

Sistemas Informàtics

Scripting en Python 03. Guia



Autores: Sergi García, Alfredo Oltra

Actualizado Noviembre 2025



SCRIPTING EN PYTHON - PARTE 03 - GUIA

1. ¿QUÉ HACER?

En esta unidad, vamos a mostrarte cómo ejecutar comandos de Linux/Windows y obtener su salida como una variable. Además, vamos a aprender cómo pasar argumentos utilizando la consola de Linux/Windows.

Para usar los siguientes comandos, debemos utilizar la cláusula “import” con el fin de importar módulos que contienen funciones personalizadas. Si quieres saber más sobre los módulos, visita este enlace: https://www.w3schools.com/python/python_modules.asp.

2. EJECUTAR COMANDOS DE LINUX/WINDOWS

Para ejecutar comandos de Linux/Windows y obtener su salida, puedes usar este enlace.

Un ejemplo:

```
# Importamos el módulo "subprocess", que nos permite ejecutar comandos
import subprocess
# subprocess.check_output ejecuta un comando y obtiene su salida
output = subprocess.check_output("cat /etc/services", shell=True)
```

3. PASAR ARGUMENTOS DESDE LA CONSOLA A NUESTRO PROGRAMA

Para pasar argumentos desde la consola a un programa en Python, debemos importar el módulo **sys** y usar la lista **sys.argv**.

Esta variable contiene un array (lista) donde:

```
sys.argv[0] es siempre el nombre del archivo ejecutado.
sys.argv[1] es el primer argumento pasado por consola.
sys.argv[2] es el segundo argumento.
```

...y así sucesivamente.

Esto permite que un programa reciba datos directamente desde la consola cuando se ejecuta.

```
# Importamos el módulo sys
import sys

# La función len nos dice la longitud de una lista. "sys.argv" es una lista
# que contiene los parámetros recibidos desde la consola.

if len(sys.argv) != 3:
    print("Se requieren 2 parámetros")
else:
    print(sys.argv[0]) # muestra el nombre del archivo ejecutable
    print(sys.argv[1]) # muestra el primer argumento recibido
    print(sys.argv[2]) # muestra el segundo argumento recibido
```

Se importa el módulo sys para poder acceder a sys.argv.

```
sys.argv es una lista con todos los valores escritos al ejecutar el programa.  
len(sys.argv) devuelve cuántos elementos hay en esa lista.
```

Si no hay suficientes argumentos, se muestra un mensaje de error.

Si sí hay suficientes, se imprimen:

```
sys.argv[0]: nombre del script.  
sys.argv[1]: primer argumento.  
sys.argv[2]: segundo argumento.
```

Ejemplo real de ejecución en consola

Supongamos que el archivo se llama **programa.py**.

Ejecución sin argumentos:

```
python programa.py
```

Salida:

```
2 parámetros son requeridos
```

Ejecución con dos argumentos:

```
python programa.py hola adios
```

Salida:

```
programa.py  
hola  
adios
```

Porque:

```
sys.argv[0] = "programa.py"  
sys.argv[1] = "hola"  
sys.argv[2] = "adios"
```

4. OBTENER LA FECHA + HORA ACTUAL EN PYTHON 3

Usando Python, puedes consultar la fecha y hora actual mediante el módulo “datetime” y formatearla de diferentes maneras. Por ejemplo, si quieres imprimir una cadena con la fecha en este formato “2023-02-27-11-00”, puedes usar este código:

```
from datetime import datetime  
# Obtener la fecha y hora actual  
now = datetime.now()  
# Formatear la fecha y hora (Y->año, m->mes, d->día, H->hora y M->minuto)  
formatted_date = now.strftime("%Y-%m-%d-%H-%M")  
# Imprimir la cadena formateada  
print(formatted_date)
```

este código utiliza el módulo “datetime” para obtener la fecha y la hora actuales, y después las formatea según el formato especificado usando “strftime”.

Si quieres saber más sobre los formatos admitidos por “strftime”, puedes visitar este enlace:

<https://strftime.org/>

5. HE TERMINADO DE LEER ESTE DOCUMENTO. ¿QUÉ DEBO HACER AHORA?

Hemos proporcionado varias actividades no evaluables. Están disponibles en el archivo “Scripting en Python - Parte 03 - Actividades 01”. Debes intentar resolverlas y plantear tus dudas en los foros

6. INFORMACIÓN

Puedes encontrar más información y ejemplos en este enlace

https://www.programcreek.com/python/example/2696/subprocess.check_output

<https://queirozf.com/entries/python-3-subprocess-examples#run-example-store-output-and-error-message-in-string>