Django advanced project #week_1

Writer : Taeyeong Kim, DGIST LikeLion

Date: 2019.10.09

아래 자료를 공부하기 전에, https://github.com/lizard-kim/LikeLion_study_Fall_Semester에 있는 코드를 clone하여 참고하기 바란다. 코드에 대한 자세한 정보는 주석을 통해 알 수 있으니, 이곳에서는 대략적인 설명만 할 것이다.

What is JSON?

데이터의 송수신을 JS 객체로서 수행할 수 있게끔 하는 가벼운 **문자열** 데이터 표현식.

이제는 JSON을 사용하여 Django Restful API Server와 **DATA만** 주고 받을 것이다.

> 우리는 이전에 Django web application을 개발할 때는 HTML, CSS, JS를 주고 받았다.
>
> Django Restful API Server는 뒤에서 자세히 설명할 것이다.

JSON은 어떻게 생겼을까?

```
json_example =
{
"string_name": "something",
"number_name": 3,
"null_name": null,
"bool_name": true
}

// name : value로 매핑
// 숫자, 문자(열), Boolean, 배열,객체 올 수 있음
```

JS와 똑같이 생겼다. 하지만 JS와 JSON을 어떻게 구별할까?

이를 구별하기 위해 우리는 문자열로 바꾸어 전송한다. 이를 **serialization**이라고 한다.

JSON을 python에서 어떻게 사용하는지 알아보기 위해 1week/json.py 코드를 실행시켜보자.

Class Based View(CBV)

이름에서 알 수 있듯이, views.py를 함수가 아닌, class를 통해 구현하는 방법이다. class로 view를 구현하면 class상속을 통해 얻을 수 있는 이점이 있다.

Class 상속

Class는 함수와 다르게 상속이 가능하다. 상속을 통해 부모클래스의 코드를 재활용(?)할 수 있어 함수로 모든 것을 코드로 작성해주는 작업에서 해방될 수 있다. 즉, 코드가 간단해진다.

> class 상속에 대한 개념을 까먹었다면 아래 reference를 활용하자.
>
> https://suwoni-codelab.com/python%20%EA%B8%B0%EB%B3%B8/2018/03/12/Python-Basic-class-inheritance/

이제 만들어보자!

먼저 프로젝트를 생성하자.

```
$ django-admin startproject crud
$ python manage.py startapp classcrud # 이후 settings.py에 app을 추가해주자
```

models.py

다음과 같이 model.py에 model을 정의해주자.

```
from django.db import models

# python manage.py makemigrations : python code --> DB
# python manage.py migrate : accept my model to DB

class ClassBlog(models.Model):
    title = models.CharField(max_length=100)
    created_at = models.DateTimeField(auto_now_add=True) # auto input
    updated_at = models.DateTimeField(auto_now=True) # auto input
    body = models.TextField()

    def __str__(self): # make class data to text(string) / in admin page, we can see titles by this method
        return self.title

# after this step, we should accept this model to admin.py
```

admin.py

admin에서 title을 볼 수 있게 작성하였으니, admin도 생성해주도록 하자.

```
from django.contrib import admin
from .models import ClassBlog # bring my models

admin.site.register(ClassBlog) # accept!
```

> superuser만드는 것도 까먹지 말자!

crud/urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', classcrud.views.home, name="home"),
    path('classcrud/', include(classcrud.urls)),
]
```

url을 classcrud에 분리하여 작성하기 위해 include를 사용해주자.

classcrud/urls.py

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

urlpatterns = [

    # Blog~~ is class!
    # .as_view() is a method.
    # the second parameter of path is function. So we can`t input class in
second params.

    path('', views.BlogView.as_view(), name='list'),
    path('newblog/', views.BlogCreate.as_view(), name='new'),
    path('detail/<int:pk>', views.BlogDetail.as_view(), name='detail'),
    path('update/<int:pk>', views.BlogUpdate.as_view(), name='change'),
    path('delete/<int:pk>', views.BlogDelete.as_view(), name='del'),

]
```

path 함수의 두번째 인자로는 함수가 들어간다. 따라서 views에 정의한 클래스에 `as_view()` method를 사용하여 클래스를 함수로 정의해주자.

