

## Django Study 02. Django 프로젝트 파일 뜯어보기.

Django의 프로젝트는 하나의 서비스이다.

Django의 어플리케이션은 프로젝트 안에 각각의 앱으로 여러가지 기능 단위로 나뉜다.

이전 글에서 Django 프로젝트를 생성했으니 이번에는 여기에 들어있는 파일의 구조를 알아보겠다.

```

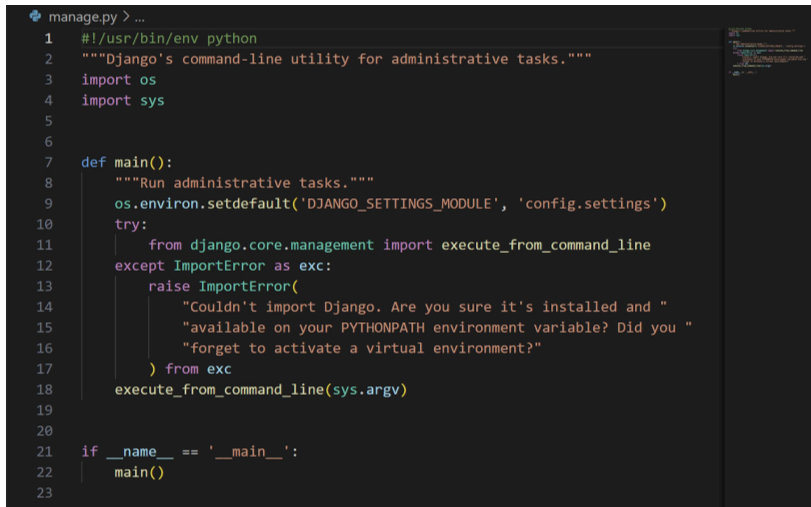
myproject/                                # 프로젝트 루트 디렉토리
├─ manage.py                             # Django 명령어 실행을 위한 스크립트
├─ myproject/                             # 프로젝트 설정 및 라우팅을 담당하는 디렉토리
│   ├── __init__.py                       # Python 패키지 식별을 위한 빈 파일
│   ├── settings.py                       # Django 프로젝트 설정
│   ├── urls.py                           # URL 패턴 정의
│   └─ wsgi.py                            # WSGI 호환 웹 서버 진입점
├─ myapp/                                 # Django 애플리케이션 디렉토리
│   ├── migrations/                      # 데이터베이스 마이그레이션을 위한 디렉토리
│   ├── __init__.py                       # Python 패키지 식별을 위한 빈 파일
│   ├── admin.py                          # Django admin 설정
│   ├── apps.py                           # 애플리케이션 설정
│   ├── models.py                         # 데이터베이스 모델 정의
│   ├── tests.py                          # 테스트 케이스 작성을 위한 파일
│   └─ views.py                           # 뷰 함수 및 클래스 정의
└─ config/                               # Django 프로젝트 설정과 관련된 디렉토리
    ├── __init__.py                       # Python 패키지 식별을 위한 빈 파일
    ├── asgi.py                           # ASGI 호환 웹 서버 진입점
    ├── settings/                         # 설정 파일 분할을 위한 디렉토리
    │   ├── __init__.py                   # Python 패키지 식별을 위한 빈 파일
    │   ├── base.py                       # 기본 설정 파일
    │   ├── development.py                # 개발 환경 설정 파일
    │   └─ production.py                  # 프로덕션 환경 설정 파일
    └─ urls.py                             # 프로젝트 루트 URL 패턴 정의

```

Django의 대표적인, 기본적인 구조는 다음과 같이 나타낼 수 있다.

처음 이 구조를 봤을 때 각각의 기능을 알고싶어 찾아보았다.

## Project



```

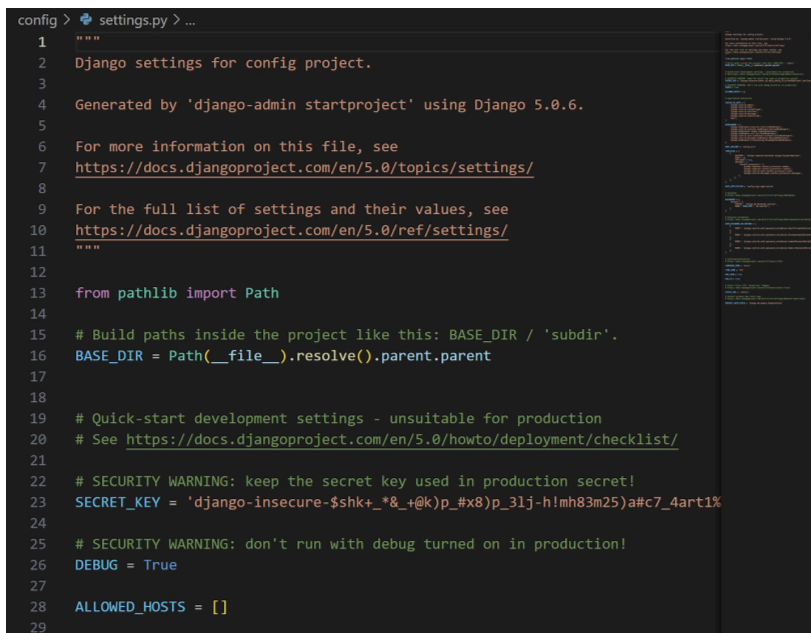
1  #!/usr/bin/env python
2  """Django's command-line utility for administrative tasks."""
3  import os
4  import sys
5
6
7  def main():
8      """Run administrative tasks."""
9      os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'config.settings')
10     try:
11         from django.core.management import execute_from_command_line
12     except ImportError as exc:
13         raise ImportError(
14             "Couldn't import Django. Are you sure it's installed and "
15             "available on your PYTHONPATH environment variable? Did you "
16             "forget to activate a virtual environment?"
17         ) from exc
18     execute_from_command_line(sys.argv)
19
20
21 if __name__ == '__main__':
22     main()
23

