MyData Documentation

*Release 0.2.0-alpha2*

James Wettenhall

April 14, 2015
1 Contents
  1.1 Overview ................................................................. 1
  1.2 Download ............................................................... 5
  1.3 MyTardis Prerequisites ............................................. 6
  1.4 Settings ................................................................. 6
  1.5 Upload Methods ....................................................... 13
  1.6 User Groups ............................................................ 18
  1.7 Mac OS X Walkthrough ............................................. 24
  1.8 MyData Tutorial, using the MyTardis Demo Server .......... 32
  1.9 MyData Client and Daemon ........................................ 48
  1.10 License ............................................................... 49

2 Indices and tables .................................................. 61
1.1 Overview

MyData is a desktop application for uploading data to MyTardis (https://github.com/mytardis/mytardis). It allows users of a data-collection instrument to have their data automatically uploaded to MyTardis simply by saving their data in a pre-defined folder structure. The simplest folder structures available is “Username / Dataset”, which is described below.

We begin with a root data directory (e.g. “C:\MyTardisUsers”) containing one folder for each MyTardis user. In the example below, we have two users with MyTardis usernames “testuser” and “wettenhj”:

![MyTardisUsers folder structure example]

Within each user folder, we can add as many folders as we like, and each one will become a “dataset” within MyTardis:

![wettenhj folder structure example]
MyData is designed to be able to run interactively or in background mode. Its setup wizard for Windows offers to create a shortcut to MyData in the user’s Startup Menu which will automatically run MyData in background mode (using its “–background” command-line argument) every time the user logs in. Many data-collection instrument PCs use a shared login account which remains logged in all day long. MyData at present cannot run as a service daemon - so it will not run while no users are logged in.

The first time you run MyData, you will be asked to enter some settings, telling the application how to connect to your MyTardis instance. You can use a MyTardis account which is shared amongst facility managers, but which general users don’t have access to.
Each time the application starts up (and when you tell it to), it will scan all of the user and dataset folders within your primary data directory (e.g. C:\MyTardisUsers) and present a list of all of the dataset folders in a tabular view (below). MyData will count the number of files within each dataset folder (including nested subdirectories), and then query MyTardis to determine how many of these files have already been uploaded. If MyData finds new files which haven’t been uploaded, it will begin uploading them (with a default maximum of 5 simultaneous uploads). You can see progress of the uploads in the “Uploads” tab.

When run in background mode (the default for the shortcut installed by the Windows Setup Wizard in the user’s “Startup Items” folder), MyData is intentionally difficult to shut down. It is designed to be shut down only by facility managers, not by an individual researcher. Closing the main window will minimize the MyData application to an icon in the System Tray (below). It is possible to exit MyData using a menu item from the System Tray icon’s pop-up menu (further below), but upon clicking on this menu item, the user will be asked for administrator privileges.
Clicking on MyData’s System Tray icon will bring up a menu, allowing you to restore MyData’s main window (the “Control Panel”) or force a “MyTardis Sync” to ensure that new data is uploaded promptly:

You can tell when MyData has finished uploading a dataset by looking at the number of files uploaded in the Status column of the Folders view. Then you can select that dataset’s row in the Folders view and click on the “Web Browser” icon to view that dataset in MyTardis.

MyTardis uses “experiments” to organize collections of datasets. When using the default “Username / Dataset” folder structure, the default name for each experiment created by MyData will be the instrument name (e.g. “Test Microscope”), followed by the data owner’s full name (if it can be retrieved from MyTardis using the username given as a folder name).

The experiment will initially be owned by the facility manager user specified in MyData’s Settings dialog (e.g. “test-facility”). MyData will then use MyTardis’s ObjectACL’s (access control lists) to share ownership with the individual researcher (e.g. “wettenhj” or “skeith”) who must have a MyTardis account. Below we can see the experiments created by MyData as owned by the facility manager user (“testfacility”).
And below, we can see user wettenhj's data - note that “wettenhj” is now the logged-in MyTardis user in the upper-right corner, instead of “testfacility”.

1.2 Download

- Download MyData for Windows: MyData_v0.2.0.exe
1.3 MyTardis Prerequisites

MyData has been developed against a fork of MyTardis’s “develop” branch called “mydata”, which is currently a few commits behind MyTardis’s “3.6” release, but with some significant additions which are required for MyData. Work is underway to package the additions in the “mydata” MyTardis branch into a separate MyTardis app to make them easy to install on top of an existing MyTardis 3.6 installation.

The MyTardis branch which is currently used with MyData can be found here: https://github.com/wettenhj/mytardis/tree/mydata

MyData stores metadata for each experiment it creates (usually including the instrument name, the researcher’s MyTardis username, and possibly the date of collection of the data). A schema must be added to MyTardis to support this:

![Django administration](image)

1.4 Settings

MyData’s Settings dialog can be opened by clicking on the icon on MyData’s toolbar. The Settings dialog will be automatically displayed each time MyData is launched, unless MyData is run in “--background” mode (which is the default for the MyData shortcut installed in the “Startup” folder on Windows if the appropriate checkbox is ticked in MyData’s setup wizard).
### 1.4.1 General

**Instrument Name**  The name of the instrument (e.g. “Nikon Microscope #1”) whose data is to be uploaded to MyTardis by this MyData instance. If an instrument record with this name doesn’t already exist in MyTardis within the facility specified below, then MyData will offer to create one (assuming that you are a member of a facility managers group for that facility in MyTardis).

