

## 1) 자바를 이용한 DI 설정 방식의 2가지 주요 애노테이션을 설명하시오.

Spring3.0 버전부터 애노테이션을 이용하여 빈과 관련된 정보를 설정할 수 있게 되었다. XML 설정 없이 자바 코드를 이용해서 빈 객체 생성과 빈 객체 간의 의존 관계 설정까지 할 수 있다. 하나의 클래스 안에 복수의 빈을 등록할 수도 있다. @Bean 애노테이션과 메서드 이름을 이용해서 컨테이너가 사용할 빈 객체를 생성한다. 자바 설정 방식에서는 빈 객체를 직접 생성한다. @Bean 메서드를 불러들여서 객체를 취득한다. XML에서는 <property> 태그나 <constructor-arg> 태그를 이용해서 설정하였으나 자바 설정에서는 직접 의존 객체를 주입해야 한다.

### 1) - ① @Configuration

빈 설정 메타 정보를 담고 있는 클래스를 선언한다. 해당 클래스가 스프링 설정으로 사용된다.

### 1) - ② @Bean

클래스 내의 메서드를 정의하여 새로운 빈 객체를 정의할 때 사용한다. name 속성을 사용하여 새로운 빈 이름을 적용할 수 있다.

## 2) 빈 객체 라이프 사이클에서 빈을 생성 후 초기화하는 방법과 빈 종료 전 전처리 과정을 수행할 수 있는 방법을 4가지를 정리하시오.

빈 객체의 라이프사이클은 초기화 -> 이용 -> 종료 3단계로 진행되고, 빈 생성 후 초기화 작업과 빈 종료 전 전처리 과정을 수행할 수 있는 방법을 제공한다.

### 2) - ① XML 기반

초기화 작업 : <bean> 요소의 init-method 속성에 메서드 지정

전처리 작업 : <bean> 요소의 destroy-method 속성에 메서드 지정

### 2) - ② 애노테이션 기반

초기화 작업 : @PostConstruct 애노테이션이 붙은 메소드를 선언

전처리 작업 : @PreDestroy 애노테이션이 붙은 메소드를 선언

### 2) - ③ 자바 기반

초기화 작업 : @Bean의 initMethod 속성에 메소드를 지정하고 그 메소드를 호출

전처리 작업 : @Bean의 destroyMethod 속성에 메소드를 지정하고 그 메소드를 호출

### 2) - ④ 인터페이스 구현

초기화 작업 : InitializeBean 인터페이스의 agterPropertiesSet 메소드를 구현

전처리 작업 : DisposableBean 인터페이스의 destroy 메소드를 구현

- 3) di-annotation.zip과 db-springjdbc.zip을 참고하여 실습하고 본인이 생성한 2개의 프로젝트 결과 출력 페이지를 PringScreen하여 첨부하시오.

### 3) - ① IHPARK-dianno

The screenshot shows the Spring Tool Suite IDE with the project 'IHPARK-dianno' open. The file 'MemberSampleMain.java' is selected in the Package Explorer. The code in the file is as follows:

```
13 // TODO Auto-generated method stub
14 System.out.println("안녕하세요 DI-ANNOTATION");
15
16 //
17 ctx = new GenericXmlApplicationContext("classpath:applicationContext.xml");
18 ctx = new ClassPathXmlApplicationContext("applicationContext.xml");
19
20 MemberService memberService = (MemberService) ctx.getBean(MemberService.class); // by Class name
21 MemberService memberService = (MemberService) ctx.getBean("memberService"); // by Component name
22
23 StudentVO vo = new StudentVO();
24 vo.setId("ihp001");
25 vo.setPassword("1234");
26 vo.setUsername("PARKINHYO");
27 vo.setSnum("2015154017");
28 vo.setDepart("SK물류");
29 vo.setMobile("010-4733-2189");
30 vo.setEmail("inhyopark122@gmail.com");
31
32 memberService.addMember(vo);
33 StudentVO member = memberService.readMember("ihp001");
34
35 System.out.println(member);
36
37
38 }
```

