

자 이제 시작이야 Django

충북대학교 멋쟁이사자처럼 박태미

4주차 강의

< 오늘의 목표 >

C_(create) R_(read) U_(update) D_(delete)

쉽게 배우기

+ git

바탕화면에 파일을 만들어보자

이 친구 ↓ (이름 : 맘대로)

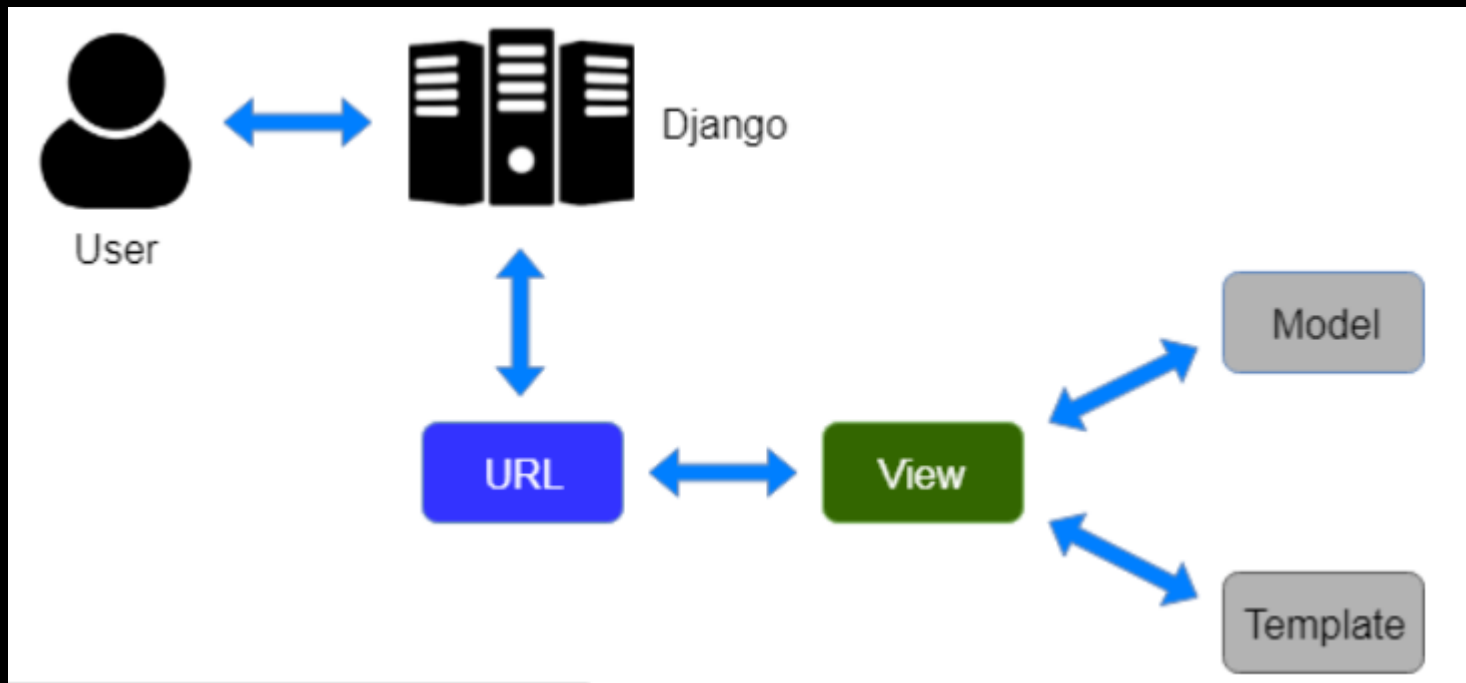


Django란?

Django = python으로 작성된 오픈 소스
웹 프레임워크

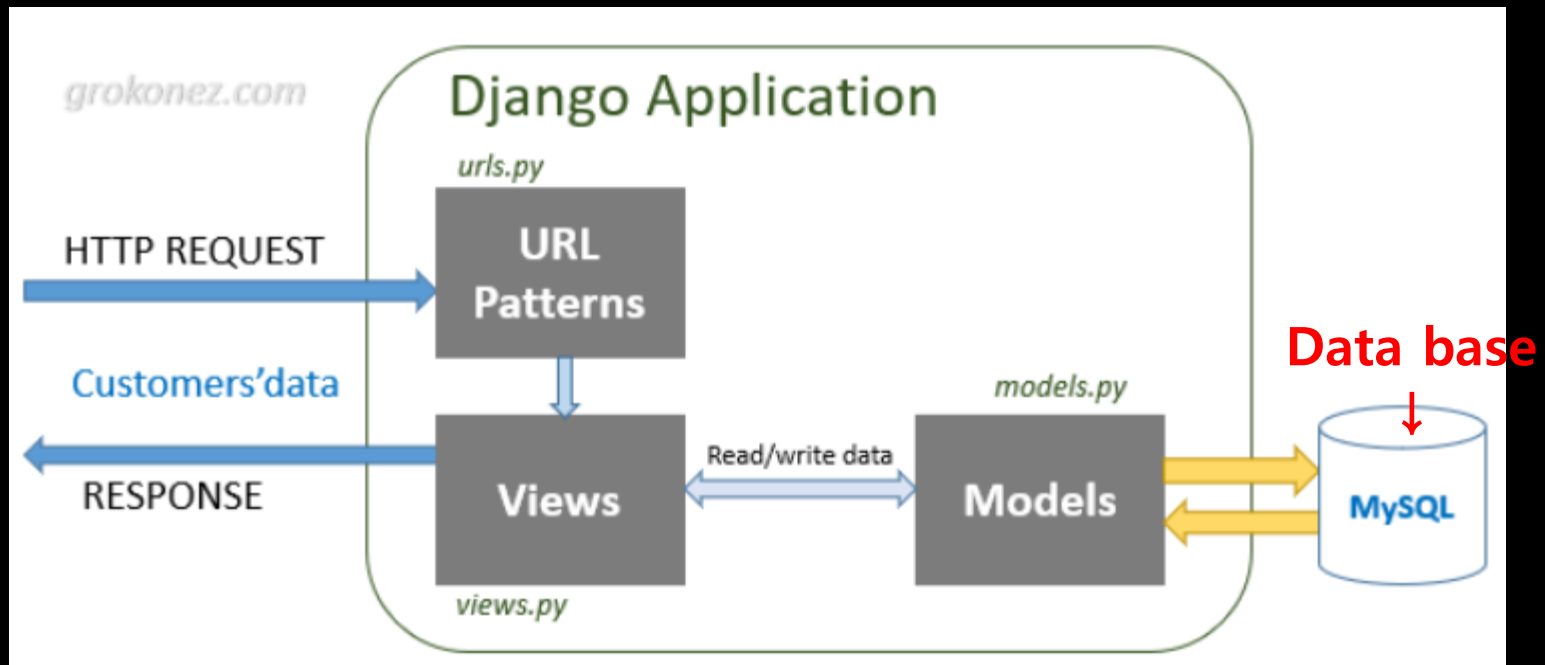
Django란?

M (Model) T (Template) V (View)



