Django Class Notes

Clarusway

Django Intro

What Are Web Frameworks:

A web framework is a software tool that provides a way to build and run web applications. As a result, you don't need to write code on your own and waste time looking for possible miscalculations and bugs.

What is MVT (Model, View, and Template)

A web-framework has a basic model view controller architecture software design pattern for developing web applications, but django is a bit different in a good way. It implements concept of Model-View-Template (MVT). MVT is slightly different from MVC. In fact the main difference between the two patterns is that Django itself takes care of the Controller part (Software Code that controls the interactions between the Model and View), leaving us with the template. The template is a HTML file mixed with Django Template Language (DTL).

Common errors

- If you get error typing `python --version` probably you did not added python to the path during installation. Uninstall python and install again by checking "*add to path*" option.
- For Microsoft Power Shell: For windows Microsoft Tech Support it might be a problem with Execution Policy Settings. To fix it, you should try executing

```
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Unrestricted
```

Create project and app:

- Create a working directory, name it as you wish, cd to new directory
- Create virtual environment as a best practice:

```
python3 -m venv env # for Windows or
python -m venv env # for Windows
virtualenv env # for Mac/Linux or;
virtualenv env -p python3 # for Mac/Linux
```

- Activate scripts:

```
.\env\Scripts\activate  # for Windows
source env/bin/activate  # for MAC/Linux
```

- See the (env) sign before your command prompt.
- Install django:

```
pip install django
```

- See installed packages:

```
pip freeze

# you will see some packages like:
asgiref==3.3.4
Django==3.2.4
pytz==2021.1
sqlparse==0.4.1

# If you see lots of things here, that means there is a problem with your virtual
env activation.
# Activate scripts again
```

- Create requirements.txt same level with working directory, send your installed packages to this file, requirements file must be up to date:

```
pip freeze > requirements.txt
```

- Create project:

```
django-admin startproject project
django-admin startproject project .
# With . it creates a single project folder.
# Avoiding nested folders
# Alternative naming:
django-admin startproject main .
```

- Various files has been created!
- Check your project if it's installed correctly:

```
python3 manage.py runserver  # or,
python manage.py runserver  # or,
py -m manage.py runserver
```

- (Optional) If you have nested project folders with the same name; change the name of the project main (parent) directory as src to distinguish from subfolder with the same name!

```
# optional
mv .\project\ src
```

- Lets create first application:
- Go to the same level with manage.py file:

```
cd .\src\
```

- Start app

```
python manage.py startapp app

# Alternative naming:
python manage.py startapp home
```

Views

- Go to views.py in app directory
- Check your interpreter on VSCode, and choose the one with virtual env
- Create first view by adding:

```

```

```
from django.http import HttpResponse

def home_view(request):
    html = "<html><body>Hello World!</body></html>"
    return HttpResponse(html)
```

- Must include URL path of new app to the project url list.
- Go to urls.py and add:

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include("app.urls")),
]
```

- Create urls.py under app, and add:

```
from django.urls import path
from .views import home_view

urlpatterns = [
    path('', home_view, name="home"),
]
```

- Go to settings.py and add under INSTALLED_APPS:

```
'app.apps.FirstappConfig'  # or
'app'
```

- Run our project:

```
python manage.py runserver
```

- Go to http://localhost:8000 in your browser, and you should see the text “Hello, world.”, which you defined in the index view.

Migrations

- Migrations are Django’s way of propagating changes you make to your models (adding a field, deleting a model, etc.) into your database schema.

- There are several commands which you will use to interact with migrations and Django's handling of database schema:
    - `migrate`, which is responsible for applying and unapplying migrations.
    - `makemigrations`, which is responsible for creating new migrations based on the changes you have made to your models.
- We did not created any model yet. But, we need to migrate default Django tables such as User.

```
python manage.py migrate
```

Login Admin Site

- To login admin site, first create a super user:

```
python manage.py createsuperuser
```

- Enter a usename and password. Pasword wil not be visible on the screen.
- After that, run the server again and go to http://localhost:8000/admin in your browser. Login with usename and password you created.

☺ **Happy Coding!** ✍

Clarusway