08 장고 프레임워크

8. 장고 프레임워크

- **최상위 폴더 생성 & 이동 : django_ex**
- **프로젝트 생성 & 이동 : django-admin startproject ycc**
  - 프로젝트는 반드시 하나 : manage.py 가 겹치기 때문
- **앱 생성 : django-admin startapp blog**
  - 하나의 프로젝트는 여러 개의 앱을 가질 수 있음

8. 장고 프레임워크

- **프로젝트 기본 파일 설명**
  - **[상위 YCC 폴더]**
  - manage.py : 건들지 말 것. 장고의 실행에 필요한 기본 파일임

  - **[하위 YCC 폴더]**
  - wsgi.py : 서버와 관련된 파일인데, 역시 건들지 말 것
  - __init__.py : 빈 파일인데, __init__.py 이름을 가진 모든 파일을 하나의 패키지로 인식하게 도와줌. 장고 뿐만 아니라 다른 패키지에서도 동일하게 적용되는 부분

  - 우리가 변경할 파일
    - settings.py : 웹사이트의 기본 경로라든가, 하는 제반 사항을 설정 (마치 게임에서 스크린샷 저장 경로를 설정하는 것과 비슷하다)
    - urls.py : 웹사이트의 url 네이밍

- **주의사항**
  - manage.py 가 있는 ycc와 settings.py가 있는 ycc가 이름이 같은데 주의할 것
  - manage.py 는 손댈 이유가 없기 때문에, ycc 폴더라고 하면 기본적으로 settings.py가 있는 ycc 폴더라고 생각하면 됨

8. 장고 프레임워크

- **프로젝트 기본 파일 중에 변경할 파일**
  - 하위ycc\settings.py
    - installed_app에 'blog' 추가가 (약 40번째 줄)
    - 'DIRS'에, BASE_DIR 추가 (약 57번째 줄)

```
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
40      'blog',
41  ]
42
43  MIDDLEWARE = [
44      'django.middleware.security.SecurityMiddleware',
45      'django.contrib.sessions.middleware.SessionMiddleware',
46      'django.middleware.common.CommonMiddleware',
47      'django.middleware.csrf.CsrfViewMiddleware',
48      'django.contrib.auth.middleware.AuthenticationMiddleware',
49      'django.contrib.messages.middleware.MessageMiddleware',
50      'django.middleware.clickjacking.XFrameOptionsMiddleware',
51  ]
52
53  ROOT_URLCONF = 'ycc.urls'
54
55  TEMPLATES = [
56      {
57          'BACKEND': 'django.template.backends.django.DjangoTemplates',
58          'DIRS': [
59              BASE_DIR,
60          ],
61          'APP_DIRS': True,
```

8. 장고 프레임워크

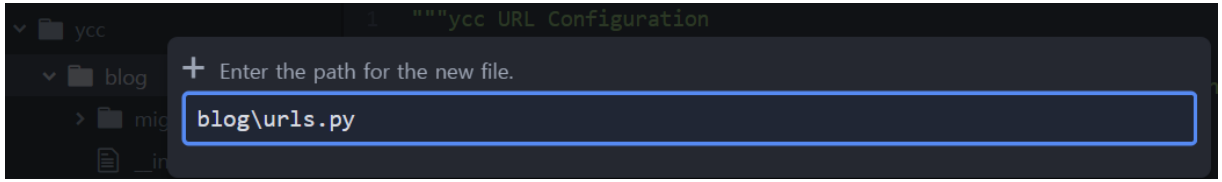
- **프로젝트 기본 파일 중에 변경할 파일**
  - 하위ycc\urls.py → urlpatterns에 path('blog', include('blog/urls.py'))
  - 하위ycc\urls.py → from Django.urls import path 옆에 include 추가

```
16  from django.contrib import admin
17  from django.urls import path, include
18
19  urlpatterns = [
20      path('admin/', admin.site.urls),
21      path('blog/', include('blog/urls.py'))
22  ]
```