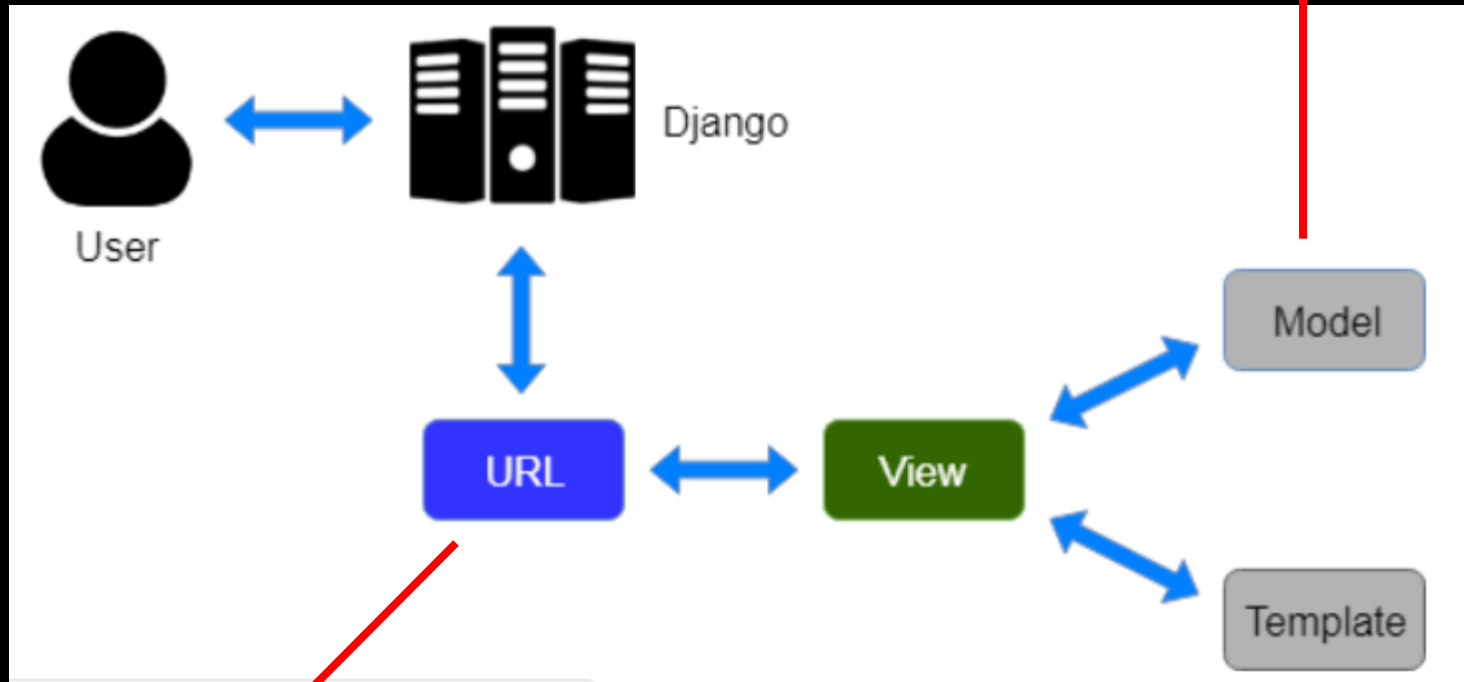
Django란?

M (Model) **T** (Template) **V** (View)

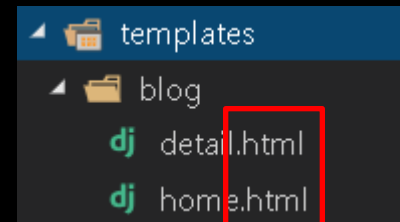


모델이 생성해주는
테이블

POST	author	title	text	created_date	published_date
id					
1					
2					
3					
4					



127.0.0.1:8000/blog/home



일단 해보자

Vscode 를 켜주세요

파일 불러오기

파일(F) -> 폴더열기

터미널 켜기

터미널(T) -> 새 터미널

(있어 보이는) 단축기도 있어요~ : ctrl + shift + `

이 친구가 잘 나왔나요?

문제 출력 디버그 콘솔 터미널

```
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/likelion_test  
$ █
```

^ ____ ^ 안녕

기본 환경 셋팅

가상 환경 만들기

가상 환경 켜기

Django 설치

가상 환경

가상 환경 : 내가 원하는 것들만 담을 수 있는 **가방**

ex) 나는 **학교**를 가기 위해 **가방1**에 **필통**과 **책**을 넣었어.

ex) 나는 **시장**를 가기 위해 **가방2**에 **지갑**을 넣었어.

나는 **웹**을 만들기 위해 **가상환경**에 **django**를 설치했어.

졸업? 학교 갈 때 사용했던 **가방1** 버려버리기 = 가상환경 삭제
!안에 있는 설치 파일 한번에 삭제 완료!

가상 환경

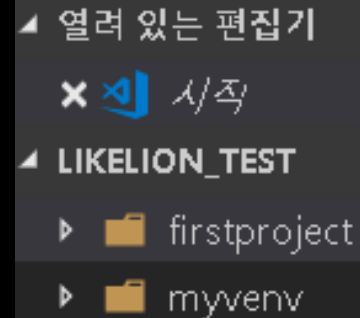
\$ python -m venv <가상환경명>

```
$ python -m venv myvenv
```




\$ Source <가상환경명>/Scripts/activate

```
$ source myvenv/Scripts/activate  
(myvenv)
```

\$ deactivate



▲ 열려 있는 편집기

- ×  시/작
- ▲ LIKELION_TEST
 - ▶  firstproject
 - ▶  myvenv

A red wavy line is drawn under the 'myvenv' folder.

```
$ pip install django==2.1.7
```

기본 환경 세팅 끝!

시작하기 전에 약속하기

※ 파일을 수정했으면 ctrl + s 키를 눌러 바로 바로 저장하기

※ 명령어에 manage.py 가 있다면

그 파일이 있는 폴더에서 명령어를 입력하기

일단 시작해보자

NameError at /blog/post/new/

name 'PostForm' is not defined

Request Method: GET

Request URL: http://127.0.0.1:8000/blog/post/new/

Django Version: 2.1.7

Exception Type: NameError

Exception Value: name 'PostForm' is not defined

Exception Location: C:\Users\박태미\Desktop\likelion_test\firstproject\blog\views.py in post_new, line 24

Python Executable: C:\Users\박태미\Desktop\likelion_test\myenv\Scripts\python.exe

Python Version: 3.7.2

Python Path: ['C:\Users\박태미\Desktop\likelion_test\firstproject',
'C:\Users\박태미\AppData\Local\Programs\Python\Python37\python.zip',
'C:\Users\박태미\AppData\Local\Programs\Python\Python37\DLLs',
'C:\Users\박태미\AppData\Local\Programs\Python\Python37\lib',
'C:\Users\박태미\AppData\Local\Programs\Python\Python37\lib',
'C:\Users\박태미\Desktop\likelion_test\myenv',
'C:\Users\박태미\Desktop\likelion_test\myenv\lib\site-packages']

Server time: Sat, 6 Apr 2019 07:32:11 +0900

Traceback [Switch to copy-and-paste view](#)

C:\Users\박태미\Desktop\likelion_test\myenv\lib\site-packages\django\core\handlers\exception.py in inner

34. response = get_response(request)

► Local vars

C:\Users\박태미\Desktop\likelion_test\myenv\lib\site-packages\django\core\handlers\base.py in _get_response

126. response = self.process_exception_by_middleware(e, request)

► Local vars

C:\Users\박태미\Desktop\likelion_test\myenv\lib\site-packages\django\core\handlers\base.py in _get_response

124. response = wrapped_callback(request, *callback_args, **callback_kwargs)

► Local vars

C:\Users\박태미\Desktop\likelion_test\firstproject\blog\views.py in post_new

24. form = PostForm()

► Local vars

오류화면

1. 당황하지 않기

2. 원인 찾기

3. 오타 확인하기

한번도 안 뜨면

그 사람 이상한 거

블로그를 만들어보자!

이제부터가 진짜 시작

1. project / app 생성 → setting 설정
→ 데이터베이스 생성 → 개발 서버 실행
2. Model 생성 → 데이터베이스에 적용
3. 장고 관리자 생성 → 관리자 페이지에서 글 작성
4. url 생성 (include) → view 작성 → template 생성/작성
→ Post 불러오기 → 특정 글 불러오기 (view+ template + url)
5. Form 만들기 → template 작성 → url 작성 →
view작성 → 글 생성

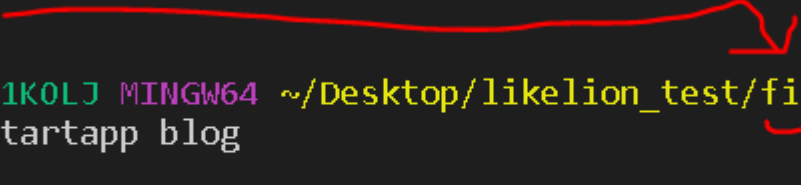
\$ django-admin startproject <project name>

```
$ django-admin startproject firstproject
```

\$ python manage.py startapp <app name>

여기서 주의할 점! manage.py

```
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/likelion_test (master)
$ cd firstproject/
(myvenv)
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/likelion_test/firstproject (master)
$ python manage.py startapp blog
(myvenv)
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/likelion_test/firstproject (master)
$
```



- firstproject
 - blog
 - firstproject
 - manage.py
 - myenv

Firstproject <-- project
└ blog <-- app

- firstproject
 - blog
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - views.py
 - firstproject
 - __pycache__
 - __init__.py
 - settings.py
 - urls.py
 - wsgi.py
 - manage.py
 - myenv

[app]

admin.py
models.py
views.py

[project]

settings.py
urls.py

setting.py

앱 이름 추가하기

시간 바꿔주기

STATIC 경로 설정해주기

```
# Application definition
```

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'blog',  
]
```

```
TIME_ZONE = 'Asia/Seoul'
```

```
USE_I18N = True
```

```
USE_L10N = True
```

```
USE_TZ = True
```

```
# Static files (CSS, JavaScript, Images)  
# https://docs.djangoproject.com/en/2.1/howto/  
static-files/
```

```
STATIC_URL = '/static/'
```

```
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
```

setting.py

```
TIME_ZONE = 'Asia/Seoul'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.1/howto/static-files/

STATIC_URL = '/static/'
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
```

- 'static' 숨김 폴더를 생성
- collectstatic 명령어를 치면 파일이 나타남
- 그곳에 정적파일 모음

데이터베이스 생성을 위해 `migrate`를 한다.

