simples

Versão 0.1

19 jun, 2019

Tutorial

Página de boas vindas da biblioteca `simples`.

- *Vamos lá: Conhecendo o simples*
- *Configuração inicial:*
- *simples package*

`simples` pode ser clonado em jecamanga/simples.

CAPÍTULO 1

Schedule

1.1 Vamos lá: Conhecendo o `simples`

1.1.1 Criando o pacote

Um pacote corresponde a um conjunto de módulos, no `simples` temos dois módulos, o módulo de `tests` e o módulo de `simples`, para saber mais sobre módulos e pacotes, você pode ver o seguinte tutorial em português no devfuria.

Estrutura

`setup.py` corresponde ao arquivo que permite a instalação do seu pacote, a pasta `tests` conterá os módulos que servirão para realizar testes em seu programa, `docs` onde ficará a documentação do seu programa, e por fim a pasta `simples` que conterá os módulos da sua biblioteca, a seguir temos uma estrutura bastante comum, com umas pequenas variações dependendo do tutorial:

```
simples/
    ├── setup.py
    |── simples/
    |── tests/
    └── docs/
```

Alguns tutoriais você pode encontrar em python-packaging, uoftcoders.

1.1.2 Utilizando Docstrings

No Python, as Docstrings correspondem a um conjunto de comentários entre `""" """` realizados no início de um módulo, função, classe ou método.

A seguir, temos alguns tutoriais que ensinam o uso de Docstrings:

- caderno de laboratorio

- python help

Exemplos

No pacote `simples` é usado o padrão do numpy.

Função soma inteiros usando o docstring do numpy:

```
def soma_inteiros(x: int, y: int)-> int:
    """Realiza a soma entre dois inteiros.

    Parameters
    ----------
    x : int
        número inteiro.
    y : int
        número inteiro

    Returns
    -------
    int
        o resultado da soma de x + y
    """
    return x + y
```

1.1.3 Criando testes

A estrutura do `simples` de forma mais básica com a pasta tests:

```
simples/
    ├── setup.py
    └──simples/
    |       |── __init__.py
    |       └── soma.py
    └──tests/
          |── __init__.py
          └── test_soma.py
```

Os testes criados para a biblioteca `simples` foi utilizando o pytest. Um dos exemplos testa se a soma é igual a 8:

```
def test_soma_inteiros():
    assert soma_inteiros(5, 3) == 8, "Resultado deve ser 8"
```

Para saber mais sobre testes no python, você pode acessar alguns dos seguintes materiais:

- Getting Started With Testing in Python
- An introduction to testing in Python
- Improve Your Python: Understanding Unit Testing
- Python Testing
- Primeiros passos com testes unitários em Python
- Teste unitário automático em Python

1.1.4 Documentando

Configuração inicial:

- Instalar o sphinx através do `pip install -U Sphinx`
- Instalar o tema do read the docs `pip install sphinx_rtd_theme`

1. Criação da pasta docs de `simples`.

No terminal:

```
foo@bar:~/simples$ mkdir docs
foo@bar:~/simples$ cd docs
foo@bar:~/simples/docs$ sphinx-quickstart
foo@bar:~/simples/docs$ sphinx-apidoc -f -o source/ ../simples
```

2. Mudanças no arquivo `conf.py`

No arquivo `conf.py` dentro da pasta docs, realizar as seguintes mudanças:

- Descomentar essa linha e modificar o caminho:

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

- Em extensões:

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

- E trocar o tema do html em:

```
html_theme = 'sphinx_rtd_theme'
```

3. Alterar o arquivo `index.rst`

No arquivo `index.rst` adicionar a seguinte linha `source/modules`:

```
.. toctree::
:maxdepth: 2
:caption: Contents:

source/modules

Indices and tables
==================

* :ref:`genindex`
* :ref:`modindex`
* :ref:`search`
```

4. Resultado

Por fim, na pasta docs, no terminal execute `make html`, o html é gerado no caminho docs/_build/html:

```
foo@bar:~/simples/docs$ make html
```

CAPÍTULO 2

Bem vindo(a) a documentação do `simples`

2.1 simples

2.1.1 simples package

Submodules

simples.soma module

soma.py contém as funções utilizadas para realizar a soma entre dois números.

`simples.soma.`**`soma_inteiros`**(*x: int*, *y: int*) → int

Realiza a soma entre dois inteiros.

Parameters

x [int] número inteiro.

y [int] número inteiro

Returns

int o resultado da soma de x + y

Examples

```
>>> soma_inteiros(4, 5)
9
```

`simples.soma.`**`soma_reais`**(*x: float*, *y: float*) → float

Realiza a soma entre dois reais.

Parameters

x [float] número real.

y [float] número real.

Returns

float o resultado da soma x + y.

Module contents

CAPÍTULO 3

Indices and tables

- genindex
- modindex
- search

Índice de Módulos do Python

S

Índice

S