



박경태  
[comsi.java@gmail.com](mailto:comsi.java@gmail.com)

# 고급 자바 프로그래밍 : STS를 이용한 Spring 프로그래밍



# 강의 내용

순서	내 용
1	<ul style="list-style-type: none"><li>• Spring IoC를 이용한 비즈니스 컴포넌트 만들기</li></ul>
2	<ul style="list-style-type: none"><li>• Spring AOP(Aspect Oriented Programming)를 이용한 공통 서비스 만들기</li><li>• Spring DAO(Data Access Object)를 이용한 데이터베이스 연동 및 트랜잭션 처리</li></ul>
3	<ul style="list-style-type: none"><li>• Spring MVC를 이용한 MVC 아키텍쳐 적용하기</li></ul>
4	<ul style="list-style-type: none"><li>• Spring MVC의 부가 기능 사용하기(파일 업로드, 다국어, 예외 처리 등)</li></ul>
5	<ul style="list-style-type: none"><li>• Spring과 MyBatis 연동하기</li><li>• Spring과 JPA 연동하기</li></ul>

## 5. 어노테이션 기반 설정

XML 파일의 과도한 설정에 대한 부담 감소를 위해 어노테이션  
을 이용한 설정을 지원

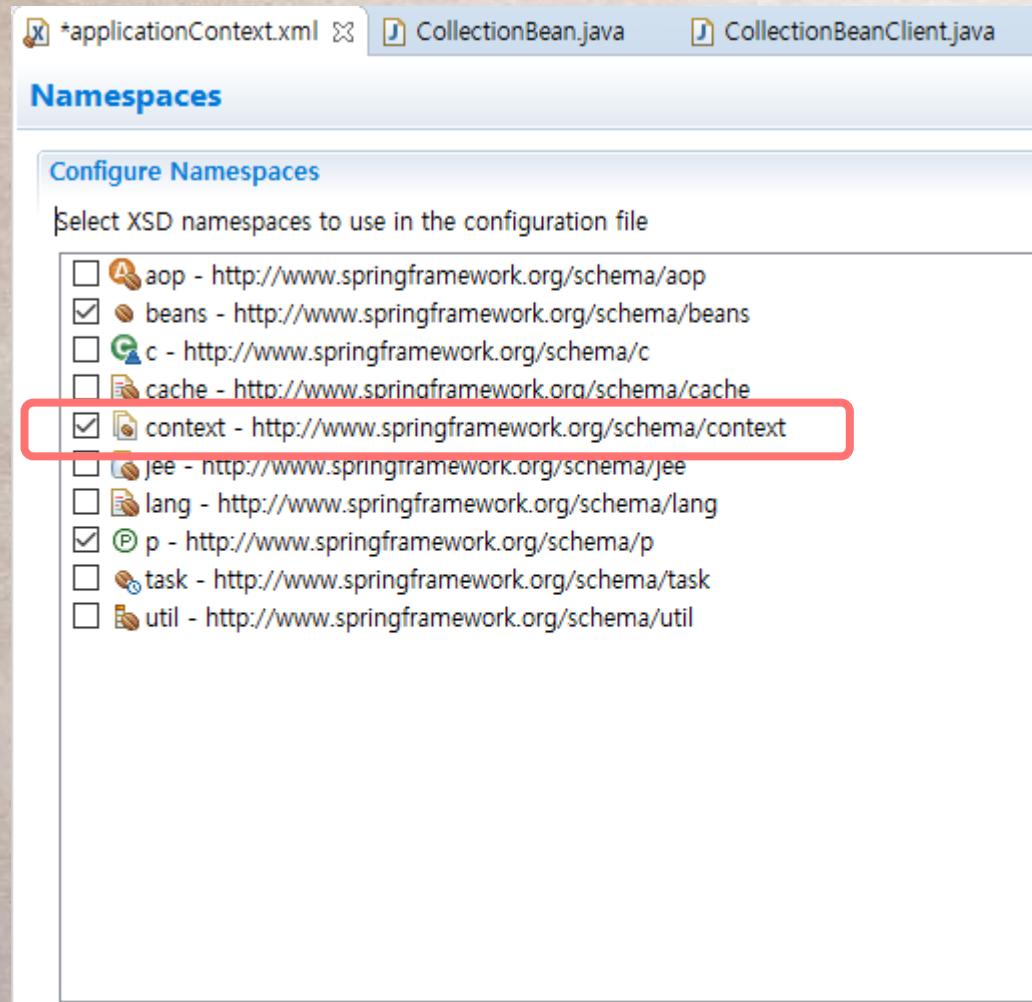
# 어노테이션??

- 일반적으로 Java Annotation은 @를 이용한 주석을 뜻함.
- 자바 어노테이션 (Java Annotation)은 자바5(1.5)부터 제공되기 시작한 기능으로 어노테이션은 한국어로 “주석”이라 번역.
- Annotation을 “주석”이라 칭할경우, 흔히 쓰던 주석처리 (//, /\*\*/) 와 같은 의미로 오해하기 쉽기 때문에, Annotation이라는 명칭을 그대로 사용.
- 주석과 Annotation으로 구분하지만 둘다 프로그래밍의 구동에 영향을 주지 않는 주석임.
- Annotation을 사용하면 코드의 가독성을 높이는 데 도움
  - @Deprecated - 가급적 사용을 자제해 달라는 의미로 개선된 메소드가 있음을 나타내고자 할 때 사용
  - @SuppressWarnings(“ ”) - 이 부분에 경고문을 출력하지 말라는 의미
  - @SuppressWarnings(“ deprecation ”) - deprecated 된 메소드를 이용했을 때 발생하는 deprecation 경고문을 띄우지 말라는 의미

## 5.1 어노테이션 설정 기초

# 1. Context 네임스페이스 추가

- 어노테이션 설정 추가- context 관련 네임스페이스와 스키마 문서 위치 등록

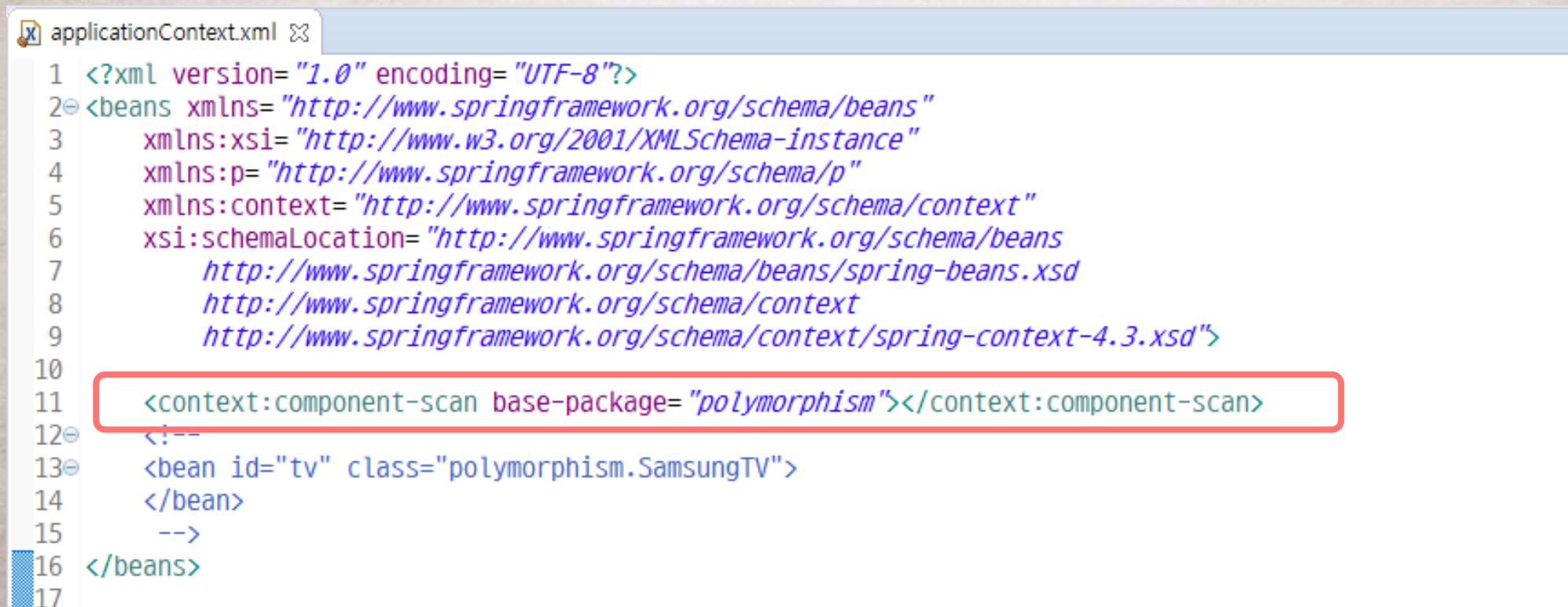


```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:n="http://www.springframework.org/schema/n"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/context
                           http://www.springframework.org/schema/context/spring-context-4.3.xsd">
```

The screenshot shows the Eclipse IDE XML code editor for 'applicationContext.xml'. The XML code defines a Spring bean configuration. It includes declarations for the 'beans' and 'context' namespaces. The 'context' namespace is declared with its URI and schema location. Lines 5 and 6, which define the 'context' namespace and its schema location, are highlighted with a red box. The tabs at the top show 'applicationContext.xml', 'CollectionBean.java', and 'CollectionBeanClient.java'.

## 2. 컴포넌트 스캔(component-scan) 설정

- <bean> 등록하지 않고 자동으로 생성하려면 <context:component-scan/> 엘리먼트를 정의
- 클래스 패스에 있는 클래스들을 스캔하여 @Component가 설정된 클래스들을 자동으로 객체 생성



The screenshot shows an IDE interface with the file "applicationContext.xml" open. The XML code defines a Spring application context with various configurations. A red rectangular box highlights the `<context:component-scan base-package="polymorphism"></context:component-scan>` line, indicating it is the key configuration for component scanning.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2<beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xmlns:p="http://www.springframework.org/schema/p"
5     xmlns:context="http://www.springframework.org/schema/context"
6     xsi:schemaLocation="http://www.springframework.org/schema/beans
7         http://www.springframework.org/schema/beans/spring-beans.xsd
8         http://www.springframework.org/schema/context
9         http://www.springframework.org/schema/context/spring-context-4.3.xsd">
10
11    <context:component-scan base-package="polymorphism"></context:component-scan>
12<!--
13<bean id="tv" class="polymorphism.SamsungTV">
14</bean>
15  -->
16</beans>
17
```

### 3. @Component

- LgTV 클래스에 대한 <bean> 등록

```
<bean class="polymorphism.LgTV"></bean>
```

<bean> 객체 생성

The screenshot shows an IDE interface with two tabs: 'applicationContext.xml' and 'LgTV.java'. The 'LgTV.java' tab is active, displaying the following code:

```
1 package polymorphism;
2
3 import org.springframework.stereotype.Component;
4
5 @Component
6 public class LgTV implements TV {
7
8     public LgTV(){
9         System.out.println("==> LgTV 객체 생성");
10    }
11}
```

@Component annotation 사용

- 컨테이너가 생성한 객체 요청

```
// Spring 컨테이너를 구동
AbstractApplicationContext factory = new GenericXmlApplicationContext("applicationContext.xml");

//BeanFactory factory = new BeanFactory();
TV tv = (TV) factory.getBean("tv")
```

컨테이너가 생성한 객체 요청시에는 요청할  
아이디나 이름 필요

- LgTV 클래스에 대한 <bean> 등록

```
<bean id="tv" class="polymorphism.LgTV"></bean>
```

```
applicationContext.xml LgTV.java TVUser.java
1 package polymorphism;
2
3 import org.springframework.stereotype.Component;
4
5 @Component("tv")
6 public class LgTV implements TV {
7
8     public LgTV(){
9         System.out.println("==> LgTV 객체 생성");
10    }
11}
```

컨테이너가 생성한 객체 요청시에는 요청할  
아이디나 이름 필요

- id나 name 속성 미지정 시 이름규칙

- 스프링 컨테이너가 클래스 객체를 생성할 때, id나 name 속성을 지정하지 않았다면, 컨테이너가 자동으로 이름을 설정
- 이름 규칙은 클래스 이름의 첫글자를 소문자로 변경하기만 하면된다.
- LgTV 클래스 객체를 요청하면 lgTV라는 이름을 사용하면됨

# 실행결과

The screenshot shows an IDE interface with the following details:

- Project Explorer:** Shows files: applicationContext.xml, LgTV.java, and TVUser.java.
- Code Editor:** Displays the TVUser.java code. The code demonstrates polymorphism by creating a Spring context from an XML file and retrieving a TV bean. It then calls methods like powerOn(), volumeUp(), volumeDown(), and powerOff() on the TV object.
- Console Output:** Shows the execution results:
  - Information about loading bean definitions from applicationContext.xml.
  - Information about preparing the context.
  - Log messages indicating the TV's actions: turning on, increasing volume, decreasing volume, and turning off.
  - Information about closing the context.

## 5.2 의존성 주입 설정

# 1. 의존성 주입 어노테이션

- 스프링에서 의존성 주입을 지원하는 어노테이션은 `@Autowired`, `@Inject`, `@Qualifier`, `@Resource`가 있다.

어노테이션	설명
<code>@Autowired</code>	주로 변수 위에 설정하여 해당 타입의 객체를 찾아서 자동으로 할당 <code>org.springframework.beans.factory.annotation.Autowired</code>
<code>@Qualifier</code>	특정 객체의 이름을 이용하여 의존성 주입할 때 사용 <code>org.springframework.beans.factory.annotation.Qualifier</code>
<code>@Inject</code>	<code>@Autowired</code> 과 동일한 제공 <code>javax.annotation.Resource</code>
<code>@Resource</code>	<code>@Autowired</code> 과 <code>@Qualifier</code> 의 기능을 결합한 어노테이션 <code>javax.inject.Inject</code>

- `@Autowired`과 `@Qualifier`는 스프링에서 제공하지만 나머지는 스프링에서 제공하지 않는다.

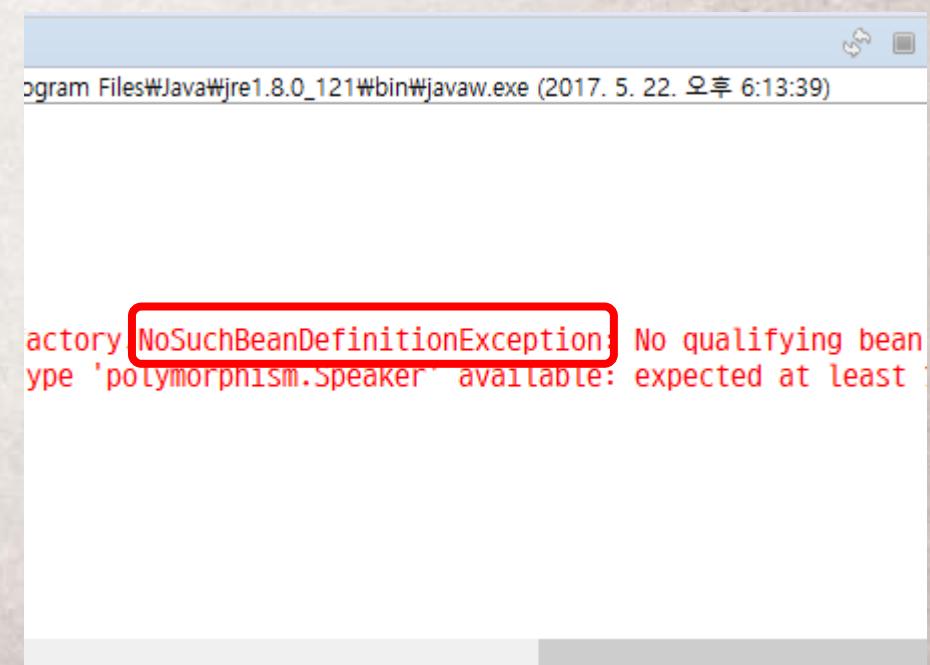
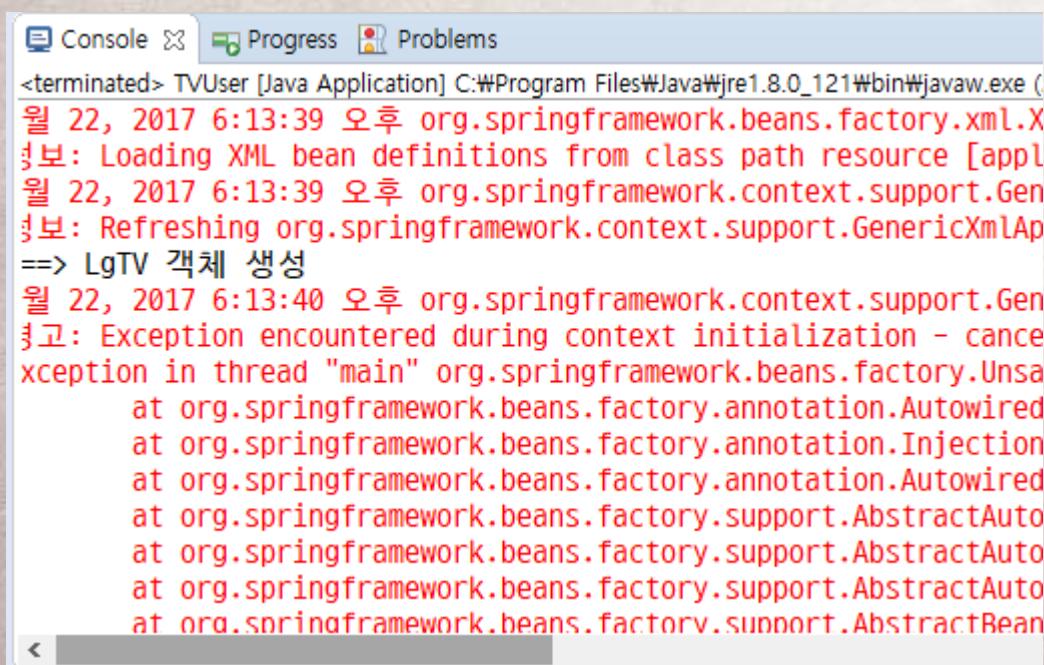
## 2. @Autowired

- 생성자나 메소드, 멤버변수 위에 모두 사용
- 어디서 사용하는 결과가 같아 상관없지만 대부분 멤버변수 위에 선언하여 사용
- 스프링 컨테이너는 멤버변수 위에 붙은 `@Autowired`를 확인하는 순간 해당 변수의 타입 체크 후 메모리에서 확인 후 그 객체를 변수에 주입
- 만약 `@Autowired`가 붙은 객체가 메모리에 없다면 `NoSuchBeanDefinitionException`을 발생 → 대상 객체가 메모리에 존재하지 않는다는 의미

## @Autowired 객체가 없을 경우 발생 오류

