TEXT CLASS REVIEW

TEMAS A TRATAR EN EL CUE

- ¿Qué es Django?
- ¿Por qué usar Django?
- Flexibilidad de instalación y configuración.
- El entorno de desarrollo Django.
- Entorno desarrollo v/s producción.
- Django v/s Python.
- Django y la estructura Web para el desarrollo.
- Soporte para bases de datos.
- Python y los entornos virtuales.
- ¿Por qué usar entornos virtuales?
- El entorno Virtual.
- Velocidad de Desarrollo.
- Estructura minimalista.
- Estructura flexible.

¿QUÉ ES DJANGO?

Es un framework web de alto nivel que fue creado en el 2005, como un proyecto realizado por Adrián Holovaty y Simón Willison en el Periódico Journal-World en Lawrence, Kansas, en los Estados Unidos. Éste permite el desarrollo rápido de sitios web seguros y mantenibles. Es gratuito y de código abierto, lo que significa que no está sujeto a ninguna plataforma en particular, y puede ejecutar sus aplicaciones en numerosas distribuciones de Linux, Windows y Mac OS. Aunado a ello, está escrito en Python teniendo una comunidad próspera y activa.

¿POR QUÉ USAR DJANGO?

Existen variadas razones. A continuación las más relevantes:

- Django es el único framework que, "por defecto", viene con un sistema de administración activo, ya listo para ser utilizado sin ningún tipo de configuración. El software no comienza desde cero, sino que incorpora módulos ya implementados, los cuales fueron pensados con buenas prácticas de desarrollo.
- Django es un framework web de alto nivel escrito en Python. Gracias a ello, hereda todas sus características y facilidades; entre éstas, escribir código bastante fácil de entender, y sobre todo, permite desarrollar aplicaciones muy rápidas y potentes.
- Su ORM (Object Relational Mapping, o manejadores de objetos relacionales), permite interactuar con una base de datos de manera muy práctica, y la amplia selección de módulos y librerías, tanto oficiales como provistas por la comunidad, hace posible incorporar todo tipo de funcionalidades, tal como proveer una API.
- Versátil: Django puede ser (y ha sido) usado para construir casi cualquier tipo de sitio web (desde sistemas manejadores de contenidos y wikis, hasta redes sociales y sitios de noticias). Puede funcionar con cualquier framework en el lado del cliente, y también devolver contenido en casi cualquier formato (incluyendo HTML, RSS feeds, JSON, XML, entre otros).
- Django permite protección contra algunas vulnerabilidades de forma predeterminada, incluidos: la inyección SQL, los scripts entre sitios, la falsificación de solicitudes entre sitios, y el clickjacking Escalable.
- Django usa un componente basado en la arquitectura "shared-nothing" (cada parte de la arquitectura es independiente de las otras). Teniendo en cuenta una clara separación entre las diferentes partes, esto significa que puede escalar para aumentar el tráfico al agregar hardware en cualquier nivel: servidores de caché, servidores de bases de datos, o servidores de aplicación. Algunos de los sitios más concurridos han escalado a Django para satisfacer sus demandas; por ejemplo: Instagram y Disqus.
- Mantenible: el código de Django está escrito usando principios y patrones de diseño para fomentar la creación de código mantenible y reutilizable. En particular, utiliza el principio No te repitas "Don't Repeat Yourself" (DRY), para que no exista una duplicación innecesaria, reduciendo la cantidad de código.

FLEXIBILIDAD DE INSTALACIÓN Y CONFIGURACIÓN

Django es extremadamente flexible en términos de cómo y dónde puede instalarse y configurarse.

Puede ser:

- Instalado en diferentes sistemas operativos.
- Usado con Python 3 y Python 2.
- Instalado desde las fuentes, desde el Python Package Index (PyPi), y en muchos casos desde la aplicación de gestión de paquetes de la computadora.
- Configurado para usar una de entre varias bases de datos, las cuales también pueden necesitar ser instaladas y configuradas por separado.
- Ejecutado en el entorno Python del sistema principal, o dentro de entornos virtuales de Python separados.

