@11ohina017

投稿日 2019年07月29日 更新日 2021年05月11日

GoogleスタイルのPython Docstringの入門

Python, Sphinx, コーディング規約, 新人プログラマ応援, docstring

前置き

本記事では、Googleスタイル形式でのPython Docstringの書き方について必要最低限に絞って説明する。

これから、Python Docstringを覚えようとしているエンジニアの参考になれば、幸いである。

Python Docstringとは

Pythonにおけるクラスや、メソッド(関数)についての説明を記載したコメント文のこと。

Docstringは、 __doc__ という変数に格納されている。

以下は、printメソッドのDocstringを表示させたもの。

```
Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
file:  a file-like object (stream); defaults to the current sys.stdout.
sep:   string inserted between values, default a space.
end:   string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.
```

自作したクラスや、メソッドにDocstringを記載しておくと、
IDE上に補足情報として表示させることや、Sphinxを使用して、ソースコードの仕様書を自動作成することが可能になる。

Sphinxとは

Sphinxは、reStructuredTextという形式で記載されたテキストををHTML、
PDFやepubなどの様々な形式へ変換することができるOSSのドキュメント生成ツール。

Pythonの公式ドキュメントはSphinxを使って書かれている。

Sphinxを用いると、Pythonのソースコード上からPython Docstringのコメント文を抽出して、
ソースコード仕様書を自動生成することが可能。

Sphixを使用した、Docstringの活用方法は下記を参照。
https://qiita.com/futakuchi0117/items/4d3997c1ca1323259844

Googleスタイル

Googleが提唱したDocstringの記法の一つ。
Docstringの記法にはreStructuredTextスタイル，Numpyスタイル，Googleスタイルの3つがある。

本記事では、Googleスタイルについて説明する。

GoogleスタイルのPython Docstring

Sphinxのサンプルコード

SphinxのHPから、Googleスタイル形式で記載されたサンプルコードを閲覧することができる。

- Example Google Style Python Docstrings

上記のソースコードをSphinxでhtmlに変換すると下記のようになる。

- https://11ohina017.github.io/google_style_code/index.html

日本語のサンプルコード

先程のサンプルコードを元に、要点のみを抜粋した日本語のサンプルコードと、ソースコードをSphinxでhtmlに変換したものを下記に記す。

- https://11ohina017.github.io/google_style_code/sample.html

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
"""モジュールの説明タイトル

* ソースコードの一番始めに記載すること
* importより前に記載する

Todo:
    TODOリストを記載
    * conf.pyの``sphinx.ext.todo`` を有効にしないと使用できない
    * conf.pyの``todo_include_todos = True``にしないと表示されない

"""

import json
import inspect

class testClass() :
    """クラスの説明タイトル

    クラスについての説明文

    Attributes:
        属性の名前 (属性の型): 属性の説明
        属性の名前 (:obj:`属性の型`): 属性の説明.

    """

    def print_test(self, param1, param2) :
        """関数の説明タイトル

        関数についての説明文

        Args:
            引数の名前 (引数の型): 引数の説明
            引数の名前 (:obj:`引数の型`, optional): 引数の説明.

        Returns:
            戻り値の型: 戻り値の説明 (例 : True なら成功, False なら失敗.)

        Raises:
```

```
            例外の名前: 例外の説明 (例 : 引数が指定されていない場合に発生 )

        Yields:
            戻り値の型: 戻り値についての説明

        Examples:

            関数の使い方について記載

            >>> print_test ("test", "message")
               test message

        Note:
            注意事項などを記載

        """
        print("%s %s" % (param1, param2) )

if __name__ == '__main__':

    test_object = testClass()
    test_object.print_test("test", "message")
```

基本的なコメントの書き方

- コメントを複数行のコメントブロック「"""」で囲む
- 「"""」の右隣にタイトルを記載する
- Docstringの対象となるのは、 モジュール、クラス、関数(メソッド)の3つ

モジュールの記載方法

ソースコードの冒頭に、ソースコード全体つまり、モジュールの説明を記載する。

注意点としては、コメント文を除いたソースコードの一番始めに記載する必要がある。
※ importより前に記載する必要がある。

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
"""モジュールの説明タイトル

モジュールの説明

"""
```

クラスの記載方法

下記のように、クラス定義の下の行に記載する。

```
class testClass() :
    """クラスの説明タイトル

    クラスについての説明文

    """
```

関数(メソッド)の記載方法

下記のように関数定義の下の行に記載する。

```
def print_test(self, param1, param2) :

    """関数の説明タイトル

    関数についての説明文
    """
```

セクションの説明

Googleスタイルでは、Attributes, Args、Returns、Yieds、Raises、Examples、Note、Todoという用途別に定義されたセクションがある。

Attributesセクション

クラスの属性の型、名前など、クラスの属性の説明を記載する。

```
Attributes:
    属性の名前 (属性の型): 属性の説明
    属性の名前 (:obj:`属性の型`): 属性の説明.
```

Argsセクション

引数の名前、型、optional(省略可能)かどうかなど、引数の説明を記載するセクション。

- インスタンスを示すselfは、Argsセクションには記載せず、省略する
- 省略可能な引数は、下記のようにoptionalを記載する
- SphinxでhtmlファイルIに変換すると、Args → Parametersに変更される

```
Args:
    引数の名前 (引数の型): 引数の説明
    引数の名前 (:obj:`引数の型`, optional): 引数の説明.
```

Returnsセクション

retrun文を使用した関数の戻り値を記載するセクション。

```
Returns:
    戻り値の型: 戻り値の説明 (例 : True なら成功, False なら失敗.)
```

Yieldsセクション

yeild文を使用した関数の戻り値を記載するセクション。

```
Yields:
    戻り値の型: 戻り値についての説明
```

Raisesセクション

例外処理に対する説明を記載するセクション。

```
Raises:
    例外の名前: 例外の説明（例 : 引数が指定されていない場合に発生 ）
```

Examplesセクション

関数、クラスの実行例について記載するセクション。

```
Examples:
    関数の使い方について記載

    >>> print_test ("test", "message")
        test message
```

- Examplesだけでなく、Exampleでも可
- クラス、メソッド以外の箇所で使用する場合は、下記のように `::` でコードブロックにする必要がある
- `\` はエスケープ文字なので、コードブロック中に `\` を表示させる場合、`\\` と記載する

```
Example:
    関数の使い方について記載

    ::

        $ python main.py \\
            "arg1" \\
            "args2" \\
```

Noteセクション

注釈について記載するセクション。
例えば、「このコードは、Python2系では動作しません」などのような注意事項を記載するといいだろう。

```
Note:
   注意事項などを記載
```

Todoセクション

Todoリストを記載するためのセクション。
実装予定の処理など、後から実施する作業などはここに記載する。

```
Todo:
    TODOリストを記載
    * conf.pyの``sphinx.ext.todo`` を有効にしないと使用できない
    * conf.pyの``todo_include_todos = True``にしないと表示されない
```

Shphixで変換後のドキュメントに表示させるには、
Sphinxの設定ファイルconf.pyを編集し、下記のように、
拡張機能[sphinx.ext.todo]を有効にする必要がある。