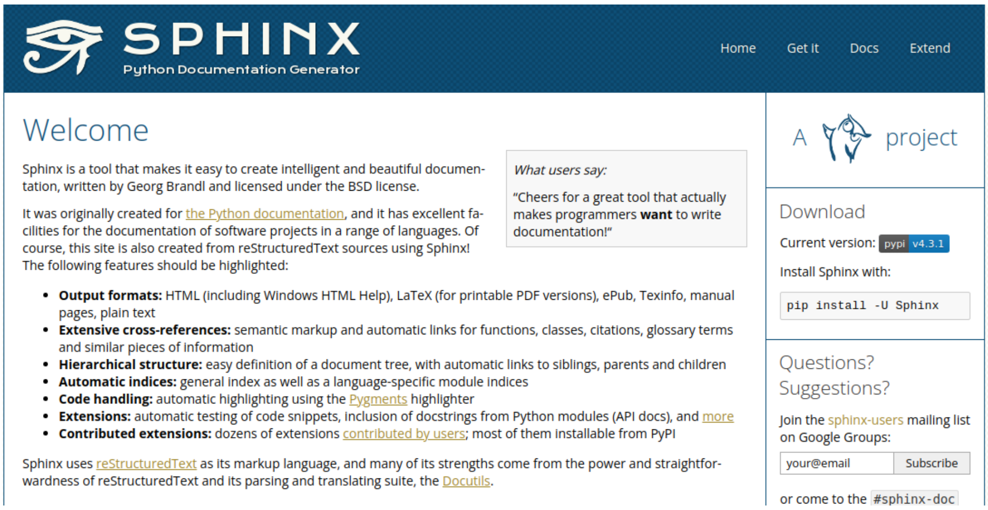
Sphinx: documentation of python scripts

Institut für den Schutz terrestrischer Infrastruktur (PI)

Exported on 12/13/2021

Table of Contents

Sphinx is a tool that helps to auto-generate code documentation. Here we make use of Sphinx within python and show a practicable example on how to get to work with this. For more information, please visit the following project page: https://www.sphinx-doc.org/en/master/



Welcome

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 Python documentation, and it has excellent facilities for the documentation of software projects in a range of languages. Of course, this site is also created from reStructuredText sources using Sphinx! The following features should be highlighted:

What users say:

"Cheers for a great tool that actually makes programmers **want** to write documentation!"

- **Output formats:** HTML (including Windows HTML Help), LaTeX (for printable PDF versions), ePub, Texinfo, manual pages, plain text
- **Extensive cross-references:** semantic markup and automatic links for functions, classes, citations, glossary terms and similar pieces of information
- **Hierarchical structure:** easy definition of a document tree, with automatic links to siblings, parents and children
- **Automatic indices:** general index as well as a language-specific module indices
- **Code handling:** automatic highlighting using the Pygments highlighter
- **Extensions:** automatic testing of code snippets, inclusion of docstrings from Python modules (API docs), and more
- **Contributed extensions:** dozens of extensions contributed by users; most of them installable from PyPI

Sphinx uses reStructuredText as its markup language, and many of its strengths come from the power and straightforwardness of reStructuredText and its parsing and translating suite, the Docutils.

Download

Current version: pypi v4.3.1

Install Sphinx with:

```
pip install -U Sphinx
```

Questions? Suggestions?

Join the sphinx-users mailing list on Google Groups:

or come to the `#sphinx-doc`

Table of Contents:

1 First Steps

First of all, lets proceed to install it. Lets start with the requirements.

1.1 Requirements

- Python (version >=3) and to have it already installed.
- Latex. This will allow us to use other functions of sphinx.
- pip. Note: In case your python manager is conda, pip comes by default as a package manager. Suggestion: You can also create a new environment. For more details see: https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html

1.2 Installation

Lets proceed to install Sphinx and Read The Docs Theme. In a terminal type:

```
$ pip install sphinx
$ pip install sphinx_rtd_theme
```

1.3 Create the 'docs' folder

Lets get hands on in our toy project. You can download the example here[1]. Go to the folder where the .zip folder has been downloaded. In a terminal type:

```
$ cd project
$ mkdir docs
$ cd docs
```

1.4 Generate documentation

Then, lets proceed to initialise it. In a terminal type:

```
$ sphinx-quickstart
```

The following information will be displayed. Answer the questions accordingly. See the code block as an example:

[1] https://wiki.dlr.de/download/attachments/355022951/project.zip?api=v2&modificationDate=1638289803620&version=1

```
$ sphinx-quickstart
Welcome to the Sphinx 4.0.2 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Selected root path: .

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]: y

The project name will occur in several places in the built documentation.
> Project name: Sphinx Tutorial
> Author name(s): O. H. Ramirez Agudelo
> Project release []: 0.1

If the documents are to be written in a language other than English,
you can select a language here by its language code. Sphinx will then
translate text that it generates into that language.

For a list of supported codes, see
https://www.sphinx-doc.org/en/master/usage/configuration.html#confval-language.
> Project language [en]: en

Creating file .../project/docs/source/conf.py.
Creating file .../project/docs/source/index.rst.
Creating file .../project/docs/Makefile.
Creating file .../project/docs/make.bat.

Finished: An initial directory structure has been created.

You should now populate your master file .../project/docs/source/index.rst and create other documentation
source files. Use the Makefile to build the docs, like so:
   make builder
where "builder" is one of the supported builders, e.g. html, latex or linkcheck.
```

Now check the files within docs. Type ls in a terminal:

```
$ ls
```

1.5 Create the HTML file

Go to the folder source. Open 'conf.py' with your favourite editor. Add the following line to the conf.py and remove the old theme line.

```
$ ls
build  make.bat  Makefile  source
$ cd source
$ code conf.py &

html_theme = "sphinx_rtd_theme"
```

Saved the changes. Go back to 'docs' and execute the following command in the terminal.

```
$ make html
```

Now go to /build folder and open the file 'index.html' in a browser.

1.6 Create the pdf version through latex

```
$ make latexpdf
```

Go to '/build/latex/' and open the newly created '.pdf' file.

2 Documentation

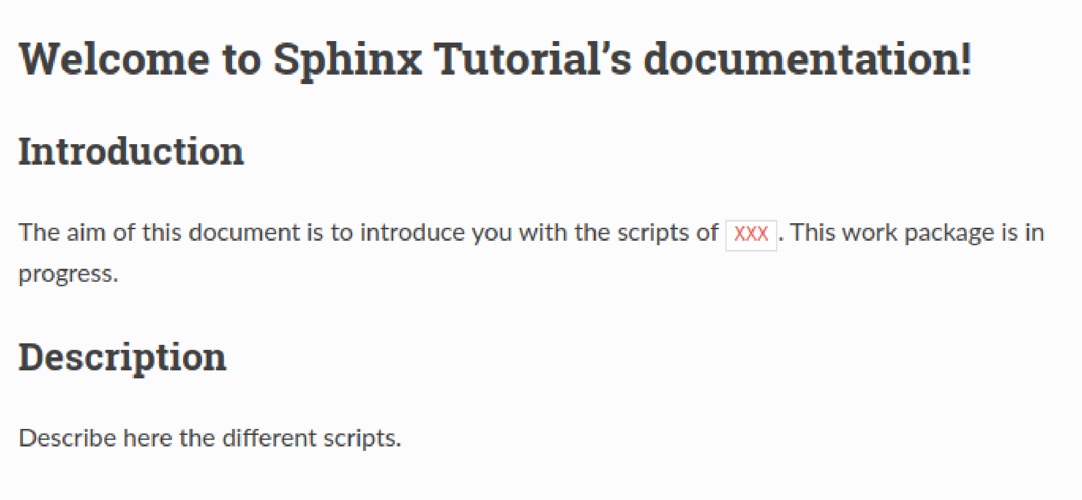
2.1 To add the description of the project in the 'index.rst'

Go to /source/index.rst

Add a description after Welcome to Sphinx tutorial's documentation!. For instance in our example lets add:

```
''''
Introduction
------------
The aim of this document is to introduce you with the scripts of ``XXX``. This work package is in progress.
Description
-----------
Describe here the different scripts.
''''
```

Execute "make html" again and have a look to the index.html. It will look similar to the next figure.



Welcome to Sphinx Tutorial's documentation!

Introduction

The aim of this document is to introduce you with the scripts of XXX. This work package is in progress.

Description

Describe here the different scripts.

2.2 Add the API reference: the description of functions and methods can be found at the `API reference`

Go to /source
Create a file called 'scripts.rst' with the next content. You can also download this file here[2]. If you download it, make sure to put it within the folder 'source'

[2] https://wiki.dlr.de/download/attachments/355022951/scripts.rst?api=v2&modificationDate=1638352781727&version=1

```
Description -- API
==================

This part of the contains details about the methods implemented for ``Service``.

.. contents::
    :local:

Modules
-------

.. contents::
    :local:

.. automodule:: functions.sync_time
    :members:
    :undoc-members:
    :show-inheritance:
```

In the conf.py, add the path to the folder "functions" module by adding it to the Python path:

```
import os
import sys
sys.path.insert(0, os.path.abspath("../../"))
```

Add 'sphinx.ext.napoleon' for a nicer view of the API's parameters:

```
extensions = [
    'sphinx.ext.autodoc',
    'sphinx.ext.intersphinx',
    'sphinx.ext.viewcode',
    'sphinx.ext.napoleon'
]
```