```
1 package polymorphism;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Component;
5
6 @Component("tv")
7 public class LgTV implements TV {
8
9     @Autowired
10    private Speaker speaker;
11
12    public LgTV(){
13        System.out.println("==> LgTV 객체 생성");
14    }
15}
```

외부 Speaker를 사용  
(SonySpeaker 또는 AppleSpeaker 객체)

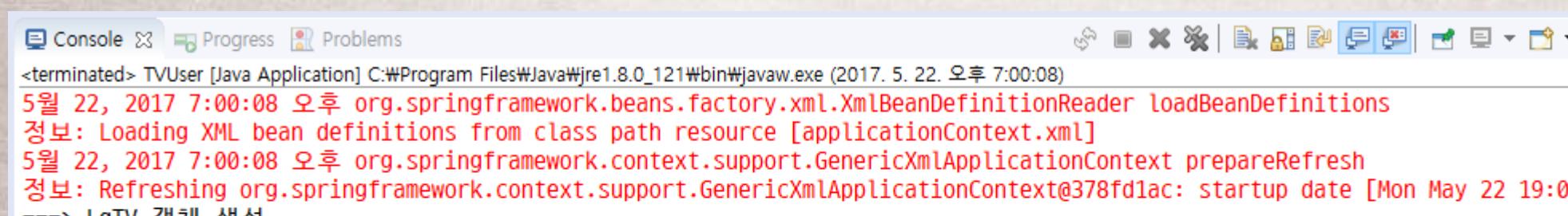


# 발생 오류 제거 - Speaker 객체를 메모리에 생성



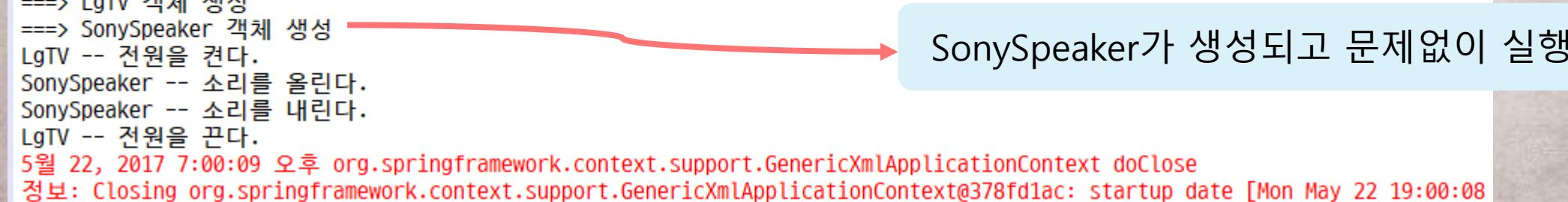
```
applicationContext.xml LgTV.java TVUser.java SonySpeaker.java Speaker.java
1 package polymorphism;
2
3 import org.springframework.stereotype.Component;
4
5 @Component("sony")
6 public class SonySpeaker implements Speaker{
7
8     public SonySpeaker(){
9         System.out.println("==> SonySpeaker 객체 생성");
10    }
11
12    public void volumeUp(){
13        System.out.println("SonySpeaker -- 소리를 올린다.");
14    }
15
16    public void volumeDown(){
17        System.out.println("SonySpeaker -- 소리를 내린다.");
18    }
19 }
```

@Component 를 사용해 Speaker 객체를 컨테이너에 생성



Console Progress Problems

