Documentation tutorial Documentation

Release 0.3.0

Patrick Vergain

2020-4-30

Contents

Documentation tutorial

Release 0.3.0

Date 2020-4-30

Authors **Patrick Vergain**

Target Tutorial about documentation

See also:

- https://gdevops.gitlab.io/tuto_devops/
- genindex
- *reStructuredText Sphinx documentation*
- https://gitlab.com/gdevops/tuto_documentation
- https://gdevops.gitlab.io/tuto_documentation/index.html
- https://devopstutodoc.readthedocs.io/en/latest/index.html
- https://readthedocs.org/projects/devopstutodoc/

CHAPTER 1

Documentation

See also:

- http://blog.smartbear.com/software-quality/bid/256072/13-reasons-your-open-source-docs-make-people-want-to-scream
- http://brikis98.blogspot.de/2014/05/you-are-what-you-document.html
- http://redaction-technique.org/index.html

1.1 Documentation news

1.1.1 Documentation news 2018

Contents

Bringing interactive examples to MDN

See also:

- https://hacks.mozilla.org/2018/03/bringing-interactive-examples-to-mdn/

1.1.2 Documentation news 2017

Contents

Autodoc-style extraction into Sphinx for your JS project

See also:

- *sphinx-js : autodoc-style extraction into Sphinx for your JS project*

1.1.3 Documentation news 2016

Contents

La documentation linux utilise sphinx

See also:

- https://github.com/return42/linuxdoc
- https://return42.github.io/linuxdoc/
- https://lwn.net/Articles/692704/

L'annonce sur https://lwn.net/Articles/692704/

1.2 Documentation Advices

See also:

- parse_API_doc
- http://producingoss.com/en/getting-started.html
- http://justwriteclick.com/book/
- http://brikis98.blogspot.de/2014/05/you-are-what-you-document.html
- http://redaction-technique.org/index.html

Contents

1.2.1 You are what you document (Monday, May 5, 2014)

See also:

- http://brikis98.blogspot.de/2014/05/you-are-what-you-document.html
- http://jacobian.org/writing/great-documentation/
- http://zachholman.com/posts/documentation/
- http://blog.parse.com/2012/01/11/designing-great-api-docs/
- http://www.mikepope.com/blog/DisplayBlog.aspx?permalink=1680
- https://github.com/PharkMillups/beautiful-docs
- http://blog.codinghorror.com/if-it-isnt-documented-it-doesnt-exist/
- http://docs.writethedocs.org/writing/beginners-guide-to-docs/

The number one cause of startup failure is not the product, but the distribution: it doesn't matter how good the product is if no one uses it.

With software, the documentation is the distribution: it doesn't matter how good the code is if no one uses it. If it isn't documented, it doesn't exist.

1.2.2 Rédaction technique

See also:

- http://redaction-technique.org/index.html

Libérez vos informations de leurs silos

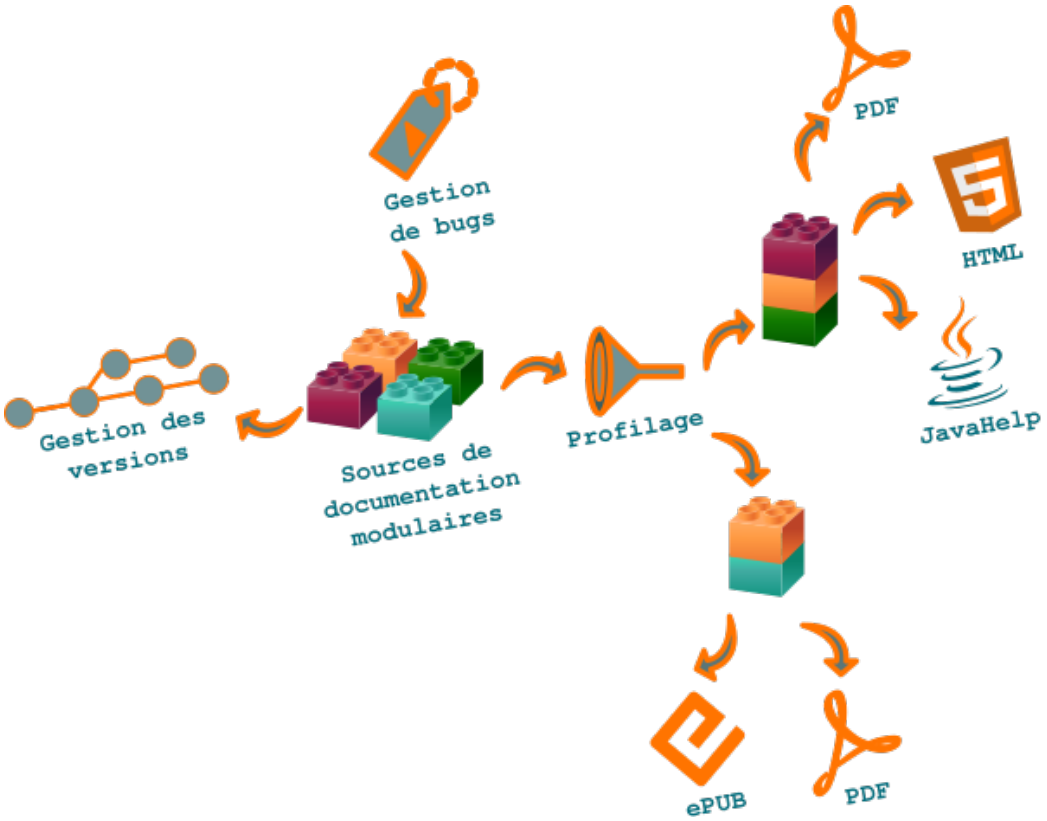
Des solutions souples et fiables libèrent vos informations des silos d'information cloisonnés où elles sont emprisonnées et sous-exploitées.

Oubliez MS Word ou FrameMaker pour passer de la maintenance de la documentation à la gestion du cycle de vie des projets documentaires modulaires !

Intégrer la documentation aux processus de développement

La documentation fait partie du logiciel. Fournie avec le produit, elle doit :

- sortir en même temps,
- suivre les mêmes cycles de vie, et
- faire l'objet des mêmes processus de production et de contrôle qualité.



Elle doit répondre idéalement aux critères suivants :

- pas de vendor lock-in (**indépendance du format** et de l'éditeur de contenu),
- chaînes de publication **libres et gratuites**,
- mise en page totalement automatisée.

Il y a quelques années encore, les seuls outils permettant de fournir des livrables de qualité au format PDF ou HTML reposaient sur des formats binaires et propriétaires qui s'intégraient mal aux systèmes de gestion de versions des équipes de développement.

Résultat : réalisée à part, la documentation technique répondait difficilement aux mêmes exigences de qualité et de délai de mise sur le marché que les produits.

DocBook, puis **DITA XML** et reStructuredText ont changé la donne : ces formats texte peuvent être modifiés avec tout type de programme, du simple éditeur de texte à l'‘IDE graphique, et s'intègrent parfaitement sous Subversion, Git ou tout autre système de gestion de versions.

1.2.3 13 Things People Hate about Your Open Source Docs

See also:

- http://blog.smartbear.com/careers/13-things-people-hate-about-your-open-source-docs/
- http://www.framablog.org/index.php/post/2013/06/28/documentation-defauts-open-source

1.2.4 Beautiful docs

See also:

- https://github.com/PharkMillups/beautiful-docs

1.2.5 Designing Great API Docs (11 Jan 2012)

See also:

- http://blog.parse.com/2012/01/11/designing-great-api-docs/

1.2.6 Docness

See also:

- http://docness.readthedocs.org/
- http://docness.readthedocs.org/en/latest/documentation-is-part-of-product.html

Consider the documentation as a component of your product, i.e. don't package it as an external product or as an option.

In software development, your "product" is usually made of the software, which itself is usually based on source code.

Docness Source code

See also:

https://github.com/benoitbryon/docness

1.2.7 Hacking distributed (february 2013)

See also:

- http://hackingdistributed.com/2013/02/11/principled-documentation/

In the beginning, there was no documentation.

These days, infrastructure software has terrible documentation.

Take heed of these commandments, for, among hackers, judgment day is every day.

1.2.8 Jacob Kaplan-Moss (November 10, 2009)

See also:

http://jacobian.org/writing/great-documentation/what-to-write/

1.2.9 Agile documentation best practices

See also:

- http://www.agilemodeling.com/essays/agileDocumentationBestPractices.htm
- http://www.agilemodeling.com/essays/agileDocumentationBestPractices.htm#sthash.nuUKabMJ.dpuf

Ideally, an agile document is just barely good enough, or just barely sufficient, for the situation at hand.

Documentation is an important part of agile software development projects, but unlike traditionalists who often see documentation as a risk reduction strategy, agilists typically see documentation as a strategy which increases overall project risk and therefore strive to be as efficient as possible when it comes to documentation.

Agilists write documentation when that's the best way to achieve the relevant goals, but there often proves to be better ways to achieve those goals than writing **static documentation**.

This article summarizes common "best practices" which agilists have adopted with respect to documentation.

Best practices for increasing the agility of documentation:

Writing Prefer executable specifications over static documents Document stable concepts, not speculative ideas Generate system documentation

Simplification Keep documentation just simple enough, but not too simple Write the fewest documents with least overlap Put the information in the most appropriate place Display information publicly

Determining What to Document Document with a purpose Focus on the needs of the actual customers(s) of the document The customer determines sufficiency

Determining When to Document Iterate, iterate, iterate Find better ways to communicate Start with models you actually keep current Update only when it hurts

General Treat documentation like a requirement Require people to justify documentation requests Recognize that you need some documentation Get someone with writing experience

1.2.10 Best Practices for Documenting Technical Procedures Melanie Seibert

See also:

- http://fr.slideshare.net/MelanieSeibert/best-practices-for-documenting-technical-procedures

1.2.11 Plone

See also:

- http://opensourcehacker.com/2012/01/08/readthedocs-org-github-edit-backlink-and-short-history-of-plone-documentation/

1.2.12 Twilio

See also:

- http://twilio.com/engineering/2012/01/18/dont-skimp-on-documentation
- http://readthedocs.org/docs/twilio-python/en/latest/index.html

1.2.13 Other advices

Write the docs

See also:

- http://docs.writethedocs.org/
- http://docs.writethedocs.org/en/2013/about/index.html
- http://www.slideshare.net/janetswisher/
- https://twitter.com/writethedocs

Contents

Write the docs vision

See also:

http://docs.writethedocs.org/en/2013/about/vision.html

Documentation is how you share your creations with the world.

If you want people to benefit from your work, they have to be able to use it.

Help me, help you. Do something and change something.

We believe it should be easy for people to start writing documentation. There should be straight-forward guides to getting started with good tools.

Write the docs news

Write the docs news

Write the docs 2013

Resources for writing good documentation 7 august 2013

See also:

- http://justwriteclick.com/book/

- http://www.justwriteclick.com
- http://docs.openstack.org
- http://api.openstack.org
- http://justwriteclick.com/2011/10/21/google-summer-of-code-doc-summit-stories/

Contents

Email

```
Anne Gentle <annegentle@gmail.com>
à:   write-the-docs@googlegroups.com
date:        7 août 2013 16:50
objet:       Re: Resources for writing good documentation
```

On Monday, August 5, 2013 11:30:23 PM UTC-5, Eric Holscher wrote:

```
Hey all,

Looks like I am going to be giving a beginners presentation about writing
docs at the PDX Python user group.
I am adding a resource section that will list good places to go to look for
information about writing docs. My current list contains:

* http://producingoss.com/en/getting-started.html
* http://docs.writethedocs.org/

What are some other good documentation resources that I might be missing ?
I hope to add all of the content from this talk back into docs.writethedocs.org,
so that it will live on.
```

My perspective comes from being a technical writer encouraging other technical writers to contribute to open source projects.

I coordinate the OpenStack documentation through collaborative authoring, treating the docs like code with reviewed merges and bug logging, triaging, and so on. So my audience differs a bit from yours, but there are a lot of overlapping concepts.

For the audience of programmers you want to coach to write, I'd start with audience analysis and task analysis. I've attended a workshop Janet Swisher gave at the 2010 Google Summer of Code Doc Sprints where this approach was extremely effective.

From http://justwriteclick.com/2011/10/21/google-summer-of-code-doc-summit-stories/

- Who is using your tool ?
- Why do they use your tool ?
- What kinds of things are they trying to do ?
- What can you assume they know ?

- What do they probably not know when they approach your tool ?

For the audience of writers you want to encourage to write for your project, I have a book with an Open Source Documentation chapter. http://justwriteclick.com/book/

It assumes a lot of pre-requisite knowledge such as audience and task analysis. I would definitely include slides about audience analysis and task analysis when coaching devs to write.

Janet Swisher does the "encourage writers to write in the open" angle too, and has a great set of presentations at http://www.slideshare.net/janetswisher/. You can take anything from my presentations at http://slideshare.net/annegentle. I can send source if you want it. Janet and I co-presented about FLOSS Manuals at a Linux conference and the thesis there was partially "find communities of writers to work on your project's docs." That's a tactic as well.

Writers not only get distracted with a style guide, but also the writing/publishing/review tools. Using a third-party "referee" style guide for style questions is ideal (like Daniel Beck said). I'd coach "just put your butt in a seat and write" the tooling can be sorted out later.

Discussing tools without focusing on the content can be a huge time waster, especially with devs. :) Encourage them to get info out of their brains.

You probably know all this instinctively, so it's great to write it down and share !

Anne

See also:

- http://www.justwriteclick.com
- http://docs.openstack.org
- http://api.openstack.org
- http://slideshare.net/annegentle

Mindshare

See also:

- http://docs.writethedocs.org/en/2013/writing/mindshare.html

Having a culture of documentation inside of a company is a great thing.

Building a culture around documentation however is a **hard thing to do**.

This will be a guide with some information and suggestions for how to go about bringing good docs into a company.

1.3 Documentation generators

See also:

- https://github.com/yoloseem/awesome-sphinxdoc
- http://blog.smartbear.com/software-quality/bid/256072/13-reasons-your-open-source-docs-make-people-want-to-scream

1.3.1 Sphinx

See also:

- https://github.com/sphinx-doc/sphinx
- https://en.wikipedia.org/wiki/Sphinx_(documentation_generator)
- https://github.com/yoloseem/awesome-sphinxdoc
- http://sphinx-doc.org/
- http://sphinx-doc.org/latest/
- http://sphinx-doc.org/examples.html
- http://rest-sphinx-memo.readthedocs.org/en/latest/Project.html
- http://rest-sphinx-memo.readthedocs.org/en/latest/References.html
- http://redaction-technique.org/index.html

Fig. 1: *Sphinx logo*

- *Description*
- *Sphinx source code (Sphinx dev guide)*
- *Sphinx applications*
- *reST Sphinx*
- *Sphinx extensions and contributed extensions*
- *Sphinx howto*
- *Hébergeurs Sphinx*
- *Sphinx examples*
- *Sphinx i18n*
- *Sphinx installation*
- *Sphinx people*
- *Sphinx tutorials*
- *Tools for Sphinx*
- *Sphinx themes*
- *Sphinx templating*
- *Sphinx translations*

- *Sphinx versions*

Description

Sphinx is a tool that makes it easy to create intelligent and beautiful documentation, written by Georg Brandl and licensed under the BSD license.

It was originally created for the new Python documentation, and it has excellent facilities for the documentation of Python projects, but C/C++ is already supported as well, and it is planned to add special support for other languages as well.

Sphinx source code (Sphinx dev guide)

See also:

- https://github.com/sphinx-doc/sphinx

Sphinx development (Development Guide)

See also:

- https://github.com/sphinx-doc/sphinx

Sphinx applications

sphinx applications

Editing sphinx doc on the web

Baow

See also:

http://www.baow.com/help/

Baow is a tool that makes it easy to organize your internet resources and create intelligent and beautiful web pages within Firefox web browser .

Code review

See also:

code_review

Gerrit

See also:

gerrit_code_review

Github Fork and Edit button

. seealso:

```
- :ref:`github_fork_and_edit_button`
- https://github.com/blog/844-forking-with-the-edit-button
- https://confluence.atlassian.com/display/BITBUCKET/Fork+a+Repo,+Compare+Code,
↪+and+Create+a+Pull+Request
```

You can commit file edits through GitHub web interface using Fork and Edit button Alternative, clone the repository using git, perform changes and push them back

Plone collective GitHub repository has open-for-all contribution access. If you want to contribute changes without asking the maintainers to merge them, please add your GitHub username to your profile on plone.org and request access here.

pydocweb

See also:

https://github.com/pv/pydocweb

Tool for collaboratively documenting Python modules via the web.

```
johnKitchin07 <johnrkitchin@gmail.com>
répondre à  sphinx-dev@googlegroups.com
à   sphinx-dev <sphinx-dev@googlegroups.com>
date    17 mars 2011 14:11
objet   [sphinx-dev] Re: Wiki based on Sphinx
liste de diffusion  <sphinx-dev.googlegroups.com> Filtrer les messages de cette liste␣
↪de diffusion
```

You might check out the Numpy documentation project (http://docs.scipy.org/numpy/Front%20Page/).

I think they have a wiki/rst/sphinx like solution for editing documentation (http://code.google.com/p/pydocweb/) that is written in Django.

copypasta and sphinx

See also:

- https://copypasta.credibl.es/
- django_ucomment

Copypasta is a collaborative editing tool. Readers submit edits, authors approve changes, everyone wins.

garlicsims and copypasta

See also:

http://blog.garlicsim.org/post/3897688973/garlicsims-documentation-is-now-user-editable

A while ago I stumbled upon a very cool tool by Kurt Mackey called Copypasta.

As the website describes it: “Copypasta is a collaborative editing tool.

Readers submit edits, authors approve changes, everyone wins."

It's still a young and experimental project, but it's quite promising. It often happens that I'm reading a friend's blog post and I see something that I want to fix, like a typo or a grammar mistake.

What I usually do is fire an email to that friend alerting him to the typo. This is of course inefficient. Copypasta lets you edit the page yourself in the browser, and then it shoots an email to the site owner with your proposed changes.

Currently it requires the site owner to put the Copypasta button on the website for it to work; in the future Kurt might release a browser plugin that will let you edit any site on the internet.

That will be really awesome.

So I decided to put Copypasta on the GarlicSim documentation site. (This is a Sphinx-based documentation site.) Now anyone can offer fixes for GarlicSim's docs!

Let me know if there are any bugs.

I hope that more Python package maintainers will do this for their projects' documentation so we could all help to improve each other's documentation.

Now what I want is a Sphinx backend for Copypasta which will be able to automatically change the documentation source in the repository, possibly creating a GitHub pull request with the changes to the docs source...

But now I'm fantasizing :)

comment from Henning

You could also try http://ucomment.org/

comment from Jonas Obrist (https://github.com/ojii/)

About the last paragraph, I implemented exactly that for *readthedocs.org* unfortunately it's not 100% working yet, but stay tuned!

comment from Robert Kern

ou may want to give the Numpy Documentation Editor a look. It's a web documentation editor for Sphinx documentation checked into a repository.

- http://docs.scipy.org/numpy/Front%20Page/
- *http://code.google.com/p/pydocweb/*

coolRR 1 day ago in reply to Robert Kern

Yeah, I remember I considered using it for the GarlicSim docs a few months ago but decided against it, don't remember why.

Looking at it now I see that it's not very easy to use. I now got into the NumPy docs and pressed "edit" and I got a confusing screen where I don't understand how I'm supposed to edit anything. Possibly UX is the reason why *pydocweb* hasn't gained wide adoption in the Python world.

Sphinx to github

See also:

- https://github.com/michaeljones/sphinx-to-github

This project is designed to help you get around the github-pages Jekyll behaviour of ignoring top level directories starting with an underscore.

This is solved in a much neater way by creating a .nojekyll in the root of you github-pages which will disable Jekyll as described here and here.

This makes this project largely useless!

Thank you to acdha for making me aware of this.

sphinx-wiki

See also:

- https://bitbucket.org/kevindunn/sphinx-wiki/wiki/Home

A Mediawiki extension that allows ReStructuredText markup, compiled via Sphinx, to be used.

Examples

This extension is used on several sites operated by the author. Two that are publicly available as university courses:

- Statistics for Engineering
- Numerical methods

On both sites you can click Edit at the top of the page to see the page's source.

sphinx tinkerer application

See also:

- http://tinkerer.me/
- http://tinkerer.me/pages/documentation.html
- pelican_blog

Contents

What is Tinkerer ?

Tinkerer is a blogging engine/static website generator powered by Sphinx.

It allows blogging in reStructuredText format, comes with out-of-the-box support for post publishing dates, authors, categories, tags, post archive, RSS feed generation, comments powered by Disqus and more.

Tinkerer is also highly customizable through Sphinx extensions.

Why Tinkerer ?

- Because "hacker" is way overused nowadays.
- Because static websites are cool.
- Because you already write documentation in RST format and you love it.
- Because you have Pygments for sharing highlighted code.
- Because you can use your favorite Sphinx extensions right away.

Hosting on bitbucket

See also:

http://tinkerer.me/doc/deploying.html#hosting-on-bitbucket

Versions

tinkerer versions

tinkerer 0.3

See also:

http://tinkerer.bitbucket.org/2012/02/09/tinkerer_beta_0_3_released.html

What's New

Tinkerer went international! Spanish and Catalan translations are now available. To have your Tinkerer blog displayed in Spanish, add the following line to your conf.py:

```
language = "es"
```

For Catalan, add:

```
language = "ca"
```

Limited support for Facebook Comments as an alternative to Disqus comments

RSS feed enhancements: RSS feed auto-discovery and feed categories

Multiple bugfixes and minor style tweaks to the Modern theme

tinkerer 0.2

- Support for Drafts.
- Fixes for cross-references and embedded images which were not displaying correctly on home page and RSS feed.
- Ensure Tinkerer runs only from the blog root directory unless when setting up a new blog (this prevents accidental deletes and mysterious exceptions).
- Minimal support for documentation - prev and next links will be displayed on pages under doc/ or docs/ path.
- Many other small extension fixes.
- CSS fixes (gradient not showing in Firefox, page not scaling correctly on retina displays).

Upgrading from 0.1

There are a couple of steps required if upgrading from 0.1:

In your conf.py replace:

```
# Add file patterns to exclude from build
exclude_patterns = []
```

with:

```
# Add file patterns to exclude from build
exclude_patterns = ["drafts/*"]
```

This will make Sphinx stop warning you about drafts not being included in the build.

Make sure your master.rst file ends with a blank line. If not, append a blank l ine at the end of it.

Thank You!

A big Thank You to everyone who showed interest in the project and for the valuable feedback you provided.

reST Sphinx

reStructuredText Sphinx documentation

This section is a brief introduction to reStructuredText (reST) concepts and syntax, intended to provide authors with enough information to author documents productively.

Since reST was designed to be a simple, unobtrusive markup language, this will not take too long.

Date 2020-4-30

See also:

- http://sphinx-doc.org/latest/index.html
- http://packages.python.org/an_example_pypi_project/sphinx.html
- http://docutils.sourceforge.net/rst.html
- http://rest-sphinx-memo.readthedocs.org/en/latest/ReST.html
- http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/_sources/source/sphinx/rest_syntax.txt

- http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/source/sphinx/rest_syntax.html
- *Sphinx*

`conf.py` Examples

See also:

- http://sphinx-doc.org/latest/config.html?highlight=conf#conf

The configuration directory must contain a file named `conf.py`. This file (containing Python code) is called the "build configuration file" and contains all configuration needed to customize Sphinx input and output behavior.

The configuration file is executed as Python code at build time (using `execfile()`, and with the current directory set to its containing directory), and therefore can execute arbitrarily complex code.

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

pypi Example

See also:

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

In your `doc/source` directory is now a python file called `conf.py`.

This is the file that controls the basics of how sphinx runs when you run a build. Here you can do this like:

- Change the version/release number by setting the `version` and `release` variables.
- Set the project name and author name.
- Setup a project logo.
- Set the default style to `sphinx` or `default`. Default is what the standard python docs use.

and much much more.

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

Exclude patterns

exclude_patterns

See also:

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

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

sphinx markup

See also:

http://sphinx-doc.org/latest/markup/index.html

sphinx inline markup

See also:

http://sphinx-doc.org/latest/markup/inline.html

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

abbr

See also:

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

An abbreviation. If the role content contains a parenthesized explanation, it will be treated specially: it will be shown in a tool-tip in HTML, and output only once in LaTeX.

Example:

```
:abbr:`LIFO (last-in, first-out)`.
```

LIFO (last-in, first-out)

doc

See also:

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

Link to the specified document; the document name can be specified in absolute or relative fashion.

If no explicit link text is given the link caption will be the title of the given document.

Example 1 : local link

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

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

Example 2 : with explicit link text

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

reference is *Index principal* or Index local

Example 3 no explicit link text

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

reference is *Documentation tutorial* or ../index

command

See also:

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

Example:

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

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

download

See also:

http://sphinx-doc.org/latest/markup/inline.html

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

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

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

An example:

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

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

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

glossary

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

```
.. glossary::

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

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

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

```
.. glossary::

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

(When the glossary is sorted, the first term determines the sort order.)

New in version 0.6: You can now give the glossary directive a `:sorted:` flag that will automatically sort the entries alphabetically.

Changed in version 1.1: Now supports multiple terms and inline markup in terms.

guilabel

See also:

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

Labels presented as part of an interactive user interface should be marked using `:guilabel:`.

This includes labels from text-based interfaces such as those created using curses or other text-based libraries.

Any label used in the interface should be marked with this role, including:

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

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

To include a literal ampersand, double it.

Example 1

```
button :guilabel:`Start`
```

button *Start*

Example 2 ampersand accelerators

`guilabel` also supports ampersand accelerators just like guilabel.

```
button :guilabel:`&Start`
```

button *&Start*

menuselection

Contents

Introduction

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

The names of individual selections should be separated by `-->`.

Example 1

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

```
:menuselection:`Start --> Programs`
```

Start → Programs

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

Example 2 ampersand accelerators

menuselection also supports ampersand accelerators just like *guilabel*.

```
:menuselection:`Start --> &Programs`
```

Start → Programs

program

See also:

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

The name of an executable program.

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

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

Example

```
:program:`Geany.exe`
```

Geany.exe

term (very important)

See also:

http://sphinx-doc.org/latest/markup/inline.html

Reference to a term in the glossary.

The glossary is created using the `glossary` directive containing a definition list with terms and definitions.

It does not have to be in the same file as the `term` markup, for example the Python docs have one global glossary in the `glossary.rst` file.

If you use a term that's not explained in a glossary, you'll get a warning during build.

Example:

```
See :term:`Sphinx`
```

See Sphinx

sphinx misc markup (very important)

index (very important)

Sphinx automatically creates index entries from all object descriptions (like functions, classes or attributes) like discussed in domains.

However, **there is also explicit markup available, to make the index more comprehensive and enable index entries in documents where information is not mainly contained in information units, such as the language reference**.

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

  For example:

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

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


...
```

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

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

! exclamation (important)

You can mark up "main" index entries by prefixing them with an exclamation mark. The references to "main" entries are emphasized in the generated index. For example, if two pages contain

```
   .. index:: Python

and one page contains ::

   .. index:: ! Python

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

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

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

This creates four index entries.

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

only : including content based on tags

Contents

Description

`.. only:: <expression>`

Include the content of the directive only if the *expression* is true. The expression should consist of tags, like this:

```
.. only:: html and draft
```

Undefined tags are false, defined tags (via the `-t` command-line option or within `conf.py`) are true. Boolean expressions, also using parentheses (like `html and (latex or draft)`) are supported.

The format of the current builder (`html`, `latex` or `text`) is always set as a tag.

New in version 0.6.

Example1

in conf.py

See also:

- https://groups.google.com/forum/#!topic/sphinx-users/uUmSCiK-UtU

```
# tags.add('student')
# tags.add('stagiaire')
tags.add('prof')
```

In a .rst file

```
.. only prof

    Solution du problème
```

pair (very important)

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

Example:

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

see

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

seealso

Like *see*, but inserts "see also" instead of "see".

single

Creates a single index entry.

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

triple

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

- `module; search path`
- `search; path, module`
- and `path; module search`.

Deprecated : module, keyword, operator, object, exception, statement, builtin

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

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

sphinx paragraph level markup

See also:

http://sphinx-doc.org/latest/markup/para.html

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

contents (très important)

See also:

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

Table-of-contents markup

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

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

Example 1

```
.. contents::
   :local:
```

Example 2

```
.. contents::
   :depth: 2
```

Contents

deprecated

See also:

- http://sphinx-doc.org/latest/markup/para.html?highlight=warning#directive-warning

`.. deprecated:: version`
Similar to *versionchanged*, but describes when the feature was deprecated. An explanation can also be given, for example to inform the reader what should be used instead. Example:

```
.. deprecated:: 3.1
   Use :func:`spam` instead.
```

Deprecated since version 3.1: Use `spam()` instead.

hlist

See also:

http://sphinx-doc.org/latest/markup/para.html

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

`.. hlist::`
This directive must contain a bullet list. It will transform it into a more compact list by either distributing more than one item horizontally, or reducing spacing between items, depending on the builder.

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

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

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

New in version 0.6.

note

See also:

- http://sphinx-doc.org/latest/markup/para.html?highlight=note#directive-note

An especially important bit of information about an API that a user should be aware of when using whatever bit of API the note pertains to. The content of the directive should be written in complete sentences and include all appropriate punctuation.

Example:

```
.. note::

   This function is not suitable for sending spam e-mails.
```

Note: This function is not suitable for sending spam e-mails.

versionadded

See also:

- http://sphinx-doc.org/latest/markup/para.html?highlight=versionadded#directive-versionadded

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

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

Example:

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

New in version 2.5: The *spam* parameter.

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

versionchanged

See also:

- http://sphinx-doc.org/latest/markup/para.html#directive-versionchanged

`.. versionchanged:: version`
Similar to `versionadded`, but describes when and what changed in the named feature in some way (new parameters, changed side effects, etc.).

warning

See also:

- http://sphinx-doc.org/latest/markup/para.html?highlight=warning#directive-warning

An important bit of information about an API that a user should be very aware of when using whatever bit of API the warning pertains to.

Warning: An important bit of information about an API that a user should be very aware ofwhen using whatever bit of API the warning pertains to.

The content of the directive should be written in complete sentences and include all appropriate punctuation.

This differs from note in that it is recommended over note for information regarding security.

sphinx domain

See also:

- http://sphinx-doc.org/latest/domains.html
- http://sphinx-doc.org/latest/ext/appapi.html#domain-api

Contents

What is a Domain ?

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

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

A domain is a collection of markup (reStructuredText directives and roles) to describe and link to objects belonging together, e.g. elements of a programming language. Directive and role names in a domain have names like domain:name, e.g. py:function. Domains can also provide custom indices (like the Python Module Index).

Having domains means that there are no naming problems when one set of documentation wants to refer to e.g. C++ and Python classes. It also means that extensions that support the documentation of whole new languages are much easier to write.

Default Domain

See also:

http://sphinx-doc.org/latest/config.html#confval-primary_domain

To avoid having to writing the domain name all the time when you e.g. only describe Python objects, a default domain can be selected with either the config value primary_domain or this directive:

```
.. default-domain:: name
```

Select a new default domain. While the primary_domain selects a global default, this only has an effect within the same file.

primary_domain

The name of the default domain. Can also be None to disable a default domain. The default is 'py'. Those objects in other domains (whether the domain name is given explicitly, or selected by a default-domain directive) will have the domain name explicitly prepended when named (e.g., when the default domain is C, Python functions will be named "Python function", not just "function").

sphinx C domain

See also:

- http://sphinx-doc.org/latest/domains.html
- http://docs.python.org/dev/_sources/c-api/unicode.txt
- http://docs.python.org/dev/c-api/unicode.html

Contents

C doc examples

See also:

- http://docs.python.org/dev/_sources/c-api/unicode.txt

C domain

```
.. default-domain:: C
```

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

```
.. c:function:: type name(signature)
```

C function

Describes a C function. The signature should be given as in C, e.g.:

```
.. c:function:: PyObject* PyType_GenericAlloc(PyTypeObject *type, Py_ssize_t nitems)
```

PyObject* **PyType_GenericAlloc**(PyTypeObject **type*, Py_ssize_t *nitems*)

This is also used to describe function-like preprocessor macros. The names of the arguments should be given so they may be used in the description.

Note that you don't have to backslash-escape asterisks in the signature, as it is not parsed by the reST inliner.

C member

Describes a C struct member. Example signature:

```
.. c:member:: PyObject* PyTypeObject.tp_bases
```

The text of the description should include the range of values allowed, how the value should be interpreted, and whether the value can be changed. References to structure members in text should use the `member` role.

C macro

```
.. rst:directive:: .. c:macro:: name
```

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

C type (structure)

```
.. rst:directive:: .. c:type:: name
```

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

C var

Describes a global C variable. The signature should include the type, such as:

```
.. c:var:: PyObject* PyClass_Type
```

Cross-referencing C constructs

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

Cross-referencing a variable

```
.. rst:role:: c:data
```

Reference a C-language variable.

Cross-referencing a function

```
.. rst:role:: c:func
```

Reference a C-language function. Should include trailing parentheses.

Cross-referencing a macro

```
.. rst:role:: c:macro
```

Reference a "simple" C macro, as defined above.

Cross-referencing a structure

```
.. rst:role:: c:type
```

Reference a C-language type.

sphinx C++ domain

See also:

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

The C++ Domain

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

The following directives are available:

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

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

```
.. cpp:function:: bool namespaced::theclass::method(int arg1, std::string arg2)

   Describes a method with parameters and types.

.. cpp:function:: bool namespaced::theclass::method(arg1, arg2)

   Describes a method without types.

.. cpp:function:: const T &array<T>::operator[]() const

   Describes the constant indexing operator of a templated array.

.. cpp:function:: operator bool() const

   Describe a casting operator here.

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

   Describe a constexpr function here.

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

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

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

Will be rendered like this:

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

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

operator bool() **const**
Describe a casting operator here.

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

std::string theclass**::name**

std::string theclass**::name**[**N**][**M**]

type theclass**::const_iterator**

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

These roles link to the given object types:

:cpp:class:
:cpp:func:
:cpp:member:
:cpp:type:
Reference a C++ object. You can give the full signature (and need to, for overloaded functions.)

Note: Sphinx' syntax to give references a custom title can interfere with linking to template classes, if nothing follows the closing angle bracket, i.e. if the link looks like this: `:cpp:class:`MyClass<T>``. This is

interpreted as a link to `T` with a title of `MyClass`. In this case, please escape the opening angle bracket with a backslash, like this: `:cpp:class:`MyClass\<T>``.

Note on References

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

---

sphinx code

autodoc

See also:

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

literalinclude

See also:

- http://sphinx-doc.org/markup/code.html
- http://pygments.org/docs/lexers/

Contents

rst:directive:: .. literalinclude:: filename

.. literalinclude:: filename

Longer displays of verbatim text may be included by storing the example text in an external file containing only plain text. The file may be included using the `literalinclude` directive. For example, to include the Python source file `example.py`, use:

```
.. literalinclude:: example.py
```

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

Tabs in the input are expanded if you give a `tab-width` option with the desired tab width.

:linenos:

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

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

:lineno-start:

See also:

- http://sphinx-doc.org/latest/changes.html?highlight=literalinclude#release-1-3-in-development

New in version 1.3: The `lineno-start` option.

The first line number can be selected with the lineno-start option. If present, linenos is automatically activated as well.

