Ralph Assets Documentation

Release 2.5.1

Allegro Group

January 25, 2016

Contents

1 Quickstart **3**
- 1.1 Assets . . . 3
- 1.2 Licences . . . 8
- 1.3 Supports . . . 10
- 1.4 Users . . . 12
- 1.5 Admin . . . 12

2 Installation **13**

3 Configuration **15**
- 3.1 Optional features . . . 15

4 Transitions **17**
- 4.1 Configuration . . . 17

5 History module **21**
- 5.1 Typical usage . . . 21
- 5.2 API . . . 21

6 Regions **23**
- 6.1 How to use it? . . . 23

7 Signals **27**
- 7.1 Transition signals . . . 27
- 7.2 Form signals . . . 27

8 Change Log **29**
- 8.1 DEV . . . 29
- 8.2 2.5.1 . . . 29
- 8.3 2.5.0 . . . 29
- 8.4 2.3.0 . . . 30
- 8.5 2.2.0 . . . 31
- 8.6 2.1.0 . . . 31
- 8.7 2.0.2-rc3 . . . 31
- 8.8 2.0.2-rc2 . . . 32
- 8.9 2.0.1-rc2 . . . 32
- 8.10 2.0.0-rc2 . . . 32
- 8.11 2.0.0-rc1 . . . 32
- 8.12 1.4.3 . . . 32

8.13 1.4.2 . . . 33
8.14 1.4.1 . . . 33
8.15 1.4.0 . . . 33
8.16 1.3.2 . . . 33
8.17 1.3.1 . . . 33
8.18 1.3.0 . . . 33
8.19 1.2.13 . . . 34
8.20 1.2.12 . . . 34
8.21 1.2.11 . . . 34
8.22 1.2.10 . . . 34
8.23 1.2.9 . . . 34
8.24 1.2.7 . . . 34
8.25 1.2.6 . . . 35
8.26 1.0.0 . . . 35

Python Module Index **37**

Ralph Assets module brings offline stock management functionality to Ralph. It maintains placing orders, purchasing and releasing of devices to the users. It is not limited to the data center. You can maintain inventory of your back office assets with it as well. In this module, you can also manage licenses and supports.

Ralph is an asset management system for your data center. It lets you see what hardware and software is installed. It also lets you keep track of who is using every device, for what, and how much it costs.

Contents:

CHAPTER 1

Quickstart

The Ralph Assets module enables management of assets, licences and supports. Any objects in this modules allows manage relation beetween them.

1.1 Assets

1.1.1 Searching and Filtering of Assets

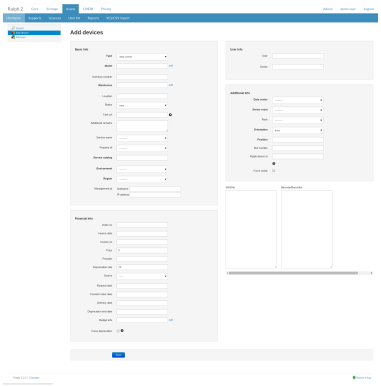
Let's start with the main screen. Here you find all your hardware assets which are in the database. Use the left column for filtering of the assets.

There are two types of assets – devices and parts. A device could be a blade server, and a part is a component of this server, for example memory or hard disk drive. A part can be assigned to a single device at a time. You can move parts from one device to another when you need it.

1.1.2 Adding Assets

data center

If you want add data center asset you need to fill a few required fields. Look at the screen below (you can zoom-in the image in your browser if you want) The fields in bold are required. Rest of fields are optional.



Let's describe each of the fields:

- **Basic info:**
  - **Type** - a read only field for data center

  - **Model** - choose a model for asset, (you can type a couple of letters to search for a given model. If no result is found, just click "Add" button to add it)
  - **Inventory number** - an alphanumeric value in your stock
  - **Warehouse** - the place where the asset is located
  - **Location** - a more exact location of the device in the building/room
  - **Status** - an asset's lifetime indicator. Newly bought assets has status "new". You can change it as required according to your own work flow
  - **Task url** - url to task in ticket system
  - **Additional remarks** - field for various data
  - **Service name** - service name to which this asset belongs
  - **Property of** - company to which this asset belongs
  - **Hostname** - read-only field telling the name of host (from Ralph-core device)
  - **Service catalog** - points to `service catalog` which asset belongs to
  - **Environment** - points to environment which asset belongs to
  - **Region** - set `region` for this asset (see `region` page for details)
- **Financial Info:**
  - **Order number** - number of the order where this asset is included
  - **Invoice date** - date of the invoice where this asset is included
  - **Invoice number** - number of the invoice where this asset is included
  - **Price** - unit price of this asset
  - **Provider** - name of the provider of this asset
  - **Depreciation rate** - number of months in which asset depreciates
  - **Source** - asset was purchased or salvaged
  - **Request date** - date of submission of the demand for this asset
  - **Provider order date** - date when order of this asset was provided
  - **Delivery date** - date of asset delivery
  - **Depreciation end date** - the end day of the depreciation
  - **Budget info** - name of budget which funds this asset
  - **Force depreciation** - force depreciation for this asset despite of `depreciation end date` value
- **User info:**
  - **User** - points to user of this asset
  - **Owner** - points to a owner of this asset
- **Aditional info:**
  - **Data center** - points to a data center which contains this asset
  - **Server room** - points to a server room which contains this asset
  - **Rack** - points to a rack which contains this asset
  - **Position** - points to position in rack from 0 to 'max u'