```
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 5. 22. 오후 7:00:08)
5월 22, 2017 7:00:08 오후 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
5월 22, 2017 7:00:08 오후 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 19:00:
==> LgTV 객체 생성
==> SonySpeaker 객체 생성
LgTV -- 전원을 켠다.
SonySpeaker -- 소리를 올린다.
SonySpeaker -- 소리를 내린다.
LgTV -- 전원을 끈다.
```



SonySpeaker가 생성되고 문제없이 실행

```
5월 22, 2017 7:00:09 오후 org.springframework.context.support.GenericXmlApplicationContext doClose
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 19:00:08]
```

### 3. @Qualifier

- 의존성 주입 대상이 되는 Speaker 타입 객체가 두개 이상일때는?
  - AppleSpeaker와 SonySpeaker 객체가 모두 메모리에 있다면?
  - 컨테이너가 어떤 객체를 할당할지 판단할 수 없기 때문에 에러 발생
- AppleSpeaker에도 @Component 선언해보자.



```
1 package polymorphism;
2
3 import org.springframework.stereotype.Component;
4
5 @Component("apple")
6 public class AppleSpeaker implements Speaker {
7
8     public AppleSpeaker(){
9         System.out.println("==> AppleSpeaker 객체 생성");
10    }
11
12    @Override
13    public void volumeUp() {
14        System.out.println("AppleSpeaker -- 소리를 올린다.");
15    }
16
17    @Override
18    public void volumeDown() {
19        System.out.println("AppleSpeaker -- 소리를 내린다.");
20    }
21
22 }
```

AppleSpeaker 클래스에 대한 annotation 추가

# 실행 결과

The screenshot shows an IDE interface with a toolbar at the top and three tabs: 'Console', 'Progress', and 'Problems'. The 'Console' tab is active, displaying the following error message:

```
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 5. 22. 오후 7:10:53)

