Welcome to SphinxBuilder's documentation!

Release 0.8.2

July 11, 2016

Contents

1 Table of contents **3**

1.1 What is Sphinx? . . . 3
1.2 Quick start . . . 3
1.3 Plone 4 . . . 4
1.4 Supported options . . . 4
1.5 Example usage . . . 4
1.6 Reference . . . 8
1.7 Changes . . . 8
1.7.1 1.1 (unreleased) . . . 8
1.7.2 1.0 (2016-07-11) . . . 8
1.7.3 0.9 (2016-06-27) . . . 8
1.7.4 0.8.2 (2013-11-28) . . . 8
1.7.5 0.8.1 (2013-11-27) . . . 8
1.7.6 0.8.0 (2013-11-27) . . . 9
1.7.7 0.7.4 (2013-11-15) . . . 9
1.7.8 0.7.3 (2013-02-16) . . . 9
1.7.9 0.7.2 (2012-10-22) . . . 9
1.7.10 0.7.1 (2012-04-29) . . . 9
1.7.11 0.7.0 (2010-09-10) . . . 9
1.7.12 0.6.3.3 (2010-07-15) . . . 9
1.7.13 0.6.3.2 (2010-02-08) . . . 9
1.7.14 0.6.3.1 (2009-09-25) . . . 9
1.7.15 0.6.3 (2009-09-09) . . . 10
1.7.16 0.5.0 (2008-12-06) . . . 10
1.7.17 0.2.1 (2008-11-18) . . . 10
1.7.18 0.2.0 (2008-11-11) . . . 10
1.7.19 0.1.1 (2008-09-11) . . . 10
1.7.20 0.1.0 (2008-09-10) . . . 10
1.8 Contributors . . . 10

zc.buildout recipe to generate and build Sphinx-based documentation in the buildout.

Date July 11, 2016

Version 0.8

Release 0.8.2

Code repository https://github.com/sdouche/collective.recipe.sphinxbuilder

Questions and comments tarek_at_ziade.org, rok_at_garbas.si, sdouche_at_gmail.com

CHAPTER 1

Table of contents

1.1 What is Sphinx?

Sphinx is the mainly tool in the Python community to build documentation. See http://sphinx.pocoo.org.

It is now used for instance by Python. See http://docs.python.org and many others

Sphinx uses reStructuredText, and can be used to write your buildout-based application. This recipe sets everything up for you, so you can provide a nice-looking documentation within your buildout, in static html, PDF or even epub.

The fact that your documentation is managed like your code makes it easy to maintain and change it.

1.2 Quick start

To use the recipe, add in your buildout configuration file a section like this:

```
[buildout]
parts =
    ...
    sphinxbuilder
    ...

[sphinxbuilder]
recipe = collective.recipe.sphinxbuilder
source = ${buildout:directory}/docs-source
build = ${buildout:directory}/docs
```

Run your buildout and you will get a few new scripts in the *bin* folder, called:

- *sphinx-quickstart*, to quickstart sphinx documentation
- *sphinxbuilder*, script that will

To quickstart a documentation project run, as you would normaly do with Sphinx:

```
$ bin/sphinx-quickstart
```

and anwser few questions and choose *docs-source* as you source folder.

To build your documentation, just run the sphinx script:

```
$ bin/sphinxbuilder
```

That's it!

You will get a shiny Sphinx documenation in *docs/html*. Write your documentation, go in *docs-source*. Everytime source is modified, *sphinxbuilder* run script again.

A good starting point to write your documentation is: http://sphinx.pocoo.org/contents.html.

1.3 Plone 4

Usage with Plone 4 is even easier:

```
[buildout]
parts =
    ...
    sphinxbuilder
    ...

[sphinxbuilder]
recipe = collective.recipe.sphinxbuilder
interpreter = ${buildout:directory}/bin/zopepy
```

Follow quick-start tutorial and do not forget to add interpreter with installed eggs to access your sourcecode with Sphinx.

1.4 Supported options

The recipe supports the following options:

build (default: *docs*) Specify the build documentation root.

source (default: *{build-directory}/source*) Speficy the source directory of documentation.

outputs (default: *html*) Multiple-line value that defines what kind of output to produce. Can be *doctest*, *html*, *latex*, *pdf* or *epub*.

script-name (default: name of buildout section) The name of the script generated

interpreter Path to python interpreter to use when invoking sphinx-builder.

extra-paths Extra paths to be inserted into sys.path.

products Extra product directories to be extend the Products namespace for old-style Zope Products.

