

At the show here there are many examples of NFC.

First, what can Qt do with NFC ?  Qt and the Qt Mobility APIs make it possible to integrate with stores and more. The upcoming release of Qt Mobility (1.2) features the Connectivity API, which allows NFC enabled phones such as the Nokia C7 to recognise NFC tags.

What are we demonstrating ?  Digia, one of our certified Qt partners has produced a nifty demo that previews what Qt can do in the NFC space. They have created an app called ShopWizer that currently runs on a Nokia C7. Applications such as Shopwizer are made possible by this API, and Digia used a technical preview version of the Connectivity API from Qt Mobility 1.2 to build the demo.

smartcards

smartcard applications
smartcard bank application

smartcard EMV (Europay, MasterCard et Visa) protocol  See Also:
  • smartcard structures

Le standard EMV voit le jour en 1994 et devient rapidement la référence de facto pour la communication entre la puce d’une carte bancaire (de débit ou de crédit) d’une part et les terminaux de vente ou distributeurs de billets d’autre part.

Les initiales EMV proviennent du nom des 3 firmes l’ayant conçu et adopté initialement, à savoir Europay, MasterCard et Visa.

Les spécifications de ce standard sont maintenues par EMVCo [1] et mises à disposition gratuitement sur leur site web.

eopayment (Common API to use all French online payment credit card processing services)  See Also:
  • http://dev.entrouvert.org/projects/eopayment/repository
  • smartcard structures

Python module to interface with French’s bank credit card online payment services.

Services supported are:
  • ATOS/SIP used by:
    – BNP under the name Mercanet,
    – Banque Populaire (before 2010/2011) under the name Cyberplus,
    – CCF under the name Elysnet,
    – HSBC under the name Elysnet,
    – Crédit Agricole under the name e-Transactions,
    – La Banque Postale under the name ScelliusNet,
    – LCL under the name Sherlocks,
    – Société Générale under the name Sogenactif
    – and Crédit du Nord under the name Webaffaires,
  • SystemPay by Banque Populaire (since 2010/2011) and SPPlus by Caisse d’épargne.

You can emit payment request under a simple API which takes as input a dictionary as configuration and an amount to pay.

You get back a transaction_id. Another unique API allows to handle the notifications coming from those services, reporting whether the transaction was successful and which one it was.

The full content (which is specific to the service) is also reported for logging purpose.

For SystemPay and SPPlus the module is totally independent from the respective implementation distributed by the Bank, but for ATOS/SIPS the kit distributed by the bank is also needed as the protocol created by ATOS is proprietary and not documented.

The spplus module also depend upon the python Crypto library for DES decoding of the merchant key

smartcards  See Also:
**felica**  FeliCa is a contactless RFID smart card system by Sony in Japan, primarily used in electronic money cards. The name stands for Felicity Card. First utilized in the Octopus card system in Hong Kong, the technology is used in a variety of cards also in countries such as Singapore and Japan.

See Also:

**mifare**  See Also:

**smartcard tools**  See Also:
- Personal Computer/Smart Card (PC/SC)

**pyscard**  See Also:
- pyscard PCSC
- Personal Computer/Smart Card (PC/SC)

Pyscard consists of smartcard.scard, an extension module wrapping Windows smart card base components (also known as PCSC) on Windows and PCSC lite on linux and Mac OS X Tiger and Leopard, and smartcard, a higher level python framework built on top of the raw PCSC API.

**pyscard examples**  To test the examples, download the last framework and go the src/smartcard/Examples/wx directory.

Examples

```bash
| noexcept
  | ---wx
  |     +---apdumanager
  |     |     | apdumanager.py
  |     |     | SampleAPDUManagerPanel.py
  |     |     | setup.py
  |     |     | images
  |     |     | mysmartcard.ico
  |     +---cardmonitor
  |     |     | cardmonitor.py
  |     |     | setup.py
  |     |     |
```
Development tools, Release 2012.06.18

|    | ---images
|    | mysmartcard.ico
|    |
|    | ---pcscdiag
|    | pcscdiag.py
|    |
|    | ---readerviewer
|    | readerviewer.py
|    | setup.py
|    |
|    | ---images
|    | readerviewer.ico

```
pyscard readerview

python reader_viewer.py
```

![A simple reader monitoring tool](image1)

```
src/reader_viewer.py
```

![Tools to and axle to a card](image2)
pyscard apdumanager

#!/usr/bin/env python

""
Simple application to send APDUs to a card.
__author__ = "http://www.gemalto.com"

Copyright 2001-2010 gemalto
Author: Jean-Daniel Aussel, mailto:jean-daniel.aussel@gemalto.com

This file is part of pyscard.
pyscard is free software; you can redistribute it and/or modify
it under the terms of the GNU Lesser General Public License as published by
the Free Software Foundation; either version 2.1 of the License, or
(at your option) any later version.
pyscard is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU Lesser General Public License for more details.
You should have received a copy of the GNU Lesser General Public License
along with pyscard; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
""

import os
import sys
import os.path
from smartcard.wx.SimpleSCardApp import *

from SampleAPDUManagerPanel import SampleAPDUManagerPanel

def we_are_frozen():
    """Returns whether we are frozen via py2exe.
    This will affect how we find out where we are located.
    From WhereAmI page on py2exe wiki."""
    return hasattr(sys, "frozen")

def module_path():
    """ This will get us the program’s directory,
    even if we are frozen using py2exe. From WhereAmI page on py2exe wiki."""

    if we_are_frozen():
        return os.path.dirname(unicode(sys.executable, sys.getfilesystemencoding( )))

    return os.path.dirname(unicode(__file__, sys.getfilesystemencoding( )))

def main(argv):
    app = SimpleSCardApp(
        appname='A tool to send apdu to a card',
        apppanel=SampleAPDUManagerPanel,

2.25. Security
appstyle=TR_SMARTCARD | TR_READER | PANEL_APDUTRACER,
appicon=os.path.join(module_path(), 'images', 'mysmartcard.ico'),
size=(800, 600))
app.MainLoop()

if __name__ == "__main__":
    import sys
    main(sys.argv)

# ! /usr/bin/env python
"
Simple smart card monitoring application.

__author__ = "http://www.gemalto.com"

Copyright 2001-2010 gemalto
Author: Jean-Daniel Aussel, mailto:jean-daniel.aussel@gemalto.com
This file is part of pyscard.

pyscard is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

pyscard is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with pyscard; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

""
import os.path
import sys
from smartcard.wx.SimpleSCardApp import *

ID_TEXT = 10000

def we_are_frozen():
    """Returns whether we are frozen via py2exe.
    This will affect how we find out where we are located.
    From WhereAmI page on py2exe wiki."""

    return hasattr(sys, "frozen")

def module_path():
    """This will get us the program’s directory, even if we are frozen using py2exe. From WhereAmI page on py2exe wiki."""

    if we_are_frozen():
        return os.path.dirname(unicode(sys.executable, sys.getfilesystemencoding( )))

    return os.path.dirname(unicode(__file__, sys.getfilesystemencoding( )))

class SamplePanel(wx.Panel, SimpleSCardAppEventObserver):
    """A simple panel that displays activated cards and readers. The panel implements the SimpleSCardAppEventObserver, and has a chance to react on reader and card activation/deactivation."""

    def __init__(self, parent):
        wx.Panel.__init__(self, parent, -1)

        sizer = wx.FlexGridSizer(0, 3, 0, 0)
        sizer.AddGrowableCol(1)
        sizer.AddGrowableRow(1)

        sizer.Add([20, 20], 0, wx.ALIGN_CENTER | wx.ALL, 5)
        sizer.Add([20, 20], 0, wx.ALIGN_CENTER | wx.ALL, 5)
        sizer.Add([20, 20], 0, wx.ALIGN_CENTER | wx.ALL, 5)
Development tools, Release 2012.06.18

sizer.Add([20, 20], 0, wx.ALIGN_CENTER | wx.ALL, 5)
self.feedbacktext = wx.StaticText(self, ID_TEXT, "", wx.DefaultPosition, wx.DefaultSize, 0)
sizer.Add(self.feedbacktext, 0, wx.ALIGN_LEFT | wx.ALL, 5)
sizer.Add([20,
sizer.Add([20,
sizer.Add([20,
sizer.Add([20,

20],
20],
20],
20],

0,
0,
0,
0,

wx.ALIGN_CENTER
wx.ALIGN_CENTER
wx.ALIGN_CENTER
wx.ALIGN_CENTER

|
|
|
|

wx.ALL,
wx.ALL,
wx.ALL,
wx.ALL,

5)
5)
5)
5)

self.SetSizer(sizer)
self.SetAutoLayout(True)

# callbacks from SimpleSCardAppEventObserver interface
def OnActivateCard(self, card):
"""Called when a card is activated by double-clicking on the card or reader tree control or t
In this sample, we just connect to the card on the first activation."""
SimpleSCardAppEventObserver.OnActivateCard(self, card)
self.feedbacktext.SetLabel(’Activated card: ’ + ‘card‘)

def OnActivateReader(self, reader):
"""Called when a reader is activated by double-clicking on the reader tree control or toolbar
SimpleSCardAppEventObserver.OnActivateReader(self, reader)
self.feedbacktext.SetLabel(’Activated reader: ’ + ‘reader‘)
def OnDeactivateCard(self, card):
"""Called when a card is deactivated in the reader tree control or toolbar."""
SimpleSCardAppEventObserver.OnActivateCard(self, card)
self.feedbacktext.SetLabel(’Deactivated card: ’ + ‘card‘)

def OnSelectCard(self, card):
"""Called when a card is selected by clicking on the card or reader tree control or toolbar."
SimpleSCardAppEventObserver.OnSelectCard(self, card)
self.feedbacktext.SetLabel(’Selected card: ’ + ‘card‘)
def OnSelectReader(self, reader):
"""Called when a reader is selected by clicking on the reader tree control or toolbar."""
SimpleSCardAppEventObserver.OnSelectReader(self, reader)
self.feedbacktext.SetLabel(’Selected reader: ’ + ‘reader‘)

def main(argv):
app = SimpleSCardApp(
appname=’A simple card monitoring tool’,
apppanel=SamplePanel,
appstyle=TR_SMARTCARD | TR_READER,
appicon=os.path.join(module_path(), ’images’, ’mysmartcard.ico’),
size=(800, 600))
app.MainLoop()
if __name__ == "__main__":
import sys
main(sys.argv)

962

Chapter 2. Development


pyscard pcscdiag

#!/usr/bin/env python

""
Example wxPython application that displays readers and inserted cards ATRs.
This example displays a snapshot of the readers and cards, there is no
automatic refresh of the readers and cards.

__author__ = "http://www.gemalto.com"

Copyright 2001-2010 gemalto
Author: Jean-Daniel Aussel, mailto:jean-daniel.aussel@gemalto.com

This file is part of pyscard.

pyscard is free software; you can redistribute it and/or modify
it under the terms of the GNU Lesser General Public License as published by
the Free Software Foundation; either version 2.1 of the License, or
(at your option) any later version.

pyscard is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License

2.25. Security 963
import smartcard.Exceptions
import smartcard.System
import smartcard.util

# wxPython GUI modules (http://www.wxpython.org)
try:
    import wx
except:
    print 'You need wxpython (http://www.wxpython.org) to run this sample from the source code!
    print 'press a key to continue'
import msvcrt
msvcrt.getch()
import sys
sys.exit()

def getATR(reader):
    """Return the ATR of the card inserted into the reader."""
    try:
        connection = reader.createConnection()
        atr = ""
        connection.connect()
        atr = smartcard.util.toHexString(connection.getATR())
        connection.disconnect()
    except smartcard.Exceptions.NoCardException:
        atr = "no card inserted"
    return atr

class pcscdiag(wx.Frame):
    def init__(self, parent, title):
        wx.Frame.__init__(self, parent, -1, title, size=(600, 400))
        w, h = self.GetClientSizeTuple()
        self.tree = wx.TreeCtrl(self, wx.NewId(), wx.DefaultPosition, (w, h), wx.TR_HAS_BUTTONS | wx.TR_EDIT_LABELS)
        self.InitTree()
        self.OnExpandAll()

    def InitTree(self):
        self.tree.AddRoot("Readers and ReaderGroups")
        readerNode = self.tree.AppendItem(self.tree.GetRootItem(), "Readers")
        for reader in smartcard.System.readers():
            childReader = self.tree.AppendItem(readerNode, 'reader')
            childCard = self.tree.AppendItem(childReader, getATR(reader))

        readerGroupNode = self.tree.AppendItem(self.tree.GetRootItem(), "Readers Groups")
        for readerGroup in smartcard.System.readergroups():
            childReaderGroup = self.tree.AppendItem(readerGroupNode, readerGroup)
            readers = smartcard.System.readers(readerGroup)
            for reader in readers:
                child = self.tree.AppendItem(childReaderGroup, 'reader')
```python
def OnExpandAll(self):
    '''expand all nodes'''
    root = self.tree.GetRootItem()
    fn = self.tree.Expand
    self.traverse(root, fn)
    self.tree.Expand(root)

def traverse(self, traverseroot, function, cookie=0):
    '''recursively walk tree control'''
    if self.tree.ItemHasChildren(traverseroot):
        firstchild, cookie = self.tree.GetFirstChild(traverseroot)
        function(firstchild)
        self.traverse(firstchild, function, cookie)

        child = self.tree.GetNextSibling(traverseroot)
        if child:
            function(child)
            self.traverse(child, function, cookie)

if __name__ == '__main__':
    app = wx.PySimpleApp()
    frame = pcsccd diag(None, "Smartcard readers and reader groups")
    frame.Show()
    app.MainLoop()
```

**pyscard based tool** See Also:
- http://pyscard.sourceforge.net/

**omnikey2ivy** See Also:
- http://pypi.python.org/pypi/omnikey2ivy/

**Introduction** omnikey2ivy emit card id on Ivy software bus.
Bus adress can be specified with “-b” command line option or environment variable IVYBUS.
Message from are “Id cardnumber), for instance : “Id 020408DD”.

**pyscard versions**

**pyscard versions**
- http://sourceforge.net/projects/pyscard/

**pyscard 1.6.12** See Also:
- http://pypi.python.org/pypi/pyscard/1.6.12
rfidiot

rfidiot intro  RFIDIOt is an open source python library for exploring RFID devices. It’s called “RFIDIOt” for two reasons:

1. I like puns. This one stands for “RFID IO tools”
2. Since I haven’t done any serious programming for a long time, I felt like an idiot having to learn a whole new language and the code probably looks like it’s written by an idiot. However, python rocks, so it was worth it!

What does it do?  It currently drives a range of RFID readers made by ACG, called the HF Dual ISO and HF Multi ISO, which are both 13.56MHz devices, and the LF MultiTag which is 125/134.2kHz.

Frosch Hitag reader/writers are also now supported. There’s no reason it couldn’t work with others, these are just the first ones I got my hands on, and since they present themselves to the O/S as standard serial devices without having to install any drivers, it made interfacing very simple (but see the Technical Note section below as I’ve had some issues recently).

I have written some example programs to read/write tags and have started on the library routines to handle the data structures of specific tags like MIFARE®. It is far from complete but I thought I’d follow the “publish early, publish often” philosophy on this one...

PC/SC (MUSCLE) devices, such as the Omnikey CardMan are also supported. I am curently testing with a CardMan 5321.

Links

- http://rfidiot.org/

wzmrtd

wzmrtd intro  wzMRTD is a software library that reads electronic passports (also well known as E-Passports or biometric passports).

Electronic passports embed a contactless smart card, which stores the data to be read by border-clearance software or by wzMRTD.

The mapping of the chip (file system, data layout) has been defined by ICAO, an international organization, in Machine Readable Travel Document (MRTD) standard.

Communication between wzMRTD and the chip is performed through a contactless reader compliant with ISO/IEC 14443 standard.

PC/SC and Pro-Active readers are supported. Accessing the data is possible only in a secure session, where the authentication key is computed from information printed on the passport (number, date of delivery and date of expiry).

The digits -printed on the Machine Readable Zone (MRZ)- can be read by an OCR scanner.

wzMRTD is an open-source project, and aims to be modified, enhanced, re-used... by anybody interested in this subject.

Why?  I decided to develop wzMRTD and to publish its source code in a reaction of mood vis-a-vis many articles published in late 2006, dealing with the “vulnerability” of electronic passports. The most significant one is this article (title : “Cracked it!”) published by the Guardian in UK. Here’s an extract :

(...we have just sucked out all the supposedly secure data and biometric information from three new passports and displayed it all on a laptop computer.)
To make a long story short, wzMRTD is an answer to this misinformation.

In my opinion, nobody shall be surprised to be able to read a passport with a proximity card reader and a small piece of software, this is precisely the intended goal (remember what “MRTD” stands for...). Relax, guys, you’ve not “cracked” anything, you’ve just read a document, and implemented what was described in it. No surprise at all, unless you forgot to read the more interesting pages...

This article in question caused a small thread on french Usenet group fr.misc.cryptologie, where interesting contributions (especially signed by Sylvain) may be read (in french).

Due to my position at Pro-Active, I had a few contactless readers, and a few contactless passports, so I decided to “crack” them as well, and to publish the project for people interested in the subject.

And afterwards? Obviously wzMRTD is not really interesting for end-user, one needs a small graphical interface to display the data and the picture, that’s why wzPASS has been written too.

Todo list/Roadmap
- Handling of DG11 and DG12,
- Linux version using pcsc-lite,
- Publish documentation.

Credits / Thanks
- Sylvain Ferey for his repeated support,
- Stanislas Dourdin, author of an early yet functionnal pre-version,
- Damien Croisot, who provided his own passport for test,
- Oberthur Card System, provider of the Utopians passports,
- Pro-active, my official provider for readers, computers, and coffee.

Links
- http://www.waazaa.org/wzmrtd/
Qu’est-ce que wzPASS ?  wzPASS est un logiciel sous Windows capable de lire et d’afficher le contenu des Passeports Electroniques (ou Passeports Biométriques). Vous devez disposer d’un lecteur sans contact (PC/SC ou Pro-Active) pour utiliser wzPASS.

wzPASS est avant tout destiné aux personnes curieuses, qui souhaitent savoir ce qu’il y a exactement dans leur propre passeport. Néanmoins, wzPASS peut très facilement s’intégrer dans une borne de consultation en libre service, par exemple installée dans une mairie, pour que l’usager recevant son nouveau passeport puisse en vérifier immédiatement le fonctionnement et le contenu.

wzPASS un logiciel libre (open source), qui s’appuie sur le composant logiciel wzMRTD.

Comment ça marche ?

Spécifications internationales du passeport  Un organisme international, l’ICAO (International Civil Aviation Organization) a publié une spécification qui assure l’interopérabilité des Passeports Electroniques entre les différents pays émetteurs (le passeport électronique est officiellement nommé MRTD, pour « Machine Readable Travel Document »). Un grand nombre de pays délivrent désormais des passeports conformes à ce standard, qui sont identifiés par ce pictogramme sur la couverture:

Le pictogramme du Passeport Electronique

Lecture sans contact  Le Passeport Electronique intègre une carte à puce sans contact. Le lecteur et la puce établissent une communication de proximité selon la norme ISO/IEC 14443 (porteuse à 13.56MHz, distance < 10cm).

wzPASS établit la communication avec la carte (le lecteur de proximité n’est qu’un moyen de liaison «passif»), lit tout le contenu du passeport, et en affiche les éléments intéressants (identité et photo de face).

Authentication pour protéger l’intimité  L’accès aux Passeports Electronique est protégé par un système d’authentification symétrique (BAC, “Basic Authentication”) qui sert à interdire la lecture à l’insu du porteur(contrainte de respect de la vie privée).

Pour pouvoir communiquer avec la puce électronique, il faut avoir le passeport ouvert sous les yeux, afin d’y lire la clé d’authentification (cette clé se déduit de données numériques imprimées dans la «Machine Readable Zone» -MRZ).

La MRZ se situe en bas de la page principale du passeport.
Les lecteurs de passeports utilisés au quotidien par les douaniers disposent d’un scanner avec système de reconnaissance optique des caractères (OCR).

Ils peuvent ainsi lire automatiquement la MRZ pour effectuer l’authentification nécessaire à la lecture de la puce. Ces équipements étant très onéreux, dans le cas de wzPASS nous saisirons manuellement la MRZ dans le logiciel.

Remerciements

- Sylvain Ferey pour son soutien régulier,
- Stanislas Dourdin qui a développé une première maquette opérationnelle,
- Damien Croisot qui a osé me prêter son passeport,
- Oberthur Card System pour les passeports Utopiens,
- Pro-Active, mon fournisseur officiel pour tous les moyens techniques et pour le café.

Download

Pré-requis  wzPASS est une application Windows qui s’exécute dans le framework .NET v.2. Vous devez télécharger et installer le runtime Microsoft .NET v.2 avant de pouvoir utiliser wzPASS.

Evidemment, pour lire votre passeport, vous aurez besoin d’un lecteur sans contact. wzPASS a été testé avec les lecteurs sans contact PC/SC de la marque OMNIKEY, et avec les lecteurs de chez Pro-Active.

Téléchargement  Veuillez lire et accepter le contrat de licence avant de télécharger ou d’installer wzPASS.

- wzPASS v0.8 programme d’installation (EXE)

Télécharger le code source  Le fichier ZIP ci-dessous contient l’ensemble du code source de wzPASS. Il s’agit d’un projet en langage C#, l’environnement de développement (IDE) utilisé est Sharp Develop.

Le programme d’installation a été réalisé avec Inno Setup

Veuillez lire et accepter le contrat de licence avant de télécharger ou d’installer wzPASS.

- wzPASS v0.8 code source (ZIP)

Autres éléments disponibles

- Télécharger wzMRTD, le composant logiciel sur lequel wzPASS est construit.

Retour d’expérience et support  Même si j’ai fait de mon mieux pour livrer un logiciel totalement opérationnel et non une version béta, il reste impossible de tester wzPASS avec tous les types de passeports en circulation, ni avec tous les lecteurs du marché.

Si vous rencontrez un problème dans wzPASS, et notamment une incompatibilité avec votre passeport, ou si vous souhaitez proposer une évolution du logiciel, n’hésitez pas à m’envoyer un e-mail.

Links
• http://www.waazaa.org/wzpass/index.php
• http://www.waazaa.org/wzpass/download.php

lafargue smartcard tools  See Also:
• https://www.lafargue.name/smart-tools/index.html
• https://www.lafargue.name/smart-tools/atr/
• http://www.sconnect.com/FAQ/index.html

https://www.lafargue.name/smart-tools/index.html  Before anything else, two things:
• This page contains tools which will give you advanced access, in particular to French Navigo cards: you must be aware that absolutely NO restricted documents were used in the process of building these tools: only standard and publicly available information was used.
• Second, though these tools use a technology called SConnect which is developed by Gemalto, Gemalto is not associated at all to these tools and their capabilities.

Sconnect

http://www.sconnect.com/FAQ/index.html  SConnect is a smart card agnostic browser extension for major web browsers for Windows, Mac OS X and Linux operating systems. The primary purpose of SConnect is to provide a connectivity bridge between JavaScript that runs in a rendered web page in browser and a smart card.

How does SConnect help solving the aforementioned problems?  Coming up with abstractions in software has always been a challenge. Some abstractions work really well and some abstractions leak. The cryptographic abstraction used by browsers to access smart card functionality unfortunately leak in the context of web applications. They were primarily conceived to be used with client applications and find themselves very stretched in a different context. SConnect addresses this problem of leaky abstractions by moving down the the abstraction stack and to use an abstraction that targets smart cards. This abstraction stack fortunately is the same across operating systems and is an industry standard. It is known as PC/SC. The implementation of this standard is installed by default on all operating system images and exposes the same API as mandated.

What SConnect does is that it helps the JavaScript that runs in the browser access the PC/SC layer of the operating system. This enables programming the logic of accessing applications in a smart card through a cross browser consistent and simple interface via JavaScript. Using this approach one can write the logic that was earlier implemented as compiled binaries, partially in JavaScript and partially in a server side language of your choice (ASP.NET, PHP, Java,
Python, etc). Not only does the end user experience get enhanced multi-fold, this technique also does not have the limitations of the classical architecture.

**Python Simple Smartcard Interpreter (PSSI)**

Python script that provides an abstract layer for smartcard reading. Thanks to it, it is possible to read a smartcard by simply adding its structure in the form of a plugin, without taking care of the communication layer.

The tool comes with several plugins, namely SIM, EMV, and NAVIGO.

- [http://code.google.com/p/pssi/](http://code.google.com/p/pssi/)
  - [http://code.google.com/p/pssi/source/browse/#svn/trunk/pssi](http://code.google.com/p/pssi/source/browse/#svn/trunk/pssi)
- [http://code.google.com/u/eric.bourry/](http://code.google.com/u/eric.bourry/)

**smartcard structures**

- [du passeport belge:](https://code.google.com/p/pssi/source/browse/trunk/pssi/plugins/belgian-eid/structures.py)
- [d’une carte EMV:](https://code.google.com/p/pssi/source/browse/trunk/pssi/plugins/emv/structures.py)
- [navigo:](https://code.google.com/p/pssi/source/browse/trunk/pssi/plugins/navigo/structures.py)
- [sim:](https://code.google.com/p/pssi/source/browse/trunk/pssi/plugins/sim/structures.py)

**Introduction**

In this wiki page, you will find some indications about how to use PSSI and how to write your own plugin.

**How to use the tool**

In order to run PSSI, make sure pyscard and the PSSI packages are installed. You can then run pssi.py, without forgetting to specify a plugin, and read your smartcards!

Usage: pssi  [options] plugin_path
Example: ./pssi.py plugins/sim

- `-a` apdu mode, show the APDUs
- `-b` choose bruteforce mode
- `-c` specify the class byte for the bruteforce mode, in hexadecimal
- `-d` choose dump mode (default, specify a plugin)
- `-h` show this help
- `-l` choose loop mode (specify a plugin)
- `-r` enable recursive mode in the bruteforce mode
- `-v` verbose mode, show the raw data along with the interpretation

Here are some examples:

* `./pssi.py plugins/emv`
* `./pssi.py plugins/navigo`
* `./pssi.py plugins/sim`

**How to write a plugin**
Setting up the environment

- Make a clean directory
- Write your plugin.py, with the following functions:
  - getClassByte: This function returns the class byte of your smart card, which is the first byte sent in every APDU.
  - getRootStructure: This function returns the root structure of your smart card.
  - getInterpretersTable: This function returns the interpreters table, which makes a correspondence between a data type and an interpreting function. An interpreting function is used to convert raw data (bytes or bits) into meaningful and human-readable information.

Adding some structures  A structure describes a file in the smartcard file hierarchy, by listing its elements. It consists of an array of tuples, whose number depends on whether you are describing a DF (Dedicated File), such as the MF (Master File), or an EF (Elementary file).

For a DF, you use 4-tuples, each one containing:
- The name of the element, which will be part of the final output
- Its type, which is a member of the FieldType class
- Its relative address, as a 2-byte-array
- The name of the Python structure detailing the contents of the element. If the TLV (Tag-Length-Value) scheme must be used, this value is -1

There is a special case, present in the EMV card for instance, where an EF can contain a series of structures, whose number and addresses are not known in advance. In this case, the type is FieldType.DFList, and, if the name of the Python structure is -1, a fifth member of the tuple can be defined, which specifies a default structure if no tag is found, or if the tag is not a key of the interpreting table.

Here is an example:

```python
structSIM = [
    ('ICC identification', FieldType.TransparentEF, [0x2f, 0xe2], structICC),
    ('DF GSM', FieldType.DF, [0x7f, 0x20], structGSM),
    ('DF Télécom', FieldType.DF, [0x7f, 0x10], structDFTel),
]
```

For an EF, you use 5-tuples, each one containing:
- The name of the element, which will be part of the final output
- Its type, which is a member of the FieldType class
- Its size, usually in bytes. If the element doesn’t have a fixed size, this number can be negative; it then specifies the amount of data following the current element, which has to be fixed
- A comment, which will be part of the final output, in order to make it user-friendly. It can be empty
- The nature of the element, which allows PSSI to know how to interpret the element, as described in the next section

Here is an example:

```python
structNumber = [
    ('Alpha identifier', FieldType.Final, -14, 'Name of the contact', FinalType.Contact),
    ('Length of relevant information', FieldType.Final, 1, 'In bytes', FinalType.Integer),
]
```
cardpeek
A tool to read the contents of smartcards / Un outil pour lire le contenu des cartes à puce

Contents

• cardpeek
  – En
  – Fr
  – smartcard structures
  – Versions

En  Cardpeek is a Linux tool to read the contents of ISO7816 smartcards. It features a GTK GUI to represent card data in a tree view, and is extendable with a scripting language (LUA).

The goal of this project is to allow smartcard owners to be better informed about what type of personal information is stored in these devices.

The tool currently reads the contents of:

• EMV cards
• Navigo public transport cards (partially supports MOBIB as well)
• Moneo ePurse cards
• The French health card “Vitale 2”
• Electronic/Biometric passports in BAC security mode.

Latest source code tarball is here.

More info on the Wiki here: http://code.google.com/p/cardpeek/wiki/Main

Fr  Cardpeek est un outil de lecture de carte à puce avec une interface graphique basée sur GTK 2.0, fonctionnant sous GNU Linux et extensible par le langage de programmation LUA.

Cardpeek est un outil qui a pour objectif de vous permettre d’accéder aux informations personnelles qui sont stockées dans vos carte à puce.

Vous pouvez ainsi être mieux informé des données qui sont collectées sur vous.

Dans cette version, l’application est capable de lire le contenu des cartes suivantes:

• Les cartes à puce bancaires EMV
• Les cartes de transport d’île de France Navigo et certaines autres cartes similaires utilisée en France ou en Belgique (support MOBIB encore imparfait).
• Les cartes Monéo

2.25. Security
Les cartes Vitales 2
Les passeports électroniques/biомétriques avec une sécurité BAC.

smartcard structures
- carte vitale 2: https://code.google.com/p/cardpeek/source/browse/trunk/dot_cardpeek_dir/scripts/vitale_2.lua

La dernière archive du code source est disponible ici.

Plus de détails sur le Wiki : http://code.google.com/p/cardpeek/wiki/Main

Versions

cardpeek versions

<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>cardpeek 0.7.1 May 13, 2012</td>
</tr>
<tr>
<td>• cardpeak 0.7.1 May 13, 2012</td>
</tr>
<tr>
<td>• Introduction</td>
</tr>
</tbody>
</table>

Introduction May 13, 2012: Cardpeek version 0.7.1, a bug correction release, is now available for Windows XP 32bits and Linux.

13 Mai, 2012: Cardpeek version 0.7.1, une version corrective, est maintenant disponible sous Windows XP 32bits et Linux.

This release does not add new features but corrects some bugs and compatibility issues:
- EMV script works on more cards, including NFC bank cards now.
- Navigo script was corrected to address some rare subway station naming issues.
- Other minor bugs and improvements.

Full documentation is available here.

Download and install from source.

Cardpeek has been ported to Windows XP (32bits), try out the Installer. Please note that this tool does not work on 64 bit windows platforms yet.

See Also:
- http://www.integrazioneweb.com/monosim/
**what is monosim?** Monosim is a simple application that can be used to read, write, update, delete and backup your sim card contacts. It open and save also some format files to manage your contacts also in a text files.

To connect monosim to your smartcard you need use a standard PCSC smartcard reader as towitoko, acs, athena, blutronics, etc.

**smartcard sim python module**  
See Also:

- http://michau.benoit.free.fr/
- http://michau.benoit.free.fr/codes/smartcard/
- http://michau.benoit.free.fr/codes/smartcard/iso7816.py
- https://code.google.com/p/pssi/source/browse/trunk/pssi/plugins/sim/structures.py
- http://ludovic.rousseau.free.fr/softwares/pcsc-tools/smartcard_list.txt

Smartcards with electrical connectors are in general all compliant with the basis standard ISO7816 - part 1-2-3-4. ISO standards are not public, but you can have a good view on it here:


The principle is to communicate with the card over a serial link, sending APDU commands, and receiving replies with always SW codes and sometimes data.

All the SIM and USIM cards are based on these standards, and in addition, on ETSI and 3GPP standards. ETSI standards are not public.

3GPP standards are [http://www.3gpp.org/specification-numbering](http://www.3gpp.org/specification-numbering).

SIM cards are easy to read as they are only mono-application: check 3GPP TS 11.11 or TS 51.011.

USIM cards are some more evolved as they are multi-applications / multi-channels cards: check ETSI TS 101.221 and 3GPP TS 31.101 and 31.102

**cardlib for smartcard**  
See Also:

- http://michau.benoit.free.fr/codes/smartcard/card/
- http://ludovic.rousseau.free.fr/softwares/pcsc-tools/smartcard_list.txt

```
| smartcard_list.txt
|--card
   `--__init__.py
      | utilities.py
      | SIM.py
      | ICC.py
      `--init__.py
      | utilities.py
      | USIM.py
      | ICC.py
      | smartcard_list.txt
      | FS.py
```

This is a new version, build from the older iso7816.py monolithic script, to request mainly SIM and USIM cards. It needs python 2.6 (may work on older version: not tested), a smartcard reader (USB or RS-232), pcsc-lite driver and daemon and its python binding pyscard.
The library is now splitted in several files:

- **utils.py**: contains facilities for parsing TV, TLV, BER_TLV records and some other little functions used by others modules
- **ICC.py**: contains the 2 main classes:
  - ISO7816: which implements a little part of the ISO standard
  - UICC: which implements part of the ETSI standard inheriting from the ISO7816 class
- **SIM.py**: contains the class SIM inheriting from ISO7816 class, implementing part of TS 51.011
- **USIM.py**: contains the class USIM inheriting from UICC class, implementing part of TS 31.102
- **FS.py**: dictionaries referencing SIM and USIM files address as described in those 3GPP standards

All file addresses and data syntax are list of short integer (e.g. `[0xAA, 0xBB, 0xCC, ...]`), as used by pyscard.

Mistakes remains surely still in these scripts, so use it with care.

UIICC and USIM classes do not implement logical channels.

ISO7816 and UIICC security conditions parsing is really not well implemented.

**Example sessions**

```python
>>> from card.ICC import ISO7816
>>> a = ISO7816()
>>> a.ATR_scan()
```

Output:

```
smartcard reader:  id3 Semiconductors CL1356A+ 0
g smart card ATR is: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 00 00 00 00 00 6B
ATR analysis:
TD1: 80
TD2: 1
supported protocols {'T=0': True, 'T=1': True)
T=0 supported True
T=1 supported True
checksum: 107
clock rate conversion factor: 372
bit rate adjustment factor: 1
maximum programming current: 50
programming voltage: 5
guard time: None
nb of interface bytes: 2
nb of historical bytes: 15
None
historical bytes: 80 4F 0C A0 00 00 03 06 03 00 00 00 00 00 00
checksum: 0x6B

using pcsc_scan ATR list file: smartcard_list.txt
smartcard ATR fingerprint:
RFID - ISO 14443 Type A Part 3 (as per PCSC std part3)
specialized Mifare Ultralight card
```
iso7816.py  The old version replace by the carlib module.

Download iso7816.py

#!/usr/bin/env python

#################################
# Python library to work on
# smartcard defined with ISO 7816
# # Specially designed SIM and USIM class
# # for ETSI / 3GPP cards
# # needs pyscard from:
# # http://pyscard.sourceforge.net/
# #
# classic python modules
import os, re
from time import sleep

# smartcard python modules from pyscard
from smartcard.CardType import AnyCardType
from smartcard.CardRequest import CardRequest
from smartcard.CardConnection import CardConnection
from smartcard.util import toHexString, toBytes
from smartcard.ATR import ATR
from smartcard.sw.SWExceptions import SWException
from smartcard.Exceptions import CardConnectionException

# generic functions
def byteToBit(byte):
    '''
    byteToBit(0xAB) -> [1, 0, 1, 0, 1, 0, 1, 1]
    convert a byte integer value into a list of bits
    '''
    bit = [0, 0, 0, 0, 0, 0, 0, 0]
    for i in range(8):
        if byte % pow(2, i+1):
            bit[7-i] = 1
            byte = byte - pow(2, i)
    return bit

def stringToByte(s):
    '''
    strinToByte('test') -> [116, 101, 115, 116]
    convert a string into a list of bytes
    '''
    bytes = []
    for c in s:
        bytes.extend(toBytes(c.encode('hex'))) 
    return bytes

def byteToString(bytelist):
    '''
    strinToByte([116, 101, 115, 116]) -> 'test'
    convert a list of bytes into a string
    '''

2.25. Security
string = ''
for b in bytelist:
    string += chr(b)
return string

def LV_parser(bytes):
    LV_parser([0x02, 0xAB, 0xCD, 0x01, 0x12, 0x34]) -> [[171, 205], [18], []]
parse Length-Value records in a bytes list
return a list of list of bytes

length coded on 1 byte
values = []
while len(bytes) > 0:
    l = bytes[0]
    values.append(bytes[1:1+l])
    bytes = bytes[1+l:]
return values

def first_TLV_parser(bytes):
    first_TLV_parser([0xAA, 0x02, 0xAB, 0xCD, 0xFF, 0x00]) -> (170, 2, [171, 205])
parse first TLV format record in a list of bytes
return a 3-Tuple: Tag, Length, Value
Value is a list of bytes
parsing of length is ETSI’style 101.220
Tag = bytes[0]
if bytes[1] == 0xFF:
    Len = bytes[2]*256 + bytes[3]
    Val = bytes[4:4+Len]
else:
    Len = bytes[1]
    Val = bytes[2:2+Len]
return (Tag, Len, Val)

def first_BERTLV_parser(bytes):
    first_BERTLV_parser([0xAA, 0x02, 0xAB, 0xCD, 0xFF, 0x00]) -> ([1, ‘contextual’, ‘constructed’, 10], [1, 2], [2, [171, 205]])
parse first BER-TLV format record in a list of bytes
return a 3-Tuple: Tag, Length, Value
Tag: [Tag length, Tag class, Tag DO, Tag number]
Length: [Length length, Length value]
Value: [Value length, Value bytes list]
parsing of length is ETSI’style 101.220
# Tag class and DO
byte0 = byteToBit(bytes[0])
if byte0[0:2] == [0,0]:
    Tag_class = ‘universal’
elif byte0[0:2] == [0,1]:
Tag_class = 'applicative'
if byte0[0:2] == [1,0]:
    Tag_class = 'contextual'
elif byte0[0:2] == [1,1]:
    Tag_class = 'private'
if byte0[2:3] == [0]:
    Tag_DO = 'primitive'
elif byte0[2:3] == [1]:
    Tag_DO = 'constructed'

# Tag coded with more than 1 byte
i = 0
if byte0[3:8] == [1,1,1,1,1,1,1,1]:
    Tag_bits = byteToBit(bytes[1])[1:8]
    i += 1
    while byteToBit(bytes[i])[0] == 1:
        i += 1
        Tag_bits += byteToBit(bytes[i])[1:8]

else:
    Tag_bits = byte0[3:8]

# Tag number calculation
Tag_num = 0
for j in range(len(Tag_bits)):
    Tag_num += Tag_bits[len(Tag_bits)-j-1]*pow(2,j)

# Length coded with more than 1 byte
if bytes[i+1] > 0x50:
    Len_num = bytes[i+1] - 0x50
    Len_bytes = bytes[i+2:i+1+Len_num]
    Len = 0
    for j in range(len(Len_bytes)):
        Len += bytes[i+1+Len_num-j]*pow(256,j)
    Val = bytes[i+1+Len_num:i+1+Len_num+Len]

else:
    Len_num = 1
    Len = bytes[i+1]
    Val = bytes[i+2:i+2+Len]

return ([i+1, Tag_class, Tag_DO, Tag_num],[Len_num, Len],[len(Val), Val])

class ISO7816:
    
    define attributes, methods and facilities for ISO-7816-4 standard smartcard

    use self.dbg = 1 or more to print live debugging information

    dbg = 0


def __init__(self, CLA=0x00):
    
    '''
    smartcard connection initialization and class CLA code definition
    use "pyscard" library services
    '''
    cardtype = AnyCardType()
cardrequest = CardRequest(timeout=1, cardType=cardtype)
self.cardservice = cardrequest.waitforcard()
self.cardservice.connection.connect()
self.reader = self.cardservice.connection.getReader()
self.ATR = self.cardservice.connection.getATR()

self.CLA = CLA
self.INS_dic = {0xA4 : 'SELECT FILE',
                0xF2 : 'STATUS',
                0xB0 : 'READ BINARY',
                0xD6 : 'UPDATE BINARY',
                0xB2 : 'READ RECORD',
                0xDC : 'UPDATE RECORD',
                0xA2 : 'SEARCH RECORD',
                0x32 : 'INCREASE',
                0xCB : 'RETRIEVE DATA',
                0xDB : 'SET DATA',
                0x20 : 'VERIFY',
                0x24 : 'CHANGE PIN',
                0x26 : 'DISABLE PIN',
                0x28 : 'ENABLE PIN',
                0x2C : 'UNBLOCK PIN',
                0x04 : 'DEACTIVATE FILE',
                0x44 : 'ACTIVATE FILE',
                0x88 : 'AUTHENTICATE',
                0x89 : 'AUTHENTICATE',
                0x84 : 'GET CHALLENGE',
                0xAA : 'TERMINAL CAPABILITY',
                0x10 : 'TERMINAL PROFILE',
                0xC2 : 'ENVELOPE',
                0x12 : 'FETCH',
                0x14 : 'TERMINAL RESPONSE',
                0x70 : 'MANAGE CHANNEL',
                0x73 : 'MANAGE SECURE CHANNEL',
                0x75 : 'TRANSACT DATA',
                0xC0 : 'GET RESPONSE'}

def disconnect(self):
    
    '''
    disconnect smartcard: stops the session
    '''
    self.cardservice.connection.disconnect()

def define_class(self, CLA=0x00):
    
    '''
    define smartcard class attribute for APDU command
    override CLA value defined in class initialization
    '''
    self.CLA = CLA

def ATR_scan(self, smlist_file='/usr/share/pcsc/smartcard_list.txt'):
print smartcard info retrieved from AnswerToReset

if pcsc_scan is installed,
use the signature file for guessing the card

print '\nsmartcard reader: ', self.reader
if self.ATR != None:
  print '\nsmart card ATR is: %s' % toHexString(self.ATR)
  print 'ATR analysis: '
  print ATR(self.ATR).dump()
  print '\nhistorical bytes: ', toHexString(ATR(self.ATR).getHistoricalBytes())
  ATRcs = ATR(self.ATR).getChecksum()
  if ATRcs :
    print 'checksum: ', "0x%X" % ATRcs
  else:
    print 'no ATR checksum'
  print '\nusing pcsc_scan ATR list file: %s' % smlist_file
if os.path.exists(smlist_file):
  smlist = open(smlist_file).readlines()
  ATRre = re.compile('(^3[BF]){1}.{1,}$')
  ATRfinger = ''
  j = 1
  for i in range(len(smlist)):
    if ATRre.match(smlist[i]):
      if re.compile(smlist[i][:len(smlist[i])-1]).match(toHexString(self.ATR)):
        while re.compile('\t.{1,}').match(smlist[i+j]):
          j += j
        if ATRfinger == '' :
          print "no ATR fingerprint found in file: %s" % smlist_file
        else:
          print "smartcard ATR fingerprint:

%smartcard ATR fingerprint:\n
%atrfinger"
      else:
          print "$s file not found" % smlist_file

def sw_status(self, sw1, sw2):

  self.sw_status(sw1=int, sw2=int) -> string

  SW status bytes interpretation as defined in ISO-7816 part 4 standard
  helps to speak with the smartcard!

  status = 'undefined status'
if sw1 == 0x90 and sw2 == 0x00: status = 'normal processing: command accepted: no further qualification'
elif sw1 == 0x61: status = 'normal processing: %i bytes still available' % sw2
elif sw1 == 0x62:
  status = 'warning processing: state of non-volatile memory unchanged'
  if sw2 == 0x00: status += ': no information given'
elif sw2 == 0x81: status += ': part of returned data may be corrupted'
elif sw2 == 0x82: status += ': end of file/record reached before reading Le bytes'
elif sw2 == 0x83: status += ': selected file invalidated'
elif sw2 == 0x84: status += ': FCI not formatted'
else: status += ': undefined SW2 code: 0x%02X' % toHexString([sw2])
elif sw1 == 0x63:
  status = 'warning processing: state of non-volatile memory changed'
  if sw2 == 0x00: status += ': no information given'
elif sw2 == 0x81: status += ': file filled up by the last write’
elif 0xC0 <= sw2 <= 0xCF:
    status += ': counter provided by %s' % toHexString([sw2])[1]
else:
    status += ': undefined SW2 code: %s' % toHexString([sw2])

elif sw1 == 0x64:
    status = 'execution error: state of non-volatile memory unchanged'
    if sw2 != 0x00:
        status += ': SW2 code %s is RFU' % toHexString([sw2])

elif sw1 == 0x65:
    status = 'execution error: state of non-volatile memory changed'
    if sw2 == 0x00:
        status += ': no information given'
    else:
        status += ': undefined SW2 code: %s' % toHexString([sw2])

else:
    status += ': undefined SW2 code: %s' % toHexString([sw2])

elif sw1 == 0x66:
    status = 'execution error: reserved for security-related issues'
    if sw2 == 0x00:
        status += ': no information given'
    elif sw2 == 0x81:
        status += ': logical channel not supported'
    else:
        status += ': undefined SW2 code: %s' % toHexString([sw2])

elif sw1 == 0x67:
    status = 'execution error: wrong length (P3 parameter)'
generic function to send apdu, receive and interpret response

force card reconnection if pycard transmission fails

```
try:
data, sw1, sw2 = self.cardservice.connection.transmit(apdu)
except CardConnectionException:
    ISO7816.__init__(self, CLA = self.CLA)
data, sw1, sw2 = self.cardservice.connection.transmit(apdu)
if apdu[1] in self.INS_dic.keys(): apdu_name = self.INS_dic[apdu[1]] + ''
else:
    apdu_name = ''
sw_stat = self.sw_status(sw1, sw2)
return
['%s' % (apdu_name, toHexString(apdu)),
'swl, sw2: %s - %s % ( toHexString([sw1, sw2]), sw_stat ),
(sw1, sw2),
data ]
```

def bf_cla(self, start=0, param=[0xA4, 0x00, 0x00, 0x02, 0x3F, 0x00]):
    
    self.bf_cla( start=int(starting CLA), param=list(bytes for selecting file 0x3F, 0x00) ) -> list( CLA which could be supported )
    
    try all classes CLA codes to check the supported ones
    print CLA suspected to be supported
    
    WARNING: can block the card definitively
    Do not do it with your own VISA card
    
    list = []
    for i in range(start, 256):
        ret = self.sr_apdu([i] + param)
        if ret[2] != (0x6E, 0x00):
            print ret
            list.append(i)
    return list

def bf_ins(self, start=0):
    
    self.bf_cla( start=int(starting INS) ) -> list( INS which could be supported )
    
    try all instructions INS codes to check the supported ones
    print INS suspected to be supported
    
    WARNING: can block the card definitively
    Do not do it with your own VISA card
    
    list = []
    for i in range(start, 256):
        print 'DEBUG: testing %d for INS code with %d CLA code' % (i, self.CLA)
        ret = self.sr_apdu([self.CLA, i, 0x00, 0x00])
        if ret[2] != (0x6D, 0x00):
            print ret
            list.append(i)
    return list

def READ_BINARY(self, P1=0x00, P2=0x00, Le=0x01):
    
    APDU command to read the content of EF file with transparent structure
    Le: length of data bytes to be read

2.25. Security
call sr_apdu method

READ_BINARY = [self.CLA, 0xB0, P1, P2, Le]
return self.sr_apdu(READ_BINARY)

def WRITE_BINARY(self, P1=0x00, P2=0x00, Data=[]):
    
    APDU command to write the content of EF file with transparent structure

    Data: list of data bytes to be written
    call sr_apdu method

    WRITE_BINARY = [self.CLA, 0xD0, P1, P2, len(Data)] + Data
    return self.sr_apdu(WRITE_BINARY)

def UPDATE_BINARY(self, P1=0x00, P2=0x00, Data=[]):
    
    APDU command to update the content of EF file with transparent structure

    Data: list of data bytes to be written
    call sr_apdu method

    UPDATE_BINARY = [self.CLA, 0xD6, P1, P2, len(Data)] + Data
    return self.sr_apdu(UPDATE_BINARY)

def ERASE_BINARY(self, P1=0x00, P2=0x00, Lc=None, Data=[]):
    
    APDU command to erase the content of EF file with transparent structure

    Lc: 'None' or '0x02'
    Data: list of data bytes to be written
    call sr_apdu method

    if Lc is None: ERASE_BINARY = [self.CLA, 0x0E, P1, P2]
    else: ERASE_BINARY = [self.CLA, 0x0E, P1, P2, 0x02] + Data
    return self.sr_apdu(ERASE_BINARY)

def READ_RECORD(self, P1=0x00, P2=0x00, Le=0x00):
    
    APDU command to read the content of EF file with record structure

    P1: record number
    P2: reference control
    Le: length of data bytes to be read
    call sr_apdu method

    READ_RECORD = [self.CLA, 0xB2, P1, P2, Le]
    return self.sr_apdu(READ_RECORD)

def WRITE_RECORD(self, P1=0x00, P2=0x00, Data=[]):
    
    APDU command to write the content of EF file with record structure

    P1: record number
    P2: reference control
    Data: list of data bytes to be written in the record
    call sr_apdu method
WRITE_RECORD = [self.CLA, 0xD2, P1, P2, len(Data)] + Data
return self.sr_apdu(WRITE_RECORD)

def APPEND_RECORD(self, P2=0x00, Data=[]):
    '''
    APDU command to append a record on EF file with record structure
    P2: reference control
    Data: list of data bytes to be appended on the record
    call sr_apdu method
    '''
    APPEND_RECORD = [self.CLA, 0xE2, 0x00, P2, len(Data)] + Data
    return self.sr_apdu(APPEND_RECORD)

def UPDATE_RECORD(self, P1=0x00, P2=0x00, Data=[]):
    '''
    APDU command to update the content of EF file with record structure
    P1: record number
    P2: reference control
    Data: list of data bytes to update the record
    call sr_apdu method
    '''
    UPDATE_RECORD = [self.CLA, 0xDC, P1, P2, len(Data)] + Data
    return self.sr_apdu(UPDATE_RECORD)

def GET_DATA(self, P1=0x00, P2=0x00, Le=0x01):
    '''
    APDU command to retrieve data object
    P1 and P2: reference control for data object description
    Le: number of bytes expected in the response
    call sr_apdu method
    '''
    GET_DATA = [self.CLA, 0xCA, P1, P2, Le]
    return self.sr_apdu(GET_DATA)

def PUT_DATA(self, P1=0x00, P2=0x00, Data=[]):
    '''
    APDU command to store data object
    P1 and P2: reference control for data object description
    Data: list of data bytes to put in the data object structure
    call sr_apdu method
    '''
    if len(Data) == 0: PUT_DATA = [self.CLA, 0xDA, P1, P2]
    elif 1 <= len(Data) <= 255: PUT_DATA = [self.CLA, 0xDA, P1, P2, len(Data)] + Data
    else: PUT_DATA = [self.CLA, 0xDA, P1, P2, 0xFF] + Data[0:255]
    return self.sr_apdu(PUT_DATA)

def SELECT_FILE(self, P1=0x00, P2=0x00, Data=[0x3F, 0x00]):
    '''
    APDU command to select file
    P1 and P2: selection control
    Data: list of bytes describing the file identifier or address
    call sr_apdu method
    '''
Development tools, Release 2012.06.18

if len(Data) == 0: SELECT_FILE = [self.CLA, 0xA4, P1, P2]
e elif 1 <= len(Data) <= 255: SELECT_FILE = [self.CLA, 0xA4, P1, P2, len(Data)] + Data
e else: SELECT_FILE = [self.CLA, 0xA4, P1, P2, 0xFF] + Data[0:255]
return self.sr_apdu(SELECT_FILE)

def VERIFY(self, P2=0x00, Data=[]):
    
    APDU command to verify user PIN, password or security codes

    P2: reference control
    Data: list of bytes to be verified by the card
    call sr_apdu method

    if len(Data) == 0: VERIFY = [self.CLA, 0x20, 0x00, P2]
e elif 1 <= len(Data) <= 255: VERIFY = [self.CLA, 0x20, 0x00, P2, len(Data)] + Data
e else: VERIFY = [self.CLA, 0x20, 0x00, P2, 0xFF] + Data[0:255]
return self.sr_apdu(VERIFY)

def INTERNAL_AUTHENTICATE(self, P1=0x00, P2=0x00, Data=[]):
    
    APDU command to run internal authentication algorithm

    P1 and P2: reference control (algo, secret key selection...)
    Data: list of bytes containing the authentication challenge
    call sr_apdu method

    INTERNAL_AUTHENTICATE = [self.CLA, 0x88, P1, P2, len(Data)] + Data
return self.sr_apdu(INTERNAL_AUTHENTICATE)

def EXTERNAL_AUTHENTICATE(self, P1=0x00, P2=0x00, Data=[]):
    
    APDU command to conditionally update the security status of the card after getting a challenge

    P1 and P2: reference control (algo, secret key selection...)
    Data: list of bytes containing the challenge response
    call sr_apdu method

    if len(Data) == 0: EXTERNAL_AUTHENTICATE = [self.CLA, 0x82, P1, P2]
e elif 1 <= len(Data) <= 255: EXTERNAL_AUTHENTICATE = [self.CLA, 0x82, P1, P2, len(Data)] + Data
e else: EXTERNAL_AUTHENTICATE = [self.CLA, 0x82, P1, P2, 0xFF] + Data[0:255]
return self.sr_apdu(EXTERNAL_AUTHENTICATE)

def GET_CHALLENGE(self):
    
    APDU command to get a challenge for external entity authentication to the card

    call sr_apdu method

    GET_CHALLENGE = [self.CLA, 0x84, 0x00, 0x00]
return self.sr_apdu(GET_CHALLENGE)

def MANAGE_CHANNEL(self, P1=0x00, P2=0x00):
    
    APDU to open and close supplementary logical channels

    P1=0x00 to open, 0x80 to close
    P2=0x00, 1, 2 or 3 to ask for logical channel number
    call sr_apdu method
if (P1, P2) == (0x00, 0x00): MANAGE_CHANNEL = [self.CLA, 0x70, P1, P2, 0x01]
else: MANAGE_CHANNEL = [self.CLA, 0x70, P1, P2]
return self.sr_apdu(MANAGE_CHANNEL)

def GET_RESPONSE(self, Le=0x01):
    ""
    APDU command to retrieve data after selection or other kind of request that should get an extensive reply
    Le: expected length of data
    call sr_apdu method
    ""
    GET_RESPONSE = [self.CLA, 0xC0, 0x00, 0x00, Le]
    return self.sr_apdu(GET_RESPONSE)

def ENVELOPPE(self, Data=[]):
    ""
    APDU command to encapsulate data (APDU or other...)
    check ETSI TS 102.221 for some examples...
    Data: list of bytes
    call sr_apdu method
    ""
    if len(Data) == 0: ENVELOPPE = [self.CLA, 0xC2, 0x00, 0x00]
    elif 1 <= len(Data) <= 255: ENVELOPPE = [self.CLA, 0xC2, 0x00, 0x00, len(Data)] + Data
    return self.sr_apdu(ENVELOPPE)

def SEARCH_RECORD(self, P1=0x00, P2=0x00, Data=[]):
    ""
    APDU command to search pattern in the current EF file with record structure
    P1: record number
    P2: type of search
    Data: list of bytes describing a pattern to search for
    call sr_apdu method
    ""
    SEARCH_RECORD = [self.CLA, 0xA2, P1, P2, len(Data)] + Data
    return self.sr_apdu(SEARCH_RECORD)

def DISABLE_CHV(self, P1=0x00, P2=0x00, Data=[]):
    ""
    APDU command to disable CHV verification (such as PIN or password...)
    P1: let to 0x00... or read ISO and ETSI specifications
    P2: type of CHV to disable
    Data: list of bytes for CHV value
    call sr_apdu method
    ""
    DISABLE_CHV = [self.CLA, 0x26, P1, P2, len(Data)] + Data
    return self.sr_apdu(DISABLE_CHV)

def UNBLOCK_CHV(self, P2=0x00, Lc=None, Data=[]):
    ""
    APDU command to unblock CHV code (e.g. with PUK for deblocking PIN)
    P2: type of CHV to unblock
    Lc: Empty or 0x10
    Data: if Lc=0x10, UNBLOCK_CHV value and new CHV value to set
    ""
```python
call sr_apdu method

if Lc is None: UNBLOCK_CHV = [self.CLA, 0x2C, 0x00, P2]
else: UNBLOCK_CHV = [self.CLA, 0x2C, 0x00, P2, 0x10] + Data
return self.sr_apdu(UNBLOCK_CHV)

class SIM(ISO7816):

def __init__(self, type='SIM'):
    initialize like an ISO7816-4 card with CLA=0xA0
    can also be used for USIM working in SIM mode, with <type='USIM'>
    e.g. when you put your new USIM in your old 2G handset
    if type == 'USIM':
        ISO7816.__init__(self, CLA=0x00)
        self.type = 'USIM'
        self.AID = None
    else:
        ISO7816.__init__(self, CLA=0xA0)
        self.type = 'SIM'
        if self.dbg:
            print '[DBG] type definition: %s' % self.type
            print '[DBG] CLA definition: %s' % hex(self.CLA)

def sw_status(self, sw1, sw2):
    status = ISO7816.sw_status(self, sw1, sw2)
    if sw1 == 0x91: status = 'normal processing, with extra info containing a command for the terminal: length of the response data %d' % sw2
    elif sw1 == 0x9E: status = 'normal processing, SIM data download error: length of the response data %d' % sw2
    elif sw1 == 0x9F: status = 'normal processing: length of the response data %d' % sw2
    elif (sw1, sw2) == (0x93, 0x00): status = 'SIM application toolkit busy, command cannot be executed at present'
    elif sw1 == 0x92:
        status = 'memory management'
        if sw2 < 16: status += ': command successful but after %d retry routine' % sw2
        elif sw2 == 0x40: status += ': memory problem'
    elif sw1 == 0x98:
        status = '_referencing management'
        if sw2 == 0x00: status += ': no EF selected'
        elif sw2 == 0x02: status += ': out of range (invalid address)'
        elif sw2 == 0x04: status += ': file ID or pattern not found'
        elif sw2 == 0x08: status += ': file inconsistent with the command'
    elif sw1 == 0x9B:
        status = 'security management'
```

Development tools, Release 2012.06.18
if sw2 == 0x02: status += ': no CHV initialized'
elif sw2 == 0x04: status += ': access condition not fulfilled, at least 1 attempt left'
elif sw2 == 0x08: status += ': in contradiction with CHV status'
elif sw2 == 0x10: status += ': in contradiction with invalidation status'
elif sw2 == 0x40: status += ': unsuccessful CHV verification, no attempt left'
elif sw2 == 0x50: status += ': increase cannot be performed, max value reached'
elif sw2 == 0x62: status += ': authentication error, application specific'
elif sw2 == 0x63: status += ': security session expired'
return status

def verify_pin(self, pin_type=1, pin=''): 
    '''verify CHV1 (PIN code) or CHV2 with VERIFY APDU command
    call VERIFY method
    '''
    if pin_type in [1, 2] and type(pin) is str and len(pin) == 4 and 0 <= int(pin) < 10000:
        PIN = [0x30+int(pin[0]), 0x30+int(pin[1]), 0x30+int(pin[2]), 0x30+int(pin[3]), 0xFF, 0xFF, 0xFF, 0xFF]
        return self.VERIFY(P2=pin_type, Data=PIN)
    else: return self.VERIFY()

def disable_pin(self, pin_type=1, pin=''): 
    '''disable CHV1 (PIN code) or CHV2 with DISABLE_CHV APDU command
    tip: do it as soon as you can when you are working with a SIM / USIM card for which you know
    the PIN!
    call DISABLE method
    '''
    if pin_type in [1, 2] and type(pin) is str and len(pin) == 4 and 0 <= int(pin) < 10000:
        PIN = [0x30+int(pin[0]), 0x30+int(pin[1]), 0x30+int(pin[2]), 0x30+int(pin[3]), 0xFF, 0xFF, 0xFF, 0xFF]
        return self.DISABLE_CHV(P2=pin_type, Data=PIN)
    else: return self.DISABLE()

def unblock_pin(self, pin_type=1, unblock_pin=''): 
    '''unblock CHV1 (PIN code) or CHV2 with UNBLOCK_CHV APDU command and set 0000 value for new PIN
    call UNBLOCK_CHV method
    '''
    if pin_type == 1: pin_type = 0
    if pin_type in [0, 2] and type(unblock_pin) is str and len(unblock_pin) == 4 and 0 <= int(unblock_pin) < 10000:
        UNBL_PIN = [0x30+int(unblock_pin[0]), 0x30+int(unblock_pin[1]), 0x30+int(unblock_pin[2]), 0x30+int(unblock_pin[3]), 0xFF, 0xFF, 0xFF, 0xFF]
        return self.UNBLOCK_CHV(P2=pin_type, Lc=0x10, Data=PIN + [0x30, 0x30, 0x30, 0x30, 0xFF, 0xFF, 0xFF, 0xFF])
    else: return self.UNBLOCK_CHV(P2=pin_type)

def parse_file_SIM(self, Data=[]): 
    '''parse a list of bytes returned when selecting a file
    interpret the content of some informative bytes for right accesses, type / format of file...
    work over the SIM file structure
    '''
    file = {}
    file['size'] = Data[2]*0xFF00 + Data[3]
    file['ID'] = Data[4:6]
    file['type'] = ('RFU', 'MF', 'DF', '', 'EF') [Data[6]]
    file['length'] = Data[12]

2.25. Security
if file['type'] == 'MF' or file['type'] == 'DF':
    file['DF_num'] = Data[14]
    file['EF_num'] = Data[15]
    file['codes_num'] = Data[16]
    file['CHV1'] = ('not initialized', 'initialized')[Data[18] & 0x80] + ': %d attempts remain' % (Data[18] & 0x0F)
    file['unblock_CHV1'] = ('not initialized', 'initialized')[Data[19] & 0x80] + ': %d attempts remain' % (Data[19] & 0x0F)
    file['CHV2'] = ('not initialized', 'initialized')[Data[20] & 0x80] + ': %d attempts remain' % (Data[20] & 0x0F)
    file['unblock_CHV2'] = ('not initialized', 'initialized')[Data[21] & 0x80] + ': %d attempts remain' % (Data[21] & 0x0F)
if len(Data) > 23: file['adm'] = Data[23:]
elif file['type'] == 'EF':
    cond = ('ALW', 'CHV1', 'CHV2', 'RFU', 'ADM_4', 'ADM_5', 'ADM_6', 'ADM_7', 'ADM_8', 'ADM_9', 'ADM_A', 'ADM_B', 'ADM_C', 'ADM_D', 'NEW')
    file['UPDATE'] = cond[Data[8] & 0x0F]
    file['READ'] = cond[Data[8] >> 4]
    file['INCREASE'] = cond[Data[9] >> 4]
    file['INVALIDATE'] = cond[Data[10] & 0x0F]
    file['REHABILITATE'] = cond[Data[10] >> 4]
    file['status'] = ('not read/updatable when invalidated', 'read/updatable when invalidated')
    + ('invalidated', 'not invalidated')[byteToBit(Data[11])][5]
    file['structure'] = ('transparent', 'linear fixed', '', 'cyclic')[Data[13]]
if file['structure'] == 'cyclic': file['INCREASE'] = byteToBit(Data[7])[1]
if len(Data) > 14: file['rec_length'] = Data[14]
return file

def parse_file_USIM(self, Data=[]):
    '''
    parse_file_USIM(Data=[0x12, 0x34, 0x56, 0x89]) -> dict(file)
    parse a list of bytes returned when selecting a file
    interpret the content of some informative bytes for right accesses, type / format of file...
    work over the USIM file structure (quite different from the SIM one)
    '''
    file = {}
    tags = {0x80 : 'size',
            0x81 : 'length',
            0x82 : 'fd',
            0x83 : 'ID',
            0x84 : 'AID',
            0x88 : 'SFI',
            0x8A : 'LCSI',
            0x8B : 'SecAtt',
            0x8C : 'SecAtt',
            0xA5 : 'proprietary',
            0x8A : 'SecAtt',
            0xC6 : 'PIN_status'}
    if Data[0] == 0x62:
        toProcess = Data[2:]
        assert( len(toProcess) == Data[1] )
        while len(toProcess) > 0:
            [T, L, V] = first_TLV_parser(toProcess)
            if T in tags.keys(): T = tags[T]
            file[T] = V
            toProcess = toProcess[L+2:]
        fd_byte = byteToBit(file['fd'][0])
if fd_byte[0:1]+fd_byte[2:5] == [0,0,0,0,0]: file['type'] = 'EF working'
elif fd_byte[0:1]+fd_byte[2:5] == [0,0,0,0,1]: file['type'] = 'EF internal'
elif fd_byte[0:1]+fd_byte[2:8] == [0,1,1,1,0,0,1]: file['type'] = 'EF with BER-TLV'
elif fd_byte[0:1]+fd_byte[2:8] == [0,1,1,1,0,0,0]: file['type'] = 'DF or ADF'
else: file['type'] = 'RFU'
if file['type'][0:2] == 'EF' :
    if fd_byte[5:8] == [0,0,0]: file['structure'] = 'no information'
    elif fd_byte[2:8] == [1,1,1,0,0,1]: file['structure'] = 'BER-TLV'
    elif fd_byte[5:8] == [0,0,1]: file['structure'] = 'transparent'
    elif fd_byte[5:8] == [0,1,0]: file['structure'] = 'linear fixed'
    elif fd_byte[5:8] == [1,1,0]: file['structure'] = 'cyclic'
else: file['structure'] = 'RFU'
else: file['type'] == 'RFU'
if len(file['fd']) == 5: file['rec_length'],file['req_num'] = file['fd'][3],file['fd'][4]
if 'size' in file.keys():
    #print 'DEBUG:', file['size']
    size = 0
    for i in range(len(file['size'])): size += file['size'][i] * pow(256,len(file['size'])-i-1)
    file['size'] = size
if self.dbg: print '[DBG] file
return file

def select(self, Data=[0x3F, 0x00]):
    '''
    self.select (Data=[0x.., 0x..]) -> dict(file) on success, None on error
    select the file
    if processing correct: get response with info on the file
    if processing correct and EF file: reads the data in the file
    works in SIM and USIM fashion
    if error, returns None
    else returns the data dictionary: check parse_file_(U)SIM methods
    last apdu available from the attribute self.last_response
    '''
    # select file and check SW; if error, returns None, else get response
    if self.type == 'SIM':
        self.last_response = self.SELECT_FILE(Data=Data)
        type_res = 0x9F
    elif self.type == 'USIM':
        self.last_response = self.SELECT_FILE(P2=0x04, Data=Data)
        type_res = 0x61
    if self.last_response[2][0] != type_res:
        if self.dbg: print '[DBG] %s' % self.last_response
        return None
    # get response and check SW; if error, return None, else parse file info
    self.last_response = self.GET_RESPONSE(Le=self.last_response[2][1])
    if self.last_response[2] !=[0x90, 0x00]:
        if self.dbg: print '[DBG] %s' % self.last_response
        return None
    data = self.last_response[3]
    if self.type == 'SIM': file = self.parse_file_SIM(data)
    elif self.type == 'USIM': file = self.parse_file_USIM(data)
if file['type'][0:2] == 'EF':
    # read EF transparent data
    if file['structure'] == 'transparent':
        self.last_response = self.READ_BINARY(Le=file['size'])
        if self.last_response[2] != (0x90, 0x00):
            if self.dbg: print '[DBG] $s' % self.last_response
                return None
        file['data'] = self.last_response[3]
    # read EF cyclic / linear all records data
    elif file['structure'] != 'transparent':
        file['data'] = []
        # for record data: need to check the number of recordings stored in the file, and iterate
        for i in range((file['size'] / file['rec_length']) + 1):
            self.last_response = self.READ_RECORD(P1=i+1, P2=0x04, Le=file['rec_length'])
            if self.last_response[2] != (0x90, 0x00):
                if self.dbg: print '[DBG] $s' % self.last_response
                    file['data'].append(self.last_response[3])

    # finally returns the whole file dictionary, containing the ['data'] key for EF file
    return file

def run_gsm_alg(self, RAND=16*[0x00]):
    '''
    self.run_gsm_alg( RAND ) -> ( SRES, Kc )
    RAND : list of bytes, length 16
    SRES : list of bytes, length 4
    Kc : list of bytes, length 8
    run GSM authentication algorithm: accepts any kind of RAND (old GSM fashion!)
    feed with RAND 16 bytes value
    return a 2-tuple with SRES and Kc, or None on error
    '''
    if len(RAND) != 16:
        print '[WNG] needs a 16 bytes input RAND value'
        return None
    # select DF_GSM directory
    DF_GSM = self.select([0x7F, 0x20])
    if self.last_response[2] != (0x90, 0x00):
        if self.dbg: print '[DBG] $s' % self.last_response
            return None
    # run authentication
    self.last_response = self.INTERNAL_AUTHENTICATE(P1=0x00, P2=0x00, Data=RAND)
    if self.last_response[2][0] != 0x9F:
        if self.dbg: print '[DBG] $s' % self.last_response
            return None
    # get authentication response
    self.last_response = self.GET_RESPONSE(Le=self.last_response[2][1])
    if self.last_response[2] != (0x90, 0x00):
        if self.dbg: print '[DBG] $s' % self.last_response
            return None
    SRES, Kc = self.last_response[3][0:4], self.last_response[3][4:]
    return [ SRES, Kc ]

def get_imsi(self):
self.get_imsi() -> string(IMSI)

read IMSI value at address [0x6F, 0x07]
return IMSI string on success or None on error

# select DF_GSM for SIM card
if self.type == ‘SIM’:
    DF_GSM = self.select([0x7F, 0x20])
    if self.last_response[2] != (0x90, 0x00):
        if self.dbg: print ‘[DBG] %s’ % self.last_response
        return None

# select appropriate USIM application directory for USIM card
if self.type == ‘USIM’ and self.AID:
    EF_DIR = self.select([0x2F, 0x00])
    if EF_DIR:
        self.AID = EF_DIR[‘data’][0][4:4+EF_DIR[‘data’][0][3]]

# select IMSI file
imsi = self.select([0x6F, 0x07])
if self.last_response[2] != (0x90, 0x00):
    if self.dbg: print ‘[DBG] %s’ % self.last_response
    return None

# and parse the received data into the IMSI structure
if ‘data’ in imsi.keys() and len(imsi[‘data’]) == 9:
    imsi_str =’
    imsi_str += str(imsi[‘data’][1] >> 4)
    for i in range(2,9):
        imsi_str += str(imsi[‘data’][i] & 0x0F )
        imsi_str += str(imsi[‘data’][i] >> 4)
    return imsi_str

# if issue with the content of the DF_IMSI file
if self.dbg: print ‘[DBG] %s’ % self.last_response
return None

# DOCUMENTATION TO BE CONTINUED HERE !!!!!

class USIM(SIM):
    ’’’
    define attributes, methods and facilities for ETSI / 3GPP USIM card
    check USIM specifications in ETSI TS 102.221 and 3GPP TS 31.102
    inherit methods and objects from SIM class
    use self.dbg = 1 or more to print live debugging information
    ’’’
    AID_RID = [
        ([0xA0, 0x00, 0x00, 0x00, 0x09], ‘GSM’),
        ([0xA0, 0x00, 0x00, 0x00, 0x87], ‘3G’),
    ]
    AID_app_code = [
        ([0x00, 0x00], ‘Reserved’),
        ([0x00, 0x01], ‘’),
        ([0x00, 0x02], ‘SIM Toolkit’),
        ([0x00, 0x03], ‘SIM API for JavaCard’),
        ([0x00, 0x04], ‘Tetra’),
    ]
Development tools, Release 2012.06.18

([0x10, 0x01], 'UICC'),
([0x10, 0x02], 'USIM'),
([0x10, 0x03], 'USIM Toolkit'),
]

AID_country_code = [
([0xFF, 0x33], 'France'),
([0xFF, 0x44], 'United Kingdom'),
([0xFF, 0x49], 'Germany'),
]

def __init__(self, type='USIM'):
    '''
    initialize like an ISO7816-4 card with CLA=0x00
    and then select the 1st AID (Application ID) available
    read from the EF_DIR file:
    >>> should be the USIM one
    can also be used for USIM working in SIM mode, with <type='SIM'>
    e.g. when you put your new USIM in your old 2G handset
    '''
    self.type = type
    if type == 'SIM': ISO7816.__init__(self, CLA=0xA0)
    elif type == 'USIM': ISO7816.__init__(self, CLA=0x00)
    if self.dbg:
        print '[DBG] type definition: %s' % self.type
        print '[DBG] CLA definition: %s' % hex(self.CLA)
        print '[DBG] EF_DIR file selection and reading...'

EF_DIR = None
self.AID = []

EF_DIR = self.select([0x2F, 0x00])
if self.dbg:
    print '[DBG] EF_DIR: %s' % EF_DIR
if EF_DIR is not None:
    AID_value = None
    # loop to parse AID address and AID data referenced in EF_DIR
    for d in EF_DIR['data']:
        if len(d) > 6 \
            and d[0] == 0x61 \
            and d[2] == 0x4F:
            AID_value = d[4:4+d[3]]
        if self.dbg:
            print '[DBG] found AID value in EF_DIR: %s' % AID_value
    self.AID.append( AID_value )
    # parse AID and select the right one for 3G USIM
    # then:
    self.last_response = self.SELECT_FILE(P1=0x04, P2=0x04, Data=AID_value)
    if self.last_response[2][0] != 0x61:
        print '[ERR] unable to select file at AID value: %s' % AID_value
        if self.dbg:
            print '[DBG] %s' % self.last_response
    else:
        self.last_response = self.GET_RESPONSE(Le=self.last_response[2][1])
        if self.last_response[2] != (0x90, 0x00):
            print '[ERR] unable to get response correctly from AID value: %s' % value
if self.dbg: print '[DBG] %s' % self.last_response
else:
    self.AID.append(self.last_response[3])

if len(self.AID) == 0:
    print '[ERR] no AID found from AID address in EF_DIR'
else:
    print '[+] USIM AID selection: %s' % self.AID
else:
    print '[ERR] unable to select EF_DIR\n%s' % self.last_response

def get_CS_keys(self):
    '''
    get_CS_keys() -> [KSI, CK, IK]
    read CS UMTS keys at address [0x6F, 0x08]
    return list of 3 keys, each are list of bytes, on success (or eventually the whole file dict)
    or None on error
    '''
    EF_KEYS = self.select([0x6F, 0x08])
    if self.last_response[2] == (0x90, 0x00):
        if len(EF_KEYS['data']) == 33:
            KSI, CK, IK = EF_KEYS['data'][0:1], EF_KEYS['data'][1:17], EF_KEYS['data'][17:33]
            print '[+] Successful CS keys selection: Get [KSI, CK, IK]
            return [KSI, CK, IK]
        else:
            return EF_KEYS
    else:
        return None

def get_PS_keys(self):
    '''
    get_PS_keys() -> [KSI, CK_PS, IK_PS]
    read PS UMTS keys at address [0x6F, 0x09]
    return list of 3 keys, each are list of bytes, on success (or eventually the whole file dict)
    or None on error
    '''
    EF_KEYSPS = self.select([0x6F, 0x09])
    if self.last_response[2] == (0x90, 0x00):
        if len(EF_KEYSPS['data']) == 33:
            KSI, CK, IK = EF_KEYSPS['data'][0:1], EF_KEYSPS['data'][1:17], EF_KEYSPS['data'][17:33]
            print '[+] Successful PS keys selection: Get [KSI, CK, IK]
            return [KSI, CK, IK]
        else:
            return EF_KEYSPS
    else:
        return None

def get_GBA_BP(self):
    '''
    get_GBA_BP() -> [[RAND, B-TID, KeyLifetime], ...], Length-Value parsing style
    read EF_GBABP file at address [0x6F, 0xD6], containing RAND and associated B-TID and KeyLifetime
    return list of list of bytes on success (or eventually the whole file dict if the format is strange)
    or None on error
    '''
    EF_GBABP = self.select([0x6F, 0xD6])
    if self.last_response[2] == (0x90, 0x00):
if len(EF_GBABP['data']) > 2:
    #RAND, B_TID, Lifetime = LV_parser( EF_GBABP['data'] )
    print('[+] Successful GBA_BP selection: Get list of [RAND, B-TID, KeyLifetime]'
    #return (RAND, B_TID, Lifetime)
    return LV_parser( EF_GBABP['data'] )
else: return EF_GBABP

def update_GBA_BP(self, RAND, B_TID, key_lifetime):
    '''
    self.get_GBA_BP() -> void (or EF_GBABP file dict if RAND not found)
    read EF_GBABP file at address [0x6F, 0xD6],
    check if RAND provided is referenced, and update the file structure with provided B-TID and key lifetime
    return nothing (or eventually the whole file dict if the RAND is not found)
    '''
    GBA_BP = self.get_GBA_BP()
    for i in GBA_BP:
        if i == RAND:
            print('[+] RAND found in GBA_BP'
            # update transparent file with B_TID and key lifetime
            self.last_response = self.UPDATE_BINARY( P2=len(RAND)+1, Data=[len(B_TID)]+B_TID+[len(key_lifetime)]+key_lifetime)
            if self.dbg: print('[DBG] %s' % self.last_response
            if self.last_response[2] == 0x90:
                print('[+] Successful GBA_BP update with B-TID and key lifetime'
                if self.dbg: print('[DBG] new value of EF_GBA_BP:
            else:
                '[+] RAND not found in GBA_BP'
    return GBA_BP

def get_GBA_NL(self):
    '''
    self.get_GBA_NL() -> [[NAF_ID, B-TID], ...] , Tag-Length-Value parsing style
    read EF_GBANL file at address [0x6F, 0xDA], containing NAF_ID and B-TID
    return list of list of bytes vector on success (or eventually the whole file dict if the format is strange)
    or None on error
    '''
    EF_GBANL = self.select( [0x6F, 0xDA] )
    if self.last_response[2] == (0x90, 0x00):
        if len(EF_GBANL['data'][0]) > 2:
            # This is Tag-Length-Value parsing, with 0x80 for NAF_ID and 0x81 for B-TID
            values = []
            for rec in EF_GBANL['data']:
                NAF_ID, B_TID = [], []
                
                while len(rec) > 0:
                    tlv = first_TLV_parser( rec )
                    if tlv[1] > 0xFF:
                        rec = rec[ tlv[1]+4 :
                    else:
                        rec = rec[ tlv[1]+2 :
                    if tlv[0] == 0x80: NAF_ID = tlv[2]
                    elif tlv[0] == 0x81: B_TID = tlv[2]
                
                values.append( [NAF_ID, B_TID] )
            print('[+] Successful GBA_NL selection: Get list of [NAF_ID, B-TID]'
            #return (NAF_ID, B_TID)
return values
else: return EF_GBANL
else: return None

def authenticate(self, RAND=[], AUTN=[], ctx='3G'):
    
    """self.authenticate(RAND, AUTN, ctx='3G') -> [key1, key2...] , Length-Value parsing style
    run the INTERNAL AUTHENTICATE command in the USIM with the right context:
    ctx = '2G', '3G', 'GBA' (MBMS or other not supported at this time)
    RAND and AUTN are list of bytes; for '2G' context, AUTN is not used
    return a list containing the keys (list of bytes) computed in the USIM, on success:
    [RES, CK, IK (, Kc)] or [AUTS] for '3G'
    [RES] or [AUTS] for 'GBA'
    [RES, Kc] for '2G'
    or None on error
    ""
    # prepare input data for authentication
    if ctx in ('3G', 'VGCS', 'GBA', 'MBMS') and len(RAND) != 16 and len(AUTN) != 16:
        return 'needs a 16 bytes input for RAND and AUTN values'
    if ctx == 'GBA': input = [0xDD]
    else: input = []
    input.extend( [len(RAND)] + RAND + [len(AUTN)] + AUTN )

    if ctx == '3G':
        P2 = 0x81
    elif ctx == 'VGCS':
        P2 = 0x82
        print '[+] Not implemented. Exit.'
        return -1
    elif ctx == 'MBMS':
        print '[+] Not implemented. Exit.'
        return -1
    elif ctx == 'GBA':
        P2 = 0x84
    else:
        # and also, if ctx == '2G'
        P2 = 0x80
        if len(RAND) != 16: return 'needs a 16 bytes input for RAND value'
        # override input value for 2G authent
        input = [len(RAND)] + RAND

    self.last_response = self.INTERNAL_AUTHENTICATE(P2=P2, Data=input)

    if self.last_response[2][0] in (0x9F, 0x61):
        self.last_response = self.GET_RESPONSE(Le=self.last_response[2][1])

    if self.last_response[2] == (0x90, 0x00):
        val = self.last_response[3]
        if P2 == 0x80:
            print '[+] Successful 2G authentication. Get [RES, Kc]'
            values = LV_parser(val)
            # returned values are (RES, Kc)
            return values
        if val[0] == 0xDB:
            if P2 == 0x81:
                print '[+] Successful 3G authentication. Get [RES, CK, IK{, Kc}]'
                values = LV_parser(val[1:])
            else:
                print '[+] Successful GBA authentication. Get [RES]'
# returned values can be (RES, CK, IK) or (RES, CK, IK, Kc)
return values

elif val[0] == 0xDC:
    print '[+] Synchronization failure. Get [AUTS]'
    values = val[2:val[1]]
    return values

else:
    return None

def GBA_derivation(self, NAF_ID=[], IMPI=[]):
    '''
    self.GBA_derivation(NAF_ID, IMPI) -> [Ks_ext_naf]

    run the INTERNAL AUTHENTICATE command in the USIM with the GBA derivation context:
    NAF_ID is a list of bytes (use stringToByte())
    "NAF domain name"||"security protocol id", eg: "application.org"||"0x010001000a" (> TLS with RSA and SHA)
    IMPI is a list of bytes
    "IMSI@ims.mncXXX.mccYYY.3gppnetwork.org" if no IMS IMPI is specifically defined in the USIM
    return a list with GBA ext key (list of bytes) computed in the USIM:
    [Ks_ext_naf]
    Ks_int_naf remains available in the USIM for further GBA_U key derivation
    or None on error
    '''
    # need to run 1st an authenicate command with 'GBA' context, so to have the required keys in the USIM
    P2 = 0x84
    input = [0xDE] + [len(NAF_ID)] + NAF_ID + [len(IMPI)] + IMPI
    self.last_response = self.INTERNAL_AUTHENTICATE(P2=P2, Data=input)

    if self.last_response[2][0] in (0x9F, 0x61):
        self.last_response = self.GET_RESPONSE(Le=self.last_response[2][1])

    if self.last_response[2] == (0x90, 0x00):
        val = self.last_response[3]
        if val[0] == 0xDB:  # not adapted to 2G context with Kc, RES
            print '[+] Successful GBA derivation. Get [Ks_EXT_NAF]'
            values = LV_parser(val[1:])
            return values
        else:
            return None
    return None

pcsc-tools  See Also:

- http://ludovic.rousseau.free.fr/softwares/pcsc-tools/README
- http://ludovic.rousseau.free.fr/softwares/pcsc-tools/smartcard_list.txt

Some tools to be used with smart cards and PC/SC  This archive contains some tools usefull for a PC/SC user.
PC/SC lite from MUSCLE.

The tools provided are:
pcsc_scan (Ludovic Rousseau <ludovic.rousseau@free.fr>) regularly scans every PC/SC reader connected to the host if a card is inserted or removed a “line” is printed

For example:

PC/SC device scanner
V 1.4.5 (c) 2001-2006, Ludovic Rousseau <ludovic.rousseau@free.fr>
PC/SC lite version: 1.3.1
0: GemPC410 0 0
1: GemPC430 0 0

Wed Aug 21 10:08:02 2002
Reader 0 (GemPC410 0 0)
Card state: Card removed,

Reader 0 (GemPC410 0 0)
Card state: Card inserted,

ATR: 3B 6D 00 FF 00 31 80 71 8E 64 48 D5 02 00 82 90 00
+ TS = 3B --> Direct Convention
+ T0 = 6D, Y(1): 0110, K: 13 (historical bytes)
  TB(1) = 00 --> Programming Param P: 0, I: 0
  TC(1) = FF --> Extra guard time: 255
+ Historical bytes: 00 31 80 71 8E 64 48 D5 02 00 82 90 00
Category indicator byte: 00 (compact TLV data object)
  Tag: 3, len: 1 (card service data byte)
    Card service data byte: 80
    - Application selection: by full DF name
    - EF.DIR and EF.ATR access services: by GET RECORD(s) command
    - Card with MF
  Tag: 7, len: 1 (card capabilities)
    Selection methods: 8E
    - DF selection by full DF name
    - Implicit DF selection
    - Short EF identifier supported
    - Record number supported
  Tag: 6, len: 4 (pre-issuing data)
    Data: 48 D5 02 00
    Mandatory status indicator (3 last bytes)
    LCS (life card cycle): 82 (Proprietary)
    SW: 9000 (Normal processing.)

Possibly identified card:
3B 6D 00 FF 00 31 80 71 8E 64 48 D5 02 00 82 90 00
  Blue for Business, American Express@Business

Reader 0 (GemPC410 0 0)
Card state: Card removed,

pcsc-tools versions

pcsc-tools versions  See Also:
http://ludovic.rousseau.free.fr/softwares/pcsc-tools/index.html

2.25. Security 999
pcesc-tools 1.4.18 18 December 2011

Hello,

I just released a new version of pcesc-tools. No new feature but some enhancements.

If you do not know what pcesc-tools is, it contains 4 tools:

- **pcsc_scan(1)** regularly scans every PC/SC reader connected to the host if a card is inserted or removed a “line” is printed.
- **ATR_analysis(1)** is a Perl script used to parse the smart card ATR. This script is called (by default) by pcsc_scan.
- **scriptor(1)** is a Perl script to send commands to a smart card using a batch file or stdin.
- **gscriptor(1)** the same idea as scriptor.pl(1) but with a Perl-Gtk2 GUI.

An equivalent of ATR_analysis is available online at [http://smartcard-atr.appspot.com/](http://smartcard-atr.appspot.com/)

**Changes**

1.4.18 - 18 December 2011, Ludovic ROUSSEAU

- gscriptor: Display hex dumps in lines of 16 bytes instead of 17
- gscriptor: Display bytes of value 0x20 as ‘ ‘ instead of ‘.’
- scriptor: Display lines of 16 bytes instead of 24
- 223 new ATRs
- pcsc_scan: Correctly detect reader Plug and Play support

1.4.17 - 18 August 2010, Ludovic ROUSSEAU

- 153 new ATRs
- Allow to build with pcesc-lite >= 1.6.2

1.4.16 - 12 January 2010, Ludovic ROUSSEAU

- 153 new ATR
- pcsc_scan.c: check for PnP support at run time instead of using a #define
- ATR_analysis: use curl instead of wget on Darwin
- gscriptor: ReaderConfig(): escape metacharacters []() in the reader name when using reader name as a pattern matching

1.4.15 - 9 January 2009, Ludovic ROUSSEAU

- 68 new ATR
- ATR_analysis:
- truncate the ATR if extra bytes are present (like on Leopard with an ATR padded with 0 up to 33 bytes)
- add value for Di=7 (7816-3:2006 page 19)
- check if Fi is RFU when calculating baud rate
- display the max frequency associated with Fi
- pcsc_scan.c: use “?PnP?Notification” mechanism when possible
1.4.14 - 11 May 2008, Ludovic ROUSSEAU

- 24 new ATR
- gscriptor, scriptor: use SCARD_SHARE_SHARED instead of SCARD_SHARE_EXCLUSIVE to not lock the card
- ATR_analysis: add support for ATR with : as byte separator (ATR reported by "opensc-tool –atr" for example)

1.4.13 - 23 March 2008, Ludovic ROUSSEAU

- 29 new ATR
- pcsc_scan: avoid a bug when the last reader is removed

epassportviewer  See Also:

- [http://code.google.com/p/epassportviewer/](http://code.google.com/p/epassportviewer/)

In 2004, the ICAO published a new version of Doc9303 that defines the specifications of the electronic passports, known as ePassports.

These passports, easily recognizable with their logo on the front cover, contain a passive contactless chip featured with some cryptographic mecanisms.

ePassport Viewer is a GPL-friendly tool to read and checks ePassports, developed by Jean-Francois Houzard and Olivier Roger during their Master Thesis achieved in the Information Security Group of the UCL in Belgium. ePassport Viewer is currently available as beta version.

Open SC  OpenSC provides a set of libraries and utilities to work with smart cards. Its main focus is on cards that support cryptographic operations, and facilitate their use in security applications such as authentication, mail encryption and digital signatures. OpenSC implements the PKCS#11_ API so applications supporting this API (such as Mozilla Firefox and Thunderbird) can use it.

On the card OpenSC implements the PKCS#15_ standard and aims to be compatible with every software/card that does so, too.

globalplatform  See Also:


The GlobalPlatform card specification provides a standard for the management of the contents on a smart card. Mainly this comprises the installation and the removal of applications.

A good overview of the GlobalPlatform card specification is given on this page: GlobalPlatform card specification. Due to the missing support of free software supporting the GlobalPlatform card specification this project was founded by Karsten Ohme.

This project offers a C library and a command line shell.

You can support this project with donations. The money will be used for the support of the ongoing development e.g. buy of smart cards, NFC phones and card readers.

globalplatform news

de  Martin Paljak <martin@martinpaljak.net>
heure de l’expéditeur  Envoyé à 09:33 (GMT+02:00). Heure locale : 11:40.
répondre à  MUSCLE <muscle@lists.musclecard.com>
à  MUSCLE <muscle@lists.musclecard.com>
Hello,

> There are two issues that I find interesting, but I’m not sure
> if you want to put information about these in the wiki:
> a) where can people get javacard cards at reasonable price?

I used to buy a bunch of Oberthur Cosmos V7 cards from smartcardfocus.com [1] but they seem to be unavailable at the moment. cryptoshop.com seem to sell JCOP 41 v2.3.1 cards but I’ve not ordered from them before (should try though).

There are probably other sources as well, if you find them, OpenSC FAQ has an entry “Where can I buy smart cards?” [2]

Why shouldn’t this information be in a wiki?

> b) what about the standards of the javacards, e.g. JCOP?
> as far as I know there is some functionality you can’t access
> once you get the finished card (e.g. change ATR, UID etc.).
> the documentation is closed under NDA, but if anyone
> found a copy available on the net, that would be interesting...

JCOP is not a standard, it is an implementation of a JavaCard platform by IBM / NXP. There are other vendors as well.

JavaCard specs as well as GlobalPlatform specs and API-s are freely available. Both can be downloaded as .zip files but extracted versions are also available from google:


There are two issues that I find interesting, but I’m not sure
if you want to put information about these in the wiki:

a) where can people get javacard cards at reasonable price?