**Facility Name**  The name of the facility containing the instrument to upload data from. A facility record must have been created by your MyTardis administrator before you can use MyData, and the default MyTardis username you enter below (the initial owner of all data uploaded by this instance) must be a member of the managers group for that facility. MyData will automatically check that a facility record matching this facility name exists on the MyTardis server specified by the MyTardis URL below. If it doesn’t exist, MyData will offer suggestions for alternative facilities which your MyTardis account is a manager of (if any).

**Contact Name**  MyData’s preferred upload method (staging) requires approval from a MyTardis administrator. This Contact Name will be used when sending confirmation that access to MyTardis’s staging area has been approved for this MyData instance.

**Contact Email**  MyData’s preferred upload method (staging) requires approval from a MyTardis administrator. This Contact Email will be used when sending confirmation that access to MyTardis’s staging area has been approved for this MyData instance.

**Data Directory**  Choose a folder where you would like to store your data. e.g. D:\Data

**MyTardis URL**  The URL of a MyTardis server running a MyTardis version compatible with MyData, e.g. http://118.138.241.91/

**MyTardis Username**  Do not put your individual MyTardis username (e.g. “jsmith”) in here. Because MyData is
designed to be able to upload multiple users’ data from an instrument PC, the default username used by MyData should generally be a facility role account, e.g. “testfacility”.

**MyTardis API Key**  API keys are similar to passwords, but they are easier to revoke and renew when necessary. Ask your MyTardis administrator for the API key corresponding to your facility role account.

### 1.4.2 Schedule

![Schedule settings](image)
**Schedule type - Once** Run the folder scans and uploads once, on the date specified by the Date field and at the time specified by the Time field.

**Schedule type - Daily** Run the folder scans and uploads every day, at the time specified by the Time field.

**Schedule type - Weekly** Run the folder scans and uploads every week on the day(s) specified by the weekday checkboxes, at the time specified by the Time field.

**Schedule type - Timer** Run the folder scans and uploads repeatedly with an interval specified by the “Timer (minutes)” field between the hours of “From” and “To”, every day.

**Schedule type - Manually** Only run the folder scans and uploads in response to user interaction - either clicking OK on the Settings Dialog, or clicking the Refresh icon on the toolbar, or clicking the “MyTardis Sync” menu item in the system tray menu.

**Schedule type - On user login** On Windows, a shortcut to MyData (configured to run in –background mode) will be placed in the current user’s Startup folder. The exact location varies, but on my machine it is “C:\Users\wettenh\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup”. This is not implemented on Mac OS X yet, but a launch agent can be created for MyData in the user’s ~/Library/LaunchAgents/ folder, or a login item could be created in the user’s ~/Library/Preferences/com.apple.loginitems.plist

**Schedule type - On system startup** If MyData is installed as a service (not yet possible), then upon selecting “On system startup” for the Schedule Type, MyData will request administrator privileges to ensure that the MyData service is installed and configured to start up automatically. If MyData is installed as a service, then the shortcut to MyData required to launch MyData’s user interface will be placed in the common Startup folder. Again, the location varies, but for me it is: “C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup”. In the case where MyData runs as a service, MyData’s settings will be saved in a central location (e.g. “C:\ProgramData\Monash\MyData” or “C:\Users\All Users\AppData\Local\Monash\MyData” (the latter is a symbolic link to the former on my machine), rather than being stored in one specific user’s account. MyData’s user interface will run as a regular user. If it detects that the MyData service is running, then it will automatically run in client mode, and leave all of the heavy lifting to the MyData service.
1.4.3 Advanced

Folder Structure - Username / Dataset Folders immediately inside the main data directory (e.g. “D:\Data\jsmith”) are assumed to be MyTardis usernames. Folders inside each user folder (e.g. “D:\Data\jsmith\Dataset1”) will be mapped to MyTardis datasets. Datasets will be automatically grouped into MyTardis experiments according to the “Experiment (Dataset Grouping)” field below.

Folder Structure - Email / Dataset Folders immediately inside the main data directory (e.g. “D:\Data\John.Smith@example.com”) are assumed to be email addresses which can be used to match MyTardis user accounts. If you wish to use email addresses as folder names, an alternative is to use the “Username / Dataset” folder structure and use email addresses for usernames in MyTardis. Folders inside each email folder (e.g. “D:\Data\John.Smith@example.com\Dataset1”) will be mapped to MyTardis datasets. Datasets will be automatically grouped into MyTardis experiments according to the “Experiment (Dataset Grouping)” field below.

Folder Structure - Username / Experiment / Dataset Folders immediately inside the main data directory (e.g. “D:\Data\jsmith”) are assumed to be MyTardis usernames. Folders inside each user folder (e.g. “D:\Data\jsmith\Experiment1”) will be mapped to MyTardis experiments. Folders inside each experiment folder (e.g. “D:\Data\jsmith\Experiment1\Dataset1”) will be mapped to MyTardis datasets.

Folder Structure - Email / Experiment / Dataset Folders immediately inside the main data directory (e.g. “D:\Data\John.Smith@example.com”) are assumed to be email addresses which can be used to match MyTardis user accounts. If you wish to use email addresses as folder names, an alternative is to use the “Username / Experiment / Dataset” folder structure and use email addresses for usernames in MyTardis. Folders inside each email folder (e.g. “D:\Data\John.Smith@example.com\Experiment1”) will be mapped to MyTardis experiments. Folders inside each experiment folder (e.g. “D:\Data\John.Smith@example.com\Experiment1\Dataset1”) will be mapped to MyTardis datasets.
Folder Structure - Username / “MyTardis” / Experiment / Dataset  Folders immediately inside the main data directory (e.g. “D:\Data\jsmith”) are assumed to be MyTardis usernames. Folders inside each “MyTardis” folder (e.g. “D:\Data\jsmith\MyTardis\Experiment1”) will be mapped to MyTardis experiments. Folders inside each experiment folder (e.g. “D:\Data\jsmith\MyTardis\Experiment1\Dataset1”) will be mapped to MyTardis datasets.

