Django 빠른 참조

모델, 뷰, 템플릿, ORM, 폼, 어드민, 인증

프로젝트 설정

프로젝트 & 앱 생성

```
pip install django
django-admin startproject mysite
cd mysite
python manage.py startapp blog
```

주요 명령

runserver	포트 8000에서 개발 서버 시작
makemigrations	모델 변경으로부터 마이그레이션 파일 생성
migrate	데이터베이스에 마이그레이션 적용
createsuperuser	어드민 슈퍼유저 생성
shell	Django가 로드된 대화형 Python 셸
test	테스트 스위트 실행

프로젝트 구조

manage.py	CLI 진입점
settings.py	프로젝트 설정
urls.py	루트 URL 설정
wsgi.py / asgi.py	서버 진입점
apps/models.py	데이터베이스 모델
apps/views.py	요청 핸들러

모델

모델 정의

```
from django.db import models

class Post(models.Model):
    title = models.CharField(max_length=200)
    body = models.TextField()
    created = models.DateTimeField(auto_now_add=True)
    published = models.BooleanField(default=False)
```

필드 타입

CharField(max_length=N)	짧은 텍스트 (max_length 필수)
TextField()	긴 텍스트 (제한 없음)
IntegerField()	정수
FloatField()	부동소수점
BooleanField()	True / False
DateTimeField()	날짜 및 시간
EmailField()	유효성 검사 포함 이메일
FileField(upload_to='')	파일 업로드

관계

```
author = models.ForeignKey(
    User, on_delete=models.CASCADE
)
tags = models.ManyToManyField(Tag, blank=True)
profile = models.OneToOneField(User, on_delete=models.CASCADE)
```

Meta & 메서드

```
class Meta:
    ordering = ['-created']
    verbose_name_plural = 'posts'

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

뷰

함수 기반 뷰

```
from django.shortcuts import render, get_object_or_404

def post_list(request):
    posts = Post.objects.filter(published=True)
    return render(request, 'blog/list.html', {'posts': posts})
```

클래스 기반 뷰

```
from django.views.generic import ListView, DetailView

class PostListView(ListView):
    model = Post
    template_name = 'blog/list.html'
    context_object_name = 'posts'
    paginate_by = 10
```

주요 CBV

ListView	객체 목록 표시
DetailView	단일 객체 표시
CreateView	객체 생성 폼
UpdateView	객체 편집 폼
DeleteView	확인 후 객체 삭제
TemplateView	템플릿 렌더링 (모델 없음)

JSON 응답

```
from django.http import JsonResponse

def api_posts(request):
    data = list(Post.objects.values('id', 'title'))
    return JsonResponse(data, safe=False)
```

템플릿

템플릿 문법

```
{{ variable }}
{{ post.title|truncatewords:30 }}
{% if user.is_authenticated %}
  <p>Welcome, {{ user.username }}!</p>
{% endif %}
```

반복 & 조건

```
{% for post in posts %}
  <h2>{{ post.title }}</h2>
  {% if forloop.last %}<hr>{% endif %}
{% empty %}
  <p>No posts yet.</p>
{% endfor %}
```

템플릿 상속

```
{# base.html #}
<html>
<body>{% block content %}{% endblock %}</body>
</html>

{# child.html #}
{% extends "base.html" %}
{% block content %}<h1>Hello</h1>{% endblock %}
```

자주 쓰는 필터

|date:"Y-m-d"	날짜 형식 지정
|default:"N/A"	빈 값의 대체값
|length	목록 항목 수
|truncatewords:N	N 단어로 제한
|safe	안전한 HTML로 표시 (이스케이프 없음)
|slugify	URL 안전 소문자 문자열

URL

URL 패턴

```
from django.urls import path, include

urlpatterns = [
    path('', views.index, name='index'),
    path('post/<int:pk>/', views.detail, name='detail'),
    path('blog/', include('blog.urls')),
]
```

경로 변환기

<int:pk>	정수 (예: 42)
<str:slug>	슬래시 없는 문자열
<slug:slug>	슬러그 (문자, 숫자, 하이픈)
<uuid:id>	UUID 형식
<path:rest>	슬래시 포함 전체 경로

URL 역방향 조회

