python for Computational Problem Solving - pCPS - Functional_Programming_Testing Lecture Slides - Class #47_#48

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

python for Computational Problem Solving Syllabus

Unit IV: Functional Programming, Modules,Testing and Debugging - 10 Hours

- Functional Programming - map, filter, reduce, max, min, lambda function
- Modules - import mechanisms,
- list comprehension
- Usage of __doc__ , __name__ , __call__
- Testing
  - Pytest , Function testing with Doctest
  - pdb debugger commands.

Usage of __doc__ , __name__ , __call__ in python

- python **docstrings** are the string literals that appear right after the definition of a function, method, class, or module.

- Inside the **triple quotation** marks is the **docstring** of the function, as it appears **right after** its **definition**.

- **Variable** that has **double underscores on both sides**, it's called **dunder** name. The dunder stands for double underscores

```
def PSection():

    '''This is function that prints the information about P Section'''

    '''This is function that prints the information about P Section'''

    print('PES University')
    print('Session September 2021 - March 2022')
    print('B.Tech First Semester')
    print('P Section')
    print('Number of Students--> 70')
    return

print(type(PSection.__doc__))
print(PSection.__doc__)
```

```
<class 'str'>
This is function that prints the information about P Section
```

```
def PSection():
    '''This is function that prints the information about P Section
This is a demo code snippet to understand __doc__ '''
    print('PES University')
    print('Session September 2021 - March 2022')
    print('B.Tech First Semester')
    print('P Section')
    print('Number of Students--> 70')
    return
print(type(PSection.__doc__))
print(help(PSection))
```

```
<class 'str'>
Help on function PSection in module __main__:

PSection()
    This is function that prints the information about P Section
    This is a demo code snippet to understand __doc__

None
```

Usage of __doc__ , __name__ , __call__ in python

- **Standard** conventions to write **single-line docstrings**:
  - Even though they are single-lined, we still use the triple quotes around these **docstrings** as they can be expanded easily later.
  - The **closing quotes** are on the **same line** as the opening quotes.
  - They should not be descriptive, rather they must follow "Do this, return that" structure ending with a period.

```
import math
def PSection():
    '''This is function that prints the information about P Section
This is a demo code snippet to understand __doc__ '''
    print('This is from the Method-->',__name__)
    print('PES University')
    print('Session September 2021 - March 2022')
    print('B.Tech First Semester')
    print('P Section')
    print('Number of Students--> 70')
    return

print(type(PSection.__name__))
print(__name__)
print(PSection.__name__)
PSection()
print(math.__name__)
```

```
<class 'str'>
__main__
PSection
This is from the Method--> __main__
PES University
Session September 2021 - March 2022
B.Tech First Semester
P Section
Number of Students--> 70
math
```

Usage of __doc__ , __name__ , __call__ in python

- **Standard** conventions to write **Multiline Docstrings:**
  - **Multiline** docstrings consist of a summary line just like a one-line docstring, followed by a more elaborate description.
  - The **PEP-257** document provides the **standard conventions** to **write multiline** docstrings for various **objects**.

```
import math
def PSection():
    '''This is function that prints the information about P Section
This is a demo code snippet to understand __doc__ '''
    print('This is from the Method-->',__name__)
    print('PES University')
    print('Session September 2021 - March 2022')
    print('B.Tech First Semester')
    print('P Section')
    print('Number of Students--> 70')
    return

print(type(PSection.__name__))
print(__name__)
print(PSection.__name__)
PSection()
print(math.__name__)
```

```
<class 'str'>
__main__
PSection
This is from the Method--> __main__
PES University
Session September 2021 - March 2022
B.Tech First Semester
P Section
Number of Students--> 70
math
```

Usage of __doc__ , __name__ , __call__ in python

- **Standard** conventions to write **Multiline Docstrings:**
  - **Multiline** docstrings consist of a summary line just like a one-line docstring, followed by a more elaborate description.
  - The **PEP-257** document provides the **standard conventions** to **write multiline** docstrings for various **objects** like
    - **Docstrings for Python Modules**
    - **Docstrings for Python Functions**
    - **Docstrings for Python Classes**
    - **Docstrings for Python Scripts**
    - **Docstrings for Python Packages**

```
import pickle
print(pickle.__doc__)
```

```
Create portable serialized representations of Python objects.

See module copyreg for a mechanism for registering custom picklers.
See module pickletools source for extensive comments.

Classes:

    Pickler
    Unpickler

Functions:

    dump(object, file)
    dumps(object) -> string
    load(file) -> object
    loads(string) -> object

Misc variables:

    __version__
    format_version
    compatible_formats
```

Usage of __doc__ , __name__ , __call__ in python

- python **__name__** is a special variable in which the name of the current python script/module being executed is stored.

