



Flask

Framework de Python

¿Qué es Flask?

- Framework web creado por el desarrollador austriaco Armin Ronacher en 2010.
 - Flask es un microframework para Python basado en Werkzeug que permite crear aplicaciones web de todo tipo rápidamente.
- Flask solo incluye el motor de plantillas Jinja y una biblioteca llamada “tool”. Sin embargo, ofrece la posibilidad de integrar funciones de terceros.
- Flask está bajo una licencia BSD. Es gratuito y de código abierto.

Primera aplicación Flask

- Preparando el entorno de programación

```
[kralos]--[main = ● ]
[D:\umar\Asignaturas\Semestre 23-24 B\612_TW_1\Repo_GitHub\Notas_TW1\code\flask\project_0]
• ▶ python -m virtualenv venv
created virtual environment CPython3.12.3.final.0-64 in 14983ms
creator CPython3Windows(dest=D:\umar\Asignaturas\Semestre 23-24 B\612_TW_1\Repo_GitHub\Notas_TW1\code\flask\project_0\venv, clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip=copy, app_data_dir=C:\Users\kralos\AppData\Local\pypa\virtualenv)
added seed packages: pip==24.0
activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
[kralos]--[main = ● ]
[D:\umar\Asignaturas\Semestre 23-24 B\612_TW_1\Repo_GitHub\Notas_TW1\code\flask\project_0]
• ▶ .\venv\Scripts\activate
(venv) r[kralos]--[main = ● ]
[D:\umar\Asignaturas\Semestre 23-24 B\612_TW_1\Repo_GitHub\Notas_TW1\code\flask\project_0]
• ▶ pip install flask
Collecting flask
  Downloading flask-3.0.3-py3-none-any.whl.metadata (3.2 kB)
Collecting Werkzeug>=3.0.0 (from flask)
  Downloading werkzeug-3.0.2-py3-none-any.whl.metadata (4.1 kB)
Collecting Jinja2>=3.1.2 (from flask)
  Downloading Jinja2-3.1.3-py3-none-any.whl.metadata (3.3 kB)
Collecting itsdangerous>=2.1.2 (from flask)
  Downloading itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from flask)
  Downloading click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting blinker>=1.6.2 (from flask)
  Downloading blinker-1.7.0-py3-none-any.whl.metadata (1.9 kB)
Collecting colorama (from click>=8.1.3->flask)
  Downloading colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Collecting MarkupSafe>=2.0 (from Jinja2>=3.1.2->flask)
  Downloading MarkupSafe-2.1.5-cp312-cp312-win_amd64.whl.metadata (3.1 kB)
Download flask-3.0.3-py3-none-any.whl (101 kB)
101.7/101.7 kB 973.4 kB/s eta 0:00:00
Download blinker-1.7.0-py3-none-any.whl (13 kB)
Download click-8.1.7-py3-none-any.whl (97 kB)
97.9/97.9 kB 1.1 MB/s eta 0:00:00
Download itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Download Jinja2-3.1.3-py3-none-any.whl (133 kB)
133.2/133.2 kB 1.6 MB/s eta 0:00:00
Download werkzeug-3.0.2-py3-none-any.whl (226 kB)
226.8/226.8 kB 1.5 MB/s eta 0:00:00
Download MarkupSafe-2.1.5-cp312-cp312-win_amd64.whl (17 kB)
Using cached colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: MarkupSafe, itsdangerous, colorama, blinker, Werkzeug, Jinja2, click, flask
Successfully installed Jinja2-3.1.3 MarkupSafe-2.1.5 Werkzeug-3.0.2 blinker-1.7.0 click-8.1.7 colorama-0.4.6 flask-3.0.3 itsdangerous-2.2.0
(venv) r[kralos]--[main = ● ]
[D:\umar\Asignaturas\Semestre 23-24 B\612_TW_1\Repo_GitHub\Notas_TW1\code\flask\project_0]
▶
```

Primera aplicación Flask

- Preparando el entorno de programación

```
• ▶ pip freeze  
blinker==1.7.0  
click==8.1.7  
colorama==0.4.6  
Flask==3.0.3  
itsdangerous==2.2.0  
Jinja2==3.1.3  
MarkupSafe==2.1.5  
Werkzeug==3.0.2
```