```
$ python manage.py migrate
```

```
$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions

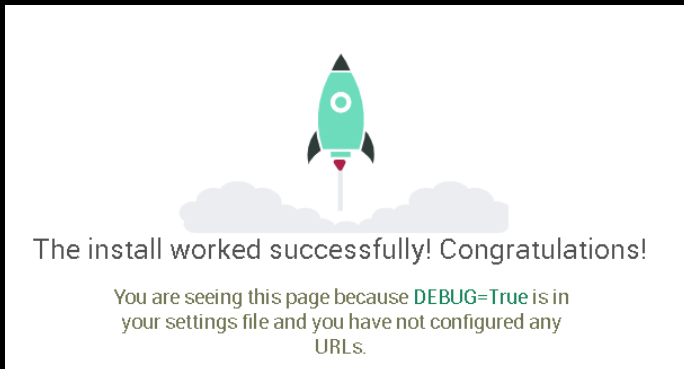
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
```

개발 서버 실행

\$ python manage.py runserver

```
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/likelion_test/firstproject (master)
$ python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
March 28, 2019 - 00:12:45
Django version 2.1.7, using settings 'firstproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```



(주소) ctrl + 좌클릭

서버를 돌려보자

<- 이 친구 나오면 성공

명령어를 다시 치고 싶다면
ctrl + c

우리는 방금 첫 웹사이트를 만들었습니다!

+ 팁

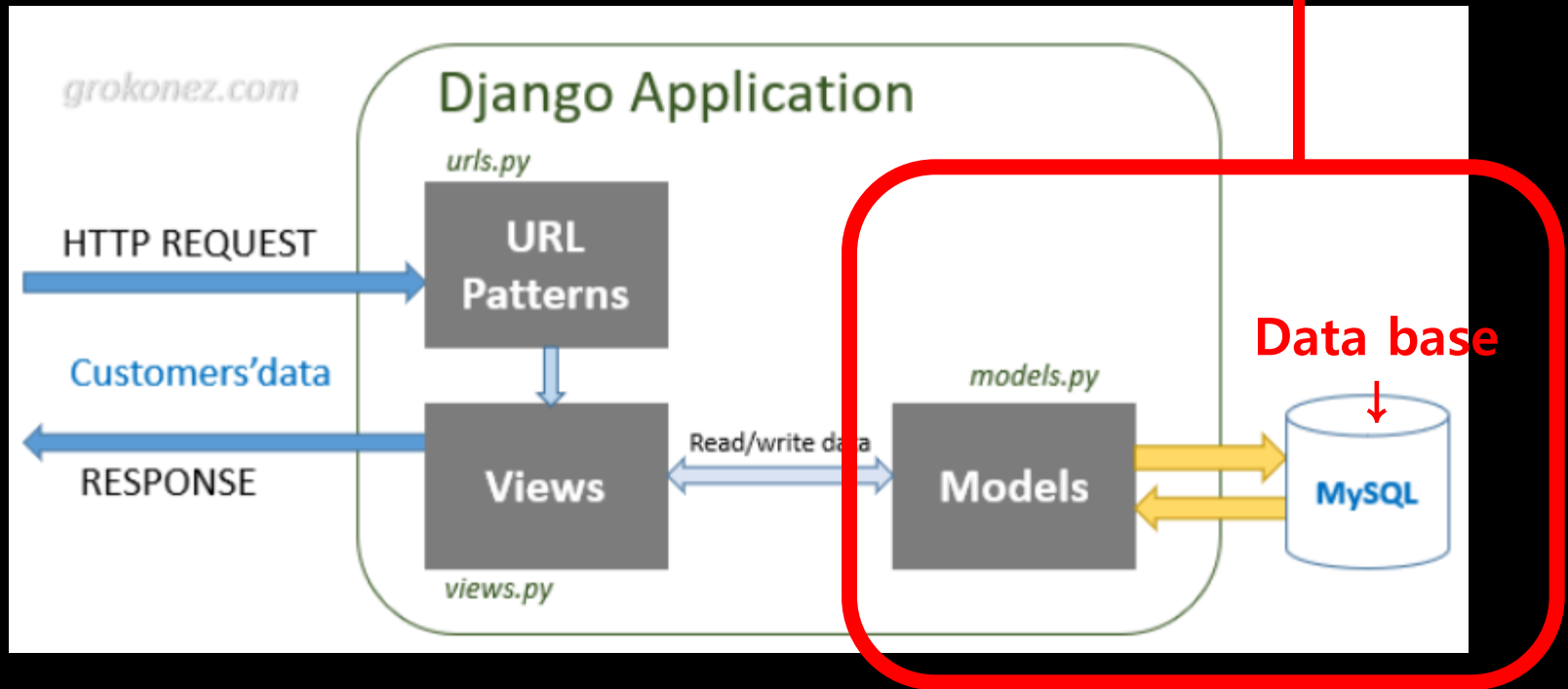
명령어가 필요 없이 코드 변경만 있다면 runserver를 계속 치지 않아도

페이지 새로고침을 하면 변경사항이 들어가있어요!

기억나시나요?

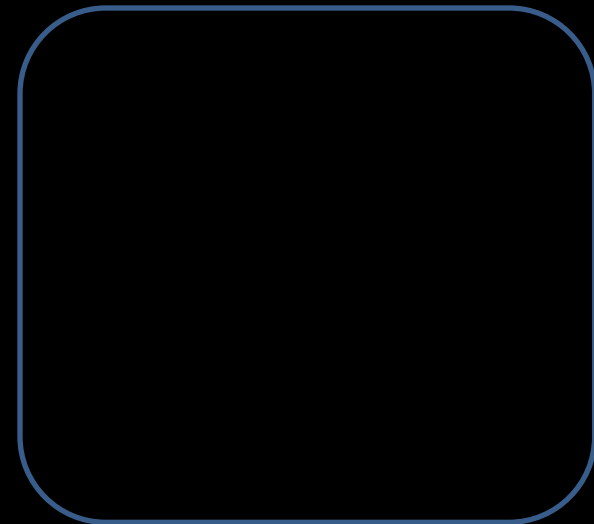
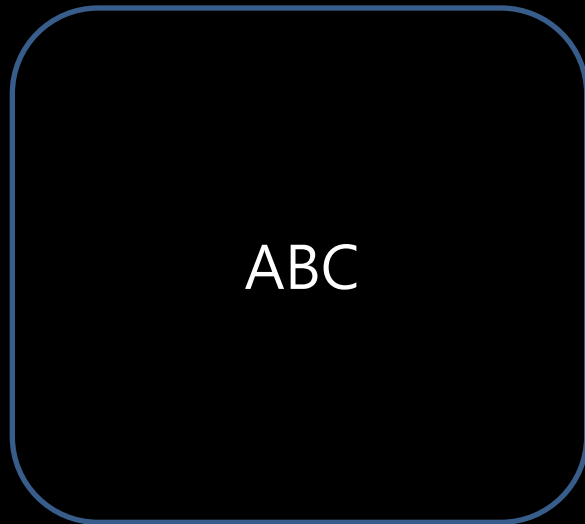
아까 했는데...