- **이렇게 하는 이유 : django 개발자들이 이렇게 해야 작동하도록 만들었기 때문. 기계적으로 반복하여 익숙해지도록 합시다.**

8. 장고 프레임워크

- **블로그 앱에 있는 모듈 수정하기**
  - blog\urls.py 만들기 + 내용




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

app_name = 'blog'

urlpatterns = [
    path('list/', views.list)
]
```

  - blog\views.py 내용 수정

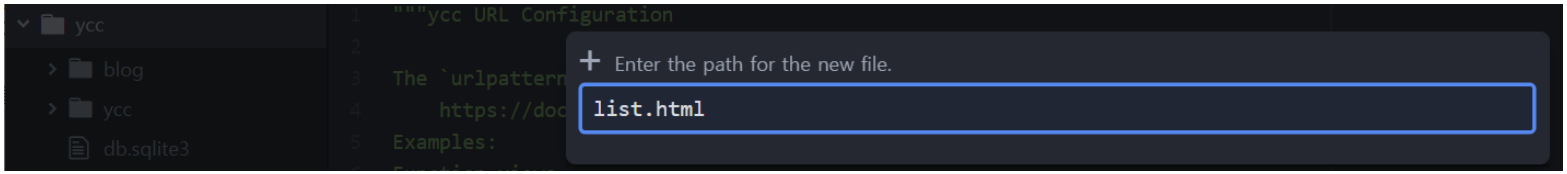
```python
from django.shortcuts import render

# Create your views here.

def list(request):
    return render(request, 'list.html')
```

8. 장고 프레임워크

- **상위 YCC 폴더에 .html 파일 만들기**
  - list.html 생성



  - 여기까지 따라하느라 고생했으니까 good!!! 이라고 적고 저장합시다.

8. 장고 프레임워크

- **서버 구동**
  - 프롬프트 → manage.py가 있는 ycc 폴더로 이동
    → python manage.py runserver
  - 크롬 → 127.0.0.1:8000/blog/list 입력

- **신기하다!!!**

8. 장고 프레임워크

- **장고 작동 방식**
  1) 127.0.0.1:8000/blog/list (URL) 을 주소창에 입력
  2) urls.py 를 따라 가보니, views.py의 list로 가라네?
  3) views.py의 list 로 가보니, list.html을 가져가라네?
  4) list.html 로 가보니, 내용이 good! 이라네?

- **기계적인 내용**
  - urls.py → views.py → .html 로 이어지는 게 한 세트라는 것만 기억하자

8. 장고 프레임워크

- **꼭 필요한 문법 설명(1) : 함수**
  - 함수 = 약속
  - 아톰 에디터에서 파란색 부분
  - print() : 괄호 안의 것을 프롬프트에서 보여주라는 “함수”

- **함수 만들어보기**

```
>>> def plus_thirty(x):
...     y = x+30
...     return y
...
>>> plus_thirty(20)
50
```

  - 헷갈리는 이유 : 우리는 y=x+30 정도면 함수라고 하는데, 파이썬 함수는 이름을 붙여줘야 함
  - plus_twenty(20)을 해보자
  - plust_thirty를 이용하여 a라는 변수에 저장해보자, 그리고 print() 함수로 출력하자

- **함수의 구성 요소**
  - def : 필수 / define의 약자
  - 함수 이름() : 필수 / 괄호 안에는 인자가 들어가도 되고 안 들어가도 되지만, 들어가는 게 일반적
  - return : 선택 / 들어가는 게 일반적 / return 되는 게 있을 경우 변수에 저장 가능

8. 장고 프레임워크

- **꼭 필요한 문법 설명(2) : import**
  - 남이 만든 모듈을 불러와서 내 것처럼 사용하기