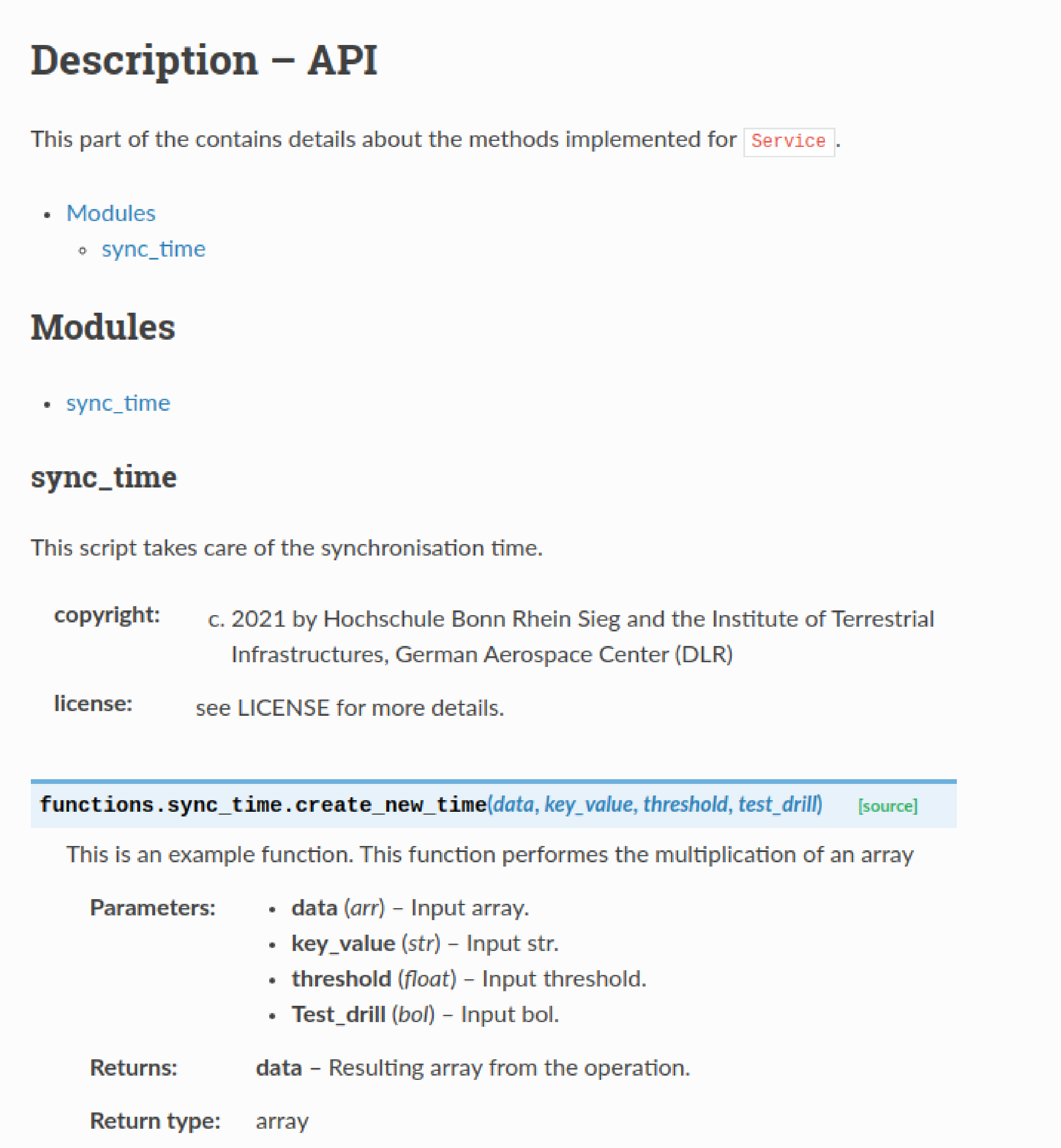
In index.rst it is important to add a line so that it reads the newly created file: 'scripts.rst'. Please update the toctree (table of contents) to the next:

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

    scripts
```

Execute again "make html" and have a look to the index.html.

The result should look more or less like this:



Description – API

This part of the contains details about the methods implemented for `Service`.

- Modules
    - sync_time

Modules

- sync_time

sync_time

This script takes care of the synchronisation time.

copyright: c. 2021 by Hochschule Bonn Rhein Sieg and the Institute of Terrestrial Infrastructures, German Aerospace Center (DLR)

license: see LICENSE for more details.

`functions.sync_time.create_new_time`(*data, key_value, threshold, test_drill*) [source]

This is an example function. This function performes the multiplication of an array

Parameters:

- **data** (*arr*) – Input array.
- **key_value** (*str*) – Input str.
- **threshold** (*float*) – Input threshold.
- **Test_drill** (*bol*) – Input bol.

Returns: **data** – Resulting array from the operation.

Return type: array