14 클래스

14. 클래스

- **클래스**
  - 새로운 type을 만드는 것
  - 익숙한 클래스들 : int, str, list …

```
>>> a = int()
>>> type(a)
<class 'int'>
>>>
>>> b = str()
>>> type(b)
<class 'str'>
>>>
>>> c = list()
>>> type(c)
<class 'list'>
```

- **클래스가 좋은 이유**
  - 특정 클래스로 지정이 되면 개발자가 만들어 둔 기능을 자동으로 쓸 수 있게 됨.

- **현실에 비유해서 말해보면**
  - 규상 = 연대생()
    - 규상.학점 → 3.0
    - 규상.학년 → 4
    - 규상.캠퍼스 → 신촌
  - 형규 = 연대생()
    - 형규.학점 → 4.3
    - 형규.학년 → 3
    - 형규.캠퍼스 → 신촌

14. 클래스

- **파이썬 기본 클래스로 보는 attribute, method**
  - Int

```
>>> a = int(123)
>>> a
123
>>> a.real
123
>>> a.imag
0
```

  - str

```
>>> b = str('yonsei')
>>> b.upper()
'YONSEI'
>>> b.split('s')
['yon', 'ei']
```

```
>>> d = str('university')
>>> d.upper()
'UNIVERSITY'
>>> d.split('r')
['unive', 'sity']
```

  - 같은 클래스는 같은 기능(attribute, method)를 갖는다.

14. 클래스

- 클래스에 대한 비유

클래스

객체(인스턴스)

```
class 붕어빵:
    pass


붕1 = 붕어빵()
붕2 = 붕어빵()
붕3 = 붕어빵()
붕4 = 붕어빵()
```

14. 클래스

- 클래스 만들기 (1)

```
class Yccian:
    univ = 'yonsei'
    coding_skill = True
```

- 클래스 사용하기 (1)

```
>>> from answer import Yccian
>>> gyusang= Yccian()
>>> gyusang.univ
'yonsei'
>>> gyusang.coding_skill
True
>>>
>>> type(gyusang)
<class 'answer.Yccian'>
```

14. 클래스

- 클래스 만들기 (2)

```
class Yccian:
    univ = 'yonsei'
    coding_skill = True

    def open_door(self, password):
        if password == 2533:
            return "open"
        else:
            return "fail"
```

- 클래스 사용하기 (2)

```
>>> gyusang = Yccian()
>>> gyusang.open_door(2533)
'open'
>>> gyusang.open_door(1111)
'fail'
```

14. 클래스

- **클래스 상속**
  - 부모클래스의 기능들을 물려받아 사용 가능

  - 상속받지 않았을 때

```
>>> gyusang.split('s')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: type object 'Yccian' has no attribute 'split'
```

  - 상속받았을 때

```
class Yccian(str):
    univ = 'yonsei'
    coding_skill = True

    def open_door(self, password):
        if password == 2533:
            return "open"
        else:
            return "fail"
```

```
>>> from answer import Yccian
>>> gyusang = Yccian
>>> gyusang.split('s')
['s']
```

14. 클래스

- **Django에서의 models 클래스 이해하기**
  - models 를 상속받아 Post라는 클래스 생성

models.py

```python
from django.db import models

# Create your models here.

class Post(models.Model):
    title = models.CharField(max_length=100)
```

  - post_list 라는 변수를 Post 클래스로 지정하여 사용

```python
from django.shortcuts import render
from .models import Post

# Create your views here.

def list(request):
    post_list = Post.objects.all()
    return render(request, 'list.html', {
        'post_list':post_list,
    })
```

14. 클래스

- **연습문제**
  - Insect 라는 클래스 설계하기
  - 설계한 내용을 바탕으로 클래스 코딩하기
  - Insect 클래스 사용하기

14. 클래스

- **연습문제 답**
  - INSECT 라는 클래스 설계하기

    - 다리 : 6개
    - 구성 : head / thorax / abdomen
    - 알을 낳는다

  - 설계한 내용을 바탕으로 클래스 코딩하기

```
class Insect:
    num_leg = 6
    parts = ['head', 'thorax', 'abdomen']

    def lay_eggs(self, num):
        self.num_eggs = num
        return self.num_eggs
```