```
# -*- coding: UTF-8 -*-
'''Returns a table of file names with the following informations

'''

__author__ = 'Patrick Vergain <pvergain@gmail.com>'


import hashlib
import logging
import zlib

from unipath import Path


log = logging.getLogger('get_table_hash')

class crc32(object):
    '''hashlib-compatible interface for CRC-32 support

    http://stackoverflow.com/questions/1742866/compute-crc-of-file-in-python
    '''
```

(continues on next page)

(continued from previous page)

```
    name = 'crc32'
    digest_size = 4
    block_size = 1

    def __init__(self, arg=''):
        self.__digest = 0
        self.update(arg)

    def copy(self):
        copy = super(self.__class__, self).__new__(__class__)
        copy.__digest = self.__digest
        return copy

    def digest(self):
        return self.__digest

    def hexdigest(self):
        return '{:08x}'.format(self.__digest)

    def update(self, arg):
        self.__digest = zlib.crc32(arg, self.__digest) & 0xffffffff


```

```
# -*- coding: UTF-8 -*-
'''Returns a table of file names with the following informations

'''

__author__ = 'Patrick Vergain <pvergain@gmail.com>'


import hashlib
import logging
import zlib

from unipath import Path


log = logging.getLogger('get_table_hash')

class crc32(object):
    '''hashlib-compatible interface for CRC-32 support

    http://stackoverflow.com/questions/1742866/compute-crc-of-file-in-python
    '''
    name = 'crc32'
    digest_size = 4
    block_size = 1

    def __init__(self, arg=''):
        self.__digest = 0
        self.update(arg)

    def copy(self):
        copy = super(self.__class__, self).__new__(__class__)
```

(continues on next page)

(continued from previous page)

```
        copy.__digest = self.__digest
        return copy

    def digest(self):
        return self.__digest

    def hexdigest(self):
        return '{:08x}'.format(self.__digest)

    def update(self, arg):
        self.__digest = zlib.crc32(arg, self.__digest) & 0xffffffff


```

:diff:

New in version 1.3: The `diff` option.

If you want to show the diff of the code, you can specify the old file by giving a diff option:

```
.. literalinclude:: example.py
   :diff: example.py.orig
```

```
--- /home/docs/checkouts/readthedocs.org/user_builds/devopstutodoc/checkouts/stable/
↪documentation/doc_generators/sphinx/rest_sphinx/code/literalinclude/example.py.orig
+++ /home/docs/checkouts/readthedocs.org/user_builds/devopstutodoc/checkouts/stable/
↪documentation/doc_generators/sphinx/rest_sphinx/code/literalinclude/example.py
@@ -7,16 +7,8 @@


 import hashlib
-
-# see http://petl.readthedocs.org/en/latest/
-# Extract, Transform and Load (Tables of Data)
-from petl import look
-import glob
-import os
-import argparse
 import logging
 import zlib
-import sys

 from unipath import Path
```

:caption:

New in version 1.3: The `caption` option.

literalinclude supports the `caption` option, with the additional feature that if you leave the value empty, the shown filename will be exactly the one given as an argument:

```
.. literalinclude:: example.py
   :linenos:
   :caption:
```

Listing 1: example.py

```
# -*- coding: UTF-8 -*-
'''Returns a table of file names with the following informations

'''

__author__ = 'Patrick Vergain <pvergain@gmail.com>'


import hashlib
import logging
import zlib

from unipath import Path


log = logging.getLogger('get_table_hash')

class crc32(object):
    '''hashlib-compatible interface for CRC-32 support

    http://stackoverflow.com/questions/1742866/compute-crc-of-file-in-python
    '''
    name = 'crc32'
    digest_size = 4
    block_size = 1

    def __init__(self, arg=''):
        self.__digest = 0
        self.update(arg)

    def copy(self):
        copy = super(self.__class__, self).__new__(__class__)
        copy.__digest = self.__digest
        return copy

    def digest(self):
        return self.__digest

    def hexdigest(self):
        return '{:08x}'.format(self.__digest)

    def update(self, arg):
        self.__digest = zlib.crc32(arg, self.__digest) & 0xffffffff


```

:emphasize-lines:

Additionally, an `emphasize-lines` option can be given to have Pygments emphasize particular lines:

```
.. code-block:: python
   :emphasize-lines: 3,5

   def some_function():
       interesting = False
       print 'This line is highlighted.'
       print 'This one is not...'
       print '...but this one is.'
```

```
def some_function():
    interesting = False
    print 'This line is highlighted.'
    print 'This one is not...'
    print '...but this one is.'
```

Changed in version 1.1: `emphasize-lines` has been added.

`:lines:` xxx

Alternately, you can specify exactly which lines to include by giving a `lines` option:

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

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

You can combine `lines` with *:emphasize-lines:*

`:encoding:` latin-1

See also:

http://sphinx-doc.org/config.html#confval-source_encoding

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

```
.. literalinclude:: example.py
   :language: python
   :encoding: latin-1
```

`:pyobject:` xxx

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

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

This would only include the code lines belonging to the `start()` method in the `Timer` class within the file.

```
:language:  python
```

See also:

- http://pygments.org/docs/lexers/
- http://sphinx-doc.org/config.html#confval-highlight_language
- http://sphinx-doc.org/markup/code.html#code-examples

The valid values for the highlighting language are:

- none (no highlighting)
- python (the default when highlight_language isn't set)
- guess (let Pygments guess the lexer based on contents, only works with certain well-recognizable languages)
- rest
- c
- ... and any other lexer name that Pygments supports.

Documenting parameters

See also:

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

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

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

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

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

returns

out1 something

out2 something else

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

It would be great to be able to write:

parameters

in1 description of in1

in2 description of in2

keywords

kw1 description of kw1

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

Thanks.

Example documenting parameters

Constants

COMMAND_SUCCESS

The CR_S_SUCCESS value is returned when a command is successfull.

Value	Name	French Message
0x00000000L	COMMAND_SUCCESS	Opération réussie.

List of error codes

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

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

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

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

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

CR_RFID_VerifierPIN

DWORD **CR_RFID_VerifierPIN**(ST_CODE_PIN **pCodePIN*)

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

Paramètres

entree

pCodePIN le code PIN à présenter au lecteur RFID.

Returns

error *see the error codes*

success *COMMAND_SUCCESS*

toctree

See also:

http://sphinx-doc.org/latest/markup/toctree.html

```
de  Luc Saffre <luc.saffre@gmail.com>
heure de l'expéditeur   Envoyé à 14:26 (GMT+02:00). Heure locale : 17:00.
répondre à  sphinx-dev@googlegroups.com
à   sphinx-dev <sphinx-dev@googlegroups.com>
date    18 février 2011 14:26
objet   [sphinx-dev] toctree glob ordering
liste de diffusion  <sphinx-dev.googlegroups.com> Filtrer les messages de cette liste␣
↪de diffusion
```

Hi,

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

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

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

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

  ?
  ??
```

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

In short: Sphinx is great :-)

Paragraphs

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

Image

See also:

- http://sphinx-doc.org/latest/domains.html#directive-rst:role

- http://docutils.sourceforge.net/docs/ref/rst/directives.html#images

image

An "image" is a simple picture:

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

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

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

For example:

```
.. image:: picture.jpeg
   :height: 100px
   :width: 200 px
   :scale: 50 %
   :alt: alternate text
   :align: right
```

Use:

```
.. image:: ../images/wiki_logo_openalea.png
```

to put an image

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

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

figure

Contents

Description

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

    figure are like images but with a caption

    and whatever else you wish to add

    .. code-block:: python

        import image
```

gives

Fig. 2: figure are like images but with a caption
and whatever else youwish to add

```
import image
```

Size in points (PDF constraints)

See also:

- http://docs.opencv.org/doc/tutorials/introduction/how_to_write_a_tutorial/how_to_write_a_tutorial.html

```
.. tabularcolumns:: m{100pt} m{300pt}
.. cssclass:: toctableopencv
=============== ========================================================
 |MatBasicIma|  **Title:** :ref:`matTheBasicImageContainer`
                *Compatibility:* > OpenCV 2.0
                *Author:* |Author_BernatG|
                You will learn how to store images in the memory and how to print out␣
↪their content to the console.
=============== ========================================================
.. |MatBasicIma| image:: images/matTheBasicImageStructure.jpg
                  :height: 90pt
                  :width:  90pt
```

We use the **point measurement** system because we are also creating PDFs.

PDFs are printable documents, where there is no such thing that pixels (px), just points (pt).

And while generally space is no problem for web pages (we have monitors with huge resolutions) the size of the paper (A4 or letter) is constant and will be for a long time in the future.

Therefore, size constrains come in play more like for the PDF, than the generated HTML code.

Geoserver example

See also:

- http://docs.geoserver.org/trunk/en/docguide/sphinx.html#images

Add images to your documentation when *possible*.

Images, such as screenshots, are a very helpful way of making **documentation understandable**.

When making screenshots, try to crop out unnecessary content (browser window, desktop, etc).

Avoid scaling the images, as the Sphinx theme automatically resizes large images.

It is also helpful to include a caption underneath the image.

Fig. 3: *The GeoServer logo as shown on the homepage.*

This image is generated by the following code:

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

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

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

kfigure : Scalable figure and image handling (needs linuxdoc extension, 2016-2017)

See also:

- https://github.com/return42/linuxdoc
- https://return42.github.io/linuxdoc/

Contents

Requirements

We have to *install linuxdoc extension*.

Introduction

The LinuxDoc brings the `kfigure` Sphinx extension which implements scalable image handling. The build for image formats depend on image's source format and output's destination format. This extension implement methods to simplify image handling from the author's POV. Directives like `kernel-figure` implement methods *to* always get the best output-format even if some tools are not installed.

- `.. kernel-image`: for image handling / a `.. image::` replacement
- `.. kernel-figure`: for figure handling / a `.. figure::` replacement
- `.. kernel-render`: for render markup / a concept to embed *render* markups (or languages). Supported markups:
  - `DOT`: render embedded Graphviz's **DOC**
  - `SVG`: render embedded Scalable Vector Graphics (**SVG**)
  - *... developable*

Used tools:

- `dot(1)`: Graphviz (http://www.graphviz.org). If Graphviz is not available, the DOT language is inserted as literal-block.
- SVG to PDF: To generate PDF, you need at least one of this tools:
  - `convert(1)`: ImageMagick (https://www.imagemagick.org)

List of customizations:

- generate PDF from SVG / used by PDF (LaTeX) builder
- generate SVG (html-builder) and PDF (latex-builder) from DOT files. DOT: see http://www.graphviz.org/content/dot-language

Figures & images

If you want to add an image, you should use the `kernel-figure` and `kernel-image` directives. E.g. to insert a figure with a scalable image format use SVG (svg_image_example):

```
.. kernel-figure::  svg_image.svg
   :alt:    simple SVG image

   SVG image example
```

The kernel figure (and image) directive support **DOT** formated files, see

- DOT: http://graphviz.org/pdf/dotguide.pdf
- Graphviz: http://www.graphviz.org/content/dot-language

A simple example (hello_dot_file):

```
.. kernel-figure::  hello.dot
   :alt:    hello world

   DOT's hello world example
```

Embed *render* markups (or languages) like Graphviz's **DOT** is provided by the `kernel-render` directives.:

```
.. kernel-render:: DOT
   :alt: foobar digraph
   :caption: Embedded **DOT** (Graphviz) code

   digraph foo {
    "bar" -> "baz";
   }
```

How this will be rendered depends on the installed tools. If Graphviz is installed, you will see an vector image. If not the raw markup is inserted as *literal-block* (hello_dot_render).

The *render* directive has all the options known from the *figure* directive, plus option `caption`. If `caption` has a value, a *figure* node is inserted. If not, a *image* node is inserted. A `caption` is also needed, if you want to refer it (hello_svg_render).

Embedded **SVG**:

```
.. kernel-render:: SVG
   :caption: Embedded **SVG** markup
   :alt: so-nw-arrow

   <?xml version="1.0" encoding="UTF-8"?>
   <svg xmlns="http://www.w3.org/2000/svg"
       version="1.1" baseProfile="full" width="70px" height="40px" viewBox="0 0 700␣
↪400">
       <line x1="180" y1="370" x2="500" y2="50" stroke="black" stroke-width="15px"/>
       <polygon points="585 0 525 25 585 50" transform="rotate(135 525 25)"/>
   </svg>
```

Substitution

Contents

Substitutions

Substitutions syntax is

```
.. |biohazard| image:: biohazard.png

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

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

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

I really like |doctest|.
```

Which renders like this:

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

I really like `doctest`.

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

Lists

Contents

Description

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

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

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

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

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

```
A. First item
B. Second item
```

Nested lists

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

```
* this is
* a list

  * with a nested list
  * and some subitems

* and here the parent list continues
```

Definition lists

Definition lists are created as follows:

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

   and can even consist of multiple paragraphs

next term
   Description.
```

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

list-table

See also:

- *list-table*

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

Source Code

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

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

   It is not processed in any way, except
   that the indentation is removed.

   It can span multiple lines.

This is a normal text paragraph again.
```

The handling of the `::` marker is smart:

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

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

include

See also:

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

The syntax:

```
.. include myfile.rst
```

Will 'inline' the given file. A common convention I use is create a global .rst file called `global.rst` and include that at the top of every page.

Very useful for links to common images or common files links, etc.

code-block (pygments, highlight)

See also:

- http://sphinx-doc.org/latest/markup/code.html#index-0
- http://pygments.org/docs/lexers/

Sphinx does syntax highlighting using the Pygments library.

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

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

    Some python code.

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

Default highlighter

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

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

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

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

python highlighter

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

```
.. code-block:: python

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

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

xml highlighter

For example, to specify XML:

```
.. code-block:: xml

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

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

console highlighter

. . . or UNIX shell:

```
.. code-block:: console

   # A comment
   sh myscript.sh
```

```
# A comment
sh myscript.sh
```

ini highlighter

. . . or a buildout.cfg:

```
.. code-block:: ini

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

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

pycon python console highlighter

... or interactive Python:

```
.. code-block:: pycon

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

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

highlighting mode for the whole document

Setting the highlighting mode for the whole document:

```
.. highlight:: console
```

All code blocks in this doc use console highlighting by default:

```
some shell commands
```

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

The full list of lexers and associated short names is here: http://pygments.org/docs/lexers/

rest Tables

csv tables

See also:

- http://docutils.sourceforge.net/docs/ref/rst/directives.html#csv-table

```
.. csv-table:: Balance Sheet
   :header: Description,In,Out,Balance
   :widths: 20, 10, 10, 10
```

(continues on next page)

(continued from previous page)

```
   :stub-columns: 1

   Travel,,230.00,-230.00
   Fees,,400.00,-630.00
   Grant,700.00,,70.00
   Train Fare,,70.00,**0.00**
```

Gives:

Table 1: Balance Sheet

Description	In	Out	Balance
Travel		230.00	-230.00
Fees		400.00	-630.00
Grant	700.00		70.00
Train Fare		70.00	**0.00**

Rest grid tables

See also:

- petl_tables_grid
- http://docutils.sourceforge.net/docs/ref/rst/restructuredtext.html#grid-tables
- http://rest-sphinx-memo.readthedocs.org/en/latest/ReST.html#tables

Two forms of tables are supported.

For *grid tables* you have to "paint" the cell grid yourself.

They look like this:

```
+------------------------+------------+----------+----------+
| Header row, column 1   | Header 2   | Header 3 | Header 4 |
| (header rows optional) |            |          |          |
+========================+============+==========+==========+
| body row 1, column 1   | column 2   | column 3 | column 4 |
+------------------------+------------+----------+----------+
| body row 2             | ...        | ...      |          |
+------------------------+------------+----------+----------+
```

Header row, column 1 (header rows optional)	Header 2	Header 3	Header 4
body row 1, column 1	column 2	column 3	column 4
body row 2	...	...	

Examples

```
+---------------------------------+------------+
| Condition                       | Decision   |
+=================================+============+
| comparison score >= threshold   | Match      |
+---------------------------------+------------+
```

(continues on next page)

(continued from previous page)

```
| comparison score < threshold    | Non-Match  |
+---------------------------------+------------+
```

Result

Condition	Decision
comparison score >= threshold	Match
comparison score < threshold	Non-Match

list-table

See also:

- http://sfepy.org/doc-devel/developer_guide.html#sfepy-directory-structure
- petl_examples
- *rest Tables*

Contents

Description

Bulleted lists can sometimes be cumbersome and hard to follow.

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

Exemple

See also:

- http://sfepy.org/doc-devel/developer_guide.html#sfepy-directory-structure

Exemple 1

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

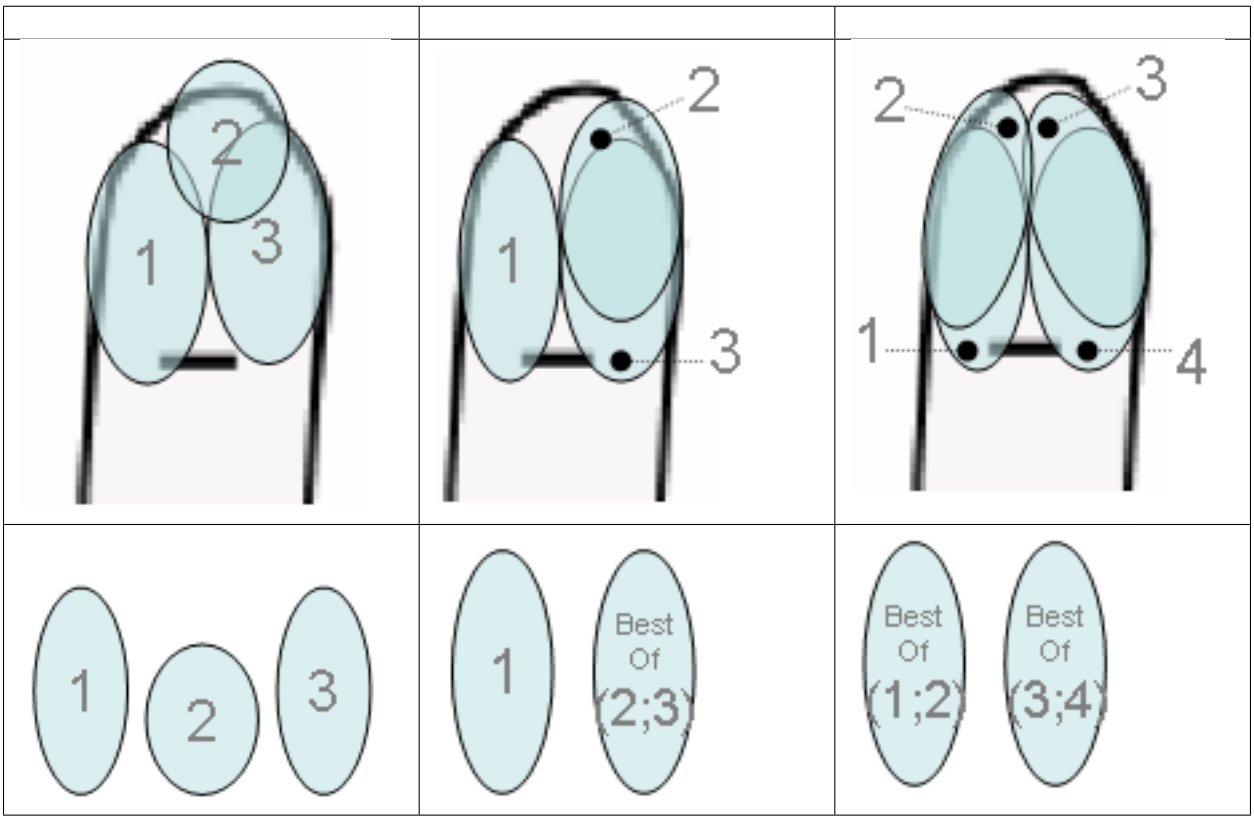
Shapes	Description
Square	Four sides of equal length, 90 degree angles
Rectangle	Four sides, 90 degree angles

This is done with the following code:

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

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

Example 2



This is done with the following code:

```
.. list-table::
   :widths: 33 33 33
   :header-rows: 1
```

(continues on next page)

(continued from previous page)

```
   * -
     -
     -

   * - .. image:: image_selection/up_image1.png
     - .. image:: image_selection/up_image2.png
     - .. image:: image_selection/up_image3.png

   * - .. image:: image_selection/down_image4.png
     - .. image:: image_selection/down_image5.png
     - .. image:: image_selection/down_image6.png
```

Example 3

Error type	Definition	Impact
False Match	Comparison decision of "match" for a biometric probe and a biometric reference that are from different fingers	System security lack
False Non-Match	Comparison decision of "non-match" for a biometric probe and a biometric reference that are from the same finger	User inconvenience

This is done with the following code:

```
.. list-table::
   :widths: 20 60 20
   :header-rows: 1

   * - Error type
     - Definition
     - Impact

   * - False Match
     - Comparison decision of "match" for a biometric probe and a biometric
       reference that are from different fingers
     - System security lack

   * - False Non-Match
     - Comparison decision of "non-match" for a biometric probe and a biometric
       reference that are from the same finger
     - User inconvenience
```

flat-table (needs linuxdoc extension, 2016-2017)

See also:

- https://github.com/return42/linuxdoc
- LinuxDoc flat table
- https://return42.github.io/linuxdoc/linuxdoc-howto/table-markup.html#rest-flat-table

Contents

Requirements

We have to *install linuxdoc extension*.

flat-table

The `flat-table` (`FlatTable`) is a double-stage list similar to the `list-table` with some additional features:

- *column-span*: with the role `cspan` a cell can be extended through additional columns
- *row-span*: with the role `rspan` a cell can be extended through additional rows
- *auto-span* rightmost cell of a table row over the missing cells on the right side of that table-row. With Option `:fill-cells:` this behavior can changed from *auto span* to *auto fill*, which automatically inserts (empty) cells instead of spanning the last cell.

options:

header-rows [int] count of header rows

stub-columns [int] count of stub columns

widths [[int] [int] ...] widths of columns

fill-cells instead of auto-span missing cells, insert missing cells

roles:

cspan [int] additional columns (*morecols*)

rspan [int] additional rows (*morerows*)

The example below shows how to use this markup. The first level of the staged list is the *table-row*. In the *table-row* there is only one markup allowed, the list of the cells in this *table-row*. Exception are *comments* (`..`) and *targets* (e.g. a ref to row 2 of table's body).

```
.. flat-table:: table title
   :header-rows: 2
   :stub-columns: 1
   :widths: 1 1 1 1 2

   * - :rspan:`1` head / stub
     - :cspan:`3` head 1.1-4

   * - head 2.1
     - head 2.2
     - head 2.3
     - head 2.4

   * .. row body 1 / this is a comment
```

(continues on next page)

(continued from previous page)

```
    - row 1
    - :rspan:`2` cell 1-3.1
    - cell 1.2
    - cell 1.3
    - cell 1.4

  * .. Comments and targets are allowed on *table-row* stage.
    .. _`row body 2`:

    - row 2
    - cell 2.2
    - :rspan:`1` :cspan:`1`
      cell 2.3 with a span over

      * col 3-4 &
      * row 2-3

  * - row 3
    - cell 3.2

  * - row 4
    - cell 4.1
    - cell 4.2
    - cell 4.3
    - cell 4.4

  * - row 5
    - cell 5.1 with automatic span to rigth end

  * - row 6
    - cell 6.1
    - ..
```

Rest Simple tables

See also:

- petl_tables_minimal

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

They look like this:

```
=====  =====  =======
A      B      A and B
=====  =====  =======
False  False  False
True   False  False
False  True   False
True   True   True
=====  =====  =======
```

A	B	A and B
False	False	False
True	False	False
False	True	False
True	True	True

sphinx Hyperlinks

Contents

External links

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

```
Example of external link:  `reST role <http://sphinx-doc.org/latest/markup/inline.html
↪#role-ref>`_ .
```

Example of external link: reST role .

Include HTML in generated Sphinx docs

See also:

- http://sphinx-doc.org/ext/intersphinx.html
- https://gist.github.com/3978232

```
Takayuki Shimizukawa shimizukawa@gmail.com
répondre à:  sphinx-dev@googlegroups.com
à:   sphinx-dev@googlegroups.com
date:    30 octobre 2012 05:22
objet:   Re: [sphinx-dev] Include HTML in generated Sphinx docs
```

intersphinx will meet your needs.

intersphinx support to link another 'Sphinx Document' by using inventory like 'objects.inv'.

But if you generate inventory by hand (or some program), you can use this mechanism. I wrote a sample: https://gist.github.com/3978232

intersphinx reference is here: http://sphinx-doc.org/ext/intersphinx.html

Best regards,

```
Takayuki SHIMIZUKAWA
Sphinx-users.jp
```

conf.py

```
extensions = ['sphinx.ext.intersphinx']

intersphinx_mapping = {
    'javaapi': ('http://api.example.com/', 'javaapi.inv'),
}
```

generate_javaapi_inv.py

```
inventory_header = u'''\
# Sphinx inventory version 2
# Project: javaapi
# Version: 2.0
# The remainder of this file is compressed with zlib.
'''.encode('utf-8')

inventory_payload = u'''\
api1 std:label -1 api.html#api1 API 1
'''.encode('utf-8')

# inventory_payload lines spec:
#   name domainname:type priority uri dispname
#
# * `name`     -- fully qualified name
# * `dispname` -- name to display when searching/linking
# * `type`     -- object type, a key in ``self.object_types``
# * `docname`  -- the document where it is to be found
# * `anchor`   -- the anchor name for the object
# * `priority` -- how "important" the object is (determines placement in search␣
↪results)
#
#   - 1: default priority (placed before full-text matches)
#   - 0: object is important (placed before default-priority objects)
#   - 2: object is unimportant (placed after full-text matches)
#   - -1: object should not show up in search at all
#

inventory = inventory_header + zlib.compress(inventory_payload)
open('javaapi.inv','wb').write(inventory)
```

index.html

```
<div class="section" id="example-link-to-outer-non-sphinx-by-using-intersphinx">
<h1>Example: Link to outer non-sphinx by using intersphinx<a class="headerlink" href="
↪#example-link-to-outer-non-sphinx-by-using-intersphinx" title="Permalink to this␣
↪headline"></a></h1>
<p><a class="reference external" href="http://api.example.com/api.html#api1" title=
↪"(in javaapi v2.0)"><em class="xref std std-ref">Java API 1</em></a></p>
```

(continues on next page)

(continued from previous page)

```
</div>
```

index.txt

```
Example: Link to outer non-sphinx by using intersphinx
======================================================================

:ref:`Java API 1 <javaapi:api1>`
```

Internal links

See also:

http://sphinx-doc.org/latest/markup/inline.html#role-ref

Internal linking is done via a special reST role .

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

link to internal links

Sphinx Sections

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

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

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

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

Explicit Markup

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

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

Directives

See also:

http://sphinx-doc.org/latest/domains.html#directive-rst:role

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

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

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

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

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

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

Footnotes

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

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

.. rubric:: Footnotes

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

You can also explicitly number the footnotes for better context.

Comments

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

Source encoding

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

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

Gotchas

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

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

Explicit Markup

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

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

Sphinx Directives

See also:

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

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

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

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

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

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

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

Sphinx Footnotes

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

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

.. rubric:: Footnotes

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

You can also explicitly number the footnotes for better context.

Comments

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

Source encoding

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

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

sidebar

See also:

http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/source/sphinx/rest_syntax.html#sidebar-directive

t is possible to create sibar

Sidebar Title

Optional Sidebar Subtitle

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

using the following code:

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

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

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

Gotchas

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

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

Sphinx extensions and contributed extensions

Sphinx extensions

See also:

- http://sphinx-doc.org/extensions.html

autodoc sphinx extension

See also:

- http://sphinx-doc.org/ext/autodoc.html

This extension can import the modules you are documenting, and pull in documentation from docstrings in a semi-automatic way.

Note: For Sphinx (actually, the Python interpreter that executes Sphinx) to find your module, it must be importable. That means that the module or the package must be in one of the directories on `sys.path` – adapt your `sys.path` in the configuration file accordingly.

For this to work, the docstrings must of course be written in correct reStructuredText. You can then use all of the usual Sphinx markup in the docstrings, and it will end up correctly in the documentation. Together with hand-written documentation, this technique eases the pain of having to maintain two locations for documentation, while at the same time avoiding auto-generated-looking pure API documentation.

autogen sphinx extension

See also:

- http://sphinx-doc.org/ext/autosummary.html#sphinx-autogen-generate-autodoc-stub-pages

The **`sphinx-autogen`** script can be used to conveniently generate stub documentation pages for items included in `autosummary` listings.

For example, the command

```
$ sphinx-autogen -o generated *.rst
```

will read all `autosummary` tables in the `*.rst` files that have the `:toctree:` option set, and output corresponding stub pages in directory `generated` for all documented items. The generated pages by default contain text of the form:

```
sphinx.util.relative_uri
========================

.. autofunction:: sphinx.util.relative_uri
```

If the `-o` option is not given, the script will place the output files in the directories specified in the `:toctree:` options.

Generating stub pages automatically

If you do not want to create stub pages with **`sphinx-autogen`**, you can also use this new config value:

```
.. confval:: autosummary_generate
```

Boolean indicating whether to scan all found documents for autosummary directives, and to generate stub pages for each.

Can also be a list of documents for which stub pages should be generated.

The new files will be placed in the directories specified in the `:toctree:` options of the directives.

sphinx.ext.napoleon (Marching toward legible docstrings)

See also:

- http://sphinxcontrib-napoleon.readthedocs.io/en/latest/
- http://www.sphinx-doc.org/en/stable/ext/napoleon.html

Contents

Description

Are you tired of writing docstrings that look like this:

```
:param path: The path of the file to wrap
:type path: str
:param field_storage: The :class:`FileStorage` instance to wrap
:type field_storage: FileStorage
:param temporary: Whether or not to delete the file when the File
   instance is destructed
:type temporary: bool
:returns: A buffered writable file descriptor
:rtype: BufferedFileStorage
```

ReStructuredText is great, but it creates visually dense, hard to read docstrings. Compare the jumble above to the same thing rewritten according to the Google Python Style Guide:

```
Args:
    path (str): The path of the file to wrap
    field_storage (FileStorage): The :class:`FileStorage` instance to wrap
    temporary (bool): Whether or not to delete the file when the File
       instance is destructed

Returns:
    BufferedFileStorage: A buffered writable file descriptor
```

Much more legible, no?

Napoleon is a Sphinx extension that enables Sphinx to parse both NumPy and Google style docstrings - the style recommended by Khan Academy.

Napoleon is a pre-processor that parses NumPy and Google style docstrings and converts them to reStructuredText before Sphinx attempts to parse them. This happens in an intermediate step while Sphinx is processing the documentation, so it doesn't modify any of the docstrings in your actual source code files.

Getting Started

1. After setting up Sphinx to build your docs, enable napoleon in the Sphinx *conf.py* file:

```
# conf.py

# Add autodoc and napoleon to the extensions list
extensions = ['sphinx.ext.autodoc', 'sphinxcontrib.napoleon']
```

2. Use *sphinx-apidoc* to build your API documentation:

```
$ sphinx-apidoc -f -o docs/source projectdir
```

Docstrings

Napoleon interprets every docstring that Sphinx autodoc can find, including docstrings on: `modules`, `classes`, `attributes`, `methods`, `functions`, and `variables`. Inside each docstring, specially formatted *Sections* are parsed and converted to reStructuredText.

All standard reStructuredText formatting still works as expected.

Docstring Sections

All of the following section headers are supported:

- `Args` *(alias of Parameters)*
- `Arguments` *(alias of Parameters)*
- `Attributes`
- `Example`
- `Examples`
- `Keyword Args` *(alias of Keyword Arguments)*
- `Keyword Arguments`
- `Methods`
- `Note`
- `Notes`
- `Other Parameters`
- `Parameters`
- `Return` *(alias of Returns)*
- `Returns`
- `Raises`
- `References`
- `See Also`
- `Warning`
- `Warnings` *(alias of Warning)*

- Warns
- Yields

Google vs NumPy

Napoleon supports two styles of docstrings: Google and NumPy. The main difference between the two styles is that Google uses indention to separate sections, whereas NumPy uses underlines.

Google style:

```
def func(arg1, arg2):
    """Summary line.

    Extended description of function.

    Args:
        arg1 (int): Description of arg1
        arg2 (str): Description of arg2

    Returns:
        bool: Description of return value

    """
    return True
```

NumPy style:

```
def func(arg1, arg2):
    """Summary line.

    Extended description of function.