1.5 Example usage

The recipe can be used without any options. We'll start by creating a buildout that uses the recipe:

```
>>> write('buildout.cfg',
... """
... [buildout]
... parts = sphinxbuilder
...
... [sphinxbuilder]
... recipe = collective.recipe.sphinxbuilder
... source = collective.recipe.sphinxbuilder:docs
... """)
```

Let's run the buildout:

```
>>> print('start ' + system(buildout))
...
start Installing sphinxbuilder.
collective.recipe.sphinxbuilder: writing MAKEFILE..
collective.recipe.sphinxbuilder: writing BATCHFILE..
collective.recipe.sphinxbuilder: writing custom sphinx-builder script..
Generated script '/sample-buildout/bin/sphinx-quickstart'.
Generated script '/sample-buildout/bin/sphinx-build'.
Generated script '/sample-buildout/bin/sphinx-apidoc'.
Generated script '/sample-buildout/bin/sphinx-autogen'...
```

What are we expecting?

A *docs* folder with a Sphinx structure:

```
>>> docs = join(sample_buildout, 'docs')
>>> ls(docs)
- Makefile
-  make.bat
```

A script in the *bin* folder to build the docs:

```
>>> bin = join(sample_buildout, 'bin')
>>> ls(bin)
- buildout
- sphinx-apidoc
- sphinx-autogen
- sphinx-build
- sphinx-quickstart
- sphinxbuilder
```

The content of the script is a simple shell script:

```
>>> script = join(sample_buildout, 'bin', 'sphinxbuilder')
>>> print(open(script).read())
cd ...docs
make html

>>> print('start ' + system(script))
start /sample-buildout/bin/sphinx-build -b html -d /sample-buildout/docs/doctrees ...src/collective/r
...
```

If we want *latex*, we need to explicitly define it:

```
>>> write('buildout.cfg',
... """
... [buildout]
... parts = sphinxbuilder
...
... [sphinxbuilder]
... recipe = collective.recipe.sphinxbuilder
... source = collective.recipe.sphinxbuilder:docs
... outputs =
...     html
...     latex
... """)
>>> print('start ' + system(buildout))
...
start Uninstalling sphinxbuilder.
```

```
Installing sphinxbuilder.
collective.recipe.sphinxbuilder: writing MAKEFILE..
collective.recipe.sphinxbuilder: writing BATCHFILE..
collective.recipe.sphinxbuilder: writing custom sphinx-builder script...
```

Let's see our script now:

```
>>> cat(script)
cd ...docs
make html
make latex
```

Finally let's run it:

```
>>> print('start ' + system(script))
start /sample-buildout/bin/sphinx-build -b html -d /sample-buildout/docs/doctrees   .../src/collectiv
...

Build finished. The HTML pages are in /sample-buildout/docs/html.
...
Build finished; the LaTeX files are in /sample-buildout/docs/latex.
Run `make' in that directory to run these through (pdf)latex...
```

If we want *pdf*, we need to explicitly define it:

```
>>> write('buildout.cfg',
... """
... [buildout]
... parts = sphinxbuilder
...
... [sphinxbuilder]
... recipe = collective.recipe.sphinxbuilder
... source = collective.recipe.sphinxbuilder:docs
... outputs =
...     html
...     latex
...     pdf
... """)
>>> print('start ' + system(buildout))
...
start Uninstalling sphinxbuilder.
Installing sphinxbuilder.
collective.recipe.sphinxbuilder: writing MAKEFILE..
collective.recipe.sphinxbuilder: writing BATCHFILE..
collective.recipe.sphinxbuilder: writing custom sphinx-builder script...
```

Let's see our script now:

```
>>> cat(script)
cd ...docs
make html
make latex
cd /sample-buildout/docs/latex && make all-pdf
```

We will skip running the script in tests, because the PDF builder depends on libraries which may not be installed.

If we want *epub*, like pdf we need to explicitly define it:

```
>>> write('buildout.cfg',
... """
```

```
... [buildout]
... parts = sphinxbuilder
...
... [sphinxbuilder]
... recipe = collective.recipe.sphinxbuilder
... source = collective.recipe.sphinxbuilder:docs
... outputs =
...     html
...     epub
... """)
>>> print('start ' + system(buildout))
...
start Uninstalling sphinxbuilder.
Installing sphinxbuilder.
collective.recipe.sphinxbuilder: writing MAKEFILE..
collective.recipe.sphinxbuilder: writing BATCHFILE..
collective.recipe.sphinxbuilder: writing custom sphinx-builder script...
```

Let's see our script now:

```
>>> cat(script)
cd ...docs
make html
make epub
```

We can also have the script run any doctests in the docs while building:

```
>>> write('buildout.cfg',
... """
... [buildout]
... parts = sphinxbuilder
...
... [sphinxbuilder]
... recipe = collective.recipe.sphinxbuilder
... source = collective.recipe.sphinxbuilder:docs
... outputs =
...     doctest
...     html
... """)
>>> print('start ' + system(buildout))
...
start Uninstalling sphinxbuilder.
Installing sphinxbuilder.
collective.recipe.sphinxbuilder: writing MAKEFILE..
collective.recipe.sphinxbuilder: writing BATCHFILE..
collective.recipe.sphinxbuilder: writing custom sphinx-builder script...
```