BeanDefinitionException: No qualifying bean of type 'polymorphism.Speaker' available: expected single matching bean but found 2: apple,sony
Speaker available: expected single matching bean but found 2: apple,sony
```

The error message is highlighted with a red rectangular box.

- @Autowired 대상이 되는 Speaker 타입의 객체가 AppleSpeaker, SonySpeaker로 두 개이고 둘다 메모리에 있으므로 의존성 주입에 문제 발생

# 해결 방법은 @Qualifier 어노테이션을 사용

The screenshot shows an IDE interface with several tabs at the top: LgTV.java, TVUser.java, SonySpeaker.java, and AppleSpeaker.java. The LgTV.java tab is active, displaying the following code:

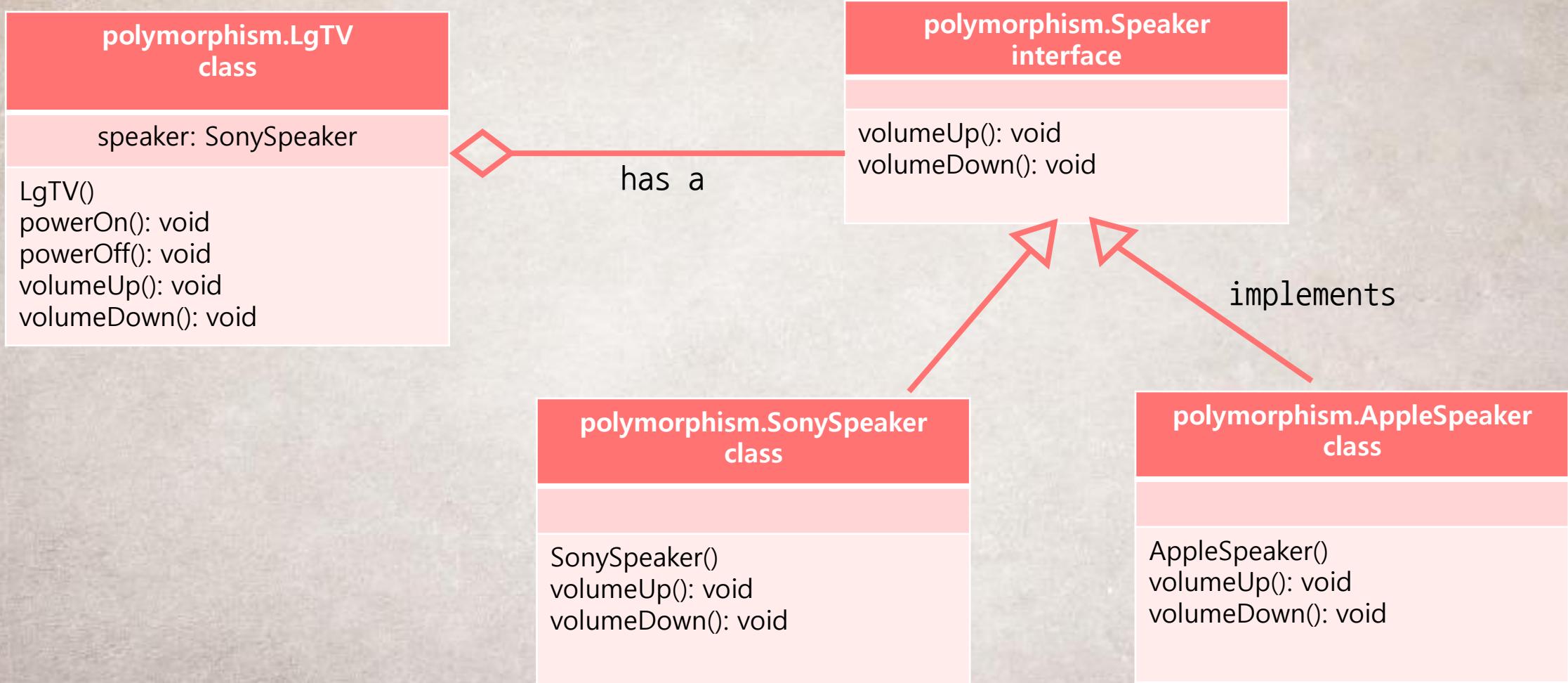
```
1 package polymorphism;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.beans.factory.annotation.Qualifier;
5 import org.springframework.stereotype.Component;
6
7 @Component("tv")
8 public class LgTV implements TV {
9
10    @Autowired
11    @Qualifier("apple")
12    private Speaker speaker;
13
14    public LgTV(){
15        System.out.println("==> LgTV 객체 생성");
16    }
17}
```

A red box highlights the `@Qualifier("apple")` annotation on the `speaker` field. A red arrow points from this highlighted code to the terminal window below, which displays the following log output:

```
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 5. 22. 오후 7:45:39)
5월 22, 2017 7:45:40 오후 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
5월 22, 2017 7:45:40 오후 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 19:45:
→ ==> AppleSpeaker 객체 생성
→ ==> LgTV 객체 생성
→ ==> SonySpeaker 객체 생성
LgTV -- 전원을 끈다.
AppleSpeaker -- 소리를 올린다.
AppleSpeaker -- 소리를 내린다.
LgTV -- 전원을 끈다.
5월 22, 2017 7:45:40 오후 org.springframework.context.support.GenericXmlApplicationContext doClose
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 19:45:40]
```

Two additional red boxes highlight the log entries for the `AppleSpeaker` object creation and its methods, and the final shutdown message for the `LgTV` object.

# TV 컴포넌트의 클래스 다이어그램



## 4. @Resource

- @Autowired는 변수의 타입을 기준으로 객체의 의존성 주입하지만
- **@Resource는 객체의 이름을 이용하여 의존성 주입을 처리**
  - name 속성을 사용하여 스프링 컨테이너가 해당 이름의 객체를 검색하여 의존성 주입 처리



```
LgTV.java TVUser.java SonySpeaker.java AppleSpeaker.java
1 package polymorphism;
2
3 import javax.annotation.Resource;
4
5 import org.springframework.stereotype.Component;
6
7 @Component("tv")
8 public class LgTV implements TV {
9
10    @Resource(name="sony")
11    private Speaker speaker;
12
13    public LgTV(){
14        System.out.println("==> LgTV 객체 생성");
15    }
}
```

# 실행 결과



The screenshot shows a Java application running in an IDE's console. The output text is in Korean, indicating the loading of bean definitions from 'applicationContext.xml' and the creation of instances for 'AppleSpeaker', 'LgTV', and 'SonySpeaker'. It also shows the speakers playing sound and the TV turning off. A red box highlights the lines where the SonySpeaker instance is shown playing sound.