You’re talking about not initialized cards. First, it’s near impossible to find cards in this state, so the doc would be useless, and second, the commands available in these modes are generally secret and not even available with NDAs outside the manufacturer. In these initialization states, cards are not javacards yet, they are totally proprietary objects, as you might already know.
Additionnaly, please avoid talking about “JCOP” when referring to Java Card. JCOP is a (not so good for all applications) NXP product, and fortunately, there are bunch of other manufacturers and cards, such as the Oberthur Cosmo, the Gemalto GXP/GCX and some others at G&D, for example.

All of these cards follow a single standard, namely Java Card.

Sebastien

smartcard on android

See Also:

- https://code.google.com/p/seek-for-android/
- https://code.google.com/p/seek-for-android/wiki/SmartcardAPI
- https://code.google.com/p/seek-for-android/wiki/UsingSmartCardAPI

...smartcard_list:

The smartcard list See Also:

http://ludovic.rousseau.free.fr/softwares/pcsc-tools/smartcard_list.txt

ICAO

ICAO

ICAO annexes

Links

- http://www2.icao.int/en/MRTD/Pages/default.aspx
- http://code.google.com/p/pypassport/
- http://www2.icao.int/en/MRTD/Pages/Doc9393.aspx

Doc 9303 Doc 9303, Machine Readable Travel Documents


Volume 2 Sets forth the specifications for biometric enhancement of the MRP to become an “e-Passport”.

Part 2 - Machine Readable Visas The third Edition was published in 2005. Specifications provide for a visa format in two sizes - Format A, sized to fill a passport page, and the smaller format B. Like the MRP the machine readable visa is a standard format consisting of a visual inspection zone and a machine readable zone.

However, the Third Edition requires that a space be provided for a portrait of the holder, and fewer layout options than the previous edition allowed.

In particular, this third edition incorporates the optional new globally interoperable standard for biometric identification of the holder and for the storage of the associated data on a contactless integrated circuit.

Volume 1 is an updated version of the second edition containing all the specifications required for a State wishing to issue a machine readable official travel document without the incorporation of machine-assisted biometric identification.

Volume 2 contains the specifications for enhancing the machine readable official travel document with the globally interoperable system of biometric identification and its associated data storage utilizing a contactless integrated circuit.

Specifications common to all parts are:

- recommended transliterations of national characters, and
- 3-letter codes for nationality and the issuing entity.

Presse

RFID à la Fédération française de pétanque : pour ne plus perdre la boule  Source: http://www.filrfid.org/article-31881662.html

Entretien avec Daniel READ, nouveau trésorier de la Fédération Française de Pétanque et Jeu Provençal (FFPJ), chargé, sous la responsabilité de Claude STIRMEL responsable fédérale de l’informatique, de la mise en oeuvre de l’application RFID et du lecteur CL 1356 A d’Orcanthis.

M. READ, quelles sont les raisons qui ont poussé la FFPJP à choisir les technologies RFID pour la gestion de ses licences et concours ?

C’est très simple. La FFPJP permettait déjà à l’ensemble de ses comités départementaux d’accéder et mettre à jour la base de données nationale de ses licenciés directement via Internet. Pour autant, sur le terrain tout était fait à la main : vérification des licences, pointage et inscription des équipes et même les tirages au sort. C’était très long, fastidieux et surtout source d’erreurs. Si l’on considère que chaque année ce sont plus de 100.000 concours qui s’organisent partout en France pour près de 360.000 licenciés, il fallait trouver une solution. Avec la RFID, toutes ces opérations sont automatisées avec une marge d’erreur proche de zéro.

Concrètement, comment avez-vous procédé ?

A partir de notre base de données, nous avons stocké sur des cartes à puce les informations nécessaires à la bonne identification des licenciés. Lors des concours, un simple passage de la carte sur le lecteur CL 1356 A d’Orcanthis permet de savoir immédiatement si le licencié est à jour de sa cotisation annuelle, a fourni un certificat médical à jour, s’il appartient à la bonne catégorie et surtout cela nous évite d’avoir à ressaisir ses coordonnées sur un listing papier : nom, prénom, numéro d’adhérent… En résulte, un énorme gain de temps et plus de fiabilité pour les organisateurs et les joueurs. Il nous est désormais possible d’enregistrer 64 doublettes (128 joueurs) en 10 minutes, là où auparavant il nous fallait plus d’une heure !

Que vous a apporté le lecteur CL 1356 A par rapport à d’autres solutions proposées sur le marché ?

Le lecteur CL 1356 A est très pratique parce qu’il est petit, léger et très facilement transportable. Il offre en plus plusieurs fonctionnalités parfaitement adaptées à gestion des concours. Simplement branché sur secteur, il indique avec un voyant lumineux si la licence du pétanqueur est bien à jour (voyant rouge s’il manque un élément) ; relié à un ordinateur, on se connecte sur notre base de données et on peut échanger directement des informations entre la carte
et le fichier national. Les licenciés peuvent par exemple nous informer en direct d’un changement d’adresse ou nous apporter leur certificat médical sans avoir à faire de démarche supplémentaire.

Vous avez effectué votre premier test grandeur nature le 22 septembre dernier lors du championnat de France des Vétérans. Tout s’est bien déroulé ?

ParfaITEMENT, alors que cette compétition regroupait tout de même 128 triplettes ! Nous en avons d’ailleurs profité pour comparer les deux modes de gestion, à la main et avec le lecteur.

Résultat : alors que les vérifications manuelles n’ont relevé aucune non-conformité, le lecteur, lui, a mis à jour 3 anomalies : 1 joueur était le frère du véritable licencié, 1 n’appartenait pas au même club que ses partenaires ce qui est interdit et un troisième joueur n’était pas un vétéran. Nous sommes donc très satisfaits de ces résultats.

Avez-vous des projets de développement du système à d’autres applications ?

Effectivement. Avec le lecteur CL 1356 A d’Orcanthus nous pouvons pleinement utiliser les potentialités des cartes à puce. De nombreuses applications additionnelles peuvent être envisagées : des partenariats commerciaux permettant aux licenciés d’avoir des réductions ou des cadeaux, mais nous souhaiterions aussi mettre en place un système de licence à points (lorsqu’un arbitre est contraint de sanctionner un joueur sa décision est inscrite sur la carte et en cas de nécessité des mesures pourraient être prises par la fédération). Après, d’autres projets, touchant à la sécurité et à l’assurance des équipements, sont en cours de réflexion pour gérer les entrées/sorties des boulodromes par exemple.

Par ailleurs, notre exemple fait déjà école puisque d’autres fédérations sportives de pétanque au niveau international commencent à réfléchir à cette solution et d’autres sports pensent s’ouvrir au système, le rugby, notamment.

Salons

Salons cartes  See Also:
- http://www.cartes.com/

Salon cartes 2011  See Also:

Infos générales

NFC  Le sujet principal du salon a été le NFC. A l’horizon de 2 ans, tous les téléphones mobiles seront NFC. Les premières applications ciblées sont la suppression du pass Navigo et le paiement type Moneo ou Carte Bleue par le téléphone mobile.

Autres applications démontrées:
- location de voiture avec déverrouillage par téléphone NFC (site pilote organisé par le Grand Toulouse avec 5 Twingo)
Plus d’info sur NFC : voir article dans magazine “Electroniques”.

**VHBR**

- Apparition des premiers lecteurs et cartes sans contact VHBR (débit 3.4MBits/s lecteur vers carte et 6.8MBits/s carte vers lecteur).
- Globalement, peu d’innovation au niveau matériel. Les valeurs ajoutées se développent autour des services et du web.

**Compte-rendu technique**

**Inside**

Présentation d’une SIM avec antenne intégrée pour ajout de la fonction NFC dans les téléphones mobiles non équipés.

Présentation de la première puce sécurisée sans contact VHBR.

**NXP**

Sortie officielle du CLRC663 successeur du CLRC632.

**DUALI**

SIC9310 : chip reader 13,56MHz présenté comme compatible NXP CLRC ???

**3M**

Lecteur portable MD6000 : ISO7816 / ISO 14443 / Optical Fingerprint / OCR

**AMS**

AS3993 nouveau chip UHF dédié low power / mobile

AS3911 [nouveau chip RF] → support du VHBR, 1W RF en sortie symétrique, fonction Squelch, détection présence carte low power (5µA), compensation active d’adaptation d’antenne par capacité commutée. → dispose d’une mesure de niveau RSSI → peut être applicable à RYB si étude lecteur 15693 long range ? (fact sheet disponible à mon bureau).

**VIRDI**

Capteur Optique OEM avec détection de Fake Finger.

**XIRING**

OPA en cours d’INGENICO sur XIRING.

**security tools**

**pyew security tool**

See Also:

Pyew is a (command line) python tool like radare and iew oriented, mainly, to analyze malware.

It does have support for hexadecimal viewing, disassembly (Intel 16, 32 and 64 bits), PE and ELF file formats (it does code analysis the right way), following direct call/jmp instructions, OLE2 format, PDF format (limited) and more.

It also supports plugins to add more features to the tool.

**Web security, audit**

See Also:

- [http://www.webappsec.org/](http://www.webappsec.org/)

**The Web Application Security Consortium (WASC)**

The Web Application Security Consortium (WASC) is a 501c3 non profit made up of an international group of experts, industry practitioners, and organizational representatives who produce open source and widely agreed upon best-practice security standards for the World Wide Web.

![w3af logo](w3af.png)

**W3af Web Application Attack and Audit Framework**  
W3af is a Web Application Attack and Audit Framework. The project’s goal is to create a framework to find and exploit web application vulnerabilities that is easy to use and extend.

**W3af Short term objectives**  
The project’s short term objectives, which are planned to be achieved in 2009 are:

- Duplicate the amount of w3af community members

  w3af is much more than a piece of software, w3af is a community that breathes Web Application Security. Our goal has to be to duplicate the amount of community members. Community members start threads in the mailing list, promote w3af in information security conferences, contribute with patches, feature requests and bug reports. Without a supporting community, there is no project.

- Become the best Open Source Web Application Scanner

  Right now there is some controversy on which is the best Open Source Web Application Scanner. One of our main short term goals is to become, without a doubt, the best Open Source Web Application Scanner.

- Become the best Open Source Web Application Exploitation Framework

  One of the advantages of being the first Web Application Exploitation Framework is that you are the best “by default”, but the w3af community doesn’t have to forget about Metasploit.
H D Moore and his crew are building a lot of Web Application exploits, and the architecture to support them. Our goal has to be to develop a stable architecture to support all the exploit types and techniques that could appear in the future.

**Long term objectives (our dreams and hopes)** The project’s long term objectives to be achieved in the projects lifetime are:

- Create the biggest community of Web Application Hackers
- Become the best Web Application Scanner
- Become the best Web Application Exploitation Framework
- Combine static code analysis and black box testing into one framework
- Become the nmap for the Web

**Vega (to test the security of web applications)** See Also:

- http://twitter.com/#!/subgraph

Vega is an open source platform to test the security of web applications.

Vega can help you find and validate SQL Injections, Cross-Site Scripting (XSS), inadvertently disclosed sensitive information, and other vulnerabilities.

It is written in Java, GUI based, and runs on Linux, OS X, and Windows.

Vega includes an automated scanner for quick tests and an intercepting proxy for tactical inspection. Vega can be extended using a powerful API in the language of the web: Javascript.

**News**

**july 2011** See Also:


Hey,

We’ve been quietly working for the last few months on a new platform for scanning and pen-testing web applications. It’s called Vega. It’s GUI-based, open source, and includes an automated scanner and intercepting proxy. We launched the beta today at FISL12.

Vega is written in Java, based on Eclipse RCP, and runs nicely on OS X, Linux, and Windows.

Vega is also really extensible: there’s an embedded Javascript interpreter for writing custom modules. The rich API is documented on our Trac wiki. Nearly all of the vulnerability checks are Javascript modules, and it’s very easy to write new ones.

Vega is developed by Subgraph in Montreal and is licensed under the EPL 1.0.

Download Vega 1.0 Beta at http://www.subgraph.com. You can find us in #subgraph on freenode.
Looking forward to hearing beta feedback.

--
David Mirza <dma at subgraph.com>
Subgraph
Vega, the Open Source Web Security Platform
http://www.subgraph.com
Twitter: @subgraph

2.26 Software engineering

2.26.1 Software engineering

Code review

See Also:

- https://secure.wikimedia.org/wikipedia/fr/wiki/Revue_de_code

Code review is systematic examination (often as peer review) of computer source code. It is intended to find and fix mistakes overlooked in the initial development phase, improving both the overall quality of software and the developers’ skills.

Reviews are done in various forms such as pair programming, informal walkthroughs, and formal inspections.

La revue de code est un examen systématique du code source d’un logiciel. Il peut être comparé au processus ayant lieu dans un comité de lecture, l’objectif étant de trouver des bugs ou des vulnérabilités potentielles ou de corriger des erreurs de conception afin d’améliorer la qualité et la sécurité du logiciel.

Une revue de code peut s’appuyer sur la vérification (manuelle ou automatisée) du respect d’un ensemble de règles de programmation.

La revue de code devient de plus en plus une étape à part entière dans tout processus de développement logiciel en particulier dans les méthodes dites agiles comme l’extreme programming.

Gerrit Code review

See Also:

- git gerrit
- Editing sphinx doc on the web

Web based code review and project management for Git based projects.

Gerrit is a web-based tool that is used for code review. Its main features are the side-by-side difference viewing and inline commenting which makes code reviews quick and simple task.

It is used together with Git version control system.
Gerrit allows authorized contributors to submit changes to Git repository, after reviews are done. Contributors can get their code reviewed with a little effort, and change get their changes quickly through the system.

References

- http://gerrit.googlecode.com/svn/documentation/2.1.6/user-upload.html#push_create

Gerrit tools

Git review  See Also:

- https://github.com/openstack-ci/git-review
- git

A git command for submitting branches to Gerrit

git-review is a tool that helps submitting git branches to gerrit for review.

Setup  git-review, by default, looks for a git remote called gerrit, and submits the current branch to HEAD:refs/for/master at that remote.

If the “gerrit” remote does not exist, git-review looks for a file called .gitreview at the root of the repository with information about the gerrit remote.

Assuming that file is present, git-review should be able to automatically configure your repository the first time it is run.

Projects using Gerrit

libusbx uses gerrit  See Also:
http://sourceforge.net/apps/trac/libusbx/milestone/v1.0.12

Contents

- libusbx uses gerrit
  - New features (libusbx 1.0.12)
New features (libusbx 1.0.12)

- Windows: reinstate HID support
- Core: toggleable logging
- Core: topology call
- Core: Add support to USB3 descriptors (#15)
- Ext: Gerrit + Jenkins, with at least Linux and Windows (MinGW) build checks

Qt uses gerrit  See Also:

http://wiki.qt-project.org/Code_Reviews

Participating in the Qt project by contributing and reviewing

This Wiki provides information on how to contribute and review code. It also provides directions to the information on Gerrit and Git used for developing Qt. Contents

1 Overview of the technical landscape
2 Using Git
3 Using Gerrit
4 Troubleshooting
   4.1 Checklist for contributing
   4.2 Troubleshooting Table

Overview of the technical landscape

The servers, database, and tools related to the Qt Open Governance code contribution process as well as their interactions are illustrated in the adjacent picture. The process has been designed to provide increased transparency and quick pass-through for code contributions, and its core is the easy-to-use code review tool, Gerrit.

The tool is available for everybody, and besides using it for its main purpose of contributors and feedback gathering, it can be utilized for studying the code of the other contributors.

Review board  See Also:

- http://www.reviewboard.org/

Review Board is a powerful web-based code review tool that offers developers an easy way to handle code reviews. It scales well from small projects to large companies and offers a variety of tools to take much of the stress and time out of the code review process.

For too long, code reviews have been too much of a chore. This is largely due to the lack of quality tools available, leaving developers to resort to e-mail and bug tracker-based solutions.
We’ve seen a lot of time and energy wasted doing code reviews both in open source projects and at companies. In both cases, code reviews were typically done over e-mail. A significant amount of time was spent in forming review requests, switching between the diff and the e-mail, and trying to understand what parts of the code the reviewer was referring to.

So in an effort to keep our sanity and improve the process both in our open source projects and at companies, we wrote Review Board. We hope it will be useful to your team too so you can focus on what’s important: writing great products.

rietveld  See Also:

Contents
- rietveld
  - Introduction
  - Python-dev example

Introduction  This project shows how to create a somewhat substantial web application using Django on Google App Engine.

In addition, I hope it will serve as a practical tool for the Python developer community, and hopefully for other open source communities.

As I’ve learned over the last two years at Google, where I developed a similar tool named Mondrian, proper code review habits can really improve the quality of a code base, and good tools for code review will improve developers’ life.

Some code in this project was derived from Mondrian, but this is not the full Mondrian tool.

Source code has been published! See the Source tab above.

--Guido van Rossum, Python creator and Google employee

Python-dev example  See Also:
[http://bugs.python.org/review/13882/show](http://bugs.python.org/review/13882/show)

CEI 61508

CEI 61508 introduction  See Also:

Définitions  Sûreté de fonctionnement est la traduction de l’anglais dependability.

FMDS (fiabilité, maintenabilité, disponibilité, sécurité) est une traduction bancale de l’anglais RAMS (Reliability, Availability, Maintainability and Safety, c’est-à-dire dans l’ordre exact, fiabilité, disponibilité, maintenabilité et sécurité, qui donne les initiales FDMS).
Project management

See Also:

- http://en.wikipedia.org/wiki/Project_management

Project management is the discipline of planning, organizing, securing, and managing resources to achieve specific goals.

A project is a temporary endeavor with a defined beginning and end (usually time-constrained, and often constrained by funding or deliverables), undertaken to meet unique goals and objectives, typically to bring about beneficial change or added value.

The temporary nature of projects stands in contrast with business as usual (or operations), which are repetitive, permanent, or semi-permanent functional activities to produce products or services.

In practice, the management of these two systems is often quite different, and as such requires the development of distinct technical skills and management strategies.

The primary challenge of project management is to achieve all of the project goals and objectives while honoring the preconceived constraints.

Typical constraints are scope, time, and budget.

The secondary—and more ambitious—challenge is to optimize the allocation and integrate the inputs necessary to meet pre-defined objectives.

Project management tools

Libreplan See Also:

http://www.libreplan.com/home/

The open source web application for project planning, monitoring and control.

Libreplan versions See Also:

- http://wiki.libreplan.org/twiki/bin/view/LibrePlan/ReleaseS

Libreplan 1.2.1 See Also:

- http://wiki.libreplan.org/twiki/bin/view/LibrePlan/LibrePlan_1_2#LibrePlan_1_2_1
Software quality

Qualité logiciel et test du logiciel

See Also:

http://www.software-tester.ch/PDF-Files/Glossaire%20Version%201.0%20Francais.pdf

Présentation générale des normes ISO

Présentation générale des principales normes ISO concernant la qualité

- ISO 10000 (lignes directrices pour la gestion de la qualité):
  - ISO 10007:2003 Lignes directrices pour la gestion de la configuration (recommandations pour l’utilisation de la gestion de la configuration au sein d’un organisme et planification...)
- ISO 14000 (gestion environnementale)

La Sûreté de Fonctionnement (SdF), CEI 61508

La norme IEC 61508 (sûreté de fonctionnement)

- http://fr.wikipedia.org/wiki/Syst%C3%A8mes_instrument%C3%A9s_de_s%C3%A9curit%C3%A9#Conformit%C3%A9_aux_normes_ANSI.2FISA_S84.01-1996_et_CEI_61508

Normes dérivées:

- CEI 61511 : procédés industriels
- EN 50126, EN 50128, EN 50129 : ferroviaire
- DO-254, DO-178B : avionique

La norme CEI est composée de 7 parties (2500 euros). Le formateur conseille d’acheter:

- la partie 2 (prescription pour les systèmes E/E/PE) relatifs à la sécurité.
- la partie 6 (lignes directrices pour l’application des parties 2 et 3)
- la partie 7 (présentation de techniques et mesures)

Vocabulaire

SdF = dependability (terme anglais). http://fr.wikipedia.org/wiki/Safety_Integrity_Level (Un Safety Integrity Level)

SIL ou niveau de ‘sécurité intégrée’ est défini comme un niveau relatif de réduction de risques inhérents à une fonction de sécurité.
Méthodes d’ingénierie, UML En ce qui concerne les méthodes d’ingénierie de la SdF on retiendra en particulier l’utilisation du formalisme UML (Unified Modeling Language ou “langage de modélisation unifié”) pour la conception. UML 2.3 propose 13 types de diagrammes (9 en UML 1.3). UML n’étant pas une méthode, leur utilisation est laissée à l’appréciation de chacun, même si le diagramme de classes est généralement considéré comme l’élément central d’UML.


Les modèles CMMi et SPICE (ISO/IEC 15504) Le modèle CMMi (Capability Maturity Model + Integration) est utilisé aux États-Unis tandis que le modèle SPICE (Software Process Improvement and Capability dEtermination). Les deux modèles sont assez similaires et visent à l’amélioration des processus de développement logiciel.

Pour SPICE on compte 6 niveaux de maturité:

- niveau 0: processus incomplet
- niveau 1: processus réalisé
- niveau 2: processus géré
- niveau 3: processus établi
- niveau 4: processus prévisible
- niveau 5: processus en optimisation

Pour le CMMi on compte 5 niveaux

- 1: initial (processus non défini)
- 2: géré (processus organisé)
- 3: défini (processus standardisé)
- 4: géré quantitativement (processus prévisible)
- 5: en optimisation (amélioration continue)

Une différence essentielle entre les 2 méthodes est que SPICE est une démarche “auto-certifiante” tandis que pour CMMI il faut passer par une certification. Le formateur conseille donc de commencer par SPICE.

Liens


Ingénierie des exigences (IE), standard IEEE 610.12:1990 See Also:


L’Ingénierie des Exigences (IE) s’applique à tout produit ou service, tout système, sous-système ou autre composant. Elle s’applique aussi bien avec les clients qu’avec les fournisseurs.

Les entrants sont les besoins ou exigences brutes spécifiées par un client. Les sortants sont les documents d’exigences:

- normes
- appel d’offre
- contrat
Le formateur nous conseille de **donner un identificateur unique à chaque exigence et d’utiliser cet identificateur unique dans les échanges avec les clients/fournisseurs.**

Si l’IE est négligée, des besoins client peuvent être incompris par le fournisseur ce qui implique:

- augmentation des coûts et délais de réalisation dû à des développements supplémentaires.
- diminution de la qualité : le produit peut ne pas répondre à certains besoins.

**L’importance de l’IE**

**Base de vérification**

- les travaux d’IE enlèvent toute ambiguïté au niveau des exigences.
- l’IE s’assure d’une implémentation correcte du produit
- l’IE rend possible la mise en œuvre d’une activité de test.

**Base d’acceptation**

- les exigences et leur spécification représentent concrètement les besoins client
- elles permettent un meilleur dialogue client/fournisseur
- elles formalisent l’accord client/fournisseur
- elles constituent également la référence pour les tests de recette et l’acceptation client.

Les exigences doivent être traçables. La terminologie du client doit être utilisée. Les termes vagues sont proscrits: “à confirmer, à définir, acceptable, adéquat, ergonomique, et/ou, état de l’art, règles de l’art, satisfaisant, souvent, suffisamment, trop, typique, etc...” Toute exigence se doit d’être vérifiable (norme 830 et 1233 de IEEE).

**Liens**


**Méthodologies de développement** Présentation de plusieurs modèles de développement:

- modèle élémentaire
- modèle en cascade
- modèle transformationnel
- modèle en spirale
- modèle par incrément/itérations
- modèle en V (le plus utilisé).

Le modèle par incrément/itérations est très intéressant puisqu’il permet de présenter des résultats rapidement et ainsi de détecter les erreurs plus tôt.

Les activités techniques concernant le développement sont:
Chacune de ces activités techniques doit être décomposé en 4 sous-activités élémentaires:

- **E (entrées)** rassemble les critères devant être réunis pour que la tâche démarre
- **T (tâche à réaliser)** : objectif de développement assigné à une personne, une équipe ou un groupe
- **C (contrôle)** contrôle du processus qui assure que le déroulement de la tâche s’effectue conformément aux règles de qualité.
- **S (sorties)** : rassemble les critères devant être satisfaits pour la tâche puisse être déclarée terminée.

En théorie, chaque activité doit se terminer avant que ne commence la suivante. Une phase se termine par la rédaction de documents qui sont vérifiés et validés (tests).

Chaque activité doit être ponctuée d’une revue.

Les activités de spécification et de conception représentent environ 40% de l’effort de développement.

**Norme concernant l’activité de spécification IEEE 830 1998**


**Les tests**


**Constats**

- plus une erreur est détectée tard plus elle est coûteuse à corriger
- le logiciel est un produit où le défaut semble tolérable
- le test souffre souvent en premier des dépassements de délai.
- le test se résume trop souvent à de la simple validation placée en bout de chaîne dans le cycle de développement.
- le développement n’est pas piloté par les tests:
  - exigences rédigées sans se soucier de leur testabilité
  - spécifications volatiles et non gérées efficacement (Ingénierie des Exigences)

**Rôle du test** Le test a pour objectif de détecter les anomalies d’un logiciel.

**Quelques principes pour les tests**

- 1: le test montre la présence de défauts, pas leur absence
- 2: tester de façon exhaustive n’est pas possible
- 3: le test doit démarrer le plus tôt possible dans le cycle de développement
- 4: un développeur ne doit pas tester ses propres programmes
• 5: la définition des sorties ou résultats attendus doit être réfléchie avant l’exécution des tests (plan de tests)
• 6: les jeux de tests doivent être écrits pour des entrées invalides ou incohérentes aussi bien que pour des entrées valides.

Les niveaux de tests

• Tests unitaires. Documents de départ:
  – plan de tests unitaires
  – spécification de test du module
  – conception détaillée

• Tests d’intégration. Documents de départ:
  – plan de tests d’intégration
  – spécification de test de validation
  – conception architecturale

• Tests systèmes. Documents de départ:
  – plan de tests de validation du système
  – spécification du système
  – manuel utilisateur et d’exploitation
  – exigences fonctionnelles

• tests d’acceptation

Techniques de tests statiques (tests sans exécution)

• revue/inspection de spécifications
• revue sur le code (80 % des anomalies logicielles sont levées)

Test statique avec codesonar de Grammatech  http://www.grammatech.com/products/codesonar/

Test statique avec LDRA testbed Design Review  Le formateur fait une démo de LDRA testbed (Design Review) et nous parle de la norme MISRA-C2004 concernant les règles de codage en C.

Il est possible de paramétrer LDRA testbed pour qu’il sorte toutes les lignes de programme ne respectant pas certaines règles MISRA par exemple.

Liens:

• http://www.ldra.com/fr/testbed.asp
• http://www.ldra.com/fr/analysestatique.asp
• http://www.ldra.com/fr/completedownload.asp?id=6
• http://en.wikipedia.org/wiki/MISRA_C
• http://www.ldra.com/fr/misracpp.asp (MISRA C++)
• http://www.ldra.com/fr/completedownload.asp?id=124
Techniques de tests dynamiques (tests avec exécution)

- exécution du programme pour s'assurer du bon fonctionnement
- injection de stimuli en entrée, vérification de l'obtention de résultats attendus en sortie.

Test dynamique avec LDRA


Saisie des fonctions et des données, exécution du code et production d’un compte rendu d’exécution.


- Exécution de tests de haute qualité
- Réduction du coût et des efforts pour les tests de régression
- Identification des anomalies et défaillances du logiciel
- Production d’un ensemble de données de test complet dont la qualité est mesurable et les résultats de test connus
- Réduction des coûts de maintenance au minimum
- Identification des pièces inutiles du système/programme qui peuvent être retirées
- Garantie que les systèmes sont fiables et comportent le moins d’erreurs possibles

Fuzzing  See Also:

- [http://www.fuzzing.org/](http://www.fuzzing.org/)

Fuzzing sulley  See Also:


Sulley is a fuzzer development and fuzz testing framework consisting of multiple extensible components. Sulley (IMHO) exceeds the capabilities of most previously published fuzzing technologies, commercial and public domain.

The goal of the framework is to simplify not only data representation but to simplify data transmission and target monitoring as well. Sulley is affectionately named after the creature from Monsters Inc., because, well, he is fuzzy.

Modern day fuzzers are, for the most part, solely focus on data generation. Sulley not only has impressive data generation but has taken this a step further and includes many other important aspects a modern fuzzer should provide.

Sulley watches the network and methodically maintains records.

Sulley instruments and monitors the health of the target, capable of reverting to a known good state using multiple methods.

Sulley detects, tracks and categorizes detected faults.

Sulley can fuzz in parallel, significantly increasing test speed.

Sulley can automatically determine what unique sequence of test cases trigger faults.

Sulley does all this, and more, automatically and without attendance.
Métriques, métrologie  See Also:
http://pypi.python.org/pypi/pynocle

Les métriques essentielles:
- clarté
- complexité
- portabilité
- maintenabilité
- testabilité
- fiabilité

Métrologie avec LDRA quality review

Points Forts (extrait de http://www.ldra.com/fr/qualityreview.asp)
- Génération automatique de rapports fournissant une documentation précise de la qualité du code
- Visualisation immédiate de la complexité du code
- Aide au recodage pour diminuer la complexité
- Métriques du code organisées into code assessment metrics
- Rapport qualité indiquant les limites des métriques de l’application
- Conséquence: La qualité du code est mesurée et très clairement visualisable

Métrologie avec Source monitor (http://www.campwoodsw.com/)  Signalé par BS (Bruno Simonet).
The freeware program SourceMonitor lets you see inside your software source code to find out how much code you have and to identify the relative complexity of your modules. For example, you can use SourceMonitor to identify the code that is most likely to contain defects and thus warrants formal review. SourceMonitor, written in C++, runs through your code at high speed. SourceMonitor provides the following:
- Collects metrics in a fast, single pass through source files.
- Measures metrics for source code written in C++, C, C#, VB.NET, Java, Delphi, Visual Basic (VB6) or HTML.
- Includes method and function level metrics for C++, C#, VB.NET, Java, and Delphi.
- Offers Modified Complexity metric option.
- Saves metrics in checkpoints for comparison during software development projects.
- Displays and prints metrics in tables and charts, including Kiviat diagrams.
- Operates within a standard Windows GUI or inside your scripts using XML command files.
- Exports metrics to XML or CSV (comma-separated-value) files for further processing with other tools.

SourceMonitor downloads
- H:\utilitairesSource monitor
Les activités du processus de V\&V (Vérification et Validation)

- planification des tests
- conception des tests
- exécution des tests
- bilan des tests

Les bonnes pratiques de tests

- définir une stratégie de test globale des produits.
- maîtriser l’Ingénierie des Exigences (IE).
- respecter les phases de test prioritaires et les critères d’entrée et de sortie de chaque phase.
- disposer d’indicateurs sur les tests et anomalies.
- avoir des exigences contractuelles fortes vis-à-vis des méthodes de conduite de projet des fournisseurs.
- maîtriser la gestion de configuration (avec subversion, git, mercurial, etc.)
- mettre en œuvre des outils industriels partagés par les différents acteurs pour la gestion des tests et des anomalies.
- contractualiser les interactions entre les équipes.

Liens sur d’autres outils d’analyse de code  See Also:


Lint

- http://en.wikipedia.org/wiki/Lint_%28software%29 In computer programming, lint was the name originally given to a particular program that flagged some suspicious and non-portable constructs (likely to be bugs) in C language source code. The term is now applied generically to tools that flag suspicious usage in software written in any computer language. The term lint-like behavior is sometimes applied to the process of flagging suspicious language usage. Lint-like tools generally perform static analysis of source code

Sonar

- http://linsolas.developpez.com/articles/java/qualite/sonar/?page=page_1#LI-A

Sonar is an open source software quality platform. Sonar uses various static code analysis tools such as Checkstyle, PMD, FindBugs, Clover to extract software metrics, which then can be used to improve software quality.

Sonar Reception  Received the Jolt Awards under testing tools category. Featured in continuous integration tools.[15] Used by the Andalusian Autonomous Government, eXo Platform, Apache Software Foundation, Marvelution
Yasca

- http://twitter.com/#!/yasca

Yasca is an open source program which looks for security vulnerabilities, code-quality, performance, and conformance to best practices in program source code. It leverages external open source programs, such as FindBugs, PMD, JLint, JavaScript Lint, PHPLint, Cppcheck, ClamAV, Pixy, and RATS to scan specific file types, and also contains many custom scanners developed for Yasca. It is a command-line tool that generates reports in HTML, CSV, XML, MySQL, SQLite, and other formats. It is listed as a tool at the well-known OWASP security project, and also in a government software security tools review at the Homeland Security web site.

Yasca introduction  Yasca is a source code analysis tool that I started writing in 2007. It could best be described as a “glorified grep script” plus an aggregator of other open-source tools. It isn’t rocket science, but then again, neither is rocket science.

Yasca can scan source code written in Java, C/C++, HTML, JavaScript, ASP, ColdFusion, PHP, COBOL, .NET, and other languages. Yasca can integrate easily with other tools, including:

- FindBugs
- PMD
- JLint
- JavaScript Lint
- PHPLint
- CppCheck
- ClamAV
- RATS
- Pixy

Yasca is designed to be very flexible and easy to extend. In fact, writing a new rule is as easy as coming up with a regular expression, the file extensions it applies to, and the name of your new rule. Place that information in a text file in the plugin directory, and run Yasca!

Moose

- http://en.wikipedia.org/wiki/Moose_%28analysis%29
- http://www.moosetechnology.org/

Moose is a free and open source platform for software and data analysis built in Pharo. Moose offers multiple services ranging from importing and parsing data, to modeling, to measuring, querying, mining, and to building interactive and visual analysis tools. Moose was born in a research context,[1] and it is currently supported by several research groups throughout the world. It is increasingly being adopted in industry.

https://www.cert.org/secure-coding/tools.html  This list of resources that might be useful for improving software security is provided for information and convenience only. As part of a Federally Funded Research and Development Center (FFRDC), the CERT Coordination Center cannot endorse any products, services, or organizations.

- http://frama-c.com/Frama-C is a suite of tools dedicated to the analysis of the source code of software written in C.
Moose

- http://en.wikipedia.org/wiki/Moose_%28analysis%29
- http://www.moosetechnology.org/

Moose is a free and open source platform for software and data analysis built in Pharo. Moose offers multiple services ranging from importing and parsing data, to modeling, to measuring, querying, mining, and to building interactive and visual analysis tools. Moose was born in a research context,[1] and it is currently supported by several research groups throughout the world. It is increasingly being adopted in industry.

https://www.cert.org/secure-coding/tools.html This list of resources that might be useful for improving software security is provided for information and convenience only. As part of a Federally Funded Research and Development Center (FFRDC), the CERT Coordination Center cannot endorse any products, services, or organizations.

- http://frama-c.com/Frama-C is a suite of tools dedicated to the analysis of the source code of software written in C.

daca  See Also:
http://qa.debian.org/daca/

Automated Code Analysis helps detect and fix bugs and other issues in source code. The DACA project aims to give users easy access to a wide set of tools to improve quality, while giving the tool’s developers a test bed, more visibility, and more feedback. Check the DACA (development) website and learn how to contribute and join the project.

cpp check  See Also:
http://cppcheck.sourceforge.net/

Ccppcheck is an static analysis tool for C/C++ code. Unlike C/C++ compilers and many other analysis tools it does not detect syntax errors in the code. Ccppcheck primarily detects the types of bugs that the compilers normally do not detect. The goal is to detect only real errors in the code (i.e. have zero false positives).

Documentation  You can read the manual or download some articles.

Software Quality in Open source projects

Qt Software Quality Engineering  See Also:
- Qt Software Quality Engineering

Links to other software quality sites

Glossaire des termes utilisés en tests de logiciels (CFTL/ISTQB)  Glossaire des termes utilisés en tests de logiciels
So here is my short list of the Psychological Elements of Software Architecture:

**Stupidity** Mental bandwidth is limited, so we’re all stupid at some point. The architecture has to be simple to understand. This is the number one rule: *simplicity beats functionality, every time*. If you can’t understand an architecture at 7am before coffee, it’s too complex.

**Selfishness** We act only out of self-interest, so the architecture must create space and opportunity for selfish acts that benefit the whole. Selfishness is often indirect and subtle. For example I’ll spend hours helping someone else understand something because that could be worth days to me later.

**Laziness** We make lots of assumptions, many of which are wrong. We are happiest when we can spend the least effort to get a result, so the architecture has to make this possible. Specifically, that means simplicity and leverage. Exploit the paths of least resistance and challenge assumptions.

**Jealousy** We’re jealous of others, which means we’ll overcome our stupidity and laziness to prove others wrong, and beat them in competition. The architecture thus has to create space for public competition based on fair rules that anyone can understand.
Reciprocity we’ll work hard to punish cheats and enforce fair rules. The architecture should be heavily rule based, telling people how to work together, but not what to work on.

Pride we’re intensely aware of our social status, and we’ll work hard to avoid looking stupid or incompetent in public. The architecture has to make sure every piece has someone’s name on it, so that person will wake up at 4am with nightmares about people laughing over the implementation.

Greed we’re ultimately economic animals (see selfishness), so the architecture has to give us economic incentive to invest in making it happen. Maybe it’s polishing our reputation as experts, maybe it’s literally making money from some skill or component. Think of architecture as a market place, not an engineering design.

Conformity we’re happiest to conform, out of fear and laziness, which can be good but is mostly a problem. The architecture should encourage diversity of thought and action. The only conformity that’s needed is adherence to rules on remixing and ownership.

Fear we’re unwilling to take risks, especially if it makes us look stupid. Fear of failure is a major reason people conform and follow the group in mass stupidity. The architecture should make silent experimentation easy and cheap, giving people opportunity for success without punishing failure.

And yes, I’m as stupid, selfish, fearful, greedy, and proud as anyone.

Issue tracking

See Also:

bugzilla Issue tracking See Also:
- www.bugzilla.org
- http://www.bugzilla.org/installation-list/
- http://www.bugzilla.org/status/changes.html

Contents
- bugzilla Issue tracking
  - Présentation
  - Qui utilise bugzilla ?
  - Evolution
Présentation  Bugzilla est un logiciel libre de système de suivi de problèmes avec interface web, développé et utilisé par l’organisation Mozilla.

Il permet le suivi de bugs ou de RFE sous la forme de « tickets ».

Logiciel de type serveur, architecture trois tiers, il est écrit en langage Perl. Il est disponible sous UNIX (Linux, BSD, etc.) et est distribué sous tri-licence MPL/LGPL/GPL.

Il est utilisé par de nombreuses organisations pour suivre le développement de nombreux logiciels, sur Internet ou dans des réseaux privés.

Les plus connus sont la Mozilla Foundation, GNOME, KDE, Red Hat et Mandriva.

Qui utilise bugzilla ?  See Also:
  • http://www.bugzilla.org/installation-list/
  • Mercurial issue_tracker

Evolution  See Also:
http://www.bugzilla.org/status/changes.html

JIRA Issue tracking  See Also:
http://en.wikipedia.org/wiki/JIRA

JIRA (/dr/ jeer-) is a proprietary issue tracking product, developed by Atlassian, commonly used for bug tracking, issue tracking, and project management.

The product name, JIRA, is not an acronym but rather a truncation of “Gojira”, the Japanese name for Godzilla. It has been developed since 2004.

roundup Issue tracking  See Also:
  • http://roundup.sourceforge.net/
  • http://www.framasoft.net/article4319.html

Roundup is a simple-to-use and -install issue-tracking system with command-line, web and e-mail interfaces.

It is based on the winning design from Ka-Ping Yee in the Software Carpentry “Track” design competition.
List of projects using roundup

- cpython

trac Issue tracking    See Also:
- http://trac.edgewall.org/

Figure 2.48: Logo Trac

Trac is an enhanced wiki and issue tracking system for software development projects. Trac uses a minimalistic approach to web-based software project management. Our mission is to help developers write great software while staying out of the way. Trac should impose as little as possible on a team’s established development process and policies.

It provides an interface to Subversion (or other version control systems), an integrated Wiki and convenient reporting facilities.

Trac allows wiki markup in issue descriptions and commit messages, creating links and seamless references between bugs, tasks, changesets, files and wiki pages. A timeline shows all current and past project events in order, making the acquisition of an overview of the project and tracking progress very easy. The roadmap shows the road ahead, listing the upcoming milestones.

trac
de  Luke Plant <L.Plant.98@cantab.net>
à  Matt Mackall <mpm@selenic.com>
c  mercurial-devel@selenic.com
date  15 mars 2011 16:59
objet  Re: Roundup -> Trac
liste de diffusion  <mercurial-devel.selenic.com> Filtrer les messages de cette liste de diffusion

Hi all,

I’m a newcomer to this list, but I thought I’d weigh in to say that I think Trac is a pretty good option.

We’ve used it for Django since the beginning, and I haven’t come across a serious contender to replace it.

Painless upgrades    Upgrade procedure is here: http://trac.edgewall.org/wiki/TracUpgrade

It does require a little bit more than ‘easy_install -U’, but I don’t think that is avoidable due to the need for database upgrades.

We upgraded Django’s trac to 0.12 recently and the only issue I’m aware of is that the ‘updates’ list has stopped sending out changesets, but I don’t think that is actually Trac related. I happen to use Trac for a personal wiki, and so just tried the upgrade from 0.11 to 0.12 and it worked fine.

Reliable search    Search has always worked pretty well for me.
Seamless mail gateway, Bulk operations on issues Not out of the box, but one of the great things about Trac is that it does have a large, active community of plugin writers:

http://trac-hacks.org/

The code is clean Python and it is pretty easy to write plugins yourself. (I’ve written a couple of simple ones).

A quick search finds this for e-mail:

https://subtrac.sara.nl/oss/email2trac/wiki/Email2tracParse

And this was already mentioned for batch modifications:

http://trac-hacks.org/wiki/BatchModifyPlugin

We use this on Django’s Trac.

Doesn’t send an endless stream of tracebacks via email I’ve never had issues with Trac crashing.

You also mentioned query building - I find the search query builder with Trac pretty powerful and friendly.

Example:

http://goo.gl/RigdX

(For me that page also shows the ‘Batch Modify’ interface, which does everything I would expect)

Trac is also very customisable in terms of what kind of things you want against each bug - milestone, version, status, triage stage, resolution, bug type, priority, severity, component - in general if any of these are not used they disappear from the interface, and you can add other custom flags, as seen above (like “Needs documentation”).

Luke

---

Luke Plant | http://lukeplant.me.uk/

Logging management

Logging

logging

from Orin Eman <orin.eman@gmail.com>
to michael.plante@gmail.com
cc libusb-devel@lists.sourceforge.net
date Tue, Jun 8, 2010 at 7:17 PM
subject Re: [Libusb-devel] Redirect libusb log messages

I’ve had enough trouble in the past with log files getting messed up in multi-threaded applications... so here are some thoughts on the matter:

- If you use FILE, you need an fflush() after each write to the stream or when the app crashes, you end up with the most interesting log entries missing.

In general, I like to be able to read a log file while my application is running. I’ve had little luck with this using stdio on Windows, whatever I did to set up the stream as unbuffered and had to resort to opening/closing the file for each log entry! I’d recommend using native file descriptors/handles that can be setup with sharing modes such that other processes can read the log file while the application is still running.

Why would you pass the format and args to a callback? I’d do the formatting first and pass nothing more than a const char * which is defined to be valid for the duration of the callback only.
libusb 1.0 apps can be multi-threaded. You might as well assume that multiple threads call into libusb and create log entries simultaneously. libusb isn’t going to know about order and nor should it care in this case. It should care that log messages don’t get mixed up. The safest way of doing this is with some form of sprintf followed by an unbuffered native file write.

Timestamps. Log entries should be timestamped IMO. I wouldn’t worry about the time spent going into OS here as a file operation takes much longer than getting the system time. If that were a worry, I’d use a sequence number instead.

Thread IDs. In multi-threaded applications, it often helps for log entries to include the thread ID of the thread that created them and I’d recommend that log messages include the thread ID.

In short, I’d recommend that parameters to a callback, if used, is a const char * only, defined to be valid for the duration of the callback;

I’d add both a timestamp and the current thread ID to log messages;

I wouldn’t pass FILE, rather a native file descriptor or file handle and let the backend do an unbuffered write of log entries to the file.

Orin.

g2log    See Also:
https://bitbucket.org/KjellKod/g2log/src

UML

UML (Unified Modeling Language)

See Also:


UML (en anglais Unified Modeling Language ou langage de modélisation unifié) est un langage de modélisation graphique à base de pictogrammes.

Il est apparu dans le monde du génie logiciel, dans le cadre de la conception orientée objet. Couramment utilisé dans les projets logiciels, il peut être appliqué à toutes sortes de systèmes ne se limitant pas au domaine informatique.

UML est l’accomplissement de la fusion de précédents langages de modélisation objet : Booch, OMT, OOSE. Principalement issu des travaux de Grady Booch, James Rumbaugh et Ivar Jacobson, UML est à présent un standard défini par l’Object Management Group (OMG).

La dernière version diffusée par l’OMG est UML 2.3 depuis mai 2010.

UML tools

2.26. Software engineering
plant UML  See Also:

- http://plantuml.sourceforge.net/faq.html
- https://pypi.python.org/pypi/sphinxcontrib-plantuml

PlantUML is used to draw UML diagram, using a simple and human readable text description.

Be careful, because it does not prevent you from drawing inconsistent diagram (like having two classes inheriting from each other, for example).

So it’s more a drawing tool than a modeling tool.

**Glossary**

**Software engineering Glossary**

**Design Patterns**  In software engineering, a design pattern is a general reusable solution to a commonly occurring problem within a given context in software design. A design pattern is not a finished design that can be transformed directly into code. It is a description or template for how to solve a problem that can be used in many different situations.

Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved. Many patterns imply object-orientation or more generally mutable state, and so may not be as applicable in functional programming languages, in which data is immutable or treated as such.

See Also:


**DRY, Don’t Repeat Yourself**  (a.k.a. DRY), which requires you to rigorously remove duplication in your code base

**exigence**  une condition ou une capacité requise par un utilisateur pour résoudre un problème ou atteindre un objectif qui doit être tenu ou possédé par un système ou composant pour satisfaire à un contrat, standard, spécification ou autre document imposé formellement [d’après IEEE 610].

See Also:


**fluent interface**  In software engineering, a fluent interface (as first coined by Eric Evans and Martin Fowler) is an implementation of an object oriented API that aims to provide for more readable code.

A fluent interface is normally implemented by using method chaining to relay the instruction context of a subsequent call (but a fluent interface entails more than just method chaining [1]). Generally, the context is

- defined through the return value of a called method
- self referential, where the new context is equivalent to the last context
- terminated through the return of a void context.

This style is marginally beneficial in readability due to its ability to provide a more fluid feel to the code[citation needed]. However, it can be detrimental to debugging, as a fluent chain constitutes a single statement for which debuggers may not allow setting up intermediate breakpoints, for instance.

See Also:

http://en.wikipedia.org/wiki/Fluent_interface
Interface Segregation Principle

See Also:
http://www.objectmentor.com/resources/articles/isp.pdf

ISO 9241  ISO 9241 is a multi-part standard from the International Organization for Standardization (ISO), covering aspects of people working with computers. It was originally titled Ergonomic requirements for office work with visual display terminals (VDTs). The standards are being retitled to the more generic Ergonomics of Human System Interaction. As part of this change, ISO is renumbering the standard so that it can cover more topics. The first part to be renumbered was part 10, now renumbered to part 110.

See Also:

ISO 9241-110  (formerly ISO9241-10, deprecated) Dialogue principles (2006) This part deals with general ergonomic principles which apply to the design of dialogues between humans and information systems:

• suitability for the task,
• suitability for learning,
• suitability for individualisation,
• conformity with user expectations,
• self descriptiveness,
• controllability, and
• error tolerance.

See Also:

Keep It Simple Stupid  (a.k.a. KISS), a funny way of saying that the simplest solution is more than sufficient.

See Also:
http://fr.wikipedia.org/wiki/Keep_it_Simple,_Stupid

Law of Demeter  The Law of Demeter (LoD) or Principle of Least Knowledge is a design guideline for developing software, particularly object-oriented programs. In its general form, the LoD is a specific case of loose coupling. The guideline was invented at Northeastern University towards the end of 1987, and can be succinctly summarized in one of the following ways:

• Each unit should have only limited knowledge about other units: only units “closely” related to the current unit.
• Each unit should only talk to its friends; don’t talk to strangers.
• Only talk to your immediate friends.

The fundamental notion is that a given object should assume as little as possible about the structure or properties of anything else (including its subcomponents).

See Also:

Liskov Substitution Principle  It should be possible to treat a derived object as if it were a base class object. This rule is one of the S.O.L.I.D. principles

See Also:
http://en.wikipedia.org/wiki/Liskov_substitution_principle
modularity  the resulting software comprises well defined, independent components. That leads to better maintainability. The components could be then implemented and tested in isolation before being integrated to form a desired software system. This allows division of work in a software development project.

See Also:

• http://en.wikipedia.org/wiki/Modularity
• http://en.wikipedia.org/wiki/Modular_programming

maintainability  The software can be restored to a specified condition within a specified period of time. For example, antivirus software may include the ability to periodically receive virus definition updates in order to maintain the software’s effectiveness

Polymorphism  Subtype polymorphism, almost universally called just polymorphism in the context of object-oriented programming, is the ability to create a variable, a function, or an object that has more than one form. The word derives from the Greek πολυμορφισμός meaning “having multiple forms”. In principle, polymorphism can however arise in other computing contexts and it shares important similarities to the concept of degeneracy in biology.

See Also:

http://en.wikipedia.org/wiki/Polymorphism_in_object-oriented_programming

portability  Portability is one of the key concepts of high-level programming. Portability is the software codebase feature to be able to reuse the existing code instead of creating new code when moving software from one environment to another. The prerequisite for portability is the generalized abstraction between the application logic and system interfaces. When one is targeting several platforms with the same application, portability is the key issue for development cost reduction.

See Also:

• http://en.wikipedia.org/wiki/Software_portability

principle of least surprise, Principle of Least Surprise  The Principle of Least Surprise (or Astonishment), which means that you should choose a solution that does include any things people might not understand, or put on the wrong track.

See Also:


Single Responsibility Principle  This principle was described in the work of Tom DeMarco1 and Meilir Page-Jones. They called it cohesion. As we’ll see in Chapter 21, we have a more specific definition of cohesion at the package level.

However, at the class level the definition is similar.

See Also:

http://www.objectmentor.com/resources/articles/srp.pdf

S.O.L.I.D.  S.O.L.I.D. is a collection of best-practice object-oriented design principles that you can apply to your design to accomplish various desirable goals like loose-coupling, higher maintainability, intuitive location of interesting code, etc. S.O.L.I.D. is an acronym for the following principles

See Also:


Software Requirements Specification, SRS  a requirements specification for a software system – is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional (or supplementary) requirements. Non-functional requirements are requirements which impose constraints on
the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

See Also:


**specification** specification: A document that specifies, ideally in a complete, precise and verifiable manner, the requirements, design, behavior, or other characteristics of a component or system, and, often, the procedures for determining whether these provisions have been satisfied. [After IEEE 610]

**Spécification** un document qui spécifie, idéalement de façon complète, précise et vérifiable, les exigences, conceptions, comportements et autres caractéristiques d’un composant ou système, et souvent, les procédures pour déterminer si ces stipulations ont été satisfaites. [d’après IEEE 610]

See Also:


La phase de spécification doit être précédée par une étude préalable, qui décrit l’existant et les attentes et exigences générales exprimées par les utilisateurs pour le domaine à informatiser. Un exemple d’attente à prendre en compte à ce stade est la langue du logiciel, qui doit être adaptée à l’utilisateur. Les spécifications reprendront ces exigences pour les décrire plus en détail.

**YAGNI, You Ain’t Gonna Need It** (a.k.a. YAGNI), which tells you to create a solution for the current problem rather than the ones you think will happen later on (since when can you predict the future?)

### 2.27 Software frameworks

#### 2.27.1 Software frameworks

**Java Virtual Machine**

See Also:


A Java virtual machine (JVM) is a virtual machine capable of executing Java bytecode. It is the code execution component of the Java software platform. Sun Microsystems stated that there are over 4.5 billion JVM-enabled devices.

**Virtual machines**

**Java Virtual Machines** See Also:


**Dalvik Java Virtual Machine** See Also:

- [Android Operating system](http://fr.wikipedia.org/wiki/Android_Operating_system)
Figure 2.49: System architecture
Dalvik est la machine virtuelle utilisée dans les appareils mobiles Android.

Elle permet d’exécuter les applications qui peuvent être écrites en java et converties en un exécutable compact Dalvik (.dex), format adapté aux systèmes limités en termes de mémoire et de vitesse du processeur.

Dalvik a été écrit par Dan Bornstein qui l’a nommé d’après le village de pêcheurs de Dalvík en Islande, où vivaient certains de ses ancêtres

**Architecture**  Unlike Java VMs, which are stack machines, the Dalvik VM is a register-based architecture.

A tool called dx is used to convert some (but not all) Java .class files into the .dex format.

Multiple classes are included in a single .dex file. Duplicate strings and other constants used in multiple class files are included only once in the .dex output to conserve space.

Java bytecode is also converted into an alternative instruction set used by the Dalvik VM.

An uncompressed .dex file is typically a few percent smaller in size than a compressed .jar (Java Archive) derived from the same .class files.

The Dalvik executables may be modified again when installed onto a mobile device. In order to gain further optimizations, byte order may be swapped in certain data, simple data structures and function libraries may be linked inline, and empty class objects may be short-circuited, for example.

As of Android 2.2, Dalvik has a just-in-time compiler.

Being optimized for low memory requirements, Dalvik has some specific characteristics that differentiate it from other standard VMs:[5]

- The VM was slimmed down to use less space
- The constant pool has been modified to use only 32-bit indices to simplify the interpreter
- Standard Java bytecode executes 8-bit stack instructions. Local variables must be copied to or from the operand stack by separate instructions. Dalvik instead uses its own 16-bit instruction set that works directly on local variables. The local variable is commonly picked by a 4-bit ‘virtual register’ field. This lowers Dalvik’s instruction count and raises its interpreter speed.

Moreover, according to Google, Dalvik has been designed so that a device can run multiple instances of the VM efficiently

**Hotspot Java Virtual Machine**  See Also:

http://fr.wikipedia.org/wiki/Machine_virtuelle_Java#HotSpot_la-machine_virtuelle_d.27Oracle

C’est la machine la plus utilisée. Elle a été créée et réalisée par Sun, elle est aujourd’hui propriété d’Oracle, depuis que ce dernier a racheté Sun.

Elle est gratuite, propriétaire jusqu’à la version 6 (stable) et libre à partir de la version 7 (non encore officielle).

Le 11 novembre 2006, Sun Microsystems a publié les sources de sa machine virtuelle HotSpot et de son compilateur javac sous licence GNU GPL.

La première version ne se comportait que comme un interpréteur.

Cette approche était pénalisante, car l’interprète passe plus de temps à interpréter qu’à exécuter.

Puis est apparue la compilation à la volée, qui traduit le bytecode en langage machine, et exécute ce langage machine.

Ensuite la machine virtuelle est devenue capable de détecter les portions les plus fréquemment utilisées pour concentrer les optimisations sur elles. Elle a été dotée d’un profilleur, et d’optimisations standards comme la mise à plat des boucles. Toutes ces opérations peuvent être faites en plusieurs passes, soit pour les améliorer progressivement, soit pour les annuler si elles se révèlent obsolètes.
The .NET Framework (pronounced dot net) is a software framework that runs primarily on Microsoft Windows.

It includes a large library and supports several programming languages which allows language interoperability (each language can use code written in other languages).

Programs written for the .NET Framework execute in a software environment (as contrasted to hardware environment), known as the Common Language Runtime (CLR), an application virtual machine that provides important services such as security, memory management, and exception handling.

The class library and the CLR together constitute the .NET Framework.

**Dotnet versions**
DotNet 4.5  See Also:


Contents

• DotNet 4.5
  – Introduction
  – What’s new in 4.5 dotnet framework
    • new asynchronous features
  – Differences between .NET 4.0 to .NET 4.5
  – Windows OS supported
    • Systèmes d’exploitation pris en charge
    • Architectures prises en charge
    • Windows XP no supported

Introduction

Version Number  4.5

Visual Studio  Visual Studio 2011

What’s new in 4.5 dotnet framework  See Also:

• http://msdn.microsoft.com/library/ms171868%28VS.110%29.aspx
• Asynchronous File Operations

new asynchronous features  In the .NET Framework 4.5 Developer Preview, new asynchronous features were added to the C# and Visual Basic languages. These features add a task-based model for performing asynchronous operations. To use this new model, use the asynchronous methods in the I/O classes.

Differences between .NET 4.0 to .NET 4.5  See Also:

• http://go-mono.com/status/

Windows OS supported

Systèmes d’exploitation pris en charge

• Windows 7 (x86 et x64)
• Windows 8 Consumer Preview (x86 et x64)
• Windows Server 2008 R2 (x64)
• Windows Server 8 Beta (x64)

Architectures prises en charge

• 32 bits (x86)
• 64 bits (x64)
Windows XP no supported  See Also:

- http://community.sharpdevelop.net/blogs/christophwille/archive/2012/03/03/sharpdevelop-platform-roadmap.aspx

.NET 4.5 will likely not support XP (maybe not even Vista)

DotNet 4.0  See Also:


- **Version Number**  4.0
- **Release Date**  2010-04-12
- **Visual Studio**  Visual Studio 2010

### 2.28 Test software

#### 2.28.1 Testing tool

**Fuzzing Test**

See Also:

https://secure.wikimedia.org/wikipedia/en/wiki/Fuzz_testing

Fuzz testing or fuzzing is a software testing technique that provides invalid, unexpected, or random data to the inputs of a program.

If the program fails (for example, by crashing or failing built-in code assertions), the defects can be noted

**Fusil**

See Also:

- https://bitbucket.org/haypo/fusil/src/
- https://bitbucket.org/haypo/fusil/wiki/Home

Fusil is the fuzzer is a Python library used to write fuzzing programs.

It helps to start process with a prepared environment, start network client or server, and create mangled files.

Fusil has many probes to detect program crash: watch process exit code, process stdout/syslog for text patterns, session duration, cpu usage, etc.

Fusil is based on a multi-agent system architecture. It computes a session score used to guess fuzzing parameters like number of injected errors to input files.

Available fuzzing projects: ClamAV, Firefox (contains an HTTP server), gettext, gstreamer, identify, libc_env, libc_printf, libexif, linux_syscall, mplayer, php, poppler, vim, xterm.

Website: http://bitbucket.org/haypo/fusil/wiki/Home
Usage  Fusil is a library and a set of fuzzers called “fusil-...”. To run a fuzzer, call it by its name. Example:

$ fusil-gettext
Fusil version 0.9.1 -- GNU GPL v2
http://bitbucket.org/haypo/fusil/wiki/Home
(...)
[0][session 13] Start session
[0][session 13] ------------------------------------------------------------
[0][session 13] PID: 16989
[0][session 13] Signal: SIGSEGV
[0][session 13] Invalid read from 0x0c1086e0
[0][session 13] - instruction: CMP EDX, [EAX]
[0][session 13] - mapping: 0x0c1086e0 is not mapped in memory
[0][session 13] - register eax=0x0c1086e0
[0][session 13] - register edx=0x00000019
[0][session 13] ------------------------------------------------------------
[0][session 13] End of session: score=100.0%, duration=3.806 second
(...)
Success 1/1!
Project done: 13 sessions in 5.4 seconds (414.5 ms per session), total 5.9 seconds, aggressivity: 19.0%
Total: 1 success
Keep non-empty directory: /home/haypo/prog/SVN/fusil/trunk/run-3

Features  Why using Fusil instead your own hand made C script?

• Fusil limits child process environment: limit memory, use timeout, make sure that process is killed on session end
• Fusil waits until system load is load before starting a fuzzing session
• Fusil creates a session directory used as the process current working directory and Fusil only creates files in this directory (and not in /tmp)
• Fusil stores all actions in fusil.log but also session.log for all actions related of a session
• Fusil has multiple available probes to compute session score: guess if a sessions is a succes or not
• Fusil redirects process output to a file and searches bug text patterns in the stdout/stderr (Fusil contains many text patterns to detect crashes and problems)

Python test tool

Picka

• http://pypi.python.org/pypi/picka
Picka generates randomized data either from lists of known good data (or bad) stored in a sqlite database, or by generating randomized realistic data, using string formatting (behind the scenes). Picka has a function for any field you would need filled in. With selenium, something like would populate the “field-name-here” box for you, 100 times with random names.

test tools

watchr tool

See Also:

2.28. Test software 1039
Summary
Agile development tool that monitors a directory tree, and triggers a user defined action whenever an observed file is modified. Its most typical use is continuous testing, and as such it is a more flexible alternative to autotest. Features

watchr is:
- Simple to use
- Highly flexible
- Evented (Listens for filesystem events with native c libs)
- Portable (Linux, *BSD, OSX, Solaris, Windows)
- Fast (Immediately reacts to file changes)

Most importantly it allows running tests in an environment that is agnostic to:
- Web frameworks (rails, merb, sinatra, camping, invisible, ...)
- Test frameworks (test/unit, minitest, rspec, test/spec, expectations, ...)
- Ruby interpreters (ruby1.8, ruby1.9, MRI, JRuby, Rubinius, ...)
- Package frameworks (rubygems, rip, ...)

2.29 Version Control Systems

2.29.1 Version Control Systems

git

See Also:
- https://secure.wikimedia.org/wikipedia/en/wiki/Git_%28software%29
- http://whygitisbetterthanx.com/

Introduction

Linus Torvalds has quipped about the name git, which is British English slang for a stupid or unpleasant person:[4]

“I’m an egotistical bastard, and I name all my projects after myself. First Linux, now git.

git installation See Also:
- http://developer.qt.nokia.com/wiki/Git_Installation

Git on Mac OS X On Mac OS X, you can get Git using MacPorts [macports.org].

After installing MacPorts, do:
Development tools, Release 2012.06.18

```
sudo port selfupdate
sudo port sync
sudo port install git-core
add /opt/local/bin to your PATH
```

Note that this also adds another ssh into your path. You can safely remove :file:/opt/local/bin/ssh*, :file:/opt/local/bin/scp, :file:/opt/local/bin/sftp (especially if you want to use Leopard’s built-in ssh-agent).

**Git on Windows**  **See Also:**

http://code.google.com/p/msysgit/

For Windows there exists a convenient installer as part of the MSysGit project:

---

**Note:** When the installer presents you with the option of how to put Git into your PATH choose the third option of putting Git and all its tools into your path!

---

**Git on Linux**  All major Linux distributions offer up-to-date packages of Git.

Use your distribution specific package manager to install Git.

Note that on Debian based distributions the Git package is actually called `git-core`.

**Building Git from Source**  Git is easy to build from source. Just :

- download the latest release from [http://git.or.cz/][git.or.cz],
- unpack the sources,
- run configure
- followed by make followed by make install. This will install it locally into your home directory, the binaries end up in `$HOME/bin`.

**Dependency list (Kubuntu 8.4):**

- `libzlcore-dev`
- `libssl-dev`
- `libcurl4-openssl-dev`
- `libexpat1-dev`
- `tclx8.3`
- `asciidoce`


Steps to create patch :

1. Cloning the Gcompris repository:

   ```
git clone git://git.gnome.org/gcompris

```

2.29. **Version Control Systems**
2. To check the current branch:
   ```
   git branch
   *master
   ```

3. Create a new branch for the patch:
   ```
   git checkout -n srishti
   ```

4) Again recheck the branch:
   ```
   git branch
   *master
   srishti
   ```

5) Performing alterations

6) To check the alterations perform:
   ```
   git diff
   ```

7) To commit all the changes in the branch:
   ```
   git commit -a
   ```

8) And finally perform format patch command:
   ```
   git format-patch master --stdout > new.patch
   ```

**git flow** See Also:

- https://github.com/nvie/gitflow
- http://nvie.com/posts/a-successful-git-branching-model/

https://github.com/nvie/gitflow Git extensions to provide high-level repository operations for Vincent Driessen’s branching model

**Getting started** See Also:

- http://sandofsky.com/blog/git-workflow.html
- http://codesherpas.com/screencasts/on_the_path_gitflow.mov
- http://vimeo.com/16018419

For the best introduction to get started with git flow, please read Jeff Kreeftmeijer’s blog post:

Or have a look at one of these screen casts:

- A short introduction to git-flow (by Mark Derricutt)
- On the path with git-flow (by Dave Bock)
Benefits of the new model

Hotfixes, typically bug fixes, can be made simultaneously on stable and develop branch. The git-flow package takes care of the merging to both.

Because that’s so easy now, a stable version can be provided and supported for a longer time. Features can be published while under development. My idea is that feature branches should basically work, but the bar will be lower. People can have a look at them, or start their own and make me integrate them.

Uses of Feature Branch

For example, in the new feature branch, a couple of boring things are happening. Support for frame stack will reduce the diff, as will some work to match CPython’s choices for exception line numbers. Completing will take a while, but should not block a release. So this is best done in the feature branch, esp. as nothing is going to really depend on it.
General Picture  As you can see from this diagram, I am working mostly on documentation things. The new and improved README on develop, which is closer to a User Manual in PDF form, and other organization things, may get a release before the PyCon DE next week. The README also describes this process.

Hope is that with this approach, I will improve transparency (you can see earlier what i am working on, because there is now a place where things may break (develop) or may not yet be integrated or completed fully (feature branches) and yet be public.

The overhead appears to minimal thanks to git-flow. Developing hotfixes is actually easier, when done on the stable branch, because problems cannot originate from the current development work that may or may not be all that perfect yet.

Yours, Kay Hayen

git tips

Greg Kroah-Hartman  Best +Git tip I learned in a long time:

:::

    gitk ORIG_HEAD.. 

after pulling a branch from someone.

You can also do:

    git shortlog ORIG_HEAD..

or:

    git diff --stat --summary ORIG_HEAD..

as well.

And if things went wrong:

    git reset --hard ORIG_HEAD

to revert the pull (although I thought there was some other way to do that, but can’t remember it at the moment.)

Many thanks to +David Miller for that one, it’s already proving to be indispensable.

Tree of your branch merges. very useful if you want to follow the features you add

    git log --graph --oneline --all

Git commands
Git tutorials  See Also:

- http://nvie.com/git-model
- http://github.com/nvie/gitflow
- http://progit.org/
- http://help.github.com/
- http://www-cs-students.stanford.edu/~blynn/gitmagic/
- http://git-scm.com/documentation
- http://wiki.winehq.org/GitWine
- http://whygitisbetterthanx.com/

Contents

- Git tutorials
  - Git for pitivi

Git for pitivi  See Also:

- http://wiki.pitivi.org/wiki/Git

This page is not meant to be a general tutorial for Git; for that, see the GNOME Git page, the official Git tutorial/documentation page and git ready.

In this page, we will cover some more advanced usage and the specifics of how we use Git in the Pitivi project.

This is aimed at people coming from Bazaar or Subversion.

git tools

git gerrit  See Also:

- http://code.google.com/p/gerrit/
- http://wiki.qt-project.org/Gerrit_Introduction
- Gerrit Code review

Main features  Gerrit is a web-based tool that is used for code review.

Its main features are the side-by-side difference viewing and inline commenting which makes code reviews quick and simple task.

It is used together with Git version control system.

Gerrit allows authorized contributors to submit changes to Git repository, after reviews are done. Contributors can get their code reviewed with a little effort, and changes get their changes quickly through the system.
References

- http://gerrit.googlecode.com/svn/documentation/2.1.6/user-upload.html#push_create

git python tools  See Also:

*python VCS modules*

git use cases

git mozilla use cases  See Also:


git qt use case  See Also:

- http://developer.qt.nokia.com/wiki/Git_Installation
- http://wiki qt-project.org/Gerrit_Introduction
- http://wiki qt-project.org/Code_Reviews
- http://developer.qt.nokia.com/wiki/Git_Installation

Git installation  See Also:

http://developer.qt.nokia.com/wiki/Git_Installation

Git introduction  See Also:

http://wiki qt-project.org/Git_Introduction

git web hosting

bitbucket git hosting  See Also:

*bitbucket*

github web hosting  See Also:

- https://github.com/
- http://help.github.com/create-a-repo/
- https://github.com/repositories/new
- http://en.wikipedia.org/wiki/GitHub

Figure 2.52: Logo Github
GitHub is a web-based hosting service for software development projects that use the Git revision control system. GitHub offers both commercial plans and free accounts for open source projects.

According to the Git User’s Survey in 2009, GitHub is the most popular Git hosting site.

**Description**

The site provides social networking functionality such as feeds, followers and the network graph to display how developers work on their versions of a repository.

GitHub also operates a pastebin-style site called Gist,[4] wikis for individual repositories, and web pages that can be edited through a Git repository.

As of January 2010, GitHub is operated under the name GitHub, Inc.

The software that runs GitHub was written using Ruby on Rails and Erlang by GitHub, Inc. (previously known as Logical Awesome) developers Chris Wanstrath, PJ Hyett, and Tom Preston-Werner.

**Fonctionnalités**

GitHub propose l’hébergement de projets sous Git. Le site fournit également des fonctionnalités de type réseaux sociaux : flux, suivi de personnes ou de projets, graphes de réseau pour les dépôts, etc...

Il dispose aussi d’un pastebin nommé Gist, et propose un wiki et une page web pour chaque dépôt.

**Example on github**

```bash
# See Also:
https://github.com/pvergain/devtools
```

**Global setup**

**Set up git**

```bash
git config --global user.name "Patrick Vergain"
git config --global user.email pvergain@gmail.com
```

**Next steps**

```bash
mkdir devtools
cd devtools
git init
touch README
git add README
```
Development tools, Release 2012.06.18

```bash
git commit --message 'first commit'
git remote add origin git@github.com:pvergain/devtools.git
git push -u origin master
```

Existing Git Repo?

```bash
cd existing_git_repo
git remote add origin git@github.com:pvergain/devtools.git
git push -u origin master
```

pvdevtools on github   See Also:

https://github.com/pvergain/pvdevtools

Contents

- pvdevtools on github
  - Setup on GNU/linux ubuntu
  - Create A Repo
- Global setup

Setup on GNU/linux ubuntu   See Also:

http://help.github.com/linux-set-up-git/

```bash
2016 sudo aptitude install git-core
2017 sudo aptitude install git-gui
2018 sudo aptitude install git-doc
2019 cd ~/.ssh
2020 ls
2021 mkdir key_backup
2022 cp id_rsa* key_backup
2023 rm id_rsa*
2024 ssh-keygen -t rsa -C "pvergain@gmail.com"
2025 ssh -T git@github.com
2026 cd..
2027 ssh-add ~/.ssh/id_rsa
2028 ssh -T git@github.com
2029 git config --global github.user pvergain
2030 git config --global github.token 1da1d809f5834bb85ff1efe497ffbf99
```

Create A Repo   See Also:

- http://help.github.com/create-a-repo/
- https://github.com/repositories/new

Global setup
Set up git

```
git config --global user.name "Patrick Vergain"
```

```
git config --global user.email pvergain@gmail.com
```

Next steps

```
cd pvdevtools
```

```
git init
```

```
git add .
```

```
git commit -m 'first commit'
```

```
git remote add origin git@github.com:pvergain/pvdevtools.git
```

```
git push -u origin master
```

```
patrick@vercors:/home/pvergain$ git push -u origin master
Counting objects: 9016, done.
Compressing objects: 100% (7429/7429), done.
Writing objects: 100% (9016/9016), 23.42 MiB | 27 KiB/s, done.
Total 9016 (delta 1756), reused 0 (delta 0)
To git@github.com:pvergain/pvdevtools.git
  * [new branch] master -> master
Branch master set up to track remote branch master from origin.
```

Projects on github

Projects on github

zeromq on github See Also:

https://github.com/

See Also:

- [http://www.zeromq.org/docs:contributing](http://www.zeromq.org/docs:contributing)

The ØMQ Community uses the git version control system, and the GitHub cloud for its repositories.

The ØMQ Community consists of a number of projects, each with its own public git repository, or “git”.

Each git has an owner or owners (with admin rights), contributors (with read/write rights), and users (with read access).

Each project organizes its git as it wishes. A project is “official” when it is held under the zeromq organisation at GitHub, otherwise it is “unofficial”.

To make your project “official”, discuss on the mailing list. The zeromq organisation owners group have the privileges to create new projects.

We use github’s pull request model for changes. To make a change you fork the git in question, make and test your changes, and then make a pull request.

The benefit of a pull request is that it is trackable, like an issue.

Pull requests only work from a fork back to the original project.
Windows client for github  See Also:
  • http://windows.github.com/
  • http://github-windows.s3.amazonaws.com/setup.exe

Git API

github API  See Also:
  • http://develop.github.com/

github API v3  See Also:
  • http://developer.github.com/v3/
This describes the resources that make up the official GitHub API v3.
If you have any problems or requests please contact support.
Note: This API is in a beta state. Breaking changes may occur.
  • Schema
  • Client Errors
  • HTTP Verbs
  • Authentication
  • Pagination
  • Rate Limiting
  • Cross Origin Resource Sharing
  • JSON-P Callbacks

git on windows

Setting the proxy  See Also:
  • http://www.kernel.org/pub/software/scm/git/docs/git-config.html
  • http://help.github.com/win-set-up-git/

git config --global http.proxy gateway:3128

Git extensions  See Also:
http://sourceforge.net/projects/gitextensions/
Git Extensions is a toolkit to make working with Git under Windows more intuitive. The shell extension will integrate in Windows Explorer and presents a nice context menu on files.

Source code  Source code can be found here: http://github.com/spdr870/

git versions
git 1.7  See Also:
https://raw.github.com/gitster/git/master/Documentation/RelNotes/1.7.0.txt

git 1.7.9  See Also:
https://raw.github.com/gitster/git/master/Documentation/RelNotes/1.7.9.txt

Updates since v1.7.8

- gitk updates accumulated since early 2011.
- git-gui updated to 0.16.0.
- git-p4 (in contrib/) updates.
- Git uses gettext to translate its most common interface messages into the user’s language if translations are available and the locale is appropriately set. Distributors can drop new PO files in po/ to add new translations.
- The code to handle username/password for HTTP transactions used in “git push” & “git fetch” learned to talk “credential API” to external programs to cache or store them, to allow integration with platform native keychain mechanisms.
- The input prompts in the terminal use our own getpass() replacement when possible. HTTP transactions used to ask for the username without echoing back what was typed, but with this change you will see it as you type.
- The internals of “revert/cherry-pick” have been tweaked to prepare building more generic “sequencer” on top of the implementation that drives them.
- “git rev-parse FETCH_HEAD” after “git fetch” without specifying what to fetch from the command line will now show the commit that would be merged if the command were “git pull”.
- “git add” learned to stream large files directly into a packfile instead of writing them into individual loose object files.
- “git checkout -B <current branch> <elsewhere>” is a more intuitive way to spell “git reset –keep <elsewhere>”.
- “git checkout” and “git merge” learned “–no-overwrite-ignore” option to tell Git that untracked and ignored files are not expendable.
- “git commit –amend” learned “–no-edit” option to say that the user is amending the tree being recorded, without updating the commit log message.
- “git commit” and “git reset” re-learned the optimization to prime the cache-tree information in the index, which makes it faster to write a tree object out after the index entries are updated.
- “git commit” detects and rejects an attempt to stuff NUL byte in the commit log message.
- “git commit” learned “–S” to GPG-sign the commit; this can be shown with the “–show-signature” option to “git log”.
- fsck and prune are relatively lengthy operations that still go silent while making the end-user wait. They learned to give progress output like other slow operations.
- The set of built-in function-header patterns for various languages knows MATLAB.
- “git log –format=<format>” learned new %g[nNeE] specifiers to show information from the reflog entries when walking the reflog (i.e. with “-g”).
- “git pull” can be used to fetch and merge an annotated/signed tag, instead of the tip of a topic branch. The GPG signature from the signed tag is recorded in the resulting merge commit for later auditing.
- “git log” learned “–show-signature” option to show the signed tag that was merged that is embedded in the merge commit. It also can show the signature made on the commit with “git commit -S”.

2.29. Version Control Systems
• "git branch --edit-description" can be used to add descriptive text to explain what a topic branch is about.

• "git fmt-merge-msg" learned to take the branch description into account when preparing a merge summary that "git merge" records when merging a local branch.

• "git request-pull" has been updated to convey more information useful for integrators to decide if a topic is worth merging and what is pulled is indeed what the requestor asked to pull, including:
  – the tip of the branch being requested to be merged;
  – the branch description describing what the topic is about;
  – the contents of the annotated tag, when requesting to pull a tag.

• "git pull" learned to notice 'pull.rebase' configuration variable, which serves as a global fallback for setting 'branch.<name>.rebase' configuration variable per branch.

• "git tag" learned “--cleanup” option to control how the whitespaces and empty lines in tag message are cleaned up.

• "gitweb" learned to show side-by-side diff.

Also contains minor documentation updates and code clean-ups.

Fixes since v1.7.8 Unless otherwise noted, all the fixes since v1.7.8 in the maintenance releases are contained in this release (see release notes to them for details).

git 1.7.8 See Also:

https://raw.github.com/gitster/git/master/Documentation/RelNotes/1.7.8.txt

• Some git-svn, git-gui, git-p4 (in contrib) and msysgit updates.

• Updates to bash completion scripts.

• The build procedure has been taught to take advantage of computed dependency automatically when the compiler supports it.

• The date parser now accepts timezone designators that lack minutes part and also has a colon between “hh:mm”.

• The contents of the /etc/mailname file, if exists, is used as the default value of the hostname part of the committer/author e-mail.

• “git am” learned how to read from patches generated by Hg.

• “git archive” talking with a remote repository can report errors from the remote side in a more informative way.

• “git branch” learned an explicit --list option to ask for branches listed, optionally with a glob matching pattern to limit its output.

• “git check-attr” learned “--cached” option to look at .gitattributes files from the index, not from the working tree.

• Variants of “git cherry-pick” and “git revert” that take multiple commits learned to “--continue” and “--abort”.

• “git daemon” gives more human readable error messages to clients using ERR packets when appropriate.

• Errors at the network layer is logged by “git daemon”.

• “git diff” learned “--minimal” option to spend extra cycles to come up with a minimal patch output.

• “git diff” learned “--function-context” option to show the whole function as context that was affected by a change.

• “git difftool” can be told to skip launching the tool for a path by answering ‘n’ to its prompt.
• “git fetch” learned to honor transfer.fsckobjects configuration to validate the objects that were received from the other end, just like “git receive-pack” (the receiving end of “git push”) does.
• “git fetch” makes sure that the set of objects it received from the other end actually completes the history before updating the refs. “git receive-pack” (the receiving end of “git push”) learned to do the same.
• “git fetch” learned that fetching/cloning from a regular file on the filesystem is not necessarily a request to unpack a bundle file; the file could be “.git” with “gitdir: <path>” in it.
• “git for-each-ref” learned “%(contents:subject)”, “%(contents:body)” and “%(contents:signature)”. The last one is useful for signed tags.
• “git grep” used to incorrectly pay attention to .gitignore files scattered in the directory it was working in even when “–no-index” option was used. It no longer does this. The “–exclude-standard” option needs to be given to explicitly activate the ignore mechanism.
• “git grep” learned “–untracked” option, where given patterns are searched in untracked (but not ignored) files as well as tracked files in the working tree, so that matches in new but not yet added files do not get missed.
• The recursive merge backend no longer looks for meaningless existing merges in submodules unless in the outermost merge.
• “git log” and friends learned “–children” option.
• “git ls-remote” learned to respond to “–h”(elp) requests.
• “mediawiki” remote helper can interact with (surprise!) MediaWiki with “git fetch” & “git push”.
• “git merge” learned the “–edit” option to allow users to edit the merge commit log message.
• “git rebase -i” can be told to use special purpose editor suitable only for its insn sheet via sequence.editor configuration variable.
• “git send-email” learned to respond to “–h”(elp) requests.
• “git send-email” allows the value given to sendemail.aliasfile to begin with “~” to refer to the $HOME directory.
• “git send-email” forces use of Authen::SASL::Perl to work around issues between Authen::SASL::Cyrus and AUTH PLAIN/LOGIN.
• “git stash” learned “–include-untracked” option to stash away untracked/ignored cruft from the working tree.
• “git submodule clone” does not leak an error message to the UI level unnecessarily anymore.
• “git submodule update” learned to honor “none” as the value for submodule.<name>.update to specify that the named submodule should not be checked out by default.
• When populating a new submodule directory with “git submodule init”, the $GIT_DIR metainformation directory for submodules is created inside $GIT_DIR/modules/<name>/ directory of the superproject and referenced via the gitfile mechanism. This is to make it possible to switch between commits in the superproject that has and does not have the submodule in the tree without re-cloning.
• “gitweb” leaked unescaped control characters from syntax hiliter outputs.
• “gitweb” can be told to give custom string at the end of the HTML HEAD element.
• “gitweb” now has its own manual pages.

Also contains other documentation updates and minor code cleanups.

**Fixes since v1.7.7** Unless otherwise noted, all fixes in the 1.7.7.X maintenance track are included in this release.

• HTTP transport did not use pushurl correctly, and also did not tell what host it is trying to authenticate with when asking for credentials. (merge deba493 jk/http-auth later to maint).
Development tools, Release 2012.06.18

- “git blame” was aborted if started from an uncommitted content and the path had the textconv filter in effect. (merge 8518088 ss/blame-textconv-fake-working-tree later to maint).
- Adding many refs to the local repository in one go (e.g. “git fetch” that fetches many tags) and looking up a ref by name in a repository with too many refs were unnecessarily slow. (merge 17d68a54d jp/get-ref-dir-unsorted later to maint).
- Report from “git commit” on untracked files was confused under core.ignorecase option. (merge 395c7356 jk/name-hash-dirent later to maint).
- “git merge” did not understand “/<pattern>” as a way to name a commit.
- “git push” on the receiving end used to call post-receive and post-update hooks for attempted removal of non-existing refs. (merge 160b81ed ph/push-to-delete-nothing later to maint).
- Help text for “git remote set-url” and “git remote set-branches” were misspelled. (merge c49904e fc/remote-set-url-usage-fix later to maint). (merge 656cdf0 jc/remote-set-branches-usage-fix later to mai

git 1.7.0  See Also:
https://raw.githubusercontent.com/gitster/git/master/Documentation/RelNotes/1.7.0.txt

Notes on behaviour change

- “git push” into a branch that is currently checked out (i.e. pointed at by HEAD in a repository that is not bare) is refused by default.

  Similarly, “git push $there :$killed” to delete the branch $killed in a remote repository $there, when $killed branch is the current branch pointed at by its HEAD, will be refused by default.

  Setting the configuration variables receive.denyCurrentBranch and receive.denyDeleteCurrent to ‘ignore’ in the receiving repository can be used to override these safety features.

- “git send-email” does not make deep threads by default when sending a patch series with more than two messages. All messages will be sent as a reply to the first message, i.e. cover letter.

  It has been possible already to configure send-email to send “shallow thread” by setting sendemail.chainreplyto configuration variable to false. The only thing this release does is to change the default when you haven’t configured that variable.

- “git status” is not “git commit –dry-run” anymore. This change does not affect you if you run the command without argument.

- “git diff” traditionally treated various “ignore whitespace” options only as a way to filter the patch output. “git diff –exit-code -b” exited with non-zero status even if all changes were about changing the amount of whitespace and nothing else; and “git diff -b” showed the “diff –git” header line for such a change without patch text.

  In this release, the “ignore whitespaces” options affect the semantics of the diff operation. A change that does not affect anything but whitespaces is reported with zero exit status when run with –exit-code, and there is no “diff –git” header for such a change.

- External diff and textconv helpers are now executed using the shell. This makes them consistent with other programs executed by git, and allows you to pass command-line parameters to the helpers. Any helper paths containing spaces or other metacharacters now need to be shell-quoted. The affected helpers are GITEXTERNALDIFF in the environment, and diff.*.command and diff.*.textconv in the config file.

- The –max-pack-size argument to ‘git repack’, ‘git pack-objects’, and ‘git fast-import’ was assuming the provided size to be expressed in MiB, unlike the corresponding config variable and other similar options accepting a size value. It is now expecting a size expressed in bytes, with a possible unit suffix of ‘k’, ‘m’, or ‘g’.
mercurial

See Also:

- https://secure.wikimedia.org/wikipedia/en/wiki/Mercurial
- http://mercurial.selenic.com/wiki/ (official site)
- https://plus.google.com/u/0/112477627281544607334/posts
- hg clone https://pvergain@bitbucket.org/mirror/mercurial-crew

Contents

- mercurial
  - Introduction
  - Glossary
  - Downloads
  - Mercurial issue_tracker
  - Mercurial versions
  - Mercurial sources
    * Main repositories
  - Links
  - Other infos

Introduction

Mercurial is a cross-platform, distributed revision control tool for software developers. It is mainly implemented using the Python programming language, but includes a binary diff implementation written in C.

It is supported on Windows, Unix-like systems including FreeBSD, OSX and Linux.
Mercurial is primarily a command line program but graphical user interface extensions are available. All of Mercurial’s operations are invoked as keyword options to its driver program hg, a reference to the chemical symbol of the element mercury.

Mercurial’s major design goals include high performance and scalability, decentralized, fully distributed collaborative development, robust handling of both plain text and binary files, and advanced branching and merging capabilities, while remaining conceptually simple.[3] It includes an integrated web interface.

The creator and lead developer of Mercurial is Matt Mackall. Released under the terms of the GNU General Public License (version 2 or any later version).

Glossary

See Also:
http://mercurial.selenic.com/wiki/Glossary

Downloads

- http://mercurial.selenic.com/

Mercurial issue_tracker

Mercurial issue_tracker See Also:
http://bz.selenic.com/

La nouvelle du passage à bugzilla

Hi, folks -

After many years of beloved service, the roundup-based bug tracker at mercurial.selenic.com/bts has been retired. The new bug tracker is a Bugzilla instance at http://bz.selenic.com/

All users, bugs and attachments have been migrated.

If you had an account on the old bug tracker, the email address you used to sign up is your username. Passwords could not be migrated - you’ll need to select “forgot password” to set up a new password.

You can change your login name afterwards if need be.

There’s not a 1-1 mapping between old and new bug IDs, alas.

Mercurial versions

mercurial versions

mercurial 2.1
add: support adding explicit files in subrepos
annotate: append newline after non newline-terminated file listings (BC)
annotate: show full changeset hash when invoked with --debug and -c
annotate: support diff whitespace filtering flags (issue3030)
bisect: remove superfluous parameter in label()
bookmarks: automatically advance bookmark on naked update (BC) (issue2894)
bookmarks: clarify help for -i/--inactive
bookmarks: mark divergent bookmarks with book8pathalias when source in [paths]
bookmarks: shadow divergent bookmarks of foo with foo@n
bugzilla: make XMLRPC interface support http and https access
changeset_printer: display changeset phase on debug level
cloned: get all bookmarks before updating
copies: rewrite copy detection for non-merge users
copies: split the copies api for "normal" and merge cases (API)
cygwin: add cygwin specific normcase logic
diff: always use / in paths in diff
dispatch: exit with 8-bit exit code
filectx: fix cmp() of file starting with ‘\\n’
forget: support forgetting explicit paths in subrepos
grep: make multiline mode the default (BC)
hgcia: fix diffstat support
hooks: prioritize run order of hooks
id: add command line options for handling ssh and https urls
largefiles: add --normal option to hg add (issue3061)
largefiles: add error checking to tags conversion (issue3092)
largefiles: add tests for uncovered codepaths (issue3092)
largefiles: cache new largefiles for new heads when pulling
largefiles: check if largefile could be found when archiving (issue3193)
largefiles: correctly download new largefiles when merging
largefiles: correctly handle dirstate status when rebasing
largefiles: correctly handle newly added largefile on other side of merge
largefiles: display remote errors from putlfile (issue3123) (issue3149)
largefiles: don’t reference uninitialized variable (issue3092)
largefiles: fix caching largefiles from an aliased repo (issue3212)
largefiles: fix confusion upon removal of added largefile (issue3176)
largefiles: fix inappropriate locking (issue3182)
largefiles: fix output of hg summary (issue3060)
largefiles: implement addremove (issue3064)
largefiles: optimize performance of status on largefiles repos (issue3136)
largefiles: optimize status when files are specified (issue3144)
largefiles: remove empty directories upon update (issue3202)
mail: mbox handling as a part of mail handling, refactored from patchbomb

mail: use quoted-printable for mime encoding to avoid too long lines (issue3075)
merge: give a special message for internal:merge failure (issue3105)
mq: add a warning about uncommitted changes for qfinish
mq: have mq create secret changeset only
notify: add option for writing to mbox
notify: change behavior of "changegroup" hook
phase: report phase movement

phases: {phase} template keyword display the phase name
phases: add a phases command to display and manipulate phases
phases: add a phases.publish option
phases: implements simple revset symbol
phases: prevent mq to qimport immutable change
phases: prevent rebase to rebase immutable changeset
push: propagate --new-branch and --ssh options when pushing subrepos
rebase: add a "D" short option for detach
rebase: allow --detach when --rev is used

rebase: reinstate old-style rev spec support for the source and base (issue3181)
rebase: write series file without removed mq patches
revset: add remote() predicate to lookup remote revisions
revset: allow slashes in symbols

setup: fix py2exe generation broken by c3a6ec304055 (issue3116)
ssh: quote remote paths (issue2983)
sslutil: abort properly if no certificate received for https connection
sslutil: show fingerprint when cacerts validation fails
status: support revsets with --change

subrepo: avoid syncing bookmarks twice on clone (issue3191)
tag: invalidate tag cache immediately after adding new tag (issue3210)
util: don’t encode ':' in url paths
win32mbcs: allow win32mbcs extension to be enabled on cygwin platform

mercurial 2.0

mercurial 2.0.2 (2012-01-01) This is a regularly-scheduled bug-fix release.

