# **¡Tu primer proyecto en Django!**

Parte de este capítulo está basado en los tutoriales de Geek Girls Carrots (<http://django.carrots.pl/>).

Parte de este capítulo está basado en el [tutorial django-marcador](http://django-marcador.keimlink.de/) bajo licencia de Creative Commons Attribution-ShareAlike 4.0 internacional. El tutorial de django-marcador tiene derechos de autor de Markus Zapke-Gündemann et al.

¡Vamos a crear un blog sencillo!

El primer paso es iniciar un nuevo proyecto de Django. Básicamente, significa que vamos a lanzar unos [scripts](https://es.wikipedia.org/wiki/Script) proporcionados por Django que nos crearán el esqueleto de un proyecto de Django. Son solo un montón de directorios y archivos que usaremos más tarde.

Los nombres de algunos archivos y directorios son muy importantes para Django. No deberías renombrar los archivos que estamos a punto de crear. Moverlos a un lugar diferente tampoco es buena idea. Django necesita mantener una cierta estructura para poder encontrar cosas importantes.

Recuerda ejecutar todo en el virtualenv. Si no ves un prefijo (myvenv) en tu consola necesitas activar tu virtualenv. Explicamos cómo hacerlo en el capítulo de **Instalación de Django** en la sección **Trabajar con virtualenv**. Basta con escribir myvenv\Scripts\activate en Windows o source myvenv/bin/activate en Mac OS X o Linux.

**Create project: OS X or Linux**

En Mac OS X o Linux deberías ejecutar el siguiente comando en la consola. **¡No te olvides de agregar el punto . al final!**:

(myvenv) ~/djangogirls$ django-admin startproject mysite .

El punto . es crucial porque le dice al script que instale Django en el directorio actual (para el cual el punto . sirve de abreviatura)

**Nota** Cuando escribas los comandos de arriba acuérdate de que sólo tienes que escribir la parte que empieza por django-admin. La parte (myvenv) ~/djangogirls$ que mostramos aquí es solo un ejemplo del mensaje que aparecerá en tu línea de comandos.

**Create project: Windows**

En Windows deberías ejecutar el siguiente comando en la consola. **¡No te olvides de agregar el punto . al final!**:

(myvenv) C:\Users\Name\djangogirls> django-admin.exe startproject mysite .

El punto . es crucial porque le dice al script que instale Django en el directorio actual (para el cual el punto . sirve de abreviatura)

**Nota** Cuando escribas los comandos de arriba acuérdate de que sólo tienes que escribir la parte que empieza por django-admin. La parte (myvenv) C:\Users\Name\djangogirls>que mostramos aquí es solo un ejemplo del mensaje que aparecerá en tu línea de comandos.

django-admin es un script que creará los archivos y directorios para ti. Ahora deberías tener una estructura de directorios parecida a esta:

djangogirls  
├───manage.py  
└───mysite  
 settings.py  
 urls.py  
 wsgi.py  
 \_\_init\_\_.py

**Nota**: en la estructura de tus directorios también verás un directorio llamado venv que creamos anteriormente.

manage.py es un script que ayuda con la administración del sitio. Con él podremos (entre otras cosas) iniciar un servidor web en nuestra computadora sin necesidad de instalar nada más.

El archivo settings.py contiene la configuración de tu sitio web.

¿Recuerdas cuando hablamos de una cartera que debía comprobar dónde entregar una carta? El archivo urls.py contiene una lista de los patrones utilizados por urlresolver.

Por ahora vamos a ignorar el resto de archivos porque no los vamos a cambiar. ¡Sólo acuérdate de no borrarlos accidentalmente!

## **Cambiar la configuración**

Vamos a hacer algunos cambios en mysite/settings.py. Abre el archivo usando el editor de código que has instalado anteriormente.

**Nota**: ten en cuenta que settings.py es un archivo regular, como cualquier otro. Puedes abrirlo desde el editor de código, usando las acciones del menú "Archivo -> Abrir". Esto te debería mostrar una ventana en la que puedes navegar hacia tu archivo settings.py y seleccionarlo. Alternativamente, puedes abrir el archivo navegando a la carpeta de djangogirls en tu escritorio y haciendo botón derecho en él. Luego, debes seleccionar tu editor de código de la lista de opciones. Seleccionar el editor de código es importante debido a que puedes tener otros programas instalados que pueden abrir el archivo, pero no te dejarán editarlo.

Sería bueno tener el horario correcto en nuestro sitio web. Ve a la [lista de husos horarios de Wikipedia](https://en.wikipedia.org/wiki/List_of_tz_database_time_zones) y copia tu zona horaria (TZ). (por ejemplo, Europe/Berlin )

En settings.py, encuentra la línea que contiene TIME\_ZONE y modifícala para elegir tu zona horaria. Por ejemplo:

mysite/settings.py

TIME\_ZONE = 'Europe/Berlin'

Un código de lenguaje consiste en el lenguaje (por ejemplo en es para inglés or de para alemán) y en el código del país (por ejemplo, de para alemania o ch para suiza). Querrás agregar esto si quieres que los botones y las notificaciones de Django estén en tu idioma. Así tendrás el botón "Cancel" traducido al idioma definido aquí. [Django viene con un montón de traducciones preparadas](https://docs.djangoproject.com/en/1.11/ref/settings/#language-code).