    Parameters
    ----------
    arg1 : int
        Description of arg1
    arg2 : str
        Description of arg2

    Returns
    -------
    bool
        Description of return value

    """
    return True
```

NumPy style tends to require more vertical space, whereas Google style tends to use more horizontal space. Google style tends to be easier to read for short and simple docstrings, whereas NumPy style tends be easier to read for long and in-depth docstrings.

The Khan Academy recommends using Google style.

The choice between styles is largely aesthetic, but the two styles should not be mixed. Choose one style for your project and be consistent with it.

For full documentation see http://sphinxcontrib-napoleon.readthedocs.org

Contributed sphinx extensions

See also:

- https://bitbucket.org/birkenfeld/sphinx-contrib
- https://bitbucket.org/birkenfeld/sphinx/wiki/Home

Sphinx autorun

See also:

- https://github.com/thewtex/sphinx-contrib/tree/master/autorun
- http://perso.crans.org/besson/runblock.html

Contents

Description

`Autorun` is an extension for *Sphinx* that can execute the code from a runblock directive and attach the output of the execution to the document.

For example:

```
.. runblock:: pycon

    >>> for i in range(5):
    ...     print i
```

Produces:

```
>>> for i in range(5):
...     print i
1
2
3
4
5
```

Another example:

```
.. runblock:: console

    $ date
```

Produces:

```
$ date
Thu  4 Mar 2010 22:56:49 EST
```

Currently autorun supports `pycon` and `console` languages. It's also possible to configure autorun (from *conf.py*) to run other languages.

Installation

Installing from sources:

```
$ hg clone http://bitbucket.org/birkenfeld/sphinx-contrib/
$ cd sphinx-contrib/autorun
$ python setup.py install
```

To enable autorun add 'sphinxcontrib.autorun' to the `extension` list in *conf.py*:

```
extensions.append('sphinxcontrib.autorun')
```

The documentation is in the doc/ folder.

Exemples

See also:

- http://perso.crans.org/besson/runblock.html

Blockdiag (simple diagram images generator)

See also:

- http://blockdiag.com/
- http://interactive.blockdiag.com/
- http://www.slideshare.net/TakeshiKomiya/blockdiag-a-simple-diagram-generator
- http://blockdiag.com/en/blockdiag/sphinxcontrib.html
- https://pypi.python.org/pypi/sphinxcontrib-blockdiag
- https://bitbucket.org/birkenfeld/sphinx-contrib/src/e60f176286fe/blockdiag/setup.py
- http://mentors.debian.net/package/sphinxcontrib-blockdiag
- https://twitter.com/#!/tk0miya

Contents

Introduction

Once again thanks a lot for your prompt replies. In case you are curious or you ever have to face the same problem than I, I finally chose blockdiag.

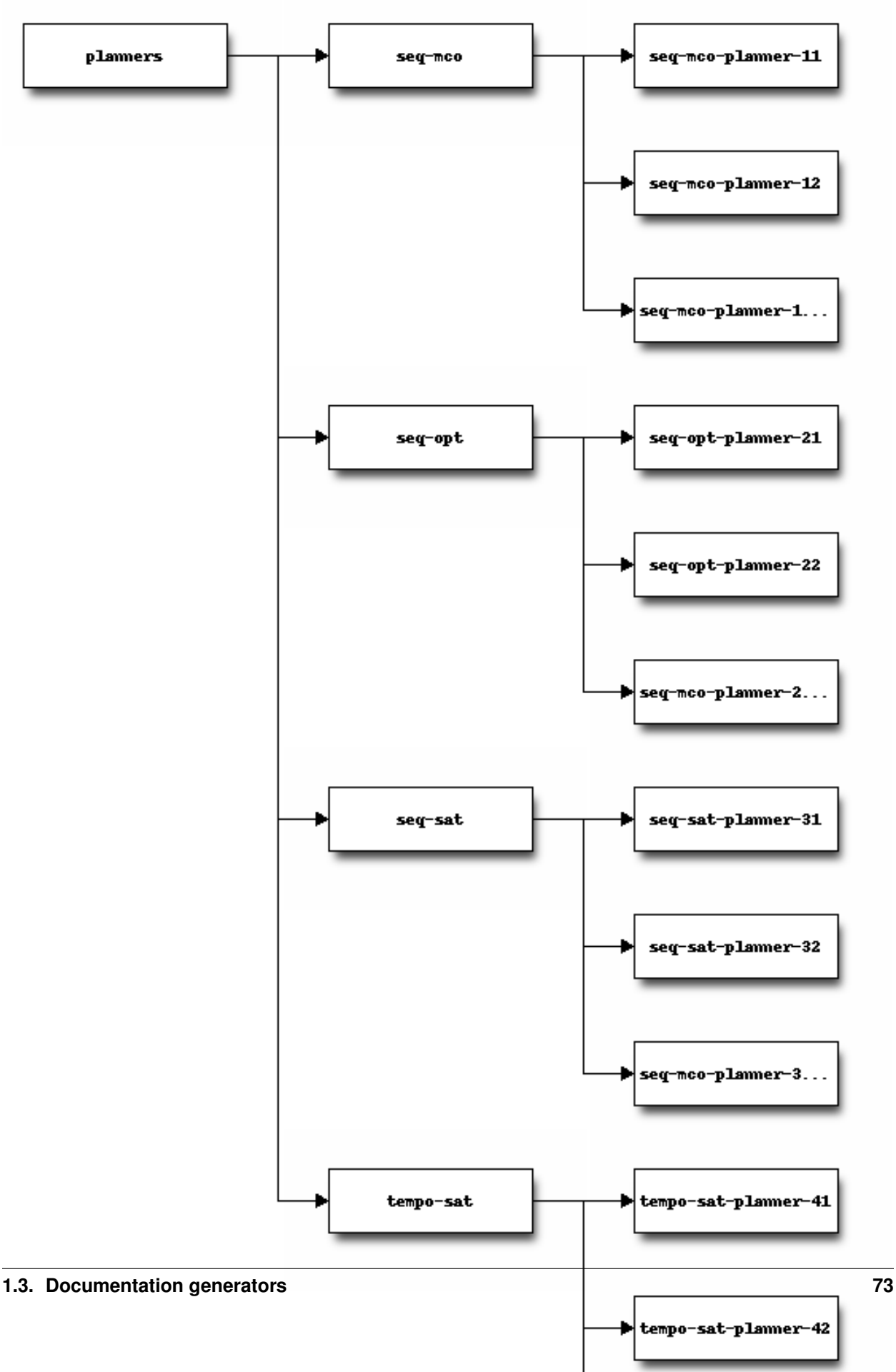
With only a few statements:

```
.. blockdiag::

   {
      planners -> seq-mco -> seq-mco-planner-11;
      planners -> seq-mco -> seq-mco-planner-12;
      planners -> seq-mco -> "seq-mco-planner-1...";
      planners -> seq-opt -> seq-opt-planner-21;
      planners -> seq-opt -> seq-opt-planner-22;
      planners -> seq-opt -> "seq-mco-planner-2...";
      planners -> seq-sat -> seq-sat-planner-31;
      planners -> seq-sat -> seq-sat-planner-32;
      planners -> seq-sat -> "seq-mco-planner-3...";
      planners -> tempo-sat -> tempo-sat-planner-41;
      planners -> tempo-sat -> tempo-sat-planner-42;
      planners -> tempo-sat -> "seq-mco-planner-4...";
   }
```

I could generate the attached figure and embed it in the html and pdf docs generated with sphinx.

Just awesome!!


planners
seq-mco
seq-mco-planner-11
seq-mco-planner-12
seq-mco-planner-1...
seq-opt
seq-opt-planner-21
seq-opt-planner-22
seq-mco-planner-2...
seq-sat
seq-sat-planner-31
seq-sat-planner-32
seq-mco-planner-3...
tempo-sat
tempo-sat-planner-41
tempo-sat-planner-42

blockdiag pycon-japan-retrospective

See also:

https://tarekziade.wordpress.com/2011/09/05/pycon-japan-retrospective/

Komiya Takeshi showed me his tool called blockdiag, which is a DSL you can use to add diagrams in your documentation. The nice thing is that it provides a Sphinx extension so you can add diagrams in your documentation through simple expressions, and have Sphinx generate for you the diagrams on the fly.

There's even an interactive online shell: http://interactive.blockdiag.com/

I've challenged Komiya to write a few diagrams I have for some Mozilla projects using his tool, and it took a few seconds for him to build them. So, I am going to use this in the future.

Doxygen contributed extensions

See also:

- http://www.stack.nl/~dimitri/doxygen/
- http://pcsclite.alioth.debian.org/api/index.html

doxygen installation on GNU/Linux

Contents

Prerequisite

We must be **root** to install the doxygen library (for centos).

Under ubuntu, we must be root only for make install.

Install in /opt/doxygen/1.7.2

cd <yourpath>/doxygen-1.7.2

./configure –prefix /opt/doxygen/1.7.2 –with-doxywizard

Results

```
cd /tmp/doxygen-1.7.2
./configure --prefix /opt/doxygen/1.7.2 --with-doxywizard
make
```

other option for configure: –enable-debug-log –enable-examples-build

sudo make install

Make the link to the current doxygen version

```
su - root
cd /opt/doxygen
ln -s 1.7.2 current
```

export PATH=/opt/doxygen/current/bin:$PATH

Breathe sphinx extension

See also:

- https://github.com/michaeljones/breathe
- http://breathe.readthedocs.org/en/latest/index.html

Contents

Description

This is an extension to restructured text and Sphinx to be able to read and render the Doxygen xml output.

It is an easy way to include Doxygen information in a set of documentation generated by Sphinx.

The aim is to produce an autodoc like support for people who enjoy using Sphinx but work with languages other than Python. The system relies on the Doxygen's xml output.

Gammu breathe example

See also:

- http://wammu.eu/docs/manual/index.html

Doxylink

See also:

- https://pythonhosted.org/sphinxcontrib-doxylink/
- https://github.com/thewtex/sphinx-contrib/tree/master/doxylink

Contents

Description

A Sphinx extension to link to external Doxygen API documentation.

Usage

Please refer to the documentation for information on using this extension.

Installation

This extension can be installed from the Python Package Index:

```
pip install sphinxcontrib-doxylink
```

Alternatively, you can clone the sphinx-contrib repository from BitBucket, and install the extension directly from the repository:

```
hg clone http://bitbucket.org/birkenfeld/sphinx-contrib
cd sphinx-contrib/doxylink
python setup.py install
```

Doxylink Examples

Doxylink Examples

Cantera

See also:

https://cantera.github.io/docs/sphinx/html/index.html

Pointclouds

See also:

- http://www.pointclouds.org/documentation/

Shark

See also:

- http://image.diku.dk/shark/sphinx_pages/build/html/index.html
- http://image.diku.dk/shark/sphinx_pages/build/html/rest_sources/tutorials/for_developers/managing_the_documentation.html

Contents

Linking to doxygen

See also:

- http://image.diku.dk/shark/sphinx_pages/build/html/rest_sources/tutorials/for_developers/writing_tutorials.html#linking-to-doxygen
- http://image.diku.dk/shark/doxygen_pages/html/group__shark__globals.html#ga100af03d8327fc61cf0a2f54501b67a4

Here's a doxylink:

```
:doxy:`choleskyDecomposition`
```

integrate doxygen documentation in sphinx

There is an easy way to include Doxygen information in a set of documentation generated by Sphinx.

Create 2 directories for sphinx and doxygen

Copy the doxygen output files in _build/html/_downloads directory

copy_doxygen_doc.bat

```
Télécharger le fichier de commandes DOS
```

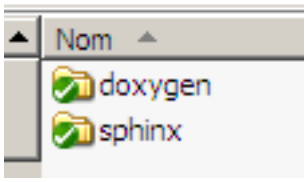


Fig. 4: *Directories for sphinx and doxygen*

```
python copytree_doxygen.py

pause
```

Copy with python shutil.rmtree and shutil.copytree

Télécharger le fichier python

```
# -*- coding: utf-8 -*-
"""
Copie des données doxygen.

Les données générées par doxygen sont dans le répertoire doxygen/html

Il faut copier ces données dans le répertoire sphinx/_build/html/_downloads

Précondition: le répertoire destination doit être vide.

.. function:: copytree(src, dst[, symlinks=False[, ignore=None]])

   Recursively copy an entire directory tree rooted at *src*.  The destination
   directory, named by *dst*, must not already exist; it will be created as well
   as missing parent directories.  Permissions and times of directories are
   copied with :func:`copystat`, individual files are copied using
   :func:`copy2`.


.. function:: rmtree(path[, ignore_errors[, onerror]])

   .. index:: single: directory; deleting

   Delete an entire directory tree; *path* must point to a directory (but not a
   symbolic link to a directory).  If *ignore_errors* is true, errors resulting
   from failed removals will be ignored; if false or omitted, such errors are
   handled by calling a handler specified by *onerror* or, if that is omitted,
   they raise an exception.

"""

from shutil import copytree, rmtree


destination = '_build/html/_downloads'
```

(continues on next page)

(continued from previous page)

```
# le répertoire destination doit être vide
rmtree(destination)

# copie vers le répertoire destination
copytree(src='../doxygen/html', dst=destination)
```

Download the index.html source file from sphinx documents

First sphinx document

```
Le but de la bibliothèque logicielle est de proposer une API C permettant la
communication entre:

- l'application :term:`EUCLIDE`
- un ensemble de 40 :term:`boîtiers`
- un :term:`lecteur de cartes à puces` (RFID)
- une :term:`clé RF`


:download:`Télécharger  la documentation doxygen du projet CR_Euclide <../doxygen/
↪html/index.html>`.
```

Second sphinx document

```
.. _doxygen_documentation:

=======================
Doxygen Documentation
=======================

:download:`Télécharger  la documentation doxygen du projet CR_Euclide <../../../
↪doxygen/html/index.html>`.
```

Tree docs directory

```
+---doxygen
|   |   TagFile.xml
|   |
|   +---def
|   |       doxygen.def
|   |
|   \---html
|       |   annotated.html
|       |   bc_s.png
|       |   classes.html
|       |   class_ryb_eliot_1_1_antennas-members.html
|       |   _write_tag_form_8cs_source.html
|       |   _write_tag_form_8_designer_8cs.html
```

(continues on next page)

(continued from previous page)

```
|       |   _write_tag_form_8_designer_8cs_source.html
|       |
|
\---sphinx
    |   index.rst
    |   make.bat
    |   Makefile
    |
    |
    +---_build
    |   \---html
    |       +---_downloads
    |       |   |   annotated.html
    |       |   |   bc_s.png
    |       |   |   classes.html
    |       |   |
    |       |   \---search
    |       |           all_5f.html
    |       |           all_61.html
    |
    +---_static
    \---_templates
```

MIT rst tools

See also:

- http://web.mit.edu/kerberos/krb5-1.10/krb5-1.10/doc/rst_tools/

Contents

How to deploy the Doxygen output in Sphinx project

The text below is meant to give the instructions on how to incorporate MIT Kerberos API reference documentation into Sphinx document hierarchy.

The Sphinx API documentation can be constructed :

- with (*Part B*)
- or without (*Part A*) the bridge to the original Doxygen HTML output.

Pre-requisites

- python 2.5+ with Cheetah, lxml and xml extension modules installed;
- For part B only:
  - Sphinx "doxylink" extension;
  - Doxygen HTML output

Part A:

Transforming Doxygen XML output into reStructuredText (rst) without the bridge to Doxygen HTML output.

1. Delete lines containing text "Doxygen reference" from the template files func_document.tmpl and type_document.tmpl;
2. In the Doxygen configuration file set GENERATE_XML to YES. Generate Doxygen XML output;
3. Suppose the Doxygen XML output is located in doxy_xml_dir and the desired output directory is rst_dir. Run:

   ```
   python doxy.py -i  doxy_xml_dir -o rst_dir -t func
   ```

   This will result in the storing of the API function documentation files in rst format in the rst_dir. The file names are constructed based on the function name. For example, the file for krb5_build_principal() will be krb5_build_principal.rst

   Run:

   ```
   python doxy.py -i  doxy_xml_dir -o rst_dir -t typedef
   ```

   It is similar to the API function conversion, but for data types. The result will be stored under rst_dir/types directory

   Alternatively, running:

   ```
   python doxy.py -i  doxy_xml_dir -o rst_dir
   ```

   or:

   ```
   python doxy.py -i  doxy_xml_dir -o rst_dir -t all
   ```

   converts Doxygen XML output into reStructuredText format files both for API functions and data types;
4. In krb_appldev/index.rst add the following section to point to the API references:

   ```
   .. toctree::
       :maxdepth: 1

       refs/index.rst
   ```

5. Copy the content of rst_dir into krb_appldev/refs/api/ directory and rst_dir/types into krb_appldev/refs/types directory;
6. Rebuild Sphinx source:

   ```
   sphinx-build source_dir build_dir
   ```

Part B: Bridge to Doxygen HTML output

1. Transform Doxygen XML output into reStructuredText.

   In src/Doxygen configuration file request generation of the tag file and XML output:

```
GENERATE_TAGFILE       = krb5doxy.tag
GENERATE_XML           = YES
```

2. Modify Sphinx conf.py file to point to the "doxylink" extension and Doxygen tag file:

```
extensions = ['sphinx.ext.autodoc', 'sphinxcontrib.doxylink']
doxylink = { ' krb5doxy' : ('/tmp/krb5doxy.tag, ' doxy_html_dir ') }
```

   where doxy_html_dir is the location of the Doxygen HTML output

3. Continue with steps 3 - 6 of Part A.

Robin : bridge between doxygen (XML) and sphinx via mongodb

See also:

- https://bitbucket.org/reima/robin
- mongodb_database

Contents

Warning: marche sur l'exemple fourni mais pas avec mes projets :)

Robin

We're happy to announce robin, a new Doxygen/C++ to Sphinx bridge.

Robin provides an easy-to-use, easy-to-hack integration of Doxygen documentation into Sphinx.

Robin is licensed under the BSD and can be found at Bitbucket: https://bitbucket.org/reima/robin

Features

- Robust extraction of Doxygen XML data via an easy-to-hack parser
- Intermediate data is stored in a database (mongodb) for simple extraction and processing

- Directive-driven output; each directive provides callbacks and hooks which allows for deep customization
- Automated generation of driver ReST documents: Similar to automodule; however, robin generates actual ReST documents which can be inspected

Prerequisites

Robin expects a running mongodb on the local host.

It uses a minimal set of external libraries: Pymongo, sphinx, progressbar.

All of the dependencies can be easily installed using pip or easy_install.

Robin has been developed with Python 2.7; we have not tested previous versions.

Getting started

- Run Doxygen to generate XML documentation (GENERATE_XML=YES)
- Run extract-doxygen <path to XML> <project name>
- Run create-rst <project name> This generates several directories (classes, groups, etc.) Include the groups.rst into your toc
- Add 'robin.sphinx' to the Sphinx extensions
- Build (make html) for TOC update
- Build again (make clean && make html)

Status

We're using robin internally for a large C++ codebase, and there are a few minor issues left that we hope to resolve soon (all of them are tracked on Bitbucket.)

After that, we expect that robin will go into "maintenance" mode focusing on bug fixes only.

If someone is interested in contributing, please get in touch with us.

Cheers, the robin developers

Sphinx hieroglyph extension

See also:

- https://github.com/nyergler/hieroglyph

Contents

Introduction

hieroglyph is an extension for Sphinx which builds HTML5 slides from ReStructured Text documents.

Installing

You can install Hieroglyph using easy_install or pip:

$ easy_install hieroglyph

Using Hieroglyph

Add Hieroglyph as a Sphinx extension to your configuration:

```
extensions = [
    'hieroglyph',
]
```

Build your slides

```
$ sphinx-build -b slides sourcedir outdir
```

Where sourcedir is the directory containing the Sphinx conf.py file and outdir is where you want your slides to output to.

License

Hieroglyph is made available under a BSD license; see LICENSE for details.

Included slide CSS and JavaScript originally based on HTML 5 Slides licensed under the Apache Public License.

Examples

- https://raw.github.com/AndreaCrotti/pyconuk2012_slides/master/zeromq/zeromq.rst

javasphinx extension

See also:

- https://github.com/bronto/javasphinx

javasphinx is an extension to the Sphinx documentation system which adds support for documenting Java projects.

It includes a Java domain for writing documentation manually and a javasphinx-apidoc utility which will automatically generate API documentation from existing Javadoc markup.

LinuxDoc Sphinx-doc extensions for sophisticated C developer (NEW, 2016-07)

See also:

- https://github.com/return42/linuxdoc
- https://return42.github.io/linuxdoc/

Contents

Introduction

The **LinuxDoc library** contains sphinx-doc extensions and command line tools to extract documentation from C/C++ source file comments.

Even if this project started in context of the Linux-Kernel documentation, you can use these extensions in common sphinx-doc projects.

LinuxDoc is hosted at github: https://github.com/return42/linuxdoc

LinuxDoc feature overview

kernel-doc A "C" friendly markup, the parser, the Sphinx-doc extension and some tools. The kernel-doc markup embeds documentation within the C source files, using a few simple conventions. The parser grabs the documentation from source and generates proper reST [ref]. The parser is written in Python and its API is used by the corresponding Sphinx-doc extension. Command line tools shipped with:

- kernel-autodoc: Suitable for automatic API documentation [ref].
- kernel-lintdoc: *Lint* kernel-doc comments from source code [ref].
- kernel-doc: A command line interface for kernel-doc's parser API [ref].

kernel-doc-man A man page builder. An extension/replacement of the common Sphinx-doc *man* builder also integrated in the kernel-doc Sphinx-doc extension [ref].

flat-table A diff and author friendly list-table replacement with *column-span*, *row-span* and *auto-span* [ref].

cdomain A customized Sphinx's C Domain extension. Suitable for demanding projects [ref].

kfigure Sphinx extension which implements scalable image handling. Simplifies image handling from the author's POV. Wins where Sphinx-doc image handling fails. Whatever graphic tools available on your build host, you always get the best output-format. Graphviz's DOT format included *[ref]*.

kernel-include A replacement for the `include` reST directive. The directive expand environment variables in the path name and allows to include files from arbitrary locations ref:*[ref] <kernel-include-directive>*.

Install LinuxDoc

Install bleeding edge.:

```
pip install git+http://github.com/return42/linuxdoc.git
```

As the LinuxDoc lib evolving constantly, an update should be carried out regularly.:

```
pip install --upgrade git+http://github.com/return42/linuxdoc.git
```

If you are a developer and like to contribute to the LinuxDoc lib, fork on github or clone and make a developer install:

```
git clone https://github.com/return42/linuxdoc
cd linuxdoc
make install
```

Below you see how to integrate the LinuxDoc sphinx extensions into your sphinx build process. In the `conf.py` (sphinx config) add the LinuxDoc extensions:

```
extensions = [
  'linuxdoc.rstFlatTable'      # Implementation of the 'flat-table' reST-directive.
  , 'linuxdoc.rstKernelDoc'    # Implementation of the 'kernel-doc' reST-directive.
  , 'linuxdoc.kernel_include'  # Implementation of the 'kernel-include' reST-
↪directive.
  , 'linuxdoc.manKernelDoc'    # Implementation of the 'kernel-doc-man' builder
  , 'linuxdoc.cdomain'         # Replacement for the sphinx c-domain.
  , 'linuxdoc.kfigure'         # Sphinx extension which implements scalable image␣
↪handling.
]
```

Flat table

See also:

flat-table (needs linuxdoc extension, 2016-2017)

sphinx odt2sphinx extension

See also:

- https://bitbucket.org/cdevienne/odt2sphinx
- open_odf

Transform a OOo Writer document into Sphinx documentation sources

rstspreadsheet sphinx extension

See also:

- http://pypi.python.org/pypi/rstspreadsheet

Add the *spreadsheet* directive to reStructuredText for Docutils and Sphinx

rst2qhc (Qt)

See also:

http://code.google.com/p/rst2qhc/

Convert a collection of restructured text files into a Qt Help file and (optional) a Qt Help Project file.

sphinx report

See also:

- http://code.google.com/p/sphinx-report/

Introduction

SphinxReport is a report generator that is implemented as an extension to Sphinx.

Its purpose is to facilitate writing scientific reports interpreting large and changing datasets.

It is designed to assist the iterative analysis during the development of a computational scientific pipeline as understanding of the data grows.

Once the pipeline settles down, SphinxReport permits an easy transition towards the automatic report generation needed when the pipeline is run on new data sets.

sphinx-js : autodoc-style extraction into Sphinx for your JS project

See also:

- https://github.com/erikrose/sphinx-js
- https://pypi.python.org/pypi/sphinx-js/

Contents

Why ?

When you write a JavaScript library, how do you explain it to people? If it's a small project in a domain your users are familiar with, JSDoc's alphabetical list of routines might suffice.

But in a larger project, it is useful to intersperse prose with your API docs without having to copy and paste things.

sphinx-js lets you use the industry-leading Sphinx documentation tool with JS projects. It provides a handful of directives, patterned after the Python-centric autodoc ones, for pulling JSDoc-formatted documentation into reStructuredText pages.

And, because you can keep using JSDoc in your code, you remain compatible with the rest of your JS tooling, like Google's Closure Compiler.

Setup

See also:

https://raw.githubusercontent.com/erikrose/sphinx-js/master/README.rst

1. Install JSDoc using npm. `jsdoc` must be on your `$PATH`, so you might want to install it globally:

```
npm install -g jsdoc
```

   We're known to work with jsdoc 3.4.3.

2. Install sphinx-js, which will pull in Sphinx itself as a dependency:

```
pip install sphinx-js
```

3. Make a documentation folder in your project by running `sphinx-quickstart` and answering its questions:

```
cd my-project
sphinx-quickstart

  > Root path for the documentation [.]: docs
  > Separate source and build directories (y/n) [n]:
  > Name prefix for templates and static dir [_]:
  > Project name: My Project
  > Author name(s): Fred Fredson
  > Project version []: 1.0
  > Project release [1.0]:
  > Project language [en]:
  > Source file suffix [.rst]:
  > Name of your master document (without suffix) [index]:
  > Do you want to use the epub builder (y/n) [n]:
  > autodoc: automatically insert docstrings from modules (y/n) [n]:
  > doctest: automatically test code snippets in doctest blocks (y/n) [n]:
  > intersphinx: link between Sphinx documentation of different projects (y/n)␣
↪[n]:
  > todo: write "todo" entries that can be shown or hidden on build (y/n) [n]:
  > coverage: checks for documentation coverage (y/n) [n]:
  > imgmath: include math, rendered as PNG or SVG images (y/n) [n]:
  > mathjax: include math, rendered in the browser by MathJax (y/n) [n]:
  > ifconfig: conditional inclusion of content based on config values (y/n) [n]:
  > viewcode: include links to the source code of documented Python objects (y/n)␣
↪[n]:
  > githubpages: create .nojekyll file to publish the document on GitHub pages (y/
↪n) [n]:
  > Create Makefile? (y/n) [y]:
  > Create Windows command file? (y/n) [y]:
```

4. In the generated Sphinx conf.py file, turn on `sphinx_js` by adding it to `extensions`:

```
extensions = ['sphinx_js']
```

5. If your JS source code is anywhere but at the root of your project, add `js_source_path = '../somewhere/else'` on a line by itself in conf.py. The root of your JS source tree should be where that

setting points, relative to the conf.py file. (The default, `../`, works well when there is a `docs` folder at the root of your project and your source code lives directly inside the root.)

6. If you have special jsdoc configuration, add `jsdoc_config_path = '../conf.json'` (for example) to conf.py as well.
7. If you're documenting only JS and no other languages, you can set your "primary domain" to JS in conf.py:

```
primary_domain = 'js'
```

Then you can omit all the "js:" prefixes in the directives below.

sphinx UML extensions

Sphinx plantum extension

See also:

- https://pypi.python.org/pypi/sphinxcontrib-plantuml
- plant_uml

Usage

First, you may need to specify plantuml command in your conf.py:

plantuml = ['java', '-jar', '/path/to/plantuml.jar']

Instead, you can install a wrapper script in your PATH:

% cat <<EOT > /usr/local/bin/plantuml #!/bin/sh -e java -jar /path/to/plantuml.jar "$@" EOT % chmod +x /usr/local/bin/plantuml

Then, write PlantUML text under .. uml:: directive:

```
.. uml::

   Alice -> Bob: Hi!
   Alice <- Bob: How are you?
```

Sphinx pyreverse extension

See also:

- https://github.com/alendit/sphinx-pyreverse

Contents

Presentation

A simple sphinx extension to generate a UML diagram from python module.

Install

Install with:

```
pip install -e git+https://github.com/alendit/sphinx-pyreverse.git
```

Usage

Add "sphinx-pyreverse" to your conf.py (make sure it is in the PYTHONPATH).

Call the directive with path to python module as content:

```
.. uml::
        {{path to the module}}
```

Requires pyreverse from pylint

sphinxcontrib-dashbuilder

See also:

- https://pypi.python.org/pypi/sphinxcontrib-dashbuilder
- https://bitbucket.org/shimizukawa/sphinxcontrib-dashbuilder

Contents

Description

Sphinx builder extension to generate a 'Documentation Set' for *dash API browser*.

Zeal

See also:

- *Zeal documentation browser*

Sphinx excel table

See also:

- https://crate.io/packages/sphinxcontrib-exceltable/#files
- https://crate.io/packages/sphinxcontrib-exceltable

Contents

Description

Module sphinxcontrib.exceltable is an extension for Sphinx, that adds support for including spreadsheets, or part of them, into Sphinx document.

- Documentation: http://packages.python.org/sphinxcontrib-exceltable
- Bugs: http://bitbucket.org/birkenfeld/sphinx-contrib/

Sphinx howto

Sphinx howto

Contents

How to exclude some files or directories ?

Use exclude_patterns in your :file:‘conf.py‘file.

If you want to exclude the `.venv` directory produced by pipenv when using **PIPENV_VENV_IN_PROJECT=1**

```
# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This pattern also affects html_static_path and html_extra_path .
exclude_patterns = [
        '_build',
        'Thumbs.db',
        '.DS_Store',
        # pipenv virtualenv with PIPENV_VENV_IN_PROJECT=1
        '.venv'
]
```

Hébergeurs Sphinx

Hébergeurs Sphinx

Hébergement Sphinx sur gitlab pages

See also:

- https://docs.gitlab.com/ce/user/project/pages/#explore-gitlab-pages

This projet is on gitlab (https://gitlab.com/pvergain/pvbookmarks)

Hébergement Sphinx sur github pages

See also:

- https://help.github.com/categories/20/articles
- http://daler.github.io/sphinxdoc-test/includeme.html

Automatic setup and deployment for sphinx docs.

This project is intended to be used to deploy sphinx project on:

- Github Pages
- Rsync
- heroku

Sphinx on Read the docs

See also:

- http://read-the-docs.readthedocs.org/en/latest/index.html
- http://read-the-docs.readthedocs.org/en/latest/getting_started.html

Contents

Introduction

Read the Docs hosts documentation for the open source community.

It supports Sphinx docs written with reStructuredText, and can pull from your Subversion, Bazaar, Git, and Mercurial repositories.

The code is open source, and available on github:

```
git clone http://github.com/rtfd/readthedocs.org.git
```

Read the docs on twitter

- https://twitter.com/readthedocs

New theme (4th of november 2013)

See also:

- http://ericholscher.com/blog/2013/nov/4/new-theme-read-the-docs/

Using it

There are two ways that you can use this theme on Read the Docs.

The first is to simply leave your html_theme variable set to default. This is now the default Read the Docs theme.

You can also set RTD_NEW_THEME = True in your project's conf.py, and we will use our theme when building on Read the Docs no matter what html_theme is set to.

After you change these settings, simply rebuild your docs and the theme should update. More information about the theme can be found on the theme documentation page

If you want to continue using the old theme, simply set RTD_OLD_THEME = True in your conf.py.

Conclusion

This theme is a great addition to the documentation ecosystem. It is highly functional and beautiful, allowing users to easily navigate and find what they need.

We have a few more tricks up our sleeves, but theme is ready to launch today. Over the next few weeks we'll be adding a bit more functionality to it, which should be even more delightful.

I hope that you enjoy using it. If you have any feedback, please open an issue on GitHub. To follow announcements for Read the Docs, follow us on Twitter.

If you want to support work like this, help fund development on Read the Docs on Gittip.

Projects on Read the docs

Projects on Read the docs

Fedmsg

See also:

- https://fedmsg.readthedocs.org/en/latest/
- https://github.com/fedora-infra/fedmsg
- https://fedmsg.readthedocs.org/en/latest/
- https://github.com/fedora-infra/fedmsg/tree/develop/doc

Fedmsg Sphinx theme (cloud)

See also:

- https://github.com/fedora-infra/fedmsg/tree/develop/doc
- https://github.com/fedora-infra/fedmsg/blob/develop/doc/conf.py
- https://fedmsg.readthedocs.org/en/latest/
- *Sphinx cloud theme*

Flask Funnel

See also:

- https://github.com/rehandalal/flask-funnel

Contents

Summary

A Flask extension for compressing/minifying assets.

A Flask-Script submanager is also provided.

Documentation

Documentation is at http://flask-funnel.readthedocs.org/en/latest/.

Install

To install:

```
$ pip install Flask-Funnel
```

You can also install the development version https://github.com/rehandalal/flask-funnel/tarball/master#egg=Flask-Funnel-dev:

```
$ pip install Flask-Funnel==dev
```

or:

```
$ git clone git://github.com/rehandalal/flask-funnel.git
$ mkvirtualenv flaskfunnel
$ python setup.py develop
$ pip install -r requirements.txt
```

Test

To run tests from a tarball or git clone:

```
$ python setup.py test
```

Flask funnel Sphinx theme (flask small)

See also:

- https://github.com/writethedocs/docs/blob/master/docs/conf.py
- http://docs.writethedocs.org/en/2013/
- *Flask small*

Python Guide

See also:

- https://github.com/kennethreitz/python-guide

Contents

Hitchhiker's Guide to Python

Python Best Practices Guidebook

Work in progress. If you'd like to help, please do. There's a lot of work to be done.

This guide is currently under heavy development. This opinionated guide exists to provide both novice and expert Python developers a best-practice handbook to the installation, configuration, and usage of Python on a daily basis.

Topics include:

- Platform/version specific installations
- Py2app, Py2exe, bbfreeze, pyInstaller
- Pip / virtualenv
- Documentation. Writing it.
- server configurations / tools for various web frameworks
- fabric
- exhaustive module recommendations, grouped by topic/purpose
- Testing. Jenkins + tox guides.
- How to interface w/ hg from git easily
- what libraries to use for what

If you are not fond of reading reStructuredText, there is an almost up-to-date HTML version at docs.python-guide.org.

Python guide Sphinx theme (kr)

See also:

- https://github.com/kennethreitz/python-guide/blob/master/docs/conf.py
- http://docs.python-guide.org/en/latest/
- *Kr Theme*

Requests

See also:

- *Requests*

Write the docs

See also:

- http://docs.writethedocs.org/
- https://twitter.com/writethedocs

Introduction

Write the Docs is a place where the art and science of documentation can be practiced and appreciated.

There are a lot of people out there that write docs, but there isn't a good place to go to find information, ask questions, and generally be a member of a community of documentarians.

We hope to slowly solve this problem by building a place with high quality information about the art of writing documentation.

Along with that, we hope to open communication between all the awesome people out there writing documentation.

Resources

- Online documentation: http://docs.writethedocs.org/
- Conference: http://conf.writethedocs.org/
- IRC: #writethedocs on freenode
- Twitter: http://twitter.com/writethedocs
- Mailing List: https://groups.google.com/forum/?fromgroups=#!forum/write-the-docs
- Issues & feature requests: https://github.com/writethedocs/docs/issues
- Source repository: https://github.com/writethedocs/docs

Building these docs

This required virtualenv. If you don't have it installed, first run `pip install virtualenv`.

To build this repo locally, run:

```
make develop
make documentation
```

Write the docs Sphinx theme (kr)

See also:

- https://github.com/writethedocs/docs/blob/master/docs/conf.py
- http://docs.writethedocs.org/en/2013/
- *Kr Theme*

Write the docs source code

See also:

- https://github.com/writethedocs/docs

Sphinx examples

Projects using Sphinx

See also:

- http://sphinx-doc.org/latest/examples.html

Contents

Very nice doc

The Atomic Simulation Environment (ASE)

See also:

- https://wiki.fysik.dtu.dk/ase/

Contents

Introduction

The Atomic Simulation Environment (ASE) is the common part of the simulation tools developed at CAMd. ASE provides Python modules for manipulating atoms, analyzing simulations, visualization etc.

Source documentation

See also:

- https://trac.fysik.dtu.dk/projects/ase/browser/trunk/doc
- https://trac.fysik.dtu.dk/projects/ase/browser/trunk/doc/conf.py

Buildout

See also:

- http://www.buildout.org/index.html

Buildout is a Python-based build system for creating, assembling and deploying applications from multiple parts, some of which may be non-Python-based. It lets you create a buildout configuration and reproduce the same software later.

Sphinx neuronvisio documentation

See also:

http://mattions.github.com/neuronvisio/

Source documentation

See also:

- https://github.com/mattions/neuronvisio/tree/master/docs
- https://github.com/mattions/neuronvisio/blob/master/docs/conf.py

Passlib (very nice cloud_sptheme)

See also:

See also:

- http://packages.python.org/passlib/index.html
- http://packages.python.org/passlib/contents.html
- passlib_library
- *Sphinx cloud theme*

Contents

Source documentation

See also:

- http://code.google.com/p/passlib/source/browse/#hg%2Fdocs
- http://code.google.com/p/passlib/source/browse/docs/conf.py

Sfepy

See also:

- http://sfepy.org/doc-devel/index.html

Contents

Source documentation

See also:

- https://github.com/sfepy/sfepy/tree/master/doc
- http://sfepy.org/doc-devel/release_tasks.html#useful-git-commands

C++ doc (with doxylink, breathe, . . .)

Sphinx C++ examples

Contents

Doxylink examples

See also:

Doxylink Examples

Nice doc

Askbot

See also:

http://askbot.org/doc/index.html

Bottle.py

See also:

- http://bottlepy.org/docs/0.11

Bottle is a fast, simple and lightweight WSGI micro web-framework for Python.

Python USB API for Canon digital cameras

See also:

- http://packages.python.org/canon-remote/index.html
- https://bitbucket.org/xxcn/canon-remote/src/8cd492925d71/doc/conf.py

Ceph

See also:

- http://ceph.com/docs/
- https://github.com/ceph/ceph/tree/master/doc

Contents

Announce

