실전! FastAPI 입문

신동현

섹션 0. Orientation

강좌 소개

강의 목적

실무에 바로 적용 가능한 FastAPI 사용법을 익힌다
(프로젝트 구성, 테스트 코드, 리팩터링, 디버깅, 버전관리 등)

수강 대상

FastAPI가 처음이신 분
웹 개발이 처음이신 분

심화 강의

Python Domain-Driven Design(계획 중)

섹션 0. Orientation

강좌 소개

다루지 않는 내용

기초 Python 문법

UI

템플릿 엔진 활용한 프론트엔드(e.g. Jinja)

비동기 프로그래밍(AsyncIO)

섹션 0. Orientation

강사 소개

신동현

- 백엔드 개발자
  - 에이슬립 슬립루틴 서비스 개발
  - (전) 에이블리 광고 사업팀

- PyCon Korea 21/22/23 Speaker
- AWS StartUp Security GameDay 1등

- 기술 블로그: qu3vipon.com
- 깃허브: github.com/qu3vipon

섹션 0. Orientation

요구사항

Python 3.10+

Docker

IDE: PyCharm

섹션 0. Orientation

Python 가상 환경 구축

Mac

python3.10 -m venv example	가상 환경 생성
cd example	example directory 이동
source bin/activate	가상 환경 활성화

Windows

python3.10 -m venv example	가상 환경 생성
cd example	example directory 이동
Scripts\activate.bat	가상 환경 활성화

섹션 0. Orientation

Docker 설치

Docker Desktop 설치

Mac https://docs.docker.com/desktop/install/mac-install/

Windows https://docs.docker.com/desktop/install/windows-install/

섹션 0. Orientation

PyCharm 설치

PyCharm Community Edition

Mac https://www.jetbrains.com/pycharm/download/#section=mac

Windows https://www.jetbrains.com/pycharm/download/#section=windows

섹션 1. FastAPI 알아보기

FastAPI란?

GitHub Star History

최근(2018년 말)에 만들어진 파이썬 웹 프레임워크로 단기간 내에 많은 인기
https://star-history.com/#tiangolo/fastapi&Date

사용하는 기업

MicroSoft, Uber, Netflix와 같은 글로벌 테크 기업에서 FastAPI 사용
https://fastapi.tiangolo.com/#opinions

비교

	<Django+DRF>	<FastAPI>
기능	Battery Included	경량 Framework
확장성	제한적	자유로움
성능		Django 보다 우수

섹션 1. FastAPI 알아보기

FastAPI 장점

성능

직관적인 디자인
- path, query, body, response, dependency

Type Hints
- Pydantic
- Data validation
- IDE Support

비동기처리
- AsyncIO, Background Task

자동 API 문서 생성
- OpenAPI, SwaggerUI

섹션 1. FastAPI 알아보기

클라이언트-서버 모델

클라이언트
서비스 요청자

서버
서비스 제공자

장점
확장 가능성 하나의 서버에서 다수의 클라이언트 대응 가능
관심사의 분리 역할을 분리해서 성능, 보안에 대한 최적화

웹 서비스

HTTP 요청 클라이언트 -> (네트워크) -> 서버
<API>
HTTP 응답 클라이언트 <- (네트워크) <- 서버

섹션 1. FastAPI 알아보기

API

Application Programming Interface

Interface 예시

리모컨

기능을 외부에서 사용할 수 있게 제공

사용자 -> 버튼(물리 신호) -> 전기 신호 -> 장비

에어컨

<내부>	<외부>
_실내온도 감지	전원 On/Off
_실외기 조작	온도 조절
	바람 세기 조절

섹션 1. FastAPI 알아보기

RESTful API

REST Representational State Transfer

Resource

URL를 통해 고유한 리소스 표현

예시 /api/v1/posts
/api/v1/posts/123/comments

Method

HTTP Method 통해 API의 동작 표현

예시 GET /api/v1/posts -> posts 조회
POST /api/v1/posts -> post 생성
PATCH /api/v1/posts/123 -> post 수정
DELETE /api/v1/posts/123 -> post 삭제