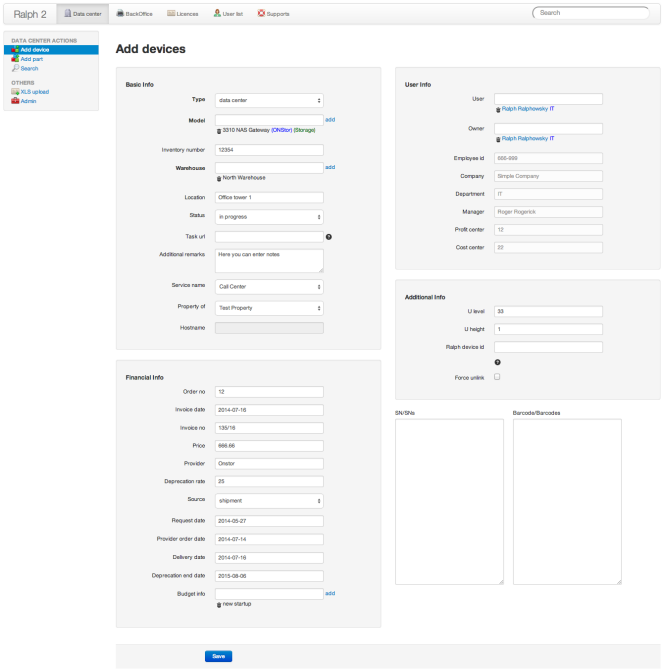
- **Orientation** - a side of rack, depends on `position`

  if `position` = 0 options are: *left*, *right*

  if `position` > 0 options are: *front*, *back*, *middle*
- **Slot number** - points to slot number in *blade* when asset model is *blade*
- **Ralph device id** - points to to Ralph-Core device
- **Force unlink** - if picked and "`ralph device id`" is already linked to other asset, this option forces to unlink it and link this asset to newly unlinked device
- **Create stock device** - create a Ralph-core device

back office

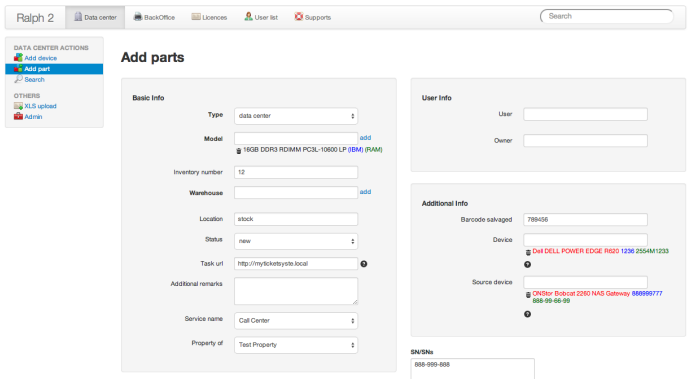
Now, let's add some devices and parts. Click the "Add device" option from the top of submenu.



The serial number or barcode field is required for assets. You can paste serial numbers and barcodes in series, thus allowing you to batch-add many devices of the same type.

1.1.3 Adding Parts

In the same way you can add parts to the database, and then bind the parts to devices. To do this, choose "Add part" from the menu.



- When a part is marked as `salvaged`, you can enter the old barcode data here.

1.1.4 Fields

Asset fields has been split into sections in forms:

- **Basic info:**
  - **Type** - a read only field for data center, back office or administration for back offic. Administration is used for assets like buildings etc.
  - **Model** - type a couple of letters to search for a given model. If no result is found, just click "Add" button to add it.
  - **Inventory number** -
  - **Warehouse** - the place where the asset is located.
  - **Location** - a more exact location of the device in the building/room.
  - **Status** - an asset's lifetime indicator. Newly bought assets has status "new". You can change it as required according to your own work flow.
  - **Task url** - url to task in ticket system
  - **Additional remarks** - additional info.
  - **Service name** - service name to which the asset belongs
  - **Property of** - to which the company belongs asset
- **Financial Info:**
  - **Price** - the unit price of the asset.
  - **Provider** - the name of the provider of the asset.
  - **Depreciation rate** - number of months this device deprecates
  - **Source** - if this device was purchased or salvaged
  - **Request date** - date of submission of the demand for the device.
  - **Delivery date** - date of device delivery.
  - **Deprecation end date** - the end day of the depreciation
  - **Order number**, **Invoice date**, **Invoice no**, **Provider order date**, **Budget info**.
- **User info:**
  - **User** - device user.

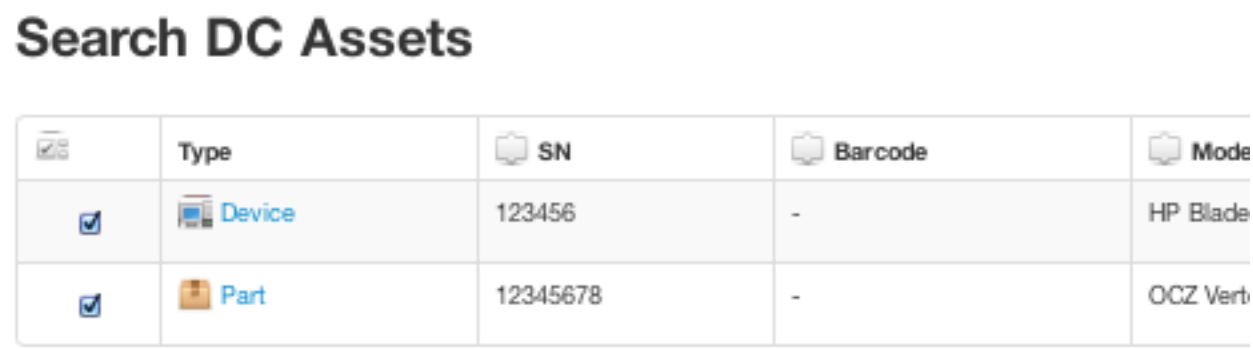
  - **Owner** - device owner.

- **Aditional info:**
  - **U level** - "U" level of installation device.
  - **U height** - how large the device is, in "U".
  - **Ralph device id** - ID device detected by Ralph Scan.

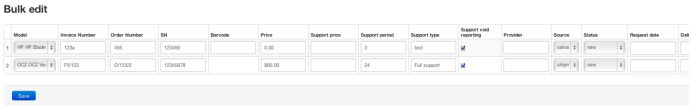
1.1.5 Bulk Editing

It is often required to edit multiple assets at once. For example, when you want to move them from one warehouse to another. There is a special mode called "bulk edit" for this case.

To activate this mode, go to the search screen, and select multiple assets using check marks on the left side.

