Author: Aaron Valoroso

Title: How to get started with Django

Date: December 26, 2017

**What is Django:** Django is a free and open source web application framework written in Python. A framework is nothing more than a collection of modules that make development easier. They are grouped together, and allow you to create applications or websites from an existing source, instead of from scratch.

**What is a framework:** is an abstraction in which software providing generic functionality can be selectively changed by additional user-written code, thus providing application-specific software. A software framework provides a standard way to build and deploy applications. Essentially a framework is a code base having specific syntax and ways of doing things.

**Where to get started:** It all depends on what platform you are working from honestly. To make your like easier I would recommend Linux or at least Macintosh. If you don’t have either one then try using a virtual setup. If you are on a Linux or Mac then do the following:

1. Go to the Anaconda website and install their software. This application makes working with Python much easier and installing different python applications.

Go to: <https://www.anaconda.com/download/#macos>

This should install latest version of python or at least a stable version.

1. Type: conda install update
2. Type: conda create --name django-2.0

Let create an environment that we can work in and not mess up our own main python on root.

1. Type: source activate django-2.0
2. Type: conda install pip

I had to use pip in order to install the Django platform.

1. Type: pip install django
2. Type: pip install django update

For the next set create a folder for your website and move into it.

Go to: <https://www.djangoproject.com/download/>

For any download information that I might have missed out on.

Go to: <https://docs.djangoproject.com/en/2.0/intro/tutorial01/>

For creating and working on the website go to the following.

1. Type: django-admin startproject mysite
2. Type: cd mysite
3. Type: python manage.py migrate
4. Type: python manage.py runserver

If you go to <http://127.0.0.1:8000> in any web browser then you should see your application up an running!

**Projects vs. apps**

What’s the difference between a project and an app? An app is a Web application that does something, a Weblog system, a database of public records or a simple poll app. These are also un-pluggable and pluggable within Django. A project is a collection of configuration and apps for a particular website. A project can contain multiple apps. An app can be in multiple projects.

**Current Directory**

mysite/

manage.py

mysite/

\_\_init\_\_.py

settings.py

urls.py

wsgi.py

**Current Files**

* The outer **mysite/** root directory is just 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.
* 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.
* **mysite/settings.py**: Settings/configuration for this Django project.
* **mysite/urls.py**: The URL declarations for this Django project; a “table of contents” of your Django-powered site.
* **mysite/wsgi.py**: An entry-point for WSGI-compatible web servers to serve your project.

**Models:**

A model is the single, definitive source of truth about your data. It contains the essential fields and behaviors of the data you’re storing. Django follows the [DRY Principle](https://docs.djangoproject.com/en/2.0/misc/design-philosophies/#dry). The goal is to define your data model in one place and automatically derive things from it.

**Migrations:**

Migrations are how Django stores changes to your models (and thus your database schema) - they’re just files on disk. You can read the migration for your new model if you like; it’s the file **polls/migrations/0001\_initial.py**. Don’t worry, you’re not expected to read them every time Django makes one, but they’re designed to be human-editable in case you want to manually tweak how Django changes things.

Migrations are very powerful and let you change your models over time, as you develop your project, without the need to delete your database or tables and make new ones - it specializes in upgrading your database live, without losing data. We’ll cover them in more depth in a later part of the tutorial, but for now, remember the three-step guide to making model changes:

* Change your models (in **models.py**).
* Run [**python manage.py makemigrations**](https://docs.djangoproject.com/en/2.0/ref/django-admin/#django-admin-makemigrations) to create migrations for those changes
* Run [**python manage.py migrate**](https://docs.djangoproject.com/en/2.0/ref/django-admin/#django-admin-migrate) to apply those changes to the database.