Folder Structure - User Group / Instrument / Full Name / Dataset  Folders immediately inside the main data directory (e.g. “D:\Data\SmithLab”) are assumed to be MyTardis user groups. The actual group name in MyTardis (e.g. “TestFacility-SmithLab”) may have a prefix (e.g. “TestFacility-”) prepended to it, specified by the “User Group Prefix” field below. Each user group folder should contain exactly one folder (e.g. “D:\Data\SmithLab\Nikon Microscope #1”) specifying the name of the instrument. Using this scheme allows copying data from multiple instruments to a file share with the instrument name folder allowing users to distinguish between datasets from different instruments on the file share. Folders inside each instrument folder (e.g. “D:\Data\SmithLab\Nikon Microscope #1\John Smith”) indicate the name of the researcher who collected the data or the researcher who owns the data. Access control in MyTardis will be determined by the user group (“Smith Lab”), whereas the researcher’s full name will be used to determine the default experiment (dataset grouping) in MyTardis. Folders inside each full name folder (e.g. “D:\Data\SmithLab\Nikon Microscope #1\John Smith\Dataset1”) will be mapped to MyTardis datasets.

Check For Missing Folders  This option provides extra validation of the folder structure. For example, if you choose a folder structure of “Username / Dataset” but you don’t have any Dataset folders, then when this checkbox is ticked, MyData’s settings validation will stop and display an error as shown below.

Experiment (Dataset Grouping)  Defines how datasets will be grouped together into experiments in MyTardis. Currently, this field is automatically populated when you select a folder structure (above), and cannot be modified further.

User Group Prefix  Used with the “User Group / Instrument / Full Name / Dataset” folder structure. Folders immediately inside the main data directory (e.g. “D:\Data\SmithLab”) are assumed to be MyTardis user groups. The actual group name in MyTardis (e.g. “TestFacility-SmithLab”) may have a prefix (e.g. “TestFacility-”) prepended to it.

Ignore datasets older than  MyData is designed to be used for uploading recent data. If it is configured to use an existing data directory containing a large backlog of old data, it is advisable to instruct MyData to ignore old datasets so that it focuses on uploading the recent datasets.

Max # of upload threads  The maximum number of uploads to perform concurrently. If greater than one, MyData will spawn multiple scp (secure copy) processes which (for large datafiles) may impact significantly on CPU usage of your system, which could affect other applications running alongside MyData. The default value is 5.

### 1.4.4 Locking and Unlocking MyData’s Settings

At the bottom of MyData’s Setting dialog is a Lock/Unlock button, whose label toggles between “Lock” and “Unlock” depending on whether the Settings dialog’s fields are editable or read-only. When the Settings dialog’s fields are editable, clicking the “Lock” button will make them read-only, preventing any further changes to MyData’s settings until an administrator has unlocked the settings. The locked status will persist after closing and relaunching MyData. Clicking the “Lock” button displays the confirmation dialog below.
Once MyData’s settings are locked, all of the fields in the Settings dialog will become read-only.

Clicking on the “Unlock” button will result in a request for administrator privileges.
Once administrator privileges have been verified, it will be possible to modify MyData’s settings again.

N.B. This is NOT a security mechanism - it is a mechanism for preventing the accidental modification of settings in a production workflow. It does not prevent advanced users from determining where MyData saves its last used configuration to disk (e.g. C:\Users\jsmith\AppData\Local\Monash University\MyData\MyData.cfg) and updating the settings outside of MyData.

### 1.4.5 Saving and Loading Settings

Each time you click OK on the Settings Dialog, your settings are validated, and then saved automatically to a location within your user home folder, which is OS-dependent, e.g. “C:\Users\jsmith\AppData\Local\Monash University\MyData\MyData.cfg” or “/Users/jsmith/Library/Application Support/MyData/MyData.cfg”.

The settings file is in plain-text file whose format is described here: [https://docs.python.org/2/library/configparser.html](https://docs.python.org/2/library/configparser.html). An example can be found here: [MyDataDemo.cfg](https://docs.python.org/2/library/configparser.html).

Any facilities with potentially malicious users may wish to consider what happens if a user gets hold of an API key for a facility role account, saved in a MyData configuration file. The API key cannot be used in place of a password to log into MyTardis’s web interface, but it can be used with MyTardis’s RESTful API to gain facility manager privileges. These privileges would not include deleting data, but for a technically minded user familiar with RESTful APIs, the API key could potentially be used to modify another user’s data. Facilities need to decide whether this is an acceptable risk. Many facilities already use shared accounts on data-collection PCs, so the risk of one user modifying another user’s data subdirectory is already there.

Settings can be saved to an arbitrary location chosen by the user by clicking Control-s (Windows) or Command-s (Mac OS X) from MyData’s Settings dialog, keeping in mind the risks stated above. A saved settings file can then be dragged and dropped onto MyData’s settings dialog to import the settings. This feature is currently used primarily by MyData developers for testing different configurations. It is expected that the MyData settings for each individual instrument PC will remain constant once the initial configuration is done.

### 1.5 Upload Methods

MyData supports two methods for uploading data to MyTardis:

1. HTTP POST
2. SCP via Staging
“HTTP POST” (MyData’s default upload method) is automatically enabled as soon as you have entered some basic settings into MyData (see Settings and Downloading and installing the demo configuration for MyData). The “HTTP POST” method is easy to get up and running quickly for trying out MyData with small datasets. 