```
Georg Brandl georg.brandl@gmail.com
répondre à:  sphinx-users@googlegroups.com
à:   sphinx-users@googlegroups.com
date:        5 mars 2014 07:50
objet:       Re: [sphinx-users] Another project using Sphinx: Ceph
```

Am 03.03.2014 09:03, schrieb Lenz Grimmer: > Hi, > > I just recently started converting the internal documentation of a project I am > working on from OpenOffice documents to Sphinx and must say I really love it! > > While browsing the Sphinx documentation, I read on > http://sphinx-doc.org/examples.html that one should report projects that use > Sphinx for their documentation. > > I've started looking into Ceph some time ago and noticed that their > documentation is based on Sphinx as well: http://ceph.com/docs/ (Sources at > https://github.com/ceph/ceph/tree/master/doc). > > Might be worth adding to the (already quite impressive) list?

Sure, I've done so now.

thanks, Georg

Django

See also:

- https://docs.djangoproject.com/en/dev/
- https://github.com/django/django/tree/master/docs
- django

Contents

How the Django documentation works

See also:

- https://github.com/django/django/tree/master/docs

The documentation in this tree is in plain text files and can be viewed using any text file viewer.

It uses ReST (reStructuredText) [1], and the Sphinx documentation system [2]. This allows it to be built into other forms for easier viewing and browsing.

To create an HTML version of the docs:

- Install Sphinx (using `sudo pip install Sphinx` or some other method)
- In this docs/ directory, type `make html` (or `make.bat html` on Windows) at a shell prompt.

The documentation in _build/html/index.html can then be viewed in a web browser.

[1] http://docutils.sourceforge.net/rst.html [2] http://sphinx-doc.org/

Exquires

See also:

- http://exquires.ca/index.html

Introduction

The EXQUIRES test suite (hereby referred to as EXQUIRES) is an open source framework for assessing the accuracy of image upsampling methods.

EXQUIRES can also be used to compare image difference metrics, or to measure the impact of various factors, including test image selection and properties, downsampler choice, resizing ratio, etc.

Dpm

See also:

http://readthedocs.org/docs/dpm/en/latest/index.html

dpm (data package manager) is a command line tool and python library and for working with Data Packages and interacting with data hubs like those powered by CKAN such http://thedatahub.org/.

dpm is a *simple* way to 'package' data building on *existing* packaging tools developed for code. By putting data in a package, it gets labelled with standardized metadata and can be put in a dpm repository, such as http://thedatahub.org/ or a local one. Once in such a repository, the packages are easy to find and retrieve.

Eyesopen

See also:

http://www.eyesopen.com/documentation

Sphinx gammu documentation

Gammu sphinx documentation

See also:

http://wammu.eu/docs/manual/index.html

```
<gsavix@gmail.com>
heure de l'expéditeur   Envoyé à 13:15 (GMT-02:00). Heure locale : 10:35.
répondre à  sphinx-dev@googlegroups.com
à   sphinx-dev@googlegroups.com
date    18 février 2011 13:15
objet   Re: [sphinx-dev] GSoC and Breathe
liste de diffusion  <sphinx-dev.googlegroups.com> Filtrer les messages de cette liste␣
↪de diffusion
```

i think this is very usefull, i use debian lenny and spend 1 week to put gammu documentation that use sphinx, breathe and doxygen to work properly (with help of http://wammu.eu michal cihar) and now i could use this in "telecentros" public libraries for social public projects (with open source) here in são paulo.

thanks for all effort in sphinx, breathe and doxygen!

Gammu is a project providing abstraction layer for cell phones access.

It covers wide range of phones, mostly focusing on AT compatible phones and Nokia phones.

This manual describes all parts of Gammu, starting with information about Gammu project, going through API documentation for both python-gammu API and libGammu and covering SMS Daemon as well.

Other document

- http://gitorious.org/gammu/gsm-docs

github2 using sphinx

See also:

- http://packages.python.org/github2/
- seealso:: github2

This is a Python library implementing all of the features available in version 2 of the Github API.

Example

```
    class Github(object):

        def __init__(self, username=None, api_token=None, requests_per_second=None,
                     access_token=None, cache=None, proxy_host=None,
                     proxy_port=8080):
            """
            An interface to GitHub's API:
            http://develop.github.com/

            .. versionadded:: 0.2.0
            The ``requests_per_second`` parameter
            .. versionadded:: 0.3.0
            The ``cache`` and ``access_token`` parameters
            .. versionadded:: 0.4.0
            The ``proxy_host`` and ``proxy_port`` parameters

            :param str username: your own GitHub username.
            :param str api_token: can be found at https://github.com/account
            (while logged in as that user):
            :param str access_token: can be used when no ``username`` and/or
            ``api_token`` is used. The ``access_token`` is the OAuth access
            token that is received after successful OAuth authentication.
            :param float requests_per_second: indicate the API rate limit you're
            operating under (1 per second per GitHub at the moment),
            or None to disable delays. The default is to disable delays (for
            backwards compatibility).
            :param str cache: a directory for caching GitHub responses.
            :param str proxy_host: the hostname for the HTTP proxy, if needed.
            :param str proxy_port: the hostname for the HTTP proxy, if needed (will
            default to 8080 if a proxy_host is set and no port is set).
            """

        self.request = GithubRequest(username=username, api_token=api_token,
                                     requests_per_second=requests_per_second,
                                     access_token=access_token, cache=cache,
                                     proxy_host=proxy_host,
                                     proxy_port=proxy_port)
        self.issues = Issues(self.request)
        self.users = Users(self.request)
        self.repos = Repositories(self.request)
        self.commits = Commits(self.request)
        self.organizations = Organizations(self.request)
        self.teams = Teams(self.request)
        self.pull_requests = PullRequests(self.request)
```

(continues on next page)

(continued from previous page)

```
def get_network_meta(self, project):
    """Get Github metadata associated with a project

    :param str project: GitHub project
    """
     return self.request.raw_request("/".join([self.request.github_url,
                                               project,
                                               "network_meta"]), {})

def get_network_data(self, project, nethash, start=None, end=None):
    """Get chunk of Github network data

    :param str project: GitHub project
    :param str nethash: identifier provided by ``get_network_meta``
    :param int start: optional start point for data
    :param int stop: optional end point for data
    """
    data = {"nethash": nethash}
    if start:
        data["start"] = start
    if end:
        data["end"] = end

    return self.request.raw_request("/".join([self.request.github_url,
                                              project,
                                              "network_data_chunk"]),
                                              data)

def _handle_naive_datetimes(f):
    """Decorator to make datetime arguments use GitHub timezone

    :param func f: Function to wrap
    """
    def wrapper(datetime_):
        if not datetime_.tzinfo:
            datetime_ = datetime_.replace(tzinfo=GITHUB_TZ)
        else:
            datetime_ = datetime_.astimezone(GITHUB_TZ)
        return f(datetime_)
    wrapped = wrapper
    wrapped.__name__ = f.__name__
    wrapped.__doc__ = (
        f.__doc__
        + """\n    .. note:: Supports naive and timezone-aware datetimes"""
    )
    return wrapped


@_handle_naive_datetimes
def datetime_to_ghdate(datetime_):
    """Convert Python datetime to Github date string

    :param datetime datetime_: datetime object to convert
    """
    return datetime_.strftime(GITHUB_DATE_FORMAT)
```

Macaron: Python O/R Mapper

See also:

- http://nobrin.github.com/macaron/

Macaron is a small and simple object-relational mapper (ORM) for SQLite. It is distributed as a single file module which has no dependencies other than the Python Standard Library.

Macaron provides provides easy access methods to SQLite database. And it supports Bottle web framework through plugin mechanism.

Example:

```
>>> import macaron
>>> macaron.macaronage(dbfile="members.db")
>>> team = Team.create(name="Houkago Tea Time")
>>> team.members.append(name="Azusa", part="Gt2")
<Member object 1>
>>> macaron.bake()
>>> azu = Member.get("part=?", ["Gt2"])
>>> print azu
<Member 'Azusa : Gt2'>
>>> macaron.db_close()
```

Mediagobelin

See also:

- http://docs.mediagoblin.org/index.html

GNU MediaGoblin is a platform for sharing photos, video and other media in an environment that respects our freedom and independence.

This is a Free Software project. It is built by contributors for all to use and enjoy.

If you're intrested in contributing, see the wiki which has pages that talk about the ways someone can contribute.

Parcel (html_theme = 'flask')

See also:

See also:

- http://parcel.readthedocs.org/en/latest/index.html

Contents

Source documentation

See also:

- https://bitbucket.org/andrewmacgregor/parcel/src/cac478ebf5eb/docs?at=default
- https://bitbucket.org/andrewmacgregor/parcel/src/cac478ebf5eb/docs/conf.py?at=default

Pylons

See also:

- http://docs.pylonsproject.org/en/latest/index.html

PysSCes

See also:

- http://packages.python.org/PySCeS/
- http://pysces.sourceforge.net

PySCeS: the Python Simulator for Cellular Systems is an extendable toolkit for the analysis and investigation of cellular systems. It is available for download at http://pysces.sourceforge.net

Python GTK+ 3 Tutorial

See also:

http://python-gtk-3-tutorial.readthedocs.org/en/latest/index.html

Python

See also:

- http://docs.python.org/

Contents

python py3k

See also:

http://docs.python.org/py3k/

Source documentation

See also:

- https://bitbucket.org/python_mirrors/cpython/src/979567d33376/Doc/conf.py
- http://hg.python.org/cpython/file/a9d4cf7d15b9/Doc/conf.py

python dev

See also:

http://docs.python.org/dev/

Sphinx Prody documentation

Fig. 5: *Prody analysis and modeling of protein structural dynamics*

See also:

http://www.csb.pitt.edu/ProDy/index.html

Contents

Source documentation

See also:

- https://github.com/abakan/ProDy/tree/master/doc
- https://github.com/abakan/ProDy/blob/master/doc/conf.py

Renpy (2013)

See also:

- renpy

Requests

See also:

- https://github.com/kennethreitz/requests
- http://docs.python-requests.org/en/latest/
- https://readthedocs.org/projects/requests/downloads/
- http://requests.readthedocs.org/en/latest/

Simpy Documentation

See also:

- http://packages.python.org/sympy/

Sphinx

Contents

Description

The sphinx doc is written with sphinx of course !

See also:

http://sphinx-doc.org/latest/index.html

Contents

Source documentation

See also:

- https://bitbucket.org/birkenfeld/sphinx/
- https://bitbucket.org/birkenfeld/sphinx/src/6e960412308f/doc
- https://bitbucket.org/birkenfeld/sphinx/src/6e960412308f/doc/conf.py

SQLAlchemy 0.7 Documentation

See also:

- http://docs.sqlalchemy.org/en/latest/intro.html#documentation-overview

Urwid Documentation

See also:

- http://excess.org/urwid/docs/

Wikimedia sphinx projects

Contents

E3_analysis

See also:

- e3_analysis

Classic doc

Tuleap

See also:

- https://tuleap.net/doc/en/index.html
- http://www.tuleap.org
- tuleap

Contents

Announce

```
Am 11.02.2014 09:36, schrieb manon.midy@enalean.com:
> Hi,
>
> I'm working at Enalean, the software provider of Tuleap Open ALM, the first 100%
> Open Source Enterprise ALM (GPL licence). Browing the page mentionning the
> projects using Sphinx http://sphinx-doc.org/examples.html, I realized Tuleap
> Open ALM is not included while the documentation of the project is managed
> thanks to Sphinx.Could you please add Tuleap Open ALM in the list of examples
> and the link to its doc: https://tuleap.net/doc/en/

Manon MIDY
manon.midy@enalean.com
Enalean
www.tuleap.org
```

Sure, I've just added it.

thanks, Georg

Non python projects

See also:

- http://ericholscher.com/blog/2014/feb/11/sphinx-isnt-just-for-python/

I have heard a few times over the past couple months that Sphinx is "mainly for Python projects". This line of thinking makes sense, because Sphinx was created to document Python itself. Sphinx however, is a generic documentation tool that is capable of documenting any software project.

The goal of Sphinx is to help you write prose documentation. Prose docs work great for any kind of software you are documenting.

What it doesn't handle particularly well is generation of docs from source code.

This is a task that is best left to a language-specific tooling, so I don't see this as a major downside of Sphinx.

linux kernel

See also:

- https://www.kernel.org/doc/html/latest/doc-guide/sphinx.html
- https://lwn.net/Articles/692704/

Contents

Introduction

The Linux kernel uses Sphinx to generate pretty documentation from reStructuredText files under `Documentation`. To build the documentation in HTML or PDF formats, use `make htmldocs` or `make pdfdocs`. The generated documentation is placed in `Documentation/output`.

The reStructuredText files may contain directives to include structured documentation comments, or kernel-doc comments, from source files. Usually these are used to describe the functions and types and design of the code. The kernel-doc comments have some special structure and formatting, but beyond that they are also treated as reStructuredText.

Finally, there are thousands of plain text documentation files scattered around `Documentation`. Some of these will likely be converted to reStructuredText over time, but the bulk of them will remain in plain text.

Sphinx-doc extensions for sophisticated C developer

See also:

https://github.com/return42/linuxdoc

Sphinx references

Contents

sphinx on plume

See also:

- https://www.projet-plume.org/fiche/sphinx

Sphinx i18n

Sphinx i18n

See also:

- https://bitbucket.org/shimizukawa/sphinx-intl

Sphinx installation

Installation d'un projet sphinx

See also:

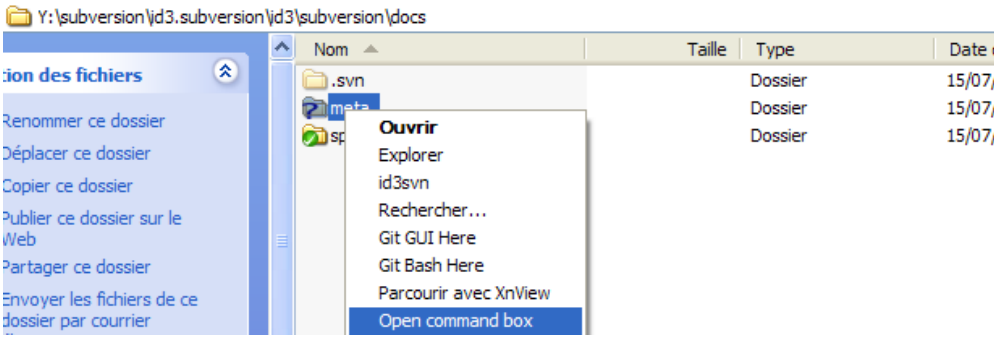
- http://sphinx-doc.org/latest/index.html
- http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/source/sphinx/sphinx.html

Installation d'une documentation Sphinx 'reStructuredText' sous Windows

See also:

- *Open a windows console with the stex extension*
- http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/source/sphinx/sphinx.html

1. **ouvrir une fenêtre de commande**



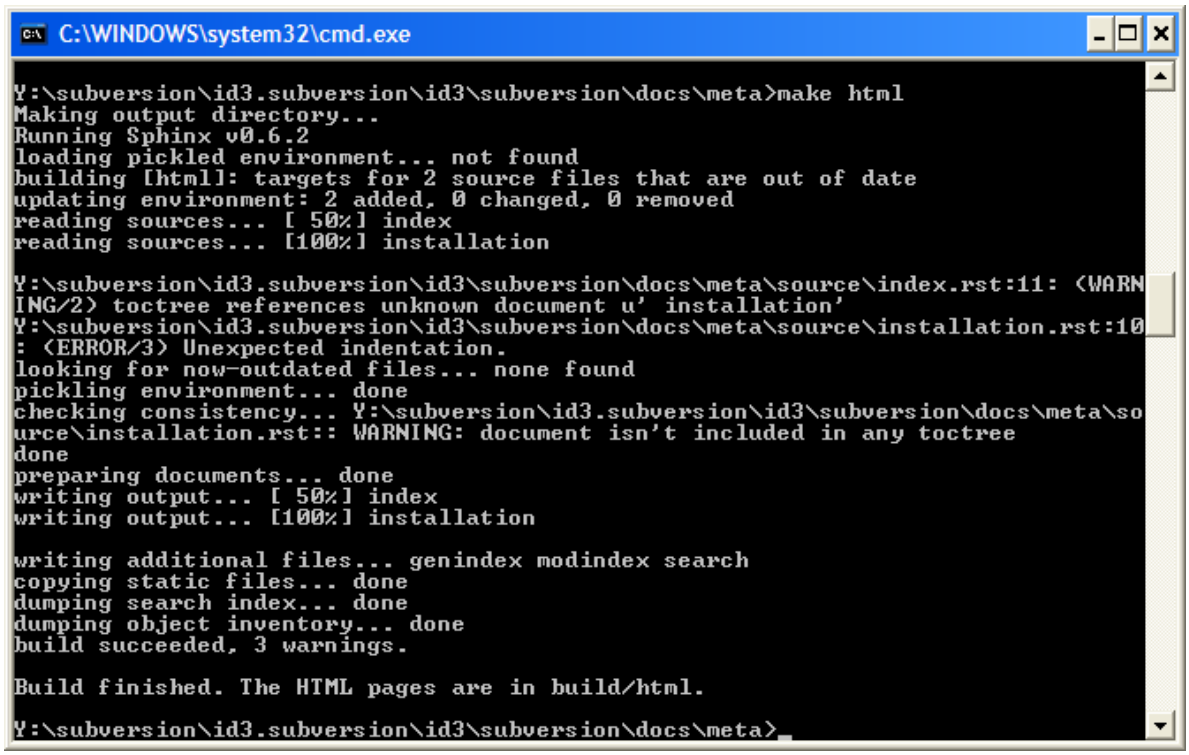
2. taper les commandes suivantes

```
> cd meta
> sphinx-quickstart
```

Production de la documentation html

1. **taper la commande suivante**

```
> make html
```



Sphinx people

Sphinx people

Eric Holscher

See also:

- http://ericholscher.com/
- https://twitter.com/ericholscher
- https://readthedocs.org/
- https://github.com/rtfd/readthedocs.org

Contents

Interesting projects on Read the Docs: Teaching

As the maintainer of Read the Docs, I spend a lot of time looking through random projects, and getting inspired. People have been doing lots of interesting things with the project, and I'd like to highlight some of them.

This edition is focused on teaching. All of these projects are trying to teach something, and doing it in different ways. Some are community contributed guides that have many authors, where some are a single person trying to distill their experience into something valuable for others.

The projects mentioned here will be featured on the homepage of Read the Docs until I do another posting, where those new projects will take their place.

Georg Brandl

See also:

- http://www.pocoo.org/team/
- https://twitter.com/birkenfeld
- georg_brandl_python

Contents

Sphinx

See also:

- http://sphinx-doc.org/

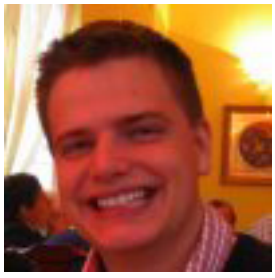
Welcome

What users say:

```
"Cheers for a great tool that actually makes programmers want to write documentation!"
```

Sphinx is a tool that makes it easy to create intelligent and beautiful documentation, written by Georg Brandl and licensed under the BSD license.

Presentation



Georg Brandl is a Python core developer since 2005, and cares for its documentation at docs.python.org.

He is blogging on pythonic.pocoo.org.

His IRC nickname is birkenfeld, and you can contact him via email at georg@python.org.

Follow Georg on twitter: @birkenfeld

Georg Brandl and Brett Cannon to Receive PSF Community Awards (Thursday, August 07, 2008)

See also:

- http://pyfound.blogspot.fr/2008/08/georg-brandl-and-brett-cannon-to.html

At the July Board meeting of the PSF Board of Directors, PSF Community Awards were awarded to Georg Brandl and Brett Cannon.

Georg has been an enthusiastic contributor to the core for several years, and a while ago stunned the Python development world by building the *Sphinx* documentation system as an alternative to the LaTeX-based system we had been using previously, and converting the Python documentation to use it.

Brett has also been an active core developer for many years, but was nominated for his infrastructure work in migrating the Python bug-tracking system off of SourceForge to our own Roundup instance, and for his efforts keeping the Python developer introduction updated.

Georg and Brett richly deserve recognition for their contributions.

Congratulations to Brett and Georg, and thanks for all your hard work!

Sphinx tutorials

Sphinx tutorials

Contents

Rest Sphinx

See also:

- http://sphinx-doc.org/latest/index.html

- http://rest-sphinx-memo.readthedocs.org/en/latest/ReST.html
- http://rest-sphinx-memo.readthedocs.org/en/latest/References.html

Python

See also:

- http://docs.python.org/dev/documenting/

matplotlib

See also:

http://matplotlib.sourceforge.net/sampledoc/

This is a tutorial introduction to quickly get you up and running with your own sphinx documentation system. We'll cover installing sphinx, customizing the look and feel, using custom extensions for embedding plots, inheritance diagrams, syntax highlighted ipython sessions and more. If you follow along the tutorial, you'll start with nothing and end up with this site – it's the bootstrapping documentation tutorial that writes itself!

Openalea

See also:

http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/source/sphinx/sphinx.html

Plone

See also:

- https://docs.plone.org/about/index.html
- https://github.com/collective/collective.developermanual

Github Fork and Edit button

See also:

- *Github Fork and Edit button*
- https://github.com/blog/844-forking-with-the-edit-button
- https://confluence.atlassian.com/display/BITBUCKET/Fork+a+Repo,+Compare+Code,+and+Create+a+Pull+Request

You can commit file edits through GitHub web interface using Fork and Edit button Alternative, clone the repository using git, perform changes and push them back

Plone collective GitHub repository has open-for-all contribution access. If you want to contribute changes without asking the maintainers to merge them, please add your GitHub username to your profile on plone.org and request access here.

Geoserver

See also:

http://docs.geoserver.org/trunk/en/docguide/sphinx.html

Documentation style guide sphinx

See also:

https://github.com/benoitbryon/documentation-style-guide-sphinx

Documentation style guide mercurial

See also:

http://mercurial.selenic.com/wiki/HelpStyleGuide#Sections

Tools for Sphinx

Tools for building sphinx

Qt program

See also:

- https://gist.github.com/1672347
- http://pymolurus.blogspot.com/2012/01/documentation-viewer-for-sphinx.html

Eric Holscher, one of the creators of Read The Docs, recently posted about the importance of a documentation culture in Open Source development, and about things that could be done to encourage this.

He makes some good points, and Read The Docs is a very nice looking showcase for documentation.

Writing good documentation is difficult enough at the best of times, and one practical problem that I face when working on Sphinx documentation is that I often feel I have to break away from composing it to building it, to see how it looks - because the look of it on the page will determine how I want to refine it.

What I've tended to do is work iteratively by making some changes to the ReST sources, invoking make html and refreshing the browser to show how the built documentation looks.

This is OK, but does break the flow more than a little (for me, anyway, but I can't believe I'm the only one).

I had the idea that it would be nice to streamline the process somewhat, so that all I would need to do is to save the changed ReST source – the building and browser refresh would be automatically done, and if I had the editor and browser windows open next to each other in tiled fashion, **I could achieve a sort of WYSIWYG effect with the changes appearing in the browser a second or two after I saved any changes.**

I decided to experiment with this idea, and needed a browser which I could easily control (to get it to refresh on-demand). I decided to use Roberto Alsina's 128-line browser, which is based on QtWebKit and PyQt.

Roberto posted his browser code almost a year ago, and I knew I'd find a use for it one day :-)

The code (MIT licensed) is available from here. As it's a single file standalone script, I haven't considered putting it on PyPI – it's probably easier to download it to a $HOME/bin or similar location, then you can invoke it in the docs directory of your project, run your editor, position the browser and editor windows suitably, and you're ready to go!

```
#!/usr/bin/env python
#
# Copyright (C) 2012 Vinay Sajip. Licensed under the MIT license.
#
# Based on Roberto Alsina's 128-line web browser, see
#
# http://lateral.netmanagers.com.ar/weblog/posts/BB948.html
#
import json
import os
import subprocess
import sys
import tempfile
from urllib import pathname2url

import sip
sip.setapi("QString", 2)
sip.setapi("QVariant", 2)

from PyQt4 import QtGui,QtCore,QtWebKit, QtNetwork

settings = QtCore.QSettings("Vinay Sajip", "DocWatch")

class Watcher(QtCore.QThread):
        """
        A watcher which looks for source file changes, builds the documentation,
        and notifies the browser to refresh its contents
        """
        def run(self):
                self._stop = False
                watch_command = 'inotifywait -rq -e close_write --exclude \'"*.html"\
↪' .'.split()
                make_command = 'make html'.split()
                while not self._stop:
                        # Perhaps should put notifier access in a mutex - not␣
↪bothering yet
                        self.notifier = subprocess.Popen(watch_command)
                        self.notifier.wait()
                        if self._stop:
                                break
                        subprocess.call(make_command)
                        # Refresh the UI ...
                        self.parent().changed.emit()

        def stop(self):
                self._stop = True
                # Perhaps should put notifier access in a mutex - not bothering for␣
↪now
                if self.notifier.poll() is None:    # not yet terminated ...
                        self.notifier.terminate()

class MainWindow(QtGui.QMainWindow):
        """
        A browser intended for viewing HTML documentation generated by Sphinx.
        """
        changed = QtCore.pyqtSignal()
```

(continues on next page)

(continued from previous page)

```
        def __init__(self, url):
                QtGui.QMainWindow.__init__(self)
                self.sb=self.statusBar()

                self.pbar = QtGui.QProgressBar()
                self.pbar.setMaximumWidth(120)
                self.wb=QtWebKit.QWebView(loadProgress = self.pbar.setValue,␣
↪loadFinished = self.pbar.hide, loadStarted = self.pbar.show, titleChanged = self.
↪setWindowTitle)
                self.setCentralWidget(self.wb)

                self.tb=self.addToolBar("Main Toolbar")
                for a in (QtWebKit.QWebPage.Back, QtWebKit.QWebPage.Forward, QtWebKit.
↪QWebPage.Reload):
                        self.tb.addAction(self.wb.pageAction(a))

                self.url = QtGui.QLineEdit(returnPressed = lambda:self.wb.
↪setUrl(QtCore.QUrl.fromUserInput(self.url.text())))
                self.tb.addWidget(self.url)

                self.wb.urlChanged.connect(lambda u: self.url.setText(u.toString()))
                self.wb.urlChanged.connect(lambda: self.url.setCompleter(QtGui.
↪QCompleter(QtCore.QStringList([QtCore.QString(i.url().toString()) for i in self.wb.
↪history().items()]), caseSensitivity = QtCore.Qt.CaseInsensitive)))

                self.wb.statusBarMessage.connect(self.sb.showMessage)
                self.wb.page().linkHovered.connect(lambda l: self.sb.showMessage(l,␣
↪3000))

                self.search = QtGui.QLineEdit(returnPressed = lambda: self.wb.
↪findText(self.search.text()))
                self.search.hide()
                self.showSearch = QtGui.QShortcut("Ctrl+F", self, activated = lambda:␣
↪(self.search.show() , self.search.setFocus()))
                self.hideSearch = QtGui.QShortcut("Esc", self, activated = lambda:␣
↪(self.search.hide(), self.wb.setFocus()))

                self.quit = QtGui.QShortcut("Ctrl+Q", self, activated = self.close)
                self.zoomIn = QtGui.QShortcut("Ctrl++", self, activated = lambda:␣
↪self.wb.setZoomFactor(self.wb.zoomFactor()+.2))
                self.zoomOut = QtGui.QShortcut("Ctrl+-", self, activated = lambda:␣
↪self.wb.setZoomFactor(self.wb.zoomFactor()-.2))
                self.zoomOne = QtGui.QShortcut("Ctrl+=", self, activated = lambda:␣
↪self.wb.setZoomFactor(1))
                self.wb.settings().setAttribute(QtWebKit.QWebSettings.PluginsEnabled,␣
↪True)

                self.sb.addPermanentWidget(self.search)
                self.sb.addPermanentWidget(self.pbar)

                self.load_settings()

                self.wb.load(url)
                self.watcher = Watcher(self)

                self.changed.connect(self.wb.reload)
```

(continues on next page)

(continued from previous page)

```
                self.watcher.start()

        def load_settings(self):
                settings.beginGroup('mainwindow')
                pos = settings.value('pos')
                size = settings.value('size')
                if isinstance(pos, QtCore.QPoint):
                        self.move(pos)
                if isinstance(size, QtCore.QSize):
                        self.resize(size)
                settings.endGroup()

        def save_settings(self):
                settings.beginGroup('mainwindow')
                settings.setValue('pos', self.pos())
                settings.setValue('size', self.size())
                settings.endGroup()

        def closeEvent(self, event):
                self.save_settings()
                self.watcher.stop()

if __name__ == "__main__":
        if not os.path.isdir('_build'):
                # very simplistic sanity check. Works for me, as I generally use
                # sphinx-quickstart defaults
                print('You must run this application from a Sphinx directory␣
↪containing _build')
                rc = 1
        else:
                app=QtGui.QApplication(sys.argv)
                path = os.path.join('_build', 'html', 'index.html')
                url = 'file:///' + pathname2url(os.path.abspath(path))
                url = QtCore.QUrl(url)
                wb=MainWindow(url)
                wb.show()
                rc = app.exec_()
        sys.exit(rc)
```

Ironpython

Update: Another advantage of using the subprocess / command line approach to notification is that it's easy to slot in a solution for a platform which doesn't support inotify.

Alternatives are available for both Windows and Mac OS X. For example, on Windows, if you have IronPython installed, the following script could be used to provide the equivalent functionality to inotifywait (for this specific application):

```
import clr
import os

from System.IO import FileSystemWatcher, NotifyFilters

stop = False
```

(continues on next page)

(continued from previous page)

```
def on_change(source, e):
        global stop
        if not e.Name.endswith('.html'):
                stop = True
        print('%s: %s, stop = %s' % (e.FullPath, e.ChangeType, stop))

watcher = FileSystemWatcher(os.getcwd())
watcher.NotifyFilter = NotifyFilters.LastWrite | NotifyFilters.FileName
watcher.EnableRaisingEvents = True
watcher.IncludeSubdirectories = True
watcher.Changed += on_change
watcher.Created += on_change

while not stop:
        pass
```

Mac OS X

Whereas for Mac OS X, if you install the MacFSEvents package, the following script could be used to provide the equivalent functionality to inotifywait (again, for this specific application):

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

import os

from fsevents import Observer, Stream

stop = False

def on_change(e):
        global stop
        path = e.name
        if os.path.isfile(path):
                if not path.endswith('.html'):
                        stop = True
        print('%s: %s, stop = %s' % (e.name, e.mask, stop))

observer = Observer()
observer.start()
stream = Stream(on_change, os.getcwd(), file_events=True)
observer.schedule(stream)
try:
        while not stop:
                pass
finally:
        observer.unschedule(stream)
        observer.stop()
        observer.join()
```

Automatically-build-sphinx-documentation

See also:

- http://www.hackzine.org/posts/2011/09/11/automatically-build-sphinx-documentation

```
#!/bin/bash
## Automatically build Sphinx documentation upon file change
## Copyright (c) 2011 Samuele ~redShadow~ Santi - Under GPL

WORKDIR="$( dirname "$0" )"
while :; do
        ## Wait for changes
        inotifywait -e modify,create,delete -r "$WORKDIR"
        ## Make html documentation
        make -C "$WORKDIR" html
done
```

Rst lint

See also:

- http://svn.python.org/projects/python/trunk/Doc/tools/rstlint.py

```
# Check for stylistic and formal issues in .rst and .py
# files included in the documentation.
#
# 01/2009, Georg Brandl

# TODO: - wrong versions in versionadded/changed
#       - wrong markup after versionchanged directive
```

Sphinx themes

Sphinx themes

See also:

- http://sphinx-doc.org/theming.html

Basicstrap Theme

See also:

- http://pythonhosted.org/sphinxjp.themes.basicstrap/
- https://github.com/tell-k/sphinxjp.themes.basicstrap

Contents

Introduction

Basicstrap style theme for Sphinx.

Features

- provide `basicstrap` theme for render HTML document.
- using Twitter Bootstrap.
- support Responsive Design.
- change the layout flexibility.
- Google Web Fonts available.
- Font Awesome available.
- easily change the design. by Bootswatch.

Set up

Make environment with pip:

```
$ pip install sphinxjp.themes.basicstrap
```

Make environment with easy_install:

```
$ easy_install sphinxjp.themes.basicstrap
```

Convert Usage

setup conf.py with:

```
extensions = ['sphinxjp.themecore']
html_theme = 'basicstrap'
```

and run:

```
$ make html
```

Caution: Caution when upgrading from 0.1.1 to 0.2.0

- In version 0.1.1, the header color was black in the default, it has become white in 0.2.0.
- If you like the black color header, please set to True the 'header_inverse' option.

Requirement

- Python 2.7 or later (not support 3.x)
- Sphinx 1.1.x or later.
- sphinxjp.themecore 0.1.3 or later

Using

- Twitter Bootstrap 2.2.2
- jQuery 1.8.3
- Bootswatch
- Font Awesome 3.0

License

- sphinxjp.themes.basicstrap Licensed under the MIT license .
- Twitter Bootstrap is licensed under the Apache license.
- Bootswatch is licensed under the Apache license.
- Font Awesome is licensed unde the license.

See the LICENSE file for specific terms.

Bootstrap Theme

See also:

- http://ryan-roemer.github.com/sphinx-bootstrap-theme/README.html
- https://github.com/ryan-roemer/sphinx-bootstrap-theme

Contents

Introduction

This Sphinx theme integrates the Twitter Bootstrap CSS / JavaScript framework with various layout options, hierarchical menu navigation, and mobile-friendly responsive design.

Sphinx cloud theme

See also:

- http://packages.python.org/cloud_sptheme/

- http://inasafe.readthedocs.org/en/latest/index.html
- https://bitbucket.org/ecollins/cloud_sptheme
- *Passlib (very nice cloud_sptheme)*

Contents

Introduction

This page contains examples of various features of the Cloud theme. It's mainly useful internally, to make sure everything is displaying correctly.