```
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 5. 22. 오후 7:53:06)
5월 22, 2017 7:53:06 오후 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
5월 22, 2017 7:53:07 오후 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 19:53:
==> AppleSpeaker 객체 생성
==> LgTV 객체 생성
==> SonySpeaker 객체 생성
LgTV -- 전원을 켜다.
SonySpeaker -- 소리를 올린다.
SonySpeaker -- 소리를 내린다.
LgTV -- 전원을 끈다.
5월 22, 2017 7:53:07 오후 org.springframework.context.support.GenericXmlApplicationContext doClose
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 19:53:07]
```

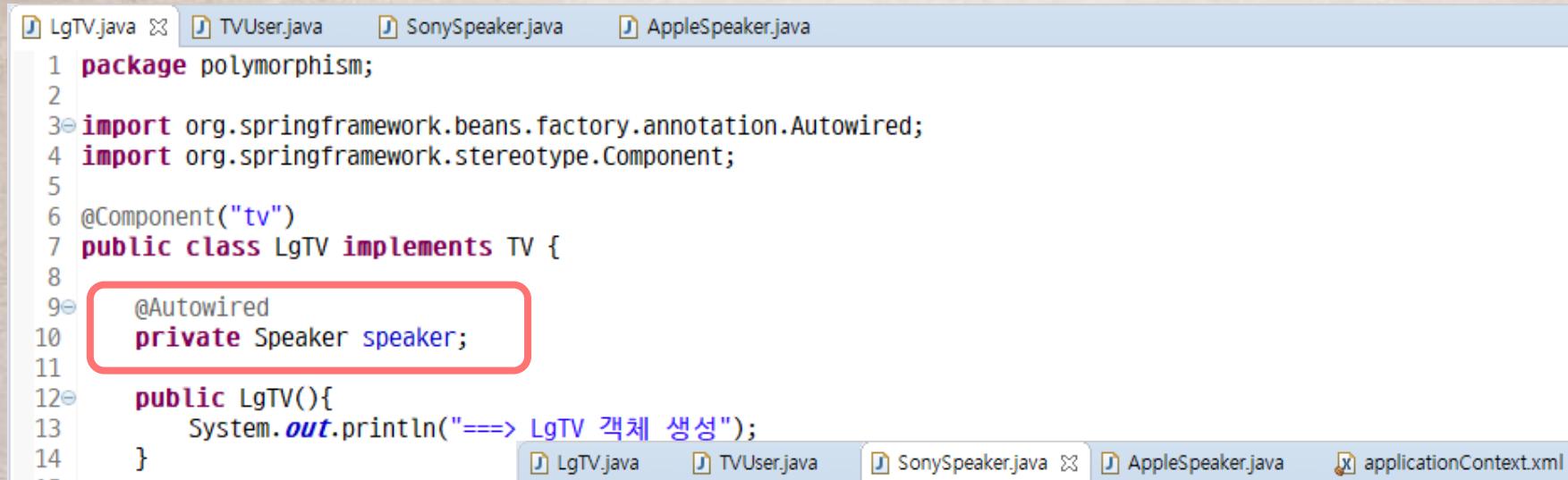
- @Resource와 같은 기능의 어노테이션으로 @Inject가 있고 둘 다 이름 기반의 의존성 주입을 처리한다.

# 5. 어노테이션과 XML 설정 병행하여 사용하기

- XML 설정 방식
  - XML 방식은 자바 소스를 수정하지 않고 XML 파일의 설정만 변경하면 실행되는 Speaker를 교체할 수 있어서 유지보수가 편함
  - XML 설정에 대한 부담이 역시 존재
  - 자바 소스에 의존관계와 관련된 메타데이터가 없으므로 XML 설정을 해석해야만 의존성 주입을 확인할 수 있다.
- 오노테이션 방식
  - XML 설정에 대한 부담이 없고, 의존관계에 대한 정보가 자바 소스에 있어서 사용하기 편하다.
  - 의존성 주입할 객체의 이름이 자바소스에 명시되어야 하므로 자바소스를 수정하지 않고 Speaker를 교체할 수 없다는 문제가 발생

# 병행하여 사용하기

- 자동 객체 생성을 위한 @Component를 제거하고 사용할 객체를 스프링 설정 파일에 <bean> 등록하여 처리



```
LgTV.java TVUser.java SonySpeaker.java AppleSpeaker.java
1 package polymorphism;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Component;
5
6 @Component("tv")
7 public class LgTV implements TV {
8
9     @Autowired
10    private Speaker speaker;
11
12    public LgTV(){
13        System.out.println("==> LgTV 객체 생성");
14    }
15}
```



```
SonySpeaker.java applicationContext.xml
1 package polymorphism;
2
3 public class SonySpeaker implements Speaker{
4
5     public SonySpeaker(){
6         System.out.println("==> SonySpeaker 객체 생성");
7     }
8
9     @Override
10    public void volumeUp(){
11        System.out.println("SonySpeaker -- 소리를 올린다.");
12    }
13
14    @Override
15    public void volumeDown(){
16        System.out.println("SonySpeaker -- 소리를 내린다.");
17    }
18}
```

@Component를 제거

```
LgTV.java TVUser.java SonySpeaker.java AppleSpeaker.java *applicationContext.xml AbstractApplicationContext.class
1 <?xml version="1.0" encoding="UTF-8"?>
2<beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:p="http://www.springframework.org/schema/p"
5   xmlns:context="http://www.springframework.org/schema/context"
6   xsi:schemaLocation="http://www.springframework.org/schema/beans
7       http://www.springframework.org/schema/beans/spring-beans.xsd
8       http://www.springframework.org/schema/context
9       http://www.springframework.org/schema/context/spring-context-4.3.xsd">
10
11<context:component-scan base-package="polymorphism"></context:component-scan>
12<bean class="polymorphism.SonySpeaker"></bean>
13</beans>
14
<context:component-scan base-package="polymorphism"></context:component-scan>
<bean class="polymorphism.AppleSpeaker"></bean>
```

AppleSpeaker 사용할 경우

## • 실행 결과