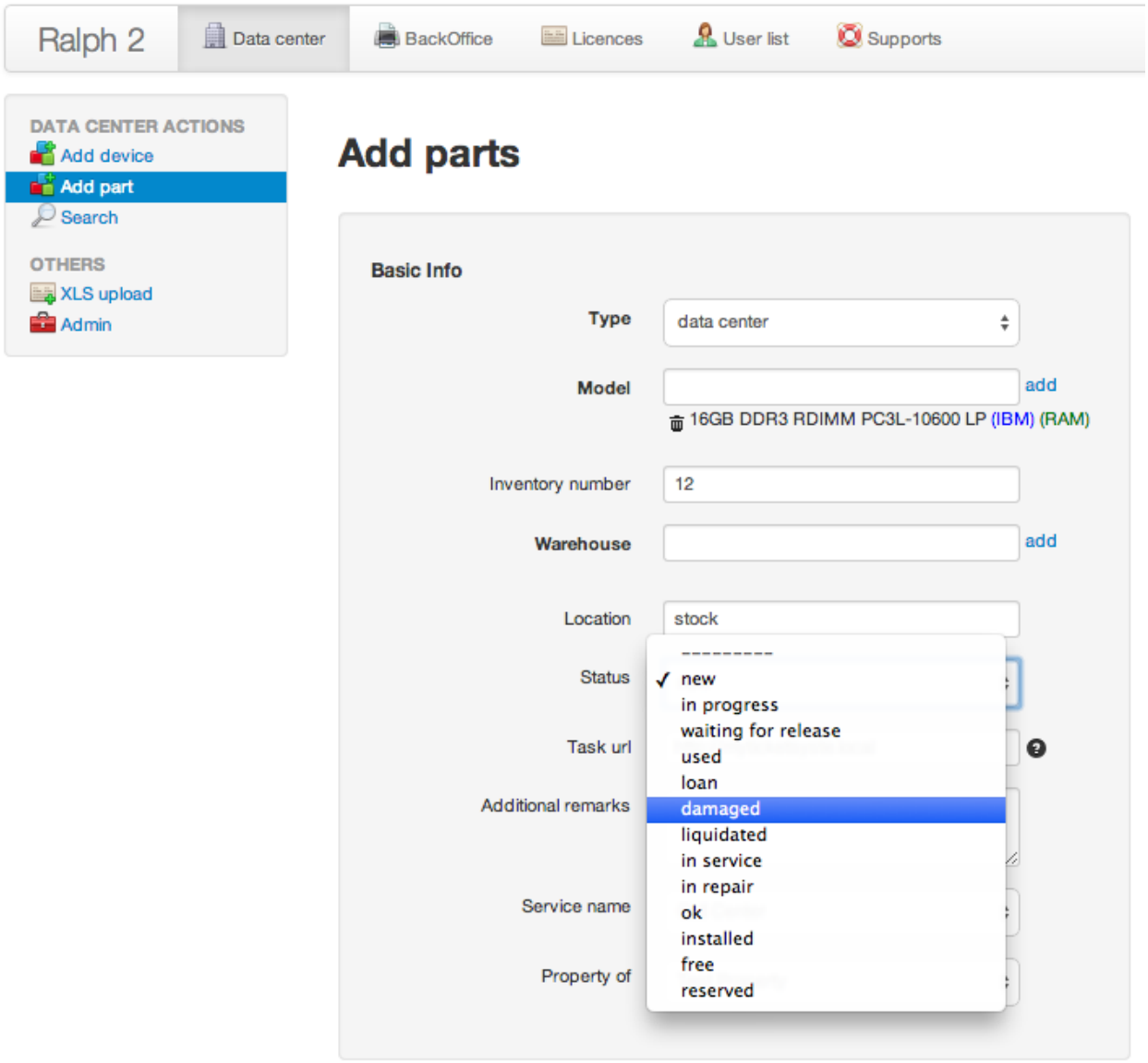


When ready, choose "Edit selected" from the bulk edit actions.



On the next screen you can edit those records all at once by changing the appropriate fields. When you fill one field with the desired value, you can propagate this value to all records by clicking on the "plus" mark near the current cell.

1.1.6 Work Flow and Statues



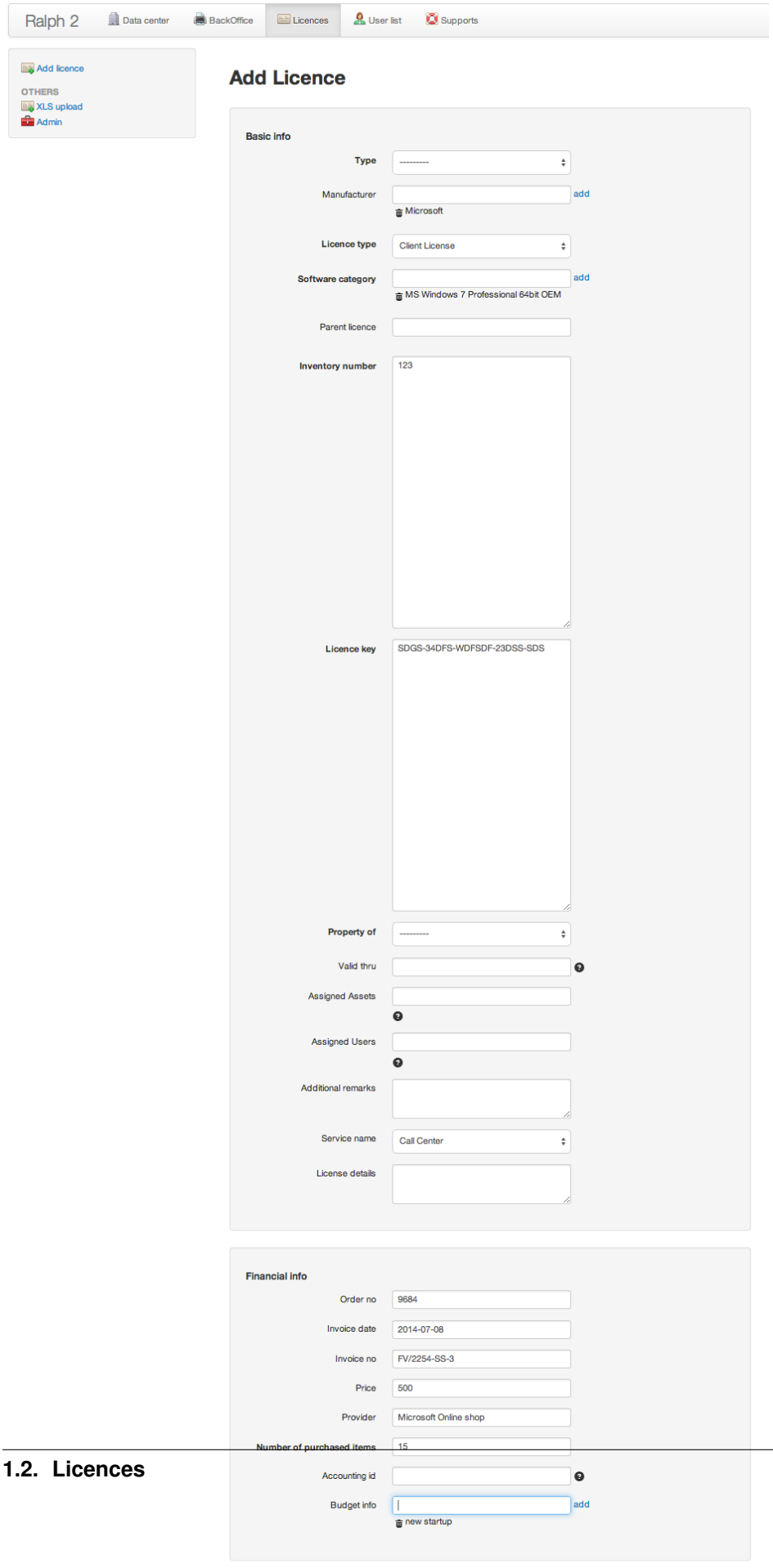
In this version there are no limits for moving assets from one status to another. You can freely change statuses. All changes will be recorded, allowing you to inspect the flow later.

