

[과제 6] Request 객체 실습 _

`request` 객체는 Django에서 클라이언트(브라우저)의 요청 정보를 담고 있는 핵심 객체. 뷰 함수에서 가장 많이 쓰이며, 사용자가 보낸 데이터, 요청 방식, 로그인 정보 등을 확인할 수 있게 함.

✓ 기본 구조

Django에서 모든 뷰 함수는 첫 번째 인자로 `request` 객체를 받습니다.

```
def my_view(request):  
    ...
```

이 `request` 는 `HttpRequest` 객체이며, 사용자가 보낸 HTTP 요청(POST, GET 등)의 모든 정보를 포함하고 있습니다.

🧩 주요 속성과 기능

1. `request.method`

- 요청 방식 (문자열)
- 예: `'GET'`, `'POST'`, `'PUT'`, `'DELETE'`

```
if request.method == 'POST':  
    ...
```

2. `request.GET`

- `GET` 방식으로 전달된 쿼리 파라미터 (딕셔너리처럼 사용)
- 예: `/search/?q=hello` → `request.GET['q'] == 'hello'`

```
query = request.GET.get('q', '') # 기본값 "" 설정
```

3. `request.POST`

- `POST` 방식으로 전달된 폼 데이터

```
username = request.POST.get('username')
password = request.POST.get('password')
```

4. `request.FILES`

- 업로드된 파일 정보

```
uploaded_file = request.FILES['myfile']
```

5. `request.user`

- 현재 로그인한 사용자 객체 (`User` 또는 `AnonymousUser`)
- `is_authenticated` 로 로그인 여부 확인 가능

```
if request.user.is_authenticated:
    print(request.user.email)
```

6. `request.session`

- 서버 측 세션 데이터 (로그인 상태 유지, 장바구니 등)

```
request.session['cart'] = ['item1', 'item2']
```

7. `request.COOKIES`

- 클라이언트가 보낸 쿠키 정보

```
theme = request.COOKIES.get('theme')
```

8. `request.META`

- HTTP 헤더 및 환경 변수 딕셔너리
- 예: `User-Agent` , `IP` , `Referer` 등

```
user_agent = request.META.get('HTTP_USER_AGENT')
client_ip = request.META.get('REMOTE_ADDR')
```

9. `request.path`

- 요청된 URL 경로 (쿼리스트링 제외)

```
# URL이 /login/?next=/home/ 일 때
request.path # "/login/"
```

10. `request.build_absolute_uri()`

- 현재 URL의 절대 경로 반환 (도메인 포함)

```
request.build_absolute_uri()
# 예: "http://localhost:8000/home/"
```

✓ 실전 예시

```
def dashboard(request):
    if request.method == 'GET' and request.user.is_authenticated:
        username = request.user.username
        ip = request.META.get('REMOTE_ADDR')
        return render(request, 'dashboard.html', {'username': username, 'ip': ip})
    else:
        return redirect('login')
```

✓ 요약 표

속성	설명
<code>request.method</code>	요청 방식 (<code>GET</code> , <code>POST</code>)
<code>request.GET</code> , <code>request.POST</code>	폼, 쿼리 파라미터
<code>request.FILES</code>	업로드 파일

<code>request.user</code>	로그인 사용자 정보
<code>request.session</code>	세션 저장소
<code>request.COOKIEs</code>	클라이언트 쿠키 정보
<code>request.META</code>	HTTP 헤더 등 메타 정보
<code>request.path</code>	현재 URL 경로
<code>request.build_absolute_uri()</code>	절대 URL 생성

Tip: 디버깅할 때

```
import pprint
pprint.pprint(request.__dict__)
```

혹은

```
print(request.GET)
print(request.POST)
print(request.META)
```

Django에서 `request` 객체 실습을 위한 **프로젝트 전체 구조 + 상세 명령어 + 모든 파일 구성**을 처음부터 끝까지 단계별로 구현. 😊