이제 여기를 건들 거예요



Model

DB



models.py

```
from django.db import models
from django.utils import timezone

# Create your models here.

class Post(models.Model):
    author = models.ForeignKey('auth.User', on_delete=models.CASCADE)
    title = models.CharField(max_length=200)
    text = models.TextField()
    created_date = models.DateTimeField(default=timezone.now)
    published_date = models.DateTimeField(blank=True, null=True)

    def publish(self):
        self.published_date = timezone.now()
        self.save()

    def __str__(self):
        return self.title
```

외부 파일 импорт

models.py

POST	author	title	text	created_date	published_date
id					
1					
2					
3					
4					

테이블
이름

테이블
필드 속성

테이블
컬럼 이름

```
from django.db import models
from django.utils import timezone
```

```
# Create your models here.
```

```
class Post(models.Model):
```

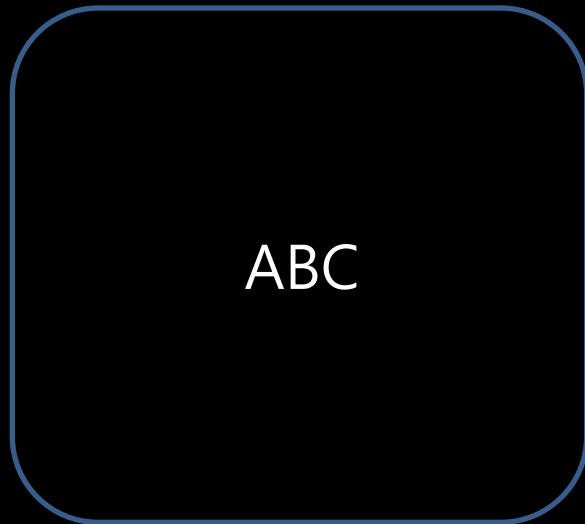
```
    author = models.ForeignKey('auth.User', on_delete=models.CASCADE)
    title = models.CharField(max_length=200)
    text = models.TextField()
    created_date = models.DateTimeField(default=timezone.now)
    published_date = models.DateTimeField(blank=True, null=True)
```

```
    def publish(self):
        self.published_date = timezone.now()
        self.save()
```

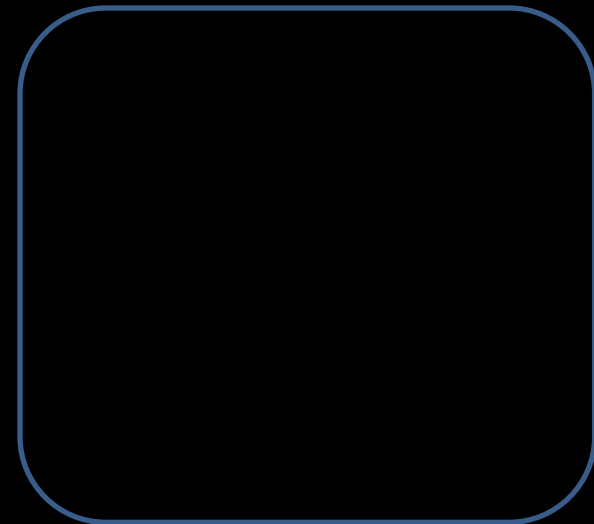
```
    def __str__(self):
        return self.title
```

Model

DB



저장



Django에게 변경된 점 알려주기

```
$ python manage.py makemigrations (blog)
```

```
$ python manage.py migrate (blog)
```

```
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/li
$ python manage.py makemigrations
Migrations for 'blog':
  blog\migrations\0001_initial.py
    - Create model Post
(myvenv)
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/li
$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, blog, con
Running migrations:
  Applying blog.0001_initial... OK
(myvenv)
TaeMePark@DESKTOP-021K0LJ MINGW64 ~/Desktop/li
```

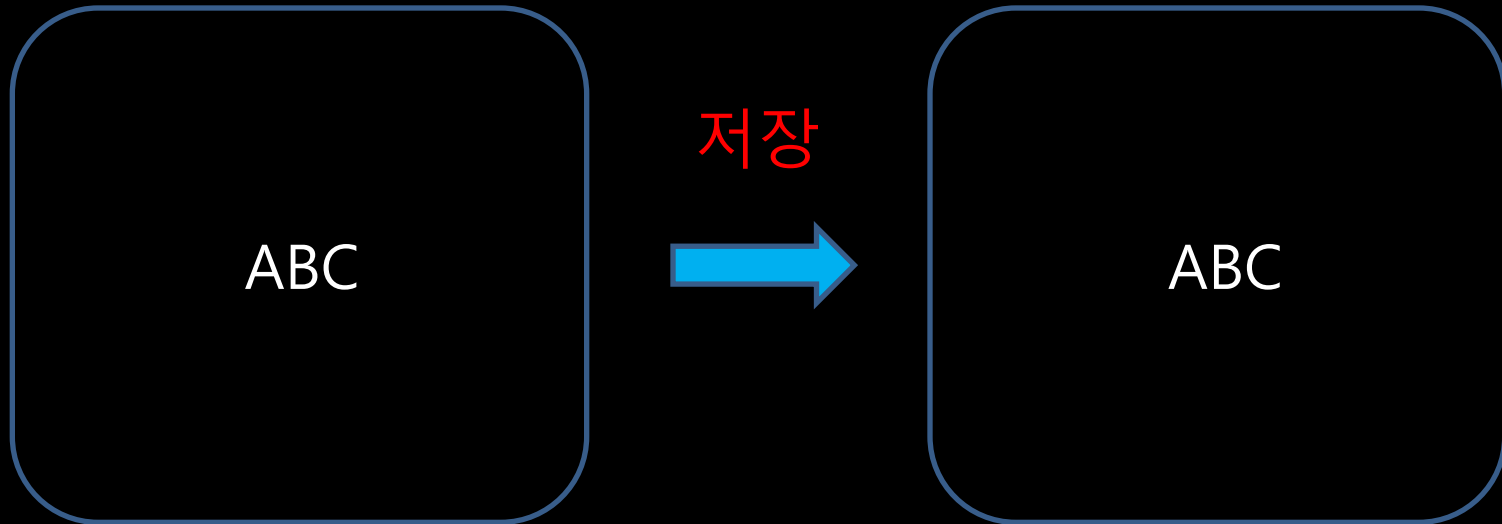


이렇게 뜬다면 성공!

아니면 오류를 해결해보고 다시 도전

Model

DB



Post 모델을 데이터베이스에 저장 완료

Django 관리자 = django admin

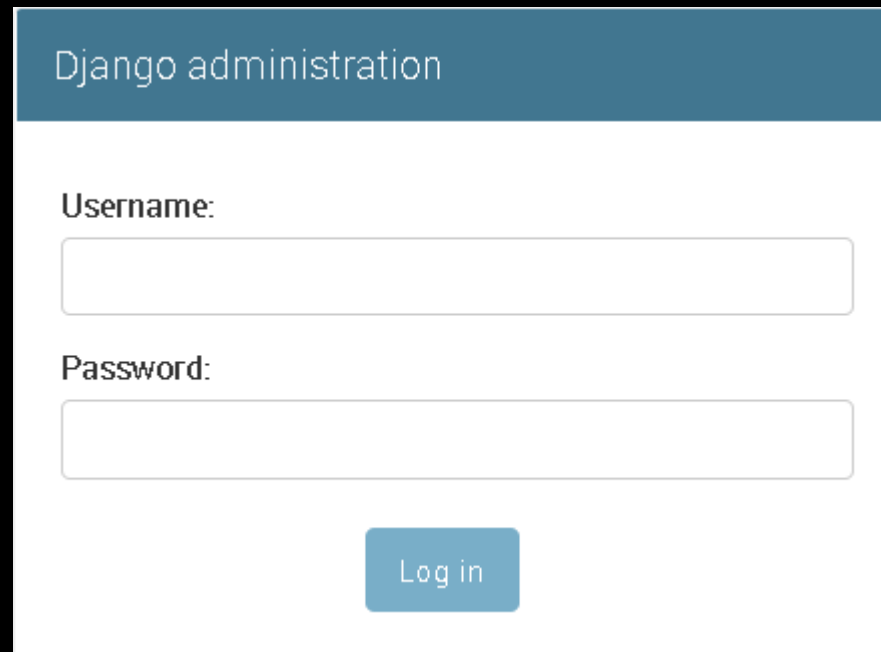
Django는 관리자 페이지 앱을 자동 생성해줍니다.

