

MODULO 3 – Capítulo 6

Extra – superbásico Python

Instalación de Python 3 en Windows 11

1. Descargar el instalador

- Ve a la página oficial de Python: <https://www.python.org/downloads/>
- Descarga la última versión estable de Python para Windows.

2. Ejecutar el instalador

- Abre el archivo .exe descargado.
- Marca la casilla "**Add Python to PATH**" (muy importante).
- Haz clic en "**Install Now**".

3. Verificar la instalación

- Abre el **Símbolo del sistema** (cmd) o **PowerShell** y ejecuta:

```
python --version
```

o

```
python3 --version
```

- Si la instalación fue correcta, aparecerá la versión de Python instalada.

4. Configurar Python para ejecutar programas

- Puedes ejecutar scripts de Python escribiendo en la terminal:

```
python nombre_del_script.py
```

- Para trabajar en modo interactivo, simplemente ejecuta:

```
python
```

- También puedes usar el **IDLE** de Python (instalado por defecto) o instalar editores como **VS Code**, **PyCharm** o **Sublime Text**.

Sintaxis Básica en Python

1. Creación de Variables

```
nombre = "Juan"  
edad = 25  
altura = 1.75
```

```
es_mayor = True # Booleano
```

2. Funciones

```
def saludar(nombre):  
    print(f"Hola, {nombre}")  
  
saludar("Carlos")
```

3. Importación de Módulos

```
import math # Importa todo el módulo  
print(math.sqrt(16)) # Raíz cuadrada  
  
from random import randint # Importa solo una función  
print(randint(1, 10)) # Número aleatorio entre 1 y 10
```

4. Estructura Condicional (if)

```
edad = 20  
  
if edad >= 18:  
    print("Eres mayor de edad")  
else:  
    print("Eres menor de edad")
```

5. Bucle while

```
contador = 0  
  
while contador < 5:  
    print(f"Contador: {contador}")  
    contador += 1
```

6. Bucle for

```
for i in range(5): # Itera de 0 a 4  
    print(f"Iteración {i}")  
  
lista = ["Python", "Java", "C++"]  
for lenguaje in lista:  
    print(lenguaje)
```

7. Listas y Diccionarios

```
frutas = ["Manzana", "Banana", "Cereza"]  
print(frutas[0]) # Acceder al primer elemento  
  
persona = {"nombre": "Luis", "edad": 30}  
print(persona["nombre"]) # Acceder a un valor del diccionario
```

Código en Python utilizando el framework `Flask` para crear una API que cumple con las especificaciones indicadas. La aplicación escucha en un puerto dado como argumento y devuelve valores numéricos aleatorios para los endpoints `/usuarios`, `/precio` y `/plazas_libres`. Además, incluye una página de ayuda en la raíz (`/`) que describe la funcionalidad de la API.

Código:

```
# app.py

from flask import Flask, jsonify
import random
import sys

app = Flask(__name__)

# Función para generar números aleatorios dentro de un rango
def get_random_value(low, high):
    return random.randint(low, high)

# Ruta principal: Información de ayuda sobre la API
@app.route('/')
def help():
    return """
    <h1>Bienvenido a la API de Datos Aleatorios</h1>
    <p>Esta API proporciona datos aleatorios para los siguientes endpoints:</p>
    <ul>
        <li><code>/usuarios</code>: Devuelve un número aleatorio entre 0 y 35.</li>
        <li><code>/precio</code>: Devuelve un número aleatorio entre 46 y 57.</li>
        <li><code>/plazas_libres</code>: Devuelve la diferencia entre 35 y el valor de <code>/usuarios</code>.</li>
    </ul>
    <p>Ejemplo de uso: <code>http://localhost:<puerto>/usuarios</code></p>
    """

# Endpoint /usuarios
@app.route('/usuarios', methods=['GET'])
def usuarios():
    usuarios = get_random_value(0, 35)
    return jsonify({"usuarios": usuarios})

# Endpoint /precio
@app.route('/precio', methods=['GET'])
def precio():
    precio = get_random_value(46, 57)
    return jsonify({"precio": precio})
```

```
# Endpoint /plazas_libres
@app.route('/plazas_libres', methods=['GET'])
def plazas_libres():
    usuarios = get_random_value(0, 35)
    plazas_libres = 35 - usuarios
    return jsonify({"plazas_libres": plazas_libres})

if __name__ == '__main__':
    if len(sys.argv) != 2:
        print("Uso: python app.py <puerto>")
        sys.exit(1)

    try:
        port = int(sys.argv[1])
    except ValueError:
        print("El puerto debe ser un número entero.")
        sys.exit(1)

    app.run(host='0.0.0.0', port=port)
```

Explicación del Código:

1. Importaciones :

- `Flask`: Framework para crear la API.
- `random`: Para generar números aleatorios.
- `sys`: Para capturar el puerto como argumento desde la línea de comandos.

2. Función `get_random_value` :

- Genera un número aleatorio dentro de un rango dado.

3. Endpoints :

- `/`: Muestra una página de ayuda con información sobre los endpoints disponibles.
- `/usuarios`: Devuelve un número aleatorio entre 0 y 35.
- `/precio`: Devuelve un número aleatorio entre 46 y 57.
- `/plazas_libres`: Calcula y devuelve la diferencia entre 35 y el valor generado para `/usuarios`.

4. Captura del Puerto :

- El puerto se pasa como argumento al ejecutar el script.
- Se valida que el argumento sea un número entero.

5. Ejecución :

- La aplicación se ejecuta en `0.0.0.0` (todas las interfaces de red) en el puerto especificado.

Ejecución:

Para ejecutar la aplicación, guarda el código en un archivo llamado `app.py` y luego ejecútalo desde la terminal:

```
python app.py 5000
```

Esto hará que la aplicación escuche en el puerto `5000`. Puedes acceder a los endpoints en las siguientes URLs:

- `http://localhost:5000/`: Información de ayuda.
- `http://localhost:5000/usuarios`: Número aleatorio entre 0 y 35.
- `http://localhost:5000/precio`: Número aleatorio entre 46 y 57.
- `http://localhost:5000/plazas_libres`: Diferencia entre 35 y el valor de `/usuarios`.