목표

- `GET`, `POST`, `SESSION`, `USER`, `META` 등 `request` 객체의 핵심 요소 실습
- 실습 웹 페이지에서 request 정보 직접 확인

1단계: 가상환경 + 프로젝트 시작

```
# 가상환경 만들기 (선택)
python -m venv venv
source venv/bin/activate # Windows는 venv\Scripts\activate

# Django 설치
pip install django
```

```
# 프로젝트 생성
django-admin startproject myproject
cd myproject

# 앱 생성
python manage.py startapp request_test
```

2단계: settings.py 설정

`myproject/settings.py` 열고 아래 항목 확인/수정:

```
INSTALLED_APPS = [
    ...
    'request_test',
    'django.contrib.sessions',
    'django.contrib.messages',
]

# 템플릿 경로 자동 설정됨 (추가 설정 X)

# 로그인 시 @login_required 용도
LOGIN_URL = '/admin/login/'
```

3단계: 전체 디렉토리 구조

```
myproject/
├── manage.py
├── myproject/
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py    ← 루트 URL 연결
│   └── ...
├── request_test/
│   ├── __init__.py
│   └── views.py   ← request 실행 뷰
```

```
| |— urls.py    ← 앱 내부 URL 설정
| |— templates/
| |   |— request_test/
| |       |— request_info.html ← 실습 템플릿
```



4단계: 실습 뷰 작성

request_test/views.py

```
from django.shortcuts import render

def request_info_view(request):
    context = {
        'method': request.method,
        'get_data': request.GET,
        'post_data': request.POST,
        'user': request.user,
        'is_logged_in': request.user.is_authenticated,
        'session_value': request.session.get('demo', '없음'),
        'user_agent': request.META.get('HTTP_USER_AGENT', '알 수 없음'),
        'client_ip': request.META.get('REMOTE_ADDR', '알 수 없음'),
        'path': request.path,
        'full_url': request.build_absolute_uri()
    }

    # 세션에 값 저장
    request.session['demo'] = '세션에서 저장한 값입니다.'

    return render(request, 'request_test/request_info.html', context)
```



5단계: 앱 URL 연결

request_test/urls.py 새로 생성:

```
from django.urls import path
from .views import request_info_view
```

```
urlpatterns = [
    path('request-info/', request_info_view, name='request_info'),
]
```

루트 URL에서 앱 연결

myproject/urls.py

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

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



6단계: 템플릿 생성

폴더 생성:

```
mkdir -p request_test/templates/request_test
```

request_info.html

- 코드 완성 : 10점

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>Request 실습</title>
</head>
<body>
    <h2>🔍 request 객체 실습</h2>

    <p><strong>요청 방식:</strong> {{ _____ }}</p> # ← 여기에 동작 코드를
작성하세요 (1점)
    <p><strong>요청 경로:</strong> {{ _____ }}</p> # ← 여기에 동작 코드를
```

작성하세요 (1점)

<p>전체 URL: {{ _____ }}</p> # ← 여기에 동작 코드를 작성하세요 (1점)

<p>클라이언트 IP: {{ _____ }}</p> # ← 여기에 동작 코드를 작성하세요 (1점)

<p>User-Agent: {{ _____ }}</p> # ← 여기에 동작 코드를 작성하세요 (1점)

<hr>

<h3>📄 GET 요청 정보</h3>

<pre>{{ _____ }}</pre> # ← 여기에 동작 코드를 작성하세요 (1점)

<h3>📄 POST 요청 정보</h3>

<pre>{{ _____ }}</pre> # ← 여기에 동작 코드를 작성하세요 (1점)

<hr>

<h3>👤 사용자 정보</h3>

<p>로그인 여부: {{ _____ }}</p> # ← 여기에 동작 코드를 작성하세요 (1점)
{% if is_logged_in %}

<p>사용자: {{ _____ }}</p> # ← 여기에 동작 코드를 작성하세요 (1점)
{% else %}

<p>익명 사용자입니다.</p>
{% endif %}

<hr>

<h3>📦 세션 데이터</h3>

<p>session['demo']: {{ _____ }}</p> # ← 여기에 동작 코드를 작성하세요 (1점)

<hr>

<h3>🔥 요청 테스트</h3>

<form method="get">

<input type="text" name="search" placeholder="GET 요청 파라미터">

<button type="submit">GET 전송</button>