1.2 Licences

Ralph Assets allows you to store information about software licenses. Adding and editing is performed in much the same way as in assets.

1.2.1 Adding License

To add a license, click the "Add support" option from the top of submenu.


Ralph 2
Data center
BackOffice
Licences
User list
Supports
Add licence
OTHERS
XLS upload
Admin
Add Licence
Basic info
Type

Manufacturer
add
Microsoft
Licence type
Client License
Software category
add
MS Windows 7 Professional 64bit OEM
Parent licence
Inventory number
123
Licence key
SDGS-34DFS-WDFSDF-23DSS-SDS
Property of

Valid thru
Assigned Assets
Assigned Users
Additional remarks
Service name
Call Center
License details
Financial info
Order no
9684
Invoice date
2014-07-08
Invoice no
FV/2254-SS-3
Price
500
Provider
Microsoft Online shop
Number of purchased items
15
Accounting id
Budget info
add
new startup

1.2.2 Fields

Licence fields are split into 2 section: **Basic info** and **Financial info**. **Financial info** contains very important field, **Number of purchased items**. This field ability to store Multi-Seat licence.

1.2.3 Relations

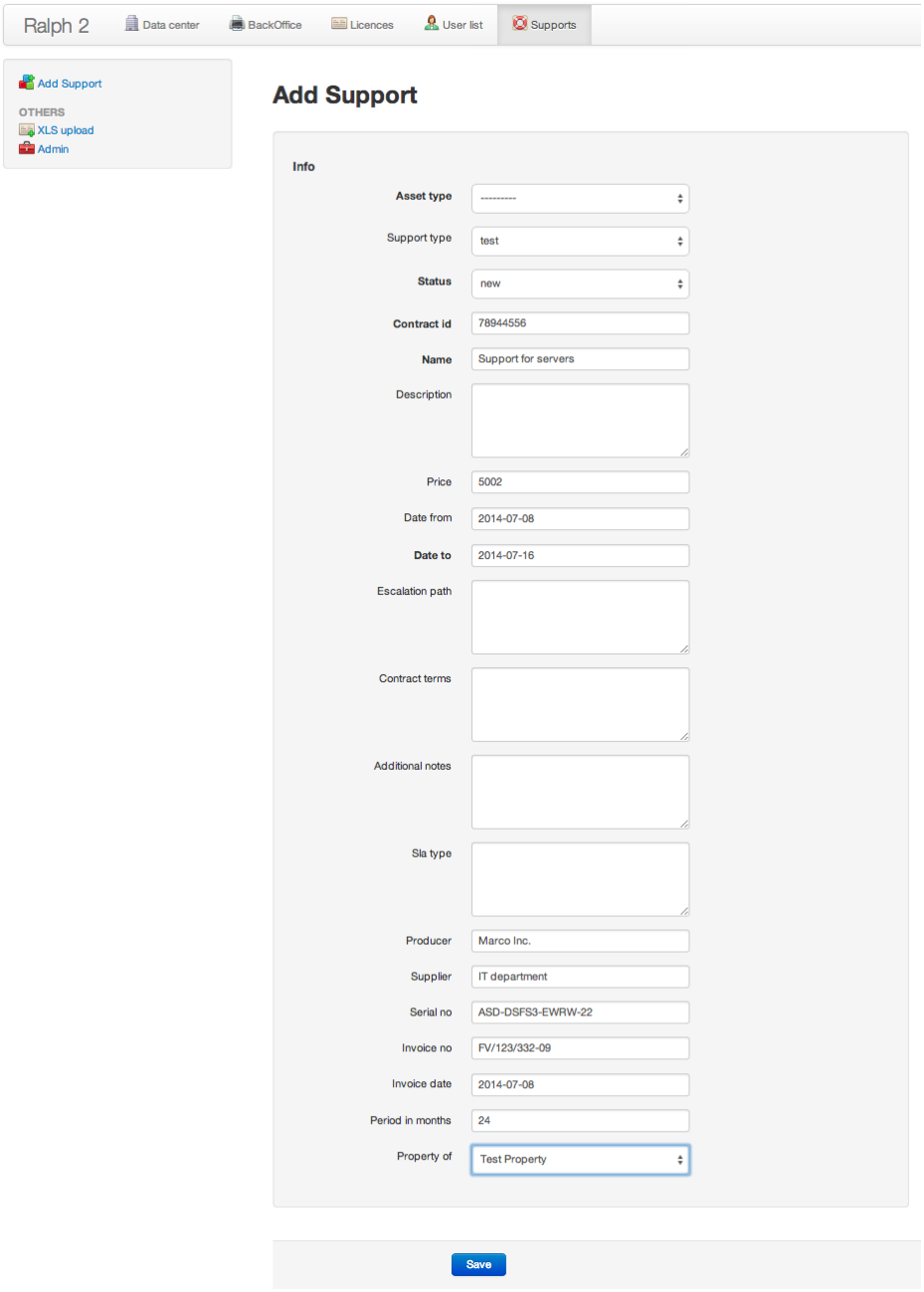
Licenses may be related to the relationship with the user or device. In asset and user form, during the search are shown only unassigned license, that is, those that have still free slots.