14. 클래스

- **연습문제 답**
  - Insect 클래스 사용하기

```
>>> from answer import Insect
>>> ant = Insect()
>>> ant.num_leg
6
>>> ant.parts
['head', 'thorax', 'abdomen']
>>> ant.num_eggs
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'Insect' object has no attribute 'num_eggs'
```

```
>>> ant.lay_eggs(10)
10
>>> ant.num_eggs
10
```

15 Post 분리

15. Post 분리

- urls.py 수정

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:num>/', views.post),
9  ]
```

15. Post 분리

- views.py 수정

views.py

```python
from django.shortcuts import render
from .models import Post

# Create your views here.

def list(request):
    post_list = Post.objects.all()
    return render(request, 'list.html', {
        'post_list':post_list
    })

def post(request, num):
    post_for_pk = Post.objects.get(pk=num)
    return render(request, 'post.html', {
        'post_for_pk':post_for_pk,
        'num':num
    })

```

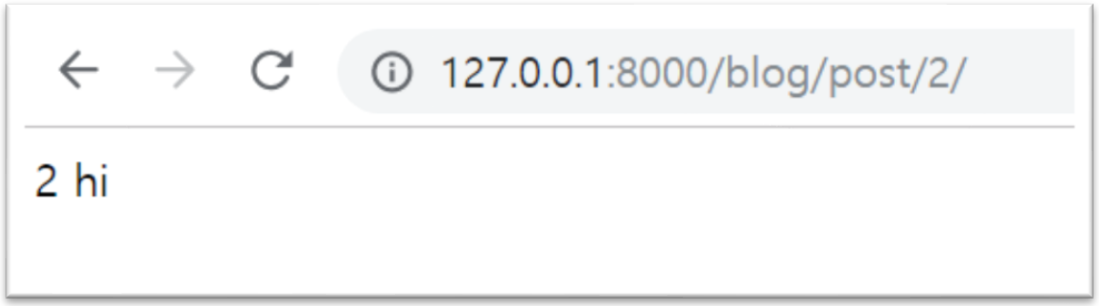
15. Post 분리

- post.html 파일 만들기

post.html

```
1  {{num}}
2
3
4  {{post_for_pk.content}}
```

- 결과



15. Post 분리

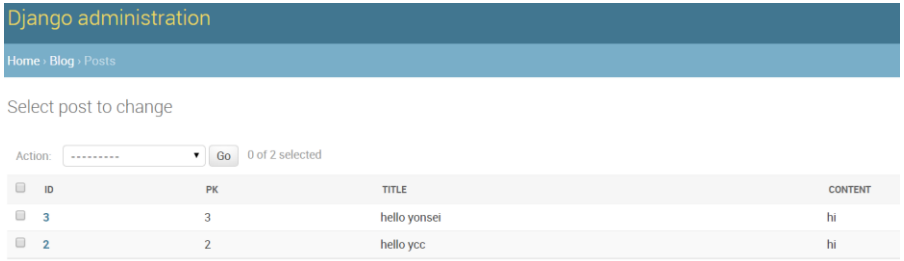
- pk
  - Primary key
- id
  - 아시잖아요 ㅎ
- **모델을 만들 때 자동으로 만들어짐**

models.py

```python
from django.db import models

# Create your models here.

class Post(models.Model):
    title = models.CharField(max_length=100)
    content = models.CharField(max_length=100)
```

admin.py