views.py

```
from django.shortcuts import render
from django.utils import timezone

from django.urls import reverse_lazy # redirection function
from django.views.generic.list import ListView # show data
from django.views.generic.detail import DetailView
from django.views.generic.edit import CreateView, UpdateView, DeleteView # add
data
from .models import ClassBlog

def home(request):
```

```
    return render(request, 'home.html')

class BlogView(ListView):
    # ListView 클래스를 상속, 객체들의 목록을 가져온다 /  html 템플릿 필요 : 블로그 리스트를 담은 html : (소문자모델)_list.html
    #template_name = 'classcrud/exampleName.html' //템플릿의 html파일 이름을 원하는 것으로 하고 싶을 때 직접 지정하기
    #context_object_name = 'blog_list' //이제 object_list라고 html 에서 불러오는 것이 아닌, blog_list로 불러온다.
    model = ClassBlog # 객체는 모델에서 정의한대로 입력된다.

class BlogCreate(CreateView):    # html : form(입력공간)을 갖고 있는 html : (소문자모델)_form.html
    model = ClassBlog
    fields = ['title', 'body'] # 우리가 수정할 부분
    success_url = reverse_lazy('list') # redirection. list = our url

class BlogDetail(DetailView):   # html : 상세 페이지를 담은 html : (소문자모델)_detail.html
    model = ClassBlog

class BlogUpdate(UpdateView):   # html : form(입력공간)을 갖고 있는 html 필요 : (소문자모델)_form.html
    model = ClassBlog
    fields = ['title', 'body']
    success_url = reverse_lazy('list')

class BlogDelete(DeleteView):   # html : 이거 진짜 지울거야??? 확인메시지 날리는 html : (소문자모델)_confirm_delete.html
    model = ClassBlog
    success_url = reverse_lazy('list')

# html 이름이 정해져 있다. 다르게하려면 추가적인 작업 필요.
# next step : making templates
```

각각의 class가 django가 제공해주는 class를 상속하여 간단히 구현된 것을 볼 수 있다(우리는 model과 field만 넘겨주면 된다!). 자세한 설명은 주석을 읽어보자.

git에서 cloning 한 자료를 살펴보면 templates의 이름이 길게 정해져 있는 것을 볼 수 있다. 주석을 보면 이해하겠지만, 장고에서 기본적으로 주어진 클래스를 상속하여 사용하기 때문에, html 파일이름을 정해진대로 사용해야 한다.

html에서 object를 어떻게 받아 사용하는지는 코드를 한번 읽어보면 쉽게 이해할 수 있을 것이다.

Run server

서버를 돌려 잘 작동하는지 확인해보자.

Http Request & Method

Client와 Server는 Http라는 규약 안에서 통신한다. Django rest framework에서는 다음과 같은 Method를 사용한다.

Method	Meaning
GET	요청받은 URI의 정보를 검색하여 응답한다.
POST	요청된 자원을 생성(CREATE)한다.
PUT	요청된 자원을 수정(UPDATE)한다.
DELETE	요청된 자원을 삭제한다
PATCH	요청된 자원의 일부를 교체(수정)한다
OPTION	웹서버에서 지원되는 메소드의 종류 확인

> 이밖에도 많은 Method가 존재한다. 일단, 가장 대표적인 6가지만 알고 넘어가자

이제 httpie를 통해서 위의 method가 어떻게 동작하는지 살펴보자. 아래의 코드를 터미널에 치면서 확인해보자.

Httpie install

먼저 httpie를 설치하자.

`$ pip install --upgrade httpie`

`$ http`

http명령어로 잘 설치되었는지 확인해보자. 명령어에 대한 설명이 나오면 잘 설치된 것이다.

Using httpie

httpie 명령어는 *http [flags] [METHOD] URL [ITEM[ITEM]]* 의 형식으로 이루어져 있다.

Get command

`$ http get example.com`

`$ http get :8000` 이 명령어는 로컬로 서버를 돌렸을 때 응답을 받을 수 있는 명령어이다.

`$ http get "http://127.0.0.1:8000/classcrud/"` 글의 목록을 볼 수 있는 url에서 정보를 받아왔다. url은 본인 서버의 주소를 사용하도록 하자.

Post command

`$ http --form post "http://127.0.0.1:8000/classcrud/newblog/" title="new title" body="new body"` 이 command는 서버에서 응답을 거부한다. (403 error) 이는 장고의 기본적인 보안관련 기능으로 인해 거부당하는 것이다.

Delete command

`$ http delete "http://127.0.0.1:8000/classcrud/detail/1"` 이 command도 같은이유로 거부당할 것이다.

더 많은 정보를 알고 싶으면 *https://httpie.org* 에 접속하여 공부해보자.

References

likelion 강의 소스코드 : https://github.com/kangtegong/django-RESTful-API

class 상속 관련 개념 : https://suwoni-codelab.com/python%20기본/2018/03/12/Python-Basic-class-inheritance/