Let's see our script now:

```
>>> cat(script)
cd ...docs
make doctest
make html
```

Again, we will skip running them, this time to avoid a recursive fork bomb. ;)

If we want *extra-paths*, we can define them as normal paths or as unix wildcards (see *fnmatch* module)

```
>>> write('buildout.cfg',
... """
```

```
... [buildout]
... parts = sphinxbuilder
...
... [sphinxbuilder]
... recipe = collective.recipe.sphinxbuilder
... source = collective.recipe.sphinxbuilder:docs
... extra-paths =
...     develop-eggs/
...     eggs/*
... """)
>>> print('start ' + system(buildout))
...
start Uninstalling sphinxbuilder.
Installing sphinxbuilder.
collective.recipe.sphinxbuilder: writing MAKEFILE..
collective.recipe.sphinxbuilder: writing BATCHFILE..
collective.recipe.sphinxbuilder: writing custom sphinx-builder script..
collective.recipe.sphinxbuilder: inserting extra-paths...
```

1.6 Reference

1.7 Changes

1.7.1 1.1 (unreleased)

- Nothing changed yet.

1.7.2 1.0 (2016-07-11)

- Added windows support. [tkhyn]

1.7.3 0.9 (2016-06-27)

- Added automatic testing with travis-ci.org. [reinout]
- Trying to fix install on python 3. [reinout]
- Current code location is https://github.com/reinout/collective.recipe.sphinxbuilder [reinout]
- Documentation is now on readthedocs: http://collectiverecipesphinxbuilder.readthedocs.io/ [reinout]

1.7.4 0.8.2 (2013-11-28)

- Prevent a warning when installing with python 3 [reinout]

1.7.5 0.8.1 (2013-11-27)

- Add Python 3 classifier

1.7.6 0.8.0 (2013-11-27)

- Added python 3 support [reinout]

1.7.7 0.7.4 (2013-11-15)

- Update to Buildout 2
- Cleanup the code to make PEP8 tool happy

1.7.8 0.7.3 (2013-02-16)

- Patch sphinx-build script to terminate with sys.exit()
- Add warnings-html makefile option
- Install sphinx-apidoc and sphinx-autogen scripts
- Rename all txt files to rst ones

1.7.9 0.7.2 (2012-10-22)

- Requires Sphinx >= 1.1

1.7.10 0.7.1 (2012-04-29)

- Add the epub output.
- **Fixed tests:**
  - Use required package versions during tests
  - Use standard *doctest* instead of *zope.testing.doctest*.

1.7.11 0.7.0 (2010-09-10)

- Requires Sphinx >= 1.0

1.7.12 0.6.3.3 (2010-07-15)

- Added *doctest* option to recipe's *output* options (tseaver)
- Relaxed required version of Sphinx to allow versions later than 0.6.4 (but still less than 0.7dev).

1.7.13 0.6.3.2 (2010-02-08)

- Fixed interpreter options [iElectric]

1.7.14 0.6.3.1 (2009-09-25)

- Problems with previous release [garbas]

1.7.15 0.6.3 (2009-09-09)

- Update to Sphinx 0.6.3 [garbas]
- Simplify sphinxbuilder [garbas]
- Update documentation [garbas]
- Interpreter options [iElectric]
- Added logging [iElectric]

1.7.16 0.5.0 (2008-12-06)

- Making it compatible with latest sphinx 0.5 [Rok Garbas]
- Allow for specifying 'product_directories' in order to be able to document old-style Zope Products. [Sidnei]

1.7.17 0.2.1 (2008-11-18)

- Manifest file fixed and making fix release [Rok Garbas]

1.7.18 0.2.0 (2008-11-11)

- Source tree generated every time under parts/<buildout-section-name> [Rok Garbas]
- Finds conf options, source, static and template files using entry_point 'collective.recipe.sphinxbuilder' [Rok Garbas]
- Custom source folder at docs/source [Rok Garbas]
- Build section moved to docs/html, docs/latex [Rok Garbas]

1.7.19 0.1.1 (2008-09-11)

- Using a sphinx-build local to the environment [Tarek]

1.7.20 0.1.0 (2008-09-10)

- Initial implementation [Tarek Ziade]
- Created recipe with ZopeSkel [Tarek Ziade].

1.8 Contributors

- Tarek Ziade, original author
- Rok Garbas
- Sidnei da Silva
- Hans-Peter Locher
- Domen Kozar

- Tres Seaver
- Reinout van Rees, maintainer
- Sébastien Douche