레벨 1 스프링 부트 3으로 개발 입문하기

00장 개발 환경 구축하기

01장 자바 개발자가 알아두면 좋은 지식

서버와 클라이언트

데이터베이스

아이피와 포트

라이브러리와 프레임워크

백엔드 개발자의 업무

- 서버 측 애플리케이션을 개발하는 일
- 과제 할당 → 과제 분석 → 개발 → 테스트(리뷰) → QA 및 버그 수정 → 배포 → 유지보수

백엔드 프로그래밍 언어

자바 애너테이션

02장 스프링 부트 3 시작하기

스프링과 스프링 부트

스프링 콘셉트 공부하기

- 제어의 역전
    - 객체의 생성과 관리를 개발자가 하는 것이 아니라 프레임워크가 대신하는 것
- 의존성 주입
    - 외부에서 객체를 주입받아 사용하는 것
- 빈과 스프링 컨테이너

- 관점 지향 프로그래밍
  - 프로그래밍을 할 때 핵심 관점과 부가 관점을 나누어서 개발하는
- 이식 가능한 서비스 추상화
  - 어느 기술을 사용하던 일관된 방식으로 처리하도록 하는 것

스프링 부트 3 둘러보기

- 스프링 부트 스타터 살펴보기
- 자동 구성
- 스프링 부트 3와 자바 버전
  - 텍스트 블록: """
  - formatted() 메서드: %d 같은 식으로 값 파싱
  - 레코드: 데이터 전달을 목적으로 하는 객체를 더 빠르고 간편하게 만들기 위한 기
  - 패턴 매칭: 타입 확인을 위해 사용하던 instanceof 키워드를 조금 더 쉽게 사용할 수 있게 해 줌
  - 자료형에 맞는 case 처리

스프링 부트 3 코드 이해하기

- `@SpringBootApplication` 이해하기
  - `@SpringBootConfiguration` : 스프링 부트 관련 설정을 나타내는 애너테이션
  - `@ComponentScan` : 사용자가 등록한 빈을 읽고 등록하는 애너테이션
  - `@EnableAutoConfiguration` : 스프링 부트에서 자동 구성을 활성화하는 애너테이션
- `@Component` 애너테이션이 있는 클래스는 빈으로 등록되며, `@Controller` , `@RestController` , `@Configuration` , `@Repository` , `@Service` 모두 `@Component` 애너테이션을 가지고 있음

03장 스프링 부트 3 구조 이해하기

- 프레젠테이션 계층(컨트롤러)
- 비즈니스 계층(서비스)
- 퍼시스턴스 계층(리포지터리)
- 데이터베이스

04장 스프링 부트 3와 테스트

- 테스트 코드
  - given-when-then 패턴
    - given: 테스트 실행을 준비하는 단계
    - when: 테스트를 진행하는 단계
    - then: 테스트 결과를 검증하는 단계
- JUnit
  - 자바 언어를 위한 단위 테스트 프레임워크
- AssertJ
  - JUnit과 함께 사용해 검증문의 가독성을 확 높여주는 라이브러리
- 제대로 테스트 코드 작성
  - `@SpringBootTest` : 메인 클래스와 포함되어 있는 빈을 찾은 다음 테스트용 애플리케이션 컨텍스트라는 것을 만듭니다
  - `@AutoConfigureMockMvc` : MockMvc를 생성하고 자동으로 구성하는 애너테이션
  - MockMvc: 애플리케이션을 서버에 배포하지 않고도 테스트용 MVC 환경을 만들어 요청 및 전송, 응답 기능을 제공하는 유틸리티 클래스, 컨트롤러를 테스트할 때 사용되는 클래스
  - perform(): 요청을 전송하는 역할을 하는 메서드
  - accept(): 요청을 보낼 때 무슨 타입으로 응답을 받을지 결정하는 메서드
  - andExpect(): 응답을 검증
  - jsonPath("$[0].${필드명}"): JSON 응답값의 값을 가져오는 역할을 하는 메서

```
// MockMvcRequestBuilders 안 될 때
import static org.springframework.test.web.servlet.request.Mo
import static org.springframework.test.web.servlet.result.Moc
import static org.springframework.test.web.servlet.result.Moc
import static org.springframework.test.web.servlet.result.Moc
```

05장 데이터베이스 조작이 편해지는 ORM

- 데이터베이스

- ORM
  - 자바의 객체와 데이터베이스를 연결하는 프로그래밍 기법입니다
- JPA와 하이버네이트
  - JPA: 자바에서 관계형 데이터베이스를 사용하는 방식을 정의한 인터페이스
  - 하이버네이트: JPA 인터페이스를 구현한 구현체이자 자바용 ORM 프레임워크
  - 엔티티: 데이터베이스의 테이블과 매핑되는 객체
  - 엔티티 매니저: 엔티티를 관리해 데이터베이스와 애플리케이션 사이에서 객체를 생성, 수정, 삭제하는 등의 역할
  - 영속성 컨텍스트: JPA의 중요한 특징 중 하나, 엔티티를 관리하는 가상의 공간
    - 1차 캐시: 내부에 가지고 있음, 키는 엔티티의 기본키
    - 쓰기 지연
    - 변경 감지
    - 지연 로딩
  - 엔티티 상태: 분리 / 관리/ 비영속/ 삭제
- 스프링 데이터와 스프링 데이터 JPA
  - 스프링 데이터: 비즈니스 로직에 더 집중할 수 있게 데이터베이스 사용 기능을 클래스 레벨에서 추상화
  - 스프링 데이터 JPA: 스프링 데이터의 공통적인 기능에서 JPA의 유용한 기술이 추가된 기능
  - 조회 메서드
    - 전체 조회: findAll() 메서드 사용
    - 아이디로 조회: findById() 메서드 사용
    - 특정 칼럼으로 조회: 쿼리 메서드 명명 규칙에 맞게 정의 후 사용
  - 추가, 삭제 메서드
    - 레코드 추가: save()
    - 한꺼번에 여러 레코드 추가: saveAll()
    - 아이디로 레코드 삭제: deleteById()
    - 모든 레코드 삭제: deleteAll()

- 수정 메서드
  - 조회 후 트랜잭션 범위 내에서 필드값 변경