1.3 Supports

Ralph Assets allows you to store information about supports. Adding and editing is performed in much the same way as in assets.

1.3.1 Adding Support

To add a support, click the "Add support" option from the top of submenu.



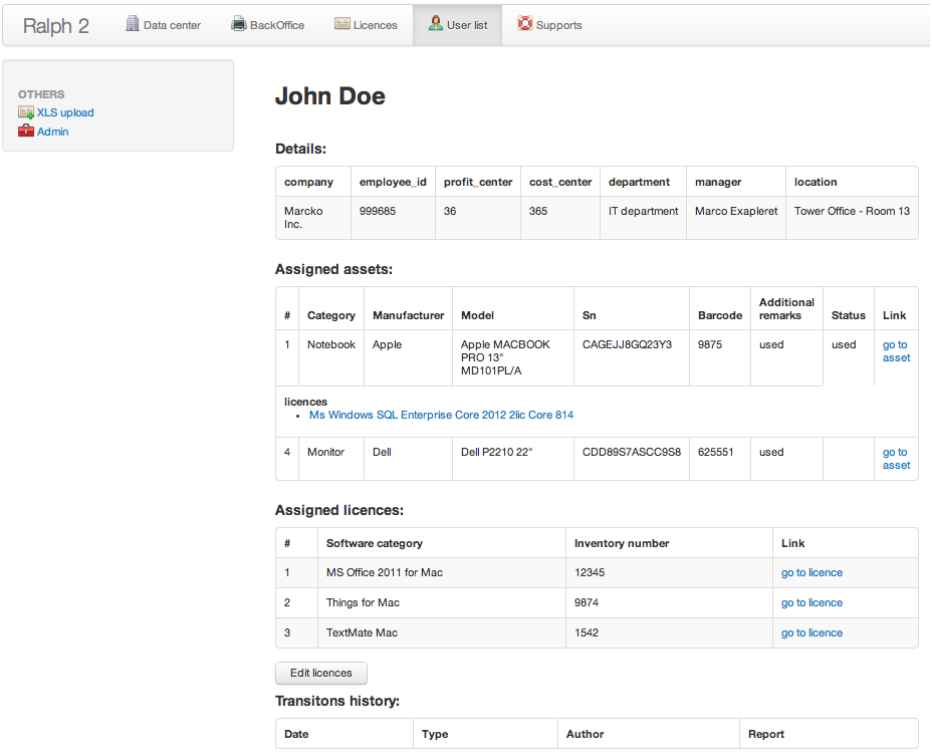
1.3.2 Relations

Support can be assigned to a device. On the asset form page, there is the option of marking device that requires a support. This is valuable information that helps you better manage supports.

1.4 Users

1.4.1 User Page

User page contains all information about user. User's devices, licenses, personal information and transition history.



1.5 Admin

Administration interface is accessible from within the menu.

Here you can define

- models,
- categories,
- warehouses,
- other dictionary data.

CHAPTER 2

Installation

Installation requirements:

1. Install Ralph .

Asset installation:

1. Install the `ralph_assets` package from PyPi by running:

```
pip install ralph_assets
```

2. After installation add a line in settings

```
PLUGGABLE_APPS = ['assets',]
```

3. Run:

```
ralph migrate ralph_assets
```

That's it. Now just run Ralph as described in its documentation, and login to the Ralph system. You will see an additional item, "Assets" in the main menu.

CHAPTER 3

Configuration

3.1 Optional features

3.1.1 Autogenerated domain hostname (disabled)

One optional information about *assets* is a *domain hostname*. This information can be autogenerated by **Ralph Assets**. To achieve that you need to set a few settings (in the Django's standard way), like:

```
ASSETS_AUTO_ASSIGN_HOSTNAME = True
```

This action cause that all assets:

- edited in *Edit form* or *Bulk form*,
- having set *Asset Owner* & *Asset Model* & *Asset model category code* (according to: *Asset.can_generate_hostname* property),
- and changed *Asset status* from any (except *in progress*) status to *in progress*,

will have autogenerated *hostname domain*.

Autogenerated format is described by this setting variable *ASSET_HOSTNAME_TEMPLATE*:

Default value is

```
ASSET_HOSTNAME_TEMPLATE = {
    'prefix': '{{ object.country_code|upper }}'
            '{{ object.model.category.code|upper }}',
    'postfix': '',
    'counter_length': 5,
}
```

where `prefix` and `postfix` options takes template string (rendered by Django's template engine so you can use standard (or custom) template tags and filters). Template context contains `object` variable which is an *asset instance*. The `counter_length` variable describes constant length of counter.

There is also another related option:

```
HOSTNAME_FIELD_HELP_TIP = 'Autogenerated if owner & model are set.'
```

This setting shows help tip next to *hostname field* in form.

Transition configurations is described *here*

CHAPTER 4

Transitions

Give the opportunity to take advantage of the transition, which facilitating multi-changes in assets including generation report file.