- **__name__** is a **built-in** variable in python that **stores** the **name** of the **current module/script** being **executed**.

```
import math
def PSection():
    '''This is function that prints the information about P Section
This is a demo code snippet to understand __doc__ '''
    print('This is from the Method-->',__name__)
    print('PES University')
    print('Session September 2021 - March 2022')
    print('B.Tech First Semester')
    print('P Section')
    print('Number of Students--> 70')
    return

print(type(PSection.__name__))
print(__name__)
print(PSection.__name__)
PSection()
print(math.__name__)
```

```
<class 'str'>
__main__
PSection
This is from the Method--> __main__
PES University
Session September 2021 - March 2022
B.Tech First Semester
P Section
Number of Students--> 70
math
```

Usage of __doc__ , __name__ , __call__ in python

- If the **current module** is **executing** then the **__name__** variable is **assigned** the value **__main__ else** it **simply** contains the **name** of the **module** or **script**.
- Generally, the execution of a python program(script) starts from the very first line that is at the indentation level 0 of that program.
- When a python program is executed, before its execution a **__name__** variable is **created**.
- This **variable** can be used as an **alternate** for the **main** method in python.

```
import math
def PSection():
    '''This is function that prints the information about P Section
This is a demo code snippet to understand __doc__ '''
    print('This is from the Method-->',__name__)
    print('PES University')
    print('Session September 2021 - March 2022')
    print('B.Tech First Semester')
    print('P Section')
    print('Number of Students--> 70')
    return

print(type(PSection.__name__))
print(__name__)
print(PSection.__name__)
PSection()
print(math.__name__)
```

```
<class 'str'>
__main__
PSection
This is from the Method--> __main__
PES University
Session September 2021 - March 2022
B.Tech First Semester
P Section
Number of Students--> 70
math
```

Usage of __doc__ , __name__ , __call__ in python

- The **__init__** method is **used** for **object constructors** and the **__call__ method** for making **object callable**.

- The **callable()** method **returns** the **boolean** value for whether the **object** appears to be **callable**.

- This **function** returns **True** if the object is **callable**; **else**, it returns **False**.

```
class ClassRoom:
    University = 'PES University'

    def __init__(self,Section = 'P Section',Session = 'September 2021 to March 2022'):
        ClassRoom.Section = Section
        ClassRoom.Session = Session

    def __call__(self):
        return True

print('Class is Callable? ',callable(ClassRoom))
print('Class is Callable? ',ClassRoom.__call__)

P = ClassRoom()
print('Object is Callable? ',callable(P))
print('__call__ is Callable? ',P.__call__)
print('Function __call__ is Callable? ',P.__call__())
```

```
Class is Callable?  True
Class is Callable?  <function ClassRoom.__call__ at 0x7f05f4152af0>
Object is Callable?  True
__call__ is Callable?  <bound method ClassRoom.__call__ of <__main__.ClassRoom object at 0x7f05f410ba00>>
Function __call__ is Callable?  True
```

Usage of __doc__ , __name__ , __call__ in python

- It is also **possible** that this function may return **True** even when the **object** is **not callable**.

```
class ClassRoom:
    University = 'PES University'

    def __init__(self,Section = 'P Section',Session = 'September 2021 to March 2022'):
        ClassRoom.Section = Section
        ClassRoom.Session = Session

    def __call__(self):
        return True

print('Class is Callable? ',callable(ClassRoom))
print('Class is Callable? ',ClassRoom.__call__)

P = ClassRoom()
print('Object is Callable? ',callable(P))
print('__call__ is Callable? ',P.__call__)
print('Function __call__ is Callable? ',P.__call__())
```

```
Class is Callable?  True
Class is Callable?  <function ClassRoom.__call__ at 0x7f05f4152af0>
Object is Callable?  True
__call__ is Callable?  <bound method ClassRoom.__call__ of <__main__.ClassRoom object at 0x7f05f410ba00>>
Function __call__ is Callable?  True
```

End of class #47, #48

Portions completed for ISA-2

(Unit 3 and Unit 4)

Thank you

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University
nitin.pujari@pes.edu

For Course Digital Deliverables visit www.pesuacademy.com