섹션 1. FastAPI 알아보기

ToDo 서비스 만들기

GET

전체 ToDo 조회: /api/v1/todos
단일 ToDo 조회: /api/v1/todos/<id>

POST

ToDo 생성: /api/v1/todos

PATCH

ToDo 수정: /api/v1/todos/<id>

DELETE

ToDo 삭제: /api/v1/todos/<id>

섹션 1. FastAPI 알아보기

Status Code

2xx

200 OK	요청 성공, 범용적, GET/POST/PUT/PATCH
201 Created	요청 성공, 새로운 자원 생성, POST
204 No Content	요청 성공, 응답할 자원 없음, DELETE

4xx

400 Bad Request	요청 실패, 요청이 잘못된 경우(query param, body)
401 Unauthorized	인증 실패
403 Forbidden	권한 문제 또는 잘못된 method
404 Not Found	자원이 없는 경우 또는 잘못된 endpoint

5xx

500 Internal Server Error	범용적인 서버 에러
502 Bad Gateway	Reverse Proxy에서 서버의 응답을 처리할 수 없는 경우
503 Service Unavailable	서버가 요청을 처리할 수 없는 경우(e.g. 일시적 부하, 서버 다운)

섹션 2. 데이터베이스

데이터베이스란?

데이터베이스

대량의 데이터를 영구적으로 저장/관리하기 위한 시스템

분류

- 관계형 데이터베이스(Relational database, RDB)
  관계형 모델에 기반해서 데이터를 테이블, 행, 열 구조로 관리(=Schemaful)
  Oracle, MySQL, PostgreSQL, Sqlite, …

- 비관계형 데이터베이스(NoSQL)

Key-value	Redis, etcd
Document	MongoDB
Wide-column	Cassandra, ScyllaDB
Timeseries	Apache Druid, InfluxDB
Graph	Neo4j

섹션 2. 데이터베이스

sqlalchemy 소개

sqlalchemy

관계형 데이터베이스를 사용하기 위한 High-level 인터페이스를 제공하는 Python 라이브러리

ORM, Query, Transaction, Connection Pooling

ORM(Object-Relational Mapping)

관계형 데이터베이스를 객체 지향 프로그래밍(OOP)에 대응하여 사용하는 프로그래밍 기술

하나의 테이블 = 하나의 클래스
하나의 행(레코드) = 하나의 객체

<데이터베이스>
id | username
1 | qu3vipon

<Python>
user = User(id=1, username='qu3vipon')

섹션 3. 테스트 코드

섹션 소개

섹션 1	REST API 실습	=> Swagger UI
섹션 2	데이터베이스 & ORM 적용	=> Swagger UI
섹션 3	테스트 코드 적용	

섹션 3. 테스트 코드

테스트 코드란?

테스트 코드

시스템의 품질과 신뢰성을 검증하기 위한 코드

장점

‘코드 변경 -> 기능 점검’ 과정 자동화 => 반복적인 과정을 줄여주어 생산성 향상

개발자가 시스템에 대한 안정성 확신 => 유연한 코드 변경 및 리팩터링

심화

Unit Test, Integration Test, Regression Test, TDD, BDD 등

다양한 테스트 종류와 방법론 존재

섹션 3. 테스트 코드

PyTest

PyTest

테스트 코드 작성을 위한 Python 라이브러리

vs. Unittest

간결한 문법

assert문

함수 단위 테스트 지원

fixture 지원

테스트 데이터 관리

섹션 4. 리팩터링

Dependency Injection

의존성 주입

다양한 클래스나 함수 간의 강한 결합을 줄이기 위해 사용되는 기술

의존성을 사용하는 컴포넌트가 직접 정하는 것이 아니라 외부에서 전달(주입)