4.1 Configuration

Transition is disabled by default. To enable it, set settings as follow:

```
ASSETS_TRANSITIONS['ENABLE'] = True
```

Defining your own transition requires adding transition object to the database. Actually we support following transitions: `RELEASE-ASSET`, `LOAN-ASSET`, `RETURN-ASSET`, `CHANGE-HOSTNAME`.

Each transition has default slug defined in settings. You don't have to change anything in settings and use predefined slugs in transition definition objects.

To change slugs update settings variable eg.:

```
ASSETS_TRANSITIONS['SLUGS']['RELEASE'] = "your-custom-slug"
```

Default slugs:

- `release-asset` - for `RELEASE-ASSET` transition
- `loan-asset` - for `LOAN-ASSET` transition
- `return-asset` - for `RETURN-ASSET` transition
- `change-chostname` - for `CHANGE-HOSTNAME` transition

Actions available in transitions:

- `assign_loan_end_date` - fill loan end date in form.
- `assign_owner` - assign new user into assets.
- `assign_user` - assign new owner into assets.
- `assign_warehouse` - assign new warehouse into assets.
- `change_status` - change status into defined in `to_status` Transition field.
- `change_hostname` - change hostname with selected country code.
- `release_report` - generate release report file.
- `return_report` - generate return report file.

- `unassign_licences` - remove all licences assigned into assets.
- `unassign_loan_end_date` - clear loan end date field in assets.
- `unassign_owner` - remove owner assigned into assets.
- `unassign_user` - remove user assigned into assets.

4.1.1 Reports

To generate reports files, report template should be uploaded into 'Report odt source' model. Created model's slug should be specified. or Created model should have specified slug. And configure INKPY module.

Slug definition per report may be overridden in settings file eg.:

```
ASSETS_REPORTS['RELEASE-ASSET']['SLUG'] = 'your-slug'
```

You can use predefined slugs:

- `release-asset` - for release asset transition
- `loan-asset` - for loan asset transition
- `return-asset` - for return asset transition
- `change-hostname` - for manually change hostname country

4.1.2 Reports locale

Reports generation uses Django's *LANGUAGE_CODE* setting, however there is an option to changed that. You can force locale only for reports by setting *GENERATED_DOCS_LOCALE* in django's *settings*, eg:

GENERATED_DOCS_LOCALE = 'pl'

If so, all generated reports will have polish locale.

4.1.3 Multilanguage support

Note: Multilanguage support introduced with version 2.5.0 of ralph_assets.

For transition:

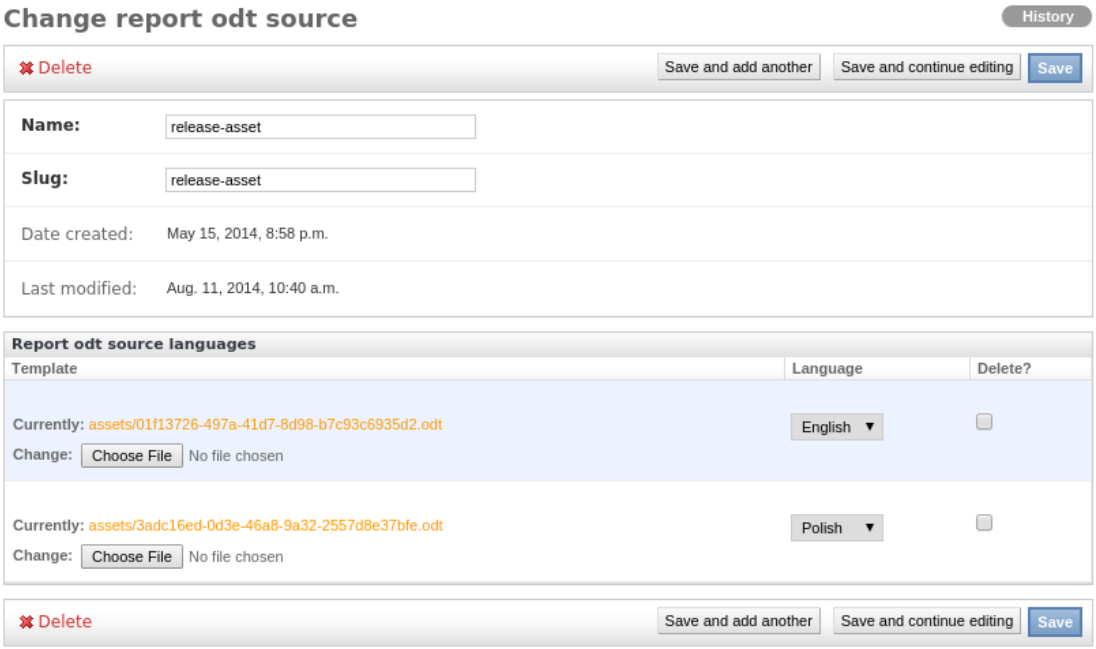
- `release-asset`
- `loan-asset`
- `return-asset`

you can attach diffrent template for configured language. Configuration is very simple, see the structure below:

```
REPORT_LANGUAGES = {
    'choices': (
        ('en', 'English'),
        ('pl', 'Polish'),
    ),
    'default': 'en',
}
```

Values `choices` and `default` are mandatory.

Now in *Change report odt source* (admin section) you can upload ODT templates for each predefined language.



4.1.4 Post transition signal

The transition feature sends post transition signal. Arguments defined by the signal are:

- `user` - signed in user executing transition,
- `assets` - assets used in transition,
- `transition` - tranistion which is executed.

This is an example of the signal receiver:

```
import django.dispatch
from ralph_assets import signals

@django.dispatch.receiver(signals.post_transition)
def post_transition_handler(sender, user, assets, transition, **kwargs):
    pass
```

CHAPTER 5

History module

This module observe any changes on registered models and use Django's signals to detect changes on model. Field excluded from history (on default):

- `created`,
- `modified`,
- `invoice_date`,
- `cache_version`,
- `rght`,
- `level`,
- `lft`,
- `tree_id`,

You can change this list by overriding `exclude_fields_from_history()` method in model which is registered.

Features:

- one history view for all models,
- simple API,
- stored information from all field types included `ForeignKey` and `ManyToManyField`,
- bulk create for many changes,
- based on Django's content types.