Installation

```
pip install -U cloud_sptheme
```

Inline Text

```
Inline literal: ``literal text``.
```

Inline literal: `literal text`.

```
External links are prefixed with an arrow: `<http://www.google.com>`_.
```

External links are prefixed with an arrow: http://www.google.com.

```
But email links are not prefixed: bob@example.com.
```

But email links are not prefixed: bob@example.com.

```
Issue tracker link: :issue:`5`.
```

```
extensions = [
    # http://pythonhosted.org/cloud_sptheme/index.html#extensions
    # https://bitbucket.org/ecollins/cloud_sptheme

    'cloud_sptheme.ext.issue_tracker',
]

# set path to issue tracker:
issue_tracker_url = "https://bitbucket.org/ecollins/cloud_sptheme/issue/{issue}"
```

Admonition Styles

```
.. note::
    This is a note.
```

Note: This is a note.

```
.. warning::

    This is warning.
```

Warning: This is warning.

```
.. seealso::

    This is a "see also" message.
```

See also:

This is a “see also” message.

```
.. todo::

    This is a todo message.

    With some additional next on another line.
```

Todo: This is a todo message.

With some additional next on another line.

```
.. deprecated:: XXX This is a deprecation warning.
```

Deprecated since version XXX: This is a deprecation warning.

```
.. rst-class:: floater

.. note::
    This is a floating note.
```

Note: This is a floating note.

Table Styles

```
extensions = [
    # http://pythonhosted.org/cloud_sptheme/index.html#extensions
    # https://bitbucket.org/ecollins/cloud_sptheme

    'cloud_sptheme.ext.table_styling',
]
```

```
.. table:: Normal Table
```

Table 2: Normal Table

Header1	Header2	Header3
Row 1	Row 1	Row 1
Row 2	Row 2	Row 2
Row 3	Row 3	Row 3

```
.. rst-class:: plain

.. table:: Plain Table (no row shading)
```

Table 3: Plain Table (no row shading)

Header1	Header2	Header3
Row 1	Row 1	Row 1
Row 2	Row 2	Row 2
Row 3	Row 3	Row 3

```
.. rst-class:: centered

.. table:: Centered Table
```

Table 4: Centered Table

Header1	Header2	Header3
Row 1	Row 1	Row 1
Row 2	Row 2	Row 2
Row 3	Row 3	Row 3

```
.. rst-class:: fullwidth

.. table:: Full Width Table
```

Table 5: Full Width Table

Header1	Header2	Header3
Row 1	Row 1	Row 1
Row 2	Row 2	Row 2
Row 3	Row 3	Row 3

```
.. table:: Table Styling Extension
    :widths: 1 2 3
    :header-columns: 1
    :column-alignment: left center right
    :column-dividers: none single double single
    :column-wrapping: nnn
```

```
.. rst-class:: html-toggle

.. _toggle-test-link:
```

Toggleable Section

This section is collapsed by default. But if a visitor follows a link to this section or something within it (such as *this*), it will automatically be expanded.

```
.. rst-class:: html-toggle expanded
```

Toggleable Subsection

Subsections can also be marked as toggleable. This one should be expanded by default.

```
.. rst-class:: emphasize-children
```

Section With Emphasized Children

Mainly useful for sections with many long subsections, where a second level of visual dividers would be useful.

Child Section

Should be have slightly lighter background, and be indented.

```
.. rst-class:: html-toggle
```

Toggleable Subsection

Test of emphasized + toggleable styles. Should be collapsed by default.

ReadTheDocs

See also:

- https://github.com/fedora-infra/fedmsg/blob/develop/doc

To use this theme on http://readthedocs.org:

1. If it doesn't already exist, add a pip `requirements.txt` file to your documentation (e.g. alongside `conf.py`). It should contain a minimum of the following lines:

   ```
   sphinx
   cloud_sptheme
   ```

   ... as well as any other build requirements for your project's documentation.

2. When setting up your project on ReadTheDocs, enter the path to `requirements.txt` in the *requirements file* field on the project configuration page.

3. ReadTheDocs will now automatically download the latest version of `cloud_sptheme` when building your documentation.

Project using cloud_sptheme

Fedmsg (on read the docs)

See also:

- *Fedmsg Sphinx theme (cloud)*

Passlib conf.py

```
# -*- coding: utf-8 -*-
#
# Passlib documentation build configuration file, created by
# sphinx-quickstart on Mon Mar  2 14:12:06 2009.
#
# This file is execfile()d with the current directory set to its containing dir.
#
# Note that not all possible configuration values are present in this
# autogenerated file.
#
# All configuration values have a default; values that are commented out
# serve to show the default.

import sys, os

options = os.environ.get("PASSLIB_DOCS", "").split(",")

#make sure passlib in sys.path
doc_root = os.path.abspath(os.path.join(__file__,os.path.pardir))
source_root = os.path.abspath(os.path.join(doc_root,os.path.pardir))
sys.path.insert(0, source_root)

# If extensions (or modules to document with autodoc) are in another directory,
```

(continues on next page)

(continued from previous page)

```
# add these directories to sys.path here. If the directory is relative to the
# documentation root, use os.path.abspath to make it absolute, like shown here.
#sys.path.insert(0, os.path.abspath('.'))

#building the docs requires the Cloud sphinx theme & extensions
# https://bitbucket.org/ecollins/cloud_sptheme
#which contains some sphinx extensions used by passlib
import cloud_sptheme as csp

#hack to make autodoc generate documentation from the correct class...
import passlib.utils.md4 as md4_mod
md4_mod.md4 = md4_mod._builtin_md4

# -- General configuration -----------------------------------------------------

# If your documentation needs a minimal Sphinx version, state it here.
needs_sphinx = '1.0'

# Add any Sphinx extension module names here, as strings. They can be extensions
# coming with Sphinx (named 'sphinx.ext.*') or your custom ones.
extensions = [
    # standard sphinx extensions
    'sphinx.ext.autodoc',
    'sphinx.ext.todo',

    #add autdoc support for ReST sections in class/function docstrings
    'cloud_sptheme.ext.autodoc_sections',

    #adds extra ids & classes to genindex html, for additional styling
    'cloud_sptheme.ext.index_styling',

    #inserts toc into right hand nav bar (ala old style python docs)
    'cloud_sptheme.ext.relbar_toc',

    # add "issue" role
    'cloud_sptheme.ext.issue_tracker',
    ]

# Add any paths that contain templates here, relative to this directory.
templates_path = ['_templates']

# The suffix of source filenames.
source_suffix = '.rst'

# The encoding of source files.
source_encoding = 'utf-8'

# The master toctree document.
master_doc = 'contents'
index_doc = 'index'

# General information about the project.
project = u'Passlib'
copyright = u'2008-2012, Assurance Technologies, LLC'

# The version info for the project you're documenting, acts as replacement for
# |version| and |release|, also used in various other places throughout the
```

(continues on next page)

(continued from previous page)

```
# built documents.
#

# version: The short X.Y version.
# release: The full version, including alpha/beta/rc tags.
from passlib import __version__ as release
version = csp.get_version(release)

# The language for content autogenerated by Sphinx. Refer to documentation
# for a list of supported languages.
#language = None

# There are two options for replacing |today|: either, you set today to some
# non-false value, then it is used:
#today = ''
# Else, today_fmt is used as the format for a strftime call.
#today_fmt = '%B %d, %Y'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
exclude_patterns = [
    # disabling documentation of this until module is more mature.
    "lib/passlib.utils.compat.rst",

    # may remove this in future release
    "lib/passlib.utils.md4.rst",
]

# The reST default role (used for this markup: `text`) to use for all documents.
#default_role = None

# If true, '()' will be appended to :func: etc. cross-reference text.
add_function_parentheses = True

# If true, the current module name will be prepended to all description
# unit titles (such as .. function::).
#add_module_names = True

# If true, sectionauthor and moduleauthor directives will be shown in the
# output. They are ignored by default.
#show_authors = False

# The name of the Pygments (syntax highlighting) style to use.
pygments_style = 'sphinx'

# A list of ignored prefixes for module index sorting.
modindex_common_prefix = [ "passlib." ]

# -- Options for all output -------------------------------------------------
todo_include_todos = "hide-todos" not in options
keep_warnings = "hide-warnings" not in options
issue_tracker_url = "gc:passlib"

# -- Options for HTML output ------------------------------------------------

# The theme to use for HTML and HTML Help pages.  See the documentation for
# a list of builtin themes.
```

(continues on next page)

(continued from previous page)

```
138 html_theme = 'redcloud'
139
140 # Theme options are theme-specific and customize the look and feel of a theme
141 # further.  For a list of options available for each theme, see the
142 # documentation.
143 if html_theme in ['cloud', 'redcloud']:
144     html_theme_options = { "roottarget": index_doc, "collapsiblesidebar": True }
145     if 'for-pypi' in options:
146         html_theme_options['googleanalytics_id'] = 'UA-22302196-2'
147         html_theme_options['googleanalytics_path'] = '/passlib/'
148 else:
149     html_theme_options = {}
150 html_theme_options.update(issueicon=r'"\21D7"')
151
152 # Add any paths that contain custom themes here, relative to this directory.
153 html_theme_path = [csp.get_theme_dir()]
154
155 # The name for this set of Sphinx documents.  If None, it defaults to
156 # "<project> v<release> documentation".
157 html_title = project + " v" + release + " Documentation"
158
159 # A shorter title for the navigation bar.  Default is the same as html_title.
160 html_short_title = project + " " + version + " Documentation"
161
162 # The name of an image file (relative to this directory) to place at the top
163 # of the sidebar.
164 html_logo = os.path.join("_static", "masthead.png")
165
166 # The name of an image file (within the static path) to use as favicon of the
167 # docs.  This file should be a Windows icon file (.ico) being 16x16 or 32x32
168 # pixels large.
169 html_favicon = "logo.ico"
170
171 # Add any paths that contain custom static files (such as style sheets) here,
172 # relative to this directory. They are copied after the builtin static files,
173 # so a file named "default.css" will overwrite the builtin "default.css".
174 html_static_path = ['_static']
175
176 # If not '', a 'Last updated on:' timestamp is inserted at every page bottom,
177 # using the given strftime format.
178 html_last_updated_fmt = '%b %d, %Y'
179
180 # If true, SmartyPants will be used to convert quotes and dashes to
181 # typographically correct entities.
182 html_use_smartypants = True
183
184 # Custom sidebar templates, maps document names to template names.
185 #html_sidebars = {}
186
187 # Additional templates that should be rendered to pages, maps page names to
188 # template names.
189 #html_additional_pages = {}
190
191 # If false, no module index is generated.
192 #html_domain_indices = True
193
194 # If false, no index is generated.
```

(continues on next page)

(continued from previous page)

```
#html_use_index = True

# If true, the index is split into individual pages for each letter.
#html_split_index = False

# If true, links to the reST sources are added to the pages.
#html_show_sourcelink = True

# If true, "Created using Sphinx" is shown in the HTML footer. Default is True.
#html_show_sphinx = True

# If true, "(C) Copyright ..." is shown in the HTML footer. Default is True.
#html_show_copyright = True

# If true, an OpenSearch description file will be output, and all pages will
# contain a <link> tag referring to it.  The value of this option must be the
# base URL from which the finished HTML is served.
#html_use_opensearch = ''

# This is the file name suffix for HTML files (e.g. ".xhtml").
#html_file_suffix = None

# Output file base name for HTML help builder.
htmlhelp_basename = project + 'Doc'


# -- Options for LaTeX output --------------------------------------------------

# The paper size ('letter' or 'a4').
#latex_paper_size = 'letter'

# The font size ('10pt', '11pt' or '12pt').
#latex_font_size = '10pt'

# Grouping the document tree into LaTeX files. List of tuples
# (source start file, target name, title, author, documentclass [howto/manual]).
latex_documents = [
  (index_doc, project + '.tex', project + u' Documentation',
   u'Assurance Technologies, LLC', 'manual'),
]

# The name of an image file (relative to this directory) to place at the top of
# the title page.
#latex_logo = None

# For "manual" documents, if this is true, then toplevel headings are parts,
# not chapters.
#latex_use_parts = False

# If true, show page references after internal links.
#latex_show_pagerefs = False

# If true, show URL addresses after external links.
#latex_show_urls = False

# Additional stuff for the LaTeX preamble.
#latex_preamble = ''
```

(continues on next page)

(continued from previous page)

```
252
253 # Documents to append as an appendix to all manuals.
254 #latex_appendices = []
255
256 # If false, no module index is generated.
257 #latex_domain_indices = True
258
259
260 # -- Options for manual page output --------------------------------------------
261
262 # One entry per manual page. List of tuples
263 # (source start file, name, description, authors, manual section).
264 man_pages = [
265     (index_doc, project, project + u' Documentation',
266      [u'Assurance Technologies, LLC'], 1)
267 ]
```

Flask Theme

See also:

- http://packages.python.org/Flask-Themes/
- http://flask.pocoo.org/extensions/
- https://bitbucket.org/andrewmacgregor/parcel/src/cac478ebf5eb/docs?at=default

Contents

Introduction

Flask small

See also:

- https://github.com/mitsuhiko/flask-sphinx-themes/tree/master/flask_small

Contents

Projects using this theme

See also:

- *Flask funnel Sphinx theme (flask small)*

Haiku Theme

See also:

- stem_sphinx_haiku
- http://sphinx-doc.org/theming.html

Projects using this theme

See also:

- stem

Kr Theme

See also:

- https://github.com/kennethreitz/kr-sphinx-themes

Projects using this theme

See also:

- *Write the docs Sphinx theme (kr)*
- *Python guide Sphinx theme (kr)*

Python doc Theme

See also:

- http://hg.python.org/cpython/file/22e88355acd6/Doc/tools/sphinxext

Where can I find a sphinx theme used in Python official documentation site ?

```
Sujet:  Re: [sphinx-users] Where can I find a sphinx theme used in Python official␣
↪documentation site?
Date :  Sat, 1 Feb 2014 04:52:34 -0800 (PST)
De :    Hiroki Watanabe <hwatanabe.japan@gmail.com>
Répondre à :    sphinx-users@googlegroups.com
Pour :  sphinx-users@googlegroups.com
```

Hello,

Thank you very much !

I tried it. It worked well except color of sidebar's collapse button.

The followings are things I tried.

Searching good themes, I noticed an extension theme, # 'sphinxjp.themes.basicstrap' was also excellent. # For while, I will use it for my purpose.

Things I tried

1. Go the following site: http://hg.python.org/cpython/file/22e88355acd6/Doc/tools/sphinxext/pydoctheme
2. Download a pydoctheme as a zip archive by clicking a "zip" link where you can see left side of the site.
3. Unzip the archive under sphinx's source folder like this: source/_themes/pydoctheme/theme.conf
4. Edit conf.py like this:

```
sys.path.append(os.path.abspath('_themes'))
html_theme_path = ['_themes']
html_theme = 'pydoctheme'
```

5. Execute 'make html' You will notice that you can not see a collapse button on the sidebar. So,
6. Add 'collapsiblesidebar' option and its color to conf.py:

```
html_theme_options = {
   'collapsiblesidebar': True,
   'sidebarbtncolor': '#eeeeee',
}
```

A problem still exists. When you hover the collapse button, it will disappear due to it having the same color of background.

I don't know how to solve this problem, maybe some CSS knowledge is required.

Best regards,

201421 135421 UTC+9 Takayuki SHIMIZUKAWA:

```
Hi,

I think this is the one:
http://hg.python.org/cpython/file/22e88355acd6/Doc/tools/sphinxext
<http://hg.python.org/cpython/file/22e88355acd6/Doc/tools/sphinxext>
However, I have not used it.

Regards,
--
Takayuki SHIMIZUKAWA
http://about.me/shimizukawa
```

Shark Theme

See also:

- http://image.diku.dk/shark/sphinx_pages/build/html/rest_sources/tutorials/for_developers/managing_the_documentation.html#sphinx-and-doxygen-html-header-injection

Sphinx templating

Sphinx templating

See also:

- http://sphinx-doc.org/templating.html
- http://sphinx-doc.org/latest/theming.html?highlight=template

Contents

Add javascript files

See also:

- http://sphinx-doc.org/ext/appapi.html#sphinx.application.Sphinx.add_javascript

Creating themes

See also:

http://sphinx-doc.org/latest/theming.html?highlight=template

As said, themes are either a directory or a zipfile (whose name is the theme name), containing the following:

- A `theme.conf` file, see below.
- HTML templates, if needed.
- A `static/` directory containing any static files that will be copied to the output static directory on build. These can be images, styles, script files.

The `theme.conf` file is in INI format[1] (readable by the standard Python `ConfigParser` module) and has the following structure:

```
[theme]
inherit = base theme
stylesheet = main CSS name
pygments_style = stylename

[options]
variable = default value
```

[1] It is not an executable Python file, as opposed to `conf.py`, because that would pose an unnecessary security risk if themes are shared.

- The **inherit** setting gives the name of a "base theme", or `none`. The base theme will be used to locate missing templates (most themes will not have to supply most templates if they use `basic` as the base theme), its options will be inherited, and all of its static files will be used as well.
- The **stylesheet** setting gives the name of a CSS file which will be referenced in the HTML header. If you need more than one CSS file, either include one from the other via CSS' `@import`, or use a custom HTML template that adds `<link rel="stylesheet">` tags as necessary. Setting the *html_style* config value will override this setting.
- The **pygments_style** setting gives the name of a Pygments style to use for highlighting. This can be overridden by the user in the *pygments_style* config value.
- The **options** section contains pairs of variable names and default values. These options can be overridden by the user in *html_theme_options* and are accessible from all templates as `theme_<name>`.

CSS and roles

sphinx CSS and custom-interpreted-text-roles

See also:

- http://docutils.sourceforge.net/docs/ref/rst/directives.html#custom-interpreted-text-roles
- http://sphinx-doc.org/latest/config.html?highlight=html_static_path#confval-html_static_path

role:: underlined

I've added

```
span.underlined {
    text-decoration: underline;
}
```

to the `default.css` CSS file.

and

```
.. role:: underlined

:underlined:`test underlined`
```

to the test.rst and it is underlined!

Thank You!

Tip: vertical text

```
From: Boris Kheyfets <kheyfboris@gmail.com>
Date: 2012/9/5
Subject: [sphinx-dev] Tip: vertical text
To: sphinx-dev@googlegroups.com
```

Here's a minimal working example:

```
css:
span.vertical {
    writing-mode:tb-rl;
    -webkit-transform:rotate(270deg);
    -moz-transform:rotate(270deg);
    -o-transform: rotate(270deg);
    display:block;
    width:20px;
    height:20px;
}
```

```
rst:
.. role:: vertical

:vertical:`test`
```

default.css file

html_static_path

A list of paths that contain custom static files (such as style sheets or script files). Relative paths are taken as relative to the configuration directory. They are copied to the output directory after the theme's static files, so a file named `default.css` will overwrite the theme's `default.css`.

Changed in version 0.4: The paths in *html_static_path* can now contain subdirectories.

Changed in version 1.0: The entries in *html_static_path* can now be single files.

TIPS

Sphinx templating tips

Custom title page

Custom title page (1)

See also:

- https://bitbucket.org/birkenfeld/sphinx/src/tip/doc/conf.py
- http://sphinx-doc.org/config.html#confval-html_additional_pages
- https://bitbucket.org/birkenfeld/sphinx/src/tip/doc/_templates/

On Fri, Apr 4, 2014 at 1:58 PM, Julia Evans <jwevans01@gmail.com> wrote:

```
I'm new to Sphinx, and I'd like to create a custom title page. I have
figured out how I can suppress the title page by adding 'maketitle': ' ',
in the conf.py file, but how do I replace it with a custom cover page?
And If I can, what format doe sit have to be in?

I'm using the "howto" sphinx template.
```

Sphinx uses its own custom front page in https://bitbucket.org/birkenfeld/sphinx/src/tip/doc/conf.py by doing;

html_sidebars = {'index': ['indexsidebar.html', 'searchbox.html']} html_additional_pages = {'index': 'index.html'}

documented at http://sphinx-doc.org/config.html#confval-html_additional_pages and http://sphinx-doc.org/config.html#confval-html_sidebars]

where index.html and indexsidebar.html are in https://bitbucket.org/birkenfeld/sphinx/src/tip/doc/_templates/

It also uses:

master_doc = 'contents'

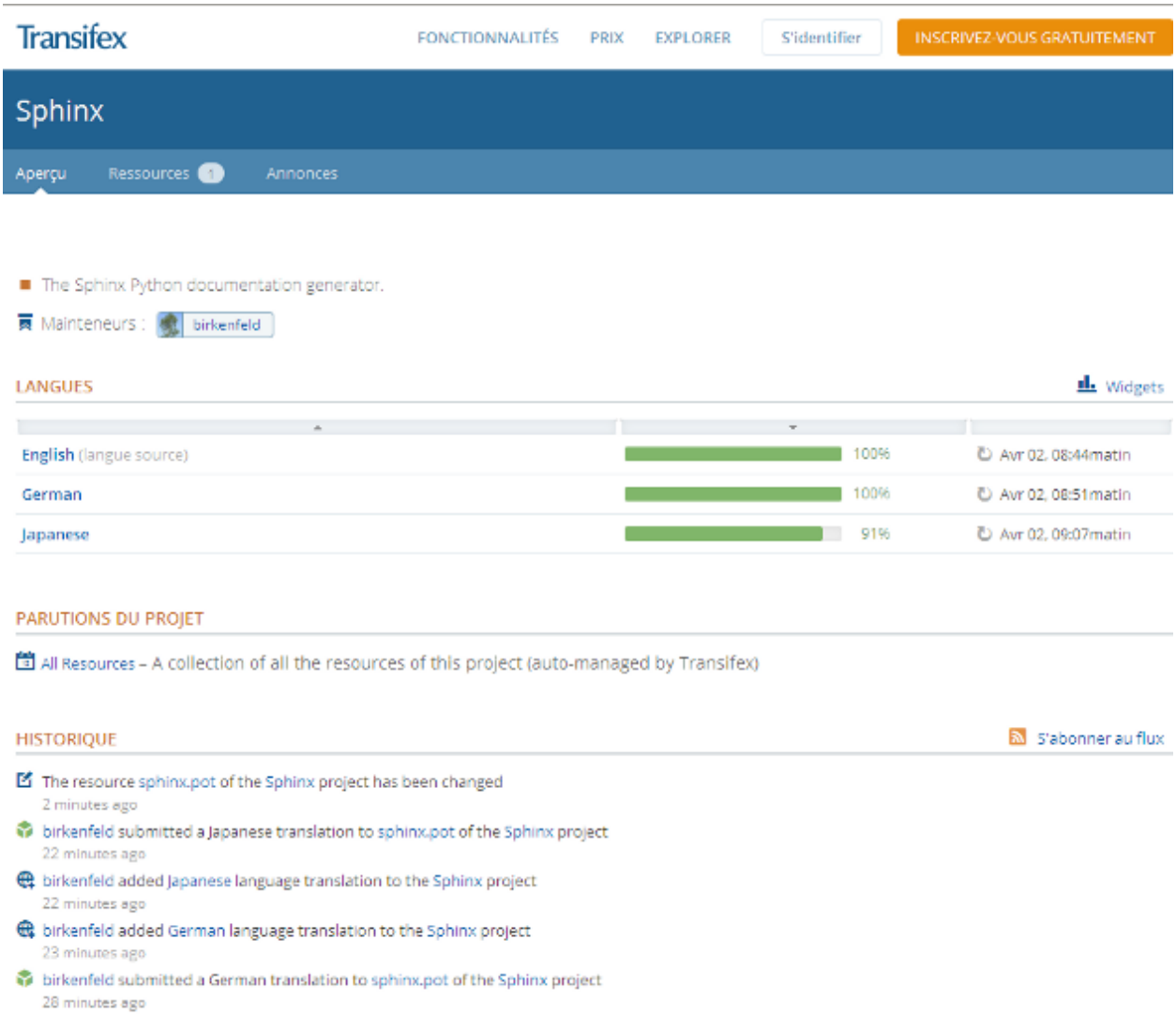
to rename the Sphinx generated main file.

Sphinx translations

Sphinx translations

See also:

- https://www.transifex.com/projects/p/sphinx-1/



Announce

```
Georg Brandl <georg@python.org>
répondre à:  sphinx-users@googlegroups.com
à:   sphinx-users@googlegroups.com
date:    2 avril 2013 11:00
objet:   [sphinx-users] Re: Call for translation updates - Now on transifex
```

Hi,

since editing raw .po files is not a good workflow for everyone, I registered Sphinx on Transifex:

https://www.transifex.com/projects/p/sphinx-1/

If you want to translate/update to your language there, please drop me an email and I will add you to the corresponding language group.

cheers, Georg

```
On 04/01/2013 12:26 PM, Georg Brandl wrote:
> Hi all,
>
> for the pending 1.2 release I'd like to update as many locales as possible.
> If you speak one of the languages that Sphinx supports and can spare the
> time, it would be great if you could look over the message catalog
> (sphinx.po) in your language at
> https://bitbucket.org/birkenfeld/sphinx/src/tip/sphinx/locale and open an
> issue/pull request with the necessary update or the new file.
>
> Thanks, Georg
>
```

Sphinx versions

Sphinx versions

Sphinx 1.6.3 (2017-07-02)

See also:

- https://github.com/sphinx-doc/sphinx/blob/master/CHANGES
- https://github.com/sphinx-doc/sphinx/tree/1.6.3

Features added

- latex: hint that code-block continues on next page (refs: #3764, #3792)

Bugs fixed

- #3821: Failed to import sphinx.util.compat with docutils-0.14rc1
- #3829: sphinx-quickstart template is incomplete regarding use of alabaster
- #3772: 'str object' has no attribute 'filename'

- Emit wrong warnings if citation label includes hyphens (refs: #3565)
- #3858: Some warnings are not colored when using –color option
- #3775: Remove unwanted whitespace in default template
- #3835: sphinx.ext.imgmath fails to convert SVG images if project directory name contains spaces
- #3850: Fix color handling in make mode's help command
- #3865: use of self.env.warn in sphinx extension fails
- #3824: production lists apply smart quotes transform since Sphinx 1.6.1
- latex: fix `\sphinxbfcode` swallows initial space of argument
- #3878: Quotes in auto-documented class attributes should be straight quotes in PDF output
- #3881: LaTeX figure floated to next page sometimes leaves extra vertical whitespace
- #3885: duplicated footnotes raises IndexError
- #3873: Failure of deprecation warning mechanism of `sphinx.util.compat.Directive`
- #3874: Bogus warnings for "citation not referenced" for cross-file citations
- #3860: Don't download images when builders not supported images
- #3860: Remote image URIs without filename break builders not supported remote images
- #3833: command line messages are translated unintentionally with `language` setting.
- #3840: make checking `epub_uid` strict
- #3851, #3706: Fix about box drawing characters for PDF output
- #3900: autosummary could not find methods
- #3902: Emit error if `latex_documents` contains non-unicode string in py2

Sphinx 1.2.3 (2014-09-01)

See also:

- https://github.com/sphinx-doc/sphinx/blob/master/CHANGES
- https://github.com/sphinx-doc/sphinx/tree/1.2.1

Release 1.1.1 (Nov 1, 2011)

- #791: Fix QtHelp, DevHelp and HtmlHelp index entry links.
- #792: Include "sphinx-apidoc" in the source distribution.
- #797: Don't crash on a misformatted glossary.
- #801: Make intersphinx work properly without SSL support.
- #805: Make the `Sphinx.add_index_to_domain` method work correctly.
- #780: Fix Python 2.5 compatibility.

Release 1.1 (Oct 9, 2011)

See also:

- http://sphinx-doc.org/changes.html#release-1-1-oct-9-2011

Incompatible changes

- The `py:module` directive doesn't output its `platform` option value anymore. (It was the only thing that the directive did output, and therefore quite inconsistent.)
- Removed support for old dependency versions; requirements are now:
  - Pygments >= 1.2
  - Docutils >= 0.7
  - Jinja2 >= 2.3

Features added

- Added Python 3.x support.
- New builders and subsystems:
  - Added a Texinfo builder.
  - Added i18n support for content, a `gettext` builder and related utilities.
  - Added the `websupport` library and builder.
  - #98: Added a `sphinx-apidoc` script that autogenerates a hierarchy of source files containing autodoc directives to document modules and packages.
  - #273: Add an API for adding full-text search support for languages other than English. Add support for Japanese.
- Markup:
  - #138: Added an `index` role, to make inline index entries.
  - #454: Added more index markup capabilities: marking see/seealso entries, and main entries for a given key.
  - #460: Allowed limiting the depth of section numbers for HTML using the `toctree`'s `numbered` option.
  - #586: Implemented improved `glossary` markup which allows multiple terms per definition.
  - #478: Added `py:decorator` directive to describe decorators.
  - C++ domain now supports array definitions.
  - C++ domain now supports doc fields (`:param x:` inside directives).
  - Section headings in *`only`* directives are now correctly handled.
  - Added `emphasize-lines` option to source code directives.
  - #678: C++ domain now supports superclasses.
- HTML builder:
  - Added `pyramid` theme.

  - #559: ::*html_add_permalinks* is now a string giving the text to display in permalinks.
  - #259: HTML table rows now have even/odd CSS classes to enable "Zebra styling".
  - #554: Add theme option `sidebarwidth` to the basic theme.
- Other builders:
  - #516: Added new value of the ::*latex_show_urls* option to show the URLs in footnotes.
  - #209: Added ::*text_newlines* and ::*text_sectionchars* config values.
  - Added ::*man_show_urls* config value.
  - #472: linkcheck builder: Check links in parallel, use HTTP HEAD requests and allow configuring the timeout. New config values: ::*linkcheck_timeout* and ::*linkcheck_workers*.
  - #521: Added ::*linkcheck_ignore* config value.
  - #28: Support row/colspans in tables in the LaTeX builder.
- Configuration and extensibility:
  - #537: Added ::*nitpick_ignore*.
  - #306: Added *env-get-outdated* event.
  - `Application.add_stylesheet()` now accepts full URIs.
- Autodoc:
  - #564: Add ::*autodoc_docstring_signature*. When enabled (the default), autodoc retrieves the signature from the first line of the docstring, if it is found there.
  - #176: Provide `private-members` option for autodoc directives.
  - #520: Provide `special-members` option for autodoc directives.
  - #431: Doc comments for attributes can now be given on the same line as the assignment.
  - #437: autodoc now shows values of class data attributes.
  - autodoc now supports documenting the signatures of `functools.partial` objects.
- Other extensions:
  - Added the `sphinx.ext.mathjax` extension.
  - #443: Allow referencing external graphviz files.
  - Added `inline` option to graphviz directives, and fixed the default (block-style) in LaTeX output.
  - #590: Added `caption` option to graphviz directives.
  - #553: Added `testcleanup` blocks in the doctest extension.
  - #594: ::*trim_doctest_flags* now also removes `<BLANKLINE>` indicators.
  - #367: Added automatic exclusion of hidden members in inheritance diagrams, and an option to selectively enable it.
  - Added ::*pngmath_add_tooltips*.
  - The math extension displaymath directives now support `name` in addition to `label` for giving the equation label, for compatibility with Docutils.
- New locales:
  - #221: Added Swedish locale.

  - #526: Added Iranian locale.
  - #694: Added Latvian locale.
  - Added Nepali locale.
  - #714: Added Korean locale.
  - #766: Added Estonian locale.
- Bugs fixed:
  - #778: Fix "hide search matches" link on pages linked by search.
  - Fix the source positions referenced by the "viewcode" extension.

1.3.2 Authorea

See also:

- https://www.authorea.com/

Contents

Introduction

Authorea is the collaborative platform for research.

Write and manage your technical documents in one place.

1.3.3 Doxygen

See also:

- http://www.stack.nl/~dimitri/doxygen/index.html
- https://github.com/doxygen/doxygen.git
- https://github.com/doxygen/doxygen
- *Doxygen contributed extensions*

Fig. 6: *Doxygen logo*

Contents

Introduction

Doxygen is a documentation system for C++, C, Java, Objective-C, Python, IDL(Corba and Microsoft flavors), Fortran, VHDL, PHP, C#, and to some extent D.

It can help you in three ways:

- It can generate an on-line documentation browser (in HTML) and/or an off-line reference manual (in $mbox{LaTeX}$) from a set of documented source files. There is also support for generating output in RTF (MS-Word), PostScript, hyperlinked PDF, compressed HTML, and Unix man pages. The documentation is extracted directly from the sources, which makes it much easier to keep the documentation consistent with the source code.
- You can configure doxygen to extract the code structure from undocumented source files. This is very useful to quickly find your way in large source distributions. You can also visualize the relations between the various elements by means of include dependency graphs, inheritance diagrams, and collaboration diagrams, which are all generated automatically.
- You can even *abuse* doxygen for creating normal documentation (as I did for this manual).

Doxygen is developed under Linux and Mac OS X, but is set-up to be highly portable. As a result, it runs on most other Unix flavors as well. Furthermore, executables for Windows are available.

Furthermore, executables for Windows are available.

Example

```
///
/// <summary>
/// CR_RFID_DecouvrirLecteurs() Retourne la liste des lecteurs PS/SC de type
/// sous la forme d une liste de noms de la forme : "CodeRousseau RFID".
/// </summary>
/// <param name="ppListeNomsLecteursRFID"> [out] pointeur sur une liste de noms de␣
↪lecteurs RFID.</param>.
/// <remarks>
/// </remarks>
/// <returns>
/// - COMMAND_SUCCESS: La commande s'est bien passee.
/// - COMMAND_FAILED: La commande a echoue.
/// </returns>
extern DWORD CR_RFID_DecouvrirLecteurs(ST_LIST_STRING **ppListeNomsLecteursRFID);
```

Doxygen source code

See also:

- https://github.com/doxygen/doxygen
- https://github.com/doxygen/doxygen.git

In May 2013, Doxygen moved from subversion to git hosted at github:

```
https://github.com/doxygen/doxygen
```

Enjoy,

Dimitri van Heesch (dimitri at stack.nl)

Issues, bugs, requests, ideas

Use the bug tracker to report bugs:

- **current list:**
  - Bugzilla
- **Submit a new bug or feature request**
  - Enter bug

Projects using doxygen

Projects using doxygen

See also:

- http://www.stack.nl/~dimitri/doxygen/projects.html

Contents

Introduction

Doxygen supports a number of output formats where HTML is the most popular one. I've gathered some nice examples of real-life projects using doxygen.

These are part of a larger list of projects that use doxygen. If you know other projects, let me know and I'll add them.