```
1  from django.shortcuts import render
2
3  # Create your views here.
4
5  def list(request):
6      return render(request, 'list.html')
```

  - 장고 개발자들이 django폴더의 shortcuts.py에 render 라는 함수를 만들어 둔 것임.
  - 찾아보기

- **import 연습**
  - import random
  - random.randint(1,100)

  - from random import randint
  - randint(1,100)

8. 장고 프레임워크

- **함수가 좋은 이유**
  - 그 함수가 얼마나 복잡하게 설계되었든 상관 없이, 함수 사용법만 알면 되기 때문

  - randint 사용법을 잊어버림.
  - randint(100) → 오류 발생
    1) 구글링 : How to use randint / randint document
    2) 오류내역 복붙 : randint() missing 1 required positional argument: 'b'
    3) 직접 파이썬 파일 탐구

```
import random
for x in range(10):
  print random.randint(1,101)
```

```
217     def randint(self, a, b):
218         """Return random integer in range [a, b], including both end points.
219         """
220
221         return self.randrange(a, b+1)
```

5 results found for 'randint'

randint

8. 장고 프레임워크

- **꼭 필요한 문법 설명(2) : import**
  - 함수를 만들어 저장하고, import 로 불러와서 사용해보기
  - TIP : CTRL + /

- **import 연습**
  - abccc.py

abccc.py

```
1  def a_plus_b(a, b):
2      c = a + b
3      return c
```

  - plus.py

plus.py

```
1  import abccc
2  print(abccc.a_plus_b(1,2))
3
4  sum1 = abccc.a_plus_b(3,5)
5  print(sum1)
```

plus.py

```
1  from abccc import a_plus_b
2
3  sum2 = a_plus_b(1,4)
4  print(sum2)
5  print(a_plus_b(3,4))
```

8. 장고 프레임워크

- **꼭 필요한 문법 설명(3) : 선언**
  - 프로그래밍 언어에서 =은 비교가 아니라, 선언.
  - a = 50 → 이제부터 a를 50과 같은 것으로 취급하겠다.
  - b = ‘blog’→ 문자를 선언할 때는 따옴표 혹은 쌍따옴표 필수

  - 변수

  - 비교는 ==
  - 50==49
  - a==49
  - a==50

8. 장고 프레임워크

- **복습 문제**
  1) 상위 ycc 폴더로 이동하세요.
  2) Community라는 이름의 app을 만드세요.
  3) 127.0.0.1:8000/community/write/ 라는 url을 만드세요.
  4) 위 url을 입력했을 때 “hello naver” 라는 글자가 쓰여 있는 hello.html을 만들고 연결하세요.

- **응용 문제**
  5) hello naver를 클릭했을 때, 네이버 홈으로 이동하는 링크를 거세요.
  6) 저번에 만들었던 list.html을 살짝 변경하여, ‘good!!’를 클릭하면 ‘127.0.0.1:8000/community/write/’ 주소로 이동하는 링크를 거세요.

  ※ 1주차에 했던 1장 〈HTML 이해〉를 참고

8. 장고 프레임워크

- **복습 문제 답**

1) community 라는 이름의 앱 만들기

Anaconda Prompt

```
(base) C:₩Users₩송규호₩pyworks₩django_ex₩ycc3>django-admin startapp community
```

2) settings.py 에서 앱 등록하기

settings.py urls.py

```
27
28  ALLOWED_HOSTS = []
29
30
31  # Application definition
32
33  INSTALLED_APPS = [
34      'django.contrib.admin',
35      'django.contrib.auth',
36      'django.contrib.contenttypes',
37      'django.contrib.sessions',
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
40      'blog',
41      'community',
42  ]
```

3) urls.py에서, 'community/' 라는 url 만들기

