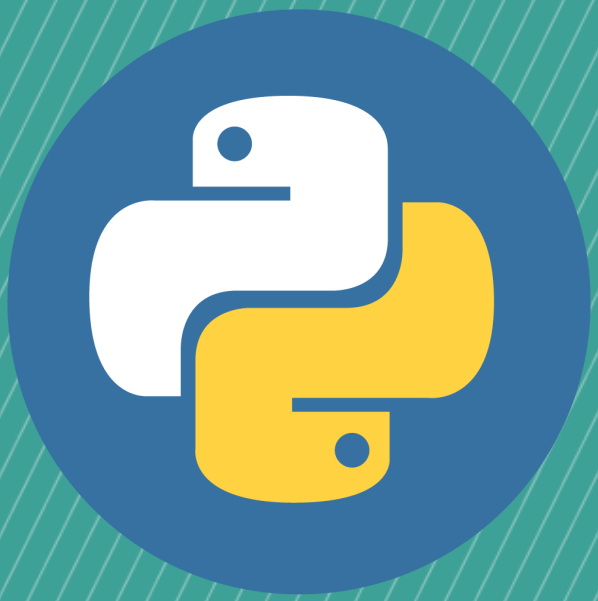


DOCUMENTACIÓN EN *Python*

PROTECO.MX

IMPORTANCIA Y MÓDULO PYDOC

La documentación es importante porque permite que diversos programadores puedan entender lo que ya está programado previamente, de nada sirve un programa sin una buena documentación, puesto que nadie (a excepción de quien lo programó) sabrá cómo funciona.

Para documentar un programa en Python se utiliza un modulo llamado "Pydoc", este módulo permite generar un archivo html en base a los comentarios hechos mediante docstring.

EJEMPLO:

Vamos a ejemplificar cómo realizar correctamente la documentación de un programa con pydoc mediante un ejemplo, el código que se usará como ejemplo es el siguiente:

Welcome Guide | modulo.py | GrafoS.c

```
"""Descripcion del modulo"""
class UnaClase(object):
    """Descripcion de la clase"""
    def un_metodo(self):
          """Descripcion de un metodo"""

def una_funcion():
    """Esta es una funcion"""
```

01

Guardaremos el programa de ejemplo como “modulo.py”. La convención docstring consiste en comentar, utilizando tres comillas, cada apartado del código, como se aprecia en la imagen.

02

Función help

En un código que se encuentre comentado con docstring podemos obtener una pequeña ayuda de lo que éste realizará mediante la función help.

Para ello debemos situar nuestra terminal donde se encuentra el archivo que queremos ver (si es un módulo de los que Python instala por defecto, este paso se omite), abriremos el intérprete de python en nuestra terminal, importamos el módulo, y utilizamos la función help para ver qué es lo que hace.

```
andrea@andrea-HP-ENVY-Laptop-13-ad0xx:~/Documentacion$ python3
Python 3.6.3 (default, Oct  3 2017, 21:45:48)
[GCC 7.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import modulo
>>> help(modulo)
```

Ejecutando esto obtendremos una descripción general del módulo:

```
Help on module modulo:

NAME
    modulo - Descripción del módulo

CLASSES
    builtins.object
        UnaClase

    class UnaClase(builtins.object)
     |  Descripción de la clase
     |
     |  Methods defined here:
     |
     |  un_metodo(self)
     |      Descripción de un método
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors defined here:
     |
     |  __dict__
     |      dictionary for instance variables (if defined)
     |
:
```

03

Si queremos salir de la descripción presionamos la tecla q.

Se pueden accesar a la descripción de cada estructura del código utilizando las sentencias:

- help(modulo) # Ver una descripción general del módulo
- help(modulo.UnaClase) # Ver una descripción general de la clase "UnaClase"
- help(modulo.UnaClase.un_metodo) # Ver una descripción general del método "un_metodo" de la clase "UnaClase"
- help(modulo.una_funcion) #Ver la descripción general de una función

04

Generar el html

Para generar un html con la documentacion del codigo temenos que irnos con nuestra terminal a la ubicacion del modulo que deseamos generar el html, posteriormente ejecutar la sentencia: pydoc -w <nombreModulo>

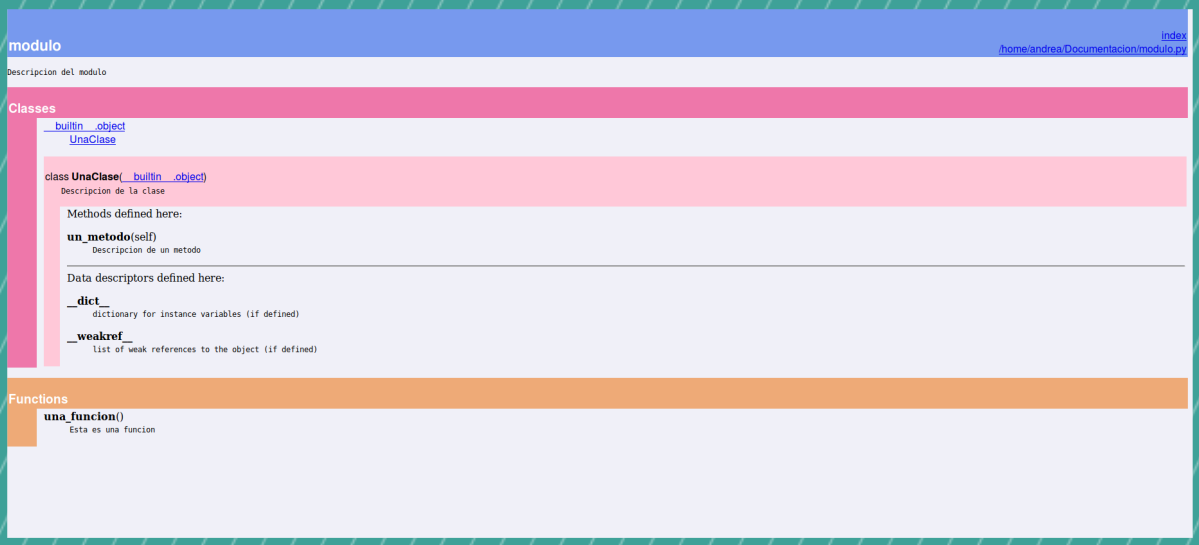
```
andrea@andrea-HP-ENVY-Laptop-13-ad0xx:~/Documentacion$ pydoc -w modulo
wrote modulo.html
andrea@andrea-HP-ENVY-Laptop-13-ad0xx:~/Documentacion$
```

Dicha sentencia nos generará dos archivos uno de ellos será el html y otro de ellos sera un archivo .pyc (Python Compiled File.) que es un archivo compilado del programa original,

05

Abrir el html

Abrimos el html con nuestro navegador (google chrome, firefox, internet explorer) y nos saldrá:



modulo

index
/home/andrea/Documentacion/modulo.py

Descripcion del modulo

Classes

__builtin__.object
UnaClase

class **UnaClase**(__builtin__.object)
Descripcion de la clase

Methods defined here:

un_metodo(self)
Descripcion de un metodo

Data descriptors defined here:

__dict__
dictionary for instance variables (if defined)

__weakref__
list of weak references to the object (if defined)

Functions

una_funcion()
Esta es una funcion

CON ESTOS SENCILLOS PASOS NUESTRO PROGRAMA

YA ESTA DOCUMENTADO