But for large datasets and large datafiles, the “SCP via Staging” method is preferred for the following reasons:

1. For large datafile uploads (multiple gigabytes), the “HTTP POST” method can put significant strain on the MyTardis server’s memory, affecting all users connecting to that MyTardis server.
2. Partially complete uploads can be resumed when using “SCP via Staging”, but not when using “HTTP POST”.
3. The “HTTP POST” method only allows one concurrent upload, because it uses the “poster” Python library, which uses “urllib2” which is not thread-safe.

1.5.1 Concurrent Upload Threads and Subprocesses launched by MyData

The maximum number of upload threads can be specified in the advanced tab of MyData’s Settings Dialog (see Advanced). This setting will have no effect when using the “HTTP POST” upload method, which has a maximum of one concurrent upload.

When using multiple upload threads, you won’t see multiple “MyData” processes running in your process monitor / task manager, but you will see multiple “scp” (secure copy) processes running which are launched from “MyData” as subprocesses. You may also see multiple “ssh” processes which are used to run remote commands on MyTardis’s staging server to determine the size of an incomplete upload on MyTardis’s staging server and to append an uploaded chunk to a partially uploaded datafile in MyTardis’s staging area.

While using MyData’s “SCP to Staging” upload method, you may also notice a “dd” subprocess running for each upload. “dd” is used to extract a chunk to upload from a datafile.

On Mac OS X, in addition to the brief “ssh” processes described above, you may also notice some lingering “ssh” processes (one per upload thread), which are used to set up a “ControlMaster” ssh process (see http://www.openbsd.org/cgi-bin/man.cgi?query=ssh_config), which allows MyData to reuse an existing SSH connection for appending additional datafile chunks to a partial upload.

OpenSSH’s ControlMaster/ControlPath functionality is not available in Windows builds of OpenSSH: http://stackoverflow.com/questions/20959792/is-ssh-controlmaster-with-cygwin-on-windows-actually-possible So we can’t use this method to reuse SSH connections for SCP-uploading subsequent datafile chunks on Windows. Out of necessity, MyData creates a new SSH (“SCP”) connection for every chunk uploaded, at least it does for large datafiles.

For small datafiles uploads on Windows, if the chunk size is too small, then calling “scp.exe” repeatedly will waste time reconnecting to the same MyTardis staging server repeatedly after only spending a fraction of a second actually uploading each chunk. If the chunk size is too large, then MyData won’t be able to display smooth progress updates. For small datafiles on Windows (less than 10 MB), MyData upload the entire datafile with one call to “scp.exe”, so you won’t see incremental progress updates in MyData’s Uploads view.

1.5.2 HTTP POST

MyData’s “HTTP POST” upload method uses the “Via multipart form POST” method of MyTardis’s RESTful API. For more details, see: https://mytardis.readthedocs.org/en/latest/api.html#via-multipart-form-post

1.5.3 SCP to Staging

MyData’s “SCP to Staging” upload method uses the “Via staging location” method of MyTardis’s RESTful API. For more details, see: https://mytardis.readthedocs.org/en/latest/api.html#via-staging-location
When using the “SCP to Staging” method, MyData informs MyTardis of its intention to upload a datafile (and registers the filename, size and checksum in a Datafile record in MyTardis), and MyTardis then supplies MyData with a temporary location to upload the datafile to. MyData will be granted access to upload the datafile to that temporary location using scp (secure copy). The server which MyData connects when uploading via scp (known as the “staging server”) may be the same as the MyTardis server, or it may be a different server which mounts the same storage as MyTardis. MyTardis runs scheduled background tasks to check for datafiles which have been registered but not yet verified, and for unverified datafiles which were to be uploaded via staging, MyTardis will copy the uploaded datafile from the staging area to its final destination (MyTardis’s file store) while checking its size and calculating its MD5 checksum to verify its integrity.

The first time a user runs MyData, they will see a warning indicating that MyData’s preferred upload method (SCP via staging) hasn’t yet been approved by the MyTardis administrator:

MyData uploads some basic information about the instrument PC and about the MyData installation to its MyTardis server. This basic information is called an “uploader” record. Once an uploader record has been created in MyTardis, no other users (of MyTardis’s RESTful API) will be able to access the uploader record unless they know its MAC address, a unique string associated with the MyData user’s active network interface (Ethernet or WiFi). A single MyData user could create multiple uploader records from each PC they run MyData on, one for each network interface on each PC.
The MyTardis administrator can approve the request in the Django Admin interface (after adding the public key to the appropriate /home/mydata/.ssh/authorized_keys file):
Below is a sample of a MyTardis administrator’s notes made (in the approval_comments field in MyTardis’s UploadRegistrationRequest model) when approving one of these upload requests:

Ran the following as root on the staging host (118.138.241.33):

```
$ adduser mydata
$ mkdir /home/mydata/.ssh
$ echo "ssh-rsa AAAAB3NzaC... MyData Key" > /home/mydata/.ssh/authorized_keys
$ chown -R mydata:mydata /home/mydata/.ssh/
$ chmod 700 /home/mydata/.ssh/
$ chmod 600 /home/mydata/.ssh/authorized_keys
$ usermod -a -G www-data mydata
```

N.B.: The test below was only possible because the MyData user submitting the request and the MyTardis administrator approving the request were the same person. Normally, the MyTardis administrator wouldn’t have access to the MyData user’s private key.