```

</form>

<form method="post">
    {% csrf_token %}
    <input type="text" name="message" placeholder="POST 요청 내용">
    <button type="submit">POST 전송</button>
</form>
</body>
</html>

```

7단계: 실행 및 테스트

```

python manage.py makemigrations
python manage.py migrate
python manage.py runserver

```

웹 브라우저에서:

- 접속: <http://localhost:8000/request-info/>
- GET/POST 폼 테스트
- 로그인 상태 확인하려면 `/admin/` 에서 사용자 생성 → 로그인 후 다시 `/request-info/`

실습 결과 확인 가능한 항목

항목	설명
<code>request.method</code>	GET/POST 요청 방식
<code>request.GET</code>	URL 쿼리 파라미터
<code>request.POST</code>	폼 전송 값
<code>request.user</code>	로그인 사용자 정보
<code>request.session</code>	서버 측 세션 저장 값
<code>request.META</code>	IP, 브라우저 정보 등 환경정보
<code>request.path</code>	현재 URL
<code>request.build_absolute_uri()</code>	전체 URL

제출 1.

실행화면 (GET) 캡처 제출 : 2점

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/request-info/". The page title is "request 객체 실습". The main content area displays the following information:

- 요청 방식:** GET
- 요청 경로:** /request-info/
- 전체 URL:** http://127.0.0.1:8000/request-info/
- 클라이언트 IP:** 127.0.0.1
- User-Agent:** Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36

GET 요청 정보

<QueryDict: {}>

POST 요청 정보

<QueryDict: {}>

사용자 정보

로그인 여부: True

사용자: admin

세션 데이터

session['demo']: 세션에서 저장한 값입니다.

요청 테스트

GET 요청 파라미터: GET 전송

POST 요청 내용: POST 전송

제출 2.

실행화면(POST) 캡처 제출 : 2점

←

→

↻

127.0.0.1:8000/request-info/


request 객체 실습


요청 방식: POST

요청 경로: /request-info/


전체 URL: http://127.0.0.1:8000/request-info/

클라이언트 IP: 127.0.0.1


User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36


GET 요청 정보

```
<QueryDict: {}>
```



POST 요청 정보

```
<QueryDict: {'csrfmiddlewaretoken': ['lh0lXrCPqWR84cw5SVaarfuxkFZJjW9EZGXcW0qryOMt9b0W7tJb7QZPG4FrQ004'], 'message': ['TEST']}>
```


사용자 정보

로그인 여부: True

사용자: admin


세션 데이터

session['demo']: 세션에서 저장한 값입니다.


요청 테스트

Django에서 `request.FILES` 를 사용한 파일 업로드 기능을 구현하는 예제

목표

- 사용자가 파일(PDF, 이미지 등)을 업로드
- 서버에서 `request.FILES` 로 파일 처리
- 업로드된 파일을 저장하고 경로를 출력
- 업로드된 파일을 웹에서 볼 수 있도록 설정

전체 흐름 요약

1. HTML 폼에서 `<input type="file">` 으로 업로드
2. `request.FILES` 로 서버에 전달
3. Django 모델 필드(`FileField` , `ImageField`)에 저장
4. 업로드된 파일 경로를 보여주기

1단계: settings.py 설정

`settings.py` 에서 아래 항목 추가:

```
import os

# MEDIA 관련 설정
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

2단계: 모델 만들기 (`models.py`)

