Python & Django

12월 10일 수업내용 정리

고은경

Python: 클래스

- **클래스란?**
  - 객체 지향의 가장 기본적인 개념
  - 관련 **속성(변수)**과 **동작(함수)**을 하나의 범주로 묶어 실세계의 사물 표현
  - 클래스로부터 객체를 얼마든지 얻을 수 있음

- **왜 사용할까?**
  - 중복되는 코드를 줄이기 위해

1, 2번 중 하나는 반드시 써줘야 함!!

- **클래스 정의 문법** ➔

```
class 이름:
    def __init__(self):
        멤버 초기화
    메서드 정의
```

1 (def __init__(self): ~ 멤버 초기화)

2 (메서드 정의)

- **클래스 엑세서**
  - 클래스의 모든 멤버는 공개, 누구나 사용 가능 ➔ 아무나 쉽게 변경 가능한 문제

Python: 클래스

클래스 엑세서 {

```
class Coffee:
    def __init__(self, name, price):
        self.name = name
        self.price = price

    def price_print(self):
        print(self.name+"는 가격이", self.price,"원 입니다")

am = Coffee("아메리카노", 4100)
am.price = 10000
am.price_print()

es = Coffee("에스프레소", 3900)
es.price_print()

lat = Coffee("카페라떼", 4500)
lat.price_print()
```

Python: 클래스

- **상속**
  - 기존 클래스를 확장하여 멤버를 **추가**하거나 동작을 **변경**하는 방법
  - 상속받는 클래스는 기존 클래스의 **모든 멤버와 메서드**를 물려 받음

- **클래스 상속 정의 문법** ➔

```
class 클래스이름 (상속받을클래스):
    def __init__(self):          } 1
        멤버 초기화
    메서드 정의                  } 2
```

1, 2번 중 하나는 반드시 써줘야 함!!

```
class Am(Ep):
    def __init__(self):
        super().__init__()
        self.price += 200

    def make(self):
        super().make()
        print("물을 추가합니다")
am = Am()
am.get_price()
```

Python: 클래스

- **클래스 상속의 장점**
  - 중복 코드 적어 유지, 보수가 쉬움
  - 코드의 재사용 가능성이 높아짐

- **정적 메서드**
  - 클래스에 포함되는 단순한 유틸리티 메서드
  - 특정 객체 소속 x, 클래스와 관련 동작 x
  - Ex) 에스프레소가 '몇 잔 ' 팔렸는지?

➔

```
class 이름:
    @staticmethod
    def 메서드_이름():
        메서드 내용
```

- **특수 메서드**
  - 미리 약속된 작업을 수행하는 메서드
  - __str__: 객체 문자열화
  - __repr__: 객체의 표현식 만들기
  - __len__: 객체 길이 조사

Django

- **장고란 무엇인가?**
  - 파이썬으로 만들어진 웹 애플리케이션 프레임워크
  - 풀스택 프레임워크

- **장고를 사용하는 이유?**
  - 웹사이트 신속 개발 가능
  - 보안, 유지보수 편리
  - 많은 참고자료
  - 활발한 커뮤니티

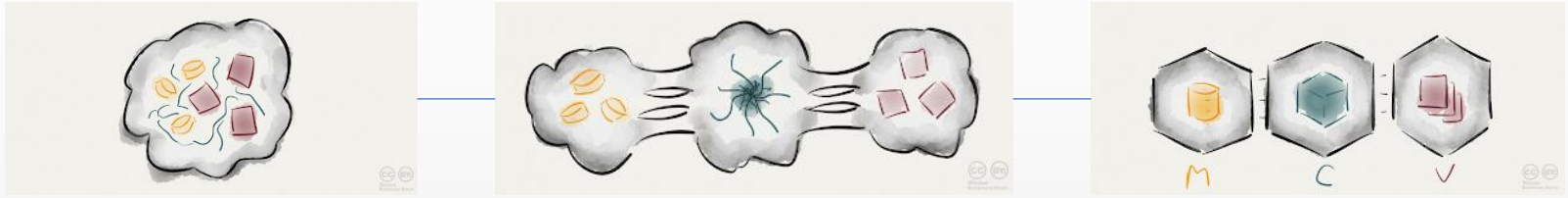
Django

- **장고의 특징**
  - **풀스택**: 데이터베이스 연동, MTV패턴, 템플릿 엔진, 세션 관리
  - **보안**: 개발자 실수 고려 ➔ 패스워드 단방향 저장, SQL 인젝션, CSRF 공격
  - **유지보수**: 유지보수 쉬운 디자인 패턴(MTV), 재사용 쉽도록 application 그룹화
  - **확장성:** 다른 컴포넌트 사용 가능(ex. sqlite3, mysql, vue.js...)
  - **다양한 용도:** 다양한 종류의 웹사이트 개발

MVC vs MTV ??

장고 동작 알아보기

- **MVC(Model View Controller) 패턴**
  - Model: 데이터 구조
  - View: 사용자 화면
  - Controller: Biz 로직

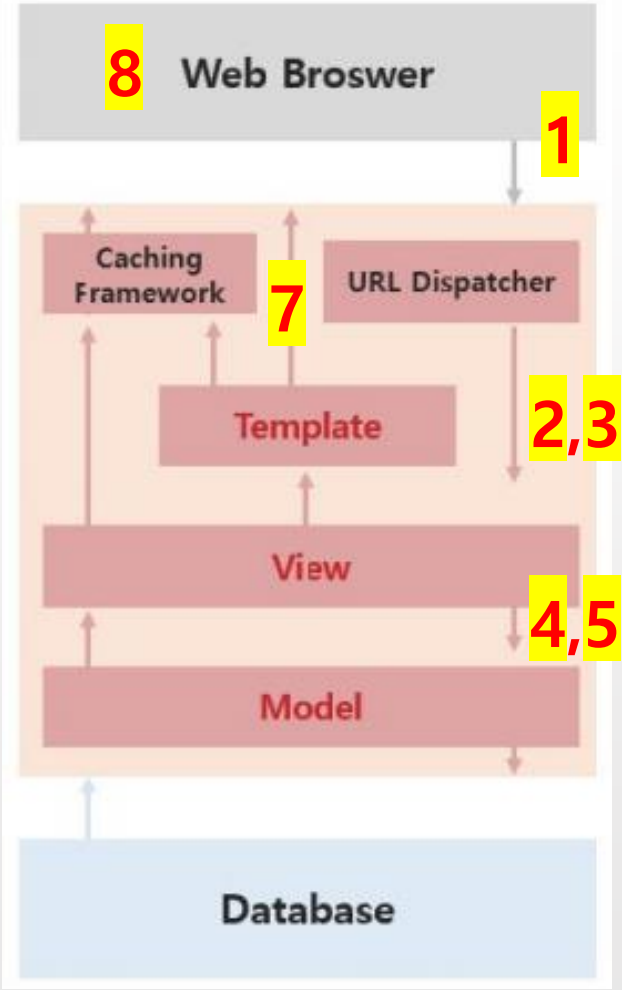


➔ 즉, 비즈니스 로직과 사용자 인터페이스 분리

- **MTV(Model Template View) 패턴**
  - MVC의 View → Template
  - MVC의 Controller → View

장고 동작 알아보기

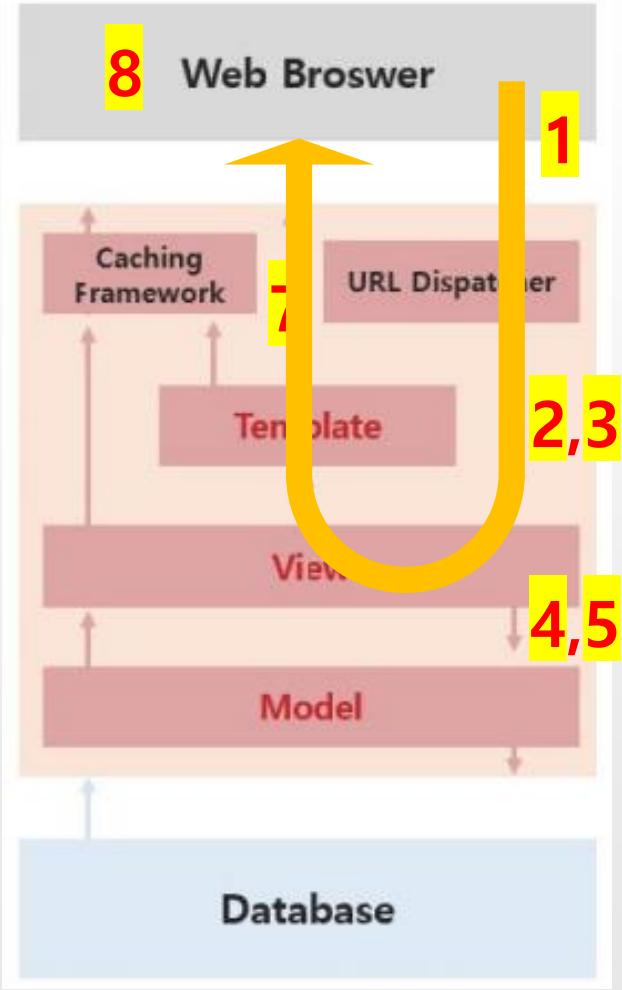
1. 웹 브라우저에서 이벤트가 발생
2. URL Dispatcher가 URL 분석
3. URL에 맞는 View를 호출
4. View에서 필요한 Model 요청
5. View에서 Model 처리
6. View에서 사용자에게 보여질 Template 호출
7. Template에서 웹 브라우저로 HTML 전송
8. 웹 브라우저에서 HTML 시각화



장고 동작 알아보기

1. 웹 브라우저에서 이벤트가 발생
2. URL Dispatcher가 URL 분석
3. URL에 맞는 View를 호출
4. View에서 필요한 Model 요청
5. View에서 Model 처리
6. View에서 사용자에게 보여질 Template 호출
7. Template에서 웹 브라우저로 HTML 전송
8. 웹 브라우저에서 HTML 시각화

Hello World 실습



Hello World 실습

1. 프로젝트 생성
2. 경로 설정
3. Terminal 입력
   - pip install Django
   - django-admin startproject config . (스페이스+.)
   - python manage.py migrate (manage.py는 m정도까지 치고 'TAB'키 눌러도 나타남)
   - python manage.py runserver (서버 실행하면서 많이 사용)
   - 나오는 링크 클릭하면 서버 실행
   - 실행된 서버 주소창에 **/admin** 추가하면 관리자계정 입력창 (이 상태에선 계정 없음)
   - python manage.py createsuperuser (관리자 계정 생성)
   - python manage.py runserver (서버 재실행)
   - **/admin** 해서 관리자 이름과 패스워드 입력
   - python manage.py startapp hello (Hello World App 생성)

Hello World 실습

명령어	설명
startapp	프로젝트에 앱을 새로 만듬
makemigrations	앱의 모델 변경사항을 정리
migrate	디비에 모델의 변경사항을 적용
runserver	테스트 서버를 실행
createsuperuser	관리자 계정을 생성

Hello World 실습

4. 스크립트 입력

- hello/views.py

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
def hello_world(request):
  return HttpResponse('Hello World!')
```

- hello 폴더에 urls.py 파일 생성
  - [New -> Python File]

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

urlpatterns = [
   path('', views.hello_world, name='hello_world')
]
```

URL Dispatcher

- config/urls.py

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

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

URL Dispatcher

5. Terminal 입력

- python manage.py runserver
- 실행된 서버 주소창에 **/hello_world** 추가
- 제대로 뜨는지 확인!! 완료!!

cf) 서버 종료 명령은 ctrl + c

Thank you