Cambia el código de lenguaje modificando la siguiente línea:

mysite/settings.py

LANGUAGE\_CODE = 'de-ch'

También necesitaremos agregar una ruta para los archivos estáticos (aprenderemos todo sobre los archivos estáticos y CSS más tarde en este tutorial). Ve hacia abajo hasta el *final* del archivo, y justo por debajo de la línea de STATIC\_URL, agrega una nueva línea con STATIC\_ROOT:

mysite/settings.py

STATIC\_URL = '/static/'  
STATIC\_ROOT = os.path.join(BASE\_DIR, 'static')

Cuando DEBUG está en True y ALLOWED\_HOSTS está vacío, el host es validado entre ['localhost', '127.0.0.1', '[::1]']. Esto no coincidirá con nuestro nombre de host en PythonAnywhere una vez que despleguemos nuestra aplicación, así cambiémoslo a lo siguiente:

mysite/settings.py

ALLOWED\_HOSTS = ['127.0.0.1', '<tu-usuario>.pythonanywhere.com']

## **Configurar una base de datos**

Hay una gran variedad de opciones de bases de datos para almacenar los datos de tu sitio. Utilizaremos la que viene por defecto, sqlite3.

Esta ya está configurado en esta parte de tu archivo mysite/settings.py:

mysite/settings.py

DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.sqlite3',  
 'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),  
 }  
}

Para crear una base de datos para nuestro blog, ejecutemos lo siguiente en la consola: python manage.py migrate (necesitamos estar en el directorio de djangogirls que contiene el archivo manage.py). Si eso va bien, deberías ver algo así:

(myvenv) ~/djangogirls$ python manage.py migrate  
Operations to perform:  
 Apply all migrations: auth, admin, contenttypes, sessions  
Running migrations:  
 Rendering model states... DONE  
 Applying contenttypes.0001\_initial... OK  
 Applying auth.0001\_initial... OK  
 Applying admin.0001\_initial... OK  
 Applying admin.0002\_logentry\_remove\_auto\_add... OK  
 Applying contenttypes.0002\_remove\_content\_type\_name... OK  
 Applying auth.0002\_alter\_permission\_name\_max\_length... OK  
 Applying auth.0003\_alter\_user\_email\_max\_length... OK  
 Applying auth.0004\_alter\_user\_username\_opts... OK  
 Applying auth.0005\_alter\_user\_last\_login\_null... OK  
 Applying auth.0006\_require\_contenttypes\_0002... OK  
 Applying auth.0007\_alter\_validators\_add\_error\_messages... OK  
 Applying sessions.0001\_initial... OK

Y, ¡terminamos! Es hora de iniciar el servidor web y ver si está funcionando nuestro sitio web!

## **Iniciar el servidor web**

Debes estar en el directorio que contiene el archivo manage.py (en la carpeta djangogirls). En la consola, podemos iniciar el servidor web ejecutando python manage.py runserver:

(myvenv) ~/djangogirls$ python manage.py runserver

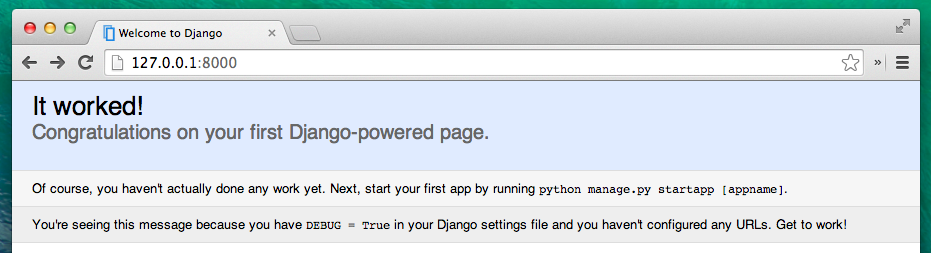
Si estás en Windows y te falla con un error UnicodeDecodeError, utiliza en su lugar este comando:

(myvenv) ~/djangogirls$ python manage.py runserver 0:8000

Ahora todo lo que necesitas hacer es comprobar que tu sitio se esté ejecutando. Abre el navegador (Firefox, Chrome, Safari, Internet Explorer o el que utilices) y escribe la dirección:

http://127.0.0.1:8000/

¡Felicitaciones! ¡Has creado tu primer sitio web y lo has iniciado usando un servidor web! ¿No es genial?



Mientras el servidor web esté corriendo, no verás un nuevo prompt en la línea de comandos para ingresar más comandos. La terminal aceptará nuevo texto ingresado pero no ejecutará los comandos. Esto es porque el servidor web se ejecuta constantemente para escuchar las nuevas peticiones que ingresen.

Hemos visto cómo funcionan los servidores web en el capítulo **¿Cómo funciona internet?**

Para escribir más comandos mientras el servidor web está funcionando, abre una nueva consola y activa el virtualenv. Para parar el servidor web, pasa a la ventana donde se esté ejecutando y pulsa Ctrl+C, las teclas Control y C a la vez (en Windows puede que tengas que pulsar Ctrl+Pausa).

¿Preparado para el próximo paso? ¡Es momento de crear algo de contenido!