Guide to Installing Python, Django, and Executing "Runserver"

Installing Python:

Django itself is written purely in Python, so the first step in installing the framework is to make sure you have Python installed.

Installation

If you're on Linux or Mac OS X, you probably have Python already installed. Type `python` at a command prompt (or in Applications/Utilities/Terminal, in OS X). If you see something like this, then Python is installed:

```
Python 2.7.3rc2 (default, Apr 22 2012, 22:30:17)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Otherwise, you'll need to download and install Python. It's fast and easy, and detailed instructions are available at http://www.python.org/download/. Once downloaded add the python interpreter executable to your system path so it's easily accessible from the command line.

Installing Django:

Installing an Official Release

Official releases have a version number, such as 1.4.2, 1.4.1 or 1.4, and the latest one is always available at http://www.djangoproject.com/download/.

On Windows you have to install Django from the tarball. At the previous link download the latest release of the official version. Then follow these instructions:

First download the tarball, which will be named something like `Django-1.4.2.tar.gz`. (It doesn't matter which local directory you download this file into; the installation process will put Django's files in the right place.) Use 7-Zip (http://www.djangoproject.com/r/7zip/) to unzip `.tar.gz` files. Once you've unzipped the file, start up a DOS shell (the "Command Prompt") with administrator privileges and run the following command from within the directory whose name starts with `Django-`:

```
$ python setup.py install
```

In case you're curious: Django's files will be installed into your Python installation's `site-packages` directory – a directory where Python looks for third-party libraries. Usually it's in a place like `/usr/lib/python2.7/site-packages`.

Testing the Django installation:

For some post-installation positive feedback, take a moment to test whether the installation worked. In a command shell, change into another directory (e.g., *not* the directory that contains the `django` directory) and start the Python interactive interpreter by typing `python`. If the installation was successful, you should be able to import the module `django`:

```
>>> import django
>>> django.VERSION
(1, 4, 2, 'final', 0)
```

Running the Development Server:

For some more post-installation positive feedback, let's run the Django development server to see our barebones application in action.

The Django development server (also called the "runserver" after the command that launches it) is a built-in, lightweight Web server you can use while developing your site. It's included with Django so you can develop your site rapidly, without having to deal with configuring your production server (e.g., Apache) until you're ready for production. The development server watches your code and automatically reloads it, making it easy for you to change your code without needing to restart anything.

To start the server, change into your project container directory (`cd data2Dash_django`), if you haven't already, and run this command:

```
python manage.py runserver
```

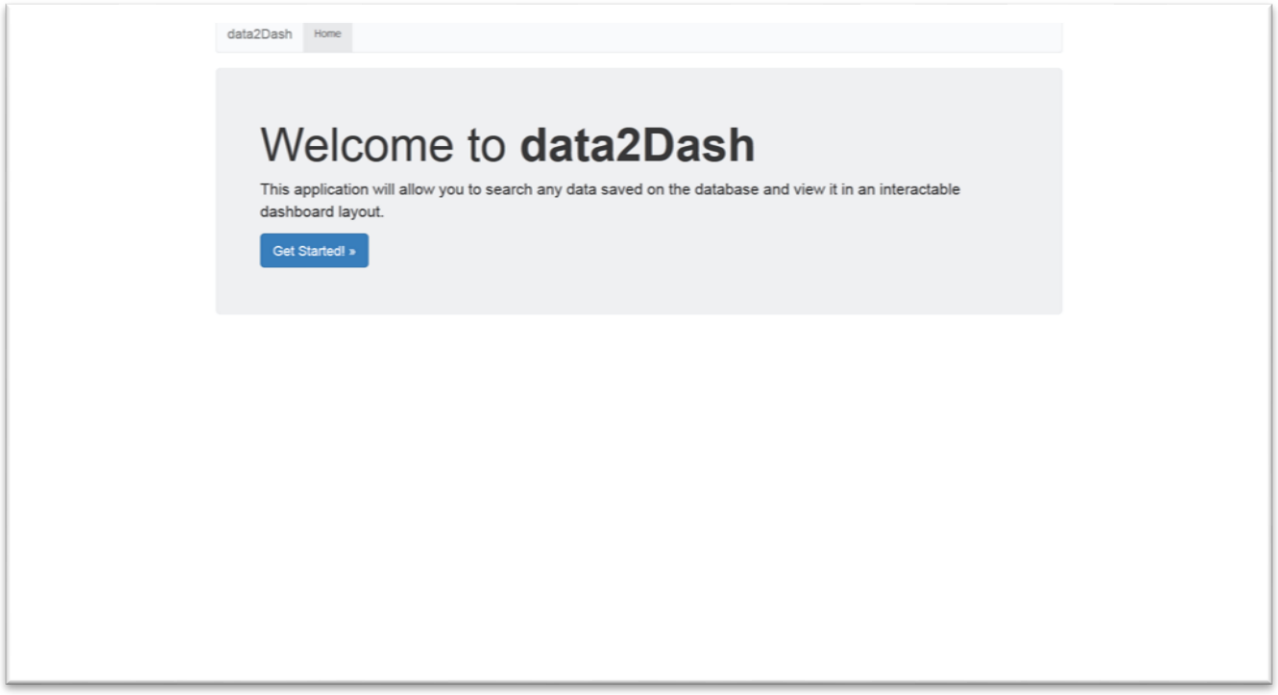
You'll see something like this:

```
Validating models...
0 errors found.

Django version 1.4.2, using settings 'mysite.settings'
Development server is running at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

This launches the server locally, on port 8000, accessible only to connections from your own computer. Now that it's running, visit http://127.0.0.1:8000/ with your Web browser. If you see the home page template then it worked! If not check to make sure your working directory is the same one that contains the projects `manage.py` file (the `runserver` definition is contained here).

Current Homepage Template:



Current Application Dashboard Template: