

# 장고 차근차근 시작하기

Second Edition

당신의 파이썬/장고 페이스메이커가 되겠습니다. ;)

EP26. media 파일을 다루는 방법

# Static & **Media** 파일

- Static 파일
  - 개발 리소스로서의 정적인 파일 (js, css, image 등)
  - 앱 / 프로젝트 단위로 저장/서빙
- **Media 파일**
  - FileField/ImageField를 통해 저장한 모든 파일
  - DB필드에는 저장경로를 저장하며, 파일은 파일 스토리지에 저장
    - 실제로 문자열을 저장하는 필드 (중요)
  - 프로젝트 단위로 저장/서빙

# Media 파일

# Media 파일 처리 순서

1. HttpRequest.FILES를 통해 파일이 전달
2. 뷰 로직이나 폼 로직을 통해, 유효성 검증을 수행하고,
3. Field/ImageField 필드에 "경로(문자열)"를 저장하고,
4. settings.MEDIA\_ROOT 경로에 파일을 저장합니다.

# Media 파일, 관련 settings 예시 ([공식문서](#))

- 각 설정의 디폴트 값
  - MEDIA\_URL = ""
    - 각 media 파일에 대한 URL Prefix
      - 필드명.url 속성에 의해서 참조되는 설정
  - MEDIA\_ROOT = ""
    - 파일필드를 통한 저장 시에, 실제 파일을 저장할 ROOT 경로

# 추천 settings

```
MEDIA_URL = '/media/'
```

```
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

# FileField와 ImageField

- FileField
  - File Storage API를 통해 파일을 저장
    - 장고에서는 File System Storage만 지원. django-storages를 통해 확장 지원.
  - 해당 필드를 옵션 필드로 두고자 할 경우, blank=True 옵션 적용
- ImageField (FileField 상속)
  - Pillow (이미지 처리 라이브러리)를 통해 이미지 width/height 획득
    - Pillow 미설치 시에, ImageField를 추가한 makemigrations 수행에 **실패**합니다.
- 위 필드를 상속받은 커스텀 필드를 만드실 수도 있습니다.
  - ex) PDFField, ExcelField 등

# 모델 필드 예시

```
class Post(models.Model):  
    author_name = models.CharField(max_length=20)  
    title = models.CharField(max_length=100)  
    content = models.TextField()  
    photo = models.ImageField(blank=True)  
    created_at = models.DateTimeField(auto_now_add=True)  
    updated_at = models.DateTimeField(auto_now=True)
```



# 사용할 만한 필드 옵션

- blank 옵션
  - 업로드 옵션처리 여부
  - 디폴트: False
- upload\_to 옵션
  - settings.MEDIA\_ROOT 하위에서 저장한 파일명/경로명 결정
  - 디폴트 : 파일명 그대로 settings.MEDIA\_ROOT 에 저장
    - 추천) 성능을 위해, 한 디렉토리에 너무 많은 파일들이 저장되지 않도록 조정하기
  - 동일 파일명으로 저장 시에, 파일명에 더미 문자열을 붙여 파일 덮어쓰기 방지

# 파일 업로드 시에 HTML Form enctype

- form method는 필히 POST로 지정
  - GET의 경우 enctype이 "application/x-www-form-urlencoded"로 고정
- form enctype을 필히 "multipart/form-data"로 지정
  - "application/x-www-form-urlencoded"의 경우, 파일명만 전송 ☹

```
<form action="" method="post" enctype="multipart/form-data">
  {% csrf_token %}
  <table>
    {{ form.as_table }}
  </table>
  <input type="submit" />
</form>
```

# upload\_to 인자

- 파일 저장 시에 upload\_to 함수를 호출하여, 저장 경로를 계산
  - 파일 저장 시에 upload\_to 인자를 변경한다고 해서, DB에 저장된 경로 값이 갱신되진 않습니다.
- 인자 유형
  - 문자열로 지정
    - 파일을 저장할 "중간 디렉토리 경로"로서 활용
  - 함수로 지정
    - "중간 디렉토리 경로" 및 "파일명"까지 결정 가능

# 파일 저장경로

- travel-20181225.jpg 파일을 업로드할 경우
  - MEDIA\_ROOT/travel-20181225.jpg 경로에 저장되며,
  - DB에는 "travel-20181225.jpg" 문자열을 저장합니다.

# 파일 저장 경로 / 커스텀

upload\_to 옵션

- 한 디렉토리에 파일을 너무 많이 몰아둘 경우, OS 파일찾기 성능 저하. 디렉토리 Depth가 깊어지는 것은 성능에 큰 영향 없음.
- 필드 별로, 다른 디렉토리 저장경로를 가지기
  - 대책 1) 필드 별로 다른 디렉토리에 저장
    - `photo = models.ImageField(upload_to="blog")`
    - `photo = models.ImageField(upload_to="blog/photo")`
  - 대책 2) 업로드 시간대 별로 다른 디렉토리에 저장
    - `upload_to`에서 strftime 포매팅을 지원
    - `photo = models.ImageField(upload_to="blog/%Y/%m/%d")`

# uuid를 통한 파일명 정하기 예시

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

def uuid_name_upload_to(instance, filename):
    app_label = instance.__class__.__meta__.app_label      # 앱 별로
    cls_name = instance.__class__.__name__.lower()        # 모델 별로
    ymd_path = timezone.now().strftime('%Y/%m/%d')         # 업로드하는 년/월/일 별로
    uuid_name = uuid4().hex
    extension = os.path.splitext(filename)[-1].lower()     # 확장자 추출하고, 소문자로 변환
    return '/'.join([
        app_label,
        cls_name,
        ymd_path,
        uuid_name[:2],
        uuid_name + extension,
    ])
])
```