Injector라고 불리는 별도의 모듈에서 관리

용어

Coupling(결합) 강하게 결합된 컴포넌트들은 수정과 변경이 어려움

Dependency(의존성) 한 컴포넌트가 올바르게 동작하기 위해 다른 요소에 의존하는 것

섹션 4. 리팩터링

Dependency Injection

Before DI

```
class Body:
    def __init__(self, color):
        self.color = color

class Car:
    def __init__(self):
        self.body = Body(color='red')

    def drive(self):
        pass

red_car  = Car()
```

섹션 4. 리팩터링

Dependency Injection

After DI

```
class Body:
    def __init__(self, color):
        self.color = color

class Car:
    def __init__(self, body):
        self.body = body

    def drive(self):
        pass

red_car = Car(body=Body(color='red'))
blue_car = Car(body=Body(color='blue'))
```

섹션 4. 리팩터링

Repository Pattern

Repository Pattern

데이터를 다루는 부분을 추상화하는 기술로 비지니스 로직과 데이터 관리의 강한 결합 없애준다.

데이터를 불러오고 저장하는 것과 같은 구체적인 구현은 감춘다.

섹션 5. 기능 고도화

회원가입 - JOIN

JOIN

공통되는 컬럼을 이용하여 두 개의 테이블을 연결해서 조회하는 기능

User

id	username
1	qu3vipon

ToDo

id	user_id	contents
1	1	FastAPI Section 1
2	1	FastAPI Section 2

```
> SELECT u.username, t.contents FROM user u JOIN todo t ON u.id = t.user_id;
```

username	contents
qu3vipon	FastAPI Section 1
qu3vipon	FastAPI Section 2

섹션 5. 기능 고도화

회원가입 - JOIN

실습

```
user_id = Column(Integer, ForeignKey("user.id"))

class User(Base):
    __tablename__ = "user"

    id = Column(Integer, primary_key=True, index=True)
    username = Column(String(256), nullable=False)
    password = Column(String(256), nullable=False)

> from sqlalchemy.schema import CreateTable
> print(CreateTable(ToDo.__table__).compile(engine))
```

섹션 5. 기능 고도화

회원가입 - JOIN

Lazy Loading

지연 로딩 연관된 객체의 데이터가 실제로 필요한 시점에 조회하는 방법

장점 첫 조회 속도가 더 빠름
단점 N+1 문제 발생

N+1 Problem

```
for todo in todos:
    print(todo.user.username)
```

Eager Loading

즉시 로딩 데이터를 조회할 때 처음부터 연관된 객체를 같이 읽어오는 방법

장점 데이터를 더 효율적으로 가져올 수 있음(N+1 방지)
단점 꼭 필요하지 않은 데이터까지 JOIN 해서 읽어올 수 있음

섹션 5. 기능 고도화

로그인 / 인증 - JWT

JWT

JSON Web Token / 사용자 인증에 사용되는 JSON 포맷의 웹 토큰

장점

- 토큰에 유저 정보를 담아 별도의 데이터 조회 없이도 인증 처리가 가능하다
- 토큰 변조를 검증할 수 있기 때문에 내장된 데이터를 신뢰할 수 있다
- 토큰을 만료 시킬 수 있다

인증 절차

클라이언트		서버
id & password	➡️	id & password 검증 / JWT 생성
JWT 저장	⬅️	JWT 반환(access_token)
API 요청(헤더: JWT)	➡️	JWT 검증

섹션 5. 기능 고도화

OTP - 캐싱

캐싱

데이터를 임시 저장소에 저장해서 사용하는 기술

특징

- 데이터 조회 속도가 빠름
- 데이터 영속성을 보장할 수 없음

활용

- 임시 데이터, 반복적으로 사용하는 데이터, 빠르게 조회되어야 데이터에 주로 활용

OTP

One-Time Password / 일회용 비밀번호

강의를 마치며

끝까지 강의를 수강해주셔서 감사합니다!