State of the art Django

Emmanuel Alfaro Brenes
Isaac Mena López
Fernando Rojas Salas

Concepto

Es un framework de python para desarrollo web disponible desde 2005

¿Cuándo usarlo?

Se recomienda para desarrollo rápido, ágil, escalable y rico en características. Para grandes cantidades de usuarios.

¿Por qué Django?

Cuenta con más de una década en uso. Aunado a que existe una comunidad muy grande y activa y que dispone más de 10 000 paquetes disponibles para su uso, lo convierte en una opción a considerar.

Ventajas

- No requiere SQL. Pues ofrece manejo de datos al estilo python. Ofrece operaciones CRUD. Ofrece manejo de plantillas. Ofrece seguridad contra inyección de código por defecto. Ofrece funcionalidad para manejo de formularios, autenticación, serialización de datos y otras muchas tareas muy comunes

- Es Open Source.

- Su diseño está implementado y basado en Python lo cual permite claridad, simplicidad, portabilidad y velocidad a la hora de diseñar e implementar el proyecto web.

- Django viene con un ORM (Object Relational Mapping) el cual permite generar menos cantidad de código y más claro a la hora intercambiar datos entre una aplicación orientada a objetos y una base de datos.

- Para su instalación se requiere de **un** comando utilizando el gestor de paquetes de Python.

- Provee una capa de abstracción (Modelos) que para estructurar y manejar los datos de la aplicación web.

- Maneja los conceptos de View el cual encapsula la lógica encargada de manejar los **requests** del usuario y retornar los debidos **responses**.

- Provee una “vista de administrador” ya incorporada y totalmente personalizable.

Desventajas

- ➢ El uso de ORM muchas veces se relaciona con un pobre diseño de base de datos.
- ➢ No recomendable para proyecto pequeños
- ➢ Diseño algo monolítico
- ➢ Sólo un request por proceso a la vez
- ➢ EspecifIcación de URL's algo compleja usando regex
- ➢ Avanza lento debido a su edad y que mantiene compatibilidad hacia atrás

¿Desde cuándo se usa?

Creado en 2003 por Adrian Holovaty y Simon Willison. Su primer versión (0.90) fue liberada en Noviembre de 2005. Desde ese momento comienza su popularización incremental.

Comparación con otras herramientas

Django vs Flask
Comparando con Django, Flask es: más ligero, menos sobrecargado, más flexible, menos enfocado fuertemente en bases de datos SQL.

Django vs Ruby on Rails
Ruby on Rails tiene una curva de aprendizaje más fuerte, es muy complejo crear un API usando RoR, por otro lado Django tiene ya un framework REST para la creación y redirección de APIS.

Una ventaja de RoR es que existen muchos plugins de terceros (Ruby gems) los cuales están bien documentados por lo general. Además la migración es relativamente sencilla mediante RoR. Para aplicaciones con lógica de tests compleja, el ambiente de pruebas de RoR es muy útil.

Django provee muy buena y amplia documentación al contrario de RoR, que cuesta encontrar buena documentación.

Django viene con un portal de administración, el cual es un panel integrado para el manejo sencillo del back end mediante una interfaz gráfica, la cual es muy sencillo de personalizar.

Guía de configuración

```
$pip install django
```

Referencias

- https://www.ibm.com/cloud/learn/django-explained
- https://hackr.io/blog/what-is-django-advantages-and-disadvantages-of-using-django
- https://datafloq.com/read/advantages-and-disadvantages-of-using-django/3050
- https://www.probytes.net/blog/advantages-disadvantages-django/
- https://docs.djangoproject.com/en/2.2/
- https://www.educba.com/django-vs-flask/