`request_test/models.py`

```
from django.db import models

class UploadedFile(models.Model):
    title = models.CharField(max_length=100)
    file = models.FileField(upload_to='uploads/')
    uploaded_at = models.DateTimeField(auto_now_add=True)

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

3단계: 마이그레이션

```
python manage.py makemigrations
python manage.py migrate
```

4단계: 템플릿 (`upload_file.html`)

`templates/request_test/upload_file.html`

- 코드 완성 : 1점

```
<h2><img alt="upload icon" data-bbox="194 198 218 212"/> 파일 업로드</h2>
<form method="post" enctype="multipart/form-data">
    {% csrf_token %}
    <label>제목:</label><br>
    <input type="text" name="title"><br><br>

    <label>파일 선택:</label><br>
    <input type="file" name="file"><br><br>

    <button type="submit">업로드</button>
</form>

{% if _____ %} % # ← 여기에 동작 코드를 작성하세요 (1점)
    <h3>업로드 결과</h3>
    <p><strong>제목:</strong> {{ title }}</p>
    <p><a href="{{ uploaded_file_url }}">업로드된 파일 보기</a></p>
{% endif %}
```

5단계: 뷰 만들기 (`views.py`)

- 코드 완성 : 2점

```
from django.shortcuts import render
from .models import UploadedFile

def file_upload_view(request):
    uploaded_file_url = None
    title = None

    if request.method == 'POST' and request.FILES.get('file'):
        file = _____['file'] # ← 여기에 동작 코드를 작성하세요 (1점)
        title = _____('title', '') # ← 여기에 동작 코드를 작성하세요 (1점)
```

```

uploaded = UploadedFile.objects.create(title=title, file=file)
uploaded_file_url = uploaded.file.url

return render(request, 'request_test/upload_file.html', {
    'uploaded_file_url': uploaded_file_url,
    'title': title
})

```

6단계: URL 연결

`request_test/urls.py`

```

from django.urls import path
from .views import file_upload_view

urlpatterns = [
    path('upload/', file_upload_view, name='upload_file'),
]

```

루트 URL(`myproject/urls.py`)에 `media` 경로도 연결:

```

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

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

if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

7단계: 테스트

```
python manage.py runserver
```

접속: <http://localhost:8000/upload/>

- 파일 선택 후 업로드
- `media/uploads/` 폴더에 파일이 저장됨
- 업로드 후 웹에서 파일 링크 확인 가능

✓ 업로드 파일 경로 구조

업로드된 파일은 `media/uploads/` 디렉토리에 저장되고, URL로는 `/media/uploads/파일이름` 형태로 접근할 수 있어요.

✓ 정리

구성 요소	설명
<code>request.FILES</code>	업로드된 파일 객체 접근
<code>FileField</code> / <code>ImageField</code>	모델에서 파일 저장용 필드
<code>enctype="multipart/form-data"</code>	HTML 폼 필수 설정
<code>MEDIA_ROOT</code> , <code>MEDIA_URL</code>	파일 저장 위치 설정

제출 3.

파일 업로드 실행화면 캡처 : 2점



- 제출
 - 제출 화면 1,2,3 (각 2점, 6점) 캡처 후 노션이나 hwp, 워드 등을 사용해서 제출 바랍니다.
 - 깃허브 코드 및 사이트 (1점) 제출바랍니다.
 - 가상강의실 제출
- 일정
 - 과제 6 총점 : 총 20점
 - 제출 화면 1,2,3 (각 2점, 6점), 깃허브 코드 및 사이트 (1점) : 총 7점
 - 코드 완성 : 13점
 - request_info.html : 10점
 - 4단계: 템플릿 (upload_file.html) : 1점
 - 5단계: 뷰 만들기 (views.py) : 2점

- 5월 7일 (수) 13:30분까지
- 늦게 제출은 인정하지 않습니다.