한번 들어가볼까요?

<주소>

localhost:8000/admin

127.0.0.1:8000/admin



The image shows a screenshot of the Django administration login interface. It features a teal header bar with the text "Django administration". Below the header, there are two input fields: "Username:" and "Password:". At the bottom center, there is a teal "Log in" button.

Django administration

Username:

Password:

Log in

로그인을 하라고 하니, 관리자 계정을 생성해봅시다.

```
$ python manage.py createsuperuser
```

```
$ python manage.py createsuperuser
Username (leave blank to use 'taemepark'): taemi
Email address:
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
(myvenv)
```

1. 이름 (필수)
2. Email (선택)
3. 비밀번호 (놀라지 마요 텍스트가 보이지 않아요)
4. 비밀번호 확인 (역시 마찬가지로)

아까 그 페이지로 가서 로그인을 해볼까요?

Django administration

Username:

Password:

[Log in](#)



Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups	+ Add	Change
Users	+ Add	Change

관리자 앱 페이지? 뭐하는데?

모델을 효율적으로 관리 할 수 있는 앱

모델을 가지고 테스트 가능



Django 관리자 앱으로 모델을 불러오기

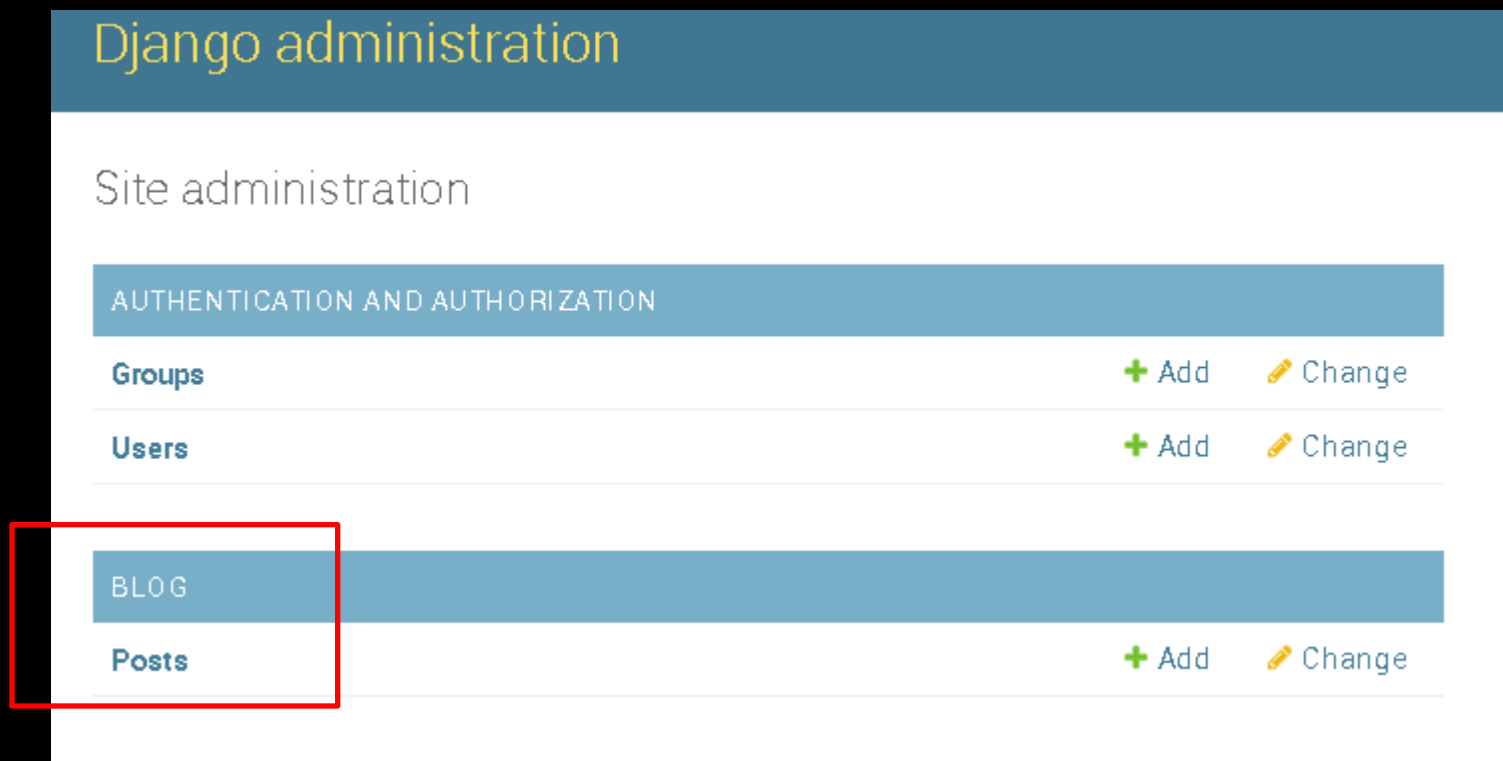
admin.py

```
1 from django.contrib import admin
2 from .models import Post
3
4 admin.site.register(Post)
```

models 에 있는 Post 를 데려와서

Admin.site.register() 을 통해 등록

다시 돌려보면 이렇게 **Blog(앱)**에 있는 **Post(모델)**가 추가 되었습니다.



관리자 앱 페이지를 통해 글을 작성해봅시다!

Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups

[+ Add](#) [Change](#)

Users

[+ Add](#) [Change](#)

BLOG



Posts

[+ Add](#) [Change](#)

아까 저희가 입력한 필드와 비교해보면, 나타나있죠



```
class Post(models.Model):  
    author = models.  
    title = models.  
    text = models.T  
    created_date =  
    published_date
```



Add post

Author:  

Title:

Text:

Created date: Date: Today | 
Time: Now | 

Published date: Date: Today | 
Time: Now | 

우리는 글을 하나 생성했습니다!

Change post

Author:

taemi ▼ ✎ +

Title:

멋쟁이사자처럼

Text:

충북대학교 멋쟁이사자처럼 너무너무 좋당

Created date:

Date: 2019-03-28 Today 📅

Time: 10:08:21 Now ⌚

Published date:

Date: 2019-03-28 Today 📅

Time: 10:12:33 Now ⌚

Delete

Select post to change

Action: ----- ▼

Go

0 of 1 selected

☐ POST

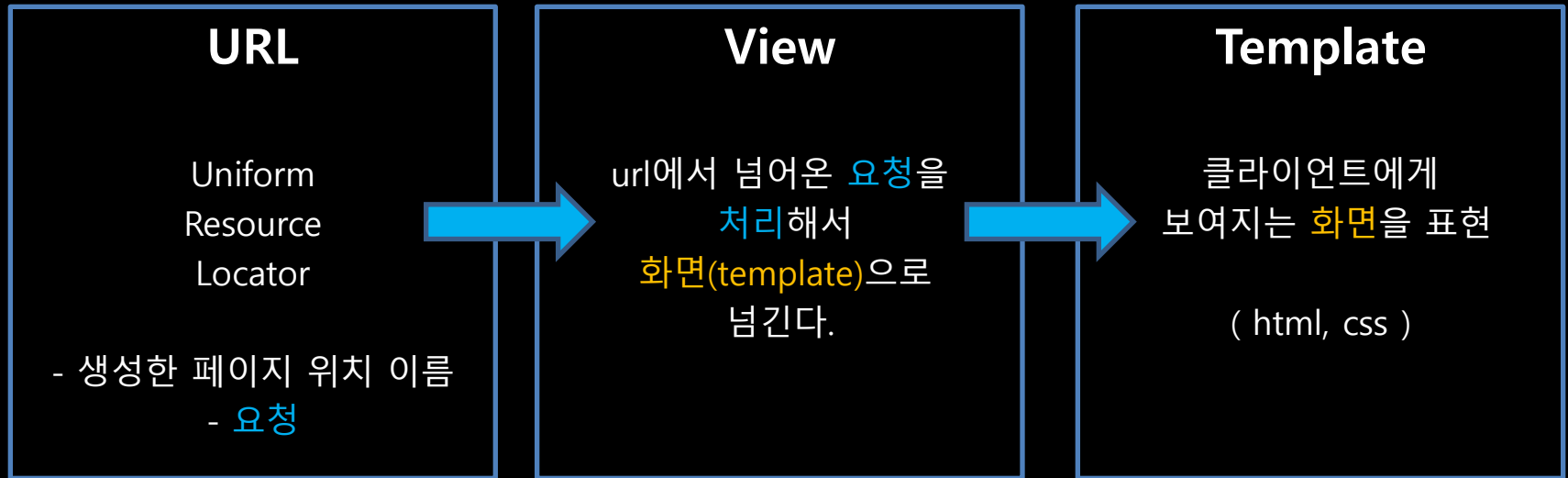
☐ 멋쟁이사자처럼

1 post

다음으로 우리가 해야 할 것은?