C projects

C doxygen projects

libpcsclite

See also:

- http://pcsclite.alioth.debian.org/api/index.html

libusbk

See also:

- http://libusbk.sourceforge.net/UsbK3/index.html
- http://code.google.com/p/usb-travis/

C# projects

C Projects using doxygen

ini_parser

See also:

csharp_ini_parser

ParsingException

See also:

http://code.google.com/p/ini-parser/source/browse/trunk/%20ini-parser/src/IniFileParser/Exceptions.cs

```
using System;

namespace IniParser
{
    /// <summary>
    /// Represents an error ococcurred while parsing data
    /// </summary>
    public class ParsingException : Exception
    {
        /// <summary>
        /// Initializes a new instance of the <see cref="ParsingException"/> class.
        /// </summary>
        public ParsingException(){ }

        /// <summary>
        /// Initializes a new instance of the <see cref="ParsingException"/> class.
        /// </summary>
        /// <param name="msg">The message describing the exception cause.</param>
        public ParsingException(string msg)
```

(continues on next page)

(continued from previous page)

```
            : base(msg) { }

        /// <summary>
        /// Initializes a new instance of the <see cref="ParsingException"/> class.
        /// </summary>
        /// <param name="msg">The message describing the exception cause.</param>
        /// <param name="innerException">An inner exception.</param>
        public ParsingException(string msg, Exception innerException)
            : base(msg, innerException) { }
    }
}
```

KeyDataCollection

See also:

http://code.google.com/p/ini-parser/source/browse/trunk/%20ini-parser/src/IniFileParser/KeyDataCollection.cs

```
using System;
using System.Collections;
using System.Collections.Generic;

namespace IniParser
{
    /// <summary>
    /// <para>Represents a collection of Keydata.</para>
    /// </summary>
    public class KeyDataCollection : ICloneable, IEnumerable<KeyData>
    {
        #region Initialization

        /// <summary>
        /// Initializes a new instance of the <see cref="KeyDataCollection"/> class.
        /// </summary>
        public KeyDataCollection()
        {
            _keyData = new Dictionary<string, KeyData>();
        }

        /// <summary>
        /// Initializes a new instance of the <see cref="KeyDataCollection"/> class
        /// from a previous instance of <see cref="KeyDataCollection"/>.
        /// </summary>
        /// <remarks>
        /// Data is deeply copied
        /// </remarks>
        /// <param name="ori">
        /// The instance of the <see cref="KeyDataCollection"/> class
        /// used to create the new instance.</param>
        public KeyDataCollection(KeyDataCollection ori)
        {
            _keyData = new Dictionary<string, KeyData>();
            foreach ( string key in _keyData.Keys )
                _keyData.Add(key, (KeyData)ori._keyData[key].Clone() );
        }
```

(continues on next page)

(continued from previous page)

```
        #endregion

        #region Properties

        /// <summary>
        /// Gets or sets the value of a concrete key.
        /// </summary>
        /// <remarks>
        /// If we try to assign the value of a key which doesn't exists,
        /// a new key is added with the name and the value is assigned to it.
        /// </remarks>
        /// <param name="keyName">Name of the key</param>
        /// <returns>
        /// The string with key's value or null
        /// if the key was not found.
        /// </returns>
        public string this[string keyName]
        {
            get
            {
                if (_keyData.ContainsKey(keyName))
                    return _keyData[keyName].Value;

                return null;
            }

            set
            {
                if (!_keyData.ContainsKey(keyName))
                    return;

                _keyData[keyName].Value = value;

            }
        }

        /// <summary>
        /// Return the number of keys in the collection
        /// </summary>
        /// <value>An integer with the number of keys in the collection.</value>
        public int Count
        {
            get { return _keyData.Count; }
        }

        #endregion

        #region Public Methods

        /// <summary>
        /// Adds a new key with the specified name and empty value and comments
        /// </summary>
        /// <remarks>
        /// A valid key name is a string with NO blank spaces.
        /// </remarks>
        /// <param name="keyName">New key to be added.</param>
```

(continues on next page)

(continued from previous page)

```
        /// <returns>
        /// <c>true</c> if a new empty key was added
        /// <c>false</c> otherwise.
        /// </returns>
        /// <exception cref="ArgumentException">If the key name is not valid.</
↪exception>
        public bool AddKey(string keyName)
        {
            //Checks valid key name
            //if ( !Assert.StringHasNoBlankSpaces(keyName) )
            //    throw new ArgumentException("Key name is not valid");

            if ( !_keyData.ContainsKey(keyName) )
            {
                _keyData.Add(keyName, new KeyData(keyName));
                return true;
            }

            return false;
        }

        /// <summary>
        /// Adds a new key with the specified name and value and comments
        /// </summary>
        /// <remarks>
        /// A valid key name is a string with NO blank spaces.
        /// </remarks>
        /// <param name="keyName">New key to be added.</param>
        /// <param name="keyData">KeyData instance.</param>
        /// <returns>
        /// <c>true</c> if a new empty key was added
        /// <c>false</c> otherwise.
        /// </returns>
        /// <exception cref="ArgumentException">If the key name is not valid.</
↪exception>
        public bool AddKey(string keyName, KeyData keyData)
        {
            if (AddKey(keyName))
            {
                _keyData[keyName] = keyData;
                return true;
            }

            return false;

        }

        /// <summary>
        /// Adds a new key with the specified name and value and comments
        /// </summary>
        /// <remarks>
        /// A valid key name is a string with NO blank spaces.
        /// </remarks>
        /// <param name="keyName">New key to be added.</param>
        /// <param name="keyValue">Value associated to the kyy.</param>
        /// <returns>
        /// <c>true</c> if a new empty key was added
```

(continues on next page)

(continued from previous page)

```
        /// <c>false</c> otherwise.
        /// </returns>
        /// <exception cref="ArgumentException">If the key name is not valid.</
↪exception>
        public bool AddKey(string keyName, string keyValue)
        {
            if (AddKey(keyName))
            {
                _keyData[keyName].Value = keyValue;
                return true;
            }

            return false;

        }

        /// <summary>
        /// Retrieves the data for a specified key given its name
        /// </summary>
        /// <param name="keyName">Name of the key to retrieve.</param>
        /// <returns>
        /// A <see cref="KeyData"/> instance holding
        /// the key information or <c>null</c> if the key wasn't found.
        /// </returns>
        public KeyData GetKeyData(string keyName)
        {
            if (_keyData.ContainsKey(keyName))
                return _keyData[keyName];
            return null;
        }

        /// <summary>
        /// Sets the key data associated to a specified key.
        /// </summary>
        /// <param name="data">The new <see cref="KeyData"/> for the key.</param>
        public void SetKeyData(KeyData data)
        {
            if (data != null)
            {
                if (_keyData.ContainsKey(data.KeyName))
                    RemoveKey(data.KeyName);

                AddKey(data.KeyName, data);
            }
        }

        /// <summary>
        /// Gets if a specifyed key name exists in the collection.
        /// </summary>
        /// <param name="keyName">Key name to search</param>
        /// <returns><c>true</c> if a key with the specified name exists in the␣
↪collectoin
        /// <c>false</c> otherwise</returns>
        public bool ContainsKey(string keyName)
        {
            return _keyData.ContainsKey(keyName);
        }
```

(continues on next page)

(continued from previous page)

```
        /// <summary>
        /// Deletes a previously existing key, including its associated data.
        /// </summary>
        /// <param name="keyName">The key to be removed.</param>
        /// <returns>
        /// <c>true</c> if a key with the specified name was removed
        /// <c>false</c> otherwise.
        /// </returns>
        public bool RemoveKey(string keyName)
        {
            return _keyData.Remove(keyName);
        }

        #endregion

        #region IEnumerable<KeyData> Members

        /// <summary>
        /// Allows iteration througt the collection.
        /// </summary>
        /// <returns>A strong-typed IEnumerator </returns>
        public IEnumerator<KeyData> GetEnumerator()
        {
            foreach ( string key in _keyData.Keys )
                yield return _keyData[key];
        }

        #region IEnumerable Members

        /// <summary>
        /// Implementation needed
        /// </summary>
        /// <returns>A weak-typed IEnumerator.</returns>
        IEnumerator IEnumerable.GetEnumerator()
        {
            return _keyData.GetEnumerator();
        }

        #endregion

        #endregion

        #region ICloneable Members


        /// <summary>
        /// Creates a new object that is a copy of the current instance.
        /// </summary>
        /// <returns>
        /// A new object that is a copy of this instance.
        /// </returns>
        public object Clone()
        {
            return new KeyDataCollection(this);
        }
```

(continues on next page)

(continued from previous page)

```
        #endregion

        #region Non-public Members

        /// <summary>
        /// Collection of KeyData for a given section
        /// </summary>
        private readonly Dictionary<string, KeyData> _keyData;

        #endregion

    }
}
```

Doxygen formats

Doxygen formats

Doxygen dash docset

See also:

- *Dash*
- http://kapeli.com/docsets#doxygen

Description

Doxygen can generate docsets from source files of C, C++, C#, PHP, Objective-C, Java, Python (and some others).

These are the entries you need to add into your Doxygen config file to make it generate a docset (note: the last 3 entries are optional):

```
GENERATE_DOCSET   = YES
DISABLE_INDEX     = YES
SEARCHENGINE      = NO
GENERATE_TREEVIEW = NO
```

When Doxygen is done generating the documentation, run make inside the generated folder.

Afterwards, scroll down to the bottom of this page where you'll find some tips on how to improve your docset.

Doxygen versions

Doxygen versions

See also:

- http://www.stack.nl/~dimitri/doxygen/manual/changelog.html

Doxygen 1.8.8 (21-08-2014)

See also:

- http://www.stack.nl/~dimitri/doxygen/manual/changelog.html#log_1_8_8
- Bug 731947 - Support for PlantUML [view]
- Add BREAD_CRUMB_TRAIL. [view]

Doxygen 1.8.7 (21-04-2014)

See also:

- http://www.stack.nl/~dimitri/doxygen/manual/changelog.html#log_1_8_7

Doxygen 1.8.5 (23-08-2013)

See also:

- http://www.stack.nl/~dimitri/doxygen/changelog.html

Contents

Changes

Doxygen's source code is now managed using git and GitHub.

Automatic builds and regression tests are scheduled via Travis CI.

Configuration data for the config file, the documentation, and the wizard are now produced from a single source (thanks to Albert)

All translation files have been migrated to UTF-8 (thanks to Petr Prikryl)

Added black box testing framework and a set of tests.

Doxygen 1.8.4 (19-05-2013)

See also:

- http://www.stack.nl/~dimitri/doxygen/changelog.html
- http://www.stack.nl/~dimitri/doxygen/manual/changelog.html#log_1_8_4

Contents

Changes

- id 686384: When INLINE_SIMPLE_STRUCTS is enabled, also structs with simple typedefs will be inlined.
- Doxywizard: scrolling with mouse wheel no longer affects the values in the expert view.
- id 681733: More consistent warnings and errors.

New features

- Added support for "clang assisted parsing", which allows the code to also be parsed via libclang (C/C++ frontend of LLVM) and can improve the quality of the syntax highting, cross-references, and call graphs, especially for template heavy C++ code. To enable this feature you have to configure doxygen with the –with-libclang option. Then you get two new configuration options: CLANG_ASSISTED_PARSING to enable or disable parsing via clang and CLANG_OPTIONS to pass additional compiler options needed to compile the files. Note that enabling this feature has a significant performance penality.
- Included patch donated by Intel which adds Docbook support. This can be enabled via GENERATE_DOCBOOK and the output location can be controlled using DOCBOOK_OUTPUT. Docbook specific sections can be added using docbookonly . . . enddocbookonly
- Added support for UNO IDL (interace language used in Open/Libre Office), thanks to Michael Stahl for the patch.

Stores data gathered by doxygen in a sqlite3 database

- Included patch by Adrian Negreanu which stores data gathered by doxygen in a **sqlite3** database. Currently still work in progress and can only be enabled using –with-sqlite3 during ./configure.

SVG

- For interactive SVG graphs, edges are now highlighted when hovered by the mouse.

Statistics

- Include patch by Adrian Negreanu to show duration statistics after a run. You can enable this by running doxygen with the **-d Time** option.

LATEX_EXTRA_FILES

- Included patch by Markus Geimer which adds a new option LATEX_EXTRA_FILES which works similarily to HTML_EXTRA_FILES in that it copied specified files to the LaTeX output directory.

C++11

- id 698223: Added support for C++11 keyword alignas

Added support for processing DocSets

- id 693178: Added support for processing DocSets with *Dash* (thanks to Bogdan Popescu for the patch

Other

- id 684782: Added option EXTERNAL_PAGES which can be used to determine whether or not pages importated via tags will appear under related pages (similar to EXTERNAL_GROUPS).
- id 692227: Added new MathJax command MATHJAX_CODEFILE which supports including a file with MathJax related scripting to be inserted before the MathJax script is loaded. Thanks to Albert for the patch.
- id 693537: Comments in the config file starting with ## will now be kept when upgrading the file with doxygen -u (and doxygen -s -u). Thanks to Albert for the patch.
- id 693422: Adds support for Latvian (thanks to a patch by Lauris).
- Included language updates for Ukrainian, Romanian, and Korean

Doxygen 1.8.0 (25-02-2012)

See also:

http://www.stack.nl/~dimitri/doxygen/changelog.html

Contents

* *tableofcontents*
* *targets for 64 bits*

Changes

Auto list items can now consist of multiple paragraphs.

The indentation of the (first line) of a new paragraph detemines to which list item the paragraph belongs or if it marks the end of the list.

When UML_LOOK is enabled, relations shown on the edge of a graph are not shown as attributes (conform to the UML notation)

Updated the manual and improved the look

Made the contents in the navigation tree more consistent for groups, pages with subpages, and grouped subpages.

id 669079: Latex: made the margins of latex page layout smaller using the geometry package.

doxytag removed

The tool doxytag has been declared obsolete and is removed (it wasn't working properly anyway). Same goes for the installdox script.

Updated the copyright in source code and doxywizard "about" to 2012. id 668008: HTML version of the manual now has the treeview enabled for easier navigation.

New features

Markdown support

See also:

- http://michelf.com/projects/php-markdown/extra/#table
- http://freewisdom.org/projects/python-markdown/Fenced_Code_Blocks

Added support for *Markdown* formatting. This is enabled by default, but can be disabled by setting MARKDOWN_SUPPORT to NO.

When enabled the following is processed differently:

- tabs are converted to spaces according to TAB_SIZE.
- blockquotes are created for lines that start with one or more >'s (amount of >'s detemine the indentation level).
- emphasis using *emphasize this* or emphasis this or strong emphasis using **emphasis this**. Unlike classic Markdown 'some_great_indentifier' is not touched.
- code spans can be created using back-ticks, i.e. *here's an example*
- Using three or more -'s or * 's alone on a line with only spaces will produce a horizontal ruler.
- A header can be created by putting a ===== (for h1) or —– (for h2) on the next line or by using 1 to 6 #'s at the start of a line for h1-h6.

- auto lists item can also start with + or * instead of only -
- ordered lists can be made using 1. 2. . . . labels.
- verbatim blocks can be produced by indenting 4 additional spaces. Note that doxygen uses a relative indent of 4 spaces, not an absolute indent like Markdown does.
- Markdown style hyperlinks and hyperlink references.
- Simple tables can be created using the Markdown Extra format.
- Fenced code blocks are also supported, include language selection.
- files with extension .md or .markdown are converted to related pages.
- See the section about Markdown support in the manual for details.

It is now possible to add user defined tabs or groups of tabs to the navigation menu using the layout file (see the section of the manual about customizing the output for details).

tableofcontents

Added new command tableofcontents (or [TOC] if you prefer Markdown) which can be used in a related page with sections to produce a table of contents at the top of the HTML page (for other formats the command has no effect).

When using SVG images and INTERACTIVE_SVG is enabled, a print icon will be visible along with the navigation controls to facilitate printing of the part of the graph that is visible on screen.

Added obfuscation of email addresses for the HTML output to make email harvesting more difficult.

targets for 64 bits

Added build targets for 64 bit Windows (thanks to Vladimir Simonov). The installer script is also updated to install a 64 bit version of doxygen on 64 bit systems and the 32 bit version on 32 bit systems.

Added support for using the HTML tag <blockquote> in comments.

Included patch by Gauthier Haderer that fixes some issues with the dbus XML parser.

Added support for Markdown style fenced code blocks.

Added option to @code command to force parsing and syntax highlighting according to a particular language.

Section of pages are now added to the navigation index.

Added support for cell alignment and table header shading in LaTeX and RTF output.

Added -d filteroutput option to show the output of an input filter (thanks to Albert for the patch).

id 668010: Latex: for Windows doxygen new generates a makepdf.bat file in the latex output dir to create the latex documentation.

Doxygen 1.7.6.1

See also:

http://www.stack.nl/~dimitri/doxygen/changelog.html

Changes

Doxygen now reports its cache usage (for the symbol and the lookup cache) at the end of a run (if QUIET=NO), and recommends settings for SYMBOL_CACHE_SIZE and LOOKUP_CACHE_SIZE for your project if either cache is too small.

New features

Added new option LOOKUP_CACHE_SIZE to control the internal cache doxygen uses to find symbols given their name and a context.

- Python: added support for @staticmethod

1.3.4 Javadoc

See also:

- http://johnmacfarlane.net/pandoc/index.html

Contents

Introduction

`Javadoc` est un outil développé par Sun Microsystems permettant de créer une documentation d'API en format HTML depuis les commentaires présents dans un code source en Java.

Javadoc est le standard industriel pour la documentation des classes Java.

La plupart des IDEs créent automatiquement le javadoc HTML.

1.3.5 Jekyll

See also:

- http://jekyllrb.com/

Contents

Overview

Jekyll is a simple, blog-aware, static site generator.

It takes a template directory containing raw text files in various formats, runs it through Markdown (or Textile) and Liquid converters, and spits out a complete, ready-to-publish static website suitable for serving with your favorite web server.

Jekyll also happens to be the engine behind GitHub Pages, which means you can use Jekyll to host your project's page, blog, or website from GitHub's servers for free.

Tools

Jekyll tools

Octopress

See also:

- http://octopress.org/docs/
- https://twitter.com/octopress
- https://twitter.com/imathis

Octopress is [Jekyll](https://github.com/mojombo/jekyll) blogging at its finest.

1. **Octopress sports a clean responsive theme** written in semantic HTML5, focused on readability and friendliness toward mobile devices.
2. **Code blogging is easy and beautiful.** Embed code (with [Solarized](http://ethanschoonover.com/solarized) styling) in your posts from gists, jsFiddle or from your filesystem.
3. **Third party integration is simple** with built-in support for Pinboard, Delicious, GitHub Repositories, Disqus Comments and Google Analytics.
4. **It's easy to use.** A collection of rake tasks simplifies development and makes deploying a cinch.
5. **Ships with great plug-ins** some original and others from the Jekyll community — tested and improved.

1.3.6 JSDoc

See also:

- http://usejsdoc.org/

Contents

Getting Started

JSDoc 3 is an API documentation generator for JavaScript, similar to JavaDoc or PHPDoc.

You add documentation comments directly to your source code, right along side the code itself.

The JSDoc Tool will scan your source code, and generate a complete HTML documentation website for you.

Bootstrap themes

JSDoc bootstrap themes

docstrap

See also:

- https://github.com/terryweiss/docstrap

Description

DocStrap is Bootstrap based template for JSDoc3.

In addition, it includes all of the themes from Bootswatch giving you a great deal of look and feel options for your documentation, along with a simple search.

Additionally, it adds some options to the conf.json file that gives you even more flexibility to tweak the template to your needs.

It will also make your teeth whiter.

1.3.7 Mkdocs

See also:

- http://www.mkdocs.org/

Contents

Overview

`MkDocs` is a fast, simple and downright gorgeous static site generator that's geared towards building project documentation.

Documentation source files are written in Markdown, and configured with a single YAML configuration file.

The documentation site that we've just built only uses static files so you'll be able to host it from pretty much anywhere.

GitHub project pages and Amazon S3 are good hosting options.

Upload the contents of the entire site directory to wherever you're hosting your website from and you're done.

1.3.8 Pandoc

See also:

- http://johnmacfarlane.net/pandoc/index.html
- https://github.com/jgm/pandoc
- https://raw.github.com/jgm/pandoc/master/README
- haskell_language

Contents

Introduction

If you need to convert files from one markup format into another, pandoc is your swiss-army knife.

Pandoc can convert documents in markdown, reStructuredText, textile, HTML, DocBook, LaTeX, or MediaWiki markup to:

- HTML formats: XHTML, HTML5, and HTML slide shows using Slidy, Slideous, S5, or DZSlides.
- Word processor formats: Microsoft Word docx, OpenOffice/LibreOffice ODT, OpenDocument XML
- Ebooks: EPUB version 2 or 3, FictionBook2
- Documentation formats: DocBook, GNU TexInfo, Groff man pages
- TeX formats: LaTeX, ConTeXt, LaTeX Beamer slides
- PDF via LaTeX
- Lightweight markup formats: Markdown, reStructuredText, AsciiDoc, MediaWiki markup, Emacs Org-Mode, Textile

See also:

- https://raw.github.com/jgm/pandoc/master/README
- [markdown]: http://daringfireball.net/projects/markdown/
- [reStructuredText]: http://docutils.sourceforge.net/docs/ref/rst/introduction.html
- [S5]: http://meyerweb.com/eric/tools/s5/
- [Slidy]: http://www.w3.org/Talks/Tools/Slidy/
- [Slideous]: http://goessner.net/articles/slideous/
- [HTML]: http://www.w3.org/TR/html40/
- [HTML 5]: http://www.w3.org/TR/html5/
- [XHTML]: http://www.w3.org/TR/xhtml1/
- [LaTeX]: http://www.latex-project.org/

- [beamer]: http://www.tex.ac.uk/CTAN/macros/latex/contrib/beamer
- [ConTeXt]: http://www.pragma-ade.nl/
- [RTF]: http://en.wikipedia.org/wiki/Rich_Text_Format
- [DocBook]: http://www.docbook.org/
- [OPML]: http://dev.opml.org/spec2.html
- [OpenDocument]: http://opendocument.xml.org/
- [ODT]: http://en.wikipedia.org/wiki/OpenDocument
- [Textile]: http://redcloth.org/textile
- [MediaWiki markup]: http://www.mediawiki.org/wiki/Help:Formatting
- [groff man]: http://developer.apple.com/DOCUMENTATION/Darwin/Reference/ManPages/man7/groff_man.7.html
- [Haskell]: http://www.haskell.org/
- [GNU Texinfo]: http://www.gnu.org/software/texinfo/
- [Emacs Org-Mode]: http://orgmode.org
- [AsciiDoc]: http://www.methods.co.nz/asciidoc/
- [EPUB]: http://www.idpf.org/
- [GPL]: http://www.gnu.org/copyleft/gpl.html "GNU General Public License"
- [DZSlides]: http://paulrouget.com/dzslides/
- [ISO 8601 format]: http://www.w3.org/TR/NOTE-datetime
- [Word docx]: http://www.microsoft.com/interop/openup/openxml/default.aspx
- [PDF]: http://www.adobe.com/pdf/
- [reveal.js]: http://lab.hakim.se/reveal-js/
- [FictionBook2]: http://www.fictionbook.org/index.php/Eng:XML_Schema_Fictionbook_2.1

Pandoc source code

Pandoc has a publicly accessible git repository at github. To get a local copy of the source:

```
git clone git://github.com/jgm/pandoc.git
```

After cloning the repository (and in the future after pulling from it), you should do:

```
git submodule update --init
```

to pull in changes to the templates.

You can automate this by creating a file .git/hooks/post-merge with the contents:

```
#!/bin/sh
git submodule update --init
```

and making it executable:

```
chmod +x .git/hooks/post-merge
```

The modules Text.Pandoc.Definition, Text.Pandoc.Builder, and Text.Pandoc.Generics are in a separate package pandoc-types.

The code can be found in this repository (http://github.com/jgm/pandoc-types).

Versions

Pandoc Versions

Pandoc 1.13.1

1.4 Good documentation

Contents

See also:

- http://brikis98.blogspot.de/2014/05/you-are-what-you-document.html

1.4.1 CakePHP

See also:

- http://cakephp.org/
- http://book.cakephp.org/2.0/fr/contributing/documentation.html

1.4.2 Passlib

See also:

https://passlib.readthedocs.io/en/stable/index.html

1.4.3 Sfepy

See also:

http://sfepy.org/doc-devel/index.html

1.4.4 Shark

See also:

- http://image.diku.dk/shark/sphinx_pages/build/html/index.html

1.5 Formats Documentation

1.5.1 Input formats

Markdown

See also:

- http://fr.wikipedia.org/wiki/Markdown
- http://en.wikipedia.org/wiki/Markdown
- http://michelf.com/projects/php-markdown/extra/#table
- http://devdocs.io/markdown/

Contents

Introduction

Markdown est un langage de balisage léger créé par *John Gruber* et Aaron Swartz.

Le but de la syntaxe Markdown est d'offrir une syntaxe facile à lire et à écrire.

C'est-à-dire qu'un document formaté selon Markdown devrait pouvoir être publié comme tel, en texte, sans donner l'impression qu'il a été marqué par des balises ou des instructions de formatage.

Bien que la syntaxe Markdown ait été influencée par plusieurs filtres de conversion de texte vers HTML existants — incluant Setext, atx, Textile, reStructuredText, Grutatext et EtText — la source d'inspiration principale de la syntaxe Markdown est le format du courrier électronique en mode texte.

People

Markdown people

Contents

Aaron Swartz

See also:

- aaron_swartz

John Gruber

See also:

- http://en.wikipedia.org/wiki/John_Gruber

John Gruber (born 1973) is a writer from the greater Philadelphia, Pennsylvania area of the United States.

Gruber received his Bachelor of Science in computer science from Drexel University.

He worked for Bare Bones Software from 2000 to 2002 and Joyent from 2005 to 2006.

His main project is Daring Fireball, a technology blog that has become his full-time job. His side projects include *Markdown*, a lightweight markup language, and a podcast called The Talk Show that he hosts on the Mule Radio Syndicate network

Tools

Markdown tools

Contents

Documentr

See also:

- http://documentr.org/page/documentr/1.x/home

`documentr` is a Web-based tool to edit and present software documentation. It allows to easily maintain documentation for multiple products and product branches.

Easy to use markup language

`documentr` uses Markdown along with some extensions such as tables and macros.

misaka

See also:

http://misaka.61924.nl/

A Python 2 and 3 binding for Sundown, a really fast Markdown parser implemented in C.

Misaka is written in Cython and C.

And it features a set of Markdown extensions and customizable renderers.

Just like the Sundown binding for Ruby, Redcarpet.

Tutorials

Markdown tutorials

ReStructuredText format

See also:

reStructuredText Sphinx documentation

Openalea Rest Documentation

See also:

- http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/_sources/source/sphinx/rest_syntax.txt
- http://openalea.gforge.inria.fr/doc/openalea/doc/_build/html/source/sphinx/rest_syntax.html

ReStructuredText (reST) Sphinx doc

See also:

reStructuredText Sphinx documentation

ReStructuredText tools

See also:

- http://docutils.sourceforge.net/docs/user/links.html

Docutils ReStructuredText

See also:

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

Description

Docutils is an open-source text processing system for processing plaintext documentation into useful formats, such as HTML, LaTeX, man-pages, open-document or XML.

It includes reStructuredText, the easy to read, easy to use, what-you-see-is-what-you-get plaintext markup language.

Front-end dev tools

rst2html

See also:

- http://docutils.sourceforge.net/docs/user/tools.html#rst2html-py
- http://docutils.sourceforge.net/docs/howto/html-stylesheets.html

Contents

Description

The rst2html.py front end reads standalone reStructuredText source files and produces HTML 4 (XHTML 1) output compatible with modern browsers that support cascading stylesheets (CSS).

A stylesheet is required for proper rendering; a simple but complete stylesheet is installed and used by default (see Stylesheets below).

For example, to process a reStructuredText file "test.txt" into HTML:

rst2html.py test.txt test.html

Now open the "test.html" file in your favorite browser to see the results. To get a footer with a link to the source file, date & time of processing, and links to the Docutils project, add some options:

```
rst2html.py -stg test.txt test.html
```

Stylesheets

rst2html.py inserts into the generated HTML a cascading stylesheet (or a link to a stylesheet, when passing the "–link-stylesheet" option).

A stylesheet is required for proper rendering.

The default stylesheet (docutils/writers/html4css1/html4css1.css, located in the installation directory) is provided for basic use.

To use a different stylesheet, you must specify the stylesheet's location with a "–stylesheet" (for a URL) or "–stylesheet-path" (for a local file) command-line option, or with configuration file settings (e.g. ./docutils.conf or ~/.docutils).

To experiment with styles, please see the guide to writing HTML (CSS) stylesheets for Docutils.

Versions

Docutils versions

See also:

- http://docutils.sourceforge.net/RELEASE-NOTES.html

Docutils 0.12 (2014-07-06)

See also:

- http://docutils.sourceforge.net/HISTORY.html#release-0-12-2014-07-06

docs/ref/rst/directives.txt Update "math" and "csv-table" descriptions.

docutils/parsers/rst/directives/images.py Fix [258] figwidth="image" generates unitless width value.

docutils/parsers/rst/states.py Improve error report when a non-ASCII character is specified as delimiter, quote or escape character under Python 2. Fixes [249] and [250].

docutils/writers/html4css1/__init__.py Don't add newline after inline math. Thanks to Yury G. Kudryashov for the patch.

docutils/writers/latex2e/__init__.py Fix [239] Latex writer glues paragraphs with figure floats. Apply [116] by Kirill Smelkov. Don't hardcode large for subtitle.

docutils/writers/odf_odt/__init__.py Apply patch by Jakub Wilk to fix bug [100].

test/test_error_reporting.py Fix [223] by removing redundant tests we do not have control over.

test/test_nodes.py Apply [115] respect fixed 2to3 string literal conversion behavior.

Release 0.11 (2013-07-22)

See also:

- http://docutils.sourceforge.net/HISTORY.html#release-0-11-2013-07-22

General Apply [2714873] Fix for the overwritting of document attributes. Support embedded aliases within hyperlink references. Fix [228] try local import of docutils components (reader, writer, parser, language module) before global search.

docutils/nodes.py Fix [3601607] node.__repr__() must return str instance.

docutils/parsers/rst/directives/__init__.py Fix [3606028] assert is skipped with python -O.

docutils/parsers/rst/directives/images.py Apply [3599485] node source/line information for sphinx translation.

docutils/parsers/rst/directives/tables.py Fix [210] Python 3.3 checks CVS syntax only if "strict" is True.

docutils/parsers/rst/states.py Fix [157] Line block parsing doesn't like system message. Always import our local copy of roman.py (report Larry Hastings).

docutils/transforms/references.py Fix [3607029] traceback with embedded alias pointing to missing target.

docutils/utils/__init__.py Fix [3596884] exception importing docutils.io.

docutils/writers/html4css1/__init__.py Fix [3600051] for tables in a list, table cells are not compacted. New setting stylesheet_dirs: Comma-separated list of directories where stylesheets are found. Used by stylesheet_path when expanding relative path arguments. New default for math-output: HTML math.css. Avoid repeated class declarations in html4css1 writer (modified version of patch [104]).

docutils/writers/latex2e/__init__.py Drop the simple algorithm replacing straight double quotes with English typographic ones. Activate the SmartQuotes transform if you want this feature. Fix literal use of babel shorthands (straight quote, tilde, . . .). Fix [3603246] Bug in option "–graphicx-option=auto". New setting stylesheet_dirs.

docutils/writers/manpage.py Fix [3607063] handle lines starting with a period. Fix option separating comma was bold (thanks to Bill Morris).

Hovercraft

See also:

- https://github.com/regebro/hovercraft
- http://bartaz.github.com/impress.js/#/bored
- https://github.com/regebro/hovercraft/tree/master/docs

Contents

Hovercraft Documentation

See also:

https://hovercraft.readthedocs.org/en/1.0/

Hovercraft source code

See also:

https://github.com/regebro/hovercraft

Announce

See also:

http://regebro.wordpress.com/2013/02/07/presenting-hovercraft-the-merge-of-convenience-and-cool/

reStructuredText

reStructuredText, reST or RST, is a so called "lightweight" markup language. Although there is nothing lightweight about it, it is in fact massive and have an incredible amount of features, which is one reason it's popular.

It's used a lot within the Python community, and is often used to write everything from readme files to books.

There are other languages that have their benefits, most notably Markdown and textile, but I don't know if any of them have the feature set required for Hovercraft, and I'm used to reStructuredText.

There are several ways to make slides from reStructuredText.

One is included in the docutils library that implements reStructuredText, and it can generate **S5 slides**.

Another is Landslide, which i have used as it has a presenter console. But these generate standard left-to-right slide presentations in HTML. Nothing wrong with that, but it's a bit boring.

Enter impress.js.

impress.js

impress.js is a tool to make HTML presentations that zoom and rotate.

It's cool and I used that to make a zooming/panning version of my talk on intellectual properties. And then I made a presentation about Calendaring for PyCon PL 2012 and PloneConf 2012. And by that time I was seriously tired of it, because you write your presentations in HTML, and that sucks in itself, and then you have to position each slide separately.

That worked fine in the first case, where I had this huge image to zoom around it. But for the Calendaring talk I ended up having to reposition a lot of slides each time I needed to insert or delete a slide. Not a practical solution.

I needed to somehow be able to write reStrcuturedText and get impress.js out. After a false start with a template for Landslide, I hit on the solution: Use docutils to generate XML from reStructuredText and the use XSLT to transform it into an impress.js presentation.

That worked, and the solution is Hovercraft!

Hovercraft!

The merging of convenience and cool!

Hovercraft! is a tool to make impress.js presentations from reStructuredText.

Documentation is currently sparse, but available in the documentation subdirectory.

Why?

As a programmer, I like making my presentations in some sort of text-markup.

GUI tools feel restricted and limited when it comes to creating the presentation, simply writing it in text makes it easier to move things around as you like.

But the tools that exist to make presentations from text-markup will make slideshows that has a sequence of slides from left to right. That was fine until Prezi arrived, with zooms and slides and twists and turns.

But Prezi is a GUI toool. And it's closed source. But the open source community fixed that problem with impress.js.

But impress.js is an HTML tools. Sitting and writing HTML is an annoying pain. **It's not a very smooth tool compared reStructuredText or markdown.**

It's especially annoying since you have to sit and add x/y/zoom/rotation for each slide, and if you insert a new slide in the middle, you have to change everything around.

There are GUI tools to layout impress.js presentations but they are all in alpha-state, and doesn't work very well. They also do not support having presenter notes via impress-console, a feature I of course need. After all, that's why I wrote it.

So, I wanted to make impress.js presentations from reStructuredText.

That turned out to be easy, I make landslide-impress, a template for Landslide that create an impress.js presentation. But I ran into a limitation of Landslide. There was no way to get the position information out from the reStructuredText to impress.js.

As such, with Landslide all I could do with impress.js was slides that boringly went from left to right, completely losing the whole point of impress.js.

So, I have to make something myself. Hence: Hovercraft!

Contributors

Hovercraft! was written by Lennart Regebro <regebro@mail.com>, and is licensed as CC-0, except for the following:

- `reST.xsl` is (c) Michael Alyn Miller <malyn@strangeGizmo.com> and published under a BSD-style license included in reST.xsl itself.
- `impress.js` is (c) Bartek Szopka (@bartaz) released under MIT and GPL licenses. See the impress.js page for more information.

Demo

See also:

- https://github.com/regebro/hovercraft
- http://bubbly.colliberty.com/slides/PyConUS-2013

Hovercraft news

Hovercraft news

Hovercraft 2013

Hovercraft 2013

See also:

- http://regebro.wordpress.com/2013/03/25/pycon-us-2013-wrap-up-ambitions-and-results/

Contents

Slides

See also:

- http://bubbly.colliberty.com/slides/PyConUS-2013

http://rst.ninjs.org

See also:

- http://rst.ninjs.org/
- https://github.com/anru/rsted

Simple online editor for reStructuredText on Flask.

