CS study 13주차

Spring Bean의 생명주기 😎

스프링 빈(Spring Bean)이란?

Spring IoC 컨테이너가 관리하는 자바 객체

스프링 IoC Container(=Container)란?

객체를 생성하고 관리하고 책임지고 의존성을 관리해주는 컨테이너

스프링 컨테이너에 Bean 등록

1. @Component 어노테이션 (컴포넌트 스캔)
2. @Configuration/@Bean, 자바 코드로 직접 스프링 빈 등록

@Component

가장 쉬운 방법

어노테이션을 사용하려면 컴포넌트 스캔이 필수

클래스 선언부 위에 @Component 어노테이션 사용

@Controller, @Service, @Repository···

```
import ...

@Repository
public interface UserRepository extends JpaRepository<
    User findByUserId(String userId);
```

자바 코드로 직접 스프링 빈 등록

자바 설정 클래스를 만들어 사용 해야함

설정 클래스 만들고 @Configuration 어노테이션 추가

그리고 특정 타입을 리턴하는 메서드를 만들고 @Bean 어노테이션 추가

```java
@Configuration
public class SpringConfig {

    @Bean
    public MemberService memberService() { return new MemberService(memberRepository()); }

    @Bean
    public MemberRepository memberRepository() { return new MemoryMemberRepository(); }
}
```

등록된 스프링 빈 사용

@Autowired 어노테이션을 통해 의존성을 주입받아 사용함

1. 생성자 주입
2. 필드 주입
3. 수정자 주입

필드 주입 방법

```java
@Slf4j
@RestController
@RequestMapping("/qna")
@Api("Qna Controller")
public class QnaController {
    @Autowired
    QnaService service;

    @GetMapping
    @ApiOperation(value="전체 qna",notes="전체 qna를 불러옵니다")
    public ResponseEntity<List<QnaDto>> getAllQna() throws Exception{
        List<QnaDto> list=service.getAllQna();
        ResponseEntity<List<QnaDto>> res=new ResponseEntity(list, HttpStatus.OK);
        log.info("qna 개수 확인: "+list.size());
        return res;
    }
```

Bean Scope

Spring은 기본적으로 모든 Bean을 Singleton으로 생성하여 관리함

따라서 Spring을 통해 Bean을 주입받으면 언제나 주입받은 Bean은 동일한 객체라는 가정하에 개발을 하게 됨

컨테이너와 빈의 생명주기

컨테이너 자체에도 생명 주기가 있다.

컨테이너는 초기화 될 때 Bean 객체를 등록, 생성, 주입하고

컨테이너가 종료될 때 Bean을 소멸시킨다.

컨테이너와 빈의 생명주기

1. 설정 파일을 읽어 Bean 객체 생성 (=인스턴스화)
2. Bean 객체에 의존성 주입
3. 빈 생성 생명주기 콜백 (초기화)

   -> 어플리케이션에서 빈 사용
4. 빈 소멸 생명주기 콜백

빈의 생명주기 콜백 3가지

1. 스프링에서 제공하는 인터페이스
2. 설정 정보에서 초기화 메서드, 종료 메서드 지정
3. @PostConstruct, @PreDestroy 어노테이션

@PostConstruct, @PreDestroy 어노테이션

스프링에 종속적인 기술이 아닌, 자바 표준 코드이기 때문에 다른 컨테이너에서도 동작

```
@PostConstruct
public void initialize() throws Exception {
    // 초기화
}
@PreDestroy
public void close() throws Exception {
    // 메모리 반납, 연결 종료와 같은 과정
    }
}
```

결론

빈의 생명 주기는

1. 객체 생성 및 의존관계 주입
2. 초기화
3. 사용
4. 종료