• alias: shortcut command matching so shadowing works properly (issue3104)
• branch: warn on branching
• changelog: handle decoding of NULs in extra more carefully (issue3156)
• convert: improve exception reporting for SVN logstream
• diff: don’t crash when diffing a revision with a deleted subrepo (issue3153)
• fetch: fix unneeded commit when no merge attempted (issue2847)
• fetch: patch cornercase in children calculation (issue2773)
• largefiles: copy files in binary mode (issue3164)
• largefiles: don’t require a user cache (issue3088) (issue3155)
• largefiles: fix ‘hg status’ abort after merge
• largefiles: handle merges between normal files and largefiles (issue3084)
• merge: check filename case collision between changesets for branch merging
• pathauditor: switch normcase logic according to case sensitivity of filesystem
• progress: check for ui.quiet and ui.debugflag before we write
• rollback: always call destroyed() (regression from 1.9)
• util: fix url.str() for windows file URLs
• windows: use upper() instead of lower() or os.path.normcase()

mercurial 2.0.0 (2011-11-01)  See Also:
http://mercurial.selenic.com/wiki/WhatsNew#Mercurial_2.0_.282011-11-01.29

Major features
• New core graft command (similar to transplant extension)
• New largefiles extension

Core changes
• commit: abort when there are uncommitted subrepos, use the -S option to recurse
• help: most commands now have usage examples with help -v
• import: add –edit switch
• revset: add more keywords for bisection
• revert: introduce short option -C for –no-backup
• log: add new ‘bisect’ style that prints the bisection status
• hgweb: add a “web/logoimg” setting to customize the web logo image
• subrepo: pull revisions on demand when archiving hg subrepos

Extension changes
• color: add styles for tags
• convert: added bookmarks support in filemap
• eol: add new eol.fix-trailing-newline setting
• eol: eol.only-consistent can now be specified in .hgeol
• export: add %m to file format string (first line of the commit message)
• mq: make queue print current queue name
• rebase: add –edit switch
• rebase: add –rev option to rebase
• rebase: allow rebase on ancestor

2.29. Version Control Systems
Development tools, Release 2012.06.18

- share: introduce unshare command
- transplant: add –edit option

Mercurial sources


hg clone https://pvergain@bitbucket.org/mirror/mercurial-crew

Main repositories

- http://selenic.com/repo/hg – the main repository, bleeding edge, solely managed by mpm who often pulls from other repositories (mirrored at http://hg.intevation.org/mercurial/)
- http://selenic.com/repo/hg-stable – the stable repository, solely managed by mpm, contains official releases with safe and important fixes applied, always a subset of the main repository

Bugfixes are committed to the branch named stable and pushed to the main repository. This branch is automatically pushed to hg-stable to provide a separate repository.

No new features are going to the stable branch, mainline (the default branch) will be branched again near each release.

Links

See Also:


Other infos

hg mercurial commands

hg branch

hg bookmark  See Also:

- http://mercurial.selenic.com/wiki/Bookmarks

Contents

- hg bookmark
  - History
  - Overview
  - Secret bookmark
History  
Since version 1.8 (d4ab9486e514) the bookmark command is part of core.

The behaviour of the core bookmarks is different from the original Bookmarks Extension.

For example the track.current option was removed and the behavior of track.current=True is now the default behavior of bookmarks.

For more details please check the Bookmarks wiki page.

Author  David Soria

Overview  
Bookmarks are references to commits that are automatically updated when new commits are made.

If you do hg bookmark feature the feature bookmark refers to the current changeset. As you work and commit changes the bookmark will move forward with every commit you do.

The bookmark will always point to the latest revision in your line of work. Since bookmarks are automatically updated when committing to the changeset they are pointing to, they are especially useful to keep track of different heads.

They can therefore be used for trying out new features or pulling changes that have yet to be reviewed. Bookmarks can be used to access the commit whenever a usual lookup is allowed (wherever a command takes -r revision, revision can be a bookmark name), therefore you can merge and update bookmarks by their names.

Secret bookmark  
They do not exist yet, The suggestion is to have:

- local-bookmark: only exists in your repository and not exchanged.
- public-bookmark: are always exchanged on pull//push//clone (as non secret-changeset are)

hg forget  See Also:

- http://www.selenic.com/mercurial/hg.1.html#forget
- git_rm

git equivalent command  See Also:

http://stackoverflow.com/questions/4135351/is-hg-forget-the-same-as-git-rm-cache

for the record: hg forget is the same as git rm --cached

hg graft  New in version 2.5: hg graft

See Also:

- http://www.selenic.com/mercurial/hg.1.html#graft

hg graft [OPTION]... REVISION...

This command uses Mercurial’s merge logic to copy individual changes from other branches without merging branches in the history graph.

This is sometimes known as ‘backporting’ or ‘cherry-picking’.

By default, graft will copy user, date, and description from the source changesets.

Changesets that are ancestors of the current revision, that have already been grafted, or that are merges will be skipped.
If a graft merge results in conflicts, the graft process is interrupted so that the current merge can be manually resolved. Once all conflicts are addressed, the graft process can be continued with the `-c/-continue` option.

Note
The `-c/-continue` option does not reapply earlier options.

**Examples**  
copy a single change to the stable branch and edit its description:

```
hg update stable
hg graft --edit 9393
```

graft a range of changesets with one exception, updating dates:

```
hg graft -D "2085::2093 and not 2091"
```

continue a graft after resolving conflicts:

```
hg graft -c
```

show the source of a grafted changeset:

```
hg log --debug -r tip
```

Returns 0 on successful completion.

### Appendix A

This keyword can be used to find revisions that “match” one or more fields of a given set of revisions.

A revision matches another if all the selected fields (description, author, files, date, parents and/or substate) match the corresponding values of those fields on the source revision.

By default this keyword looks for revisions that whose metadata match (description, author and date) making it ideal to look for duplicate revisions.

matching takes 2 arguments (the second being optional):

1. `-rev`: a revset representing a _single_ revision (e.g. `tip`, `.`, `p1(,)`, etc) 2. `-field(s) to match`: an optional field or list of fields to match.

   By default matching will match the metadata fields (description, author and date). When matching more than one field, they must be input as a list. When matching a single field there is no need to surround it in () to make it a list.
Examples  1.- Look for revisions with the same metadata (author, description and date) as the 11th revision:

    hg log -r "matching(11)"

2.- Look for revisions with the same description as the 11th revision:

    hg log -r "matching(11, description)"

You do not need to type the whole ‘description’ word. You could also use ‘descript’ or ‘desc’ but not ‘d’ because ‘d’ also matches ‘date’ and ‘date’ takes preference because fields are matched in alphabetical order.

3.- Look for revisions with the same author as the current revision:

    hg log -r "matching(., author)"

You could use ‘user’ rather than ‘author’ to get the same result.

4.- Look for revisions with the same description _AND_ author as the tip of the repository:

    hg log -r "matching(tip, (author, desc))"

5.- Look for revisions touching the same files as the the tip of the repository

    hg log -r "matching(tip, files)"

6.- Look for revisions whose subrepos are on the same state as the parent of the tip of the repository

    hg log -r "matching(p1(tip), substate)"

7.- Look for revisions whose author and files both match the tip or the parent of the tip of the repository:

    hg log -r "matching(p1(tip):tip, (a, f))"

hg merge  See Also:


Is it possible to, instead of merging one branch into another existing branch, merge 2 branches into a 3rd new branch?

Just merge your 2 existing branches and consider the merge as the tip of the 3rd new branch and the previous heads of the merged branches as your 1st and 2nd branch:

    o  changeset: 3:92692c4a6b12
    |\  bookmark: masala
    | |  summary: merge salt and pepper
    | | | o  changeset: 2:a5f955adf03d
    | | | |  bookmark: pepper
    | | | |  summary: add some pepper
    | | | o  changeset: 1:2b56f2dc115f
    | | | |  bookmark: salt
    | | | |  summary: add some salt
    | | o  changeset: 0:e992ce7dd508
    | summary: initial

Here bookmarks have been used to mark different lines in development. So if you want to work in the new 3rd branch, update to masala, if you want to work on your 1st branch, update to salt, and similar for the 2nd branch update to pepper before you continue to work and commit.

If you prefer working with named branches (instead of bookmarks), just issue a hg branch masala before you commit the merge of revision 2 and 1.
The basic message is that although the graph only has one head, you are free to interpret it as 3 different lines of development.

Now, let’s say you want to continue the work in the 2nd branch, pepper:

```bash
$ hg up pepper
... hack ...
$ hg ci -m "need more pepper"
```

And then you have some ideas for the salt thing:

```bash
$ hg up salt
... hack ...
$ hg ci -m "less salt please"
```

Now the history graph shows your 3 branches more clearly:

```
o changeset: 5:d1f8eb72119a
  | bookmark: salt
  | summary: less salt please
  |
  | o changeset: 4:acc9b01f584f
  |  | bookmark: pepper
  |  | summary: need more pepper
  |  |
  +---o changeset: 3:92692c4a6b12
  |   | bookmark: masala
  |   | summary: merge salt and pepper
  |   |
  | o changeset: 2:a5f955adf03d
  |  | summary: add some pepper
  |  |
  | o changeset: 1:2b56f2dc115f
  |   | summary: add some salt
  |   |
  o changeset: 0:e992ce7dd508
  summary: initial
```

An alternative to bookmarks and named branches is to use different clones for individual branches. That is you clone your repo with the unmerged branches and merge them in the clone. Which approach is best, depends on your specific workflow and personal preferences.

---

**Hi,**

>> I have to admit I don’t know how to read this output or what I should look >> for in the way of conflicts.
Are these resources helpful? (You don’t have to read them all, but if the first doesn’t work, try the following one.)

About log:
- http://mercurial.selenic.com/wiki/GraphlogExtension

About merges, heads and parents:
- http://hginit.com/04.html
- http://mercurial.selenic.com/wiki/Merge
- http://mercurial.selenic.com/wiki/UnderstandingMercurial#Revisions.2C_changesets.2C_heads.2C_and_tip
- http://mercurial.selenic.com/wiki/MultipleHeads

To help us troubleshoot your issue, can you confirm this: you were in a clone with the 2.5 branch checked out (“hg id” will tell that), you pulled from 2.6, and the update action (from “pull -u”) failed.

>> [...] I think that would make my local change unnecessary.

Correct.

>> So, my next project is to try and figure out how to undo my change.

http://mercurial.selenic.com/wiki/PruningDeadBranches#Using_clone If you follow that method, don’t forget to copy your .hg/hgrc to the new clone.

>> This seemed to work:

“Seemed” is the right word :( More on http://mercurial.selenic.com/wiki/Revert or “hg help revert”.

> No. Revert just reverts local (non-committed) changes.

This is untrue; Skip used a -r argument. See above. One thing to understand is that revert changes file contents only, whereas update moved you on the revision graph.

Advanced commands like strip (bundled with mq BTW, not rebase) should not be recommended lightly to people who are still learning the normal use of Mercurial, IMO. Throwing mentions of rebase, strip, transplant can be harmful. Let’s focus on clone, pull, update and merge first. (That’s why the devguide tries to select one or two workflows; we know that Mercurial is hella flexible, but choice is not a good thing when you’re learning.)

Regards

de  R. David Murray <rdmurray@bitdance.com>
à Glenn Linderman <v+python8g.nevcal.com>
cc python-dev@python.org
date 23 mars 2011 07:46

> I don’t recall 5-12 step sequences in the DVCS PEP when I read it once,
> for any of the tools, but things progressed from the time I read it, so
> maybe they would all have longer sequences.
> > Back when I used non-distributed VCS systems like SCCS, RCS, PVCS, CVS,
> and Clearcase, I don’t recall any operations that took more than 2 or 3
> commands to achieve (merges, of course, were onerous in some of those,
> and locking sometimes stalled progress in some ways in some of those).

The 12 step cases are exactly merges, and the merges themselves are easy with hg.

For simply getting a patch in to the default branch, svn had:

```bash
svn up
<apply patch etc>
svn up
svn ci
```

Whereas hg has:

```bash
hg pull -u
<apply patch etc>
hg commit
hg push
```

Same number of steps, but as has been pointed out, the hg push guarantees no one else has made changes, whereas that safety-belt svn up before the svn ci doesn’t.

**hg phase  See Also:**

- [http://mercurial.selenic.com/wiki/Phases#Introduction](http://mercurial.selenic.com/wiki/Phases#Introduction)
- [http://mercurial.selenic.com/wiki/Phases#Available_Phases](http://mercurial.selenic.com/wiki/Phases#Available_Phases)
- [http://mercurial.selenic.com/wiki/Phases#Phase_Movements](http://mercurial.selenic.com/wiki/Phases#Phase_Movements)
- [http://mercurial.selenic.com/wiki/Phases#Upgrade_Notes](http://mercurial.selenic.com/wiki/Phases#Upgrade_Notes)
- [http://mercurial.selenic.com/wiki/Phases#Publishing_Repository](http://mercurial.selenic.com/wiki/Phases#Publishing_Repository)

**Contents**

- hg phase
  - Introduction
  - Introduction To Mercurial Phases (Part I)
  - Introduction To Mercurial Phases (Part II)
    - Controlling automatic phase movement
  - Introduction To Mercurial Phases (Part III)
  - Conclusion

**Introduction**  The phase concept improves safety of history rewriting and control over changesets exchanged (read more). This phase concept aims to “just works” in a transparent manner for any user (read more). It is part of Core and always enabled in any new client but doesn’t prevent older client to work on a repository (read more). Advanced user may decides to handle phase manually (read more).

Like bookmarks, phases are not stored in history and thus are not permanent and leave no audit trail.

This concept is introduced in Mercurial 2.1
Introduction To Mercurial Phases (Part I)  See Also:
   - http://www.logilab.org/blogentry/88203

Introduction To Mercurial Phases (Part II)  See Also:
   - http://www.logilab.org/blogentry/88219

This is the second part of a series of posts about the new phases feature we implemented for mercurial 2.1.
The first part talks about how phases will help mercurial users, this second part explains how to control them.

Controlling automatic phase movement  Sometimes it may be desirable to push and pull changesets in the draft phase to share unfinished work. Below are some cases:
   - pushing to continuous integration,
   - pushing changesets for review,
   - user has multiple machines,
   - branch clone.

You can disable publishing behavior in a repository configuration file:

[phases]
publish=False

Introduction To Mercurial Phases (Part III)  See Also:
   - http://www.logilab.org/blogentry/88259

This is the final part of a series of posts about the new phases feature we implemented for mercurial 2.1.
The first part talks about how phases will help mercurial users, the second part explains how to control them.
This one explains what people should take care of when upgrading.

Conclusion  Mercurial’s phases are a simple concept that adds always on and transparent safety for most users while not preventing advanced ones from doing whatever they want.

Behind this safety-enabling and useful feature, phases introduce in Mercurial code the concept of sharing mutable parts of history.
The introduction of this feature paves the way for advanced history rewriting solutions while allowing safe and easy sharing of mutable parts of history.
I’ll post about those future features shortly.

**hg patches/hg push --mq**

Floris Bruynooghe <floris.bruynooghe@gmail.com>
à  bitbucket-users@googlegroups.com
date 9 mars 2011 14:04
objet [Bitbucket] Re: How do I commit to a patch queue?
liste de diffusion <bitbucket-users@googlegroups.com> Filtrer les messages de cette liste de diffusion
Basically the .hg/patches directory is a versioned mercurial repo itself. Your problem is that you forgot to commit this patches repo itself before pushing it.

The old way of pusing the changes is:

```bash
cd .hg/patches
hg st
hg ci -m "updated this cool patch"
hg push
```

Or with a new enough mercurial the --mq option can be used:

```bash
hg st --mq
hg ci --mq -m "...
hg push --mq
```

Regards Floris

**Subrepository**


**Introduction**  Subrepositories is a feature that allows you to treat a collection of repositories as a group. This will allow you to clone, commit to, push, and pull projects and their associated libraries as a group.

This feature was introduced in a preliminary form in Mercurial 1.3 and has been improved steadily since then. There are still some commands that lack proper support for sub-repositories, but we will fix them as we come across them and as we figure out how to best make them subrepo-aware.

For those used to Subversion, this concept is closest to what you can achieve with Subversion directories marked with the svn:externals property. Mercurial 1.5 has support for using Subversion repositories as subrepos.

**Basic usage**

**Start**  To start using subrepositories, you need two repositories, a main repo and a nested repo:

```bash
$ hg init main $ cd main $ hg init nested
Next we’ll mark ‘nested’ as a subrepository by creating an entry for it in the special .hgsub file.

$ echo nested = nested > .hgsub
$ hg add .hgsub

On the left hand side of the assignment is the path in our working dir (‘nested’), and the right hand side is a URL or path to pull from. Here we’re simply going to pull from ‘nested’ using a path relative to main. This says ‘anyone who can find our main repo can find the nested repo just by tacking nested onto that path’.

Note that the nested repository must actually exist for the line in .hgsub to do anything. For instance, if rather than creating a local nested repository you attempt to link to a pre-existing remote one, you must ALSO clone that repository:

```bash
$ echo nested = https://example.com/nested/repo/path > .hgsub
$ hg add .hgsub
$ hg clone https://example.com/nested/repo/path nested
```

If you intend to track something other than the current revision of the default branch this is also the time when you would update the subrepo to the desired revision.

Now let’s add some files to nested, and add them.

```bash
$ echo test > nested/foo
$ hg -R nested add nested/foo
$ hg -R nested commit --message “Initial commit.”
```

2.1.1. SVN subrepositories
As of version 1.5, Mercurial can also support other repository types for your subrepo. For example, if you wanted a subrepo that referred to a Subversion repository, you would do something like this:

$ echo 'nested = [svn]https://example.com/nested/trunk/path' >.hgsub $ hg add .hgsub $ svn co https://example.com/nested/trunk/path nested Currently, Mercurial treats all URLs that do not begin with a ‘[<repo type>]’ as beginning with ‘[hg]’.

2.1.2. Git subrepositories

As of version 1.8, Mercurial supports git subrepositories:

$ echo ‘nested = [git]git://example.com/nested/repo/path.git’ > .hgsub $ hg add .hgsub $ git clone git://example.com/nested/repo/path.git nested

When we commit, Mercurial will attempt to create a consistent snapshot of the state of the entire project and its subrepos. It does this by first attempting to commit in all modified subrepos and then recording the state of all subrepos.

$ hg ci -m test committing subrepository nested

Subrepo states are stored in a .hgsubstate file that is managed automatically by Mercurial.

2.3. Update

Whenever Mercurial encounters a changeset containing subrepos, it will attempt to pull the specified subrepos and update them to the appropriate state:

$ cd ..
$ hg clone main main2
updating working directory
pulling subrepo nested
requesting all changes
adding changesets
adding manifests
adding file changes
added 1 changesets with 1 changes to 1 files
2 files updated, 0 files merged, 0 files removed, 0 files unresolved

$ cat main2/nested/foo test Subrepos may also contain their own subrepos and Mercurial will recurse as necessary.

Push  Mercurial will automatically attempt to first push all subrepos of the current repository when you push. This will ensure new changesets in subrepos are available when referenced by top-level repositories.

2.5. Pull

Notably, the ‘pull’ command is not recursive. This is because Mercurial won’t know which subrepos are required until an update to a specific changeset is requested. To get pull and update in one step, use ‘pull –update’.

Note that this matches exactly how pull works without subrepositories:

‘hg pull’ gives you the upstream changesets but doesn’t affect your working directory. ‘hg update’ updates the contents of your working directory (both in the top repo and in all subrepos) 2.6. Synchronizing in subrepositories

Subrepos don’t automatically track the latest changeset of their sources. Instead, they are updated to the changeset that corresponds with the changeset checked out in the top-level changeset. This is so developers always get a consistent set of compatible code and libraries when they update.

Thus, updating subrepos is a manual process. Simply run ‘hg pull’ and ‘hg up’ in the target subrepo, test in the top-level repo, then commit in the top-level repo to record the new combination.
The pull in the subrepo is necessary because the automatic subrepo pull that happens when you run ‘hg up’ in the top-level repo only pulls the exact revision it needs, as specified in .hgsubstate. Any changesets that are more recent than that one will not get pulled.

2.7. Delete

To remove a subrepo from the parent repo, you must delete the subrepo definition from the .hgsub file at the top level of the parent repo. Once you do this, the subrepo tree will show up as a set of unknown files when you run hg status, and you can delete the files.

3. Caveats

As this is a complex new feature, there are a number of rough edges:

Some commands require a -S or –subrepos switch to operate on subrepos (available since Mercurial 1.7) Many commands are not aware of subrepos Update/merge currently can’t remove subrepositories entirely as that might lose local-only changes There’s no support for merging across renaming/moving subrepos Collisions between normal files and subrepos are not handled Subrepo updates currently ignores URLs and revision filters Commit doesn’t propagate some flags like -A to subrepos

hg strip   See Also:

• http://mercurial.selenic.com/wiki/Strip
• http://hgbook.red-bean.com/read/mercurial-queues-reference.html#id446269

The hg strip command removes a revision, and all of its descendants, from the repository. It undoes the effects of the removed revisions from the repository, and updates the working directory to the first parent of the removed revision.


de Antoine Pitrou <solipsis@pitrou.net>
heure de l’expéditeur   Envoyé à 03:41 (GMT+01:00). Heure locale : 10:18.
ad python-dev@python.org
date 23 mars 2011 03:41
objet  Re: [Python-Dev] Workflow proposal
On Wed, 23 Mar 2011 10:39:01 +0900

"Stephen J. Turnbull" <stephen@xemacs.org> wrote:
> Executive summary:
> If we’re really serious about serializing the public branches, mq
> seems to be the way to go.

I really think that at this point we should continue practicing with the current setup before deciding on refinements or changes.

> > 6. Use "hg strip" (dangerous!) to delete the local merges to 3.2 and
> > "default". Leave the original commit in "3.1" alone.
> > I would suggest "hg strip --keep" which leaves the working copy
> > unchanged.

Now, “hg strip” should definitely be absent of any recommended or even suggested workflow. It’s a power user tool for the experimented developer/admin. Not the average hg command.
By the way, whether or not mq gets used is pretty much anyone’s private decision (unless we decide that patches must get split into logical units and mailbombed for review ala mercurial-devel, which would be a significant departure from our development habits).

Regards

hg qclone  See Also:

• http://mercurial.selenic.com/wiki/MqMerge

de  rndblnch <rndblnch@gmail.com>
à  python-dev@python.org
date  24 mars 2011 15:31
objet  Re: [Python-Dev] Let’s get PEP 380 into Python 3.3

> > Anyone without push rights for hg.python.org may want to fork the
> > mirror the bitbucket folks set up during the PyCon sprints to avoid
> > having to push the whole repository up to a separate hosting site:
> > https://bitbucket.org/mirror/cpython
> >
> > Done here:
> > <https://bitbucket.org/rndblnch/cpython-pep380/>

Now that I have figured out how to use patch queues with bitbucket, I will maintain greg’s pep380 implementation as a patch on top of cpython here: <https://bitbucket.org/rndblnch/cpython-pep380/>

Testing it should be as simple as:

% hg qclone https://bitbucket.org/rndblnch/cpython-pep380
% cd cpython-pep380
% hg qpush pep380

and then the usual ./configure etc.

To test on an already existing clone of <http://hg.python.org/cpython>:

% hg clone http://hg.python.org/cpython
% cd cpython
% hg update -C default

you should enable mq extension...:

% hg init --mq

get the pep380 patch and apply it...:

% pushd .hg/patches/
% hg pull https://bitbucket.org/rndblnch/cpython-pep380/.hg/patches
% hg update
% popd
% hg qpush pep380
and then the usual ./configure etc.

The patch is now visible here: <https://bitbucket.org/rndblnch/cpython-pep380/qseries?apply=t&qs_apply=pep380>

Hope this helps,

hg revert
  • http://hgbook.red-bean.com/read/finding-and-fixing-mistakes.html

[Python-Dev] Useful page in the Hg book by Nick Coghlan <ncoghlan@gmail.com>

Nick Coghlan <ncoghlan@gmail.com>
à python-dev@python.org
date 17 mars 2011 00:09
objet [Python-Dev] Useful page in the Hg book

As I’m sure there will be plenty of erring as we get used to Hg: http://hgbook.red-bean.com/read/finding-and-fixing-mistakes.html

For simple cases of attempting to push a single commit that gets rejected by the server, hg rollback/hg pull/hg commit/hg push/ may provide a cleaner history than doing a dummy merge.

hg backout also further emphasises the benefits of working in feature branches - I was able to revert the default branch in my sandbox to match upstream by backing out a single changeset.

Cheers, Nick.

hg rebase See Also:
http://mercurial.selenic.com/wiki/RebaseProject

-------- Message original --------
Sujet: Re: [Python-Dev] Hg: inter-branch workflow
Date : Sun, 20 Mar 2011 16:39:50 +0000
De : Georg Brandl <g.brandl@gmx.net>
Pour : python-dev@python.org

On 20.03.2011 16:21, Guido van Rossum wrote: > What is “rebase”? Why does everyone want it and hate it at the same time?

Basically, rebase is a way to avoid having pointless merge commits on the same branch.

Let’s say you have the following history in your local repository:

... — X — A — B

where X is the last commit that is in the remote repository, and A and B are commits you made locally. Now you pull from the remote, and see that others have pushed more history in the meantime. The graph now looks this:

    A — B
     /
... — X — C — D — E

To get your history pushed, you need to make a merge commit:

    A — B
   /

... — X — C — D — E — M
Then you can push A, B and M to the remote.

Now, “hg pull –rebase” prevents that by “re-basing” the A-B history line onto the latest remote head. After rebasing, the history looks like this:

... — X — C — D — E — A’ — B’
without the need to merge before the push.

Obviously, if the merge above gave conflicts, the rebasing process will also give conflicts. In both cases you need to solve them before continuing.

The reason why rebasing is not universally applied is that the rebased changesets are different from the original ones (therefore I wrote A’ and B’) – even if the diff is the same, the parents are not, and therefore the changeset id (hash) changes. This is called “changing history”, and frowned upon by purists. In reality it works fine if you know the limits: rebasing really only should be applied if the changesets are not already known somewhere else, only in the local repo you’re working with.

Georg

hg resolve  See Also:
http://mercurial.selenic.com/wiki/ResolveConflict

Example

gc resolve -m source/conf.py

hg status  See Also:
• http://mercurial.selenic.com/wiki/status

The hg strip command removes a revision, and all of its descendants, from the repository. It undoes the effects of the removed revisions from the repository, and updates the working directory to the first parent of the removed revision.

Example hg status --rev 749:746

gc status --rev 749:746

Mercurial extensions  See Also:
• http://mercurial.selenic.com/wiki/ShareExtension

Color Mercurial extensions

[extensions] color  See Also:

In $HOME/Mercurial.ini (Windows)

[extensions]
color =
hg-git extension  

See Also:

- http://hg-git.github.com/
- http://pypi.python.org/pypi/hg-git/0.2.6
- using appharbor with mercurial and hg-git

push and pull from a Git server using Mercurial.

This extension lets you communicate (push and pull) with a Git server.

This way you can use Git hosting for your project or collaborate with a project that is in Git.

It is basically feature complete and pretty stable now, but there are still some edge cases we don’t handle well yet and it may be slow in some circumstances.

The user interface is also still subject to change. However, there are now a lot of people using it effectively, so please do use it and let me know if you run into anything.

Source

- hg clone git://github.com/schacon/munger.git

The easy way  

Run easy_install hg-git, then add make sure the following is in your ~/.hgrc:

```ini
[extensions]
hgext.bookmarks =
hggit =
```

...and that’s it!

The more involved way  

First, install version 0.4.0 or newer of dulwich. You can do easy_install ‘dulwich>=0.4.0’ if you have setuptools installed. Clone this repository somewhere, or download a snapshot, then make the ‘extensions’ section in your ‘~/.hgrc’ file look something like this:

```ini
[extensions]
hgext.bookmarks =
hggit = [path-to]/hg-git/hggit
```

That will enable the Hg-Git extension for you. The bookmarks section is not compulsory, but it makes some things a bit nicer. Bookmarks will be translated to git heads when pushing.

shelve extensions  

See Also:

http://mercurial.selenic.com/wiki/ShelveExtension

onsub Mercurial extensions  

See Also:

- http://mercurial.selenic.com/wiki/OnsubExtension
- https://bitbucket.org/aragost/onsub/

This extension is not distributed with Mercurial

Author  

MartinGeisler
Contents

- onsub Mercurial extensions
  - Overview
  - Configuration

Overview  The onsub extension will traverse all subrepositories and execute a command in each.
This can be used to update all subrepositories with one command:

$ hg onsub "hg pull -u"

The extension provides a number of environment variables for you to use in your commands:

- **HG_REPO**: Absolute path to the top-level repository in which the onsub command was executed.
- **HG_SUBPATH**: Relative path to the current subrepository from the top-level repository.
- **HG_SUBURL**: URL for the current subrepository as specified in the containing repository’s .hgsub file.
- **HG_SUBSTATE**: State of the current subrepository as specified in the containing repository’s .hgsubstate file.

Consult hg help onsub after enabling the extension for the full and up-to-date documentation.

Configuration  Configure your .hgrc to enable the extension by adding following lines:

```
[extensions]
onsub = path/to/onsub/onsub.py
```

Mercurial development

[PATCH] eol: add an allcsethook (issue2665)

Antoine Pitrou <solipsis@pitrou.net>
à mercurial-devel@selenic.com
date  28 février 2011 20:28
objet  [PATCH] eol: add an allcsethook (issue2665)

# HG changeset patch
# User Antoine Pitrou <solipsis@pitrou.net>
# Date 1298920868 -3600
# Branch stable
# Node ID f8e1163f93a599c1948f208a11b752d5c3dd4b15
# Parent  c54c35307a7ab547de32268d42ef2ecdc37bec2
eol: add an allcsethook (issue2665)
eol’s built-in hook checks the content of modified files in the final changegroup changeset. This allows people to fix their blunder in a subsequent commit and then push the whole bunch.

Another wish can be to reject any individual changeset containing wrong line endings, because they pollute history (as well as the subsequent line ending-fixing changesets). The allcsethook achieves that.

diff --git a/hgext/eol.py b/hgext/eol.py
--- a/hgext/eol.py
+++ b/hgext/eol.py
@@ -79,6 +79,14 @@
Remember to enable the eol extension in the repository where you install the hook.

Another hook named ‘eol.allcsethook’ is provided. While ‘eol.hook’ checks the contents of files in the last changeset of a changegroup (allowing people to fix their bogus commit in a later changeset before pushing both), ‘eol.allcsethook’ examines all incoming changesets for line ending mistakes. This places a stricter burden on developers while allowing for a cleaner public history (no repository pollution with line ending-fixing changsets).

See :hg: ‘help patterns’ for more information about the glob patterns used.

"""
@@ -128,6 +136,40 @@

+def checkfile(ui, repo, node, file):
+ """check that a file in the given node has expected EOLs"""
+ for pattern, target in ui.configitems('encode'):
+ if match.match(repo.root, '', [pattern])(file):
+ data = node[file].data()
+ if ((target == "to-lf" and \r\n in data) or
+ (target == "to-crlf" and singlelf.search(data))):
+ return target
+ # Ignore other rules for this file
+ break
+
+def allcsethook(ui, repo, node, hooktype, **kwargs):
+ """verify that files in all new csets have expected EOLs"""
+ failed = []
+ for rev in xrange(repo[node].rev(), len(repo)):
+ ctx = repo[rev]
+ for f in ctx.files():
+ if f not in ctx:
+ # File was removed in rev
+ continue
+ target = checkfile(ui, repo, ctx, f)
+ if target:
+ failed.append((ctx, f, target))
+ if failed:
+ msgs = []
+ for ctx, f, target in failed:
+ if target == "to-lf":
+ msgs.append(_("%s in %s should not have CRLF line endings") % (f, ctx))
+ elif target == "to-crlf":
+ ...
+    msgs.append(_("%s in %s should not have LF line endings")
+                 % (f, ctx))
+    raise util.Abort("\n".join(msgs))
+
def hook(ui, repo, node, hooktype, **kwargs):
    """verify that files have expected EOLs"""
    files = set()
@@ -137,18 +179,13 @@
    for f in files:
        if f not in tip:
            continue
-        for pattern, target in ui.configitems('encode'):
-            if match.match(repo.root, '', [pattern])(f):
-                data = tip[f].data()
-                if target == "to-lf" and "\r\n" in data:
-                    raise util.Abort(_("%s should not have CRLF line endings")
-                                      % f)
-                elif target == "to-crlf" and singlelf.search(data):
-                    raise util.Abort(_("%s should not have LF line endings")
-                                      % f)
-                    # Ignore other rules for this file
-                    break
-    target = checkfile(ui, repo, tip, f)
+    target = checkfile(ui, repo, tip, f)
    if target == "to-lf":
        raise util.Abort(_("%s should not have CRLF line endings")
                        % f)
    elif target == "to-crlf":
        raise util.Abort(_("%s should not have LF line endings")
                        % f)

def preupdate(ui, repo, hooktype, parent1, parent2):
    #print "preupdate for %s: %s -> %s" % (repo.root, parent1, parent2)
    diff --git a/tests/test-eol-hook.t b/tests/test-eol-allcsethook.t
    copy from tests/test-eol-hook.t b/tests/test-eol-allcsethook.t
    copy to tests/test-eol-allcsethook.t
    --- a/tests/test-eol-hook.t
    +++ b/tests/test-eol-allcsethook.t
@@ -10,7 +10,7 @@
> eol =
> [hooks]
- > pretxnchangegroup = python:hgext.eol.hook
+ > pretxnchangegroup = python:hgext.eol.allcsethook
> EOF
$ hg clone main fork
 updating to branch default
@@ -20,16 +20,15 @@
 Create repo
 $ cat > .hgeol <<EOF
 > [patterns]
- > mixed.txt = BIN
- > crlf.txt = CRLF
- > ***.txt = native
+ > a.txt = CRLF
+ > ***.txt = LF
 > EOF
 $ hg add .hgeol

2.29. Version Control Systems
+ $ hg commit -m 'Commit .hgeol' -d 2010-01-01 -u someuser

+ $ printf "first\nsecond\nthird\n" > a.txt
+ $ hg add a.txt
+ $ hg commit -m 'CRLF a.txt' -d 2010-01-01 -u someuser
+ $ printf "third\n" > b.txt
+ $ hg add b.txt
+ $ hg commit -m 'CRLF b.txt' -d 2010-01-01 -u someuser
+ $ printf "first\nsecond" > a.txt
+ $ hg rm b.txt
+ $ hg commit -m 'CRLF a.txt, rm b.txt' -d 2010-01-01 -u someuser

$ hg push ../main

pushing to ../main

searching for changes

1078 Chapter 2. Development

- added 1 changesets with 1 changes to 1 files
- error: pretxnchangegroup hook failed: a.txt should not have CRLF line endings
- added 3 changesets with 3 changes to 2 files
- error: pretxnchangegroup hook failed: a.txt in 586b8cd9e131 should not have LF line endings
- b.txt in ca83fc6c9fac should not have CRLF line endings

transaction abort!
rollback completed

- abort: a.txt should not have CRLF line endings
- abort: a.txt in 586b8cd9e131 should not have LF line endings
- b.txt in ca83fc6c9fac should not have CRLF line endings

[255]

- $ printf "first\nsecond\nthird\n" > a.txt
- $ hg commit -m 'LF a.txt (fixed)'
- $ hg push ../main
- pushing to ../main
- searching for changes
- adding changesets
- adding manifests
- adding file changes
- added 2 changesets with 2 changes to 2 files

- $ printf "first\nsecond\nthird\n" > a.txt
- $ hg commit -m 'CRLF a.txt'
- $ printf "first\nsecond\nthird\n" > a.txt
- $ hg add a.txt
- $ hg commit -m 'CRLF a.txt' -d 2010-01-01 -u someuser
- $ printf "third\n" > b.txt
- $ hg add b.txt
- $ hg commit -m 'CRLF b.txt' -d 2010-01-01 -u someuser
- $ printf "first\nsecond" > a.txt
- $ hg rm b.txt
- $ hg commit -m 'CRLF a.txt, rm b.txt' -d 2010-01-01 -u someuser

$ hg push ../main

pushing to ../main

searching for changes

1078 Chapter 2. Development

- added 2 changesets with 2 changes to 1 files
- error: pretxnchangegroup hook failed: a.txt should not have CRLF line endings
- added 3 changesets with 3 changes to 2 files
- error: pretxnchangegroup hook failed: a.txt in 586b8cd9e131 should not have LF line endings
- b.txt in ca83fc6c9fac should not have CRLF line endings

transaction abort!
rollback completed

- abort: a.txt should not have CRLF line endings
- abort: a.txt in 586b8cd9e131 should not have LF line endings
- b.txt in ca83fc6c9fac should not have CRLF line endings

[255]

- $ printf "first\nsecond\nthird\n" > a.txt
- $ hg commit -m 'LF a.txt (fixed)'
- $ hg push ../main
- pushing to ../main
- searching for changes
- adding changesets
- adding manifests
- adding file changes
- added 2 changesets with 2 changes to 1 files

- $ printf "first\nsecond\nthird\n" > crlf.txt
- $ hg add crlf.txt
- $ hg commit -m 'LF crlf.txt'
- $ hg push ../main
- pushing to ../main

1078 Chapter 2. Development
- searching for changes
- adding changesets
- adding manifests
- adding file changes
- added 1 changesets with 1 changes to 1 files
- error: pretxnchangegroup hook failed: crlf.txt should not have LF line endings
- transaction abort!
- rollback completed
- abort: crlf.txt should not have LF line endings
- [255]

- \$ printf "first\r\nsecond\r\nthird\r\n" > crlf.txt
- \$ hg commit -m ‘CRLF crlf.txt (fixed)’
- \$ hg push ../main
- searching for changes
- adding changesets
- adding manifests
- adding file changes
- added 2 changesets with 2 changes to 1 files

:: index:: PEP 385

PEP 385 (Migrating from Subversion to Mercurial)  See Also:

- http://www.python.org/dev/peps/pep-0385/
- http://hg.python.org/hooks/

>> Author: antoine.pitrou
>> Date: Mon Feb 28 19:22:36 2011
>> New Revision: 88676
>>
>> Log:
>> Update PEP 385 with latest hooks work
>>
>> Modified:
>> peps/trunk/pep-0385.txt
>> Modified: peps/trunk/pep-0385.txt
>>
>> --- peps/trunk/pep-0385.txt (original)
>> @@ -262,7 +262,22 @@
>> on every build slave for the branch in which the changeset occurs.
>>
>> The ‘hooks repository’ contains ports of these server-side hooks to
>> Mercurial. One additional hook could be beneficial:
>> +Mercurial, as well as a couple additional ones:
>> +
>> + check branch heads: a hook to reject pushes which create a new head on

2.29. Version Control Systems
Hello,

I’d like to use a private git repository, then I need to migrate my actual private mercurial repository.

I’ve tried to use https://bitbucket.org/repo/import but my mercurial url is not found by bitbucket.

You’ll have to convert it locally using hg-git or similar. This blog posts should help you:

hg github  See Also:

• http://pypi.python.org/pypi/hg-github/0.1.2

hg-github is a Mercurial extension that wraps hg-git, and supports a work-flow where repositories are hosted on Bitbucket and mirrored on GitHub.

This work-flow normally requires adding Git paths to each repository’s config file, and creating Mercurial bookmarks pointing to the GitHub repository’s branch name. hg-github takes care of these for you automatically.

hg-github is BSD licensed.

Mercurial development process

• http://mercurial.selenic.com/wiki/MercurialDevelopmentProcess

Python Developer’s Guide

• http://potrou.net/hgdevguide/

Hg init

• http://hginit.com/index.html

Tortoise Hg manual

• http://tortoisehg.bitbucket.org/manual/2.0

Aragost blog

• http://www.aragost.com/en/services/mercurial/blog/

Mozilla mercurial guide

• http://wiki.frenchmozilla.org/index.php/Mercurial

Mercurial hgrc (config hg)

Mercurial hgrc examples
Pour ceux que ça intéresse, voici mes configurations git & hg :

- https://bitbucket.org/haypo/misc/src/tip/conf/gitconfig
- https://bitbucket.org/haypo/misc/src/tip/conf/hgrc

**hgrc de Victor Stinner**  **See Also:**
https://bitbucket.org/haypo/misc/src/tip/conf/hgrc

```
[ui]
username = Victor Stinner <victor.stinner@haypocalc.com>
verbose = True

# on commit, hgeditor opens an editor with the diff for the changelog
# http://selenic.com/hg/raw-file/tip/hgeditor
editor = /home/haypo/bin/hgeditor

diff
    git = on
    showfunc = on

[extensions]
color=
    # enable hg glog
graphlog=
pager=
    progress=
    rebase=
    record=
    # enable hg strip
mq=
    # enable hg histedit: https://bitbucket.org/durin42/histedit/src
hg_histedit=

[color]
    # vim colors (bg=dark theme)
diff.diffline = green bold
diff.file_a = green bold
diff.file_b = green bold
diff.hunk = yellow bold
diff.deleted = red bold
diff.inserted = cyan bold

[pager]
    # less -F: quit if one screen
    # less -S: chop long lines
    # less -R: raw (keep colors)
    # less -X: no init (don’t see termcap init sequences)
pager = LESS='FSRX' less

[merge-patterns]
** = internal:merge
```

1082 Chapter 2. Development
Mercurial a besoin d’être débridé en activant plein d’options et extensions que ça soit utilisable de base. Mais avec la bonne config, il y a des moins en moins de différences entre git & hg. Les développeurs hg m’ont par exemple dit qu’ils ont (enfin !) implémenté la syntaxe rev^, rev^n, rev~n, ... Et un autre développeur m’a dit qu’avec la prochaine version de Mercurial, le rebase aura des performances correct (grâce à un nouveau format de stockage).

Les commandes que j’utilise couramment

```
git log [fichier] <=> hg log [fichier]
git blame [fichier] <=> hg blame [fichier]
git st <=> hg st
```

```
git diff [fichier] <=> hg diff [fichier]
git ci <=> hg ci
```

```
git add -i [fichier] <=> hg record [fichier]
git rebase -i rev <=> hg histedit rev
```

```
git out <=> hg out
```

```
git pull --rebase && git push <=> hg pull --rebase && hg push
```

**Mort aux commits de merge (de Mercurial) ! Rebase for the win!**

Victor

(pour “git out” j’ai “triché” avec mon ~/.gitconfig)

### gitconfig de Victor Stinner

See Also:

https://bitbucket.org/haypo/misc/src/tip/conf/gitconfig

```
[color]
diff = auto
ui = auto

[user]
name = Victor Stinner
email = victor.stinner@haypocalc.com

[alias]
out = log remotes/origin/master..HEAD --pretty='format:%Cred%h%Creset %s' --color
ci = commit -v --untracked-files=no
st = status
rebasei = rebase -i remotes/origin/master

# Add "&& true" to not pass command line options to the last command
# eg. git svnup --help doesn’t pass --help option to svn rebase
svnout = log git-svn..HEAD --pretty='format:%Cred%h%Creset %s' --color
svnup = !git svn rebase && true
svncl = !git svn rebase && git svn dcommit && true
svnrebase = rebase -i git-svn
```

---

**Conseils d’Aurélien Bompard**

Date : Sun, 2 Oct 2011 08:55:01 +0200

De : Aurelien Bompard <aurelien@bompard.org>
Répondre à : Liste de diffusion des membres de l’association <afpy-membres@lists.afpy.org>
Pour : afpy-membres@lists.afpy.org

> (pour “git out” j’ai “triché” avec mon ~/.gitconfig)

Tu as “git cherry -v” qui ressemble un peu à ce que tu fais (mais sans la couleur à ma connaissance).

> git pull –rebase && git push <=> hg pull –rebase && hg push > Mort aux commits de merge (de Mercurial) !

Rebase for the win!

Pour le faire automatiquement tu peux aussi faire un:

```
git config branch.master.rebase true
git config branch.autosetuprebase always
```

La première ligne ajoutera “–rebase” automatiquement sur tes pull, et la seconde règlera cette option sur les nouvelles branches que tu pourrais créer.

Au passage, je vois aussi que tu utilises color.ui = auto. Il se peut que le pager ait du mal à afficher les couleurs dans le terminal, auquel cas il faut que tu ajoutes:

```
git config --global core.pager "less -FXRS"
```

(tu l’as fait dans ta conf Mercurial)

Et enfin, il est toujours bon d’ajouter:

```
git config --global core.editor vim
git config --global merge.tool vimdiff
```

parce que bon, quand même, faut pas déconner. (je reste dans le ton du thread)

Aurélien

Mercurial hgrc examples

