

Day3

📎 자료	Django
☰ 구분	Django

Django Model

- Model의 핵심개념과 ORM을 통한 데이터베이스를 조작을 이해한다. (내일(금)요일에 ORM 한다.)
- Django는 웹 애플리케이션의 데이터를 구조화하고 조작하기 위한 추상적인 계층(모델)을 제공한다.

Database

- 체계화된 데이터의 모임
- 검색 및 구조화 작업을 보다 쉽게 하기 위해 조직화된 데이터를 수집하는 저장 시스템
- Database의 기본구조 (스키마, 필드, 레코드, PK)

스키마(Schema)

- 뼈대(Structure)
- 데이터베이스에서 자료의 구조, 표현 방법, 관계 등을 정의한 구조

테이블(Table)

- 필드와 레코드를 사용해 조직된 데이터 요소들의 집합
- 관계(Relation)이라고도 부른다.
- 테이블을 만들기 위해서는 어떤 데이터, 타입을 사용할지 설계가 필요하다.(=스키마)

필드

	A	B	C	D
1	id	name	age	email
2	PK 1	hong	42	hong@gmail.com
3	2	kim	16	kim@naver.com
4	3	kang	29	kang@hotmail.com
5	4	chol	8	choi@hanmail.com

테이블

레코드

- **필드(field)**

- 속성, 컬럼(Column)
- 각 필드에는 고유한 데이터 형식이 지정된다. (int, text 등)

- **레코드(record)**

- 튜플, 행(Row)
- 테이블의 데이터는 레코드에 저장된다.
- 위에서는 4명의 고객정보가 저장되어 있으므로 레코드는 4개이다.

- **PK(Primary Key)**

- 기본 키
- 각 레코드의 고유한 값(식별자로 사용)
- 기술적으로 다른 항목과 절대로 중복될 수 없는 단일 값(unique)
- 데이터베이스 관리 및 테이블 가 관계 설정시 주요하게 활용된다.
- 예시: 주민등록번호 (각 사람들이 가진 주민번호는 모두 다르다.)

- **쿼리(Query)**

- 데이터를 조회하기 위한 명령어
- 조건에 맞는 데이터를 추출하거나 조작하는 명령어(주로 테이블형 자료구조에서)
- “Query를 날린다.” → “데이터베이스를 조작한다.”

Model

웹 애플리케이션의 데이터를 구조화하고 조작하기 위한 도구

- Django는 Model을 통해 데이터에 접근하고 조작한다.
- 사용하는 데이터들의 필수적인 필드들과 동작들을 포함한다.
- 저장된 데이터베이스의 구조 (layout)
- 일반적으로 각각의 모델은 하나의 데이터베이스 테이블에 매핑한다.
 - 모델 클래스 1개 == 데이터베이스 테이블 1개

Model 작성하기

1. 새 프로젝트(crud), 새 앱(articles) 작성 및 등록
2. models.py 작성하기
 - 모델클래스를 작성하는 것은 데이터 베이스 테이블의 스키마를 정의하는 것이다.
 - “모델 클래스” == “테이블 스키마”

```
# articles/models.py

from django.db import models


class Article(models.Model):
    title = models.CharField(max_length=30)
    content = models.TextField()
```


- 각 모델은 django.models.Model의 서브클래스로 django.db.models 모듈의 Model클래스를 상속받아 구성된다. → 클래스 상속 기반 형태의 Django 프레임워크 개발
- models 모듈을 통해 DB타입을 정의
- Article에 필요한 데이터 구조 정의
- title & content : 클래스 변수(속성)명으로 DB필드의 이름이다.
- models.CharField(max_length=30) : 클래스 변수 값(models모듈의 Field클래스), DB 필드의 데이터 타입을 지정

Django Model Field

Django Tutorial Part 3: Using models - Web 개발 학습하기 | MDN

803

 <https://developer.mozilla.org/ko/docs/Learn/Server-side/Django/Models>

 mdn web docs

1. CharField(max_length=None)

- 길이의 제한이 있는 문자열을 넣을 때 사용
- 데이터베이스와 Django의 유효성 검사시 사용

2. TextField()

- 글자 수가 많을 때 사용
- max_length 입력시 입력단계에는 반영되나 모델&데이터베이스 단계에서 미반영
- 유효성 검증 X

```
# articles/models.py

from django.db import models

class Article(models.Model):
    title = models.CharField(max_length=30)
    content = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
```

1. DatetimeField(auto_now_add)

- 최초 생성일자 자동 생성(게시판 글 작성 날짜)

2. DatetimeField(auto_now)

- 최초 수정일자 자동 생성 (게시판 글 작성 날짜)

3. EmailField(auto_now)

- 이메일 형식에 맞게 작성 하는 필드

Migrations

- Django가 모델에 생긴 변화를 실제 DB에 반영하는 방법
- 이전까지 단계는 스키마 작업으로 이를 반영하는 작업이 migration

- Migrations 주요 명령어

1. makemigrations

- `$ python manage.py makemigrations`
- 현재 내 모델 상태를 데이터베이스에 반영할 수 있는 하나의 migration 상태(설계도)로 만들 때 사용한다.
- 테이블 생성 및 수정시 반드시 migration할 것
- 명령어 실행 후 initial.py 파일 생성 확인
- initial파일에는 기본으로 설정되는 id(PK값)과 model에 입력한 정보가 들어있다.

2. migrate

- `$ python manage.py migrate`
- makemigrations로 만든 설계도(migration)를 실제 데이터베이스(db.sqlite3)에 반영하는 과정
- 모델의 변경사항과 데이터 베이스를 동기화
- 설치시 데이터베이스에 필요한 기본 환경을 셋팅해 준다.

[참고] → 아래것들은 몰라도 문제될것 없음... 말 그대로 참고임

3. showmigrations

- `$ python manage.py showmigrations`
- migrations 파일들이 migrate 됐는지 여부 확인하는 용도
- [X]표시가 있으면 migrate완료 되었다는 의미

4. sqlmigrate

- `$ python manage.py sqlmigrate articles 0001`
- 해당 migrations 파일이 SQL문으로 어떻게 해석될 지 미리 확인 가능

migration 3단계

1. models.py에 변경사항 발생!
2. migration 생성 (**makemigrations**) ! 그리고 난 후에

3. DB 반영 (모델과 DB 동기화) (migrate)

Admin site

Django의 가장 강력한 기능 중 하나인 automatic admin interface

- 관리자 페이지
 - 사용자가 아닌 서버의 관리자가 활용하기 위한 페이지
 - 모델 class를 admin.py에 등록하고 관리
 - 레코드 생성 여부 확인에 매우 유용하며 직접 레코드를 삽입할 수 도 있다.

admin 계정 생성

- username과 password를 입력해 관리자 계정 생성
 - `python manage.py createsuperuser`
 - email은 선택사항, password는 입력이 보이지 않으나 입력되고 있음

```
SSAFY@DESKTOP-DOGVPU MINGW64 ~/Desktop/crud
$ python manage.py createsuperuser
Username (leave blank to use 'ssafy'): admin
Email address: admin@admin.kr
Password:
Password (again):
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

admin site 로그인

- `http://127.0.0.1:8000/admin/` 으로 접속 후 로그인
- 현재는 계정만 만든 상태로 관리자 화면에 모델 클래스가 보이지 않는다.

admin에 모델 클래스 등록

- 모델의 record를 보기 위해서 admin.py에 등록이 필요하다.

```
# aricles/admin.py

from django.contrib import admin
from .models import Article

admin.site.register(Article)
```

- 등록 후에는 Add를 통해 입력이 가능하다.

서버 켜서 admin 들어가서 이것저것 눌러보기 !!! <끝>