MEDIA

유저들이 올리는 자료들을 Media라고 부른다.

•파일 내용이 아닌, 파일 경로를 저장한다.

FileField
• django storage api (디폴트: FileSystemStorage) 를 이용하여, 업로드된 파일을 저장

The FileSystemStorage Class

***class* FileSystemStorage(*location=None*, *base_url=None*, *file_permissions_mode=None*, *directory_permissions_mode=None*)** **[source]**

The **FileSystemStorage** class implements basic file storage on a local filesystem. It inherits from **Storage** and provides implementations for all the public methods thereof.

location

Absolute path to the directory that will hold the files. Defaults to the value of your **MEDIA_ROOT** setting.

base_url

URL that serves the files stored at this location. Defaults to the value of your **MEDIA_URL** setting.

file_permissions_mode

The file system permissions that the file will receive when it is saved. Defaults to **FILE_UPLOAD_PERMISSIONS**.

directory_permissions_mode

The file system permissions that the directory will receive when it is saved. Defaults to **FILE_UPLOAD_DIRECTORY_PERMISSIONS**.

Note

The **FileSystemStorage.delete()** method will not raise an exception if the given file name does not exist.

• upload_to 인자 : 저장할 파일 경로를 지정

• ex) "uploads/", "uploads/%Y/%m/%d" 혹은 함수를 지정

ImageField
• FileField 상속

• 이미지 처리 라이브러리 (Pillow) 설치가 필요하다.
-원래 PIL(Python Image Library)가 있었는데, 7~8년째 업그레이드가 되지 않아서 어떤 사람이 Pillow를 만듦. PIL처럼 동작하기 때문에 편리하다.
-고성능의 이미지 처리 라이브러리는 아니다.
-서버는 리눅스를 쓰면서 고성능의 이미지 처리를 하고 싶다면 Image Magick + wand를 사용하자.

• 업로드받을 때, 이미지 여부를 검사.
• width/height 속성 지원

*** html form 의 enctype 을 "multipart/form-data" 로 지정하지 않으면, 파일내용은 전송되지 않고, 파일명만 전송된다.
ex) <form action='' method= 'post' enctype= 'multipart/form-data'>

클라이언트에서 서버에게 오는 것은 크게 세 가지다.

request.GET - 파라미터를 주소에 실어온다.

request.POST - <QueryDict: {'photo': [''], 'is_agree': ['on'], 'comments_num': ['0'], 'phone': [''], 'categories': ['1'], 'csrfmiddlewaretoken': ['Bij406Ng6KbMWahG31JheQ5Odm10ieyE'], 'title': ['asfasgag'], 'tags': ['1'], 'content': ['asgsag']}>

request.FILES - <MultiValueDict: {'photo': [<InMemoryUploadedFile: profileImg.jpg (image/jpeg)>]}>

• settings.py

MEDIA_ROOT = os.path.join(BASE_DIR, "media")
업로드 된 파일들이 저장될 경로

MEDIA_URL = '/media/'
파일을 업로드한 후 링크를 눌러보면 다음과 같이 나온다. http://localhost:8000/media/profileImg.jpg . 이는 static 파일들도 마찬가진데, Django는 static 서빙은 자동적으로 해주지만 media는 그렇지 않기 때문에 urlpattern을 추가해줘야한다.

• blog/models.py
from programming.utils import random_name_upload_to

class Post(models.Model):
photo = models.ImageField(blank=True, upload_to = random_name_upload_to)

만약 blank = True가 아니라면, migrate를 할 때 디폴트 값을 지정해달라고 하는데, 이 때 입력해야하는 것은 **파일의 경로다.
**upload_to를 지정해주면 media/blog/post 디렉토리에 업로드된다. 그런데 한 디렉토리에 너무 많은 파일을 몰아 넣으면 좋지 않기 때문에 upload_to='%Y/%m/%d' 식으로 매일 디렉토리가 생성되게 할 수도 있다. timezone.now().strftime('%Y/%m/%d')가 내부적으로 실행되어 2016/01/29 디렉토리가 생성된다.

**이보다 더 세부적으로, 그리고 파일명도 랜덤하게 저장해주려면 새로운 함수를 생성하자. 파일명에 한글이 있다면 인코딩 문제가 발생할 수도 있기 때문에 바꾸는게 바람직하다. 장고 프로젝트 전체에서 쓸 수 있게 모듈화하면 더 좋다.

programming/programming/utils.py

import os
from uuid import uuid4
from django.utils import timezone

def random_name_upload_to(instance, filename):
app_label = instance.__class__._meta.app_label
cls_ame = instance.__class__.__name__.lower()
ymd_path = timezone.now().strftime('%Y/%m/%d')
name = uuid4().hex
extension = os.path.splitext(filename)[-1].lower()
return os.path.join(app_label, cls_name, ymd_path,name+extension)

• urls.py

from.django.conf import settings
from django.conf.urls.static import static
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

print(static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)) 해보면
[<RegexURLPattern None ^media\/(?P<path>.*)$>] 이라는 리스트가 나온다. 이를 list의 extend 메소드를 통해 urlpatterns에 추가하는 것이 '+'.

이렇게 매칭되는 url을 추가해주면 이제 media 파일들도 서빙 가능하다.

Serving files uploaded by a user during development.

During development, you can serve user-uploaded media files from **MEDIA_ROOT** using the **django.contrib.staticfiles.views.serve()** view.

This is not suitable for production use! For some common deployment strategies, see Deploying static files.

For example, if your **MEDIA_URL** is defined as **/media/**, you can do this by adding the following snippet to your urls.py:

```
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    # ... the rest of your URLconf goes here ...
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Note

This helper function works only in debug mode and only if the given prefix is local (e.g. **/media/**) and not a URL (e.g. **http://media.example.com/**).

• views.py

```
def post_new(request):
    if request.method == "POST":
        form = PostForm(request.POST, request.FILES)
        if form.is_valid():
            post = form.save()
            return redirect('blog.views.post_detail', post.pk)
    else:
        form = PostForm()
    return render(request, 'blog/post_form.html',{
        'form':form,
        })
```

이제는 파일(이미지)를 폼을 통해서 제출하기 때문에 PostForm의 인자로 request.FILES를 같이 보내줘야한다.

• post_detail.html

```
{% if post.photo %}
        <img src = '{{post.photo.url}}' />
        <li>{{post.photo.url}}</li>
        <li>{{post.photo.path}}</li>
{% endif %}
```

post.photo에는 파일의 경로가 저장되어 있다.(이미지를 제출한 경우) post.photo == True 이기 때문에 if문이 실행되고,
-- 이미지 --
- /media/profileImg_ogt65ch.jpg
- /Users/Prodia/dev/programming/media/profileImg_ogt65ch.jpg
가 출력된다.