Las aplicaciones web Django pueden ejecutarse en casi cualquier máquina donde pueda funcionar el lenguaje de programación Python: Windows, Mac OS X, Linux/Unix, Solaris, entre otros. Casi cualquier computadora debería tener el rendimiento necesario para ejecutar Django durante el desarrollo.

EL ENTORNO DE DESARROLLO DJANGO

Es una instalación de Django en la computadora local que puede ser usada para desarrollar y probar aplicaciones Django antes de desplegarlas al entorno de producción, por ejemplo:

```
$ python manage.py runserver
```

Este entorno es similar a la forma de un servidor, pero no puede admitir el acceso a la red externa o el acceso simultáneo de varias personas. Por lo tanto, el sitio web debe implementarse en un servidor, Django se ejecuta localmente con runserver, y su implementación en el servidor conforma adecuar diferentes archivos de configuración.

ENTORNO DE DESARROLLO V/S PRODUCCIÓN

El entorno de desarrollo es donde se crean las aplicaciones, y suele contener configuraciones que les ayudan en el proceso. Por ejemplo: cuando tenemos el DEBUG en True, pues cualquier error que surja, nos mostrará en el frontend para facilitarnos la depuración del mismo.

En cambio, el entorno de producción es el que está dirigido al usuario final, es decir, el lugar donde correrá el código una vez está funcionando públicamente. Aquí no tendría mucho sentido activar el DEBUG, ya que sería una completa falla de seguridad, pues cualquiera podría provocar un fallo y analizar el registro para recopilar información privada.

DJANGO V/S PYTHON

La principal diferencia entre estos es que Python es un lenguaje de programación, y Django es un marco de desarrollo Web de Python (framework).

En el siguiente cuadro se visualizan algunas otras diferencias:

Django	Python
Es un marco web.	Es un lenguaje de programación.
Está desarrollado por Django Software Foundation.	Está desarrollado por Python Software Foundation
Fue lanzado en 2005.	Fue lanzado en 1991.
Está escrito en Python.	También está escrito en lenguaje C, pero la implementación predeterminada se llama CPython.

Se utiliza para el desarrollo web.	Se utiliza para desarrollar marcos como: Django, Flask, Análisis de datos, A.I, entre otros.
Básicamente, es un marco MVT (Plantilla de vista de modelo) construido sobre Python.	Básicamente, es un lenguaje de programación interpretado, interactivo, orientado a objetos, y de alto nivel que se ejecuta en el nivel del compilador.
Se utiliza principalmente en aplicaciones y servidores basados en web.	Se utiliza para crear una aplicación web, análisis de datos, desarrollo de software de inteligencia artificial, entre otros.
Django Software Foundation.	Python Software Foundation.
www.djangoproject.com	www.python.org

DJANGO Y LA ESTRUCTURA WEB PARA EL DESARROLLO

Django es un framework para el desarrollo web, minimalista (estructura básica) y muy potente, que cubre todos los componentes necesarios para realizar un proyecto web con calidad. Además, ofrece un panel administrativo con la posibilidad de ser personalizado, entregando toda estructura y comandos por terminal para generar nuevos módulos del sistema.

Para entender la estructura Web, es necesario conocer el código de instalación de Django y sus requisitos:

- Tener instalado python3.
- Tener instalado pip (manejador de paquetes de Python).
- Instalar el paquete para generar ambientes virtuales de Python (necesario solo para Linux).

SOPORTE PARA BASE DE DATOS

Django apoya oficialmente las siguientes bases de datos:

- PostgreSQL.
- MariaDB.
- MySQL.
- Oracle.
- SQLite.

También hay una serie de backends de bases de datos proporcionados por terceros. Django intenta admitir tantas funciones, como sea posible, en todos los servidores de base de datos. Sin embargo, no todos los backends de la base de datos son iguales.