```
[paths]
default = https://pvergain@bitbucket.org/pvergain/devtools_doc

[ui]
username = Patrick Vergain <pvergain@gmail.com>
verbose = True

[diff]
git = on
showfunc = on

[extensions]
color=
  # enable hg glog
graphlog=
pager=
progress=
rebase=
record=
  # enable hg strip
```
mq=

[color]
# vim colors (bg=dark theme)
diff.diffline = green bold
diff.file_a = green bold
diff.file_b = green bold
diff.hunk = yellow bold
diff.deleted = red bold
diff.inserted = cyan bold

[pager]
# less -F: quit if one screen
# less -S: chop long lines
# less -R: raw (keep colors)
# less -X: no init (don’t see termcap init sequences)
pager = LESS=’FSRX’ less

[merge-patterns]
** = internal:merge

[hostfingerprints]

Mercurial .hgignore

Example

  .*\.pyc
  .*\.egg
  .*\.so
  .*\.o
  ^build/
  ^_build/

Mercurial news

Aragost blog  See Also:

http://www.aragost.com/en/services/mercurial/blog/

Posted on December 9, 2010 by Martin Geisler  Martin Geisler holds a PhD in Computer Science from the University of Aarhus, Denmark. He has been involved in the Mercurial project for three years and is now a member of the core developer team.

He relocated to Zurich in the beginning of 2010 to work full-time as a Mercurial consultant for aragost Trifork.

This is a new blog centered around the Mercurial universe: we will review new extensions, provide tips and tricks, show interesting workflows, and generally do my best to keep you updated on the latest news about Mercurial.

Mercurial python news
I don’t think we have to ask them to do anything special, as long as they can submit their contributions under the form of a patch. Whether they use named branches, separate clones, mercurial queues, the pbranch extension, or even the basic hg-as-an-svn-synonym workflow you suggested should be none of our business, IMO.

A DVCS allows non-committers to manage (augment, fix, synchronize) their patches more easily until they get committed. I don’t think it will change the committers’ work a lot, though. I don’t know how Mercurial could make the task of committing an outsider’s patch significantly simpler.

**using appharbor with mercurial and hg-git**  
See Also:

- [hg-git extension](http://support.appharbor.com/kb/getting-started/using-appharbor-with-mercurial-and-hg-git)

**2011/03/10 appharbor now integrates with bitbucket**  
A few weeks ago AppHarbor and Bitbucket met up for a few beers to see how we could address requests from the community for Mercurial integration.

We’re happy to announce that AppHarbor now integrates with Bitbucket!

AppHarbor is a cloud service that competes with Microsoft Azure, a platform that lets developers deploy their code directly from Visual Studio.

Every time you push to a Bitbucket repository, Bitbucket will notify AppHarbor of the push. AppHarbor will then pull your code, build and deploy it.

In the coming weeks we’ll be working together to further improve the integration between the two platforms.

**Mercurial tips**  
See Also:

- [http://hgit.com](http://hgit.com)
- [http://docs.python.org/devguide/](http://docs.python.org/devguide/)

**Tips1 : merging heads**  
See Also:


A while ago Éric suggested a nice tip to make merges easier and since I haven’t seen many people using it and now I got a chance to use it again, I think it might be worth showing it once more:
Development tools, Release 2012.06.18

# so assume you just committed some changes:
$ hg ci Doc/whatsnew/3.3.rst -m 'Update and reorganize the whatsnew entry for PEP 393.'

# you push them, but someone else pushed something in the meanwhile, so the push fails
$ hg push
pushing to ssh://hg@hg.python.org/cpython
searching for changes
abort: push creates new remote heads on branch 'default'!
(you should pull and merge or use push -f to force)
# so you pull the other changes
$ hg pull -u
pulling from ssh://hg@hg.python.org/cpython
searching for changes
adding changesets
adding manifests
adding file changes
added 4 changesets with 5 changes to 5 files (+1 heads)
not updating, since new heads added
(run 'hg heads' to see heads, 'hg merge' to merge)
# and use "hg heads ." to see the two heads (yours and the one you pulled) in the current branch
$ hg heads .
changeset: 72521:e6a2b54c1d16
tag: tip
user: Victor Stinner <victor.stinner at haypocalc.com>
date: Thu Sep 29 04:02:13 2011 +0200
summary: Fix hex_digit_to_int() prototype: expect Py_UCS4, not Py_UNICODE
changeset: 72517:ba6ee5cc9ed6
user: Ezio Melotti <ezio.melotti at gmail.com>
date: Thu Sep 29 08:34:36 2011 +0300
summary: Update and reorganize the whatsnew entry for PEP 393.
# here comes the tip: before merging you switch to the other head (i.e. the one pushed by Victor),
# if you don’t switch, you’ll be merging Victor changeset and in case of conflicts you will have to review
# and modify his code (e.g. put a Misc/NEWS entry in the right section or something more complicated)
$ hg up e6a2b54c1d16
6 files updated, 0 files merged, 0 files removed, 0 files unresolved
# after the switch you will merge the changeset you just committed, so in case of conflicts
# reviewing and merging is much easier because you know the changes already
$ hg merge
1 files updated, 0 files merged, 0 files removed, 0 files unresolved
(branch merge, don’t forget to commit)
# here everything went fine and there were no conflicts, and in the diff I can see my last changeset
$ hg di
diff --git a/Doc/whatsnew/3.3.rst b/Doc/whatsnew/3.3.rst
[...]
# everything looks fine, so I can commit the merge and push
$ hg ci -m 'Merge heads.'
$ hg push
pushing to ssh://hg@hg.python.org/cpython
searching for changes
remote: adding
changesets

2.29. Version Control Systems
remote: adding manifests
remote: adding file changes
remote: added 2 changesets with 1 changes to 1 files
remote: buildbot: 2 changes sent successfully
remote: notified python-checkins at python.org of incoming changeset
   ba6ee5cc9ed6
remote: notified python-checkins at python.org of incoming changeset
e7672fe3cd35

This tip is not only useful while merging, but it’s also useful for python-checkins reviews, because the “merge” mail has the same diff of the previous mail rather than having 15 unrelated changesets from the last week because the committer didn’t pull in a while.

Tip 2 – extended diffs If you haven’t already, enable git diffs, adding to your ~/.hgrc the following two lines:

```
[diff]
git = True
```

(This is already in the devguide, even if ‘git = on’ is used there. The mercurial website uses git = True too.) More info: http://hgtip.com/tips/beginner/2009-10-22-always-use-git-diffs/

Tip 3 – extensions: color, progress I personally like the ‘color’ extension, it makes the output of commands like ‘hg diff’ and ‘hg stat’ more readable (e.g. it shows removed lines in red and added ones in green). If you want to give it a try, add to your ~/.hgrc the following two lines:

```
[extensions]
color =
```

If you find operations like pulling, updating or cloning too slow, you might also want to look at the ‘progress’ extension, which displays a progress bar during these operations:

```
[extensions]
progress =
```

Tip 4 – porting from 2.7 to 3.2 The devguide suggests:

```
hg export a7df1a869e4a | hg import --no-commit -
```

but it’s not always necessary to copy the changeset number manually. If you are porting your last commit you can just use ‘hg export 2.7’ (or any other branch name):

* using the one-dir-per-branch setup:
  wolf at hp:~/dev/py/2.7$ hg ci -m ‘Fix some bug.’
  wolf at hp:~/dev/py/2.7$ cd ../3.2
  wolf at hp:~/dev/py/3.2$ hg pull -u ../2.7
  wolf at hp:~/dev/py/3.2$ hg export 2.7 | hg import --no-commit -
* using the single-dir setup:
  wolf at hp:~/dev/python$ hg branch
  2.7
  wolf at hp:~/dev/python$ hg ci -m ‘Fix some bug.’
  wolf at hp:~/dev/python$ hg up 3.2 # here you might enjoy the progress
extension
wolf at hp:~/dev/python$ hg export 2.7 | hg import --no-commit -

And then you can check that everything is fine, and commit on 3.2 too. Of course it works the other way around (from 3.2 to 2.7) too.
I hope you’ll find these tips useful.
Best Regards, Ezio Melotti

TortoiseHg  See Also:

- https://bitbucket.org/tortoisehg/thg/wiki/thg

Downloads

- http://bitbucket.org/tortoisehg/thg/downloads

Source on bitbucket

hg clone https://bitbucket.org/tortoisehg/thg

On ubuntu

sudo aptitude install libqscintilla-dev
sudo aptitude install python-qscintilla2

Windows

- http://bitbucket.org/tortoisehg/thg/downloads/tortoisehg-2.0.0-hg-1.8-x86.msi

TortoiseHg news

[thg-dev] 2.0 is out, 2 mars 2011 00:38

de  Steve Borho <steve@borho.org>
date  2 mars 2011 00:38
objet  [thg-dev] 2.0 is out
liste de diffusion  <thg-dev.googlegroups.com> Filtrer les messages de cette liste de diffusion

There were a slew of last minute snafus, but the packages are now out in the wild. The website and wiki are updated and backups are uploading to codeplex right now. I probably won’t upload packages to sourceforge till tomorrow.
Thanks for all the hard work.
I’m going to be off-grid for about 4 hours, I have a hockey game to attend. If something comes up.. it’ll have to wait :)

mercurial use case  See Also:

http://hg.piranha.org.ua/sphinxedhg/docs/test-hgrc.html

2.29. Version Control Systems
We’ve been using Mercurial for a couple of months now and it’s improved our deployment process A LOT already. This is the good part.

The system we have in place is working but it’s still error prone if you’re not careful or rushing. This leave me wondering if there’s ways we could improve it or... maybe we’re completely off the track :)

Our environment consist of:

Local development workstation (each developer) Development server (hosting the DB & the central repository) An acceptance server (Where QA is done) A staging server (Where we stage the release branch, we then robocopy to the live systems) A little background on the reason why we switched:

We are in a work environment that often have us switching from task to task, leaving many pending tasks. Many would become stale and clutter up the main branch when we were back on CVS. Deployment was a nightmare as you had to work around lines that needed to go live and others that didn’t using Beyond Compare.

Mercurial with named branches and easy merging solves this for us. So not knowing what to expect we set it up.

First we cleaned up our production source, pruning dead files, etc.

We FTP’d that on staging and made this our new repository as "default", this was to be our stable branch ready to be deployed at all time.

Afterward, we did an hg clone to the development server and had each developer hg clone from the development default branch.

On acceptance where we do QA we did an hg clone of the development server’s default branch.

At this point we have a stable copy of the code everywhere, everyone is eager to start!

local machine are pushing to dev, acceptance pulls from dev and staging is completely isolated and can pull from wherever if the path is provided.

Now the idea behind this was that the default branch on our system would always be a copy of the code on the live server, provided that we remembered to pull before starting a new branch. When starting a new feature we would:

hg pull -b default #synch up hg update default hg branch {newFeature} #newFeature completely isolated from other changes.

*work on {newFeature} Oh no! A bug! this is unrelated to what I am currently working on lets call it {bugFix111}. This appear to call for a new branch independant of my feature; go back to updated default. This will isolate newFeature and bugFix111 from each other and each can go live independently as they are based on default.

hg update default hg pull -u hg branch {bugFix111} Once work is completed on say {bugFix111}

hg push -F {bugFix111} #send our fix to the main central repo on dev. Go to acceptance:

hg pull -b {bugFix111} #pull the fix from the central repo (dev). hg merge {bugFix111} #merge the code in the default QA branch. hg commit -m “Merging {bugFix111} into default”

*QA sign off on the fix We have to branch off acceptance - default were QA take place and release where we merge the stuff as it get signed off.

hg update release hg merge {bugFix111} #fix is now ready to go live hg commit -m “Merging {bugFix111} into release” On staging:

hg pull -b release {PATH TO ACCEPTANCE REPO} hg merge release hg commit -m “Merging {bugFix111} into staging default” hg tag release{date} *robocopy to live *run task that pull from staging to dev so that they synch up again. This has been working for us and save up some deployment time as it’s a breeze to just robocopy the stable release branch.
Issues  What we have noticed is:

It’s easy to goof up a merge when merging the second time on release, this seem against the flow. We can break it after the QA sign off.

Could get QA to test our release branch as well but it seems like duplicating resource, the goal is just to have features isolated and being able to send them one at a time.

We can completely blow it up by merging release over something wrong, e.g. hg merge release when you are on the default on acceptance completely overwrites it.

If we forget to pull before starting a new branch we are working off the wrong base.

Few other oddities but those are the biggest hurdles.

I realize this is a long post, but hopefully the answers will help other Mercurial newbies like me trying to set up a decent workflow at their company.

Response  Why not pull on staging from the QA branch? Then the merge job has already been done and validated, i.e if the commit has some manual merge you will import it on staging also.

Otherwise you have to replicate the merge on staging as you are doing it now.

Because the QA branch at any given time can have feature1, feature2, ..., featureN merged on it for QA. What if I want to send only feature3? I can’t pull the entire branch to staging. This is what I meant by stale changes, we don’t necessarily follow a specific release cycle. If we let the QA test each independently by allowing him to switch branch then the merge is never tested.

@jfrobishow: But, if you want to ship feature 3 only, shouldn’t then QA also has tested this feature isolated from the other features? In that case there should be a merge for feature 3 only in the QA clone, this could be pulled by staging.

@Oben Sonne: good point... currently we don’t do that. Everything that is ready for QA is in the default branch on acceptance. The QA test each feature one by one yes, but never isolated from other potential defect introduced by new features also on the QA branch. Once they sign off we pull it to staging but that’s never tested as a whole or very little. So far, this has worked as it allowed us to catch future conflicts before they happened live. Maybe it’s our QA system that needs an overhaul :) –

mercurial web hosting

bitbucket  See Also:

- https://bitbucket.org/

Figure 2.53: Bitbucket logo
Bitbucket is a web-based hosting service for projects that use either the Mercurial or Git revision control systems.

Bitbucket offers both commercial plans and free accounts.

It offers free accounts with unlimited numbers of private repositories (which can have up to five users in the case of free accounts) as of September 2010. Private repositories are not shown on profile pages - if a user has only private repositories, the website will claim “This user has no repositories”.

The service is written in Python.

It is similar to GitHub, which uses only Git. In a 2008 blog post, Bruce Eckel compared Bitbucket favorably to Launchpad, which uses Bazaar.

**History**  
Bitbucket was previously an independent startup. On September 29th, 2010, Bitbucket was acquired by VC-funded Atlassian.

Initially, Bitbucket only offered hosting support for Mercurial projects.

**On October 3rd, 2011, Bitbucket officially announced support for Git hosting**

**Bitbucket API**

**Bitbucket API**  
See Also:

- [http://confluence.atlassian.com/display/BITBUCKET/Using+the+Bitbucket+REST+APIs;jsessionid=E538608FE49371B36BE031C443A3FE9F](http://confluence.atlassian.com/display/BITBUCKET/Using+the+Bitbucket+REST+APIs;jsessionid=E538608FE49371B36BE031C443A3FE9F)

**Introduction to Bitbucket’s REST APIs**  
Bitbucket’s REST APIs provide access to resources (data entities) via URI paths.

To use a REST API, your application will make an HTTP request and parse the response.

By default, the response format is JSON. If you wish, you can request XML or YAML instead of JSON.

Your methods will be the standard HTTP methods like GET, PUT, POST and DELETE.

Because the REST API is based on open standards, you can use any web development language to access the API.

Bitbucket’s APIs provide the following REST resources:

- Changesets
- Emails
- Events
- Followers
- Groups
- Groups Privileges
- Invitations
- Issue Comments
- Issues
- Privileges
- Repositories
- Services
- Source
Subversion (svn)

See Also:

- http://subversion.apache.org/
- http://code.google.com/p/svnbook/

Apache Subversion (often abbreviated SVN, after the command name svn) is a software versioning and a revision control system distributed under a free license.

Developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation.

Its goal is to be a mostly-compatible successor to the widely used Concurrent Versions System (CVS).

Svn commands

Svn commands

See Also:

- http://subversion.apache.org/

svn branch tools
Development tools, Release 2012.06.18

svn combinator

svn combinator  See Also:

  • https://bazaar.launchpad.net/~divmod-dev/divmod.org/trunk/view/head:/Combinator/README.txt

Combinator is a tool to help developers use and manage multiple branches of software hosted in a Subversion (SVN from now on) repository.

If you are not familiar with branches and merging in SVN, I recommend you read Chapter 4 of Version Control with Subversion at some point - it’s not a pre-requisite for using Combinator or reading this tutorial, but it definitely helps to have an understanding of what’s going on behind the scenes.

Conversion to

Subversion conversion to

Subversion conversion to git

Confluence  See Also:

http://blogs.atlassian.com/2012/01/moving-confluence-from-subversion-to-git/

Versions

Apache Subversion versions  See Also:

  • http://subversion.apache.org/roadmap.html
  • http://svn.apache.org/repos/asf/subversion/trunk/CHANGES

Apache Subversion 1.8.0  See Also:

  • http://subversion.apache.org/docs/release-notes/1.8.html
  • http://svn.apache.org/repos/asf/subversion/trunk/CHANGES

See Also:

  • http://svn.haxx.se/dev/archive-2011-10/0152.shtml
  • http://subversion.apache.org/docs/release-notes/1.7.html

Contents

  • Announce

Announce  We are pleased to announce the release of Apache Subversion 1.7.5.

This is the most complete Subversion release to date, and we encourage users of Subversion to upgrade as soon as reasonable.

Please see the release announcement and the change log for more information about this release.

To get this release from the nearest mirror, please visit our download page.
Apache Subversion 1.7.0  See Also:
  • http://svn.haxx.se/dev/archive-2011-10/0152.shtml
  • http://subversion.apache.org/docs/release-notes/1.7.html

Announce  I’m happy to announce the release of Apache Subversion 1.7.0. This is a major new feature release of Subversion, containing many new features, bugfixes and general usability improvements. The full extent of the changes is too large to enumerate here, and I encourage you to read the release notes (linked to below).

Release notes for the 1.7.x release series may be found at:
  • http://subversion.apache.org/docs/release-notes/1.7.html

Tools

Apache Subversion tools  See Also:
  • http://subversion.apache.org/
  • http://fr.wikipedia.org/wiki/Apache_Subversion

Tortoisesvn  See Also:
  • http://tortoisesvn.net/

TortoiseSVN is an easy-to-use SCM / source control software for Microsoft Windows and possibly the best standalone Apache Subversion client there is.

It is implemented as a Windows shell extension, which makes it integrate seamlessly into the Windows explorer.

Since it’s not an integration for a specific IDE you can use it with whatever development tools you like.

Tortoisesvn versions

Tortoisesvn 1.7.7  See Also:
http://tortoisesvn.net/Changelog.txt

Tortoisesvn 1.7.4  See Also:
http://tortoisesvn.net/Changelog.txt

We’re proud to announce that TortoiseSVN 1.7.3 has been released. It is linked against Subversion 1.7.2

Due to some nasty bugs in TortoiseSVN 1.7.2 which in some specific situations could make it crash, we’re releasing this version out of sync with SVN releases.

TortoisePlink was also updated to Plink 0.62 which had a security issue fixed.

We recommend that you update to this version as soon as possible.

  • **BUG:** Issue #169: Error “Access to C:Users<windowsAccount> was denied.” (Stefan)
  • **BUG:** Issue #170: svn url formatting broken on `switch`. (Stefan)
  • **BUG:** Issue #171: Right-Click Refresh not working in Repo Browser correctly. (Stefan)
  • **NEW:** Issue #172: Option to turn off case-fixing. (Stefan)
• BUG: Issue #174: applying patch fails with “path/path not found”. (Stefan)
• BUG: Issue #175: Scroll to moved block doesn’t work. (Stefan)
• BUG: Issue #176: Diff settings “restore defaults” doesn’t work correctly. (Stefan)
• BUG: Issue #177: TortoiseMerge shows long lines incorrect. (Stefan)
• BUG: Issue #178: copy of changed files list when sorted copies wrong items. (Stefan)
• NEW: Issue #180: Optimize repo browser for big repositories. (Stefan)
• BUG: Issue #182: Tagging externals stopped working. (Stefan)
• BUG: Issue #183: exporting single files from log dialog doesn’t work. (Stefan)

Tortoisesvn 1.7.3 See Also:
http://tortoisesvn.net/Changelog.txt

We’re proud to announce that TortoiseSVN 1.7.3 has been released. It is linked against Subversion 1.7.2

Due to some nasty bugs in TortoiseSVN 1.7.2 which in some specific situations could make it crash, we’re releasing this version out of sync with SVN releases.

TortoisePlink was also updated to Plink 0.62 which had a security issue fixed.

We recommend that you update to this version as soon as possible.

• BUG: Issue #149: Crash in OpenSSL. (Stefan)
• BUG: Issue #150: Repobrowser: Don’t show externals in checkout selection mode. (Stefan)
• BUG: Issue #152: log crashes with too many bugtrack links. (Stefan)
• BUG: Issue #155: crash when accessing a server with an invalid ssl certificate. (Stefan)
• BUG: Issue #156: subdir within WC not recognized as WC. (Stefan)
• BUG: Issue #157: Title Bar not updating with currently viewed repository. (Stefan)
• BUG: Issue #158: IBugTraqProvider2 CheckCommit returns a relative commonURL. (Stefan)
• BUG: Issue #159: folder browse dialog can be closed with invalid paths. (Stefan)
• BUG: Issue #160: Repo browser shows incorrect contents for directories that differ only by case. (Stefan)
• NEW: Issue #113: Option to turn off pre-fetching and external parsing in repo browser. (Stefan)
• BUG: Issue #161: repo browser shows invalid commands for the root. (Stefan)
• BUG: Line wrapping problems in Log Messages dialog. (Stefan)
• BUG: Issue #166: TortoiseMerge slow with long lines. (Stefan)

Tortoisesvn 1.7.2 See Also:
http://tortoisesvn.net/Changelog.txt

• BUG: Issue #111: Crash in initializing the revision graph. (Stefan)
• CHG: Issue #112: Unescaped urls in switch dialog. (Stefan)
• BUG: Issue #115: non-existing folder shown in repo browser. (Stefan)
• BUG: Issue #110 (again): Bogus entries in Language Combo on x64. (Stefan)
• BUG: Issue #116: Language Packs don’t have the dialog resources translated. (Stefan)
• BUG: Issue #117: Property name has only one char. (Stefan)
• BUG: Issue #118: Impossible to remove properties from repo browser. (Stefan)
• BUG: Certificate Popups when smartcards are used. (Stefan)
• BUG: Issue #120: Copying part of URL copies entire URL in GUI. (Stefan)
• BUG: Issue #121: Authentication prompt when editing svn:externals. (Stefan)
• BUG: Issue #122: Column widths in repobrowser. (Stefan)
• BUG: Issue @123: Crash when switching from single to double pane view. (Stefan)
• BUG: Issue #124: painting artifact in commit dialog. (Stefan)
• BUG: Issue #125: missing conflict summary. (Stefan)
• BUG: Issue #126: crashes in repo browser. (Stefan)
• BUG: Issue #127: missing bug id in log dialog. (Stefan)
• BUG: Issue #130: out-of-date update misses dragged items. (Stefan)
• BUG: Issue #132: Cleanup fails to find externals in subfolders. (Stefan)
• BUG: Issue #133: crash in log dialog when “show only affected paths” is active. (Stefan)
• CHG: Issue #134: Show cached overlays for blocked paths. (Stefan)
• BUG: Issue #136: SubWCRev can use wrong casing for relative paths. (Stefan)
• BUG: Issue #108: log dialog uses wrong file if sorted. (Stefan)
• BUG: Issue #137: overlays don’t propagate from externals to their parent folders. (Stefan)
• BUG: Issue #141: Inline diff shown for ignored whitespace lines. (Stefan)
• BUG: Issue #142: Crash in TMerge on doubleclick. (Stefan)
• BUG: Issue #143: Stealing locks from CSVNStatusListCtrl doesn’t work. (Stefan)
• NEW: Issue #145: Fill in image in open dialog. (Stefan)
• BUG: Issue #146: Repo browser won’t show a directory whose parent directory cannot be read. (Stefan)

Tortoisesvn tools See Also:
• http://tools.tortoisesvn.net/

commitmonitor See Also:
• http://commitmonitor.googlecode.com
• http://tools.tortoisesvn.net/CommitMonitor.html

Figure 2.55: Commit monitor
The CommitMonitor is a small tool to monitor Apache™ Subversion® repositories for new commits.

It has a very small memory footprint and resides in the system tray.

If you have Snarl installed, CommitMonitor automatically uses Snarl to show its popups.

CommitMonitor is available under the GP licence (GPL).

You can either download an msi installer or a zipped exe file, whatever suits you: Download page.

Of course, the source code is available too.

You can browse or check it out directly from the Subversion repository.

Git, mercurial, DVCS

patch management

Patchutils is a small collection of programs that operate on patch files. Interdiff generates an incremental patch from two patches against a common source. For example, if you have applied a pre-patch to a source tree, and wish to apply another pre-patch (which is against the same original source tree), you can use interdiff to generate the patch that you need to apply. You can also use this to review changes between two pre-patches.

Combinediff generates a single patch from two incremental patches, allowing you to merge patches together. The resulting patch file only alters each file once.

Filterdiff will select the portions of a patch file that apply to files matching (or, alternatively, not matching) a shell wildcard.

Fixcvsdiff is for correcting the output of ‘cvs diff’.

Rediff corrects hand-edited patches, by comparing the original patch with the modified one and adjusting the offsets and counts.

Lsdiff displays a short listing of affected files in a patch file, along with (optionally) the line numbers of the start of each patch.

Splitdiff separates out patches from a patch file so that each new patch file only alters any given file once. In this way, a file containing several incremental patches can be split into individual incremental patches.

Recountdiff fixes up counts and offsets in a unified diff.

Unwrapdiff fixes word-wrapped unified diffs.

```
/usr/bin/install -c -m 644 './doc/interdiff.1' '/opt/patchutils/0.3.1/share/man/man1/interdiff.1'
/usr/bin/install -c -m 644 './doc/filterdiff.1' '/opt/patchutils/0.3.1/share/man/man1/filterdiff.1'
/usr/bin/install -c -m 644 './doc/fixcvsdiff.1' '/opt/patchutils/0.3.1/share/man/man1/fixcvsdiff.1'
/usr/bin/install -c -m 644 './doc/rediff.1' '/opt/patchutils/0.3.1/share/man/man1/rediff.1'
/usr/bin/install -c -m 644 './doc/editdiff.1' '/opt/patchutils/0.3.1/share/man/man1/editdiff.1'
/usr/bin/install -c -m 644 './doc/combinediff.1' '/opt/patchutils/0.3.1/share/man/man1/combinediff.1'
/usr/bin/install -c -m 644 './doc/lsdiff.1' '/opt/patchutils/0.3.1/share/man/man1/lsdiff.1'
/usr/bin/install -c -m 644 './doc/splitdiff.1' '/opt/patchutils/0.3.1/share/man/man1/splitdiff.1'
/usr/bin/install -c -m 644 './doc/grepdiff.1' '/opt/patchutils/0.3.1/share/man/man1/grepdiff.1'
/usr/bin/install -c -m 644 './doc/recountdiff.1' '/opt/patchutils/0.3.1/share/man/man1/recountdiff.1'
/usr/bin/install -c -m 644 './doc/unwrapdiff.1' '/opt/patchutils/0.3.1/share/man/man1/unwrapdiff.1'
/usr/bin/install -c -m 644 './doc/dehtmldiff.1' '/opt/patchutils/0.3.1/share/man/man1/dehtmldiff.1'
/usr/bin/install -c -m 644 './doc/flipdiff.1' '/opt/patchutils/0.3.1/share/man/man1/flipdiff.1'
/usr/bin/install -c -m 644 './doc/espdiff.1' '/opt/patchutils/0.3.1/share/man/man1/espdiff.1'
```

[root@houx bin]# 11
total 348
Where is it? Released tarballs (as well as pre-test releases) are here. For discussion about patchutils, try the patchutils mailing list. To subscribe to this list, which is very low traffic, send a message to: <patchutils-list-subscribe@sources.redhat.com>.

The git repository is available to view at fedorahosted.org:

```bash
git clone git://git.fedorahosted.org/git/patchutils.git
```

See Also:

- [http://cyberelk.net/tim/software/patchutils](http://cyberelk.net/tim/software/patchutils)
- [http://savannah.nongnu.org/projects/quilt](http://savannah.nongnu.org/projects/quilt)
- [http://mercurial.selenic.com/wiki/MqExtension#Using_Mercurial_Queues](http://mercurial.selenic.com/wiki/MqExtension#Using_Mercurial_Queues)

**PATH update**

```bash
export PATH=/opt/patchutils/current/bin:$PATH
```

Remarks about repositories

de Sébastien Douche <sdouche@gmail.com>
heure de l’expéditeur Envoyé à 14:07 (GMT+02:00). Heure locale : 15:13.
répondre à gitfr@googlegroups.com
à gitfr@googlegroups.com
cc Pierre8r <pierre8r-gg@yahoo.fr>
date 30 mars 2011 14:07
objet Re: Est-il conseiller de créer un repository par projet, ou un repository pour l’ensemble des projets ?

> Est-il conseiller de créer un repository par projet ou un repository pour l’ensemble des projets ?

Faire un repos pour tout est pour moi une hérésie et doit être banni.

Même avec SVN, la convention est de faire les répertoires trunk, tags et branches à la racine, mais par fainéantise (et parce que SVN permet le checkout partiel), tout est dans un seul dépôt.
Avoir plusieurs projets dans un seul dépôt possède plusieurs inconvénients majeurs :

- le dépôt risque de devenir obèse On ne peut pas cloner un bout de repos, c’est tout ou rien. Les commandes vont ralentir vu la taille du working dir.
- un historique incohérent Des commits sans relation entre eux, des multiples tags (comment vous allez tagger le projet X en 0.2 et le projet Y en 1.3 ?)[1].
- une profusion de branches Qui va gérer la compréhension
- des développeurs qui commettent tous au même endroit Vive le bordel.

Et vu les lacunes de Hg dans la gestion des branches par rapport à Git, c’est encore plus important de le faire comme cela.

Le seul défaut des dépôts multiples est la gestion qui doit suivre : administration (création, acl, backup) et obligation d’utiliser un outil “projet”.

Comment résoudre ce dernier point :

- git submodule peut être suffisant
- utiliser un outil comme repo, développer par des Javaistes pour le projet Android.
- utiliser Maven, Grails...

En Python, j’utilise Buildout (outil projet comme Maven) et Buildbot (outil ci comme Jenkins) et quelques scripts maison pour manipuler les répos.

Mais ça dépend aussi de ce que vous appelez “projet” :).

[1] obligé d’utiliser les namespaces

### 2.30 Web development

#### 2.30.1 Web development

**Blogs**

**mynt**

See Also:

- http://mynt.mirroredwhite.com/quickstart/
- http://mynt.mirroredwhite.com/

**Introduction**

Another static site generator ?

With the ever growing population of static site generators, all filling a certain need, I’ve yet to find one that allows the generation of anything but the simplest of blogs.
That’s where mynt comes in, being designed to give you all the features of a CMS with none of the often rigid implementations of those features.

**Nikola**

**See Also:**

### Contents
- Nikola
  - Why Static Websites ?
  - What Can Nikola Do ?
  - Seealso

**Why Static Websites ?** Static websites are safer, use fewer resources, and avoid vendor and platform lockin. You can read more about this in the Nikola Handbook.

**What Can Nikola Do ?** It has many features, but here are some of the nicer ones:
- Blogs, with tags, feeds, archives, comments, etc.
- Themable
- Fast builds, thanks to doit
- Flexible
- Small codebase (programmers can understand all of Nikola in a couple of hours)
- reStructuredText as input language
- Easy image galleries (just drop files in a folder!)
- Syntax highlighting for almost any programming language or markup
- Multilingual sites
- Doesn’t reinvent wheels, leverages existing tools.

**Seealso** **See Also:**

*Blogs*

**Pelican**

**See Also:**
- http://blog.ziade.org/2012/03/05/moving-to-pelican/
Introduction  Pelican is a simple weblog generator, written in Python.

- Write your weblog entries directly with your editor of choice (vim!) in reStructuredText or Markdown
- A simple CLI tool to (re)generate the weblog
- Easy to interface with DVCSes and web hooks
- Completely static output is easy to host anywhere

Pelican currently supports:

- Blog articles and pages
- Comments, via an external service (Disqus). (Please note that while useful, Disqus is an external service, and thus the comment data will be somewhat outside of your control and potentially subject to data loss.)
- Theming support (themes are created using jinja2)
- PDF generation of the articles/pages (optional)
- Publication of articles in multiple languages
- Atom/RSS feeds
- Code syntax highlighting
- Import from WordPress, Dotclear, or RSS feeds
- Integration with external tools: Twitter, Google Analytics, etc. (optional)

Seealso  See Also:

sphinx tinkerer application

Haridas  See Also:

http://haridas.in/wordpress-blog-migrated-to-pelican.html

Initially this blog was a wordpress blog. As you know the reasons for that. Wordpress is the most successful and feature rich blogging engine. So I chose it without any doubt. I didn’t even search for a python alternative blog engine at that time. But after a while I felt the difficulties while adding some changes to the wordpress code, bugs with the wordpress plugins and more importantly the Web UI is very annoying while typing new contents.

If we are considering the technical points, wordpress is a dynamic blog engine, it is an over killing for blogs and other wiki based sites. Main issue is with speed of page rendering. The static pages are way faster compared to the dynamic contents. I know wordpress has caching plugins to make the static pages out of dynamic one but that are not a very reliable solution.

Actually above points are all come to my mind after seeing the static site generators and their awesome features and flexibility.
Advantages of static site generators over Dynamic blog engines

- Serve html directly, so very fast.
- Easy maintenance of the site, very less pain with server setup.
- Use your favourite Text editors for blog posting. I use VIM :).
- It use Markdown and Restructured Text Syntaxes for blog entry. So we just need to type the post in normal text with simple formating. So don’t worry about the html formatting while typing the content.
- Host it on Github, and very easy version control and site backup.
- Very easy to customize the Themes or other internals if you want.

The next step was which static site generator I will choose for my site. There are a lot of them were implemented in Python and Ruby. I went through some of them and finally chose Pelican a Python based static site generator.

Cloud computing

See Also:


Cloud computing, informatique en nuage ou infonuagique est un concept qui consiste à déporter sur des serveurs distants des traitements informatiques traditionnellement localisés sur des serveurs locaux ou sur le poste client de l’utilisateur.

Bien que l’anglicisme Cloud computing soit largement utilisé en français, on rencontre également les francisations informatique virtuelle, informatique dans le nuage, informatique en nuage, informatique dématérialisée, ou encore infonuagique

Ce concept est vu comme une évolution majeure par certains analystes, ou comme un piège marketing par d’autres, notamment par Richard Stallman.

En France, la commission générale de terminologie et de néologie précise que l’informatique en nuage est une forme particulière de gérance de l’informatique, puisque l’emplacement des données dans le nuage n’est pas porté à la connaissance des clients.

Les utilisateurs ou les entreprises ne sont plus gérants de leurs serveurs informatiques mais peuvent ainsi accéder de manière évolution à de nombreux services en ligne sans avoir à gérer l’infrastructure sous-jacente, souvent complexe.

Les applications et les données ne se trouvent plus sur l’ordinateur local, mais – métaphoriquement parlant – dans un nuage (cloud) composé d’un certain nombre de serveurs distants interconnectés au moyen d’une excellente bande passante indispensable à la fluidité du système. L’accès au service se fait par une application standard facilement disponible, la plupart du temps un navigateur Web.

Cloud application platform

Heroku cloud application platform  See Also:


Agile deployment for Ruby, Node.js, Clojure, Java, Python and Scala.

Get up and running in minutes, and deploy instantly with git. Focus 100% on your code, and never think about servers, instances, or VMs again.
**Instant Deployment**  Deploying an app is simple and easy. No special tools needed, just a plain git push. Deployment is instant, whether your app is big or small.

Read more about git deployment.

**Continuous Deployment**  Easily create testing, staging, and production versions of your app and deploy to and between them instantly. The **dyno manifold** insures all parts of your app are updated and bounced gracefully, **routing** continues seamlessly, and traffic is held for data migrations by custom maintenance controls.

**Poká-yoke**  Move quickly and with confidence: Heroku follows a poká-yoke (mistake-proof) design philosophy, including role-based permissions, integrity checks during push, and robust release management and rollback controls.

If in doubt, watch every detail of the deployment process in real-time with **Logplex**.

**Bruce Heckel example**  See Also:

http://www.artima.com/weblogs/viewpost.jsp?thread=335549

Heroku is a cloud application platform. They seem to have the goal of making cloud deployment of your app as easy as possible, and they also seem to be trying to adapt to whatever technology you happen to be using.

Just today (September 30, 2011) they announced support for Python. You can find a walkthrough here and there’s specific information about using it with Django here.

Because I’ve been preparing for this speaking trip to Europe, I have yet to delve into the details of Heroku. My friend James Ward (who now works there) asked to see the code for this article, and upon receiving it had it set up in less than 30 minutes (even though he is a Python novice and hasn’t used Web.py).

Here are the steps he used (with the basic Heroku stuff already set up):

Create a new file in your src dir named “Procfile” containing:

```
web: python server.py ${PORT}
```

Create a new file in your src dir named “requirements.txt” containing:

```
web.py
```

Create the git repo, add the files to it, and commit them:

```
git init
git add .
git commit -m init
```

Create the app on Heroku:

```
heroku create -s cedar
```

Deploy the app:

```
git push heroku master
```

It’s kind of clever the way they use git as the deployment mechanism. I have to say I’m pretty impressed by how quickly this could be deployed on Heroku. I think it’s a tribute to the simplicity of Web.py that there weren’t any snags.

Here is the app running under Heroku.
Picloud cloud application platform  See Also:
https://www.picloud.com/

Re-inventing the Cloud  Leverage the power of the cloud with only 3 lines of code. Leave the load balancing, auto scaling, and server management to us.

Imagination at Work  Our customers use us to crawl the web, process images and videos, calculate analytics, run simulations, and much more.

The Sky is the Limit  Open network connections, use custom native libraries, and run your computation for as long as you need.

Pay as You Use  No minimum fees. No server costs. We charge you for the exact time your functions take to run down to the millisecond.

Python cloud module  See Also:
• http://pypi.python.org/pypi/cloud
• http://docs.picloud.com/quickstart.html

PiCloud is a cloud-computing platform that integrates into the Python Programming Language. It enables you to leverage the computing power of Amazon Web Services without having to manage, maintain, or configure virtual servers.

When using this Python library known as cloud, PiCloud will integrate seamlessly into your existing code base. To offload the execution of a function to our servers, all you must do is pass your desired function into the cloud library. PiCloud will run the function on its high-performance cluster. As you run more functions, our cluster auto-scales to meet your computational needs.

Before using this package, you will need to sign up a PiCloud account.

The cloud library also features a simulator, which can be used without a PiCloud account. The simulator uses the multiprocessing library to create a stripped down version of the PiCloud service. This simulated service can then run jobs locally across all CPU cores.

Collaborative edition

Contents
• Collaborative edition
  – Framapad
  – LibreOffice and telepathy
  – Stypi

Framapad

See Also:
http://framapad.org
LibreOffice and telepathy

See Also:
- http://people.gnome.org/~michael/blog/2012-03-26-collaboration.html
- https://wiki.documentfoundation.org/Development/Gsoc/Ideas#Telepathy_for_collaboration
- http://telepathy.freedesktop.org/wiki/

One of the last, big missing features in LibreOffice is collaborative editing. To fill this hole Eike Rathke (of RedHat) has been working for some weeks getting Telepathy code hooked into the LibreOffice core. This was chosen to allow us to setup a channel for multi-way communication over existing Instant Messaging (IM) protocols, without requiring any form of server.

Stypi

See Also:
- http://blog.stypi.com

Django Web framework

Django IDE, Djangode

See Also:
http://code.google.com/p/djangode/
A QT4-based Python IDE aimed at making Django development easier.

Django packages See Also:
http://www.djangopackages.com/

Django blog packages See Also:
http://www.djangopackages.com/grids/g/blogs/

Django biologeek See Also:
- https://bitbucket.org/david/biologeek/

Quelques informations pour vous aider, même si je ne pense pas qu’il soit utile d’installer ce site qui est très spécifique mais plutôt de consulter les parties de code qui peuvent vous aider dans la compréhension de certaines astuces/bonnes pratiques Python/Django.

Licence Voir COPYING ou http://sam.zoy.org/wtfpl/ (merci Sam :-)).
**Installation**  Dépendances :
- django-tagging inclus à la version 0.2.1 http://code.google.com/p/django-tagging/
- pyRdfa à installer à la version 1.1 http://www.w3.org/2007/08/pyRdfa/
- markdown2 inclus à la version 1.0.1.4 http://code.google.com/p/python-markdown2/

**Django zinnia**  See Also:
- https://github.com/Fantomas42/django-blog-zinnia

Simple yet powerful application for managing a blog within your Django website.
Zinnia has been made for publishing weblog entries and designed to do it well.
Basically any feature that can be provided by another reusable app has been left out. Why should we re-implement something that is already done and reviewed by others and tested ?

**Features**  More than a long speech, here the list of the main features :
- Comments
- Sitemaps
- Archives views
- Related entries
- Private entries
- RSS or Atom Feeds
- Tags and categories views
- Advanced search engine
- Prepublication and expiration
- Widgets (Popular entries, Similar entries, ...)
- Spam protection with Akismet
- Admin dashboard
- MetaWeblog API
- Ping Directories
- Ping External links
- Bit.ly support
- Twitter support
- Gravatar support
- Django- CMS plugins
- Collaborative work
- Tags autocompletion
- Entry model extendable
Development tools, Release 2012.06.18

- Pingback/Trackback support
- WYMeditor or TinyMCE support
- WordPress conversion utility
- Ready to use and extendables templates
- Windows Live Writer compatibility

Examples  See Also:

http://django-blog-zinnia.com/planet/weblogs/

Take a look at the online demo at: http://django-blog-zinnia.com or you can visit these websites who use Zinnia.
- Fantomas’ side / Mobile version.
- Professional Web Studio
- mixedCase.
- MadCad’s Page.

If you are a proud user of Zinnia, send me the URL of your website and I will add it to the list.

Django cms packages  See Also:

http://www.djangopackages.com/grids/g/cms/

Django cms  See Also:

- https://github.com/divio/django-cms
- https://groups.google.com/forum/#!forum/django-cms

A Django application for managing hierarchical pages of content, possibly in multiple languages and/or on multiple sites.

Django CMS handles the navigation rendering for you with clean, slug based URLs, and this navigation can be extended by custom Django applications

You can define editable areas, called placeholders, in your templates which you can fill with CMS plugins, by default we ship the following plugins:

- File
- Flash
- Google Map
- Link
- Picture
- HTML Snippet
- Teaser
- Text
- Video
- Twitter
However this is not the end of the list, there are many 3rd party plugins already available for you to use. Should you be unable to find a suitable plugin for you needs, writing your own is very simple.

More information on our website.

**Django CMS promotion  See Also:**

- [http://www.youtube.com/watch?v=i6Fr65PFqfk](http://www.youtube.com/watch?v=i6Fr65PFqfk)
- [https://groups.google.com/forum/#!topic/django-cms/6ZvITfSfEXA](https://groups.google.com/forum/#!topic/django-cms/6ZvITfSfEXA)

Hi,

> I have a potentially huge job offer coming up. Some healthcare organization in Germany needs a complete revamp of their current web-portal which is [...]

> *Questions:*

> - What are good portals to research developer prices? Maybe even special to the German market?
> - What other arguments come to your mind? Did I forget anything?

Apart from the other answer pulling in the big names just a few thoughts:

* Django has an excellent security track record. I am not aware of any SQL injection.
* Django is built around concepts and patterns but still light weight and no over engineered crap.
* Django is very modular. (ORM, Template eng., ...)
* The Django devs listen to critics. Just mention that very funny djangocon key note by Carl Henderson [http://www.youtube.com/watch?v=i6Fr65PFqfk](http://www.youtube.com/watch?v=i6Fr65PFqfk)
  
  And show how many of his complains got addressed. There was another Why I hate Django talk by Eric Florenzano that changed a few things as well.
* Django-CMS is a healthy FOSS project. Show the commit history. Show how many folks are participating.
* Django-CMS doesn’t get in your way too much. I actually don’t know many other CMSs. However, I had a go on this Typo3 config language... Further, it’s so simple to create your own plugins / apps in Django-CMS. Not sure how to add that to your document you will probably present to some folks that have never written a single line of code.
  
  Same applies to Django of course.
* Maybe don’t show them the Backend of let’s say Typo3 compared to Django-CMS ;) More stretch how easy it is to customise (You mentioned that already.)

Well, there will be much more arguments ;) and much better ones.

IMHO Python, Django and Django-CMS are great examples of FOSS projects.
Just because software is free doesn’t mean anything IMHO. I can’t help you w/ the developer cost. If you optimise for the lowest price you will end up w/ crap and probably pay double (for cleaning up the mess the cheap one made).
I got no clue about django-shop, just know LFS a little.

my 2 cent,

Django mezzanine  See Also:

• http://mezzanine.jupo.org/docs/overview.html
• http://pypi.python.org/pypi/Mezzanine

An open source content management platform built using the Django framework.
Mezzanine is a content management platform built using the Django framework. It is BSD licensed and designed to provide both a consistent interface for managing content, and a simple architecture that makes diving in and hacking on the code as easy as possible.
Its goal is to resemble something like Wordpress, with an intuitive interface for managing pages and blog posts.
Mezzanine takes a different approach from other Django applications in this space like Pinax or Mingus that glue together a lot of reusable apps, instead opting to provide most of its functionality included with the project by default.

Quotes

• “I am enjoying working with Mezzanine - it is good work.” - Van Lindberg
• “Impressed with Mezzanine so far.” - Brad Montgomery
• “You need to check out the open source project Mezzanine. In one word: Elegant.” - Nick Hagianis
• “Mezzanine looks pretty interesting - tempting to get me off Wordpress.” - Jesse Noller
• “Who came up with the name Mezzanine? I love it, like a platform between the client’s ideas and their published website. Very classy!” - Stephen White

Django merengue
See Also:

• http://www.merengueproject.org/

AMerengue is a full-featured CMS useful for creating websites without writing a single line of code or customizing and extending with any of the plethora of existing Django applications.

Django commenting packages
**Django ucomment**  See Also:

- [http://ucomment.org/contents/](http://ucomment.org/contents/)
- copypasta and sphinx

ucomment is a Django application that allows for web-based commenting on a document that has been written in reStructuredText.

ucomment relies heavily on Sphinx to typeset the document. A distributed version control system is used to track document revisions and comments made by web users.

And it requires Django as the glue to hold it all together.

This entire site uses ucomment, so please visit the pages below to see it in action.

You are welcome to contribute your comments, questions and feedback, by clicking on the blue sidebar at left on any of these links below

**Django translations packages**

**Django pylglot package**  See Also:

[https://github.com/omaciel/pylyglot](https://github.com/omaciel/pylyglot)

Pylyglot

Hey, are you feverishly working on the translation for the upcoming GNOME 3.0 release? Do you find yourself often wondering how a certain word was translated across the entire desktop applications?

Well, I do and this is why I developed Pylyglot, a Django-driven localization tool that let’s you search for a word and see how it has been translated for a given language.

The plan is to have fresh data straight from the GNOME git repositories every Monday for now, eventually switching to a more frequent update schedule.

Right now I’m getting a 500 error when you click on the list of packages for a certain word, but I hope I can fix it over this weekend.

There is an issue filed for it, and if you have the time and interest in helping me fix it or suggestions (and patches) to improve the overall experience, please fork the project and come chat with me on #pylyglot on freenode:

[https://github.com/omaciel/pylyglot](https://github.com/omaciel/pylyglot)

Happy translations!

**Django data packages**

**Django CSV importer**  See Also:

- Django csv

**Java FX**

See Also:

Introduction

JavaFX is a software platform for creating and delivering rich Internet applications (RIAs) that can run across a wide variety of connected devices.

The current release (JavaFX 2.1, April 2012) enables building applications for desktop, browser and mobile phones. TV set-top boxes, gaming consoles, Blu-ray players and other platforms are planned.

Java FX versions

Java FX 2.1.0  On April 27, 2012, Oracle released version 2.1 of JavaFX[9], which includes the following main features:

- H.264 and AAC support
- First official version for Mac OS X (desktop only)
- LCD text
- UI enhancements including combo box controls, charts (stacked chart), and menu bars
- Webview component now allows JavaScript to make calls to Java methods

JDK and JRE versions from 1.7.0_04 and onwards include JavaFX bundled to be installed with them.

When Java 8 is available, JavaFX will become part of the JRE/JDK

Firefox Web development

See Also:


hg clone http://hg.mozilla.org/mozilla-central/

Firefox addons

Addon fox  See Also:

- http://www.addonfox.com/help/

AddonFox is an automatic installer that let’s you choose which addons you want from the categories that interest you and then automatically installs all of the addons at once.

Over one hundred hand picked addons carefully chosen with improved descriptions.

Easily boost your Firefox with the best Firefox addons.

anonymox extension  See Also:

Collusion  See Also:
• https://addons.mozilla.org/en-US/firefox/addon/collusion/

Contents
  • Collusion
    – Introduction
    – Take control of your data
    – Telling the global tracking story
    – Building user awareness
    – Collusion is about choice

Introduction  Collusion is an experimental add-on for Firefox and allows you to see all the third parties that are tracking your movements across the Web.

It will show, in real time, how that data creates a spider-web of interaction between companies and other trackers.

Take control of your data  We recognize the importance of transparency and our mission is all about empowering users — both with tools and with information.

The Ford Foundation is supporting Mozilla to develop the Collusion add-on so it will enable users to not only see who is tracking them across the Web, but also to turn that tracking off when they want to.

Telling the global tracking story  Your data can be part of the larger story. When we launch the full version of Collusion, it will allow you to opt-in to sharing your anonymous data in a global database of web tracker data.

We’ll combine all that information and make it available to help researchers, journalists, and others analyze and explain how data is tracked on the web.

Building user awareness  Through our work with the Ford Foundation, we’ll be building outreach campaigns to help people understand online data tracking — both the benefits and the issues — so they can make their own choices about how they want to be tracked (or choose not to be tracked at all).

Collusion is about choice  Not all tracking is bad. Many services rely on user data to provide relevant content and enhance your online experience.

But most tracking happens without users’ consent and without their knowledge. That’s not okay. It should be you who decides when, how and if you want to be tracked.

Collusion will be a powerful tool to help you do that

YouTube downloader  See Also:
• http://www.bestvideodownloader.com/installcomplete/basic?cid=c33b0859-9f45-4ebf-94ad-ffbc25c6e543&qs=-1&sid=1243697

Fastest, easiest & most reliable free YouTube downloader. Highest sound quality available with M4A, MP3, MP4, AAC, FLV and HD formats!

Single click, non-intrusive, direct download button works directly within the YouTube page!
Feedly extension  See Also:

- http://aldarone.fr/bye-bye-google-reader-bonjour-feedly/

L’extension ajoute également un petit bouton à moitié transparent nommé Feedly-mini en bas à droite de chaque site (il est évidemment possible de le désactiver dans les options.) Il permet de s’abonner rapidement aux flux du site consulté ou encore de partager la page sur twitter, facebook, e-mail, etc..

Firebug extension  See Also:

- http://getfirebug.com

What is firebug

- http://getfirebug.com/whatisfirebug

Firebug integrates with Firefox to put a wealth of web development tools at your fingertips while you browse. You can edit, debug, and monitor CSS, HTML, and JavaScript live in any web page.

Fireftp extension  See Also:

- http://fireftp.mozdev.org/features.html

Features

- It’s free!
- Cross-platform: Works on Windows, Mac OS X, Linux
- Secure: SSL/TLS/SFTP support, same encryption used with online banking and shopping
- Synchronization: Keep directories in sync while navigating
- Directory Comparison: Compare directory content (compares subdirectories too!)
- International: Available in over 20 languages
- Character Set Support: UTF8 and just about any other character encoding supported
- Automatic reconnect and resuming of transfers
- Search/Filtering
- Integrity Checks of transfers (XMD5, XSHA1)
- Export/Import accounts
- Remote Editing
- File Hashing: Generate hashes of files (MD5, various SHA’s)
- Drag & Drop
- File Compression: Using MODE Z
- Timestamp Synchronization
- Proxy support
- FXP support
- Advanced properties (CHMOD, recursive CHMOD, thumbnails)
- Tutorials and help files available for support
• IPv6 support
• Open Source!
• Seamless integration with Mozilla Firefox
• ...did we mention it’s free? :-)

HTTPS Everywhere extension  See Also:
• https://www.eff.org/https-everywhere

HTTPS Everywhere is a Firefox extension produced as a collaboration between The Tor Project and the Electronic Frontier Foundation. It encrypts your communications with a number of major websites. Many sites on the web offer some limited support for encryption over HTTPS, but make it difficult to use. For instance, they may default to unencrypted HTTP, or fill encrypted pages with links that go back to the unencrypted site. The HTTPS Everywhere extension fixes these problems by rewriting all requests to these sites to HTTPS. Firefox users can get it by clicking here:

ProfileSwitcher extension  See Also:
• http://www.planet-libre.org/index.php?post_id=11284&go=external
• https://nic-nac-project.org/~kaosmos/profileswitcher-en.html

Open attribute extension  See Also:
• https://addons.mozilla.org/en-US/firefox/addon/openattribute/
• Open attribute
• http://openattribute.com/first-run-firefox/

Displays license and attribution information for Creative Commons licensed content. Open Attribute works on pages with Creative Commons licenses. If the license code doesn’t include complete attribution information, Open Attribute might not be able to generate the full attribution for you.

Tranquility  See Also:
• https://addons.mozilla.org/fr/firefox/addon/tranquility-1/

Web developer toolbar  See Also:
• https://addons.mozilla.org/fr/firefox/addon/web-developer/

What is firebug  The Web Developer extension adds various web developer tools to a browser.

Firefox issue_tracker

See Also:
https://bugzilla.mozilla.org/
Firefox versions

See Also:
http://wiki.frenchmozilla.org/index.php/Mercurial

Firefox 10 (Aurora)  See Also:

Voici les autres apports de Firefox 10
- Le support du 3D-Transform pour les CSS3
- Support de la balise bdi pour l’isolation bidirectionnelle du texte
- Ajout du CSS Style Inspector
- Support de l’API IndexedDB

Google

See Also:
- http://ploum.net/post/google-moins

Policies

Google policies  See Also:
- http://www.google.com/policies
- http://www.google.com/policies/faq

Nouvelles règles de confidentialité et conditions d’utilisation de Google

See Also:
- http://www.google.com/policies
- http://www.google.com/policies/faq
Cher utilisateur de Google,

Nous effectuons actuellement une mise à jour des règles de confidentialité de Google.

Nous avons ainsi décidé de faire évoluer près de soixante règles de confidentialité distinctes, et de les remplacer par une nouvelle version unique, à la fois complète, concise et simple à lire.

Ces nouvelles règles s’appliquent désormais à de nombreux produits et fonctionnalités afin que nous puissions vous offrir une expérience utilisateur simple et intuitive à travers tous les produits Google.

Parce que ces choses-là sont importantes, nous vous invitons à consacrer quelques minutes de votre temps à la lecture de ces nouvelles règles de confidentialité et conditions d’utilisation, accessibles depuis ce lien : http://www.google.com/policies.

Ces modifications prendront effet le 1er mars 2012.

Ces nouvelles règles illustrent notre volonté de vous garantir une expérience homogène, grâce à des produits adaptés et simples à utiliser.

Qu’il s’agisse de lire un e-mail pour vous rappeler de planifier une réunion de famille ou de trouver une vidéo pour la partager avec d’autres personnes, nous voulons faciliter votre utilisation quotidienne de Gmail, de Google Agenda, de la recherche Google, de YouTube ou de tout autre produit dont vous pourriez avoir besoin.

**Une meilleure personnalisation** Si vous êtes connecté à Google, vous pouvez, si vous le souhaitez, obtenir des suggestions de termes de recherche ou la personnalisation de vos résultats de recherche en fonction des centres d’intérêt que vous avez indiqués dans Google+, Gmail et YouTube.

Ainsi, nous serons en mesure de mieux comprendre ce vous souhaitez trouver lorsque vous saisissez, par exemple, des termes comme “Téléphone” ou “Jaguar”, et vous présentons les résultats plus rapidement que jamais.

**Le partage et la collaboration en toute simplicité** Lorsque vous créez ou publiez un document en ligne, vous souhaitez peut-être que d’autres personnes le consultent, voire y apportent leur contribution.

En s’appuyant sur la liste des personnes avec lesquelles vous avez déjà partagé du contenu (liste que vous pouvez administrer à tout moment), nous vous permettons, en quelques clics, de partager d’autres informations via les autres produits ou services Google, et ce avec un maximum de simplicité et de fiabilité.

**La protection de votre vie privée reste inchangée** Notre objectif : vous garantir une transparence et une liberté de choix maximales, grâce notamment au Google Dashboard, au Gestionnaire de préférences pour les annonces et à une multitude d’autres outils.
Nos principes de confidentialité restent inchangés. Jamais nous ne vendrons ni ne partagerons vos données personnelles sans votre autorisation (sauf dans de rares cas, comme les demandes d’ordre juridique).

**Vous avez des questions ?** Nous avons les réponses


**Avis de modification** Les nouvelles règles de confidentialité et conditions d’utilisation prendront effet le 1er mars 2012.

Si vous continuez à utiliser les produits et services Google après l’entrée en vigueur de ces modifications, votre utilisation sera régie par les nouvelles règles de confidentialité et conditions d’utilisation.

Merci de ne pas répondre à ce message. Les mails envoyés à cette adresse e-mail ne pourront être traités. Ne saisissez jamais le mot de passe associé à votre compte Google après avoir cliqué, dans un e-mail ou dans un chat, sur un lien dirigeant vers un site inconnu. Allez toujours directement sur la page du site auquel le mot de passe est associé (mail.google.com ou www.google.com/accounts/, par exemple).

Google ne vous demandera jamais votre mot de passe par e-mail, ni aucune autre information sensible ou confidentielle.

**Javascript language**

**Javascript tips**

**See Also:**

**Books on javascript**

**Eloquent javascript** **See Also:**
http://bergie.iki.fi/blog/the_universal_runtime/
- http://eloquentjavascript.net/

**Pyramid Web framework**

**Pyramid Web news**


**Web python modules**

**pattern for python**

**See Also:**
- http://code.google.com/p/pattern-for-python/
- http://www.clips.ua.ac.be/pages/pattern
• http://www.clips.ua.ac.be/media/pattern-1.4.zip

Pattern is a web mining module for the Python programming language. It bundles tools for
• data retrieval (Google + Twitter + Wikipedia API, web spider, HTML DOM parser),
• text analysis (rule-based shallow parser, WordNet interface, syntactical + semantical n-gram search algorithm, tf-idf + cosine similarity + LSA metrics)
• and data visualization (graph networks).

Python Web hosting

See Also:

An awesome year for Python web hosting

This year is shaping up to be an awesome year for Python web hosting. Not only do we have existing reliable providers like WebFaction, but we now also have a gaggle of Python specific web hosting services coming online. So many that one can easily start loosing track of who they are. As such, doing this post to list those that I know of. Are there any more that people know of that are just starting up?

epio

See Also:
http://www.ep.io/

This is ep.io, the hosted WSGI hosting and deployment solution. We take your Django, Flask, Pylons or other WSGI code, run all the annoying hosting parts for you, and make sure everything’s redundant and backed up.

In return, we bill you only for the processing time and bandwidth you actually use - we’ll even give you detailed analytics of when and where those resources are being used.

However, we’re currently in a closed beta, so this is only available to those we invite.

If you’d like an invite, fill out the form below; we’ll send you an invitation when we’re ready for the next round of invites (usually a couple of weeks, March 2011).

In the meantime, feel free to have a look through our documentation and our plans for pricing.

gondo

See Also:
http://gondor.io/

Gondor is the production infrastructure that we at Eldarion have built for our own sites and those of our clients. We’re making it available to other Django developers like you so you can focus on your site, not your infrastructure. It supports:
Development tools, Release 2012.06.18

- revision control via git or mercurial
- dependency management using pip
- database migrations via South or nashvegas
- full backups of your entire application
- asynchronous and scheduled task execution
- full-text search using Solr and django-haystack
- caching via memcached or redis

apphosted.com

See Also:
https://apphosted.com/

AppHosted provides the infrastructure to deploy and host your Python WSGI compliant applications, including auto-configuration for Django, Flask and Pylons, without the hassle of configuring and managing servers.

Instantly scale your applications as needed for a predictable, low monthly cost.

Always data

See Also:
http://www.alwaysdata.com/

Django zoom

See Also:
- http://djangozoom.com/
- http://djangozoom.com/blog/

Effortless deployment

Deploying a release of your web application should not be a time-consuming process.

DjangoZoom takes care of the tedious server setup, streamlining the deployment process, so that you can get back to writing code.

Deploy your app in 5 minutes or less!

Heroku devcenter

See Also:
- http://devcenter.heroku.com/articles/python
- http://devcenter.heroku.com/articles/django

Getting Started with Python on Heroku/Cedar.
Web people

Tim Berners Lee

See Also:

http://www.w3.org/DesignIssues/old/Architecture.html

Web services (soap, json/rpc

Python ladon framework for Web Services

See Also:

- http://pypi.python.org/pypi/ladon/
- http://packages.python.org/ladon/

Serve your python methods to several types of internet services at once.

Ladon is a framework for exposing python methods to several internet services protocols.

Once a method is ladonized it is automatically served through all the interfaces that your ladon installation contains.

Ladon is easily extendable.

Adding a new service interface is as easy as adding a single module containing a class inheriting the BaseInterface class.

Social networks

See Also:

http://fr.wikipedia.org/wiki/R%C3%A9seaux_sociaux

Un réseau social est un ensemble d’identités sociales telles que des individus ou encore des organisations reliées entre elles par des liens créés lors des interactions sociales. Il se représente par une structure ou une forme dynamique d’un groupement social. L’analyse des réseaux sociaux, basée sur la théorie des réseaux, l’usage des graphes et l’analyse sociologique représente le domaine étudiant les réseaux sociaux. Des réseaux sociaux peuvent être créés stratégiquement pour agrandir ou rendre plus efficient son propre réseau social (professionnel, amical)\(^1\).

Un réseau social représente une structure sociale dynamique se modélisant par des sommets et des arêtes. Les sommets désignent généralement des gens et/ou des organisations et sont reliées entre elles par des interactions sociales.


bitbucket

See Also:

- https://bitbucket.org/

pvergain bitbucket See Also:

https://bitbucket.org/pvergain
github

See Also:
  - https://github.org/

pvergain github  See Also:
https://github.org/pvergain

google plus

See Also:
https://plus.google.com/
Google+ (pronounced and sometimes written as Google Plus, sometimes abbreviated as G+) is a social networking and identity service, operated by Google Inc.

pvergain google_plus  See Also:
https://plus.google.com/u/0/109823873605578864987

identi.ca

See Also:
  - https://identi.ca/
Identi.ca est une solution libre de réseautage social et de microblogage. Elle est basée sur StatusNet (anciennement Laconica), un logiciel de microblogage basé sur les spécifications OpenMicroBlogging.
Les utilisateurs peuvent ajouter des textes d’une longueur maximum de 140 caractères sur le principe du microblog, reprenant la longueur exacte imposée sur Twitter. La limitation à la même longueur se justifie d'ailleurs par le fait qu’un message émis sur Identi.ca peut l’être simultanément sur Twitter.
Le système gère le protocole XMPP et permet l’exportation et l’échange de données basé sur le standard FOAF.

pvergain identica  See Also:
https://identi.ca/pvergain

linuxfr

See Also:
  - https://linuxfr.org/
seenthis

See Also:

- http://seenthis.net/fran%C3%A7ais/article/c-est-quoi-seenthis

Seenthis permet de tenir à jour un blog personnel constitué de billets courts. Il est principalement destiné à la veille d’actualité. Pour cela, il propose de mettre en valeur le référencement de pages Web, la citation d’extraits et le commentaire, grâce à une mise en forme automatique et adaptée des textes.

Il est destiné, principalement, à faciliter la recommandation de liens entre pairs.

Il est associé à un système de forums publics permettant aux participants d’échanger des idées de manière constructive (conversation publique).

Un système de thématisation avancé facilite la constitution de bases documentaires et thématiques.

pvergain seenthis  See Also:

http://seenthis.net/people/pvergain

twitter

See Also:

- https://twitter.com/

Twitter est un outil de réseau social et de microblogage qui permet à l’utilisateur d’envoyer gratuitement des messages brefs, appelés tweets (« gazouillis »), par Internet, par messagerie instantanée ou par SMS.

pvergain twitter  See Also:

https://twitter.com/#!/pvergain

wordpress

See Also:

- https://wordpress.org/
- https://twitter.com/

WordPress est un système de gestion de contenu libre écrit en PHP et reposant sur une base de données MySQL.

WordPress est surtout utilisé comme moteur de blog, mais ses fonctionnalités lui permettent également de gérer n’importe quel site web.

Il est distribué selon les termes de la GNU GPL. Le logiciel est aussi à l’origine du service WordPress.com.

pvergain wordpress  See Also:

https://twitter.com/#!/pvergain
Web testing

See Also:


Web testing with selenium

See Also:

- http://seleniumhq.org/
- http://seleniumhq.org/docs/01_introducing_selenium.html

Many, perhaps most, software applications today are written as web-based applications to be run in an Internet browser. The effectiveness of testing these applications varies widely among companies and organizations. In an era of highly interactive and responsive software processes where many organizations are using some form of Agile methodology, test automation is frequently becoming a requirement for software projects. Test automation is often the answer. Test automation means using a software tool to run repeatable tests against the application to be tested. For regression testing this provides that responsiveness.

Web time

Mementoweb

See Also:

- http://mementoweb.org/

Web tutorials

See Also:

- css tutorials

school-of-webcraft

See Also:


Mozilla

Learn

See Also:

- https://developer.mozilla.org/fr/learn
Demos  See Also:

- https://developer.mozilla.org/fr/demos/

Web of things

See Also:

- http://www.webofthings.com/
- http://epcmashup.webofthings.com/

L’internet des objets

La promesse de l’internet des objets est de construire un réseau fluide d’appareils hétérogènes connectés ensemble pour former un dispositif unique et cohérent. Mais en fait, cette promesse est une escroquerie”, attaque, bille en tête, Vlad Trifa, ingénieur à l’Institut d’informatique pervasive de l’Institut de technologie de Zurich sur la scène de Lift à Genève.
Il existe déjà plus d’une quinzaine de protocoles techniques pour assurer les communications domotiques et machines à machines (M2M), mais ceux-ci demeurent largement inconnus des programmeurs qui ne sont pas spécialisés sur ces sujets.

Si l’industrie a construit des normes pour contrôler l’internet des objets, elle est loin d’être parvenue à un accord. “La réalité aujourd’hui est que nous avons plutôt construit des intranets pour un grand nombre de choses qu’un internet des objets, chacun formant un îlot isolé de quelques appareils connectés qui n’ont pratiquement aucun moyen d’interagir les uns avec les autres”.

L’internet des objets est donc une utopie.

**Senseable Lab du MIT**

L’internet des objets qui passe par l’internet n’a rien à voir avec la domotique traditionnelle, explique Vlad Trifa. Des interfaces de programmation permettent à des services ou des outils de se brancher sur les données que ces objets produisent et ce en temps réel. Mais ce pas seulement valable pour nos objets du quotidien : c’est également de villes entières dont on parle, comme le montre le projet LiveSingapore développé par le Senseable Lab du MIT.

Le projet consiste à construire une plateforme pour recueillir des centaines de flux de données provenant de divers organismes pour produire des informations de haut niveau sur la ville et son fonctionnement.

**Smart Planet initiative**

A Rio de Janeiro, IBM, avec leur Smart Planet initiative, a construit, en partenariat avec la ville, un centre d’opération qui fonctionne indépendamment de tout organisme tout en recevant des données de plusieurs d’entre eux (vidéo). Ce centre d’opération fonctionne grâce à une batterie d’algorithmes pour contrôler, prévoir et visualiser les informations vitales en temps réel de la ville et déterminer la meilleure façon d’y répondre : quelles sont les collines les plus sujettes à des coulées de boue ? Quels hôpitaux ont des lits disponibles ? Où sont les voitures de police d’urgence et les ambulances ?...

Autre exemple. On sait que l’eau potable est un bien de plus en plus précieux et que pour nombre de grandes villes, la distribution de l’eau est un problème majeur. L’infrastructure d’adduction d’eau est souvent vieillissante et les défaillances fréquentes. La gestion du système est souvent inefficace et 30 % de l’eau est la plupart du temps perdu dans l’acheminement.

**Projet waterwise**

A Singapour, le projet WaterWise a consisté à placer des capteurs sur les conduites d’eau pour surveiller en temps réel la pression, la température, et analyser la composition chimique et biologique de l’eau... ce qui permet de repérer les fuites, de réagir plus vite à un incident. Et surtout de mieux étalonner la consommation selon la demande.

**EPCIS (Electronic Product Code Information System)**

Mais le web des objets peut avoir également des applications encore plus concrètes. Dans une chaine d’approvisionnement, on doit savoir tout le temps et en temps réel où sont les produits ou les matériaux dont on a besoin. Pour créer une logistique en temps réel, il existe des solutions techniques coûteuses, comme le réseau EPCIS (Electronic Product Code Information System), un ensemble d’outils et de normes pour le suivi et le partage de produits dotés d’étiquettes RFID. Des systèmes de ce type permettent de suivre et localiser des produits tout le long d’une chaîne d’approvisionnement.

Les systèmes d’informations EPC utilisent des systèmes complexes et peu malléables.
EPC mashups est une plateforme qui permet d'accéder à toutes les informations produites par sa chaîne logistique via le web d'une manière plus souple et plus fluide que le système d'information d'EPC et de construire des outils et des applications pour rendre la chaîne d'approvisionnement toujours plus intelligente et malléable.


About

What is WebofThings.com?

It is a web page founded by Dominique Guinard and Vlad Trifa, two researchers/geeks working at ETH Zurich and SAP Research Zurich. It is more a scrap book that is here to dump our thoughts ideas on our research and work topic, which is the Web of Things.

Unlike most Web 2.0 sites that are about advanced powerpoint engineering, we talk about real things that’s under the hood.
3.1 Hardware

3.1.1 Hardware

Computers

See Also:


Un ordinateur est une machine électronique qui fonctionne par la lecture séquentielle d’un ensemble d’instructions qui lui font exécuter des opérations logiques et arithmétiques sur des chiffres binaires.

PC Computers

See Also:


Un ordinateur personnel, encore appelé micro-ordinateur ou ordinateur individuel, est un ordinateur destiné à l’usage d’une personne et dont les dimensions sont assez réduites pour tenir sur un bureau. La première machine appelée micro-ordinateur est le Micral N, inventé en 1973 par le français François Gernelle.

Le PC a 30 ans : évolution technique See Also:


Tablet computers

See Also:


Tablet computers news
**Encourager le logiciel libre sur les tablettes**  
See Also:

- [http://aful.org/communiques/jeu-concours-afultab](http://aful.org/communiques/jeu-concours-afultab)

Les tablettes tactiles devenant de plus en plus populaires, les inquiétudes face à la fermeture de leurs systèmes d’exploitation et de leurs applications sont de plus en plus grandes. L’Association Francophone des Utilisateurs de Logiciels Libres lance donc un concours visant à promouvoir le libre sur les tablettes.

Le succès d’Android le démontre chaque jour : les systèmes ouverts sont très prisés par les possesseurs de smartphones ou de tablettes tactiles.

Canonical, par la voix de son PDG Mark Shuttleworth, a ainsi expliqué que l’interface Unity, qui remplace Gnome dans la dernière version d’Ubuntu, a été conçue pour être utilisable sur tous types d’appareils.

Mais ce qui fait le succès des smartphones ou des tablettes, ce sont aussi leurs boutiques d’applications. Canonical en est d’ailleurs conscient puisque Shuttleworth a appelé les développeurs à s’engager pour sa plateforme lors de l’Ubuntu Developer Summit et à développer des applications open-source.

Une initiative lancée par l’Association Francophone des Utilisateurs de Logiciels Libres (AFUL) va également dans ce sens. L’AFUL lance un jeu concours intitulé AFULTab dont l’objectif est de faire fonctionner les tablettes disponibles sur le marché uniquement avec des logiciels libres.

**Electronic**

**Normes Electroniques**

**Normes européennes**

**DEEE (Déchets Equipements Electriques et Electroniques) Directive 2002/96/CE**  
Applicable depuis 2005

Obligations concernent les Fabricants (dont le nom est apposé sur le produit) ou les importateurs de produits dans l’UE.

L’obligation ne concerne donc id3 que pour les produits vendus sous sa propre marque

- Nous avons obligation de générer une fiche de fin de vie à l’attention des clients et un dossier de démantèlement à l’attention des recycleurs permettant d’identifier des produits spécifiques tels que : piles, batteries, composants contenant du mercure, condensateurs contenant du PCB, LCDs, condensateurs électrolythiques h>25mm et d>25mm , etc ...

- Nous avons obligation de nous inscrire sur le registre des producteurs (auprès de l’ADEME) - c’est fait d’après SC.

- Nous avons obligation de prévoir la filière de recyclage auprès d’un organisme agréé (Eco-systèmes, Ecologic, ou ErP pour les produits électroniques) et de les pré-financer en fonction des ventes de nos produits

- Nous avons depuis 2005 l’obligation légale de faire apparaître sur nos factures l’Ecotaxe permettant de financer le recyclage ci-dessus (Cette obligation devrait être supprimée) et d’apposer la poubelle barrée sur nos produits

- Nous avons obligation légale d’évaluer le taux de recyclabilité des produits: le taux minimum imposé par la DEEE actuelle est de 75% de valorisation et 65% de recyclage (le recyclage s’entend sans la partie énergétique de la valorisation [l’énergie récupérée par un incinérateur]; soit uniquement l’aspect réutilisation des matériaux). Ce taux s’entend en masse du produit.
RoHS 2 (Reduction Of Hazardous Substances) - directive 2002/95/CE (RoHS)  Applicable depuis 2006 > directive 2011/65/UE

Sur 10 catégories décrites dans la RoHS de 2006, seule 8 étaient concernées (exemptions : cat 8 : dispositifs médicaux et cat 9 : instruments de mesure et de contrôle).


Les secteurs des transports (automobile, ferroviaire) et l’aéronautique et le spatial restent hors champ.

ATTENTION, le CE intégrera le marquage RoHS, il devra donc y avoir un dossier technique et une déclaration de conformité.

Reach : (Registration Evaluation Authorisation and Restriction of Chemicals) - Règlement 1907/2006.  See Also:


En vigueur depuis le 1/6/2007.

Reach n’est pas une directive mais un règlement, donc appliqué de manière strictement identique dans toute l’Europe.

Vocabulaire  On considère

- une substance comme étant un composé chimique simple; (Ex : Acétone, plomb, ...)
- une préparation comme étant un mélange de substances (colle, SAC305, ciment, ...)
- et un article comme étant un objet dont la composition chimique est secondaire par rapport à sa forme ou sa fonction mécanique ou électrique (pièce plastique, carte électronique ...)

LE BUT  Eliminer à terme les substances dangereuses pour l’homme ou l’environnement.

LA METHODE  L’objectif est d’évaluer les substances présentes dans une préparation ou un article et de comprendre leur impact sur santé et l’environnement en 3 étapes dites “R E A”

R pour REGISTRATION  La première phase était donc l’identification des substances avec un seuil de consommation minimum de 1 tonne par an… ce qui est loin de nous concerner. (depuis le 1/6/2008)

Les substances surveillées dites SVHC (substances of Very High Concern) sont dans un premier temps mises dans une liste de candidature.
Leur utilisation est alors soumise à notification (déclaration) si leur concentration est supérieure à 0.1% en masse/masse totale de l’article et si la quantité totale de cette substance est supérieure à 1T/an par déclarant.

Il y a 53 substances candidates à autorisation à ce jour qu’elles soient SVHC pour les humains (produits cancérigènes, mutagènes, toxiques pour la reproduction) ; SVHC pour l’environnement (persistantes, bio-accumulables, toxiques) ou perturbateurs endocriniens (qui agissent sur l’équilibre hormonal d’espèces vivantes animales ou végétales). Il est prévu d’évaluer prochainement 800 substances et de passer progressivement à 30000 !

E pour EVALUATION  L’agence ECHA : European CHemical Agency analyse les dossiers substance par substance (Evaluation des propositions d’essais des déclatants; évaluation de la conformité des dossiers, évaluations des substances) sur proposition des autorités compétentes des états membres.

A pour AUTHORISATION / RESTRICTION  Les conclusions de ces évaluations amènent l’agence à transférer les substances de cette liste de candidature

- soit à l’annexe XIV du règlement (substances soumises à Restriction) Mesures de contrôle ou interdiction d’usage (concerne actuellement 6 substances). Les substances qui ne seront pas autorisées ne seront plus disponibles sur le marché.

Si les risques sont maîtrisés; s’il est démontré que les avantages socio-économiques l’emportent sur les risques pour la santé et l’environnement, et s’il n’y a pas de solution de substitution possible,

- soit à une autorisation de commercialisation sans contrôle.

Jusque là, cela paraît relativement “simple”; mais là où cela va se compliquer c’est que le règlement prévoit pour tout fournisseur d’articles de pouvoir informer tout destinataire d’article qui en ferait la demande, sans seuil de tonnage, sur l’éventuelle présence de substances “candidates” (avec au minimim, le nom des substances concernées et les informations permettant une utilisation du produit en toute sécurité).

Pour cela il faudrait donc tracer les articles consommés; obtenir des certificats permettant de statuer sur la présence ou l’absence de substances candidates ou interdites (un article pouvant avoir plusieurs centaines de postes nomenclature ... cela fait un volume de données énorme.

De plus, les informations sont obligatoirement à notifier si la substance est depuis plus de 6 mois dans la liste candidate; il faut donc surveiller cette liste susceptible d’évoluer au moins 2x par an et gérer l’éventuelle obsolescence de lots contenant des produits devenus interdits ...

Il faut aussi conserver ces informations 10 ans.

Sur la manière de gérer cela raisonnablement, AUCUNE approche n’est actuellement proposée par le LCIE qui faisait la présentation.

Tout le domaine de l’électronique s’attend donc à naviguer dans un brouillard législatif avec le risque d’amendes prévues par le texte pour non respect de ces obligations (amendes de 3e cat).

Liens : ECHA : http://echa.europe.eu

Hardware PCB

See Also:
http://www.cadsoftusa.com/training/

Processors

See Also:
ARM Architecture

See Also:


Les architectures ARM, développées par ARM Ltd, sont des architectures RISC 32 bits.

Dotés d’une architecture relativement plus simple que d’autres familles de processeurs, et bénéficiant d’une faible consommation, les processeurs ARM sont devenus dominants dans le domaine de l’informatique embarquée, en particulier la téléphonie mobile et les tablettes.

Ces processeurs sont fabriqués sous licence par un grand nombre de constructeurs

ARM is one of the most popular architectures used in embedded Linux systems.

ARM designs CPU cores (instruction sets, caches, MMU, etc.) and sells the design to licensees

The licensees are founders (Texas Instruments, Freescale, ST Ericsson, Atmel, etc.), they integrate an ARM core with many peripherals, into a chip called a SoC, for System-on-chip

Each founder provides different models of SoC, with different combination of peripherals, power, power consumption, etc.

The concept of SoC allows to reduce the number of peripherals needed on the board, and therefore the cost of designing and building the board.

Linux supports SoCs from most vendors, but not all, and not with the same level of functionality.

Open source Hardware

See Also:


Arduino

See Also:

- http://arduino.cc/
- http://code.google.com/p/arduino/

### Contents

- Arduino
  - Introduction
  - Tools

**Introduction**  Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software.

It’s intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators.
The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing).

Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Tools  See Also:
http://arduino.cc/it/Serial/Print

Fritzing

See Also:

• http://fritzing.org/
• https://twitter.com/#!/FritzingOrg

About Fritzing  See Also:
http://hackaday.com/2012/05/10 going-from-idea-to-schematic-to-printed-pcb/

Fritzing is an open-source initiative to support designers, artists, researchers and hobbyists to work creatively with interactive electronics.

We are creating a software and website in the spirit of Processing and Arduino, developing a tool that allows users to document their prototypes, share them with others, teach electronics in a classroom, and to create a pcb layout for professional manufacturing.

Sensors

Sensors interfaces

Sensors interfaces  See Also:

• http://www.phidgets.com
• http://www.phidgets.com/programming_resources.php
The PhidgetInterfaceKit 2/2/2 allows you to connect devices to any of:

- 2 analog inputs
- 2 digital inputs
- 2 digital outputs

It provides a generic, convenient way to interface your PC with various devices. The USB dongle form factor dramatically reduces the space required to add interfacing capability to your project.
Analog inputs  Analog Inputs are used to measure continuous quantities, such as temperature, humidity, position, pressure, etc.

Phidgets offers a wide variety of sensors that can be plugged directly into the board using the cable included with the sensor.

Here is a list of sensors currently available

- IR Distance Sensor
- IR Reflective Sensor
- Vibration Sensor
- Light Sensor
- Force Sensor
- Humidity Sensor
- Temperature Sensor
- Magnetic Sensor
- Rotation Sensor
- Voltage Divider
- Touch Sensor
- Motion Sensor
- Mini Joy-Stick Pressure Sensor
- Voltage Sensor
- Current Sensor
- Slide Sensor
- Sound Sensor

Each analog input can be adjusted to sample at a data rate ranging from 1 sample to up to 1000 samples per second. Samples are transmitted to your PC every 8ms.

This feature is very useful for setting up the resolution of your data logging. Note that data rate is limited to 16ms when opening over the Phidget Webservice

Architecture  We have designed our libraries to give you the maximum amount of freedom. We do not impose our own programming model on you.

To achieve this goal we have implemented the libraries as a series of layers with the C API at the core surrounded by other language wrappers.
Libraries  The lowest level library is the C API. The C API can be programmed against on Windows, CE, OS X and Linux. With the C API, C/C++, you can write cross-platform code. For systems with minimal resources (small computers), the C API may be the only choice.

The Java API is built into the C API Library. Java, by default is cross-platform - but your particular platform may not support it (CE).

The .NET API also relies on the C API. Our default .NET API is for .NET 2.0 Framework, but we also have .NET libraries for .NET 1.1 and .NET Compact Framework (CE).

The COM API relies on the C API. The COM API is programmed against when coding in VB6, VBScript, Excel (VBA), Delphi and Labview.

The ActionScript 3.0 Library relies on a communication link with a PhidgetWebService (see below). ActionScript 3.0 is used in Flex and Flash 9.

Telemeters

Ultrasonic telemeter MAXSonar (MB7077)  See Also:

- http://fr.wikipedia.org/wiki/T%C3%A9l%C3%A8m%C3%A8tre
- http://fr.wikipedia.org/wiki/%C3%89cholocation

Télécharger la datasheet du XL-MaxSonar.

Component Overview

- 42kHz Ultrasonic sensor measures distance to objects
- RoHS Compliant
- Read from all 3 select sensor outputs: Analog Voltage, Serial, Pulse Width
- Possible dead zone 0-4 cm, objects closer than 20 cm range as 20 cm
- Resolution of 1 cm
- Operates from 3.0-5.5V
- Low 3.4mA average current requirement
- 10Hz reading rate
- Small, light weight module
- Designed for easy integration into your project or product
- Operational Temperature from -40C to +70C (-40F to +160F)
- Real-time automatic calibration (voltage, humidity, ambient noise)
- Firmware filtering for better noise tolerance and clutter rejection
- 200,000+ Hours Mean Time Between Failure
- Weather Resistant (IP67), optional Chemical Resistant F-Option
- Matches standard electrical 3/4-inch PVC pipe fittings for easy mounting (3/4” National Pipe Thread Straight)
• Compact version of our most popular outdoor rangefinder
• Maximum reported range of 765 cm (300 inches)
• Maximum range to most targets of 645cm (254 inches)
• Long range, narrow detection zone

The MB7067 and MB7077 sensors provide very short to long-range detection and ranging, in an extra compact, robust PVC housing, designed to meet **IP67 water intrusion**, and matches standard electrical ¾ PCV pipe fittings.

This sensor has a new high power output along with real-time auto calibration for changing conditions (temperature, voltage or acoustic or electrical noise) that ensure you receive the most reliable (in air) ranging data for every reading taken.

The low power 3.0V to 5.5V operation detects objects from **3-cm to 654-cm** (21.2 feet) and provides sonar range information from 20-cm out to 645-cm.

Objects from 3-cm to 20-cm typically range as 20-cm. The interface output formats are: real-time analog voltage envelope (MB7077), pulse width output (MB7067), real-time analog voltage envelope, analog voltage output, and serial digital output.

**Conversion distance / valeur affichée**  
This pin outputs analog voltage with a scaling factor of (Vcc/1024) per cm. A supply of **5000 mV** yields ~**4.9mV/cm.**, and 3.3V yields ~3.2mV/cm.

\[ f(d) = ((d \times 4.9)/5000) \times 1024 \]

Hardware limits the maximum reported range on this output to ~**700 cm at 5V** and ~600 cm at 3.3V.

The output is buffered and corresponds to the most recent range data.

**Python example**  
[Télécharger le script python calc_distance.py](#).

```python
FACTEUR = (4.9 * 1024) / 5000.0
FACTEUR_INV = 5000.0 / (4.9 * 1024)

def calc_puissance(distance):
    """This pin outputs analog voltage with a scaling factor of (Vcc/\*1024\*) per cm. A supply of \*5000 mV\* yields \*~4.9mV/cm.\*, and 3.3V yields \*~3.2mV/cm.\*"
    puissance = (distance * FACTEUR)
```

1138 Chapter 3. Hardware
return puissance

def calc_distance(puissance):
    """Calcul de la distance en fonction de la puissance.
    d = p * FACTEUR_INV
    """
    distance = (puissance * FACTEUR_INV)
    return puissance

def output_table():
    for distance in range(20,100,1):
        print('distance=', distance, 'puissance=', calc_puissance(distance))
    for puissance in range(20,100,1):
        print('puissance=', puissance, 'distance=', calc_distance(puissance))

if __name__ == '__main__':
    output_table()

Output

('distance=', 20, 'puissance=', 20.0704)
('distance=', 21, 'puissance=', 21.07392)
('distance=', 22, 'puissance=', 22.07744)
('distance=', 23, 'puissance=', 23.080959999999997)
('distance=', 24, 'puissance=', 24.08448)
('distance=', 25, 'puissance=', 25.088)
('distance=', 26, 'puissance=', 26.09152)
('distance=', 27, 'puissance=', 27.095039999999997)
('distance=', 28, 'puissance=', 28.09856)
('distance=', 29, 'puissance=', 29.10208)
('distance=', 30, 'puissance=', 30.1056)
('distance=', 31, 'puissance=', 31.109199999999997)
('distance=', 32, 'puissance=', 32.11264)
('distance=', 33, 'puissance=', 33.11616)
('distance=', 34, 'puissance=', 34.11968)
('distance=', 35, 'puissance=', 35.1232)
('distance=', 36, 'puissance=', 36.12672)
('distance=', 37, 'puissance=', 37.13024)
('distance=', 38, 'puissance=', 38.133759999999997)
('distance=', 39, 'puissance=', 39.13728)
('distance=', 40, 'puissance=', 40.1408)
('distance=', 41, 'puissance=', 41.14432)
('distance=', 42, 'puissance=', 42.14784)
('distance=', 43, 'puissance=', 43.15136)
('distance=', 44, 'puissance=', 44.15488)
('distance=', 45, 'puissance=', 45.1584)
('distance=', 46, 'puissance=', 46.161919999999997)
('distance=', 47, 'puissance=', 47.16544)
('distance=', 48, 'puissance=', 48.16896)
('distance=', 49, 'puissance=', 49.17248)
('distance=', 50, 'puissance=', 50.176)
('distance=', 51, 'puissance=', 51.17952)
('distance=', 52, 'puissance=', 52.18304)
('distance=', 53, 'puissance=', 53.18656)
('distance=', 54, 'puissance=', 54.190079999999997)
3.1. Hardware
('puissance=', 91, 'distance=', 91)
('puissance=', 92, 'distance=', 92)
('puissance=', 93, 'distance=', 93)
('puissance=', 94, 'distance=', 94)
('puissance=', 95, 'distance=', 95)
('puissance=', 96, 'distance=', 96)
('puissance=', 97, 'distance=', 97)
('puissance=', 98, 'distance=', 98)
('puissance=', 99, 'distance=', 99)
4.1 Languages

4.1.1 Languages

Language Français

See Also:

- https://fr.wikipedia.org/wiki/Fran%C3%A7ais

Le français est une langue romane parlée comme langue maternelle principalement en France (y compris en France d’outre-mer), dont elle est origininaire (la « langue d’oïl »), en Belgique (en Wallonie et à dans la région de Bruxelles-Capitale), à Monaco, dans plusieurs provinces et territoires du Canada (principalement au Québec, mais aussi en Ontario, Manitoba et au Nouveau-Brunswick), au Luxembourg (où il est l’une des trois langues officielles du pays) et en Suisse romande (le français est l’une des quatre langues officielles de la Suisse).

Il existe aussi des îlots de francophones natifs en Côte d’Ivoire (Abidjan), au Gabon (Libreville), aux États-Unis (notamment en Louisiane et au Maine), en Haïti, aux Seychelles, à l’île Maurice, au Vanuatu, dans certaines vallées italiennes (principalement dans le Val d’Aoste), dans de nombreux pays d’Afrique du Nord, d’Afrique de l’Ouest, et d’Afrique centrale, à Pondichéry (Inde), où le français est une langue très présente.

Emprunts plus récents, néologismes et évolution de l’orthographe

Article détaillé : Emprunt lexical.

Les emprunts plus récents à d’autres langues sont assez nombreux : d’abord à l’anglais (même anciens : nord, sud), puis à l’italien, aux autres langues romanes, aux langues germaniques tels que l’allemand ou le néerlandais (ainsi boulevard vient du hollandais ou du flamand bolwerk). L’arabe a fourni, et fournit encore quelques mots : alcool, algèbre, toubib, bled, etc.

Persian Language

See Also:
  • https://fr.wikipedia.org/wiki/Persan
  • https://en.wikipedia.org/wiki/Persian_language
  • https://fa.wikipedia.org/wiki/%D8%B2%D8%A8%D8%A7%D9%86_%D9%81%D8%A7%D8%B1%D8%B3%DB%8C

Le persan (), est une langue parlée en Iran (plus de 70 millions de locuteurs), en Afghanistan (16 millions), et au Tadjikistan (7 millions) en tant que langue officielle, ainsi qu’au Bahreïn (minorité iranophone) et en Ouzbékistan (minorité tadjike). Il compte au total plus de 75 millions de locuteurs et appartient au groupe indo-iranien de la famille des langues indo-européennes. C’est une langue du type « sujet-objet-verbe ».

De nos jours, les langues persanes s’écrivent surtout au moyen de l’alphabet perso-arabe, variante de l’alphabet arabe, bien qu’elles n’aient aucune parenté avec la langue arabe, dont elles diffèrent tant sur le plan de la grammaire que de la phonologie.

Au Tadjikistan et en Ouzbékistan où elle a été cyrillisée, la langue tadjike s’écrit en caractères cyrilliques.

Tutorial

See Also:
  • http://www.persiandirect.com/projects/ipa2/ipa2_tutor.htm
  • http://farsi.irantoulouse.com/

Dictionnaire

See Also:
http://www.freelang.com/dictionnaire/iranien.php

Dictionnaire FREELANG : Iranien

L’iranien est une variété de persan parlée en Iran par plus de 70 millions de locuteurs. Les autres variétés de persan sont parlées en Afghanistan (le dari) et au Tadjikistan (le tadjik, qui s’écrit en caractères cyrilliques).

Le persan est parfois appelé “farsi”, une arabisation du terme “parsi”.

C’est une langue appartenant au groupe indo-iranien des langues indo-européennes. Le persan s’écrit avec l’alphabet arabe (auquel quatre lettres ont été ajoutées pour noter quatre sons n’existant pas en arabe).

Il existe également un système de romanisation.

Hebrew Language

See Also:
  • https://en.wikipedia.org/wiki/Hebrew_language
  • https://fr.wikipedia.org/wiki/H%C3%A9breu

L’hébreu (אידיש) est une langue appartenant à la branche centre-nord de la famille des langues sémitiques. Elle est étroitement apparentée à l’arabe et aux langues araméennes. Elle compte plus de 8 millions de locuteurs en Israël et en diaspora.

L’hébreu est l’une des deux langues officielles de l’État d’Israël, avec l’arabe.
Formation des mots

Article détaillé : Grammaire hébraïque.

En hébreu tout mot peut s’analyser en deux morphèmes : le schème et la racine.

Les schèmes nominaux ou verbaux constituent des squelettes dans lesquels sont coulées les racines. Ils sont en nombre limité et associés à des sens ou des usages spécifiques.

La racine de chaque mot se dégage naturellement pour le locuteur hébraïsant qui distingue l’ajout d’une consonne préfixale ou suffixale. Une racine est généralement trilitère mais l’hébreu connaît aussi des racines quadrilitères voire quinquilitères.

C’est ainsi que l’on peut produire un adjectif, une conjugaison, une forme passive, un indicatif, etc. à partir de n’importe quelle racine, même si le mot est d’origine étrangère ou [lazim] ” comme l’écrit RAC”I : prenons le mot « téléphone » () se prononçant bien entendu « téléphone », le verbe « téléphoner » suivant la grammaire hébraïque, se dit « letalepène » () (certains prononcent « letalefène » par assimilation).

Les membres de la famille Kimhi qui ont vécu vers le milieu du Moyen Âge ont passé de nombreuses années à dénombrer et comprendre les schèmes () et ont posé les bases de la première grammaire hébraïque.

Yiddish Language

See Also:
  • https://en.wikipedia.org/wiki/Yiddish_language
  • https://fr.wikipedia.org/wiki/Yiddish

Le yiddish ( (API: [jdl] ou [jdd]) en yiddish ; également orthographié en français yidish, jiddisch, jidisch ainsi que, selon les rectifications orthographiques du français, yidich, idiche ou yidiche) est une langue germanique dérivée du haut allemand, avec un apport de vocabulaire hébreu et slave, qui a servi de langue vernaculaire aux communautés juives d’Europe centrale et orientale (ashkénazes) du Moyen Âge. Il est également parfois appelé judéo-allemand (yidish-daytsh (yi)-)

Le yiddish a été parlé par les deux tiers des Juifs du monde, soit onze millions de personnes à la veille de la Seconde Guerre mondiale. Depuis le génocide des Juifs par les nazis, au cours duquel la majorité de la population juive d’Europe a été exterminée, il est en voie de disparition1.

Les linguistes divisent l’histoire du yiddish en quatre grandes périodes :
  • le pré-yiddish, jusqu’en 1250 ;
  • le yiddish ancien, de 1250 à 1500 ;
  • le moyen yiddish, de 1500 à 1750 et,
  • enfin, le yiddish moderne, de 1750 à nos jours.

Une langue originale

Une langue métissée

Le yiddish s’écrit en alphabet hébreu, même si ce n’est pas une langue consonantique (on y rajoute des voyelles) comme l’hébreu. Sa grammaire repose sur des bases de la grammaire allemande et son vocabulaire se compose d’éléments germaniques (80%), sémitiques (10% à 15%) et slaves (environ 5%).

Un aspect intéressant du lexique est la création de mots formés d’emprunts aux multiples composantes de la langue, que ce soit l’allemand, les langues slaves ou l’hébreu. L’expression oysgemutshet un oygemartert montre à elle seule la richesse de la langue. Oys et ge sont des préfixes germaniques ; oys signifiant, “tout le temps, complètement” et
ge étant la marque du participe passé. Mutshet est d’origine slave, matert, d’origine germanique. La répétition de deux termes de sens très proches n’est pas rare en yiddish. Il est le résultat d’une influence biblique. Le yiddish ayant des accents et formes dialectales, il peut exister de notables différences entre locuteurs selon les zones linguistiques dont ils sont originaires. Le yiddish existe aussi en écriture latine (translittération) pour ses locuteurs ne lisant pas l’alphabet hébreu. Dans un article écrit en mai 1909 dans un journal de langue Yiddish de Vilno, Lebn un Visnchaft (Vie et sciences) ; Ludwik Lejzer Zamenhof, initiateur de l’espéranto défendait l’usage de l’alphabet latin pour le yiddish.

Le yiddish est une langue qui a toujours intrigué les linguistes et les philologues. Ils se demandent comment définir une langue proche de la famille des parlers indo-européens, mais comprenant en même temps un fort pourcentage d’hébraïsmes. Compte tenu de la fragmentation dialectale, il serait d’ailleurs plus juste de parler de yiddishs, au pluriel, plutôt que d’une seule et unique langue. Il existe en effet, une multitude de dialectes. Le yiddish occidental est parlé en Alsace, Suisse, Allemagne et les Pays-Bas. Le yiddish oriental est utilisé dans l’aire géographique de l’Europe de l’Est où l’on distingue trois types de locuteurs: les polakn (Polonais), les litvakes (Lituaniens) et les galitsiyaner (Galiciens). Les détracteurs du yiddish ont affirmé qu’il ne s’agissait pas d’une langue en tant que telle mais bien de différents patois germaniques. Cependant les linguistes affirment qu’ils s’agit bien d’une langue en tant que telle.

Le yiddish fut dans les années 1920 (possiblement de 1926 à 1937) une des langues officielles de la République socialiste soviétique biélorusse. Il fut également la langue officielle de l’ Oblast autonome juif en U.R.S.S.

Une langue imagée

Même si les concordances avec l’allemand sont nombreuses, les différences sont importantes. Elles sont liées au caractère particulier de la pensée juive. Le yiddish comprend des mots hébreux qui n’existent pas dans les langues non-juives comme mikvé (bain rituel). Il regorge aussi d’expressions savoureuses: hak mir nisht kayn tshaynik qui signifie littéralement “ne me cogne pas une théière” est une expression qui peut vouloir dire : arrête de jacasser pour ne rien dire ; elle emploie l’image d’une bouilloire dont le couvercle se soulève et crépite sans arrêt. Les références au monde non-juif sont aussi très présentes. Quelqu’un qui a été oublié ou ignoré va dire: “Ikh hob zikh geshmat?” ce qui signifie: “Est-ce que je me suis converti au christianisme?”. C’est l’équivalent du français: “Et moi, je sens le pâté ?” Aynredn a kind in boykh, littéralement mettre enceinte par la force de la parole, signifie en fait embobiner, convaincre quelqu’un de quelque chose d’absurde car tous les juifs le savent: fun zogn men nisht trogn, parler ne peut pas mettre enceinte, allusion à peine voilée à la conception de Jésus par l’opération du Saint-Esprit.

Les pratiques religieuses ont elles aussi donné naissance à de nombreuses expressions imagées. Le shlogn kapores, (kapparot en hébreu) est une cérémonie traditionnelle aujourd’hui tombée en désuétude sous sa forme originelle sauf chez les Hassidim. Elle consiste à faire tourner un poulet vivant au-dessus de sa tête, la veille de Yom Kippour en récitant une prière. Le poulet se charge alors des fautes de celui qui prie. Le choix du poulet peut s’expliquer ainsi. En hébreu, hevreh, coq se dit gever, ce qui peut aussi vouloir dire homme. “Œil pour œil”, “gever pour gever”. Le jeu de mot a induit la pratique rituelle. Le mot kapores, intraduisible en français a été récupéré pour de nombreuses expressions. Zayn di kapore far signifie être amoureux de, aimer quelqu’un au point d’être prêt à se sacrifier pour lui comme un poulet lors du shlogn kapores. Shlogn kapores mit signifie rabaisser, abuser d’une personne, darfn af kapores, littéralement “en avoir besoin pour les kapores” veut dire n’en avoir aucun usage. Les Juifs n’aimant désigner directement les choses horribles ou tristes utilisent volontiers l’antiphrase. Cette tendance qui prépare le terrain à l’ironie et à l’humour juif se retrouve dans les expressions yiddish. Ainsi pour parler d’un cimetière, un yiddish dit “dos gute ort”, le bon endroit ou “beys khayim”, la maison de vie. Il est parfois impossible de parler par antiphrase. dans ce cas, on rajoute: nisht far aykht gedakht, que cela vous soit épargné ou rakhmone litslan, que Dieu nous en préserve !. Le yiddish compte un nombre très important de malédictions toutes plus imaginées les unes que les autres.

Oblast autonome juif

See Also:

• https://fr.wikipedia.org/wiki/Oblast_autonome_juif
L’oblast autonome juif (en russe : Еврейская автономная область ; en yiddish : יידישע אўטונומרען א十月) est un sujet de la Fédération de Russie situé en Sibérie, sur la frontière chinoise. Il est souvent nommé Birobidjan, selon sa capitale.

Cet oblast a été fondé à l’initiative de Staline en 1934, avec le yiddish comme langue officielle (ce qu’elle n’est plus actuellement[Quand ?]). Il a conservé le statut de terre d’accueil pour les Juifs de Russie.

Ce projet politique se poursuit après la création d’Israël en 1948 : on comptait alors 30 000 Juifs dans l’oblast. Dès la mort de Staline le 5 mars 1953, la population juive du territoire ne devait cesser de décroître, tant sous Khrouchtchev que sous Brejnev et en 1959, elle n’était plus que de 9 %, chutant même à 7 % en 1970.


English Language

See Also:


L’anglais est l’une des langues les plus parlées au monde : en nombre de locuteurs dont c’est la langue maternelle, les estimations varient de 3e, après le chinois (mandarin) et l’hindoustani, à 4e, après l’espagnol

Arabic Language

See Also:
- https://en.wikipedia.org/wiki/Arabic_language

La langue arabe ( العربية , al arabya) est originaire de la péninsule Arabique. L’expansion territoriale au Moyen Âge et la diffusion du Coran répandent la langue arabe, devenue langue liturgique de l’islam, en Asie (Moyen-Orient et Proche-Orient), en Afrique du Nord et en Europe (Chypre, Crète, Péninsule ibérique, Malte et Sicile).
Parlée d’abord par les Arabes, cette langue sémitique qui se déploie géographiquement sur plusieurs continents s’étend sociologiquement à des peuples non arabes, et devient aujourd’hui langue officielle de plusieurs organismes internationaux.

**Arabe classique**

Article détaillé : [Arabe classique](#).

L’arabe ancien est celui de la poésie préislamique.

L’arabe coranique est la langue du texte sacré des musulmans, le Coran, et les textes religieux.

L’arabe classique proprement dit est la langue de la civilisation arabo-musulmane.

**Arabe standard moderne**

Article détaillé : [Arabe standard moderne](#).


C’est la langue écrite commune de tous les pays arabophones ;

**Langues arabes vernaculaires**

Article détaillé : [Arabe dialectal](#).


L’arabe est tout de même généralisé au travers de l’arabe littéraire, enseigné à tous dans le système scolaire arabe.

Les dialectes les plus importants sont l’égyptien, le chami, le maghrébin, hedjazi...

Le chami est parlé en Syrie, au Liban, en Jordanie et Palestine, le Hassanya parlé en Mauritanie, au Maroc, au Sahara occidental et dans quelques zones de l’Afrique de l’Ouest.

Généralement, entre eux, les locuteurs de dialectes différents utilisent plutôt l’arabe littéral, ou une forme simplifiée de l’arabe littéral.

**Regroupements de dialectes**

Les langues arabes, regroupées en 4 groupes principaux sont difficilement intercompréhensibles à l’intérieur de ces groupes, on peut donc distinguer une quinzaine de langues très différentes (au moins autant que les langues latines) au sein desquelles les variantes dialectales sont suffisamment fortes pour être notées.

Les variantes arabes sont issues d’une matrice elle-même diverse, la Fassiha, forme sémitique hétérogène, langue des poètes et sa forme “lingua franca” des négociations inter-tribales.

Au Maghreb par exemple, l’arabisation a commencé par l’implantation de camps arabes en Espagne et en province d’Afrique (Tunisie et Algérie orientale), à l’origine des deux langues andalouses et ifricyennes, il s’est poursuivi par
arabisation par contamination commerciale et administrative sur la population “romaine” autochtone, tandis que la rurality “barbare” a gardé la langue amazighe, les communautés urbaines maures sont apparues avec cette constante influence andalouse et ifricyenne, notamment à Fès, Tétouan, Tlemcen (etc.) et les nécessités liturgiques arabes dans ces centres universitaires, puis de l’arabisation administrative, surtout à partir des mérinides (XIIIe siècle) En parallèle, depuis le XIe siècle, et surtout le XIIe siècle, des populations arabes bédouines (sinaïtes, libyennes, cyrèniennes et peut-être yéménites) ont peuplé le Maghreb central et oriental, ainsi que les espaces sahariens, influençant, chacun avec leur dialecte propre (lié à leur origine singulière et leurs développements autonomes propres...) les populations berbères les plus sensibles.


Ils ne sont pas du tout intercompréhensibles, mais une forme de maghrébin simplifié permet une intercompréhension entre les commerçants par exemple, mais souvent le français prend le pas dans la diplomatie et le grand commerce.

**Vecteurs de rayonnement de l’arabe**

Un premier vecteur de rayonnement est la religion islamique. L’arabe est resté une langue liturgique dans la plupart des pays musulmans, bien que l’arabe coranique soit aujourd’hui éloigné de la langue arabe moderne.

Un second vecteur de rayonnement est la littérature en prose et poétique. Des écrivains non arabes ont utilisé la langue arabe pour leurs publications, par exemple le médecin et philosophe perse Avicenne. Les rois normands de Sicile se piquaient de parler l’arabe.

Un troisième vecteur de rayonnement sont les médias contemporains, journaux, radio, télévision (chaînes d’information panarabes, telles Al Jazeera ou Al-Arabiya), et les possibilités multiples de la toile (internet).

Un vecteur important plus ancien est l’emprunt à l’arabe de mots et expressions par les langues non-arabes, telles les langues romanes, dont le français.

**Italian Language**

See Also:
- https://it.wikipedia.org/wiki/Lingua_italiana

L’italien est une langue appartenant au groupe des langues romanes de la famille indo-européenne. Il existe un très grand nombre de dialectes italo-romans.

Dante a donné à l’italien le surnom de langue de sì en le comparant à la langue d’oc (occitan) et à la langue d’oil (français), selon la manière de dire « oui » dans ces trois langues.

Comme beaucoup de langues nationales, l’italien moderne est un dialecte qui a « réussi » en s’imposant comme langue propre à une région beaucoup plus vaste que sa région dialectale originelle. En l’occurrence, c’est le dialecte toscan, de Florence, Pise et Sienne, qui s’est imposé, quoique dans sa forme illustre (koiné littéraire à base florentine enrichie par des apports siciliens, latins et d’autres régions italiennes) non pas pour des raisons politiques comme c’est souvent le cas, mais en raison du prestige culturel qu’il véhiculait.

Le toscan est en effet la langue dans laquelle ont écrit Dante Alighieri, Francesco Petrarca et Giovanni Boccaccio, considérés comme les trois plus grands écrivains italiens de la fin du Moyen Âge. C’est aussi la langue de la ville de Florence, réputée pour sa beauté architecturale et son histoire prospère. C’est donc sans surprise que l’italien fut pendant longtemps la langue internationale de la culture et des arts, et que le vocabulaire de toutes les langues
européennes conserve jusqu’à nos jours un grand nombre de termes italiens sont utilisés en musique, lesquels sont même repris dans d’autres langues, comme le japonais.

**Dialectes**

Les nombreux dialectes italo-romans peuvent être classés par leurs souches linguistiques communes. Ainsi, ils sont tous originaires du latin, mais les langues antérieures à la domination romaine, les substrats, sont différents en fonction des régions et ont souvent conditionné l’évolution des dialectes.

**Parlers gallo-italiques** Les dialectes septentrionaux, parfois regroupés sous le nom d’italien septentrional forment un groupe dialectal original dans l’ensemble italien

**Vénitien** Parfois classé à tort dans les parlers gallo-italiques, il est fortement italianisé et se distingue nettement de ceux-ci.

**Italien centro-méridional**

**Dialectes centro-méridionaux ou italien centro-méridional** Dialectes toscans (substrat étrusque hypothétique):

- toscan (Florence, Pise, Sienne), dialecte servant de base à l’italien standard
- corse (classé dans le diasystème italien, mais constituant une langue par élaboration)
  - cismontano (Corse : Bastia)
  - oltramontano (Corse : Ajaccio et îles du nord de la Sardaigne)
  - gallurais (nord-est de la Sardaigne, distinct du sarde)
- sassarais (langue de transition avec le sarde) (nord-ouest de la Sardaigne, distinct du sarde et du corse)

Dialectes du centre différents du toscan:

- romain (Latium), différent du romanesco, dialecte toscan parlé à Rome
  - ombrien (Ombrie)
  - marchigiano (parlers des Marches)

Dialectes méridionaux :

- Sud (substrats osque et/ou messapien):
  - abruzzais (Abruzzes)
  - molisien (Molise)
  - apulien (nord et centre des Pouilles)
  - campanien (Naples, napolitain) ou méridional
  - lucan ou lucanién Potenza, deux variétés
- Extrême-Sud (substrat grec) :
  - salentin (sud des Pouilles)
  - calabrais (Calabre)
  - sicilien (Sicile), également substrat arabe
**Spanish Language**

See Also:

- https://es.wikipedia.org/wiki/Idioma_espa%C3%B1ol

L’espagnol, ou castillan, est une langue romane, commune de l’Espagne et de nombreuses nations d’Amérique, ainsi que d’autres territoires dans le monde associés à un moment de leur histoire à l’ancienne métropole.

La langue espagnole, originaire du nord de l’Espagne, s’est diffusée dans le Royaume de Castille et s’est développée comme la langue principale du commerce et du gouvernement. Elle fut menée en Afrique, en Amérique et en Asie Pacifique avec l’expansion de l’empire espagnol entre le XVe et le XIXe siècle.

Aujourd’hui, 400 millions de personnes ont l’espagnol comme langue maternelle, ce qui la hisse au deuxième rang mondial en nombre de locuteurs natifs

**Variations et dialectes**

Article détaillé : Dialectologie de la langue espagnole.

**Dialectes d’Espagne** Parmi les modalités les plus remarquables du castillan parlé en Espagne, on peut citer l’andalou (notamment caractérisé par la présence de seseo ou de ceceo selon les zones), le murcien, le castuo et le canarien.

**Variations en Amérique hispanique** Parmi les plus remarquables, on peut citer l’utilisation d’un autre système de pronoms personnels. Le pronom de la troisième personne du pluriel ustedes (qui sert en Espagne uniquement à s’adresser à un ensemble de personnes que l’on vouvoie) remplace en Amérique hispanique le vosotros.vosotros (2e personne du pluriel, équivalent en Espagne au « vous » français lorsque l’on s’adresse à un ensemble de personnes que l’on tutoie).

La prononciation ibérique de la consonne c (devant les lettres e et i) ou z (devant a, o et u), est une spirante interdentale (proche du th anglais). En Amérique latine le phonème se prononce toujours /s/ (phonétiquement proche du s français, le s péninsulaire étant plus palatal), un important trait commun avec le canarien et une grande partie de l’andalou. Cette prononciation, appelée en espagnol « seseo », est généralisée en Amérique hispanique.

Les traits communs avec le dialecte andalou et canarien s’expliquent par le fait que la colonisation de l’Amérique hispanique et tous les échanges commerciaux avec celle-ci ont pendant longtemps été centralisés à Séville (les îles Canaries servant alors d’intermédiaires avec la Péninsule Ibérique), ce qui permettait un meilleur contrôle des flux par la Monarchie. C’est ainsi le dialecte andalou qui a été dominant chez les migrants, qui y passaient souvent de longs mois avant de pouvoir embarquer pour le Nouveau Monde.

Les nombreux esclaves africains déportés dans l’empire espagnol ont également influencé certaines différenciations des parlers d’Amérique et développé une forme d’espagnol particulière au contact des colons, tout en apportant leur accent africain.

De grandes disparités peuvent exister au niveau du lexique. Par exemple certains mots courant dans l’espagnol péninsulaire sont obscènes en Argentine, en Colombie ou au Mexique.

**Chinese Language**

See Also:

Les langues chinoises (Zhōngguó yǔwén) ou langues sino-tibétaines, sont souvent désignées sous le nom de « chinois » (zhōngwén), appartenant à la famille des langues sino-tibétaines. Dans leur caractère le plus universel, on considère la langue écrite (wén), transcendant la prononciation des divers parlers. La langue parlée dans son caractère le plus général est le plus souvent appelée hanyu ( ), soit « langue des Han », même si d’autres groupes ethniques ont progressivement adopté cette langue. Les différents parlers peuvent être considérés comme langue (y) ou comme dialecte (huà). Le statut d’un parler comme langue ou dialecte est souvent sujet à controverses en l’absence de références écrites à la prononciation.

On distingue généralement sept grands ensembles de parlers sino-tibétaines modernes:

- le chinois du Nord (/gunhuà, langue des officiels ou /Bēifeng huà, parlés du Nord), parlé dans le Nord et le Nord-Est de la Chine. C’est la langue la plus parlée au monde (environ 850 millions de locuteurs)
- le mandarin standard (/hány, globalement ou /putonghua, en République populaire de Chine, ou /guóy, en République de Chine (Taiwan)), est la variante standardisée du chinois du Nord et la langue officielle de la République populaire de Chine, de la République de Chine (Taiwan) et de Singapour. Il est par ailleurs parlé au Viêt Nam et au Cambodge ;
- le wu (/wúy), parlé à Shanghai, dans le Jiangsu et le Zhejiang (environ 77 millions de locuteurs) ;
- le cantonais (/yuèy), parlé dans les provinces du Guangdong et du Guangxi, à Hong Kong, Macao, en Asie du Sud-Est et par certains Chinois d’outre-mer (environ 71 millions de locuteurs) ;
- le gan (/gàny), parlé dans la province du Jiangxi (environ 31 millions de locuteurs) ;
- le hakka (/kèjihuà), dans la partie nord du Guangdong, le Fujian et à Taïwan (environ 34 millions de locuteurs) ;
- le min (/mny), dans les provinces du Guangdong, Fujian et à Taïwan (environ 60 millions de locuteurs) :
  – le groupe minnan (ou hokkien) comprend, entre autres, le taïwanais, le teochew et le hainanais,
  – le groupe minbei
- le xiang (/ Xingy), parlé dans la province du Hunan (environ 36 millions de locuteurs).

Certains linguistes distinguent parfois 3 autres langues importantes :

- le jin (/jìny), dans la zone Nord, distinct du chinois du Nord, donc du putonghua ;
- le hui (/huy), distinct du wu ;
- le ping (/pínghuà), distinct du cantonais.

Les écritures

On peut trouver, entre certains dialectes, des différences plus importantes qu’entre l’espagnol et le portugais, voire entre l’anglais et l’allemand, cependant, tous utilisent la même écriture.

Les langues sino-tibétaines s’écrivent le plus souvent au moyen de caractères chinois ou sinogrammes ; ils ont évolué au cours des siècles, et ont été simplifiés en République populaire de Chine en 1956. Les caractères traditionnels conservés hors de la Chine continentale diffèrent parfois légèrement entre Taïwan et Hong Kong.

Il existe aussi un code d’écriture chinois exclusivement utilisé par des femmes, le níshū, utilisé dans une région où, autrefois, les femmes n’avaient pas le droit d’écrire.

Dans les différentes régions ou districts autonomes, des écritures locales comme le dongba ou le dai sont utilisées.
Japanese Language

See Also:


Le japonais est la langue du Japon, parlée par le peuple japonais bien qu’aucune loi ne lui donne le statut de langue officielle. Elle est en revanche la langue des documents officiels et de l’éducation. Le japonais est également utilisé par la diaspora nipponne (notamment au Brésil et au Pérou) ainsi que sur l’île de Anguar dans les Palaos.


Le japonais appartient à la famille isolée des langues japoniques. Bien que sa structure morphologique et sa syntaxe ressemblent sensiblement à celles du coréen, aucune parenté n’a pu être clairement établie à ce jour.

Par ailleurs, son vocabulaire s’est notablement enrichi, au cours de l’Histoire, par le truchement de divers emprunts : le plus remarquable est la présence de nombreux vocables issus ou dérivés de la langue chinoise écrite, ce qui explique que le japonais soit qualifié de « langue sinoxénique » ; d’autre part, la langue contemporaine effectue de fréquents emprunts à diverses langues européennes, particulièrement à l’anglais.

Origine

Le japonais a longtemps été classé dans la famille des langues altaïques avec le mongol, le turc, le toundouze et le coréen, mais l’existence même d’une famille altaïque est niée par de nombreux spécialistes. Certaines théories font du japonais une langue mixte, mêlant des éléments des langues altaïques et des langues austronésiennes.

Le japonais est donc aujourd’hui encore généralement considéré comme un isolat linguistique. Les langues ryukyu (traditionnellement classées comme « dialectes japonais ») sont les seules langues dont la parenté avec le japonais a été prouvée.

Grammaire, syntaxe et usages

Article détaillé : Grammaire japonaise.

C’est une langue agglutinante.

Le japonais est une langue dite « à tête finale » : le prédicat se place à la fin de la phrase, l’objet est placé devant le verbe, l’adjectif se met devant le substantif, et la morphologie est principalement suffixante.

Il n’y a ni article, ni genre, ni nombre ; les verbes ne se conjuguent pas selon les personnes (je, tu, il ...) ; des particules invariables indiquent la fonction du mot dans la phrase (leur rôle est donc similaire à celui des cas dans la plupart des langues agglutinantes ou flexionnelles. En japonais on précise les éléments sus mentionnés si le besoin s’en fait sentir (par exemple pour lever une ambiguïté).

Dialectes

Comme pour la plupart des langues nationales, il existe de nombreux dialectes japonais qui se distinguent par la phonologie, le vocabulaire et la grammaire. « Dialecte » se dit « hgen » () en japonais, et dans la langue courante pour désigner un dialecte on accole le suffixe ben () au nom de la localité où il est parlé. Les études de dialectologie n’utilisent cependant pas ce suffixe.

Les langues parlées dans les îles Ryky ne sont pas des dialectes du japonais mais des langues propres appelées langues ryuku, le plus connu étant l’okinawaïen. Elles appartiennent à la famille des langues japoniques avec le japonais. Elles ne sont cependant pas officiellement reconnues par le Japon comme des langues différentes du japonais, et sont traditionnellement classées comme « dialectes japonais ».

4.1. Languages 1153
L’aïnou parlé par l’ethnie des aïnous vivant dans l’extrême nord du Japon n’est pas une langue japonique et est considéré comme un isolat.

**Politesse**

Article détaillé : Keigo.

La politesse japonaise, ou en japonais keigo (けいご), dont une traduction approximative serait « langage respectueux », constitue d’un certain point de vue une langue dans la langue, et est le reflet direct de la structure et des interactions sociales.

L’utilisation de la politesse est un pré-requis dans la majorité des situations sociales : cet élément est plus important au Japon qu’en France ou en Occident.

La systémique de la politesse japonaise peut apparaître difficile au premier abord, mais ses concepts de base sont relativement faciles à intégrer. Cependant, la maîtrise de la politesse japonaise à un niveau avancé, subtil et instinctif, notamment à l’écrit, est, de l’aveu des Japonais eux-mêmes, particulièrement ardue.

Pour reprendre la définition du sinologue Sadaki Hagino, la politesse japonaise peut se définir comme « un système organisé de mots visant à exprimer la reconnaissance de différentes nuances de différence de hauteur entre plusieurs personnes » (؟). D’autres langues, comme le coréen et le javanais, connaissent un système comparable.

Alors que dans la majorité des langues occidentales la « politesse » ne s’exprime que vis-à-vis de son interlocuteur (choix du tutoiement ou du vouvoiement en français par exemple), il existe une distinction claire dans la politesse japonaise entre :

- le wadai (わだい), objet de la conversation, c’est-à-dire la personne/le groupe social dont on parle.
- le dentatsu (でんたつ), [situation de] communication, c’est-à-dire la personne/le groupe social à qui l’on parle.

Par ailleurs, la politesse japonaise repose sur la distinction fondamentale entre :

- uchi (うち, « intérieur », c’est-à-dire les membres de son propre groupe social)
- et soto (そと, « extérieur », c’est-à-dire les membres d’un groupe social différent de son propre groupe).

La politesse japonaise comporte concrètement trois dimensions relativement indépendantes :

- sonkeigo (尊敬語), langage de respect
- kenjgo (謙敬語), langage d’humilité
- teineigo (他尊語), langage de courtoisie.

Chacune de ces trois dimensions possède un certain nombre de nuances, notamment d’intensité.

Ainsi, les moyens qu’offrent la politesse japonaise permettent (et souvent la situation sociale impose) par exemple :

- de parler familièrement à quelqu’un de quelqu’un d’autre avec respect ;
- de parler courtoisement à quelqu’un de quelqu’un d’autre sans respect ;
- d’exprimer de la courtoisie à son interlocuteur sans lui exprimer de respect (voir plus haut) ;
- d’exprimer du respect à son interlocuteur (ce qui implique de lui exprimer de la courtoisie, et le plus souvent d’exprimer de la modestie envers soi-même) ;
- d’exprimer (au moyen du langage de respect et du langage de modestie) à son interlocuteur la relation entre deux personnes externes, ce qui peut se faire en parlant familièrement ou courtoisement à son interlocuteur ; etc.
Swedish Language

See Also:

- https://fr.wikipedia.org/wiki/Su%C3%A9dois
- https://fr.wikipedia.org/wiki/Dialectes_du_su%C3%A9dois

Classification

Le suédois est une langue appartenant à la branche nordique ou scandinave du groupe germanique de la famille des langues indo-européennes.

Répartition géographique

Il est parlé en Suède et en Finlande1, par environ dix millions de personnes.

Grammaire

Article détaillé : Grammaire suédoise.

Le suédois est une langue germanique à la conjugaison et à la grammaire simples, ne possédant pas de cas (à l’exception d’un reste de génitif), et dans laquelle les mots peuvent être modifiés par des ajouts de préfixes, suffixes ou par simple concaténation de mots.

Dans la conjugaison, par exemple, la personne n’a la plupart du temps aucune incidence sur la forme verbale.

Lexique

La majeure partie du vocabulaire suédois est d’origine germanique. On distingue le fonds germanique natif, et les emprunts faits aux autres langues germaniques par l’intermédiaire de l’allemand et du néerlandais, (en général, du bas-allemand, du fait de son statut de lingua franca dans la zone hanséatique), ainsi que quelques emprunts à l’anglais.

Les mots suédois hus (maison), kung (roi) et “gås” (oie) sont d’origine germanique.

La plupart des termes du vocabulaire religieux ou scientifique sont d’origine grecque ou latine, le plus souvent empruntés par l’intermédiaire du français, ou, plus récemment, de l’anglais. Plusieurs de ces formations de mots sont des emprunts, tels le mot “bomull” (coton), de l’allemand “Baumwolle” (le calque trädull n’a pas été retenu). Le suédois de Finlande a parfois des termes propres proches des mots finnois correspondant, particulièrement dans le vocabulaire juridique et administratif.

On forme facilement de nouveaux mots par synthèse comme dans les autres langues germaniques. A l’image de ce qui se passe en allemand et en néerlandais, ces mots composés peuvent devenir très longs et difficiles à employer. Par exemple, produktionstjörningssystemsprogramvaruuppdatering signifie à lui seul mise à jour du logiciel du système de contrôle de la production.


4.1. Languages
Statut officiel

Le suédois est la langue nationale de facto en Suède et une des langues officielles de la Finlande (avec le finnois) ; sur l'archipel d'Åland il est la seule langue officielle. Il y a aussi 3 municipalités en Finlande où le suédois est la seule langue officielle sur le plan municipal. L’une d’elles, Korsnäs, possède le taux le plus élevé de locuteurs natifs suédophones (98%).

En 2006, une proposition a été faite dans le Riksdag suédois pour lui donner le statut de langue officielle. Cette proposition avait le soutien du bloc de droite, ainsi que celui du parti vert, qui a voté majoritairement en sa faveur. Cependant, à cause d’une erreur dans le décompte des voix, le parti de gauche et les sociaux-démocrates, qui s’y opposaient, ont gagné le vote. Les partis de droite ont déclaré qu’ils essaieraient à nouveau, plus tard en 2006 ou au début de 2007.

Suédois contemporain

On donne le nom de suédois contemporain (nusvenska) à l’état que la langue connaît aujourd’hui, et depuis environ 1900. Avec l’industrialisation et l’urbanisation de la Suède — déjà en bonne voie dès les années 1890 — ce sont de nouvelles catégories de personnes qui commencent à faire leur entrée dans la littérature suédoise. De nombreux auteurs nouveaux, hommes politiques et autres personnages publics exerçaient une profonde influence sur la langue nationale qui se développait. Si l’on cherche un point de départ précis, on peut poser l’année 1879 (celle de La Chambre rouge), et la percée d’August Strindberg (1849-1912), un des auteurs les plus influents.

Une réforme de l’orthographe, initiée par le ministre des Affaires religieuses Fridtjuv Berg (sv) en 1906, fit une démarcation nette avec ce qu’on appellerait par la suite l’ancienne orthographe : haf devint hav (radical du verbe avoir), rodt devint rött (adjectif rouge accordé au genre neutre), etc.

La règle orthographique – pas si ancienne qu’on pourrait le croire – qui fait la différence entre les participes passés et les supins (huset är måladt, jag har målat : la maison est peinte, j’ai peint), fait partie des quelques-unes qui manquent encore à cette époque. La disparition de la graphie hv, par exemple en tête des mots interrogatifs hvem (qui), hvar (où), ont été pointées du doigt par beaucoup, parce qu’elle éloignait le suédois du danois et du norvégien.

C’est pendant le XXe siècle qu’une langue nationale commune, standardisée, vint à la portée de la grande majorité des Suédois. L’orthographe était définitivement standardisée et presque entièrement unifiée depuis la réforme de 1906. À l’exception des formes plurielles des verbes (comme vi komma, nous venons, alors que la graphie moderne est vi kommer) et de quelques différences ponctuelles dans l’ordre des mots, en particulier dans la langue écrite (par exemple l’inversion Och beslutade styrelsen att. . . , où le sujet styrelsen passe après le verbe beslutade lorsque la proposition commence par une conjonction de coordination comme och), la langue était globalement identique au suédois parlé aujourd’hui. Les formes du pluriel subsistèrent, mais furent de moins en moins utilisées, et disparurent enfin en 1950, lorsque les dernières recommandations officielles à propos de leur usage furent supprimées.

Le changement le plus visible consista en un raccourcissement des usages du suédois formel, pour aller vers le plus facile à lire et à prononcer. Les exemples les plus patents sont le raccourcissement d’un petit nombre de verbes très courants : tager devint tar (prendre), ikläda sig devint klä sig i (s’habiller). Skall (auxiliaire dénotant le futur) semble revenir, mais il est encore écrit en général sous sa forme racourcie ska. Au cours des années 1970 et 1980 apparurent des formes comme sen au lieu de sedan (adverbe ensuite, cf. anglais then), nån au lieu de någon (pronom quelqu’un, ou déterminant quelque), dom au lieu de de ou dem (pronom personnel de troisième personne du pluriel), dej au lieu de dig (cas objet du pronom personnel de deuxième personne du singulier). Ce mouvement semble s’être éteint depuis. Des conjonctions comme ehuru, därest et ity ont cédé du terrain par rapport à leurs homologues issus de la langue orale : fast, om et därför.

Depuis les années 1970, le développement (et la fabrication) d’un suédois d’usage courant, compréhensible, à l’oral, a formé un des combat les plus essentiels concernant cette langue.

Un changement important dans la réalité sociale de la langue intervint dans les années 1960 avec ce qu’on a appelé la réforme du tu (du-reformen). On considérait auparavant que le mieux était de s’adresser aux personnes d’un rang social comparable au sien ou plus élevé en utilisant un titre et un nom de famille. L’usage de herr (monsieur), fru (madame)
et fröken (mademoiselle) était en général restreint à la conversation avec des personnes dont la profession, les titres académiques ou le rang militaire n’était pas connu de leur interlocuteur. On se posait parfois la question de savoir s’il fallait s’adresser à son interlocuteur à la troisième personne. Pour résoudre ce problème, des expressions comme vad fär det lov att vara? ou tas det socker i kaffet? (utilisation de la forme passive : Est-ce que du sucre est pris dans le café ?) étaient utilisées. Au début du XXe siècle, beaucoup essayèrent de remplacer ce système compliqué de titres par le pronom vous, à l’image de ce qui était fait en français ou en allemand. Vous (ni) fut cependant rapidement utilisé comme une variante un peu moins arrogante de tu (du, ou de la troisième personne) pour s’adresser à des personnes de rang social inférieur. Avec la libéralisation et un virage à gauche de la société suédoise pendant la seconde moitié du XXe siècle, ces différences de classes devinrent moins pertinentes et du (tu) devint le terme d’adresse habituel, même au sein des communautés les plus formelles et officielles. Ce qu’on appelle le nouveau vouvoiement (“det nya niandet”) aux caisses des supermarchés, ou dans certains emplois de services, est un phénomène marginal.

Écriture

L’exemplaire le plus ancien du Äldre Västgötalagen (Loi de la Westrogothie) à avoir été préservé, datant des années 1280, est l’un des plus anciens exemples de langue suédoise écrite à l’aide de l’alphabet latin.

Le suédois utilise l’alphabet latin, avec les lettres porteuses de diacritiques à (a rond en chef), ä (a tréma ou umlaut) et ö, qui apparaissent dans les dictionnaires à la suite de la lettre z. Pour ce qui est des langues scandinaves, le à se retrouve en danois et norvégien, mais dans ces langues les lettres ä et ö ont pour équivalents respectifs les lettres æ et ø ; en islandais, on emploie æ et ö pour ces deux derniers caractères, tandis que le à n’existe pas (son équivalent est á) et que d’autres diacritiques existent (é, í, ú, ý, ainsi que les deux consonnes spéciales ð et þ).

Les caractères ä et ö se retrouvent par ailleurs dans d’autres langues, certaines germaniques (comme l’allemand), et d’autres non (le finnois par exemple, qui emploie par ailleurs, bien que très rarement, le à dans certains mots empruntés au suédois).

Dialectes

See Also:
https://fr.wikipedia.org/wiki/Dialectes_du_su%C3%A9dois

Le suédois, comme la plupart des langues nationales de l’Europe connaît un très grand nombre de variantes différentes, appelées dialectes. Parmi ces dialectes, on distingue le plus souvent six grands groupes:

- Le norrlandais, les dialectes du Norrland (Norrländska mål)
- Le svealandais, les dialectes du Svealand (Sveamål) (n’y sont pas compris les dialectes du Värmland qui sont plus proches des dialectes du Götaland méridional)
- Le gotlandais, les dialectes du Gotland (Gotländska mål)
- Le götalandais, les dialectes du Götaland (Götamål) (y sont compris aussi les dialectes du Värmland, mais non pas les dialectes des anciennes provinces danoises: la Scanie, le Blekinge et le Halland, ni les dialectes de l’île de Gotland)
- Le suédois méridional, les dialectes du sud de la Suède (Sydsvenska mål)
- Le suédois oriental, les dialectes du suédois parlés largement sur la côte ouest de la Finlande (en Ostrobotnie et Åland) aussi bien qu’à la paroisse estonienne de Noarootsi, Nuckö (Östsvenska mål)

Comme les dialectes du norvégien et du danois, les dialectes du suédois plongent leur racines dans le vieux norrois. En effet, les frontières linguistiques séparant les langues scandinaves restent très floues. Pour cette raisons, les dialectes des provinces suédoises voisines avec la Norvège et le Danemark pourraient aussi bien être considérés comme des dialectes du norvégien ou du danois. C’est au moins le cas pour les variantes les moins influencées par le suédois standard parlées en Bohuslän (vieille province de la Norvège) et en Scanie (vieille province du Danemark).
La frontière linguistique entre le suédois et le finnois, par contre, est très marquée, le finnois n’étant pas une langue indo-européenne, mais une langue finno-ougrienne.

**Old norse Language**

See Also:

- https://en.wikipedia.org/wiki/Old_Norse

Le vieux norrois (ou norrois, norois ou encore vieil islandais) correspond aux premières attestations écrites d’une langue scandinave médiévale.

Le vieux norrois est de loin la variété la mieux attestée d’ancien scandinave : le norrois « classique » est le langage dans lequel ont été rédigées les sagas islandaises des XIIe et XIIIe siècles, dont la plus connue, l’Edda décrivant avec une neutralité étonnante de la part d’un clerc la mythologie viking.

**Diffusion**

Le vieux norrois a influencé de nombreuses langues : le russe, l’anglais, le normand et le français via le normand. Les langues qui en sont aujourd’hui les plus proches sont les langues scandinaves.

En France, la toponymie normande et l’anthroponymie sont caractérisées par la présence d’appellatifs toponymiques issus du vieux norrois

- both > -beuf « barraque, village »,
- topt > -tot « établissement »,
- bekkr > -bec « ruisseau »,
- lundr > -lon « bois, forêt »,
- lunda > londe « bois, forêt »,
- thorp > torp(s), tour(p) « hameau »,
- thveit > -tuit, Thuit- « essart »,

etc.) et les nombreux noms de personnes qui y sont inclus, ainsi que l’existence de nombreux patronymes, jadis prénoms (Ouf, Ingouf, Toutain, Anquetil, Turquetil, Angot, Anfry, Estur, Doudement, etc.); ceci est dû à l’installation de colons danois, norvégiens et anglo-scandinaves au Moyen Âge qui se poursuit sur plusieurs générations après la création d’une principauté normande (en 911).
PYTHON MODULE INDEX

1
14443,??
14443Faq,??
a
AdminSys,31
AnnexesProprietary,??
b
borland,??
borland_firmware,??
c
C#_language,??
C_stdio,??
calypso,??
cli_bash,??
command_line_interface,??
CompilingDrivers (GNU/Linux), ??
cryptography,??
cygwin,??
C_language,??
d
Documenting,??
dpinst (windows XP, Vista, windows 7), ??
Drivers (windows XP, Vista, windows 7, GNU/Linux), ??
drivers_smartcard_storage (windows 7), ??
DVCS,??
e
embedded_operating_systems,??
f
felica,??
g
GCC,??
GCC64,??
git,??
github2,??
GNU_Linux_MUSCLE_news (GNU/Linux), ??
GNU_Linux_system, ??
GPL_license (linux kernel), ??
GUI, ??
h
Hardware,??
i
ICAO, ??
ICAO_annexes, ??
Innosetup,??
internationalization_i18n,??
Ironpython,??
iso15408,??
isol5693,??
j
Java_language,??
l
lafargue_smart_tools,??
liefprint (GNU/Linux), ??
liefprint_src (GNU/Linux), ??
liefprintd (GNU/Linux), ??
libhid (windows XP, Vista, windows 7, GNU/Linux), ??
libhid_news (windows XP, Vista, windows 7, GNU/Linux), ??
libopenusb (windows XP, Vista, windows 7, GNU/Linux), ??
lipcsclite_2.0.0 (GNU/Linux, Darwin), ??
libusb (windows XP, Vista, windows 7, GNU/Linux), ??
libUSB-winusb, ??
libusb_l.0.8 (GNU/Linux, Darwin), ??
libusb_l.0.9 (GNU/Linux, Darwin), ??
libusb_linux (GNU/Linux), ??
libusb_versions (GNU/Linux, Darwin), ??
libwdi (windows XP, Vista, windows 7), ??
libwdi_news (windows XP, Vista, windows 7), ??
Licences_Libres,??
linux,??

1159
Development tools, Release 2012.06.18

LinuxKernel2_6_31_USB_core (Linux kernel 2.6.31 (portux)), ??
LinuxKernel2_6_31_USB_HID (Linux kernel 2.6.31 (portux)), ??
LinuxKernel2_6_31_USB_image (Linux kernel 2.6.31 (portux)), ??
LinuxKernel2_6_31_USB_net (Linux kernel 2.6.31 (portux)), ??
LinuxKernel2_6_31_USB_storage (Linux kernel 2.6.31 (portux)), ??
LinuxKernel2_6_31_USBDocumentation (Linux kernel 2.6.31), ??
LinuxKernel2_6_31_USBSrc (Linux kernel 2.6.31 (portux)), ??
LinuxKernelUSB_commands_fr (Linux kernel), ??
LinuxKernelUSBDocumentation (Linux kernel), ??
LinuxUdev, ??
LinuxUdevReadme, ??
localization, ??
Ludovic_Rousseau (GNU/Linux), ??
Ludovic_Rousseau_blog (GNU/Linux), ??

MacOSXOperatingSystems, ??
mercurial, ??
mifare, ??
MinGW, ??
ModuleLinux (GNU/linux), ??
ModutilsLinux (GNU/linux), ??
MSVC64, ??
MSYS, ??
Multimédia, ??

Near_Field_Communication, ??
netbeans, ??
Netbeans, ??
NetbeansBugs, ??
NetbeansGCC, ??
NetbeansProperties, ??
ninja_ide, ??
Notepad++, ??

Open_data, ??

PCSC, ??
PCSC_muscle, ??
perl, ??
presse, ??
privacy, ??
programming_languages, ??
pyGBL2SIM, ??
pylibusb (GNU/Linux, Windows, MacOSX), ??
pypassword, ??
pypassword_annexes, ??
pyQt, ??
pySide, ??
PythonDistributeNews, ??
PythonGUI, ??
PythonInstallation, ??
PythonNews, ??
PythonWmiWindows, ??
pyusb (GNU/Linux, Windows, MacOSX), ??
pyusb_news (GNU/Linux, Windows), ??
pywinusb (Windows), ??

rfidiot, ??
rfidiot_intro, ??

smartcard_contactless, ??
smartcard_FAQ, ??
smartcards, ??
SoftwareEngineering, ??
SoftwareEngineering_CE161508, ??
sphinx, 3

Test, ??

USB-ip, ??
usb.ids, ??
USB_authorised, ??
USB_device, ??
USB_DFU_Device_Firmware_Upgrade, ??
USB_DFU_introduction, ??
USB_DFU_programmer, ??
USB_doc, ??
USB_host, ??
USB_muscle_news, ??
USB_on_the_go_OTG, ??
usbcid_driver_old, ??
USBDrivers (windows XP, Vista, windows 7, GNU/Linux), ??
USBFAQs, ??
USBhid, ??
USBNews, ??
usbtouls (GNU/Linux), ??
usbtouls_news (GNU/Linux), ??
usbwin32 (Windows), ??

VCS, ??
Vim, ??

W
Windows_ccid_driver_manager, ??
Windows_devcon_tool, ??
Windows_Drivers_install (windows XP, Vista, windows 7, GNU/Linux), ??
Windows_tools, ??
windres, ??
wzmrtd, ??
wzmrtd_intro, ??
wzpass, ??