Tex / Latex

See also:

- http://fr.wikipedia.org/wiki/LaTeX

Contents

Introduction

LaTeX, prononcé /la.tx/1 (prononciation) ou /la.tk/, est un langage et un système de composition de documents créé par Leslie Lamport en 1983.

```
I've always pronounced LaTeX "lah-tech," but today I heard its developer
Leslie Lamport pronounce it "lay-tech.
```

Plus exactement, il s'agit d'une collection de macro-commandes destinées à faciliter l'utilisation du « processeur de texte » TeX de Donald Knuth.

Depuis 1993, il est maintenu par le LATEX Project team.

La première version utilisée largement, appelée LaTeX2.09, est sortie en 1984.

Une révision majeure, appelée LaTeX2ϵ est sortie en 1991.

Le nom est l'abréviation de Lamport TeX. On écrit souvent LATEX, le logiciel permettant les mises en forme correspondant au logo.

Du fait de sa relative simplicité, il est devenu la méthode privilégiée d'écriture de documents scientifiques employant TeX.

Il est particulièrement utilisé dans les domaines techniques et scientifiques pour la production de documents de taille moyenne ou importante (thèse ou livre, par exemple).

Néanmoins, il peut être aussi employé pour générer des documents de types variés (par exemple, des lettres, ou des transparents).

Textile

See also:

- http://fr.wikipedia.org/wiki/Textile_%28langage%29
- http://en.wikipedia.org/wiki/Textile_%28markup_language%29

Contents

Introduction

Textile est un langage de balisage léger développé à l'origine par Dean Allen et distribué sous licence GNU.

Textile permet de générer du XHTML valide à partir de son langage de balisage. Il permet aussi d'échapper les caractères spéciaux comme les apostrophes. Originellement pour PHP, il a été implémenté dans d'autres langages de programmation tel que Ruby ou Python.

Textile est disponible, sous forme de plugin, dans de nombreux CMS.

La version 2.0 beta est sortie en 2004, la version 2.0 en 2006.

1.5.2 Input/output formats

L'Extensible Markup Language (XML)

See also:

- http://www.wikiwand.com/fr/Extensible_Markup_Language

Contents

Introduction

L'Extensible Markup Language (XML[note 1], « langage de balisage extensible » en français) est un langage informatique de balisage générique qui dérive du SGML.

Cette syntaxe est dite « extensible » car elle permet de définir différents espaces de noms, c'est-à-dire des langages avec chacun leur vocabulaire et leur grammaire, comme XHTML, XSLT, RSS, SVG…

Elle est reconnaissable par son usage des chevrons (< >) encadrant les balises.

L'objectif initial est de faciliter l'échange automatisé de contenus complexes (arbres, texte riche…) entre systèmes d'informations hétérogènes (interopérabilité).

Avec ses outils et langages associés, une application XML respecte généralement certains principes :

- la structure d'un document XML est définie et validable par un schéma,
- un document XML est entièrement transformable dans un autre document XML.

1.5.3 Output formats

Dash format

See also:

- *Dash*

EPUB format

See also:

- https://fr.wikipedia.org/wiki/Epub

Introduction

EPUB (acronyme de « electronic publication » ou « publication électronique », parfois noté ePub, EPub ou epub) est un format ouvert standardisé pour les livres numériques.

Proposé par l'International Digital Publishing Forum (IDPF), ces fichiers ont l'extension .epub.

EPUB est conçu pour faciliter la mise en page du contenu, le texte affiché étant ajusté pour le type d'appareil de lecture.

Il est également conçu comme le seul format pouvant à la fois satisfaire les éditeurs pour leurs besoins internes et la distribution.

Ce format englobe le standard Open eBook.

La dernière version standardisée, EPUB3, repose sur l'HTML5, ce qui ouvre la voie à de nombreuses extensions. Elle offre de nouvelles caractéristiques telles que la prise en charge de l'affichage de toutes les langues, un espace spécifique pour les métadonnées, un développement de l'interactivité permettant l'ajout de contenus enrichis (graphismes, typographies, multimédias).

Diverses applications permettent de créer un fichier EPUB directement ou à partir d'un fichier préexistant

Hypertext_Markup_Language (HTML) format

See also:

- http://www.wikiwand.com/fr/Hypertext_Markup_Language

Contents

Introduction

L'Hypertext Markup Language, généralement abrégé HTML, est le format de données conçu pour représenter les pages web.

C'est un langage de balisage permettant d'écrire de l'hypertexte, d'où son nom.

HTML permet également de structurer sémantiquement et de mettre en forme le contenu des pages, d'inclure des ressources multimédias dont des images, des formulaires de saisie, et des programmes informatiques.

Il permet de créer des documents interopérables avec des équipements très variés de manière conforme aux exigences de l'accessibilité du web.

Il est souvent utilisé conjointement avec des langages de programmation (JavaScript) et des formats de présentation (feuilles de style en cascade).

HTML est initialement dérivé du Standard Generalized Markup Language (SGML).

Portable Document Format (PDF)

See also:

- http://www.wikiwand.com/fr/Portable_Document_Format

- pdf_format

Contents

Introduction

Le Portable Document Format, communément abrégé en PDF, est un langage de description de pages créé par la société Adobe Systems et dont la spécificité est de préserver la mise en forme d'un fichier – polices d'écritures, images, objets graphiques, etc – telle qu'elle a été définie par son auteur, et cela quels que soient le logiciel, le système d'exploitation et l'ordinateur utilisés pour l'imprimer ou le visualiser.

Limites

Si la police de caractères avec laquelle le document a été créé n'est pas disponible sur la plate-forme utilisée pour l'affichage, elle est remplacée par une police équivalente (mais pas toujours identique) ce qui peut entraîner une modification de la mise en page.

Afin d'éviter cette difficulté, une variante a été créée, le format PDF/A. Ce format intègre les polices utilisées pour la création du document, garantissant ainsi le respect de la mise en page quelle que soit la disponibilité des polices sur les plate-formes utilisées pour l'affichage ultérieur du document.

Articles connexes : PDF/X et PDF/A-1.

Windows HTML Help format (or known as CHM)

See also:

- http://www.wikiwand.com/fr/Microsoft_Compressed_HTML

Contents

Introduction

Microsoft Compressed HTML (MCH), ou Microsoft HTML Help, est un format propriétaire pour les fichiers d'aide sur internet, développé par Microsoft et publié pour la première fois en 1997 comme étant le successeur du format Microsoft WinHelp.

Introduit sur le marché pour Windows 98, il est toujours utilisé sur Windows XP et sur Windows 7 pour des logiciels tiers.

Formats de fichier

Les fichiers d'aide portent l'extension:

- CHM (compressed HTML) pour la version 1,
- et HXS pour la version 2.

Un fichier d'aide compressé est une archive contenant plusieurs fichiers HTML, un par rubrique (topic) ainsi que les images et d'autres fichiers permettant une visualisation avec l'interface d'aide.

Le système utilise un algorithme de compression LZX.

Un fichier CHM peut être décompressé en utilisant le programme hh.exe de Microsoft Windows : il suffit de taper en ligne de commande[1]:

```
hh -decompile dossier_cible nom_de_fichier.chm
```

où nom_de_fichier est le fichier CHM que l'on veut décompresser et dossier_cible est le dossier dans lequel on va le décompresser.

Création de l'aide

Le fichier d'aide est obtenu en compilant plusieurs fichiers :

- le fichier de projet, qui porte l'extension .hhp (HTML Help Project) il fait le lien avec les différents fichiers ;
- les rubriques d'aide, sous la forme de fichiers HTML (avec l'extension .htm ou .html) ; il y a un fichier par rubrique ;
- les images auxquelles il est fait référence dans les fichiers HTML, au format GIF ;
- la table des matières, qui porte l'extension .hhc (HTML Help Contents)
- l'index, qui porte l'extension .hhk.

La compilation est faite par le programme hhc.exe, qui fait partie de la suite de développement Microsoft HTML Help SDK (software development kit), ou bien par un programme de création d'aide d'un autre éditeur (comme par exemple RoboHelp d'Adobe Systems ou Doc-To-Help de ComponentOne)

L'Extensible Markup Language (XML)

See also:

- http://www.wikiwand.com/fr/Extensible_Markup_Language

Contents

Introduction

L'Extensible Markup Language (XML[note 1], « langage de balisage extensible » en français) est un langage informatique de balisage générique qui dérive du SGML.

Cette syntaxe est dite « extensible » car elle permet de définir différents espaces de noms, c'est-à-dire des langages avec chacun leur vocabulaire et leur grammaire, comme XHTML, XSLT, RSS, SVG…

Elle est reconnaissable par son usage des chevrons (< >) encadrant les balises.

L'objectif initial est de faciliter l'échange automatisé de contenus complexes (arbres, texte riche…) entre systèmes d'informations hétérogènes (interopérabilité).

Avec ses outils et langages associés, une application XML respecte généralement certains principes :

- la structure d'un document XML est définie et validable par un schéma,
- un document XML est entièrement transformable dans un autre document XML.

1.6 Documentation projects

1.6.1 C Documentation projects

Contents

clang (doxygen)

See also:

http://clang.llvm.org/doxygen/

1.6.2 Mozilla documentation

See also:

- https://developer.mozilla.org/fr/
- https://twitter.com/mozdevnet

1.6.3 Documenting python projects

pep0257

See also:

http://www.python.org/dev/peps/pep-0257/

Contents

Rationale

The aim of this PEP is to standardize the high-level structure of docstrings: what they should contain, and how to say it (without touching on any markup syntax within docstrings).

The PEP contains conventions, not laws or syntax.

```
"A universal convention supplies all of maintainability, clarity, consistency,
and a foundation for good programming habits too. What it doesn't do is
insist that you follow it against your will. That's Python !"

--Tim Peters on comp.lang.python, 2001-06-16
```

If you violate these conventions, the worst you'll get is some dirty looks.

But some software (such as the Docutils docstring processing system pep0256) will be aware of the conventions, so following them will get you the best results.

Documenting python projects with sphinx

See also:

- *Sphinx*
- http://sphinx-doc.org/latest
- http://packages.python.org/an_example_pypi_project/sphinx.html
- http://docs.python.org/dev/documenting/

sphinx

See also:

- http://sphinx-doc.org/latest

an example pypi project

See also:

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

http://docs.python.org/dev/documenting

See also:

http://docs.python.org/dev/documenting/

1.7 Documentation tools

1.7.1 Dash

See also:

- http://kapeli.com/dash

Contents

Description

`Dash` is an API Documentation Browser and Code Snippet Manager.

Dash stores snippets of code and instantly searches offline documentation sets for 150+ APIs (for a full list, see below).

You can even generate your own docsets or request docsets to be included.

Dash user contributions

See also:

- https://github.com/Kapeli/Dash-User-Contributions

Dash in doxygen

See also:

- *Added support for processing DocSets*

Sphinx dash builder

See also:

- *sphinxcontrib-dashbuilder*

Zeal

See also:

- *Zeal documentation browser*

Documentation Browser

- 150+ offline docsets
- Instant, fuzzy search
- Great integration with other apps
- Easily download docsets
- Easily generate docsets:
  - Supports AppleDoc docsets
  - Supports Doxygen docsets
  - Supports CocoaDocs docsets
  - Supports Python / Sphinx docsets
  - Supports Ruby / RDoc docsets
  - Supports Javadoc docsets
  - Supports Scaladoc docsets
  - Supports Any HTML docse

1.7.2 Graphviz

See also:

- http://www.graphviz.org/

Contents

What is Graphviz ?

`Graphviz` is open source graph visualization software.

Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks.

It has important applications in networking, bioinformatics, software engineering, database and web design, machine learning, and in visual interfaces for other technical domains.

Used by

- *Doxygen*

1.7.3 Zeal documentation browser

See also:

- https://github.com/jkozera/zeal
- http://zealdocs.org/

Description

`Zeal` is a simple documentation browser inspired by *Dash*.

Sphinx dash builder

See also:

- *sphinxcontrib-dashbuilder*

1.8 Documentation videos

Contents

Contents

- *Documentation videos*
  - *Write the docs*
  - *RTFM - wRite The Friendly Manual*

1.8.1 Write the docs

See also:

- http://pyvideo.org/video/1795/write-the-docs

- http://pyvideo.org/speaker/25/james-bennett

The greatest piece of software in the world is useless without great documentation, but unfortunately most of us just don't write great docs.

This can be fixed, though.

Documentation doesn't need to be an afterthought, and doesn't have to be bad, and you, too, can learn how to write good docs and make that an integrated part of your development process.

1.8.2 RTFM - wRite The Friendly Manual

See also:

http://pyvideo.org/video/79/djangocon-2011–rtfm—-write-the-friendly-manual

Presented by James Bennett

An introduction to writing great documentation. Not just in the "here's some tools and how to use them" sense, but in the "here's why you should care about documentation" sense and the "how to write things people will read" sense.

1.9 Wiki documentation

1.9.1 Mediawiki

See also:

- https://fr.wikipedia.org/wiki/Mediawiki
- https://twitter.com/mediawiki
- http://identi.ca/mediawiki
- wikimedia_foundation

Contents

Introduction

MediaWiki est un moteur de wiki pour le Web.

Il est utilisé par l'ensemble des projets de la Wikimedia Foundation, ainsi que par l'ensemble des wikis hébergés chez Wikia et par de nombreux autres wikis.

Conçu pour répondre aux besoins de Wikipédia, ce moteur est en 2008 également utilisé par des entreprises comme solution de gestion des connaissances et comme système de gestion de contenu.

L'entreprise américaine Novell l'utilise notamment pour plusieurs de ses sites web qui véhiculent un trafic important.

Des associations, comme Wikitravel, Mozilla ou Ékopedia, l'ont aussi adopté.

MediaWiki est écrit en PHP, et peut aussi bien fonctionner avec le système de gestion de base de données MySQL que PostgreSQL.

C'est un logiciel libre distribué selon les termes de la GPL.

MediaWiki inclut de nombreuses fonctionnalités pour les sites à vocation collaborative : par exemple, la gestion des espaces de noms, ou encore l'utilisation de pages de discussions associées à chaque article.

The Wikimedia Foundation

See also:

- wikimedia_foundation

Historique

Initialement, Wikipédia utilisait un moteur de wiki rudimentaire écrit en Perl, appelé UseModWiki.

Le 25 janvier 2002, MediaWiki, développé par Magnus Manske, un étudiant allemand de l'université de Cologne, devient le moteur de wiki de l'encyclopédie collaborative pour laquelle il a été développé. MediaWiki a ainsi permis de disposer de plus de fonctionnalités et d'une infrastructure plus extensible (grâce à une base de données MySQL).

Les performances du logiciel ont ensuite été améliorées par Lee Daniel Crocker, avant que Brion Vibber n'en devienne le développeur le plus actif et ne prenne le rôle de dirigeant des sorties logicielles.

Depuis la sortie de la première version du script de Manske, plusieurs noms représentatifs de l'état du logiciel lui ont été donnés : « le script PHP », « phase II », « phase III », « le nouveau code source ».

Cependant il n'était pourvu d'aucun nom de produit. Après l'annonce de la création de la Wikimedia Foundation le 20 juin 2003, le wikipédien Daniel Mayer lui donne le nom « MediaWiki » par jeu de mots sur le nom « Wikimedia » et ce nom est progressivement adopté. Toutefois, la similarité des noms MediaWiki et Wikimedia (qui lui-même est déjà semblable au nom Wikipédia) est à l'origine de fréquentes confusions.

Le logo de MediaWiki a été créé par Erik Moeller à partir d'une photographie d'une fleur prise par Florence Devouard (qui sera par la suite présidente de la Wikimedia Foundation), et fut initialement soumis au concours international du nouveau logo pour Wikipédia qui s'est déroulé pendant l'été 2003.

Le logo s'est placé en troisième position à l'issue de ce concours, et a été choisi pour représenter MediaWiki plutôt que Wikipédia, tandis que le logo vainqueur a été adopté pour représenter Wikipédia, et le second pour la Wikimedia Foundation.

Les doubles crochets sur la photo autour du tournesol symbolisent le wikicode, c'est-à-dire la syntaxe utilisée par MediaWiki pour créer des hyperliens vers les autres pages du wiki.

API mediawiki

Mediawiki API

Mediawiki API tutorial

See also:

- https://www.mediawiki.org/wiki/API/Tutorial

Tutorial for MediaWiki's RESTful web service API

Why should you use the web API? Bots, AJAX, Gadgets, other things.

Roan says: generally any Ajax feature is going to use the api.php entry point.

But right now the easiest thing to do is to write a bot or to use the API clients.

Definitions

REST API for MediaWiki exposes things MediaWiki has in the database or otherwise understands does not include semantic stuff like "definition of a word in Wiktionary" or even "lead paragraph of an article"

Usage

send HTTP requests (GET or POST) to the api.php URL, receive XML or JSON or other formats.

You'll usually want JSON or XML.

w:en:JSON and w:en:XML and w:en:Representational state transfer (RESTful)

There are other things that also get casually called the MediaWiki API, like the internal interfaces that extensions and special pages can hook into.

We're not talking about that right now, just the web API. (possibly talk about how it works from the back end, if people ask)...

Developer hub mediawiki

Mediawiki developer hub

See also:

- https://www.mediawiki.org/wiki/Developer_hub
- https://meta.wikimedia.org/wiki/Wikimedia_developer_hub

- https://meta.wikimedia.org/wiki/Tech
- https://www.mediawiki.org/wiki/MDG
- https://www.mediawiki.org/wiki/How_to_become_a_MediaWiki_hacker
- http://www.aosabook.org/en/mediawiki.html

Contents

Introduction

A place for programmers who are looking for an opportunity to help with wikimedia software projects.

Developer hub

See also:

- https://www.mediawiki.org/wiki/Developer_hub

This is a high-level overview of MediaWiki development, including links to the key documents, resources and tools available to MediaWiki developers.

It is written for skilled LAMP developers who have experience using MediaWiki.

Manual:MediaWiki Developer's Guide

See also:

- https://www.mediawiki.org/wiki/MDG
- https://www.mediawiki.org/wiki/MVL

You can load it from here, generate a PDF, let a book printed, edit it or otherwise update its content. Be reminded that when overwriting this book storage page with an updated version of the book, the previously assigned categories are not automatically copied to the recent version: you need to manually copy the categories from the former version

How to become a MediaWiki hacker

See also:

- https://www.mediawiki.org/wiki/How_to_become_a_MediaWiki_hacker
- http://www.aosabook.org/en/mediawiki.html

MediaWiki is the software that powers Wikipedia, its sister projects and thousands of wikis all over the world.

It runs on most operating systems, is written in PHP, primarily uses the MySQL database server and uses jQuery as the client Javascript library.

Development of MediaWiki is primarily supported by the Wikimedia Foundation, though volunteer community developers play a huge part as well.

This page should help you get started on the path to becoming a contributor to MediaWiki. It is not a tutorial; it just points you to various places where you can go learn whatever is necessary.

Mediawiki developer news

Mediawiki developer news

Mediawiki developer 2013

Mediawiki developer Avril 2013

Mediawiki developer 13 Avril 2013

Recycling wikitech-announce for tech contributors

```
Sujet:  [Wikitech-l] Recycling wikitech-announce for tech contributors
Date :  Sat, 13 Apr 2013 12:23:21 -0700
De :    Quim Gil <qgil@wikimedia.org>
Répondre à :    Wikimedia developers <wikitech-l@lists.wikimedia.org>
Organisation :  Wikimedia Foundation
Pour :  Wikimedia developers <wikitech-l@lists.wikimedia.org>
```

Hi,

We were missing a way to notify tech contributors and volunteers about new activities, and after some discussion we have decided to recycle the unused

https://lists.wikimedia.org/mailman/listinfo/wikitech-announce

Subscribers will receive CALLS FOR ACTION ONLY e.g. for activities like the ones listed at https://www.mediawiki.org/wiki/Project:Calendar. No announcements of new releases, features removed, etc.

We have channels already for that.

We will sync the announcements at wikitech-announce with with wikitech-l and wikitech-ambassadors.

While this is not a big deal for current contributors following already wikitech-l and a number of wiki pages etc, it will help potential volunteers willing to get involved and know about opportunities to contribute.

PS: before clicking reply please read http://www.gossamer-threads.com/lists/wiki/wikitech/282349 :)

```
--
Quim Gil
Technical Contributor Coordinator @ Wikimedia Foundation
http://www.mediawiki.org/wiki/User:Qgil
```

Extensions mediawiki

Mediawiki extensions

See also:

- http://fr.wikipedia.org/wiki/Wikimedia

TimedMediaHandler

See also:

- http://www.mediawiki.org/wiki/TimedMediaHandler

The TimedMediaHandler extension is a MediaWiki extension to display audio and video files on wiki with timed text support, real time stream switching and server-side transcoding support. It includes the kaltura HTML5 player.

This project is realized in collaboration with Kaltura and led by Michael Dale, along other Media Projects.

International mediawiki

The Mediawiki groups in the world

See also:

- https://www.mediawiki.org/wiki/Groups
- Groups

Introduction

MediaWiki groups organize open source community activities within the scope of specific topics and geographical areas.

They are Wikimedia User Groups that agree on a level of coordination in the MediaWiki context.

As such, they extend the capacity of the Wikimedia Foundation in events, training, promotion and other technical activities benefiting Wikipedia, the Wikimedia movement and the MediaWiki software.

MediaWiki Groups are open to members of different specialties and levels of expertise.

The richer and more diverse the better.

Non-technical users willing to contribute and learn are welcome too!

All groups commit to the Friendly space policy.

MediaWiki groups can request support from the Wikimedia Foundation and chapters in various forms.

Groupes

Mediawiki Amerique

See also:

- http://fr.wikipedia.org/wiki/Mediawiki_Foundation

Mediawiki Amerique

Mediawiki Asie

Mediawiki Israël

Contents

Mediawiki Europe

Mediawiki France

Projets utilisant mediawiki

Projects using Mediawiki

See also:

- http://fr.wikipedia.org/wiki/Wikimedia

MediaWiki est un moteur de wiki pour le Web. Il est utilisé par l'ensemble des projets de la Wikimedia Foundation, ainsi que par l'ensemble des wikis hébergés chez Wikia et par de nombreux autres wikis.

Conçu pour répondre aux besoins de Wikipédia, ce moteur est en 2008 également utilisé par des entreprises comme solution de gestion des connaissances et comme système de gestion de contenu.

L'entreprise américaine Novell l'utilise notamment pour plusieurs de ses sites web qui véhiculent un trafic important.

Des associations, comme Wikitravel, Mozilla ou Ékopedia, l'ont aussi adopté.

MediaWiki est écrit en PHP, et peut aussi bien fonctionner avec le système de gestion de base de données MySQL que PostgreSQL.

C'est un logiciel libre distribué selon les termes de la GPL.

MediaWiki inclut de nombreuses fonctionnalités pour les sites à vocation collaborative : par exemple, la gestion des espaces de noms, ou encore l'utilisation de pages de discussions associées à chaque article.

Mediawiki LOGRE

See also:

- https://www.logre.eu/wiki/Accueil
- fablab_logre

Fablab logre

See also:

- fablab_logre

Mediawiki Parti Pirate

See also:

- http://wiki.partipirate.org/wiki/Accueil

Wikimedia

See also:

- http://fr.wikipedia.org/wiki/Wikimedia

Ecrire pour Wikimedia

See also:

- http://fr.wikipedia.org/wiki/Wikimedia

Ecrire un article

Contents

Comment créer un article

See also:

- http://en.wikipedia.org/wiki/Wikipedia:Starting_an_article
- http://fr.wikipedia.org/wiki/Aide:Comment_cr%C3%A9er_un_article

Cette page « Comment créer un article » est conçue pour vous aider à créer un article, en évitant les pièges les plus courants et les erreurs possibles.

Wikipédia:N'hésitez pas !

See also:

- http://fr.wikipedia.org/wiki/Wikip%C3%A9dia:N%27h%C3%A9sitez_pas_!

Règle officielle

Cette page expose une règle de la Wikipédia en français, une norme largement acceptée par les wikipédiens qui doit normalement être suivie par tous les rédacteurs.

Toutes les modifications de cette page doivent refléter le consensus.

Dans le doute, exposez au préalable les modifications en page de discussion.

Wikipédia:Interprétation créative des règles

See also:

http://fr.wikipedia.org/wiki/Wikip%C3%A9dia:Interpr%C3%A9tation_cr%C3%A9ative_des_r%C3%A8gles

Syntaxe Wikipedia

See also:

- http://fr.wikipedia.org/wiki/Aide:Syntaxe

Contents

Guide de la syntaxe Wiki

See also:

http://upload.wikimedia.org/wikipedia/commons/1/12/Guide_de_la_syntaxe_Wiki.pdf

Wikinews

See also:

http://fr.wikinews.org/wiki/Aide:Syntaxe_Wikinews

Wikisource

See also:

http://fr.wikisource.org/wiki/Aide:La_syntaxe_Wiki

Wikimedia projects

See also:

- http://fr.wikipedia.org/wiki/Wikimedia

Wikipedia project doc

See also:

- http://fr.wikipedia.org

Outils mediawiki

Mediawiki tools

See also:

- http://fr.wikipedia.org/wiki/Mediawiki

Git mediawiki

See also:

- https://github.com/moy/Git-Mediawiki/wiki
- http://git.kernel.org/?p=git/git.git;a=tree;f=contrib/mw-to-git

Contents

Introduction

Git-Mediawiki is a project which aim is to create a gate between git and mediawiki, allowing git users to push and pull objects from mediawiki just as one would do with a classic git repository.

For more information, read the User manual

Git mediawiki source code

See also:

- http://git.kernel.org/?p=git/git.git;a=tree;f=contrib/mw-to-git

The latest version of Git-MediaWiki is available in Git's source tree, in the directory contrib/mw-to-git.

You can download it from http://git.kernel.org/?p=git/git.git;a=tree;f=contrib/mw-to-git if needed.

You need to have the script git-remote-mediawiki in your PATH (and it needs to be executable) to use it.

Alternatively, you may install it in Git's exec path (run git –exec-path to find out where it is).

Visual Editor

See also:

- https://www.mediawiki.org/wiki/VisualEditor

Contents

Introduction

The VisualEditor project aims to create a reliable rich-text editor for MediaWiki.

It is being developed so it can be used as a MediaWiki extension, using the Parsoid project to supply HTML+RDFa.

It is currently deployed to a test namespace of this wiki; more information about this test deployment can be found on Wikimedia's blog, the FAQs, and VisualEditor:Welcome or VisualEditor:Test. Please note that the test deployment only works with the Vector skin.

Annonces

Linux-fr

See also:

- http://linuxfr.org/news/lancement-de-l-editeur-visuel-de-mediawiki

Sûrement motivée par la nouvelle interface de linuxfr.org et afin d'attirer un nombre plus grand de contributeurs au projet Wikipedia, la fondation Wikimedia a développé le logiciel open-source VisualEditor.

Il s'agit de proposer une interface WYSIWYG (ce que vous voyez est ce que vous obtenez) au moteur Wiki de la fondation afin de ne pas obliger les contributeurs potentiels à apprendre la syntaxe wikitexte.

La première version utilisable avait été présentée en juin dernier, il s'agissait plus d'un démonstrateur aux fonctionnalités très limitées que d'une version alpha.

Ce 12 décembre, l'équipe a annoncé sur le blog wikimedia le lancement de la version alpha du projet.

Cette version n'est disponible que depuis la version anglaise de Wikipedia et, ce, pour les utilisateurs enregistrés qui auront activés l'option dans leurs préférences (option désactivée par défaut). Il s'agit d'une version de test, le but étant donc de faire des retours à l'équipe.

Une fois l'option activée, pour chaque article, il est possible d'éditer en passant par un onglet un second éditeur étiqueté « VisualEditor » à côté de l'onglet « Modifier ».

En cliquant sur cet onglet, après une petite pause vous entrerez dans le VisualEditor.

De là, vous pouvez jouer, éditer et enregistrer des articles réels et avoir une idée de ce que sera l'article lorsque vous avez terminé.

À ce stade précoce du développement, il est recommandé de réaliser des vérifications après avoir sauvegardé pour s'assurer que rien n'a été cassé.

Toutes les modifications faites par VisualEditor seront affichées dans les onglets Histoire des articles avec un tag VisualEditor à côté d'eux, afin qu'il soit possible de suivre ce qui se passe.

Cette version reste limitée en fonctionnalités tel qu'indiqué dans la page VisualEditor et l'équipe rencontre des difficultés avec le langage wikitexte qui n'a cessé d'évoluer depuis le lancement de Wikipedia.

Ceci a nécessité le développement de Parsoid afin de convertir les anciens fichiers dans un format qui convient à VisualEditor pour être ensuite reconverti en wikitexte. Les tests réalisés ont été concluant dans 80% des cas et 18% des cas ont été légèrement différents. Les développeurs disent qu'ils ont l'intention d'améliorer significativement ces pourcentages, et la vitesse de Parsoid, au cours des prochains mois.

Selon le calendrier actuel, VisualEditor est prévu pour être l'éditeur par défaut de presque tous les projets Wikimedia à partir de juillet 2013.

1.9.2 Wiki tools

Django wiki

See also:

- https://github.com/benjaoming/django-wiki
- http://wiki.overtag.dk/

Contents

Demo

A demo is available here, sign up for an account to see the notification system.

wiki.overtag.dk

Community

Please use our mailing list (google group) for getting in touch on development and support:

[django-wiki@googlegroups.com](https://groups.google.com/d/forum/django-wiki)

[twitter:djangowiki](https://twitter.com/djangowiki)

THIS IS A WORK IN PROGRE...

Currently, the API is subject to smaller changes. South is used so no database changes will cause data loss. You are not encouraged to make your own fiddling with the internal parts of the wiki - the best idea is to customize it through overriding templates and making custom template tags. The second best strategy is to extend the wiki's class-based views.

Please refer to the [TODO](https://github.com/benjaoming/django-wiki/blob/master/TODO.md) for a detailed status or the Issue list.

Manifesto

Django needs a mature wiki system appealing to all kinds of needs, both big and small:

- **Be pluggable and light-weight.** Don't integrate optional features in the core.
- **Be open.** Make an extension API that allows the ecology of the wiki to grow. After all, Wikipedia consists of some [680 extensions](http://svn.wikimedia.org/viewvc/mediawiki/trunk/extensions/) written for MediaWiki.
- **Be smart.** [This is](https://upload.wikimedia.org/wikipedia/commons/8/88/MediaWiki_database_schema_1-19_%28r102798%29.png) the map of tables in MediaWiki - we'll understand the choices of other wiki projects and make our own. After-all, this is a Django project.
- **Be simple.** The source code should *almost* explain itself.
- **Be structured.** Markdown is a simple syntax for readability. Features should be implemented either through easy coding patterns in the content field, but rather stored in a structured way (in the database) and managed through a friendly interface. This gives control back to the website developer, and makes knowledge more usable. Just ask: Why has Wikipedia never changed? Answer: Because it's knowledge is stored in a complicated way, thus it becomes very static.

Ideas?

Please go ahead and post issues for discussion of ideas.

Installation

Install

To install the latest stable release:

pip install wiki

Install directly from Github, since there is no release yet:

pip install git+git://github.com/benjaoming/django-wiki.git

Configure *settings.INSTALLED_APPS*

The following applications should be listed - NB! it's important to maintain the order due to database relational constraints:

> 'django.contrib.humanize', 'south', 'django_notify', 'mptt', 'sekizai', 'sorl.thumbnail', 'wiki', 'wiki.plugins.attachments', 'wiki.plugins.notifications', 'wiki.plugins.images',

Database

To sync and create tables, do:

> python manage.py syncdb python manage.py migrate

Configure TEMPLATE_CONTEXT_PROCESSORS

Add *'sekizai.context_processors.sekizai'* to *settings.TEMPLATE_CONTEXT_PROCESSORS*. Please refer to the [Django docs](https://docs.djangoproject.com/en/dev/ref/settings/#template-context-processors) to see the current default setting for this variable.

Include urlpatterns

To integrate the wiki to your existing application, you shoud add the following lines at the end of your project's *urls.py*:

```
from wiki.urls import get_pattern as get_wiki_pattern
from django_notify.urls import get_pattern as get_notify_pattern
urlpatterns += patterns('',
    (r'^notify/', get_notify_pattern()),
    (r'', get_wiki_pattern())
)
```

Please use these function calls rather than writing your own include() call - the url namespaces aren't supposed to be customized.

The above line puts the wiki in */* so it's important to put it at the end of your urlconf. You can also put it in */wiki* by putting *'^wiki/'* as the pattern.

Settings

For now, look in [wiki/conf/settings.py](https://github.com/benjaoming/django-wiki/blob/master/wiki/conf/settings.py) to see a list of available settings.

Other tips

1. **Account handling:** There are simple views that handle login, logout and signup. They are on by default. Make sure to set settings.LOGIN_URL to point to your login page as many wiki views may redirect to a login page.

Plugins

Add/remove the following to your *settings.INSTALLED_APPS* to enable/disable the core plugins:

- *'wiki.plugins.attachments'*
- *'wiki.plugins.images'*
- *'wiki.plugins.notifications'*

The notifications plugin is mandatory for an out-of-the-box installation. You can safely remove it from INSTALLED_APPS if you also override the **wiki/base.html** template.

Background

Django-wiki is a rewrite of [django-simplewiki](http://code.google.com/p/django-simple-wiki/), a project from 2009 that aimed to be a base system for a wiki. It proposed that the user should customize the wiki by overwriting templates, but soon learned that the only customization that really took place was that people forked the entire project. We don't want that for django-wiki, we want it to be modular and extendable.

As of now, Django has existed for too long without a proper wiki application. The dream of django-wiki is to become a contestant alongside Mediawiki, so that Django developers can stick to the Django platform even when facing tough challenges such as implementing a wiki.

Contributing

This project will be very open for enrolling anyone with a good idea. As of now, however, it's a bit closed while we get the foundation laid out.

Q&A

- **Why is the module named just "wiki"?** Because "pip install wiki" returns "No distributions at all found for wiki"! :)
- **What markup language will you use?** [Markdown](http://pypi.python.org/pypi/Markdown). The markup renderer is not a pluggable part but has been internalized into core parts. Discussion should go here: https://github.com/benjaoming/django-wiki/issues/76
- **Why not use django-reversion?** It's a great project, but if the wiki has to grow ambitious, someone will have to optimize its behavior, and using a third-party application for something as crucial as the revision system is a no-go in this regard.
- **Any support for multiple wikis?** Yes, in an sense you can just imagine that you always have multiple wikis, because you always have hierarchies and full control of their permissions. See this discussion: https://github.com/benjaoming/django-wiki/issues/63

Dependencies

So far the dependencies are:

- [django>=1.4](http://www.djangoproject.com)
- [django-south](http://south.aeracode.org/)
- [Markdown>=2.2.0](https://github.com/waylan/Python-Markdown)
- [django-mptt>=0.5.3](https://github.com/django-mptt/django-mptt)
- [django-sekizai](https://github.com/ojii/django-sekizai/)
- [sorl-thumbnail](https://github.com/sorl/sorl-thumbnail)
- PIL (Python Imaging Library)
- Python>=2.5<3 (Python 3 not yet supported)

Development

In a your Git fork, run *pip install -r requirements.txt* to install the requirements.