5.1 Typical usage

Add `HistoryMixin` to model. That's all.

5.2 API

Add to history:

```
from ralph_assets.history import History

changes = [
        {
                'field_name': 'foo',
                'old_value': 123,
                'new_value': 321,
        },
        {
                'field_name': 'bar',
                'old_value': 'Lorem ips',
                'new_value': 'Lorem ipsum',
        },
]
History.objects.log_changes(asset, user, changes)
```

Get history for concrete object:

```
from ralph_assets.history import History

history = History.objects.get_history_for_this_object(asset)
```

CHAPTER 6

Regions

Imagine such case... You've got Ralph application with many assets, your company is expanding and new departments are rising. There is a big chance that you would like to isolate *assets* access among the departments. You can do it. There is a feature called *regions*.

6.1 How to use it?

The *Regions* feature depends on two things:

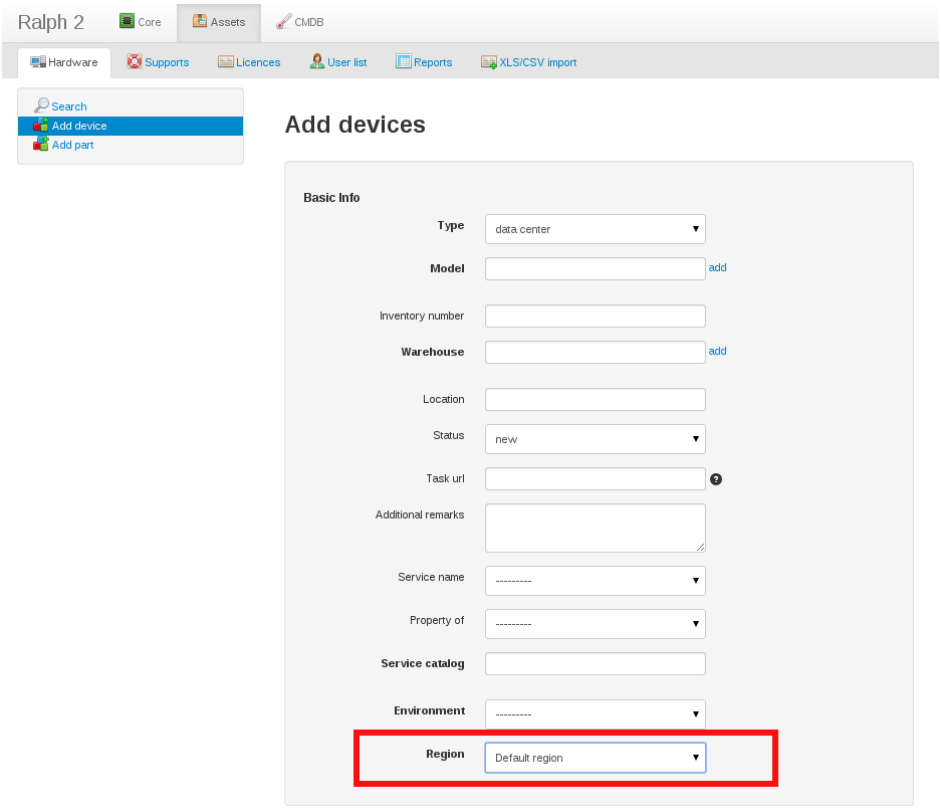
- *region* is assigned to Ralph *user* (one *region* or many *regions*),
- *region* is assigned to *asset* (only one *region*).

If asset's *region* and user's *region* match, *user* see such asset. If asset's *region* and user's *region* does not match, *user* does not see such asset.

The assigning *region* to *user* (or *asset*) is done by Ralph forms.

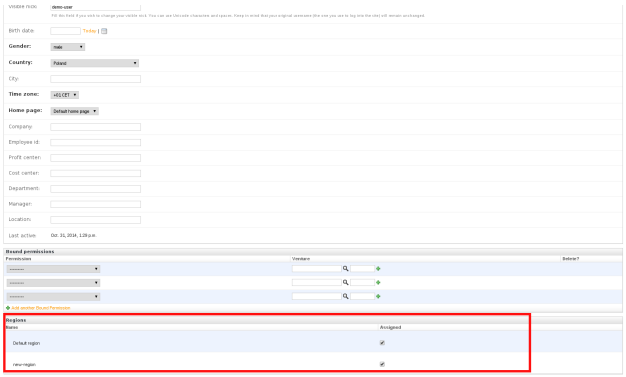
6.1.1 Assigning region to asset

Region can be assigned to *asset* by regular asset's form, like:



6.1.2 Assigning region to user

Region can be assigned to *user* by *admin* form, like:



6.1.3 Adding new region

If you would like to add new *region*, you can do it by *admin* form, like:

To avoid necessity of setting *region* on each *user*. There is a default region, called: "Default region". You can easily change it by Ralph settings, like:

```
DEFAULT_REGION_NAME = 'my-custom-name-for-default-region'
```

The above description is about using *regions* with *assets*. This is not the only option. *Regions* also can be used with *Supports* and *Licences* as well. The way of using it is analogical to *assets* described above.

CHAPTER 7

Signals

The *ralph_assets.signals* module defines a set of signals sent by the ralph_assets.

7.1 Transition signals

Use this type of signals if you want send email or something else after some transition.

7.1.1 post_transition

Arguments sent with this signal:

user A user which run transition.

assets A list of assets which undergo transition.

transition A transition object.

7.2 Form signals

Signals related with form.

7.2.1 post_customize_fields

This signal is useful if you want customize form from external application. Arguments sent with this signal:

sender The form instance.

mode A string. Possible values: `bo` or `back_office`.