# 템플릿에서 media URL 처리 예시

- 필드의 **.url** 속성을 활용하세요.
  - 내부적으로 settings.MEDIA\_URL과 조합을 처리

```

```
  - 필드에 저장된 경로에 없을 경우, .url 계산에 실패함에 유의. 그러니 안전하게 필드명 저장유무를 체크

```
{% if post.photo %}  
      
{% endif %}
```
- 참고
  - 파일 시스템 상의 절대경로가 필요하다면, **.path** 속성을 활용하세요.
    - settings.MEDIA\_ROOT와 조합

# 개발환경에서의 media 파일 서빙

- static 파일과 다르게, 장고 개발서버에서 서빙 미지원
- 개발 편의성 목적으로 직접 서빙 Rule 추가 가능

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

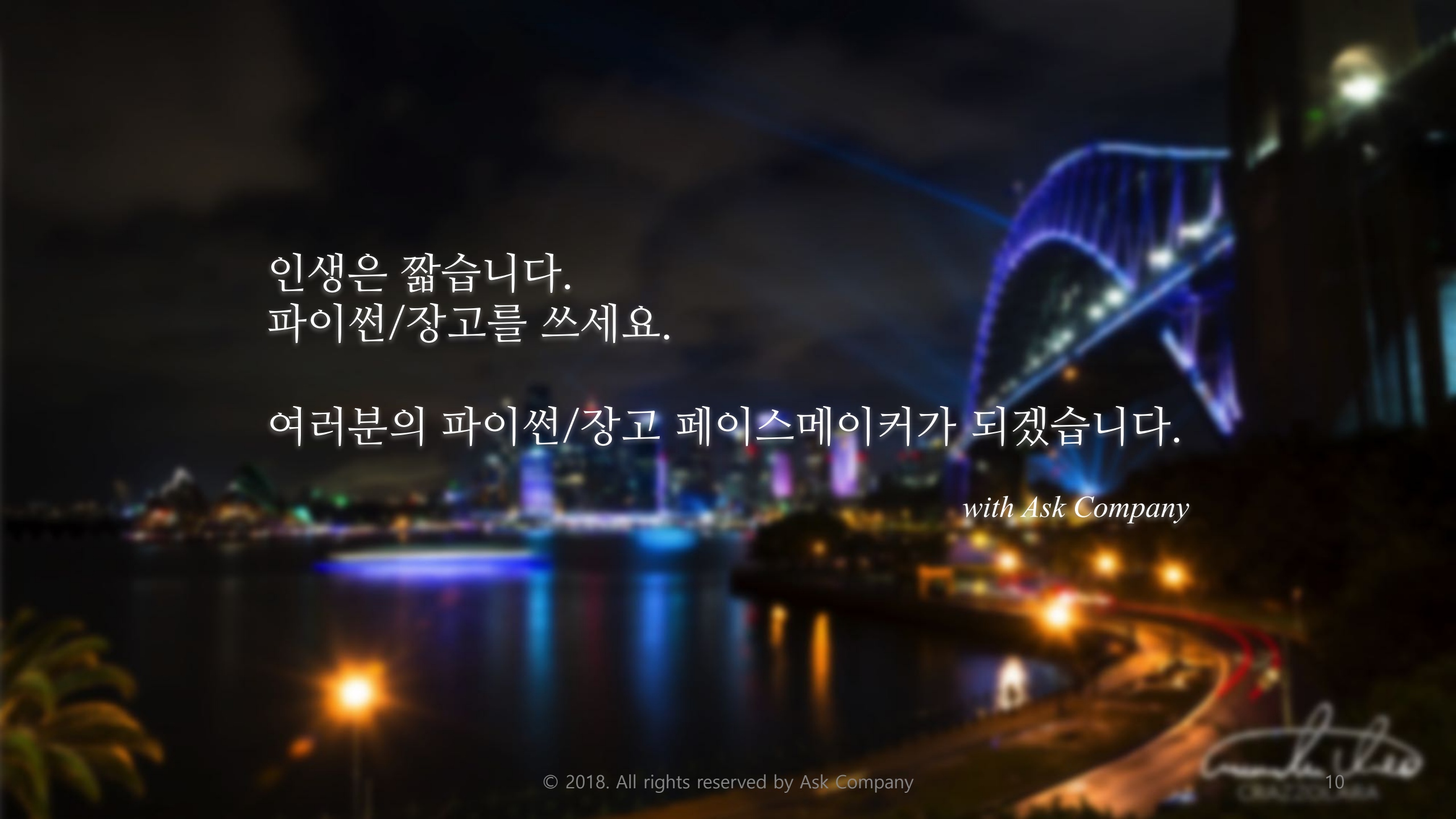
# 중략

urlpatterns += static(settings.MEDIA_URL,
                      document_root=settings.MEDIA_ROOT)
```



# File Upload Handler

- 파일크기가 2.5MB 이하일 경우
  - 메모리에 담겨 전달
  - MemoryFileUploadHandler
- 파일크기가 2.5MB 초과일 경우
  - 디스크에 담겨 전달
  - TemporaryFileUploadHandler
- 관련 설정
  - settings.FILE\_UPLOAD\_MAX\_MEMORY\_SIZE
    - ➔ 2.5MB

A nighttime photograph of a cityscape featuring a bridge with blue and white lights. The lights are reflected in the water below. The sky is dark, and the overall scene is illuminated by the city lights.

인생은 짧습니다.  
파이썬/장고를 쓰세요.

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

*with Ask Company*