다음으로 우리가 해야 할 것은?

내가 만든 페이지에서 글을 확인하고 싶다!



내가 만든 페이지에서 글을 보기 위한 과정

URL

Uniform
Resource
Locator

- 생성한 페이지 위치 이름
- 요청

admin 페이지의 url

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

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

실제 url을 쓰는 부분

localhost:8000/admin
127.0.0.1:8000/admin

url 이 요청할 곳

url 이 어떻게 구성되어 있는지 확인해보자

URL

Uniform
Resource
Locator

- 생성한 페이지 위치 이름
- 요청

admin.py

```
from django.contrib import admin
from django.urls import path
import blog.views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('blog/home', blog.views.home, name='home'),
```

실제 url을 쓰는 부분

localhost:8000/blog
127.0.0.1:8000/blog

url 에게 붙은 이름

url 이 요청할 곳

우리도 해보자

View

url에서 넘어온 요청을
처리해서
화면(template)으로
넘긴다.

views.py

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

View

url에서 넘어온 request를

함수를 실행해서

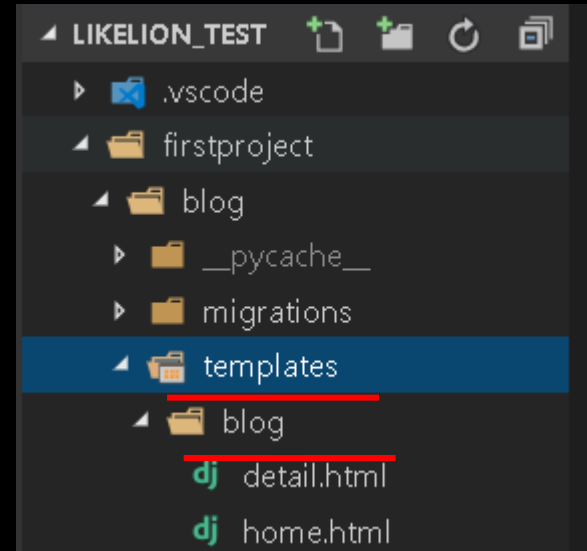
home.html 으로

넘긴다.

Template

클라이언트에게
보여지는 **화면**을 표현

(html, css)



[blog] 앱 안에

[templates] 폴더 생성 그 안에

[blog] 폴더 생성 그 안에

[home. html] 파일 생성

Template

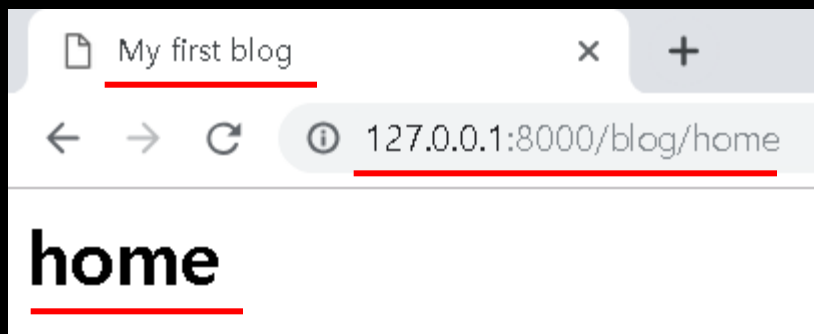
클라이언트에게
보여지는 **화면**을 표현

(html, css)

home.html

```
1 <head>
2   <title> My first blog </title>
3
4 </head>
5 <body>
6   <h1>home</h1>
7 </body>
8
```

잘 돌아가는지 확인해봅시다.



이제 정말 글을 가져와보자.

글 목록을 띄우려면 어디어디를 고쳐야 할까?

(url / view / template)

View

url에서 넘어온 요청을
처리해서
화면(template)으로
넘긴다.

이 폴더에 있는 models에서
Post 데이터들을 가져와서
posts 넣어줌

Posts라는 것을
Template에서
Posts로 사용할 수 있도록 넘겨줌

```
1 from django.shortcuts import render
2 from .models import Post
3
4 # Create your views here.
5 def home(request):
6     posts = Post.objects.all()
7     return render(request, 'blog/home.html', {'posts': posts})
```

views.py

View

url에서 넘어온 요청을
처리해서
화면(template)으로
넘긴다.

Render 함수

자리 의미 (받는 인수)

첫 번째 : request 객체 (필수)
두 번째 : 템플릿 이름 (필수)
세 번째 : 사전형 객체 (선택)

```
1 from django.shortcuts import render
2 from .models import Post
3
4 # Create your views here.
5 def home(request):
6     posts = Post.objects.all()
7     return render(request, 'blog/home.html', {'posts': posts})
```

views.py

Template

클라이언트에게
보여지는 **화면**을 표현

(html, css)

home.html

```
1 <head>
2   <title> My first blog </title>
3
4 </head>
5 <body>
6   <h1>home</h1>
7   {{posts.all}}
8
9 </body>
10
```


뜨긴 뜨는데...

home

<QuerySet [<Post: 멋쟁이사자처럼>]>

여기서 잠깐

```
1  <head>
2    <title> My first blog </title>
3
4  </head>
5  <body>
6    <h1>home</h1>
7    {{posts.all}}
8
9  </body>
10
```



Template 안에서 변수와 로직 표현을 배워보자

{{ <변수명> }} : view에서 context로 넘겨준 변수를 표현

{% <로직> %} : template을 동적으로 바꿔줄 로직 표현
(ex> for문, if문 등등..)

여기서 잠깐

home

<QuerySet [<Post: 멋쟁이사자처럼>]>

QuerySet 이 뭐지?

Query : DB에 요청(질의)하는 질의어

QuerySet : Query의 결과로 반환된 데이터의 집합

Template

클라이언트에게
보여지는 화면을 표현

(html, css)

다시 해보자

home.html

```
<haed>
  <title> My first blog </title>
</head>
<body>
  <h1>home</h1>
  {% for post in posts.all %}
    <div class="container">
      <h2>{{post.title}}</h2>
      <p>{{post.created_date}}</p>
      <p>{{post.text}}</p>
    </div>
  {% endfor %}
</body>
```

잘 나오시나요?

home

멋쟁이사자처럼

March 28, 2019, 10:08 a.m.

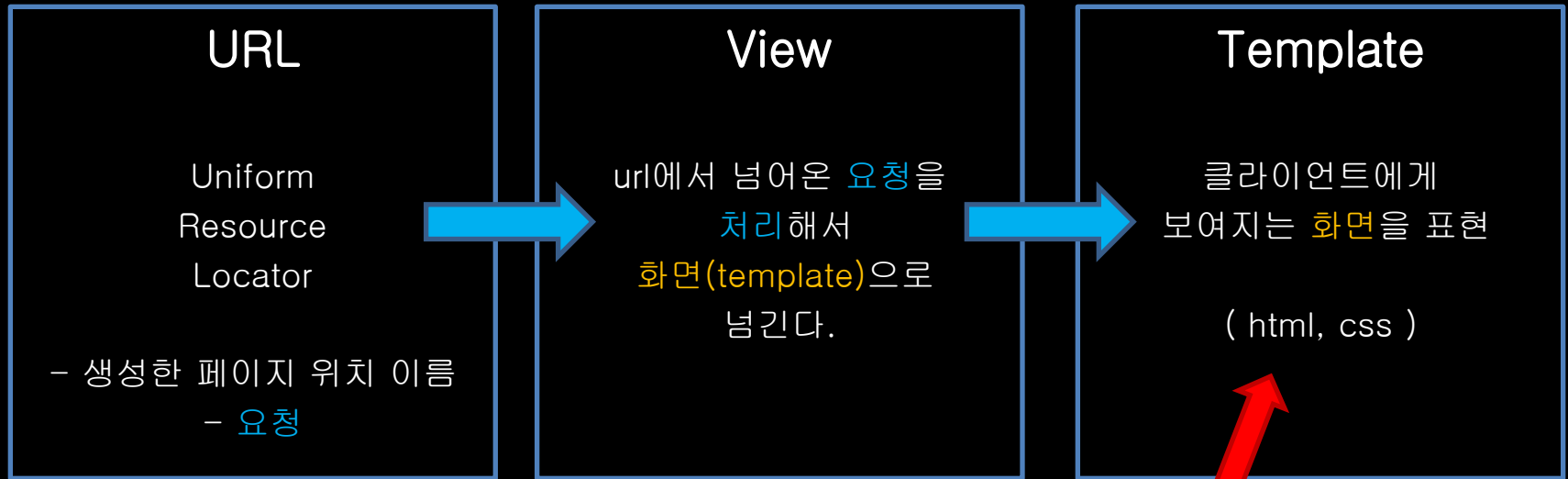
충북대학교 멋쟁이사자처럼 너무너무 좋당

오늘 강의

March 28, 2019, 3:45 p.m.

