Curso de Flask

# Flask es sencillo de aprender, tiene una documentación clara y práctica, es rápido a la hora de renderizar puede ser hasta tres veces más rapido que Django. También es fácil de realizar una API REST, la estructura de un proyecto es flexible y es ideal para aprender desarrollo web con un framework de Python.

**¿Cómo funcionan las aplicaciones en internet?**

Cuando utilizas una aplicación web puedes interactuar con ella desde una computadora hasta un dispositivo móvil, pero esto no quiere decir que consume el procesamiento de tu dispositivo. Todo lo contrario, se hace en una red de servidores.

Estos servidores unen su poder de procesamiento con el fin transmitir solicitudes a todo el mundo, a su vez utilizar servidores especializados para almacenar los datos con los cuales se está trabajando, así como los datos de los demás usuarios. Como todo esto sucede sin demora alguna, parecerá que la aplicación se está ejecutando de forma nativa en tu dispositivo.

El servidor procesa la información obtenida por el navegador, luego se realizan los procedimientos necesarios de acuerdo a la lógica de negocio de la aplicación para regresar la información solicitada al cliente.

**Ejemplo:**

Cuando utilizamos **Google Drive** el cual es una aplicación web y abrimos un documento con Google Docs, el navegador se comunica con los servidores para ver y editar el documento.

A medida que vayas editando el documento, tu navegador trabajará de la mano con los servidores para asegurarse que todos los cambios se estén guardando.

**Ventajas:**

* Muchas de las aplicaciones web que existen son gratuitas.
* Puedes acceder a tu información en cualquier momento y lugar.
* No dependes de un dispositivo en específico ya que la aplicación se encuentra almacenada en la web.

En esta clase el profesor Bernardo Cassina nos explica cómo podemos usar Flask para desarrollar aplicaciones web escritas en Python con este framework.

Flask es un framework minimalista escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo de líneas de código, busca que su infraestructura inicial sea lo más simple posible y pueda personalizarse fácilmente, puedes extender sus funcionalidades con las llamadas Flask Extensions <http://flask.pocoo.org/extensions/>

### Esta es la guía para configurar nuestro ambiente con Python 3.

Por lo general Mac ya incluye una instalación de Python, la puedes verificar ejecutando los siguientes comandos en una terminal:

python --version

python3 --version

Debemos asegurarnos de tener **python 3**. Para instalar Python puedes seguir el siguiente enlace y después regresar a esta lectura.

<https://platzi.com/clases/1378-python/14289-guia-de-instalacion-y-conceptos-basicos/>

### Instalación en Windows

Una vez que instalaste **python 3** desde [python.org](https://www.python.org/" \t "_blank) vamos a verificar que también incluimos **pip** en esta instalación. Después debes correr el siguiente comando para instalar virtualenv:

pip install virtualenv

El sistema debe haber instalado virtualenv y ahora podemos comenzar con el curso.

### Instalación en Mac

Si ya instalaste python 3 ahora corre el siguiente comando para instalar **pip**:

sudo easy\_install pip

Para install virtualenv de manera global corre:

pip install virtualenv

El sistema debe haber instalado virtualenv y ahora podemos comenzar con el curso.

**Hello World Flask**

Estos son los conceptos principales que debes entender antes de hacer un Hello World en Flask:

* **virtualenv:** es una herramienta para crear entornos aislados de Python.
* **pip:** es el instalador de paquetes para Python.
* **requirements.txt:** es el archivo en donde se colocará todas las dependencias a instalar en nuestra aplicación.
* **FLASK\_APP:** es la variable para identificar el archivo donde se encuentra la aplicación.

**Crear un nuevo entorno virtual**



**Activar el nuevo entorno virtual**



Desactivar el entorno virtual



Instalar Flask



Ver las dependencias que se instalaron con Flask



Graphical user interface, text, application

Description automatically generated

Creamos nuestro archivo requirements.txt



Graphical user interface, text, application

Description automatically generated

Comando recursivo para instalar las dependencias que coloquemos dentro de requirements.txt



Creamos nuestro archive main.py



Agregamos la variable de entorno que indica nuestro archivo de entrada



Revisamos que si se guardara



Corremos nuestro servidor

Text

Description automatically generated with low confidence

Hello World!

**Debugging en Flask**

**Debugging:** es el proceso de identificar y corregir errores de programación.

Para activar el *debug mode* escribir lo siguiente en la consola:

export FLASK\_DEBUG=1

echo $FLASK\_DEBUG

Logging: es una grabación secuencial en un archivo o en una base de datos de todos los eventos que afectan a un proceso particular.

Se utiliza en muchos casos distintos, para guardar información sobre la actividad de sistemas variados.

Tal vez su uso más inmediato a nuestras actividades como desarrolladores web sería el logging de accesos al servidor web, que analizado da información del tráfico de nuestro sitio. Cualquier servidor web dispone de logs con los accesos, pero además, suelen disponer de otros logs, por ejemplo, de errores.

Los sistemas operativos también suelen trabajar con logs, por ejemplo para guardar incidencias, errores, accesos de usuarios, etc.

A través de el logs se puede encontrar información para detectar posibles problemas en caso de que no funcione algún sistema como debiera o se haya producido una incidencia de seguridad.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

**Request y Response**

Text

Description automatically generated

Text, letter

Description automatically generated

<https://www.tutorialspoint.com/flask/flask_request_object.htm>

**Ciclos de Request y Response**

**Request-Response:** es uno de los métodos básicos que usan las computadoras para comunicarse entre sí, en el que la primera computadora envía una solicitud de algunos datos y la segunda responde a la solicitud.

Por lo general, hay una serie de intercambios de este tipo hasta que se envía el mensaje completo.

**Por ejemplo:** navegar por una página web es un ejemplo de comunicación de *request-response*.

*Request-response* se puede ver como una llamada telefónica, en la que se llama a alguien y responde a la llamada; es decir hacemos una petición y recibimos una respuesta.

Text

Description automatically generated

Text

Description automatically generated

https://youtu.be/QFrUTDfYgnM