```

### manage.py

장고의 다양한 명령어를 실행해 프로젝트와 상호작용을 할 수 있는 유틸리티.

python manage.py <command> [options]



```

1  """
2  Django settings for config project.
3
4  Generated by 'django-admin startproject' using Django 5.0.6.
5
6  For more information on this file, see
7  https://docs.djangoproject.com/en/5.0/topics/settings/
8
9  For the full list of settings and their values, see
10 https://docs.djangoproject.com/en/5.0/ref/settings/
11 """
12
13 from pathlib import Path
14
15 # Build paths inside the project like this: BASE_DIR / 'subdir'.
16 BASE_DIR = Path(__file__).resolve().parent.parent
17
18
19 # Quick-start development settings - unsuitable for production
20 # See https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/
21
22 # SECURITY WARNING: Keep the secret key used in production secret!
23 SECRET_KEY = 'django-insecure-$shk+_8_+@k)p_#x8)p_3lj-h!mh83m25)a#c7_4art1%'
24
25 # SECURITY WARNING: don't run with debug turned on in production!
26 DEBUG = True
27
28 ALLOWED_HOSTS = []
29

```

### settings.py

장고 프로젝트의 설정, config를 관리하는 파일이다.

DB, App, 인증 / (ex. 서버시간, 영한문 설정) 등을 설정할 수 있도록 도움을 준다.

```

config > urls.py > ...
1  """
2  URL configuration for config project.
3
4  The 'urlpatterns' list routes URLs to views. For more information please see
5  https://docs.djangoproject.com/en/5.0/topics/http/urls/
6  Examples:
7  Function views
8      1. Add an import: from my_app import views
9      2. Add a URL to urlpatterns: path('', views.home, name='home')
10 Class-based views
11     1. Add an import: from other_app.views import Home
12     2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13 Including another URLconf
14     1. Import the include() function: from django.urls import include, path
15     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path, include
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('', include('main.urls')),
23 ]

```

## urls.py

사이트의 url을 지정한다.

각각의 app의 경로를 지정하는 것, 서버에서 어떤 것을 요청할지 url로 알려주는 것 등등 주로 경로(route)에 관한 것들을 수행한다.

```

config > wsgi.py > ...
1  """
2  WSGI config for config project.
3
4  It exposes the WSGI callable as a module-level variable named ``application``
5
6  For more information on this file, see
7  https://docs.djangoproject.com/en/5.0/howto/deployment/wsgi/
8  """
9
10 import os
11
12 from django.core.wsgi import get_wsgi_application
13
14 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'config.settings')
15
16 application = get_wsgi_application()
17

```

## wsgi.py

장고 애플리케이션이 Web Server와 소통하는 창구(Gateway) 를 말한다. 인터페이스로 요청을 수신하고 이를 애플리케이션으로 전달해 리턴한 응답을 다시 클라이언트에게 보낸다.

# Application

프로젝트와 애플리케이션의 가장 큰 차이점은, 애플리케이션은 프로젝트 내에 생성되며 settings.py 파일이 없다는 특징을 가지고있다.

만약 이 둘이 헷갈리면 이 방법으로 구분하자!

```
python manage.py startapp [Application name]
```

이 명령어를 통해서 Application을 생성할 수 있는 것이다.

이제 Application 안에 있는 파일의 구성요소들을 알아보겠다.

(필자의 Application name은 main이었다.)

```
main > admin.py
1  from django.contrib import admin
2
3  # Register your models here.
4
```

### admin.py

장고의 관리자 기능과 연결, 관리자 페이지를 설정할 수 있는 공간이다.

```
main > apps.py > ...
1  from django.apps import AppConfig
2
3
4  class MainConfig(AppConfig):
5      default_auto_field = 'django.db.models.BigAutoField'
6      name = 'main'
7
```

### apps.py

앱의 정보, 각각의 앱 간의 추가적인 정보를 설정해줄 수 있는 공간이다.

```
main > models.py
1  from django.db import models
2
3  # Create your models here.
4
```

### models.py

앱에서 사용하는 데이터 구조를 정리하고 DB와 소통을 담당하는 파일이다. 장고의 MODEL을 정의할 수 있는 공간이다.

```
main > tests.py
1  from django.test import TestCase
2
3  # Create your tests here.
4
```

### tests.py

프로젝트가 의도한 대로 진행될 수 있도록 테스트 코드를 작성해 체크해볼 수 있는 공간이다.

```
main > views.py > index
1  from django.shortcuts import render
2
3  # Create your views here.
4  from django.shortcuts import render
5
6  # Create your views here.
7  def index(request):
8      return render(request, 'main/index.html')
```

### views.py

앱이 어떤 행동을 해낼지 앱의 기능, 로직을 작성하는 공간이다. (이 공간에는 각 페이지 안에서 실행될 함수가 들어간다.)

```
▼ main
  > __pycache__
  > migrations
  > templates
  + __init__.py
  + admin.py
  + apps.py
  + models.py
  + tests.py
  + urls.py
  + views.py
  > Scripts
```

여기까지 Django의 프로젝트 파일을 알아보았다.

다음 글은 서버를 구동하고, 관리자 계정 만들기와 Form을 사용해보겠다.

또한 Django 공식 사이트 내에 나와있는 메뉴얼을 실습해보고 글을 작성해볼 예정이다.

스터디를 진행하며 참고한 사이트

*Django 프로젝트, 앱 구조 - <https://hyonlog.tistory.com/10>*