```
from django.urls import reverse
url = reverse('detail', kwargs={'pk': 1})
# In templates: {% url 'detail' pk=post.pk %}
```

폼

모델 폼

```
from django import forms

class PostForm(forms.ModelForm):
    class Meta:
        model = Post
        fields = ['title', 'body', 'published']
```

뷰에서 폼 처리

```
def create_post(request):
    form = PostForm(request.POST or None)
    if form.is_valid():
        post = form.save(commit=False)
        post.author = request.user
        post.save()
        return redirect('detail', pk=post.pk)
    return render(request, 'blog/form.html', {'form': form})
```

템플릿에서 폼

```
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Save</button>
</form>
```

유효성 검사

```
def clean_title(self):
    title = self.cleaned_data['title']
    if len(title) < 5:
        raise forms.ValidationError("Title too short.")
    return title
```

Django 빠른 참조

어드민

모델 등록

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

@admin.register(Post)
class PostAdmin(admin.ModelAdmin):
    list_display = ['title', 'author', 'created', 'published']
    list_filter = ['published', 'created']
    search_fields = ['title', 'body']
```

어드민 옵션

list_display	목록 뷰 열
list_filter	사이드바 필터 옵션
search_fields	검색 가능한 필드
prepopulated_fields	자동 채우기 (예: 제목에서 슬러그)
readonly_fields	어드민에서 편집 불가
ordering	기본 정렬 순서

ORM 쿼리

기본 쿼리

```
Post.objects.all()               # all records
Post.objects.get(pk=1)           # single (raises if missing)
Post.objects.filter(published=True) # queryset
Post.objects.exclude(draft=True)  # exclude matches
Post.objects.count()             # total count
```

필드 조회

field__exact	정확한 일치 (기본값)
field__icontains	대소문자 무시 포함
field__gt / __lt	보다 큰 / 보다 작은
field__gte / __lte	이상 / 이하
field__in=[1,2,3]	목록 내 값
field__isnull=True	NULL 여부
field__startswith	문자열로 시작
field__range=(a,b)	a 이상 b 이하

체이닝 & 집계

```
from django.db.models import Q, Count, Avg

Post.objects.filter(
    Q(title__icontains='django') | Q(body__icontains='django')
).order_by('-created')[:10]

Post.objects.aggregate(avg_views=Avg('views'))
```

생성, 수정, 삭제

```
post = Post.objects.create(title='New', body='...')
post.title = 'Updated'
post.save()
Post.objects.filter(draft=True).update(published=False)
post.delete()
```

인증

로그인 / 로그아웃

```
from django.contrib.auth import authenticate, login, logout

user = authenticate(request, username='admin', password='pw')
if user is not None:
    login(request, user)
```

뷰 보호

```
from django.contrib.auth.decorators import login_required

@login_required
def dashboard(request):
    return render(request, 'dashboard.html')
```

인증 URL

```
# urls.py
path('accounts/', include('django.contrib.auth.urls'))
# Provides: login, logout, password_change, password_reset
```

템플릿 인증

```
{% if user.is_authenticated %}
  <p>Hi, {{ user.username }}</p>
  <a href="{% url 'logout' %}">Logout</a>
{% else %}
  <a href="{% url 'login' %}">Login</a>
{% endif %}
```

설정

주요 설정

DEBUG	개발 시 **True**, 프로덕션 시 **False**
ALLOWED_HOSTS	유효한 호스트명 목록
SECRET_KEY	암호화 서명 키 (비밀 유지)
DATABASES	DB 엔진, 이름, 호스트, 자격증명
INSTALLED_APPS	등록된 앱 목록
STATIC_URL	정적 파일 URL 접두사
MEDIA_URL / MEDIA_ROOT	사용자 업로드 파일 경로

데이터베이스 설정

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'mydb',
        'USER': 'dbuser',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

정적 파일

```
STATIC_URL = '/static/'
STATICFILES_DIRS = [BASE_DIR / 'static']
# In templates: {% load static %}
# <link href="{% static 'css/style.css' %}">
```