The folder **testproject/** contains a pre-configured django project and an sqlite database. Login for django admin is *admin:admin*. This project should always be maintained, although the sqlite database will be deleted very soon to avoid unnecessary conflicts.

[![Build Status](https://travis-ci.org/benjaoming/django-wiki.png?branch=master){}](https://travis-ci.org/benjaoming/django-wiki)

Python 2.5

Due to Markdown using elementree, you should check that you have python-celementtree: *apt-get install python-celementtree*

Acknowledgements

- The people at [edX](http://www.edxonline.org/) & MIT for finding and supporting the project both financially and with ideas.
- [django-cms](https://github.com/divio/django-cms) for venturing where no django app has gone before in terms of well-planned features and high standards. It's a very big inspiration.
- [django-mptt](https://github.com/django-mptt/django-mptt), a wonderful utility for inexpensively using tree structures in Django with a relational database backend.

CHAPTER 2

Command Line Interface (CLI)

See also:

- http://www.commandlinefu.com
- https://secure.wikimedia.org/wikipedia/en/wiki/Command_line_interpreter
- http://google-opensource.blogspot.com/2010/06/introducing-google-command-line-tool.html
- python_cli_management

Contents

2.1 Introduction

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

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

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

2.2 Usage

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

2.2.1 CLI and python

See also:

- python_cli_management

2.2.2 CLI with Perl

See also:

- http://www.activestate.com/activeperl/
- perl_language

Contents

perl usage

windows .bat file

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

UNIX .sh file

```
#!/usr/bin/sh

perl -pi -e 's/\/_ext\/_DOTDOT\/_DOTDOT//g' $1
perl -pi -e 's/\/\${CND_PLATFORM}//g' $1
```

Perl modules

ack

See also:

- http://beyondgrep.com/
- https://github.com/petdance/ack
- http://petdance.com/
- http://beyondgrep.com/ack-2.0/
- http://beyondgrep.com/why-ack/

Contents

Presentation

`ack` is a grep-like tool optimized for working with large trees of source code.

ack is a grep-like search tool that has been optimized for searching large heterogeneous trees of source code.

ack has been around since 2005.

Since then it has become very popular and is packaged by all the major Linux distributions.

It is cross-platform and pure Perl, so will run on Windows easily.

See the “Why ack ?”_ page for the top ten reasons, and dozens of testimonials.

Documentation

See also:

- http://beyondgrep.com/documentation/

Install

On windows

See also:

- http://chocolatey.org/packages/ack
- http://stevengharms.com/blog/2012/04/10/use-ack-instead-of-grep-to-parse-text-files/

Ack more tools

ack more tools

See also:

- http://beyondgrep.com/more-tools/

Contents

Tools that work with ack

vim integration

See also:

- http://www.vim.org/scripts/script.php?script_id=2572

ack.vim provides an interface between ack and the vim text editor.

For example, you can call :Ack foo, which will run ack and load ack's results into a vim buffer for manipulation and navigation.

ack.vim is available at the official vim website at http://www.vim.org/scripts/script.php?script_id=2572

Other grep-like tools

There are many ways to search source code that are more flexible and tuned to programmers than straight grep.

I suggest you take a look at some of these alternatives, for they may suit your needs better than ack.

If you have any suggestions to add to this list, please let me know at andy@petdance.com.

Ag, the Silver Searcher

Geoff Greer says "Ag is like ack, but better. It's fast. It's damn fast.

The only thing faster is stuff that builds indices beforehand, like Exuberant Ctags."

Geoff has also created a fork of AckMate that uses Ag instead of ack.

https://github.com/ggreer/the_silver_searcher

grin

See also:

- grin

"A grep program configured the way I like it", written in Python by Robert Kern.

nak

See also:

- https://github.com/gjtorikian/nak

An implementation of ack, written in Node.js.

It has inspiration from Ag, and is optimized for speed, not features.

It's completely asynchronous. Written by Garen J. Torikian.

pss

See also:

- pss

pss is an ack clone written in Python by Eli Bendersky.

It's written in pure Python with no additional modules necessary.

Indexing tools

Sometimes when you're looking at a large codebase, it makes sense to see everything as a whole.

An indexing tool may help you out.

ctags

ctags is a program almost as old as time itself.

When run against a codebase, ctags indexes various elements of the code, such as variables and functions. This lets your editor or other tools use the tags index to jump quickly to that element.

The most common ctags implementation is Exuberant ctags: http://ctags.sourceforge.net/

cscope

See also:

- http://cscope.sourceforge.net/

Cscope is a developer's tool for browsing source code.

Cscope was part of the official AT&T Unix distribution for many years, and has been used to manage projects involving 20 million lines of code.

It also can integrate with vim and Emacs.

CodeQuery

See also:

- https://github.com/ruben2020/codequery

CodeQuery indexes and queries C, C++, Java and Python source code.

It builds upon the databases of cscope and ctags, mentioned above, and provides a nice GUI tool.

Ack versions

ack versions

ack 2.0.0

See also:

- http://petdance.com/2013/04/ack-2-0-has-been-released/
- http://beyondgrep.com/ack-2.0/

Presentation

ack 2.0 has been released.

ack 2.0 has many changes from 1.x, but here are four big differences and features that long-time ack 1.x users should be aware of.

- By default all text files are searched, not just files with types that ack recognizes. If you prefer the old ack 1.x behavior of only searching files that ack recognizes, you can use the -k/–known-types option.
- There is a much more flexible type identification system available. You can specify a file type based on extension (.rb for Ruby), filename (Rakefile is a Ruby file), first line matching a regex (Matching /#!.+ruby/ is a Ruby file) or regex match on the filename itself.
- Greater support for ackrc files. You can have a system-wide ackrc at /etc/ackrc, a user-specific ackrc in ~/.ackrc, and ackrc files local to your projects.
- The -x argument tells ack to read the list of files to search from stdin, much like xargs. This lets you do things like git ls | ack -x foo and ack will search every file in the git repository, and only those files that appear in the repository.

On the horizon, we see creating a framework that will let authors create ack plugins in Perl to allow flexibility.

You might create a plugin that allows:

- searching through zip files,
- or reading text from an Excel spreadsheet, or a web page.

ack has always thrived on numerous contributions from the ack community, but I especially want to single out Rob Hoelz for his work over the past year or two.

If it were not for Rob, ack 2.0 might never have seen the light of day, and for that I am grateful.

A final note: In the past, ack's home page was betterthangrep.com.

With the release of ack 2.0, I've changed to beyondgrep.com.

"Beyond" feels less adversarial than "better than", and implies moving forward as well as upward.

beyondgrep.com also includes a page of other tools that go beyond the capabilities of grep when searching source code.

For long time ack users, I hope you enjoy ack 2.0 and that it makes your programming life easier and more enjoyable.

If you've never used ack, give it a try.

2.2.3 Command Line Interface (CLI) on GNU/linux

See also:

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

Unix shells

See also:

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

Bash: The shell for the GNU operating system

Contents

Introduction

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

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

See also:

- http://en.wikipedia.org/wiki/Bash
- http://en.wikipedia.org/wiki/Command-line_interface (CLI)

Commands Line Interface

See also:

http://www.commandlinefu.com

Research of text inside files

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

Renaming files

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

Unix shell commands

See also:

- *Command Line Interface (CLI)*

Comptage de fichiers

See also:

Command Line Interface (CLI)

Contents

Compter le nombre de fichiers include dans un projet

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

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

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

Comptage de lignes

See also:

Command Line Interface (CLI)

Contents

Compter le nombre de lignes '.h' dans un projet

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

Compter le nombre de lignes 'C/C++' dans un projet

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

Compter le nombre de lignes python dans un projet

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

Crop image files

Decouper dans une image avec imagemagick

See also:

- http://forum.ubuntu-fr.org/viewtopic.php?id=236914

```
convert image_1.png -crop 174x131+0+12 image_1_crop.png
```

Delete svn files

Contents

With the find command

```
find . -depth -name .svn -type d -exec rm -fr {} \;
```

Delete some data in a file

source guilde@guilde.asso.fr

:: Je cherche un moyen de supprimer un bloc de manière automatique d'un fichier de conf de proftpd.

Exemple de conf : Plein de lignes à garder <IfUser BLa> <Limit LOGIN> Allow 1.2.3.4 5.6.7.8... DenyAll </Limit> </IfUser> Plein de lignes à garder :

Je pense le faire avec sed mais je ne m'en sors pas. Je connais le début du bloc (<IfUser BLa>) et la fin du bloc (le premier </IfUser> après le début du bloc). [...]

Réponse d'Edgar

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

find strings in files recursively

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

Renaming files

rename command

See also:

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

Rename files from .c to .cpp

```
rename 's/\.c/\.cpp/' *.c
```

Rename files from .MOD to .MPG

```
rename 's/\.MOD/\.MPG/' *.MOD
```

With the find command + hg rename

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

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

replace string in files with rpl

recursive replace string in files with rpl

See also:

- http://www.commandlinefu.com/commands/matching/replace-string-in-files-recursively/cmVwbGFjZSBzdHJpbmcgaW4gZmlsZXMgcmVjdXJzaXZlbHk=/sort-by-votes

Contents

Dry run ('s' option)

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

Replace (without 's' option)

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

no recursive replacing with grep and sed

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

recursive replacing with find grep and sed

```
find . -name "*.rst" -print0 | xargs -0 sed -i 's/1.1.1/1.2.0/g'
```

bar

This is a small shell script intended to be used in portable Unix install scripts for showing progress bars.

The overall goal is to write a minimally complex shell script (thus a program that needs no compilation) that is as robust as possible to work on as many Bourne shells and operating systems as possible, and that implements 'cat' with an ASCII progress bar and some other nifty features.

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

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

- http://www.theiling.de/projects/bar.html

Gestion des groupes sous GNU/Linux

Ajouter un utilisateur à un groupe sous GNU/Linux

See also:

http://blog.nicolargo.com/2011/10/ajouter-un-utilisateur-a-un-groupe-sous-gnulinux.html

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

Modification du groupe primaire d'un utilisateur

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

```
usermod -g admin nicolargo
```

Ajout d'un groupe secondaire à un utilisateur existant

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

```
usermod -a -g networkadmin nicolargo
```

Ajout d'un nouvel utilisateur à un groupe primaire

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

useradd -g admin nicolargo

Ajout d'un nouvel utilisateur à un groupe secondaire

Pour ajouter le nouvel utilisateur ritchy et lui configurer un comme groupe secondaire networkadmin, il suffit d'utiliser la commande useradd:

```
useradd -G networkadmin nicolargo
```

A noter qu'il est possible d'utiliser l'option -G avec plusieurs groupes.

Exemples pour ajouter ritchy au groupe secondaire networkadmin et systemadmin:

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

Vérifier les groupes associés à un utilisateur

Rien de plus simple avec la commande groups:

```
groups ritchy
```

```
ritchy: networkadmin systemadmin
```

Resizing image files

See also:

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

mogrify command

See also:

imagemagick_tools

Resize all JPEGs in a directory

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

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

google picasa use case

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

Extract sound from a vidéo with ffmpeg

See also:

- ffmpeg_video_library

Source: Linux Partique numéro 63, p.23

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

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

Dans ce cas, la commande suivante, utilisant ffmpeg vous sera très utile.

```
ffmpeg -i input.flv -ar 44100 -ab 192k -ac 2 output.mp3
```

L'option `-i` permet de définir le fichier vidéo en entrée.

L'option `-ar` définit la fréquence d'échantillonnage (par défaut 44100Hz)

L'option `-ac` permet de définir le nombre de canaux audio

L'option `_ab` désigne le bitrate audio (64k par défaut)

write in file with python

See also:

http://hg.piranha.org.ua/sphinxedhg/docs/test-hgrc.html

Write in a file with python

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

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

mosh : the mobile shell

See also:

- http://en.wikipedia.org/wiki/Bash
- http://mosh.mit.edu
- https://github.com/keithw/mosh

Contents

Introduction

Mosh: the mobile shell

Mosh is a remote terminal application that supports intermittent connectivity, allows roaming, and provides speculative local echo and line editing of user keystrokes.

It aims to support the typical interactive uses of SSH, plus:

- Mosh keeps the session alive if the client goes to sleep and wakes up later, or temporarily loses its Internet connection.
- Mosh allows the client and server to "roam" and change IP addresses, while keeping the connection alive. Unlike SSH, Mosh can be used while switching between Wi-Fi networks or from Wi-Fi to cellular data to wired Ethernet.
- The Mosh client runs a predictive model of the server's behavior in the background and tries to guess intelligently how each keystroke will affect the screen state. When it is confident in its predictions, it will show them to the user while waiting for confirmation from the server. Most typing and uses of the left- and right-arrow keys can be echoed immediately.

As a result, Mosh is usable on high-latency links, e.g. on a cellular data connection or spotty Wi-Fi. In distinction from previous attempts at local echo modes in other protocols,

Mosh works properly with full-screen applications such as emacs, vi, alpine, and irssi, and automatically recovers from occasional prediction errors within an RTT. On high-latency links, Mosh underlines its predictions while they are outstanding and removes the underline when they are confirmed by the server.

Mosh does not support X forwarding or the non-interactive uses of SSH, including port forwarding.

Unix Shell tools

Liquidprompt

See also:

- https://linuxfr.org/news/un-prompt-bash-utile-sans-poudre-aux-yeux
- https://github.com/nojhan/liquidprompt

Contents

Introduction

Liquid prompt is a smart prompt for the "Bourne-Again" Unix shell (bash) and for Zsh.

The basic idea of the liquid prompt is to nicely display useful informations on the shell prompt, only when they are needed.

It adds carefuly chosen colors to draw your attention on what differs from the normal context.

Thus, you will notice what changes, when it changes, because you do not become accommodated to informations that are always displayed in the same way.

Liquidprompt source code

See also:

- https://github.com/nojhan/liquidprompt

Installation de Liquidprompt

Liquidprompt est simple à installer et utiliser:

```
wget https://raw.github.com/nojhan/liquidprompt/master/liquidprompt
source liquidprompt
```

Virtualenv

See also:

- http://code.doughellmann.com/virtualenvwrapper/raw/8711d04749986e8261d8bd69f19a4c7086d6883c/docs/source/tips.rst

Warning: For python3, use pyvenv !

Contents

Tips and Tricks

This is a list of user-contributed tips for making virtualenv and virtualenvwrapper even more useful. If you have tip to share, drop me an email or post a comment on this blog post and I'll add it here.

zsh Prompt

From Nat:

Using zsh, I added some bits to `$WORKON_HOME/post(de)activate` to show the active virtualenv on the right side of my screen instead.

in `postactivate`:

```
PS1="$_OLD_VIRTUAL_PS1"
_OLD_RPROMPT="$RPROMPT"
RPROMPT="%{${fg_bold[white]}%}(env: %{${fg[green]}%}`basename \"$VIRTUAL_ENV\"`%{${fg_
↪bold[white]}%})%{${reset_color}%} $RPROMPT"
```

and in `postdeactivate`:

```
RPROMPT="$_OLD_RPROMPT"
```

Adjust colors according to your own personal tastes or environment.

Updating cached `$PATH` entries

From Nat:

I also added the command 'rehash' to `$WORKON_HOME/postactivate` and `$WORKON_HOME/postdeactivate` as I was having some problems with zsh not picking up the new paths immediately.

Tying to pip's virtualenv support

Via http://becomingguru.com/:

Add this to your shell login script to make pip use the same directory for virtualenvs as virtualenvwrapper:

```
export PIP_VIRTUALENV_BASE=$WORKON_HOME
```

and Via Nat:

in addition to what becomingguru said, this line is key:

```
export PIP_RESPECT_VIRTUALENV=true
```

That makes pip detect an active virtualenv and install to it, without having to pass it the -E parameter.

Creating Project Work Directories

Via James:

In the `postmkvirtualenv` script I have the following to create a directory based on the project name, add that directory to the python path and then cd into it:

```
proj_name=$(echo $VIRTUAL_ENV|awk -F'/' '{print $NF}')
mkdir $HOME/projects/$proj_name
add2virtualenv $HOME/projects/$proj_name
cd $HOME/projects/$proj_name
```

In the `postactivate` script I have it set to automatically change to the project directory when I use the workon command:

```
proj_name=$(echo $VIRTUAL_ENV|awk -F'/' '{print $NF}')
cd ~/projects/$proj_name
```

Automatically Run workon When Entering a Directory

Justin Lily posted about some code he added to his shell environment to look at the directory each time he runs `cd`. If it finds a `.venv` file, it activates the environment named within. On leaving that directory, the current virtualenv is automatically deactivated.

Harry Marr wrote a similar function that works with git repositories.

Installing Common Tools Automatically in New Environments

Via rizumu:

I have this `postmkvirtualenv` to install the get a basic setup.

```
$ cat postmkvirtualenv
#!/usr/bin/env bash
curl -O http://python-distribute.org/distribute_setup.p... />python distribute_setup.
↪py
rm distribute_setup.py
easy_install pip==dev
pip install Mercurial
```

Then I have a pip requirement file with my dev tools.

```
$ cat developer_requirements.txt
ipdb
ipython
pastescript
nose
http://douglatornell.ca/software/python/Nosy-1.0.tar.gz
coverage
sphinx
grin
pyflakes
pep8
```

Then each project has it's own pip requirement file for things like PIL, psycopg2, django-apps, numpy, etc.

Changing the Default Behavior of `cd`

Via mae:

This is supposed to be executed after workon, that is as a `postactivate` hook. It basically overrides `cd` to know about the VENV so instead of doing `cd` to go to ~ you will go to the venv root, IMO very handy and I can't live without it anymore. if you pass it a proper path then it will do the right thing.

```
cd () {
    if (( $# == 0 ))
    then
        builtin cd $VIRTUAL_ENV
    else
        builtin cd "$@"
    fi
}

cd
```

Terminator

See also:

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

Contents

Terminator is a cross-platform GPL terminal emulator with advanced features not yet found elsewhere.

Terminator will run on any modern OS with Java 5 or later. It replaces xterm, rxvt, xwsh and friends on X11 systems, GNOME Terminal, KDE's Konsole, Apple's Terminal.app, and PuTTY on MS Windows.

Downloads

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

See also:

- http://software.jessies.org/terminator/
- http://www.commandlinefu.com/commands/browse
- http://en.wikipedia.org/wiki/Command-line_interface
- http://fr.wikipedia.org/wiki/Interface_en_ligne_de_commande
- http://fr.wikipedia.org/wiki/Windows_PowerShell

Open Terminator

2.2.4 Command Line Interface (CLI) on Windows

See also:

- windows_tools
- http://commandwindows.com/index.html
- http://windows.developpez.com/cours/ligne-commande/

- http://www.eleves.ens.fr/wintuteurs/advanced/scripting.html

cmd.exe

See also:

- http://fr.wikipedia.org/wiki/Cmd.exe

Command.com

See also:

- http://fr.wikipedia.org/wiki/Command.com

Consoles with multiple tabs

See also:

- windows_tools
- http://sourceforge.net/projects/console/

Console B with multiple tabs

See also:

- http://chocolatey.org/packages/Console2
- windows_tools

Contents

Description

Console2 is a Windows console window enhancement.

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

To install Console2, run the following command from the command line or from PowerShell:

```
C:\> cinst Console2
```

Console Z with multiple tabs

See also:

- windows_tools
- https://github.com/cbucher/console

Contents

Description

This is a modified version of Console 2 for a better experience under Windows Vista/7/8 and a better visual rendering.

Built packages are available here:

- https://github.com/cbucher/console/wiki/Downloads

```
c:\> cinst ConsoleZ
```

Cshell

See also:

- http://cshell.net/
- https://github.com/lukebuehler/CShell/blob/master/Scripts/Tutorial.csx

Cshell articles

Cshell articles 2014

See also:

- http://www.infoworld.com/t/microsoft-net/cshell-c-developers-can-bypass-visual-studio-more-often-242382

Introduction

CShell, an interactive C# scripting environment providing real-time evaluation of code, is expected to have a formal 1.0 release in one to two months, having spent two years in development, says Lukas Buhler, developer of the project.

Leveraging both the Roslyn .Net compiler project and the Mono runtime, open source CShell provides a REPL (read-eval-print-loop) console-like environment.

“Your code is directly evaluated and executed in a shell window; no separate executable has to be compiled and then run in a different process.

More elaborate code can be written in a C# script and then evaluated as one file, only one line, or a selection,” the project’s webpage says.

Buhler likens it to GitHub's recently released Atom code editor, albeit for C#.

Path editors

See also:

- windows_tools
- http://www.redfernplace.com/software-projects/patheditor/

Windows Powershell

See also:

- http://fr.wikipedia.org/wiki/Powershell
- http://www.microsoft.com/powershell
- blogs.msdn.com/powershell
- http://www.microsoft.com/windowsserver2003/technologies/management/powershell/default.mspx
- chocolatey

Introduction

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

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

À l'origine, il était prévu que PowerShell soit inclus dans Windows Vista, mais finalement les deux logiciels ont été disjoints. Microsoft a publié une version beta le 11 septembre 2005, une Release Candidate 1 le 25 avril 2006 et une Release Candidate 2 le 26 septembre 2006.

La version finale a été publiée le 14 novembre 2006. Powershell est également inclus dans Microsoft Exchange Server 2007, sorti au quatrième trimestre 2006, ainsi que la plupart des produits Microsoft sortis depuis.

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

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

Il est intégré nativement dans Windows 7 en version 2.0

For more information about Windows PowerShell, visit the following Microsoft Web sites:

- Windows PowerShell
- Windows PowerShell Blog
- Windows PowerShell SDK

Windows Powershell news

Windows Powershell 2013 news

Got powershell, MISC 67 Mai-Juin 2013, p.60-66

See also:

- misc_67

Outre ses similitudes avec Bash, Powershell est intéressant à étudier car il s'agit là : d'un shell orienté objet, donnant l'accès à l'intégralité du Framework .net, permettant d'interagir directement avec les différents composants de Microsoft (Active Directory, Exchange, SQL server, SharePoint, Hyper-V, ...), intégrant des méthodes permettant d'effectuer du parsing à la volée, permettant l'utilisation des expressions régulières et, évidemment, d'interagir directement avec le système de fichiers.

Windows Powershell tips

Get program version

See also:

- choco

Contents

Introduction

Some programs come with a built-in auto-update, like Notepad++, Firefox, LibreOffice, Google Chrome and more.

If a program gets updated without the use of Chocolatey, Chocolatey of course does not detect it. When after some time the package maintainer updates the package of the affected program, the same as the installed version of the program gets downloaded and installed again.

This behavior is of course not optimal, because it consumes time, download bandwith, cpu usage and hard disk/SSD usage with no use, especially when there are huge file sizes.

In PowerShell there exists a simple way to get the installed program versions through registry keys

For every program on 32-bit-systems and for 64-bit-software on 64-bit systems

```
# For every program on 32-bit-systems and for 64-bit-software on 64-bit systems
$programName = # name of the program
$installedVersion = (Get-ItemProperty␣
↪HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\$programName).
↪DisplayVersion
```

For 32-bit-software on 64-bit systems

> # For 32-bit-software on 64-bit systems $programName = # name of the program $installedVersion = (Get-ItemProperty HKLM:SOFTWAREWow6432NodeMicrosoftWindowsCurrentVersionUninstall$programName).DisplayVersi

The version if the installed program could then be compared to the package version. When the same version is already installed, the Install-ChocolateyPackage helper could be skipped.

How about integrating this into Chocolatey packages?

The code I posted first was not working with every program, so here the improved code bundled with two program examples:

```
# For every program on 32-bit-systems and for 64-bit-software on 64-bit systems
$programUninstallEntryName = CCleaner # The value which contains the registry-key␣
↪"DisplayName" of the affected program
$installedVersion = (Get-ItemProperty␣
↪HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\* | Select DisplayName,␣
↪DisplayVersion | Where-Object {$_.DisplayName -eq "$programUninstallEntryName"}).
↪DisplayVersion
# $installedVersion is the value of the registry-key "DisplayVersion" of the affected␣
↪program

# For 32-bit-software on 64-bit systems
$programUninstallEntryName = CDBurnerXP
$installedVersion = (Get-ItemProperty␣
↪HKLM:\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\* | Select␣
↪DisplayName, DisplayVersion | Where-Object {$_.DisplayName -eq "
↪$programUninstallEntryName"}).DisplayVersion
```

The problem is that the two registry values used here (DisplayName and DisplayVersion) do not always match with the package ID and package version.

A few examples are:

```
Here everything matches:
DisplayName: CDBurnerXP
DisplayVersion: 4.5.1.3868
Here the DisplayName-entry also contains the version:
DisplayName: LibreOffice 4.0.3.3
DisplayVersion: 4.0.3.3
Here nothing matches exactly with the package ID and the package version
DisplayName: Java 7 Update 21
DisplayVersion: 7.0.210
```

Due to these different possibilities, it is impossible to integrate this code into Chocolatey directly to automatically prevent a program from installing again when the same version is already installd.

But how about a helper for the chocolateyInstall.ps1 script? It could look like this:

```
Check-InstalledProgramVersion $programUninstallEntryName $installedVersionMatch
```

If then the program which the package wants to install is already installed in the latest version (identifiable with $installedVersionMatch), this helper should skip every other command in the chocolateyInstall.ps1 script.

Open PowerShell in explorer

See also:

- http://stackoverflow.com/questions/183901/how-to-start-powershell-from-windows-explorer

Contents

Introduction

```
Windows Registry Editor Version 5.00

[HKEY_CLASSES_ROOT\Directory\shell\powershell]
@="Open PowerShell here"
"NoWorkingDirectory"=""
"Extended"=""

[HKEY_CLASSES_ROOT\Directory\shell\powershell\command]
@="C:\\Windows\\system32\\WindowsPowerShell\\v1.0\\powershell.exe -NoExit -Command␣
↪Set-Location -LiteralPath '%L'"
```

A Python Developer's Guide to PowerShell

See also:

- http://mohd-akram.github.io/2013/05/16/a-python-developers-guide-to-powershell

Contents

Introduction

Introduction

Python is a great scripting language - it's available by default on Linux and Mac and so it's easy to quickly write a short script that runs on many systems.

However, this isn't the case on Windows. You need to install Python or wrap your application to distribute it on Windows, so we need an alternative.

Sometimes this is inconvenient, especially if you want to do something simple or deal directly with Windows. This is where PowerShell comes.

Add the directories for your default Python version to the PATH

See also:

- http://docs.python-guide.org/en/latest/starting/install/win/

Contents

Introduction

Typing the full path name for a Python interpreter each time quickly gets tedious, so add the directories for your default Python version to the PATH.

Assuming that your Python installation is in C:Python27, add this to your PATH:

```
C:\Python27\;C:\Python27\Scripts\
```

You can do this easily by running the following in powershell

```
Environment]::SetEnvironmentVariable("Path", "$env:Path;C:\Python27\;
↪C:\Python27\Scripts\", "User")
```

Windows Powershell tools

Chocolatey NuGet - kind of like apt-get, but for Windows

See also:

- chocolatey

Windows Powershell versions

Windows Powershell 3.0

See also:

http://technet.microsoft.com/en-us/magazine/hh641408.aspx

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

Don Jones

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

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

A CTP is an excellent point in the development cycle for you to start experimenting with new Microsoft technology.

Product teams can still accept and act on feedback.

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

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

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

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

CLI Subversion tools

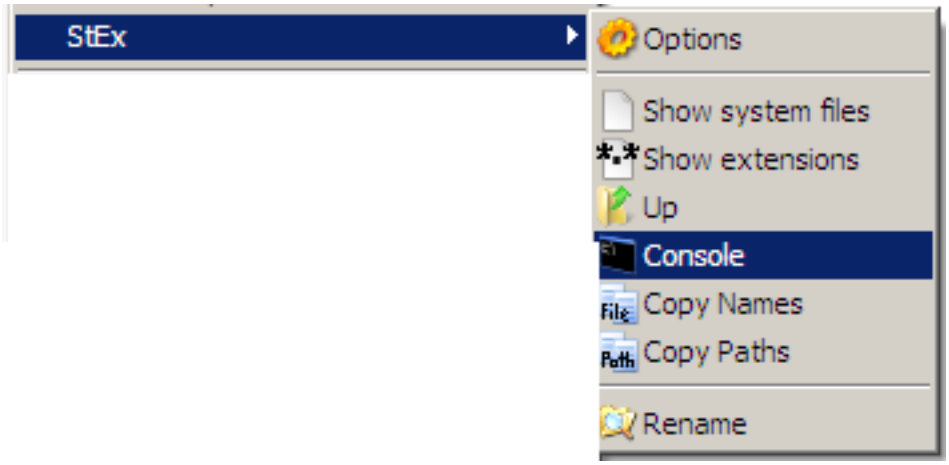
See also:

http://tools.tortoisesvn.net/index.html

Open a windows console with the stex extension

See also:

http://code.google.com/p/stexbar/downloads/list



Wmic

See also:

- wmi
- http://quux.wiki.zoho.com/WMIC-Snippets.html
- http://timgolden.me.uk/python/wmi/cookbook.html

Exemple 1

System, BIOS, Motherboard

This first example shows a few variations of the most common WMI query.

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

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

```
Domain  EnableDaylightSavingsTime  Manufacturer     Model PartOfDomain                ␣
↪            TotalPhysicalMemory  UserName
ID3GEN  TRUE                       Hewlett-Packard  HP Compaq dx2400 Microtower PC ␣
↪TRUE           3479351296           ID3GEN\pvergain
```

List of Windows services

```
wmic service get name, state
```

Resultat

```
Z:\>wmic service get name, state
Name                            State
AdobeFlashPlayerUpdateSvc       Stopped
Alerter                         Stopped
ALG                             Running
AppMgmt                         Stopped
aspnet_state                    Stopped
AudioSrv                        Running
BITS                            Stopped
Browser                         Stopped
CiSvc                           Stopped
ClipSrv                         Stopped
clr_optimization_v2.0.50727_32  Stopped
clr_optimization_v4.0.30319_32  Stopped
COMSysApp                       Stopped
CryptSvc                        Running
DcomLaunch                      Running
Dhcp                            Running
DirMngr                         Stopped
dmadmin                         Stopped
dmserver                        Running
Dnscache                        Running
Dot3svc                         Stopped
EapHost                         Stopped
ERSvc                           Stopped
Eventlog                        Running
EventSystem                     Running
FastUserSwitchingCompatibility  Stopped
FontCache3.0.0.0                Stopped
Gallio.Ambience                 Stopped
gupdate                         Stopped
gupdatem                        Stopped
gusvc                           Stopped
helpsvc                         Stopped
HidServ                         Stopped
hkmsvc                          Stopped
HTTPFilter                      Stopped
IDaAuditMonitor                 Running
IDaSessionMonitor               Running
```

(continues on next page)

(continued from previous page)

```
IDriverT                        Stopped
idsvc                           Stopped
ImapiService                    Stopped
JavaQuickStarterService         Running
lanmanserver                    Running
lanmanworkstation               Running
LightScribeService              Stopped
LmHosts                         Running
MDM                             Stopped
Messenger                       Running
mnmsrvc                         Stopped
MongoDB                         Stopped
MozillaMaintenance              Stopped
MSDTC                           Stopped
MSIServer                       Stopped
MSSQL$SQLEXPRESS                Stopped
MSSQLServerADHelper             Stopped
msvsmon90                       Stopped
napagent                        Stopped
Net Driver HPZ12                Running
NetDDE                          Stopped
NetDDEdsdm                      Stopped
Netlogon                        Running
Netman                          Running
NetTcpPortSharing               Stopped
Nla                             Running
NtLmSsp                         Stopped
NtmsSvc                         Running
ose                             Stopped
PhidgetWebservice21             Stopped
PlugPlay                        Running
Pml Driver HPZ12                Running
PolicyAgent                     Running
ProtectedStorage                Running
RasAuto                         Stopped
RasMan                          Running
RDSessMgr                       Stopped
RemoteAccess                    Stopped
RemoteRegistry                  Running
RpcLocator                      Stopped
RpcSs                           Running
RSVP                            Stopped
SamSs                           Running
SCardSvr                        Running
Schedule                        Running
seclogon                        Running
SENS                            Running
SharedAccess                    Running
ShellHWDetection                Running
Spooler                         Running
SQLBrowser                      Stopped
SQLWriter                       Running
srservice                       Running
SSDPSRV                         Running
stisvc                          Running
stllssvr                        Stopped
SwPrv                           Stopped
```

(continues on next page)

(continued from previous page)

```
SysmonLog                       Stopped
SystemExplorerHelpService       Running
TapiSrv                         Running
TermService                     Running
Themes                          Stopped
TlntSvr                         Stopped
TrkWks                          Running
upnphost                        Stopped
UPS                             Stopped
VSS                             Stopped
W32Time                         Running
WebClient                       Stopped
winmgmt                         Running
WinRM                           Stopped
WmdmPmSN                        Stopped
Wmi                             Stopped
WmiApSrv                        Running
WMPNetworkSvc                   Stopped
WPFFontCache_v0400              Running
wscsvc                          Stopped
wuauserv                        Running
WudfSvc                         Stopped
WZCSVC                          Stopped
xmlprov                         Stopped
```

Start/Stop a Windows service

Start a service

```
wmic service where (name="SCardSvr") call startservice
```

```
Z:\>wmic service where (name="SCardSvr") call startservice
Exécution (\\machine\ROOT\CIMV2:Win32_Service.Name="SCardSvr")->startservice()
Méthode exécutée.
Paramètres de sortie:
instance of __PARAMETERS
{
        ReturnValue = 10;
};
```

Stopping a service

```
wmic service where (name="SCardSvr") call stopservice
```

```
Z:\>wmic service where (name="SCardSvr") call stopservice
Exécution (\\machine\ROOT\CIMV2:Win32_Service.Name="SCardSvr")->stopservice()
Méthode exécutée.
Paramètres de sortie:
instance of __PARAMETERS
{
```

(continues on next page)

(continued from previous page)

```
        ReturnValue = 0;
};
```

State of a service

```
wmic service where (name="SCardSvr") call state
```

```
Z:\>wmic service where (name="SCardSvr") get state
State
Stopped
```

Affiche la liste courte des processus en cours, toutes les 2 secondes

See also:

- http://www.prod-info.fr/Windows/Windows-Commandes-WMIC.html

```
wmic /node:<server_name> process list brief /every:2
```

2.2.5 Google cli

See also:

- http://code.google.com/p/googlecl/
- http://code.google.com/p/googlecl/wiki/ExampleScripts#Picasa
- google_python_data_modules

Contents

GoogleCL brings Google services to the command line.

We currently support the following Google services:

Blogger

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

Calendar

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

Contacts

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

Docs

```
$ google docs edit "Shopping list"
```

Finance

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

Picasa

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

Youtube

```
$ google youtube post --category Education killer_robots.avi
```

Dependencies

GoogleCL requires Python 2.5 or 2.6 and the gdata python client library.

You can get the library from the project homepage:

http://code.google.com/p/gdata-python-client/

Gcalcli

See also:

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

`gcalcli` is a Python application that allows you to access your Google Calendar from a command line. It's easy to get your agenda, search for events, and quickly add new events. Additionally gcalcli can be used as a reminder service to execute any application you want.

Index

Symbols

Numbers

A

B

C

D

E

F

G

H

I

N

O

P

Q

R

S

T

U

V

W

X

Z