Because in this case, I had access to the private key generated by the MyData instance submitting the uploader registration request, I was able to test SSHing into the staging host from my MyData test machine using the SSH private key which MyData generated in ~/.ssh/:

```
$ ssh -i ~/.ssh/MyData mydata@118.138.241.33
[mydata@118.138.241.33 ~]$ groups
mydata mytardis
[mydata@118.138.241.33 ~]$ ls -lh /mnt/sonas/market | grep MYTARDIS
drwx------ 403 mytardis www-data 128K Nov 12 14:33 MYTARDIS_FILE_STORE
drwxrwx--- 3 mytardis www-data 32K Nov 13 15:36 MYTARDIS_STAGING
```

1.5. Upload Methods
Note the permissions above - being part of the “mytardis” group on this staging host allows the “mydata” user to write to the MYTARDIS_STAGING directory, but not to the MYTARDIS_FILE_STORE directory.

Once uploads to staging have been approved, MyData can manage multiple uploads at once (5 by default):

1.6 User Groups

Assigning access to datasets to user groups is an alternative to assigning access to individual users. A folder structure of “User Group/Instrument/Data Owner Full Name/Dataset” is supported for this purpose. As well as being used by MyData, this folder structure can be used to copy / sync data to a shared network drive. Each instrument PC will only have one instrument folder, being the name of that instrument, but when data from multiple instrument PCs is copied to the shared network drive, multiple instrument folders can appear alongside each other. The “Data Owner Full Name” folder is usually the name of the person who collected the data, but it is really just a way of grouping datasets into MyTardis experiments, i.e. it is not used to assign access control.

For more information, see the “Folder Structure - User Group / Instrument / Full Name / Dataset” section in http://mydata.readthedocs.org/en/latest/settings.html#advanced

1.6.1 Data Uploads from Instrument PCs

When using User Groups, the primary data directory used with MyData could look like this:
The first folder level within C:\UserGroups ("Group1", "Group2" etc.) is a user group defined in MyTardis. The actual group names in MyTardis may have an additional prefix ("TestFacility-") prepended to the folder name, i.e. "TestFacility-Group1", "TestFacility-Group2" etc.

The second folder level within C:\UserGroups ("Instrument 1") is the name of the data collection instrument. This folder may seem redundant, because all of the data on each instrument PC is by definition, on the same instrument PC (e.g. "Instrument 1"), but this folder level becomes useful when data from multiple instrument PCs is synced to a shared network drive.

The third folder level within C:\UserGroups ("G1Member1", "G1Member2", "G2Member1", "G2Member2" etc.) is usually the full name of the researcher who owns the data, but in some cases it is just an arbitrary collection of datasets. This corresponds to an experiment in MyTardis, which is a collection of datasets which can be made accessible to a particular user or to a group (e.g. "TestFacility-Group1").

The fourth folder level within C:\UserGroups ("Dataset001" etc.) is mapped to a MyTardis dataset.
The MyData screenshot shows the 8 datasets found within the C:\UserGroups directory on the “Instrument 1” PC. MyData counts the number of files within each dataset folder on the local disk, then counts the number of files previously uploaded to VicNode / MyTardis for that dataset (if any), and then uploads any datafiles which are not already available on VicNode / MyTardis.

Whilst MyData can recognize old data in a well-defined folder structure as described above, MyData is generally intended to be used to upload recently acquired data. An option to ignore old datasets (older than 6 months) has recently been implemented in MyData.

### 1.6.2 Data Management in MyTardis for Facility Managers

The first time MyData is run on a new instrument PC, some configuration is required - MyData’s Settings dialog is shown below. Typically a facility role account in MyTardis (“testfacility” in this case) is used to upload data. Once the data has been uploaded, access (and ownership) can be granted to individual users within MyTardis. In the case of the User Group folder structure, MyData will attempt to automatically grant read access to each dataset to all users within the data set’s user group (e.g. “TestFacility-Group1”).
The “testfacility” account in this MyTardis instance is associated with a facility record in MyTardis’s database, which means that MyTardis’s Facility Overview will be accessible when logged into MyTardis as “testfacility”. The Facility Overview lists recently uploaded datasets. The number of verified files in each dataset is the number of files which have been uploaded and confirmed to have the correct file size and MD5 checksum.
1.6.3 User and Group Management in MyTardis for Facility Managers

The “testfacility” account in this MyTardis instance is a group administrator for the “TestFacility-Group1” group, which means that they can view all members of that group by selecting Group Management from the drop-down menu available by clicking on the “testfacility” username in the upper-right corner of MyTardis.
Clicking on an experiment from the Facility Overview page (or from the My Data page or from the Home page) allows you to determine which users its datasets are accessible to. In this case, the “Instrument 1 - G1Member1” experiment is owned by “testfacility” and is accessible by the “TestFacility-Group1” group. Users can be granted access (or revoked access) using the Change User Sharing and Change Group Sharing buttons.

Researchers can log into MyTardis and view all experiments which their user group has access to. User “wettenhj” has access to the experiment “Instrument 1 - G1Member1” (below), because he is a member of the “TestFacility-Group1”
1.7 Mac OS X Walkthrough

MyData development is primarily targeting Windows, which is the OS of choice for most data-collection instrument PCs. This document aims to demonstrate that MyData can also run on Mac OS X.

On Mac OS X, after downloading and opening the disk image (DMG) file, drag the MyData application into your Applications folder and launch it. (You can then eject the disk image.)
If you are running a recent version of a Mac OS X (e.g. v10.10 Yosemite), you may find that your Mac OS X operating system doesn’t recognize MyData’s code-signing certificate and claims that MyData is from an “unidentified developer”:

Eventually MyData for Mac OS X will be built on a more recent version of Mac OS X (currently OS X 10.7.5), so this won’t be a problem. To understand this error message, open the Security & Privacy settings from your System Preferences application:
The default option is to “Allow applications downloaded from: Mac App Store and identified developers”. Because Mac OS X v10.10 doesn’t recognize MyData’s code-signing certificate, you will need to bypass this security check for MyData. Changing “Mac App Store and identified developers” to “Anywhere” would prevent this error, but it would also make your Mac less secure, so when bypassing Gatekeeper (Mac OS X’s security watchdog), it is better to do it one application at a time.

