


Semana 4.

Flask – Introducción

SISTEMAS EMBEBIDOS 
(EMBEDED SYSTEMS)

Grado Dual en Industria Digital

Campus Vitoria

Curso 2020-2021



Deusto

Facultad de Ingeniería
Ingeniaritza Fakultatea

¿Qué es Flask?

microframework de python para crear aplicaciones web, es decir, páginas web dinámicas, APIs, etc ...

¿Por qué es micro?

No require librerías o herramientas especiales. No tiene capa de abstracción para bases de datos, “form validation”, ni otros components where pre-existing third-party libraries provide common functions.



Flask vs Django

Flask vs Django: objetivos

- Django: Crear menos código, components reusables, y desarrollo rápido
- Flask:
 - ✓ Hacer una única tarea, pero hacerla bien.
 - ✓ Herramientas muy básicas de desarrollo
 - ✓ Las aplicaciones FLASK comunmente son Single-Page-Applicaciones (SPAs) -> Aplicaciones de una única página

Flask vs Django: ¿cuando usarlo?

- Django:
 - Diseñado para desarrollo rápido de aplicaciones web **complejas**
 - Proporciona a los desarrolladores únicamente las herramientas necesarias para implementar funcionalidades escalables y mantenibles
- Flask:
 - Mucho más simple que Django
 - Permite a los desarrolladores crear **pequeñas aplicaciones web más rápido**

My primera Flask App

Puede ser tan simple com un fichero python

(hola.py)

```
from flask import
```

Importar la case Flask

```
Flask app = Flask(__name__)
```

Crear instancia de la clase Flask y dar nombre al modulo de nuestra aplicación web

```
@app.route('/')
```

Le decimos a Flask que URL debe llamar a la función definida a contuación

```
def hello_world():
```

```
    return 'Hola Flask!'
```

Esto se mostrará en el navegador de Internet

Ejecutar my primera Flask App

Puede ser tan simple com un fichero python

(hola.py)

```
from flask import
Flask app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hola Flask!'
```



*Dentro del directorio donde temenos guardado hola.py
Usar la Ventana de comandos de Linux*

```
$ export FLASK_ENV=development
(estó es para entrar en modo "debug"
Y ver los errores, si los hay, en el navegador)
```

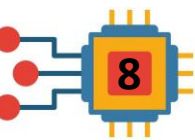
```
$ export FLASK_APP=hola.py
```

```
$ flask run
```

```
$ python3 -m flask run
```

Son equivalentes

Abrir el navegador de internet e introducir la URL
<http://127.0.0.1:5000/>



“Routing”

- Las aplicaciones Web modernas usan URLs significativos para ayudar a los usuarios
- Usa el “decorador” `route()` para **asociar una URL a una función en python.**

```
@app.route('/')  
def index():  
    return 'Página Index'
```

→ `http://127.0.0.1:5000/`



```
@app.route('/sensor1')  
def hello():  
    return 'Página Sensor 1'
```

→ `http://127.0.0.1:5000/sensor1`



Rutas con variables

- Puedes añadir variables a la URL con la partícula `<variable_name>`
- Tu función recibe la partícula `<variable_name>` como argumento
- Opcionalmente, puedes usar un convertidor para especificar el tipo del argumento, como `<converter:variable_name>`.

<code>string</code>	(default) accepts any text without a slash
<code>int</code>	accepts positive integers
<code>float</code>	accepts positive floating point values
<code>path</code>	like <code>string</code> but also accepts slashes
<code>uuid</code>	accepts UUID strings

```
@app.route('/user/<username>')
def show_user_profile(username):
    return 'User %s' % escape(username)
```

`http://127.0.0.1:5000/user/laura`



```
@app.route('/post/<int:post_id>')
def show_post(post_id):
    return 'Post %d' % post_id
```

`http://127.0.0.1:5000/post/3`



Notas importantes

- Se “debe” reiniciar la terminal para volver a lanzar la App
- Funciones asociadas a las routes (URLS) deben tener nombres únicos en la aplicación
- Modo debug: `$ export FLASK_ENV=development`
- Se debe estar dentro del directorio donde se tiene el fichero .py de la aplicación para iniciar la aplicación (`$flask run`)

Notas importantes

- Se “~~debe~~” puede reiniciar la terminal para volver a lanzar
- Alternativa: terminar el proceso (recordar semanas anteriores)

1. Ver el PID del proceso

`$ps -fA | grep python`

```
laura@DESKTOP-3BRHMF8:~/flask$ ps -fA | grep python
laura      220      150    0 10:09 pts/0    00:00:00 /usr/bin/python3 /home/laura/.local/bin/flask run
laura      226      150    0 10:09 pts/0    00:00:00 grep --color=auto python
```

2. Eliminar (kill) el proceso a partir de su PID (Process ID)

`$kill -9 220`