

Creating a project

If this is your first time using Django, you'll have to take care of some initial setup. Namely, you'll need to auto-generate some code that establishes a Django [project](#) – a collection of settings for an instance of Django, including database configuration, Django-specific options and application-specific settings.

From the command line, `cd` into a directory where you'd like to store your code, then run the following command:

```
$ django-admin startproject mysite
```

This will create a `mysite` directory in your current directory. If it didn't work, see [Problems running django-admin](#).

Note

You'll need to avoid naming projects after built-in Python or Django components. In particular, this means you should avoid using names like `django` (which will conflict with Django itself) or `test` (which conflicts with a built-in Python package).

Where should this code live?

If your background is in plain old PHP (with no use of modern frameworks), you're probably used to putting code under the web server's document root (in a place such as `/var/www`). With Django, you don't do that. It's not a good idea to put any of this Python code within your web server's document root, because it risks the possibility that people may be able to view your code over the web. That's not good for security.

Put your code in some directory **outside** of the document root, such as `/home/mycode`.

Let's look at what `startproject` created:

```
mysite/
  manage.py
  mysite/
    __init__.py
    settings.py
    urls.py
    asgi.py
    wsgi.py
```

These files are:

- The outer `mysite/` root directory is a container for your project. Its name doesn't matter to Django; you can rename it to anything you like.
- `manage.py`: A command-line utility that lets you interact with this Django project in various ways. You can read all the details about `manage.py` in [django-admin and manage.py](#).
- The inner `mysite/` directory is the actual Python package for your project. Its name is the Python package name you'll need to use to import anything inside it (e.g. `mysite.urls`).
- `mysite/__init__.py`: An empty file that tells Python that this directory should be considered a Python package. If you're a Python beginner, read [more about packages](#) in the official Python docs.
- `mysite/settings.py`: Settings/configuration for this Django project. [Django settings](#) will tell you all about how settings work.
- `mysite/urls.py`: The URL declarations for this Django project; a "table of contents" of your Django-powered site. You can read more about URLs in [URL dispatcher](#).
- `mysite/asgi.py`: An entry-point for ASGI-compatible web servers to serve your project. See [How to deploy with ASGI](#) for more details.
- `mysite/wsgi.py`: An entry-point for WSGI-compatible web servers to serve your project. See [How to deploy with WSGI](#) for more details.

Revision #1

Created 23 February 2023 06:28:58 by Admin

Updated 23 February 2023 06:46:32 by Admin