Right-click (or control-click) on MyData (instead of double-clicking with the left mouse button), and then click “Open” from the pop-up menu. You should see the follow dialog which indicates that Mac OS X can’t identify a valid code-signing certificate in the MyData app bundle, but it will let you open it anyway, if you click “Open” (instead of the default “Cancel” button):

A test MyTardis site is available for authorized MyData testers. Contact James Wettenhall at James.Wettenhall@monash.edu.au if you would like to register for testing MyData against this MyTardis test site or if you would like assistance with setting up an alternative test site for MyData. After registering as an authorized MyData tester, you will receive a MyTardis username and password. In my case, my MyTardis username is “wettenhj”.

Choose a folder where you would like to store your data. I chose “/Users/Shared/MyDataTest”:
Create a folder whose name matches your MyTardis username ("wettenhj" for me):

Put your data within your user folder, ensuring that all datafiles are grouped within folders, which will become datasets in MyTardis:

Launch MyData, and enter some basic settings in MyData’s Settings dialog (below). Each field within the Settings dialog is described here: http://mydata.readthedocs.org/en/latest/settings.html
1.7.1 Starting MyData’s Scan and Upload Processes

MyData will scan for data and attempt to upload it as soon as you click “OK” on the Settings dialog or whenever you press the refresh icon on MyData’s toolbar. If you launch MyData with a “–background” command-line argument, then it will automatically begin its scan and upload processes straight away, even without displaying its Settings dialog, assuming that valid settings have been entered previously.

1.7.2 MyData’s Upload Methods

MyData supports two upload methods - HTTP POST and SCP via staging.

The HTTP POST upload method is only intended to be used for quick demos and for small data files. Uploading large files with HTTP POST can put significant strain on the MyTardis server’s resources (particularly memory usage). The only advantage of HTTP POST is that it is easy to configure. As long as you have access to a suitable MyTardis role account (e.g. “testfacility”) and know its API key, then you can begin uploading from MyData straight away, although you should begin by testing small files only.

SCP via staging is MyData’s preferred upload method. MyData will automatically use this method as soon as it become available, but uploads via staging need to be approved by a MyTardis administrator. MyData generates an SSH key pair the first time it runs and sends the public key to the MyTardis server in a request for the ability to upload via staging. The MyTardis administrator needs to approve the request and put the public key in a suitable authorized keys file on the staging server (which could be the same as the MyTardis server). For example, the public key could be put in “/home/mydata/.ssh/authorized_keys” on the staging server.
The first time a user runs MyData, they will see a warning indicating that MyData’s preferred upload method (SCP via staging) hasn’t yet been approved by the MyTardis administrator:

![Warning Message]

The MyTardis administrator can approve the request in the Django Admin interface (after adding the public key to the appropriate /home/mydata/ssh/authorized_keys file):
Once uploads to staging have been approved, MyData can manage multiple uploads at once (5 by default):

By clicking on the web browser icon on MyData’s toolbar, you can view the uploaded data in MyTardis in your web browser. The data will be jointly owned by the facility role account (e.g. “testfacility”) and by the MyTardis user whose username (e.g. “wettenhj”) was used to name the folder containing the datasets. MyTardis allows grouping datasets together into experiments. MyData uses the instrument name (e.g. “James Mac Laptop”) and the date of creation of the dataset folders (e.g. “2014-12-18”) to define a default experiment for the datasets it uploads:
If you are authorized to log into MyTardis’s web interface as a facility manager, you can view the data in MyTardis’s new Facility Overview. Note the two owners - the facility role account ("testfacility") and the individual user ("wettenhj") who collected the data:
1.8 MyData Tutorial, using the MyTardis Demo Server

1.8.1 Downloading and Installing the MyTardis Demo Server

- Download the MyTardis Demo Server for Windows:
  
  - MyTardisDemoServer_d82f585c35c187cac0110adc13d29ff1a963dab2.exe

After downloading the MyTardis Demo Server, open the downloaded executable to begin the setup wizard, which shows the version of MyTardis being installed (from the https://github.com/wettenhj/mytardis/tree/mydata branch):
The default installation location is C:\MyTardisDemoServer:
1.8.2 Launching the MyTardis Demo Server

After installing the MyTardis Demo Server (which could take 5-10 minutes), start the MyTardis server by clicking the “MyTardis Demo Server” shortcut in the Start Menu (or in the Start Screen if you don’t have a Start Menu):

The MyTardis Demo Server will display a Command Prompt window starting in the installed directory (defaulting to C:\MyTardisDemoServer), and then navigating into the specific MyTardis version directory (e.g. C:\MyTardisDemoServer\d82f585c35c187cac0110adc13d29f1a963dab2) and then running the demo server (using “python mytardis.py runserver”). The MyTardis Demo Server bundles its own version of Python and puts it in the PATH before any other Python version you may have installed while running this Command Prompt window (see C:\MyTardisDemoServer\MyTardisDemoServer.bat).

1.8.3 Accessing the MyTardis Demo Server in your web browser

Once the MyTardis Demo Server has started, you can access its web interface by navigating to the following address: http://127.0.0.1:8000/ in your web browser (Google Chrome is a good choice).
1.8.4 Logging into the MyTardis Demo Server as a MyTardis administrator

Click the “Log In” button in the upper right corner, and log in with username “mytardis” and password “mytardis”.

