Introduction

What is doc generation for?

Once docstrings are written, documentation can be read in two ways:

1. In interactive mode using the `help` function.
2. In a dedicated document making simpler the large scale diffusion of the code.

This part presents the second way. Such a dedicated document is built from the docstrings in `*.py` files.

When to generate documentation?

Documentation generation must be done when the code API is stable. Recall that the API is all the functions, classes and methods that makes it possible to use the code without caring about its internal behaviour.

Overview

There are 3 main steps to doc generation.

1. Analysis of Python code to extract the docstrings. Conversion of these docstrings in documentation files with extension `rst`.

2. Definition of a structure for the final documentation: table of contents, sections, etc...

3. Documentation generation in 2 common formats:

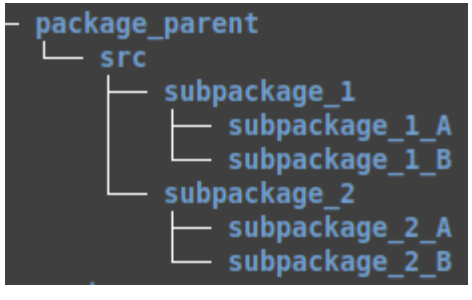
    - html
    - pdf

Details

Setting up the tools

All the doc generation process can be done using **`sphinx`**, which is itself a Python package that can be installed using `conda` or `pip`. Be careful to install `sphinx` in the environment used by the Python project.

Let's document the following package:



Here is the documentation environment set up process:

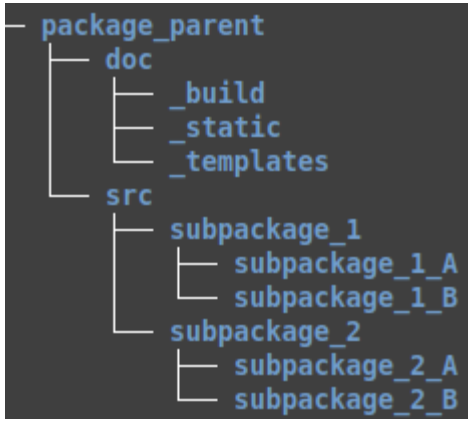
1. Open a commond prompt in directory 'package_parent'
2. Move to a new 'doc' directory
3. Run `sphinx-quickstart`

Some files are created in 'doc':

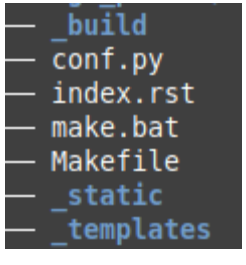
- `conf.py` : contains `sphinx` configuration for the project (step 1 et 3 décrites described in part 'Overview').
- `index.rst` : structure of the final documentation (step 2).

  This is a *reStructuredText* file (*markup language*).

Working directory is now:



And 'doc' directory:



Step 1: Converting docstrings

To have Sphinx understood the *numpy* docstring format, the ***napoleon*** plugin is needed. This is configured in `conf.py`:

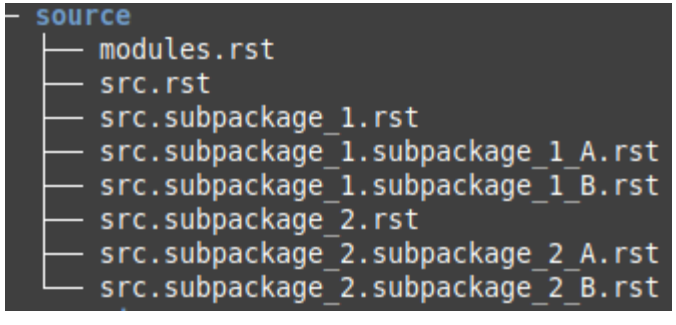
In [1]:
```
extensions = ['sphinx.ext.napoleon']
```

Then conversion can be done. Let's move to `doc` directory and run:

```
sphinx-apidoc -o source/ ../src/
```

What happens:

- directory `../src/` is the one that contains our code: the first `__init__.py` telling Sphinx where to look for the package.
- directory `source` is created and contains `rst` files.



