Here is a crib sheet of the process of creating a Django application from scratch. This is a summary sheet of the commands you've learned up to this point. It's a good idea to be very familiar with this process, and this sheet is useful for a quick reference once you've gotten a good grasp of the concepts they describe.

**Install**

**macOS:**

> virtualenv djangoEnv

> source djangoEnv/bin/activate

(djangoEnv)>pip install django

Copy

**PC:**

> python -m virtualenv djangoEnv

> call djangoEnv/scripts/activate

(djangoEnv)>pip install django

Copy

**Create project**

# 'main' is the project name used below

django-admin startproject main

Copy

**The project has been created, inspect the file & folders created as a result:**

* Folder: main
  + File: manage.py <= used to run the application.
  + Folder: main <= contains the files specific to our app.
    - File: \_\_init\_\_.py
    - File: settings.py
    - File: urls.py
    - File: wsgi.py

**A few more config items to complete the setup:**

* Navigate into the main directory that you just created.

> cd main

Copy

* Make a new apps directory

> mkdir apps

Copy

* Navigate into apps

> cd apps

Copy

* Create an \_\_init\_\_.py file. This just let's Python know that this folder is viewable by Python so that # other Python modules can look in here and access the code!

**Bash: (Mac, Linux) =>**

touch \_\_init\_\_.py

Copy

**PCs: (Command Prompt) =>**

nul> \_\_init\_\_.py <-- You can disregard the Access Denied message

Copy

**Start the new app.**

> python ../manage.py startapp first\_app

Copy

**The app 'first\_app' has been created, lets look at the updated file/folder structure:**

* Folder: main
  + Folder: apps <= created by our steps above
    - File: \_\_init\_\_.py <= created by our steps above
    - Folder: first\_app <= created as a result to the call to 'startapp'
      * File: \_\_init\_\_.py
      * File: admin.py
      * File: apps.py
      * File: models.py <= The 'M' in MVC
      * File: tests.py
      * File: views.py <= The CONTROLLER in our MVC pattern
  + File: manage.py <= used to run the application.
  + Folder: main <= contains the files specific to our app.
    - File: \_\_init\_\_.py
    - File: settings.py
    - File: urls.py
    - File: wsgi.py

Open the settings.py file an add a line to your INSTALLED\_APPS:

INSTALLED\_APPS = [

'apps.first\_app', ### <= added this line!

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

]

Copy

Open the urls.py file, add the line to import include and add a line to include all of the url routes from our app into the main app's url routes.

from django.conf.urls import url, include # Notice we added include

from django.contrib import admin

urlpatterns = [

url(r'^', include('apps.first\_app.urls')) # And now we use include to pull in our first\_app.urls...

]

Copy

Now, create the urls.py file in our 'first\_app' folder so that the final file/folder structure looks like this:

* Folder: main
  + Folder: apps <= created by our steps above
    - File: \_\_init\_\_.py <= created by our steps above
    - Folder: first\_app <= created as a result to the call to 'startapp'
      * File: \_\_init\_\_.py
      * File: admin.py
      * File: apps.py
      * File: models.py <= The 'M' in MVC
      * File: tests.py
      * File: views.py <= The CONTROLLER in our MVC pattern
      * urls.py <= the file that will handle the routes of our app
  + File: manage.py <= used to run the application.
  + Folder: main <= contains the files specific to our app.
    - File: \_\_init\_\_.py
    - File: settings.py
    - File: urls.py
    - File: wsgi.py

**Add some routes to our urls.py file!**

|  |  |
| --- | --- |
| In Django we set up our routes this way:  from django.conf.urls import url  from . import views  # ^ So we can call functions from our routes!  urlpatterns = [  url(r'^$', views.index)  ]    Copy | The Flask equivalent:  @app.route('/')  def index():    Copy |

**Finally, run the app:**

cd ..

python manage.py runserver

Copy

**Adding views & static files**

* Folder: main
  + Folder: apps <= created by our steps above
    - File: \_\_init\_\_.py <= created by our steps above
    - Folder: first\_app <= created as a result to the call to 'startapp'
      * Folder: templates
        + Folder: first\_app

Files: \*.html

* + - * Folder: static
        + Folder: first\_app

Folder: css

Folder: js

Folder: images

* + - * File: \_\_init\_\_.py
      * File: admin.py
      * File: apps.py
      * File: models.py <= The 'M' in MVC
      * File: tests.py
      * File: views.py <= The CONTROLLER in our MVC pattern
      * urls.py <= the file that will handle the routes of our app
  + File: manage.py <= used to run the application.
  + Folder: main <= contains the files specific to our app.
    - File: \_\_init\_\_.py
    - File: settings.py
    - File: urls.py
    - File: wsgi.py

**Example of a Controller action ('View' in Django:)**

# views.py

...

def index(request):

return render(request, "first\_app/index.html")

Copy

**Implement sessions in our app:**

# Need to be in same directory as manage.py file

> python manage.py makemigrations

> python manage.py migrate

**Context**

**On the server:**

from django.shortcuts import render, HttpResponse, redirect

def show\_ninja(request, ninja):

#ninja got passed in through the url parameter!

context = {'myninja' : ninja}

return render(request, '/myproject/showmyninja.html', context)

Copy

**In our templates:**

<!-- From inside /myproject/showmyninja.html -->

{{myninja}}

Copy

**Route Parameters**

**In our urls.py file:**

# Inside apps/appname/urls.py might look like this:

from django.conf.urls import url

from . import views

urlpatterns = [

url(r'^/en/(?P<djangoversion>[0-9]\.[0-9])/topics/http/urls/$', views.index)

]

Copy

**In our views.py file:**

from django.shortcuts import render, HttpResponse, redirect

# Create your views here.

def index(request, djangoversion):

print(djangoversion) # will result in a string e.g. "1.9"

print("hello, I am your first request")

return HttpResponse('hello')