Accessing MyTardis’s Django Admin Interface

The “mytardis” account in this demo server is a super administrator, i.e. it can do anything, including accessing MyTardis’s Django Admin interface from the menu item shown below.
MyTardis’s Django Admin interface looks similar to many other Django applications’ admin interfaces. Keep in mind that this interface is extremely powerful, so if you are not careful, you could delete database records without any way to recover them!

Facilities Registered in MyTardis

From the Django Admin interface, click on “Facilities” to see what facilities are available in this Demo Server. There is only one facility, named “Test Facility”.

![Facilities Registered in MyTardis](image)
Click on the “Test Facility” facility record to see the properties of the facility, including the “Test Facility Managers” user group assigned to the “Manager group” field of the facility record.

User Accounts in MyTardis

From the Django Admin interface, click on Users to see the user accounts available in this MyTardis server. The “mytardis” administrator is the only account which can access the Django Admin interface.

Click on the “testfacility” user account to see its attributes. Note that this account is a member of the “Test Facility” facility record’s manager group, named “Test Facility Managers”.

1.8. MyData Tutorial, using the MyTardis Demo Server
1.8.5 Logging into the MyTardis Demo Server as a MyTardis facility manager

Log out of the Django Admin interface, and then return to the original URL in your web browser’s address bar, i.e. http://127.0.0.1:8000/, not http://127.0.0.1:8000/admin/. Then log in with username “testfacility” and password “testfacility”, and click on the “Facility Overview” section link in the navigation bar at the top of the MyTardis home page. Since we haven’t uploaded any data yet, no data will appear in the Facility Overview, but we can confirm that the “testfacility” account has access to the Facility Overview for the “Test Facility” facility.
1.8.6 Obtaining the demo data

Download MyTardisDemoData.zip in extract it in “C:" to create the “C:\MyTardisDemoUsers” folder shown below:

1.8.7 Launching MyData

MyData can be downloaded from here: http://mydata.readthedocs.org/en/latest/download.html

Open the downloaded executable and proceed through the setup wizard to install MyData. A shortcut to MyData will then be available in the Start Menu (or the Start Screen if not using a Start Menu):
When you launch MyData interactively (which is the default, unless you give MyData.exe a command-line argument of ‘--background’), its settings dialog will appear automatically. The first time you launch MyData, its settings will be blank:

![MyData Settings Dialog](image)

1.8.8 Downloading and installing the demo configuration for MyData

Download MyDataDemo.cfg onto your Desktop and drag and drop it onto MyData’s settings dialog, which should automatically populate the fields in MyData’s settings dialog.
The Advanced tab of MyData’s settings dialog contains additional settings:
1.8.9 MyData's Settings Validation

After clicking “OK” on the settings dialog, MyData will validate the settings and inform the user of any problems it finds. When running in interactive mode, MyData will then inform the user of how many datasets it has counted within the data directory and ask the user to confirm that they want to continue.

1.8.10 MyData’s Upload Methods

MyData offers two upload methods:

- HTTP POST
• SCP to Staging

The second method (“SCP to Staging”) can handle much larger datafiles and supports multiple concurrent upload threads, however it is slightly more complicated to set up, so we won’t be covering it in this tutorial. Instead, we will stick with MyData’s default upload method (“HTTP POST”) and ignore the warning dialog below.

1.8.11 MyData’s Folders View

MyData’s Folders view lists all of the dataset folders which will be scanned for files to upload to MyTardis. For each folder, MyData displays a count of the total number of files in that folder, and the number of files which have already been uploaded to MyTardis. MyData is stateless, i.e. it won’t remember how many files were confirmed to be on MyTardis last time it was run, so each count will begin at zero and then increment by one as each file is confirmed to be available on MyTardis.

1.8.12 MyData’s Users View

MyData’s Users view (below) displays the result of MyData’s attempt to map the user folder names (“testuser1” and “testuser2”) to MyTardis user accounts. In this case, both user folder names have been successfully mapped to user accounts on our MyTardis Demo Server, but no email address has been recorded for either account in MyTardis. Many queries MyData performs against MyTardis will only work if the MyTardis account you entered in MyData’s settings dialog (“testfacility”) has sufficient permissions assigned to it, as shown on the Django Admin’s user account attributes page for the “testfacility” account. In this case, the “testfacility” account can access other users’ email addresses because it is a member of a Facility Managers group in MyTardis.
1.8.13 MyData’s Verifications View

MyData’s Verifications view (below) shows MyData’s attempts to verify whether each datafile is available on the MyTardis server, or whether it needs to be uploaded.

1.8.14 MyData’s Uploads View

MyData’s Uploads view (below) shows MyData’s upload progress. The default HTTP POST method only supports one concurrent upload, whereas the “SCP to Staging” upload method supports multiple concurrent uploads.

1.8.15 Monitoring MyData Uploads in MyTardis’s Facility Overview

After some of the datafiles have completed uploading, you can check back in your web browser to see the datafiles in MyTardis’s Facility Overview (below). You should be logged into MyTardis as the “testfacility” account (username “testfacility”, password “testfacility”).

---

**MyData Documentation, Release 0.2.0-alpha2**

**1.8.13** MyData’s Verifications View

MyData’s Verifications view (below) shows MyData’s attempts to verify whether each datafile is available on the MyTardis server, or whether it needs to be uploaded.

**1.8.14** MyData’s Uploads View

MyData’s Uploads view (below) shows MyData’s upload progress. The default HTTP POST method only supports one concurrent upload, whereas the “SCP to Staging” upload method supports multiple concurrent uploads.