```python
from django.contrib import admin
from .models import Post

# Register your models here.

@admin.register(Post)
class PostAdmin(admin.ModelAdmin):
    list_display = ('id','pk','title', 'content')
```



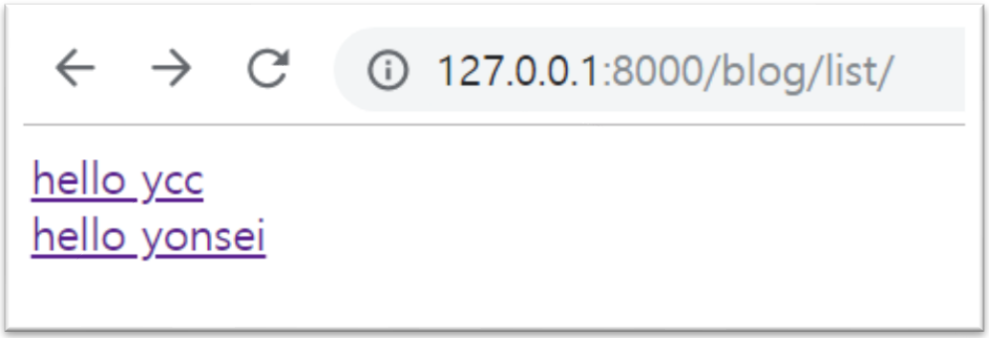
15. Post 분리

- list와 post 사이 링크 걸기

list.html

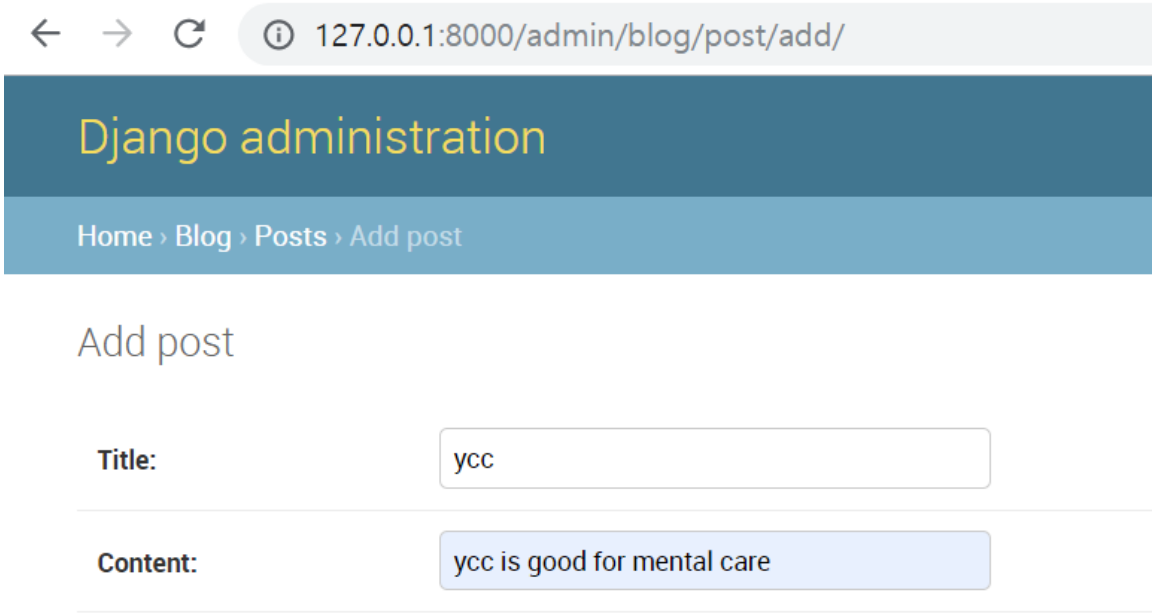
```
{% for post in post_list %}
  <a href = "http://127.0.0.1:8000/blog/post/{{post.pk}}">
  {{post.title}}
  </a>
  <br/>
{% endfor %}
```

- 결과

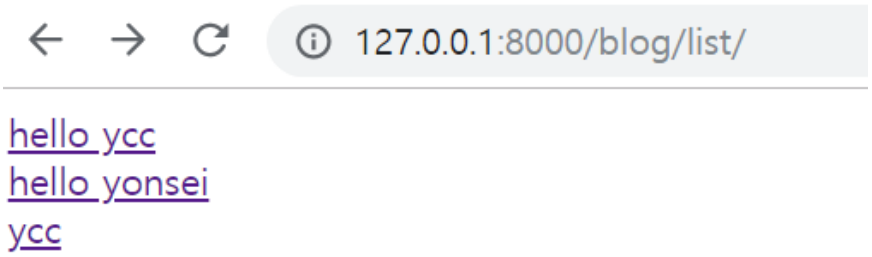


15. Post 분리

- 127.0.0.1:8000/admin으로 가서 포스트 추가하기



- 결과



16 Form

17 CSS

18 배포