관찰을까 모르겠네..

한번 더 확인해보자



```
path('blog/home', blog.views.home, name='home'),
```

```
1 from django.shortcuts import render
2 from .models import Post
3
4 # Create your views here.
5 def home(request):
6     posts = Post.objects.all()
7     return render(request, 'blog/home.html', {'posts': posts})
```


하지만

우리가 원하는 게시판의 형태가 아닙니다.

Home에는 글 제목들만 나타내고,

제목 클릭했을 때 특정 게시물만 보고 싶다면?

원하는 형태에 게시판을 만들어보자.

Home에는 글의 제목 리스트만, 제목을 클릭하면 특정 게시글이 보여야 한다.

어디어디를 고쳐야 할까?

(url / view / template)

이 정도는 우리가 할 수 있다.

1. 글 리스트만 나타내기

2. 글 제목을 클릭할 수 있도록 만들기

Home.html로 가보자

(~~text에 해당하는 부분만 지워주기 + a태그 달아주기~~)

이 정도는 우리가 할 수 있다.

home

멋쟁이사자처럼

March 28, 2019, 10:08 a.m.

오늘 강의

March 28, 2019, 3:45 p.m.

Urls.py

```
from django.contrib import admin
from django.urls import path
import blog.views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('blog/home', blog.views.home, name='home'),
    path('blog/post/<int:post_id>/', blog.views.detail, name='detail'),
```

특정 게시글마다 html을 따로 만들어 줄 수 없으니
url로 구분 짓자!

views.py

```
from django.shortcuts import render, get_object_or_404, redirect
from django.utils import timezone
from .models import Post
from .forms import PostForm

# Create your views here.
def home(request):
    posts = Post.objects
    return render (request, 'blog/home.html', {'posts':posts})

def detail(request, post_id):
    post_detail = get_object_or_404(Post, pk = post_id)
    return render(request, 'blog/detail.html',{'post': post_detail})
```

Get_object_or_404

object를 가져오고 없으면 404 에러를 띄우라는 내용의
함수

home.html

```
<head>
  <title> My first blog </title>
</head>
<body>
  <h1>home</h1>
  {% for post in posts.all %}
    <div class="container">
      <h2><a href="post/{{post.id}}/">{{post.title}}<a></h2>
      <p>{{post.created_date}}</p>
    </div>
  {% endfor %}
</body>
```

제목을 누르면

해당 url로 넘어 갈 수 있도록 설정해줍니다

detail.html

- templates
- blog
 - detail.html**
 - home.html

```
<div>  
  <h1>{{ post.title }}</h1>  
  <p>{{ post.published_date }}</p>  
  <p>{{ post.text }}</p>  
  <a href="{% url 'home' %}">홈으로</a>  
</div>
```

이젠 해당 글이 보이는 페이지가 필요하겠죠?

이 페이지에는 글의 제목, 내용, 작성 시간 등을 나타낼겁니다 ☺

home

멋쟁이사자처럼

March 28, 2019, 10:08 a.m.

오늘 강의

March 28, 2019, 3:45 p.m.

← → ↺ ⓘ 127.0.0.1:8000/blog/post/1/

멋쟁이사자처럼

March 28, 2019, 10:12 a.m.

충북대학교 멋쟁이사자처럼 너무너무 좋당

← → ↺ ⓘ 127.0.0.1:8000/blog/post/2/

오늘 강의

March 28, 2019, 3:46 p.m.

괜찮을까 모르겠네..

이젠 제목을 클릭하면 해당 글의 내용만 볼 수 있도록 만들 수 있게 되었습니다!

지금까지는 관리자 페이지에서 글 작성

이젠 직접 웹 페이지에서 글을 작성해보자

< Form >

정의 : 웹 페이지상에서 한 개 이상의 필드나 위젯들의 묶음

사용 : 사용자로부터 정보를 수집하여 서버에 제출

HTML에서 적어도 한 개 이상의 `type="submit"`인 `input` 요소를 포함

하는 `<form>...</form>` 태그 사이의 요소들의 집합으로 정의

작성해보자 form

- blog
 - __pycache__
 - migrations
 - templates
 - blog
 - detail.html
 - home.html
- __init__.py
- admin.py
- apps.py
- forms.py
- models.py
- tests.py
- views.py

```
from django import forms
from .models import Post

class PostForm(forms.ModelForm):
    class Meta:
        model = Post
        fields = ('title', 'text', 'author',)

# class PostForm(forms.Form):
#     title = forms.CharField(label='제목', max_length=100)
#     text = forms.CharField(label='제목', widget=forms.Textarea)
```

밑에 주석 부분도 같이 쳤으면 좋겠어요

```
from django import forms
from .models import Post

class PostForm(forms.ModelForm):
    class Meta:
        model = Post
        fields = ('title', 'text', 'author',)

# class PostForm(forms.Form):
#     # title = forms.CharField(label='제목', max_length=100)
#     # text = forms.CharField(label='제목', widget=forms.Textarea)
```

Author:

Text:

Title:

저장

폼 형태

위 : 만들어 놓았던 Model에 종속적인 form을 만들어

편하게 작성 (model에서 이미 필드 타입 정의)

밑 : 내가 직접 필드 설정해서 작성

Form 을 만들었으니, form을 쓰시다.

사용자가 보는 화면에 무언가 보이게 하고 싶으면?

(url / view / template)

자 그럼 글쓰기 버튼을 넣어줄까요?

```
<head>
  <title> My first blog </title>
</head>
<body>
  <h1>home</h1>
  <a href="{% url 'new'%}" class="top-menu">글쓰기</a>
  {% for post in posts.all %}
    <div class="container">
      <h2><a href="post/{{post.id}}/">{{post.title}}<a></h2>
      <p>{{post.created_date}}</p>
    </div>
  {% endfor %}
</body>
```

방식 : POST

{% csrf_token %} : 보안을 위해 작성

장고 뿐 아니라 최신의 웹 프레임워크들은 모두 CSRF를 위한 별도의 방어 방법들을 제공

```
<head>
</head>
<body>
  <h1>작성해보자 글</h1>
  <form method="POST" class="post-form">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit" class="btn btn-secondary">저장</button>
  </form>
</body>
```


template을 작성했으니, template를 이어주려면?.

(url / view / template)

```
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path('blog/home', blog.views.home, name='home'),  
    path('blog/post/<int:post_id>', blog.views.detail, name='detail'),  
    path('blog/post/new/', blog.views.post_new, name='new'),  
]
```

이제 벌써 url 작성 3번째

조금 감은 오죠?

여기서 view에 있는 post_new에 요청을 했으니,

post_new를 만들어 가볼까요

```
from django.shortcuts import render, get_object_or_404, redirect
from django.utils import timezone
from .models import Post
from .forms import PostForm
```

```
def post_new(request):
    if request.method == "POST":
        form = PostForm(request.POST)
        if form.is_valid():
            post = form.save(commit=False)
            post.published_date = timezone.datetime.now()
            post.save()
            return redirect('detail', post_id=post.pk)
    else:
        form = PostForm()
        return render(request, 'blog/new.html', {'form': form})
```

필요한 것들을 import하는 거 잊지 말아주세요!

```
def post_new(request):  
    if request.method == "POST":  
        form = PostForm(request.POST)  
        if form.is_valid():  
            post = form.save(commit=False)  
            post.published_date = timezone.datetime.now()  
            post.save()  
            return redirect('detail', post_id=post.pk)  
        else:  
            form = PostForm()  
            return render(request, 'blog/new.html', {'form':form})
```

POST 방식이면

- Form에 아까 우리가 작성했던 form을 넣어주고
- 그 form에 값이 있다면
 - Form을 저장해서 post에 넣어주고
 - 시간 설정
 - Post저장
 - Redirect : 바로 간다!