**1.8.15** Monitoring MyData Uploads in MyTardis’s Facility Overview

After some of the datafiles have completed uploading, you can check back in your web browser to see the datafiles in MyTardis’s Facility Overview (below). You should be logged into MyTardis as the “testfacility” account (username “testfacility”, password “testfacility”).

---

**Chapter 1. Contents**
For the demo server, we are using the `CELERY_ALWAYS_EAGER` setting which means that datafiles will be verified immediately, instead of as a background task. This explains why the number of verified datafiles below is always equal to the total number of datafiles for each dataset. In the screenshot below, only 6 datafiles have been uploaded from the “Amorphophallus Titanum SEM” dataset, and no datafiles have been uploaded from the other datasets yet.

1.8.16 MyTardis’s “My Data” View from a Facility Manager’s Perspective

While logged in as “testfacility” (an account whose credentials could be shared amongst the managers of “Test Facility”), click on “My Data” to see all of the “experiments” (dataset collections) created by MyData while running at that facility. MyData’s default dataset grouping uses the instrument name (“Test Microscope”) and the user’s full name (e.g. “Test User1”) to define a MyTardis “experiment” record, as seen in MyTardis’s “My Data” view below.
1.8.17 MyTardis from a Facility User’s Perspective

Log out of MyTardis, and log back in with the username “testuser1” and password “testuser1”. Now you only see the data collected by user “testuser1”, not the data collected by “testuser2”. The “Test User1” in the experiment (dataset group) names may seem redundant here, but users can share their experiments with other users, so it would be confusing if all of the shared experiments were just given a default name of “Test Microscope”.

Click on the “Test Microscope - Test User1” experiment to see the datasets included in that experiment:
Click on the “Amorphophallus Titanum SEM Dataset” to see the datafiles in that dataset:
1.9 MyData Client and Daemon

Here’s what the MyData daemon looks like when we first run it:

```
$ python MyData.py --daemon
MyData Daemon 0.2.1 (PID 4428)
Connected to MemCache Daemon (PID 5056).
Waiting for MyData client(s) to connect...
```

And here’s what the MyData client looks like when we first run it:

```
$ python MyData.py --client
MyData Client 0.2.1 (PID 5836)
Connected to MemCache Daemon (PID 5056).
Connected to MyData Daemon (PID 4428).
```

TO DO: Client should be able to retrieve already-running state from daemon (e.g. ‘displaying 10 uploads in progress’), rather than just retrieving incremental updates (‘AddRow’) from the daemon.

Plus the client would be displaying the standard MyData GUI at that point. After pressing OK on the MyData client’s settings dialog, instead of running the folder scans and uploads in the client process, the client sends a job request to the daemon to perform the folder scans and uploads. The job request can be see in the daemon’s log (~/.MyData_daemon_debug_log.txt) below:
With a shared memory system like Memcached, there is a risk of concurrency errors if you are not careful. However, the "incr()" and "decr()" methods provided are thread-safe, so we use "incr()" to increment the job ID, and then create a key/value pair in Memcached with "job_<ID>" i.e. "job_1" as the key. The maximum job ID requested by the MyData client can be stored in Memcached and safely incremented using "incr()". And similarly, the maximum job ID which has already been handled by the daemon can too be safely incremented using "incr()" method in Memcached.

Once the MyData daemon begins running the job, it starts to report events back to Memcached, which can be handled by the client:

2015-04-14 22:22:05,598 - Daemon.py - 139 - run - MyData - DEBUG - event_1: {'eventType': 'SetStatusMessage', 'message': 'Validating settings...'}

In the case above, the daemon is reporting to the client that it is currently validating the settings provided, (which may have already been validated by the client), and that the status bar message needs to be updated to reflect this.

If we now look in the MyData client log (~/.MyData_client_debug_log.txt), we can see this “SetStatusMessage” event being handled by the MyData client:


If all goes well, events will continue to propagate from the daemon to the client and the user will be able to observe the progress almost as quickly as if the client were running the tasks itself.

1.10 License

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for
software and other kinds of works.

The licenses for most software and other practical works are designed
to take away your freedom to share and change the works. By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program—to make sure it remains free
software for all its users. We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors. You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights. Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers’ and authors’ protection, the GPL clearly explains that there is no warranty for this free software. For both users’ and authors’ sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users’ freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based
on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.


The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users
can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.


No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work’s users, your or third parties’ legal rights to forbid circumvention of technological measures.


You may convey verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey,
and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

   c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

   d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

   a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

   b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical
medium customarily used for software interchange, for a price no
more than your reasonable cost of physically performing this
conveying of source, or (2) access to copy the
Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the
written offer to provide the Corresponding Source. This
alternative is allowed only occasionally and noncommercially, and
only if you received the object code with such an offer, in accord
with subsection 6b.

d) Convey the object code by offering access from a designated
place (gratis or for a charge), and offer equivalent access to the
Corresponding Source in the same way through the same place at no
further charge. You need not require recipients to copy the
Corresponding Source along with the object code. If the place to
copy the object code is a network server, the Corresponding Source
may be on a different server (operated by you or a third party)
that supports equivalent copying facilities, provided you maintain
clear directions next to the object code saying where to find the
Corresponding Source. Regardless of what server hosts the
Corresponding Source, you remain obligated to ensure that it is
available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided
you inform other peers where the object code and Corresponding
Source of the work are being offered to the general public at no
charge under subsection 6d.

A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling. In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage. For a particular
product received by a particular user, "normally used" refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product. A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

"Installation Information" for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source. The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.

If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some
trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.


You are not required to accept this License in order to receive or
run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s "contributor version".

A contributor’s "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a
party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.


If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.
Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

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 3 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, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.
Indices and tables

• genindex
• modindex
• search