```
16  from django.contrib import admin
17  from django.urls import path, include
18
19  urlpatterns = [
20      path('admin/', admin.site.urls),
21      path('blog/', include('blog.urls')),
22      path('community/', include('community.urls')),
23  ]
```

8. 장고 프레임워크

- **복습 문제 답**

4) community/urls.py 새로 만들고 기존 url에 ‘write/’ 추가하기

urls.py

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

app_name = 'community'

urlpatterns = [
    path('write/', views.write),
]
```

5) views.py에서 html과 연결하기

views.py

```python
from django.shortcuts import render

# Create your views here.

def write(request):
    return render(request, 'hello.html')

```

6) 최상위 ycc 폴더에 hello.html 만들기

hello.html

```
hello naver
```

8. 장고 프레임워크

- **응용 문제 답**

7) ycc/list.html에서 〈a〉〈/a〉태그 추가하기

list.html

```
1   <a href = "http://127.0.0.1:8000/community/write">good!!</a>
```

8) ycc/hello.html 에서 〈a〉〈/a〉태그 추가하기

hello.html

```
1   <a href = "https://www.naver.com">hello naver</a>
```

8. 장고 프레임워크

- **의문**
  - http://127.0.0.1:8000/blog/list/
  - http://127.0.0.1:8000/community/write/
  - 왜 한꺼번에 url 이름이 두 개나 바뀔까? 너무 비효율적인 것 아닌가?

- **맞습니다**
  - 마치 네이버 지식인 홈 → 네이버 웹툰 홈으로 링크가 걸린 격.
  - 네이버 지식인 홈 → 네이버 지식인 글쓰기로 링크가 걸리는 게 일반적.

- **일반적인 방법**
  - http://127.0.0.1:8000/blog/list/
  - http://127.0.0.1:8000/blog/new/
  - http://127.0.0.1:8000/blog/edit/
  - 한 앱에서의 연결성이 좋아야 한다.
  - 방금 전에 만든 community는 머리에서 싹 잊고, blog 앱에만 집중해 봅시다.

8. 장고 프레임워크

- **자동으로 url 만들기**
  - blog/urls.py 에서 'post/<int:pk>/' url 만들기

urls.py

```
1  from django.urls import path
2  from . import views
3
4  app_name = 'blog'
5
6  urlpatterns = [
7      path('list/', views.list),
8      path('post/<int:pk>/', views.post),
9  ]
```

  - views.py 에서 post 함수 만들기 → post.html 만들고 아무 말이나 쓰기

views.py

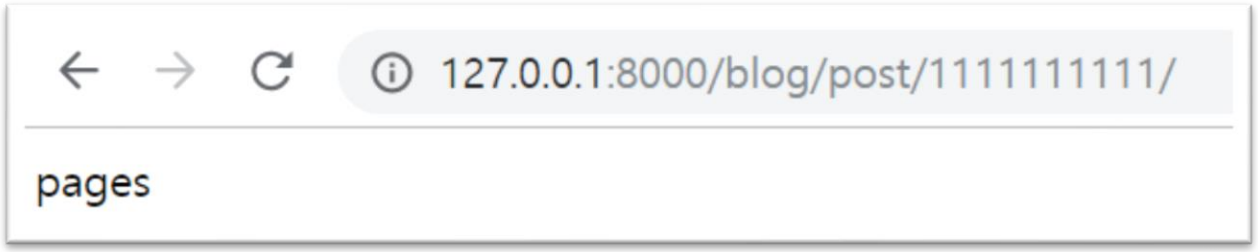
```
1  from django.shortcuts import render
2
3  # Create your views here.
4
5  def list(request):
6      return render(request, 'list.html')
7
8  def post(request, pk):
9      return render(request, 'post.html')
```