PYTHON Y LOS ENTORNOS VIRTUALES

Python creó los entornos virtuales para aislar el entorno de un proyecto de otro, y que así cada uno tenga sus propios paquetes o librerías, independientes de los demás.

Un entorno virtual es un espacio aislado del resto del Sistema Operativo, donde se tendrá una serie de dependencias instaladas de manera local. Es como si se especificara un lugar donde Python toma sus librerías; en lugar del predeterminado que usa el Sistema Operativo en el que se está trabajando. Estas dependencias son independientes de las que se tengan previamente instaladas en el Sistema Operativo, y se pueden tener tantos de estos espacios aislados como se desee.

¿POR QUÉ USAR ENTORNOS VIRTUALES?

Existen varias razones para ello, las más resaltantes son:

- Porque utilizando entornos virtuales conseguimos que las dependencias entre paquetes que necesita nuestra aplicación estén satisfechas en todos los entornos, y además es muy sencillo distribuir la configuración del entorno virtual, así como automatizar su creación entre los distintos miembros del equipo de trabajo, consiguiendo que todos trabajen bajo el mismo escenario.

- Porque en lenguajes como Python son cada vez más rápidos, lo que puede suponer que en una misma máquina se tengan aplicaciones que utilicen diferentes dependencias y versiones de un mismo paquete. Por ejemplo: tener dos aplicaciones web en producción, una que esté desarrollada con Django 1.8, y otra con Django 1.10.
- Y por último, porque los entornos virtuales son una herramienta que favorecen las nuevas metodologías de trabajo que se denominan DevOps, las cuales tratan de gestionar y automatizar la configuración. Es sencillo distribuir la configuración de nuestro entorno virtual, y automatizar la creación de ellos en diferentes infraestructuras y escenarios.

EL ENTORNO VIRTUAL

Es un directorio que contiene una instalación de Python de una versión en particular, además de unos cuantos paquetes adicionales.

Diferentes aplicaciones pueden entonces usar entornos virtuales diferentes. Para resolver, por ejemplo: una aplicación A puede tener su propio entorno virtual con una versión 1.0 instalada, mientras que una aplicación B tiene otro entorno virtual con una versión 2.0. Si la aplicación B requiere actualizar la librería a la versión 3.0, esto no afectará el entorno virtual de la aplicación A.

VELOCIDAD DE DESARROLLO

Python es un lenguaje que se utiliza comúnmente para crear prototipos y desarrollar aplicaciones rápidamente. Django da la velocidad y la potencia de Python, con muchas características integradas adicionales para ayudar a crear aplicaciones web y APIs mucho más rápido.

Desde el arranque de un proyecto, hasta crear consultas complejas y la implementación de su aplicación, Django lo tiene cubierto. No son sólo las bibliotecas principales de Django las que pueden ayudarlo a crear aplicaciones más rápidamente, éste también tiene miles de plugins con una API común para que pueda ofrecer ciertas expectativas sobre cómo utilizar cualquiera de ellos en el desarrollo del proyecto.

ESTRUCTURA MINIMALISTA

Django cuenta con una estructura minimalista y muy potente, que cubre todos los componentes necesarios para realizar un proyecto web con calidad; reduciendo a sus elementos necesarios. Además, ofrece un panel administrativo con la posibilidad de ser personalizado.

También entrega toda estructura y comandos por terminal para generar nuevos módulos del sistema. Se debe tener claro cómo se utilizan los operadores lógicos, ciclos repetitivos y condicionales.

ESTRUCTURA FLEXIBLE

También podría llamarse estructura Dogmática; ya que puede gestionar cualquier tarea en particular, ofreciendo un soporte para el desarrollo rápido en un dominio específico (resolver problemas de un tipo especial) porque la manera correcta de hacer cualquier cosa está generalmente bien comprendida y documentada, especialmente en su sitio Web: https://www.djangoproject.com/.