引き継ぎ資料 Vol.5

Pythonのライブラリ等

2016/09/??

コンセプト

“私はこんなライブラリ・ツールを使ってますよ”

要は布教

目次

目次

Pythonには大体なんでもあるよ!

すでにあるライブラリを上手に使おう!

目次

それぞれの役割

numpy 数値計算

scipy (numpyにはない)数値計算

matplotlib グラフ作成

numpy

ベクトル演算・行列演算

numpy

ベクトル演算・行列演算

numpy.ndarrayに対する演算がバリバリ可能

scipy

numpyにはない様々な演算が可能

- 極値計算
- 積分

matplotlib

グラフ作成

大体ご存知でしょうから省略しました

numpyなどとのつきあい方

ググれば大体でる(と思います)

目次

docopt

コマンドラインオプションを扱うライブラリ

Python標準のargparseより簡単かつかっこいい

インストール方法

普通にpipで導入

```
$ pip install docopt
```

docopt

コメントからコマンドラインオプションを指定

例

```
"""docopt sample

Usage:
    example_docopt.py [-ab] <arg1> <arg2>
    example_docopt.py -h

Options:
    -a, --arg1   Argument1
    -b, --ball   Ball
"""
from docopt import docopt
if __name__ == "__main__":
        print(docopt(__doc__))
```

例

```
Usage:
    example_docopt.py [-ab] <arg1> <arg2>
    example_docopt.py -h
```

実行

```
$ python example_docopt.py hoge fuga
{'--arg1': False,
 '--ball': False,
 '-h': False,
 '<arg1>': 'hoge',
 '<arg2>': 'fuga'}
```

例

```
Usage:
      example_docopt.py [-ab] <arg1> <arg2>
      example_docopt.py -h
```

実行

```
$ python example_docopt.py -a hoge fuga
{'--arg1': True,
'--ball': False,
'-h': False,
'<arg1>': 'hoge',
'<arg2>': 'fuga'}
```

Pros.

- 書くのが楽
- コメントとプログラムが必ず一致

Cons.

- 重い
- http://docopt.org/をよく見ないとたまに変な動作

用途

ソースコードの中にファイル名とか入れるのやめよう

ファイルの位置が変わったらソースも書き換え

コマンドラインオプションにしておけば呼び出し時に自由自在

目次

sphinx

Pythonに関するツール[1]

Anacondaにはデフォルトで搭載

[1] sphinxはPython限定ではない

用途

1. 一定の方式に則ってコメントを記述
2. Sphinxで処理
3. プログラムのドキュメント完成

例: コメント記述

Sphinxでは決まったルールでコメントを書くことが必要

ルール

- Sphinx形式(デフォルト)
- Google形式
  http://www.sphinx-doc.org/en/stable/ext/example_google.html
- NumPy形式
  http://www.sphinx-doc.org/en/stable/ext/example_numpy.html

例: コメント記述(Google形式)

```
"""Example Program"""


def sample_function(arg):
    """Sample function

    long description of this function

    Args:
        arg (str): text

    Returns:
        str: printed text
    """
    print(arg)
    return arg
```

例: Sphinxで処理

ソースコードからドキュメントの元を生成

```
$ sphinx-apidoc -F -o doc src
```

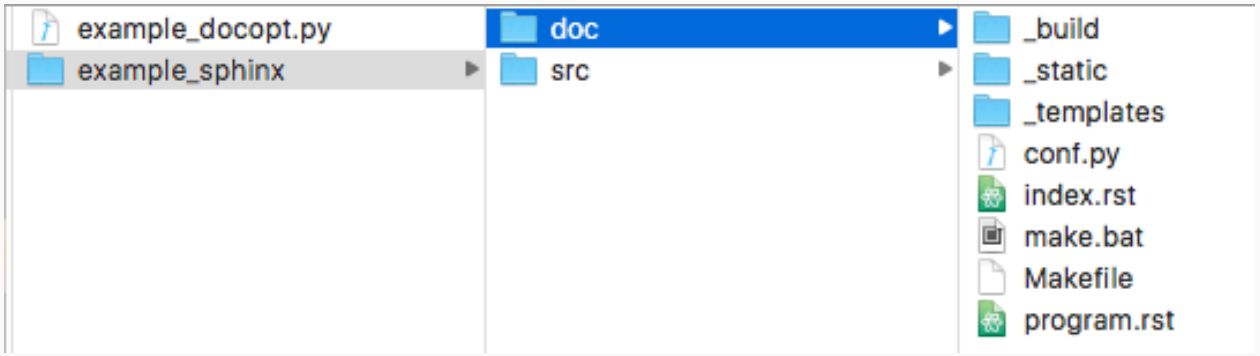
srcの中身を元にdocにドキュメントの元を生成

例: docフォルダの中身

***.rst** ドキュメントの元

Makefile ドキュメント生成のプログラム

conf.py 設定が書かれたpythonファイル



例: conf.py

設定が書かれたpython形式のファイル

記述が必要な事項

- extensionsに”sphinxcontrib.napoleon”追加
  Google形式コメントの有効化
- languageを”ja”に設定
  自動生成される部分を日本語化
- テーマの設定

詳細はサンプルコードを参照

例: ドキュメント完成

ドキュメントディレクトリへ移動

```
$ cd doc
```

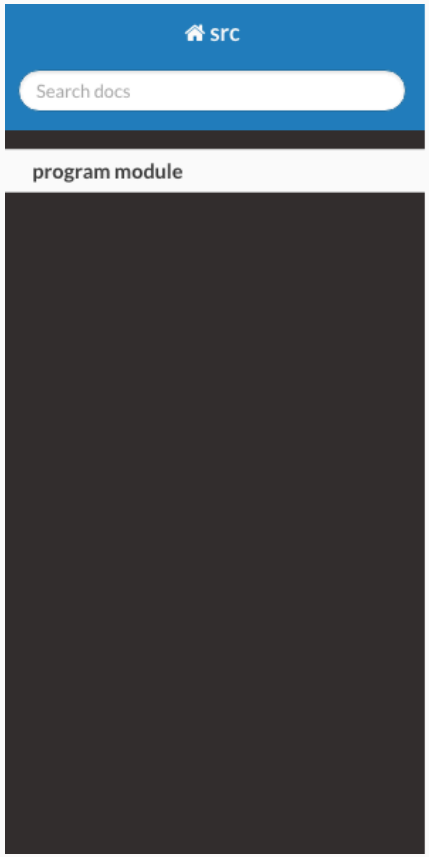
htmlファイルの生成

```
$ make html
```

pdfファイルの生成

```
$ make latexpdfja
```

例:結果(html)



Docs » program module View page source

program module

Example Program

program.sample_function(*arg*) [ソース]

Sample function

long description of this function

パラメータ: arg (*str*) – text

戻り値: printed text

戻り値の型: str

Previous

なぜ使うのか?

- ドキュメントはプログラムの理解に不可欠
- きちんとコメントを書こう
- きちんと綺麗なコメントを書こう
- ルールに則ったコメントを書こう

目次

Pythonにはライブラリがたくさん

きちんと使って楽をしよう

Questions?