post.html

```
1  pages
```

8. 장고 프레임워크

- **자동으로 url 만들기**
  - 127.0.0.1:8000/blog/post/숫자/



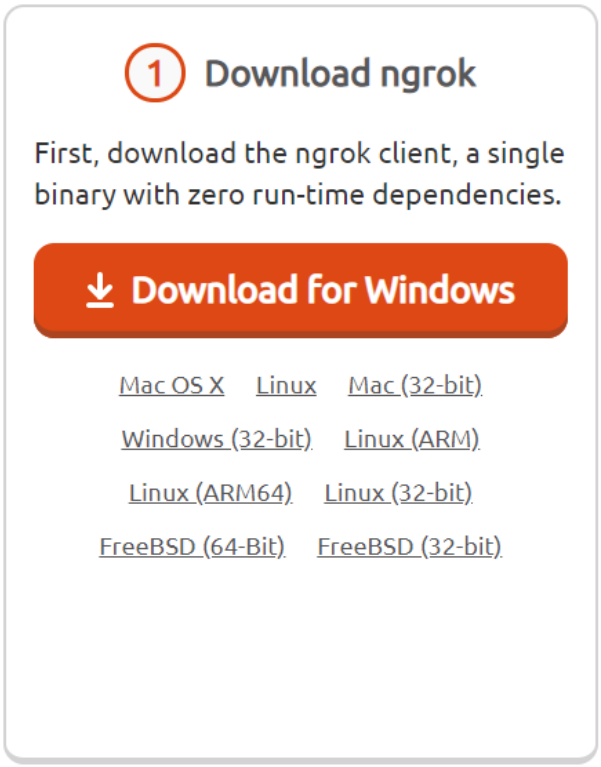
  - 127.0.0.1:8000/blog/post/문자/

  - 〈int:pk〉 → 〈pk〉

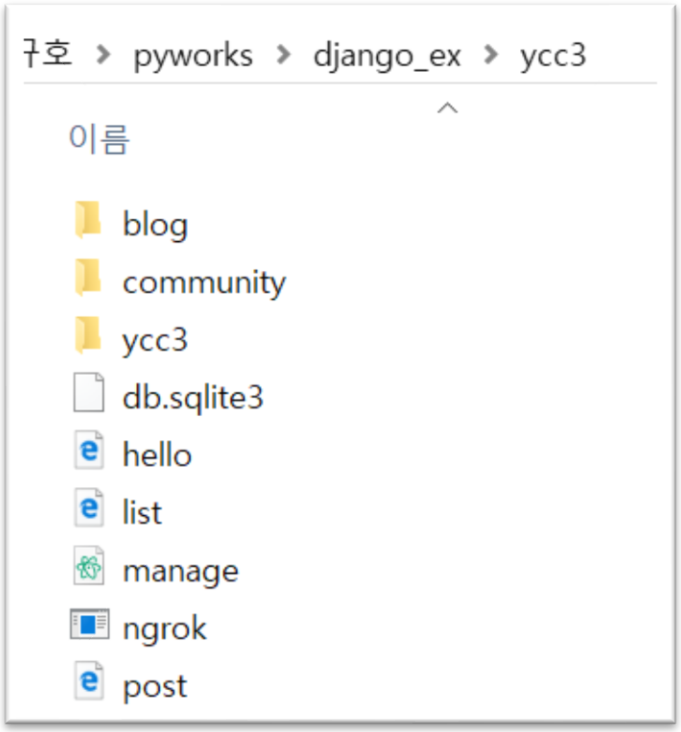
09 Ngrok 서버

9. NGROK 서버

- Ngrok 다운로드
  - https://ngrok.com/download → 다운로드 → 압축풀기



  - manage.py 가 있는 폴더로 ngrok.exe 옮기기



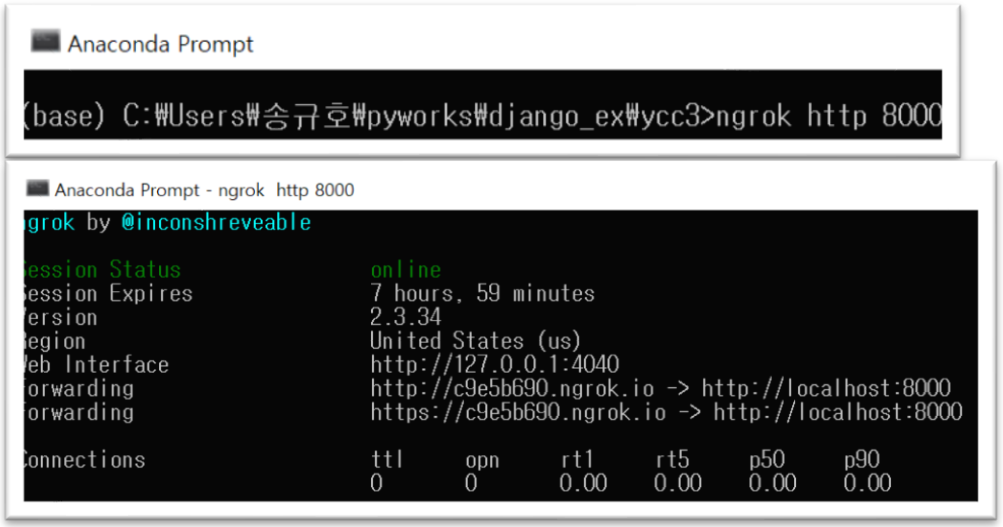
9. NGROK 서버

- **settings.py**
  - ALLOWED_HOSTS = ['*'] // * : 모든 호스트를 다 인정하겠다는 뜻