Step 2: Defining a structure

Let's order the `rst` file using the `index.rst` file:

```
3    You can adapt this file completely to your liking, but it should at least
4    contain the root `toctree` directive.
5
6  Welcome to my_package's documentation!
7  ======================================
8
9  .. toctree::
10   :maxdepth: 2
11   :caption: Contents:
12    Here is a short description for our doc page.
13
14   source/src.subpackage_2.subpackage_2_A.rst
```

note: an introduction to `rst` language is available here.

Step 3: Generating documentation

Documentation is generated using the *html* format (the one that descibes web pages).

When generating the documentation, `sphinx` must run the code. Indeed, as stated in the `rst` files of the `source` directory, `sphinx` will look for a package entitled `src`.

Thus a modification of `conf.py` is needed:

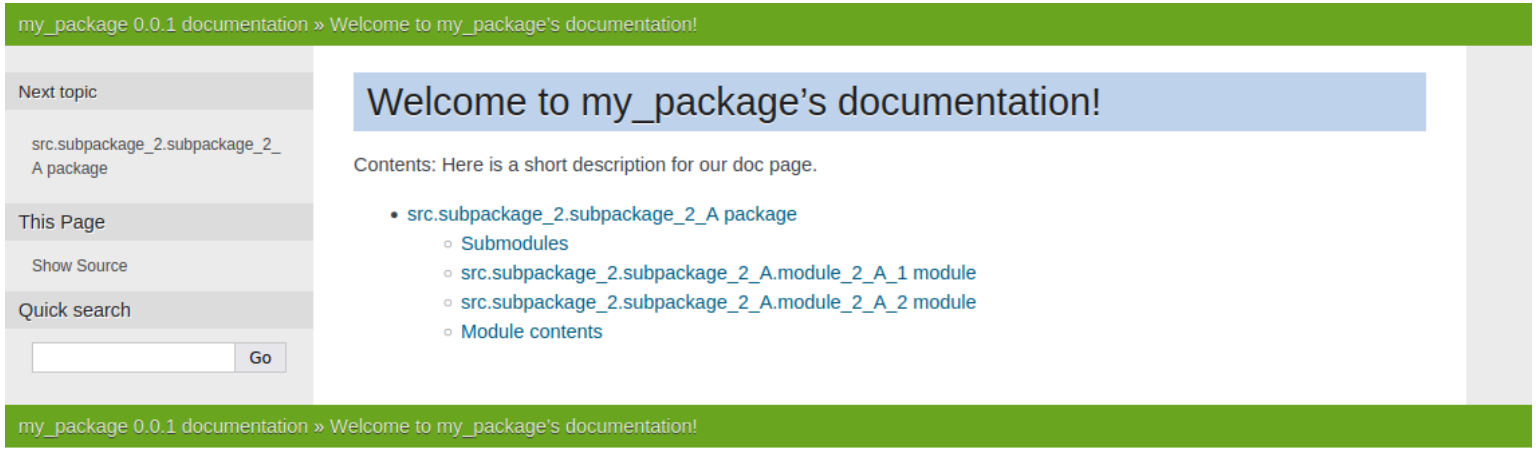
```
from sys import path
path.insert(0,  r'/absolute/path/to/dir/package_parent')
```

Then, back to commond prompt in the `doc` directory:

`sphinx-build  -M html . _build/`.

Result

HTML documentation is opened using `doc/_build/html/index.html`.



src.subpackage_2.subpackage_2_A package

Submodules

src.subpackage_2.subpackage_2_A.module_2_A_1 module

documented_function(*a*, *b*, *c=50*, *mode='sum'*)

Compute either the sum or the product of its arguments, depending on parameter *mode*.

Parameters:
- **a** (*float*) – first parameter of the operation
- **b** (*float*) – second parameter of the operation
- **c** (*float, optional*) – third parameter of the operation, by default 50
- **mode** (*{'sum', 'product'}*) – operation to run on *a*, *b* and *c*

Returns: The result of operation described by *mode*

Return type: float

Raises: **ValueError** – If mode is not one of 'sum' or 'product'

src.subpackage_2.subpackage_2_A.module_2_A_2 module

Module contents

Notes

HTML customization

HTML documentation can be customized. For instance, one can change the theme by modifying the `conf.py` file:

`html_theme = 'nature'`.

All customisation options are described here.

PDF production

A PDF generation is also possible using `latex`:

- `sphinx-build  -M pdf . _build/`
- `cd _build/latex/`
- `make`

That process requires a valid Latex installation.