아니면

- form에 postform() 넣어주고
- 다시 그 페이지로 render

render vs redirect

```
return render(request, 'blog/new.html', {'form':form})
```

```
return redirect('detail', post_id=post.pk)
```

보내는 인자가 3개

보내는 인자가 2개

Html로 넘겨줄 때 파일 경로 작성

Html로 넘겨줄 때 url name 작성

필요한 재료를 이렇게 저렇게 모아서

쿨하게 그냥 주소 던짐

짤! 하고 보여줌

이제 확인해보자!

```
$ python manage.py runserver
```

ㄱ ㄱ

글쓰기를 눌러 글을 작성하고 저장해보세요!

진짜 너무 신기하다

오늘의 목표였던 것들

블로그를 만들어보자!

이제부터가 진짜 시작

1. project / app 생성 → setting 설정
→ 데이터베이스 생성 → 개발 서버 실행
2. Model 생성 → 데이터베이스에 적용
3. 장고 관리자 생성 → 관리자 페이지에서 글 작성
4. url 생성 (include) → view 작성 → template 생성/작성
→ Post 불러오기 → 특정 글 불러오기 (view+ template + url)
5. Form 만들기 → template 작성 → url 작성 →
view작성 → 글 생성

그리고 git

~~아직 조금 더 남았어요~~

간단하게만 배워보자 git!

우리가 만든 프로젝트를 github에 올려보기

Git 이란?

프로그램 등의 소스 코드 관리를 위한

분산 버전 관리 시스템이다.

- 위키백과 -

어려운 말 : 분산 버전 관리 시스템

쉬운 말 : 여러 명의 개발자(분산)가 특정 프로젝트를

자신의 컴퓨터로 협업, 개발하면서

버전을 관리할 수 있는 시스템

사용자 정보

사용자 이름과 이메일을 설정합니다.
(한번만 설정해두면 됩니다.)

```
$ git config --global user.name "<이름>"
```

```
$ git config --global user.email <main@example.com>
```

자주 쓰이는 명령어

`git init`

실행한 위치를 Git저장소로 초기화

`git add 파일이름`

해당 파일을 Git이 추적할 수 있게 저장소에 추가

`git commit`

변경된 파일을 저장소에 제출


`git status`




현재 저장소의 상태를 출력








`git push`

저장소에 업로드





여기에서 볼 수 있습니다!

 **taem98** / **cbnu-likelion_lecture-material**

 Watch ▾ 0  Star 0  Fork 0

 Code  Issues 0  Pull requests 0  Projects 0  Wiki  Insights  Settings


Quick setup — if you've done this kind of thing before

 Set up in Desktop or  HTTPS  SSH `https://github.com/taem98/cbnu-likelion_lecture-material.git` 

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).


...or create a new repository on the command line

```
echo "# cbnu-likelion_lecture-material" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/taem98/cbnu-likelion_lecture-material.git
git push -u origin master
```



...or push an existing repository from the command line

```
git remote add origin https://github.com/taem98/cbnu-likelion_lecture-material.git
git push -u origin master
```



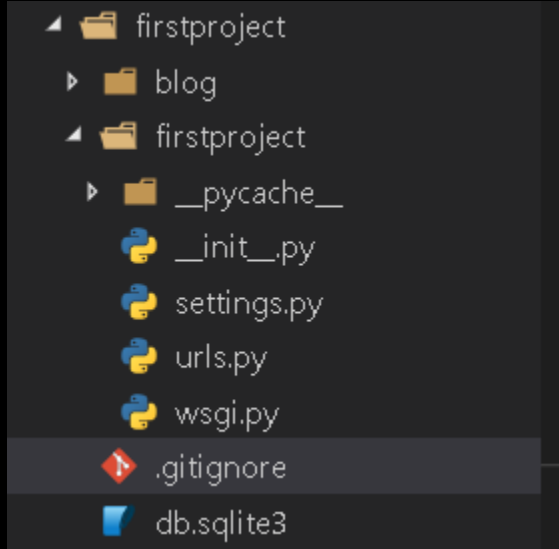
Github에 필요 없는 파일이 올라가는 것이 싫다!

<https://gitignore.io/> ← 접속하세요!

Django 치고 생성 버튼

무슨 말인진 모르겠지만 나오는 애들 전체 복사!

Github에 필요 없는 파일이 올라가는 것이 싫다!



프로젝트 파일 안에

.gitignore 파일을 만들어주고

그 안에 붙여 넣기를 하고 저장을 합니다!

일단 하라는 대로 따라해 볼까요?

명령어를 하나씩 입력해봅시다!

```
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/taem98/cbnu-likelion_lecture-material.git
git push -u origin master
```

README.md 대신 .(점)을 쳐주세요!

점은 모든 파일을 의미합니다.

```
$ git init  
Reinitialized existing Git repository in C:/Users/박태미/Desktop/likelion_test/firstproject/.git/  
(myvenv)
```

```
$ git add .  
(myvenv)
```


```
$ git commit -m "first commit"  
[master (root-commit) 255d876] first commit  
26 files changed, 268 insertions(+)  
create mode 100644 blog/__init__.py  
create mode 100644 blog/__pycache__/__init__.cpython-37.pyc  
  
create mode 100644 blog/__pycache__/admin.cpython-37.pyc  
create mode 100644 blog/__pycache__/models.cpython-37.pyc
```




```
$ git remote add origin https://github.com/taem98/cbnu-likelion_lecture-material.git  
(myvenv)
```








```
$ git push -u origin master  
Enumerating objects: 34, done.  
Counting objects: 100% (34/34), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (32/32), done.  
Writing objects: 100% (34/34), 15.03 KiB | 496.00 KiB/s, done.  
Total 34 (delta 2), reused 0 (delta 0)
```



다시 github로 가서 새로 고침을 하면

이렇게 파일들이 올라가 있습니다!





 **taem98 / cbnu-likelion_lecture-material**






 Watch **0**  Star **0**  Fork **0**


 Code  Issues **0**  Pull requests **0**  Projects **0**  Wiki  Insights  Settings





CR강의자료 


[Manage topics](#)

 **1** commit  **1** branch  **0** releases  **1** contributor

Branch: master     

 **taem98** first commit Latest commit 255d876 21 minutes ago

 blog	first commit	21 minutes ago
 firstproject	first commit	21 minutes ago
 db.sqlite3	first commit	21 minutes ago
 manage.py	first commit	21 minutes ago

Help people interested in this repository understand your project by adding a README. 

Github에 프로젝트 올리기도 성공!

과제

1. 앱 하나 더 만들기

(오늘 만들었던 블로그 + 자기소개 앱 만들기)

조건 : url에 include를 써서 앱마다 url 나누기

! 참고 !

생성하는 앱이 2개가 되면서 url 관리가 힘들어질 수 있으니

앱마다 url을 관리해보도록 합시다.

일단 앱마다 urls.py 폴더를 생성해줄까요?

1. 블로그 앱에 urls.py
2. 자기소개 앱에 urls.py

힌트 아닌 힌트 다 퍼줌

```
"""  
from django.contrib import admin  
from django.urls import path, include  
import blog  
  
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path('blog/', include('blog.urls')),  
]
```

프로젝트 urls.py

```
from django.contrib import admin  
from django.urls import path  
from blog import views
```

블로그 urls.py

```
urlpatterns = [  
    path('home/', views.home, name='home'),  
    path('home/post/<int:post_id>/', views.detail, name='detail'),  
    path('home/post/new/', views.post_new, name='new'),  
]
```

이용해서 자기소개 앱에도 이런식으로
url을 관리해주세요

+ base 편하게 깔고 가기

오늘 그럼 정말 끝!

충북대학교 멋쟁이사자처럼 7기 운영진분들, 학생분들

모두 너무 수고하셨습니다 😊