# Tryton User Documentation

a system to generate dynamically a customized user's manual

Guillem Barba Domingo, NaN·tic (www.nan-tic.com)

July 19, 2012

## Index

## Components

- Sphinx
  - sphinxcontrib-inheritance
  - trydoc
- trytond-doc
- Module sources

## Components: sphinxcontrib-inheritance

Sphinx plugin to allow generate a single document based on parts (modules).
It provides directives and syntax to insert or replace existing content of the manual in a similar way than Tryton's views.

| | |
|---|---|
| Package | https://pypi.python.org/pypi/sphinxcontrib-inheritance |
| Documentation | http://pythonhosted.org/sphinxcontrib-inheritance |
| Sources | https://bitbucket.org/nantic/sphinxcontrib-inheritance |

## Components: trydoc (sphinxcontrib)

Sphinx plugin to allow to get data from Tryton: menu and field names, and other data which has an entry in ir.model.data.
It also provides some script to setup and mantain the Sphinx project: trydoc-quickstart trydoc-symlinks trydoc-update-modules

| | |
|---|---|
| Package | https://pypi.python.org/pypi/trydoc |
| Documentation | http://pythonhosted.org/trydoc |
| Sources | https://bitbucket.org/nantic/trydoc |

## Components: trytond-doc

Provides the base documentation:

It also provides the documentation for core modules.

Sources https://bitbucket.org/trytonspain/trytond-doc

## Components: module sources

Each module contains its own documentation in **doc/<language>** directory which extends the base or inherited modules documentation.
For convenience, there is a different file for each inherited document, and they are named as the inherited file.

## Setup

1. Prerequisites:
   - Tryton instance to be used from **proteus**.
   - Sources of **trytond-doc** and modules.
2. Install requirements: proteus, sphinxcontrib-inheritance, trydoc
3. Prepare **Sphinx project** for each language and database.
   - **trydoc-quickstart**: Initialize the directory with *Makefile*: *modules.cfg*, localized *index.rst* and customized *conf.py*
   - **trydoc-update-modules**: Add installed modules in a database to *modules.conf*
   - **trydoc-symlinks**: Create symlinks to *doc/<language>* directories into manual directory with the name of module
4. **make**: Compile manual

```
$ make
```

## Write documentation

1. Think which information you want to give and where it will be placed. Which paragrafs, sections or lists have to be modified?
2. Search (or create) the inheritance point
3. Write a **.rst** file for each inherited document in **doc/<language>** directory of your module.

## Write documentation: Tips & Tricks

### Think the manual extensions

Compile the manual only with the modules in the dependencies of the module you want to document.

### Search inheritance points

Set to *True* the **inheritance_debug** option in **conf.py** file to show the available inheritance points in generated HTML.

### Debug inheritance

Set to *True* the **verbose** option print more information when compile.

## Questions

Tanks!

NaN·tic
Guillem Barba Domingo
guillem@nan-tic.com
@wallas85
linkedin.com/in/guillembarba