The Console window shows the following output:

```
15:30:08.843 [main] DEBUG org.springframework.context.support.ClassPathXmlApplicationContext - Refreshing org.springframework.context.support.ClassPathXmlApplic
15:30:09.242 [main] DEBUG org.springframework.context.support.ClassPathXmlApplicationContext - Identified candidate component class: file [C:\PARKINHYO\2020-
15:30:09.290 [main] DEBUG org.springframework.context.support.ClassPathXmlApplicationContext - Identified candidate component class: file [C:\PARKINHYO\2020-
15:30:09.290 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Loaded 7 bean definitions from class path resource [applicationContext
15:30:09.330 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'org.springframework
15:30:09.384 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'org.springframework
15:30:09.386 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'org.springframework
15:30:09.388 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'org.springframework
15:30:09.390 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'org.springframework
15:30:09.403 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'memberDAOImpl'
15:30:09.414 [main] DEBUG org.springframework.beans.factory.support.DefaultListableBeanFactory - Creating shared instance of singleton bean 'memberServiceImpl'
StudentVO [id=ihp001, passwd=1234, username=PARKINHYO, snum=2015154017, depart=SK물류, mobile=010-4733-2189, email=inhyopark122@gmail.com]
```

### 3) - ② IHPARK-spjdbc

The screenshot shows the Spring Tool Suite IDE with the project 'IHPARK-spjdbc' open. The file 'MemberSampleMain.java' is selected in the Package Explorer. The code in the file is as follows:

```
20 ctx = new ClassPathXmlApplicationContext("classpath:applicationContext.xml");
21
22 MemberService memberService = ctx.getBean(MemberService.class); // by Class name
23
24 String strID = "ihp001";
25 StudentVO vo = new StudentVO();
26 vo.setId(strID);
27 vo.setPassword(strID);
28 vo.setUsername(strID);
29 vo.setSnum(strID);
30
31 try {
32     memberService.addMember(vo);
33     StudentVO member = memberService.readMember(strID);
34     System.out.println(member);
35 }
36
37 List<StudentVO> list = memberService.readMemberList();
38 for (StudentVO svo : list) {
39     System.out.println(svo);
40 }
41
42 } catch (DataAccessException e) {
43     System.out.println(e);
44 }
45
46 finally { // Check Count
47     JdbcTemplate jdbcTemplate = ctx.getBean(JdbcTemplate.class);
48     int count = jdbcTemplate.queryForObject("SELECT COUNT(*) FROM STUDENT", Integer.class);
49 }
```

The Console window shows the following output:

```
15:38:48.399 [main] DEBUG org.springframework.jdbc.core.BeanPropertyRowMapper - Mapping column 'snum' to property 'snum' of type 'java.lang.String'
15:38:48.399 [main] DEBUG org.springframework.jdbc.core.BeanPropertyRowMapper - Mapping column 'depart' to property 'depart' of type 'java.lang.String'
15:38:48.399 [main] DEBUG org.springframework.jdbc.core.BeanPropertyRowMapper - Mapping column 'mobile' to property 'mobile' of type 'java.lang.String'
15:38:48.399 [main] DEBUG org.springframework.jdbc.core.BeanPropertyRowMapper - Mapping column 'email' to property 'email' of type 'java.lang.String'
StudentVO [id=ihp001, passwd=ihp001, username=박인호, snum=2015154017, depart=물류(ㄱ), mobile=01047332189, email=inhyopark122@gmail.com]
15:38:48.403 [main] DEBUG org.springframework.jdbc.core.JdbcTemplate - Executing SQL query [SELECT * FROM STUDENT]
StudentVO [id=hansol, passwd=hansol, username=한솔, snum=2012111000, depart=컴퓨터(ㄱ), mobile=0111119999, email=2012111000@kpu.ac.kr]
StudentVO [id=ihp001, passwd=ihp001, username=박인호, snum=2015154017, depart=물류(ㄱ), mobile=01047332189, email=inhyopark122@gmail.com]
StudentVO [id=ok9938, passwd=ok9938, username=정현, snum=2015154023, depart=컴퓨터(ㄱ), mobile=0102245678, email=ok9938@gmail.com]
StudentVO [id=shp001, passwd=shp001, username=서현, snum=2014777777, depart=IT, mobile=01052201510, email=shp001@gmail.com]
StudentVO [id=thoonk, passwd=thoonk, username=조현, snum=2015152077, depart=IT, mobile=01011112222, email=thoonk@gmail.com]
15:38:48.414 [main] DEBUG org.springframework.jdbc.core.JdbcTemplate - Executing SQL query [SELECT COUNT(*) FROM STUDENT]
15:38:48.414 [main] DEBUG org.springframework.jdbc.datasource.DataSourceUtils - Fetching JDBC Connection from DataSource
5
```