For example:

```
from django import forms

from ralph_assets.signals import post_customize_fields


@receiver(post_customize_fields)
def my_awesome_customizer(sender, mode, **kwargs):
```

```
    if(len(sender['barcode'].value) < 50)
        sender.fields['barcode'].widget = forms.widgets.TextInput()
```

CHAPTER 8

Change Log

8.1 DEV

- [scrooge-api]: changed warehouse source from warehouse to datacenter

8.2 2.5.1

Released on April 23, 2015

8.2.1 New features

- Added versioning to static files
- Added configuration-url and visualization-url to edit asset view.
- Small forms improvments. Added help texts to some fields.
- Changed look and feel for support listing (Djid)
- Change message in bulk edit location

8.3 2.5.0

Released on April 13, 2015

8.3.1 New features

- Added Region feature to assets, liceces and supports.
- Added location fields to data center asset: data-center, server-room, rack, position, slot, orientation, slot-number.
- Added data center visualization.
- Added support for Accessories.
- Removed Ralph from requirements. Now Ralph requires ralph_assets.
- Added quantity to assigned licences.

- Added bulk edition for blade servers.
- Added redirecting button to return-asset transition from backoffice bulk-edit and single edit.
- Added datacenter filter to asset reports.

8.3.2 Minor improvements

- Improved history mechanism.
- Reduced SQL queries about 50% in Licence and Hardware list view,
- Made AssetModel's Category required.
- Added purchase_order field to Asset model.
- Seperated assets statuses for data center and back office.
- Added department to DC's search form.
- Removed office info field from bulk edit.
- Skip liquidated assets in Scrooge API.
- API for supports for Scrooge.
- Renamed rack field to rack_old.
- Added new status 'to deploy' for Assets.
- **Changes in form fields:**
  - New fields were added: hostname, service catalog name and management IP address. Now these fields can be displayed on Rack Visualisation view.
  - Added *Region* to assets bulk edit form & admin form.
  - Added Licence field to back-office bulkedit.
- Mobile responsive version for asset editing.

8.3.3 Fix

- Fixed soft-deleting feature on Licenses and Supports.
- Sync parent also for blade servers - use slote_no instead position.
- Fixed counting in 'status - model' report
- Show `Full history` even when the object have empty history.
- Running post_transition as an non-blocking.

8.4 2.3.0

Released on October 2, 2014

- Added new fields service & environment to asset; both fields are synchronised with linked device from Ralph,
- Improved API Scrooge,
- Refactored history mechanism with many improvements (detect many-to-many & foreign-key changes),

- Redesigned navigation, added mode switch in assets view,
- Added new fields in search,
- Added confirmation on leaving unsaved form,,
- Redesigned report views & added new ones,
- Changes in Ralph device linking algorithm,
- Bugfixes.

8.5 2.2.0

Released on August 19, 2014

- Assets in license forms are autocompleted by device's hostname,
- Added *additional notes* field to supports search,
- Added popup with information about unsaved changes on the form,
- UI improvement in reports,

8.6 2.1.0

Released on August 1, 2014

- Added new reports feature,
- New change-hostname transition,
- Transitions send signals,
- Updated documentation,
- Updated api_ralph (assigned supports),
- Expired information in support lookup,
- Removed useless fields from BackOffice edit form,
- Bugfix - wrong order in exported CSV in Assets,

8.7 2.0.2-rc3

Released on July 9, 2014

- Added supports submodule
- Added generate hostname feature
- Added bulkedit in licences
- Minor bugfixes

8.8 2.0.2-rc2

Released on June 13, 2014

- Added asset id column to asset report

8.9 2.0.1-rc2

Released on June 6, 2014

- Bugfix - slots field is not shown when model category is blade

8.10 2.0.0-rc2

Released on June 3, 2014

- Bugfixes in API,
- Bugfixes in MANIFEST.in,
- Minor improvements in admin - Assets count column in model,
- Minor improvement in API - full model resource,
- Minor changes in model fields,
- Minor field changes in forms,
- Minor JS fixes,
- Unittests improvements - use `factory_boy`

8.11 2.0.0-rc1

- Preparing to release a stable version
- New Licence module
- Improvement in asset fields
- Simple transitions
- Bug fixes

8.12 1.4.3

- Added warning logger with cores count from ralph and assets

8.13 1.4.2

- Changed AssetModel schema. Now height_of_device is a float field
- Added to AssetModel column named cores_count
- Changed in api_pricing conditions for getting assets

8.14 1.4.1

- Added Warehouse column to template bulk_edit file

8.15 1.4.0

- Changed limit in sn field when add/edit new device
- Visual grouping fields invoice_date and invoice_report when bulk edit assets
- Added deprecation rate field to bulk edit assets
- Added warehouse field to bulk edit assets

8.16 1.3.2

- cores_count from Asset returns 0 instead of None

8.17 1.3.1

- Add invoice date column to search table

8.18 1.3.0

- Fix bulk edit autocomplete
- Added 25 as default value of deprecation_rate
- Created a method in API to retrieve warehouses
- Added fields like venture_id, is_blade, cores_count, power_consumption, height_of_device and warehouse_id to get_assets API
- Added fields like power_consumption and height_of_device to AssetModel model
- Moved category from Asset model to AssetModel model
- Added cores_count method as property to Asset model

8.19 1.2.13

- fixes of Discovered column. Also it shows now on csv reports.

8.20 1.2.12

- Improved the csv exporting system

8.21 1.2.11

- Basing deprecation on invoice date instead of delivery date

8.22 1.2.10

- Pricing api uses only devices that existed on given date
- Pricing api can use forced deprecation

8.23 1.2.9

- Merged the u_height and size attributes
- Dynamically requiring 'slots' for blade categories
- Fixed unit tests

8.24 1.2.7

Released on October 03, 2013

- Added API for Ralph.
- Required form fields are now labelled accordingly.
- `ralph_device_id` get automatically cleaned when when Device linked to it gets deleted.
- Added partial and exact searches to assets.
- Unlinking assets from devices (and searching for unlinked assets) is now possible.
- Added searching assets by `ralph_device_id`. Added option to create stock devices for unlinked assets.
- Fixed creating assets with `add part` button.
- Column `department` added to csv report in `search DC assets`.

8.25 1.2.6

Released on August 08, 2013

- Added ajax autocomlation for Asset by barcode and/or sn.
- Disabled admin deletetion for Assets.
- Added link to the Pricing App.
- Added field: last modification, asset_id to csv file.

8.26 1.0.0

- initial release

r

ralph_assets.signals, 27

R

ralph_assets.signals (module), 27