settings.py

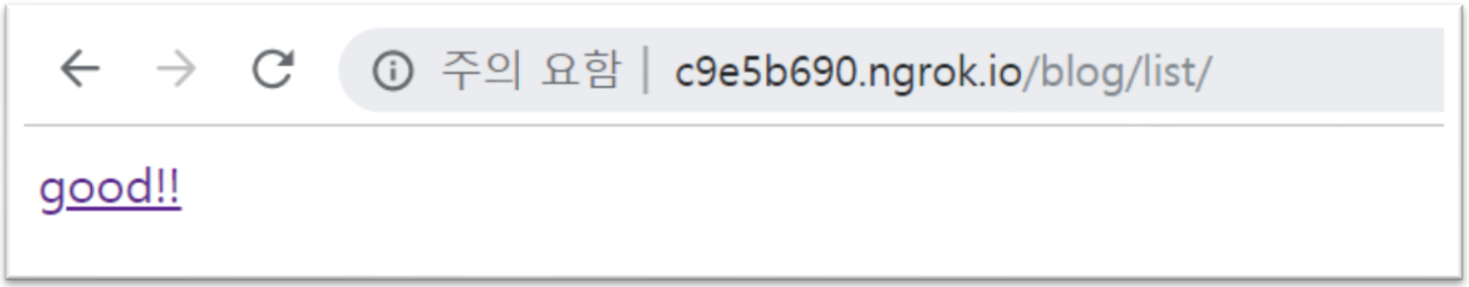
```
27
28  ALLOWED_HOSTS = ['*']
```

- **Ngrok 실행**
  - 아나콘다 프롬프트 하나 더 실행 → ngrok.exe가 있는 폴더로 이동
  - ngrok http 8000



9. NGROK 서버

- http://c9e5b690.ngrok.io/blog/list



- **스마트폰에서 들어가보기!!**
  - 주소복사 → 카톡 자기에게 보내기 → 스마트폰 카톡으로 링크 클릭

- **외부에서 접속할 수 있는 길을 임시로 뚫어줍니다.**

여기까지 튜토리얼이었고요

- 이제부터 진짜입니다.
- 지금까지 한 건, 진짜 기계적인 내용. 더 응용할 것도 없어요.
- 여러 프로젝트를 만들다 보면 100번은 반복합니다
- 당연히 익숙해진다~

- 다음주부터는 아마도 좀 어렵다.
- 시험기간은 언제부터…?