Primera aplicación Flask

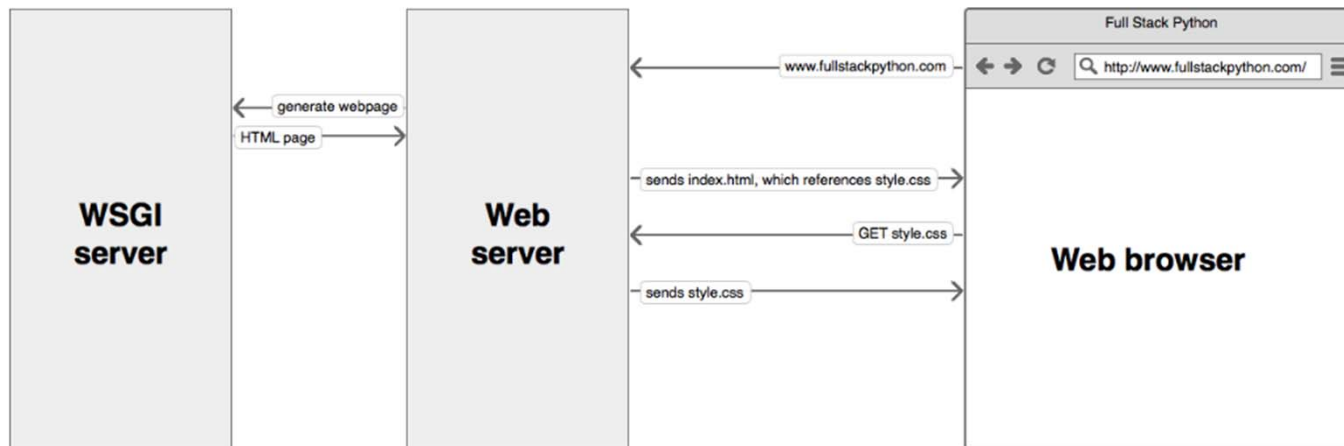
- Creando la primera aplicación Flask

```
1. from flask import Flask
2. app = Flask(__name__)
3.
4. @app.route('/')
5. def hello_world():
6.     return 'Hello, World!'
```

- Toda aplicación Flask es una instancia WSGI de la clase Flask. Por tanto, importamos dicha clase y creamos una instancia que en este caso he llamado app.

Primera aplicación Flask

- ¿WSGI?
- Significa Web Server Gateway Interface, y es una especificación estándar sobre cómo deben interactuar los servidores web y las aplicaciones web de Python



Primera aplicación Flask

- Una característica de Flask es que tendremos métodos asociados a las distintas URLs que componen nuestra aplicación.
- Es en estos métodos donde ocurre toda la lógica que queramos implementar. Dentro del patrón MVC, esta parte del código se correspondería con el controlador.
- Flask se encarga de hacernos transparente el cómo a partir de una petición a una URL se ejecuta finalmente nuestra rutina.
- Lo único que tendremos que hacer nosotros será añadir un decorador a nuestra función. En nuestro caso, hemos llamado a nuestra función `hello_world` que será invocada cada vez que se haga una petición a la URL raíz de nuestra aplicación.

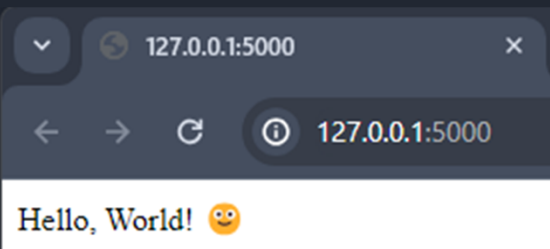
Primera aplicación Flask

- El decorador route de la aplicación (app) es el encargado de decirle a Flask qué URL debe ejecutar su correspondiente función.
- El nombre que le demos a nuestra función será usado para generar internamente URLs a partir de dicha función.
- La función debe devolver la respuesta que será mostrada en el navegador del usuario.

Primera aplicación Flask

- Probando la aplicación Flask
- Flask viene con un servidor interno que nos facilita mucho la fase de desarrollo. **No debemos usar este servidor en un entorno de producción ya que no es este su objetivo.**

```
(venv) r[kralos]--[!main ≡ ● ]
[D:\umar\Asignaturas\Semestre 23-24 B\612_TW_1\Repo_GitHub\Notas_TW1\code\flask\project_0]
▶ python -m flask run
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [25/Apr/2024 10:17:27] "GET / HTTP/1.1" 200 -
```



Referencias

- j2logo. (2019, February 25). *Tutorial de Flask en español: Desarrollando una aplicación web en Python*. J2LOGO. <https://j2logo.com/tutorial-flask-espanol/>
- *¿Qué es Flask Python? Un breve tutorial sobre este microframework*. (2023, March 1). IONOS Digital Guide; IONOS. <https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/flask/>