```
Console Progress Problems
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 5. 22. 오후 8:35:55)
5월 22, 2017 8:35:55 오후 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
5월 22, 2017 8:35:55 오후 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 20:35
==> LgTV 객체 생성
==> SonySpeaker 객체 생성
LgTV -- 전원을 켠다.
SonySpeaker -- 소리를 올린다.
SonySpeaker -- 소리를 내린다.
LgTV -- 전원을 끈다.
5월 22, 2017 8:35:56 오후 org.springframework.context.support.GenericXmlApplicationContext doClose
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@378fd1ac: startup date [Mon May 22 20:35:55]
```

# 병행사용하기

- XML 설정만 사용하기에는 <bean> 등록을 많이 해야 하고 의존관계 설정도 부담
- 어노테이션을 사용하자니 자바 소스를 수행해야 의존성 관계를 변경  
→ 변경되지 않는 객체는 어노테이션으로 설정하여 사용하고 변경될 가능성이 있는 객체는 XML 설정을 사용  
→ 우리가 직접 개발한 클래스는 XML, 어노테이션 둘 모두 사용 가능하나 라이브러리 형태로 제공하는 클래스는 XML 설정을 통해서만 사용 가능
  - Apache에서 제공하는 DB연동 처리를 할 경우에는 ' commons-dbcp-1.5.jar ' 파일을 사용해야 할 경우, XML에서 <bean> 등록을 해야 한다.

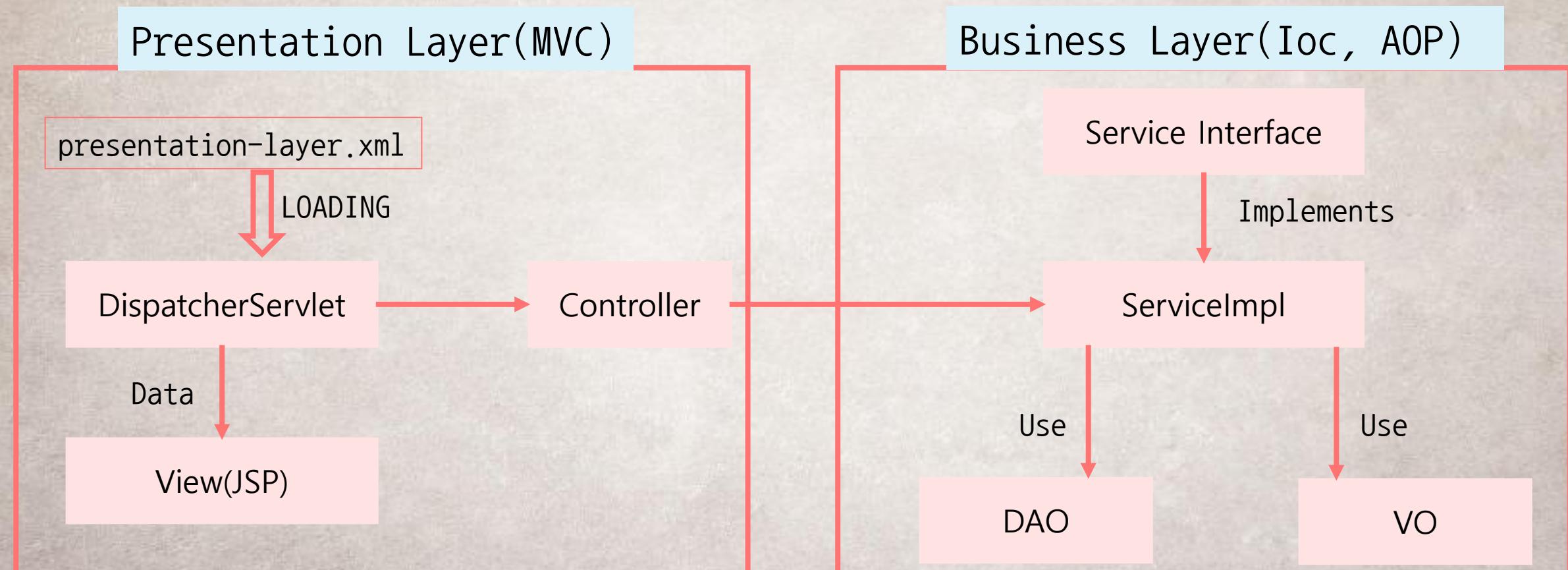
# DataSource 설정 예

```
LgTV.java TVUser.java SonySpeaker.java AppleSpeaker.java applicationContext.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2<beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:p="http://www.springframework.org/schema/p"
5   xmlns:context="http://www.springframework.org/schema/context"
6   xsi:schemaLocation="http://www.springframework.org/schema/beans
7     http://www.springframework.org/schema/beans/spring-beans.xsd
8     http://www.springframework.org/schema/context
9     http://www.springframework.org/schema/context/spring-context-4.3.xsd">
10
11<context:component-scan base-package="polymorphism"></context:component-scan>
12<bean class="polymorphism.AppleSpeaker"></bean>
13
14<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
15   <property name="driverClassname" value="org.h2.Driver"/>
16   <property name="url" value="jdbc:h2:tcp://localhost/~test"/>
17   <property name="username" value="sa"/>
18   <property name="password" value="" />
19 </bean>
20</beans>
21
```

## 5.3 추가 어노테이션

# 시스템의 전체 구조를 두 개의 레이아웃으로 표현

- 프리젠테이션 레이어 - 사용자와의 커뮤니케이션 담당
- 비즈니스 레이어 - 사용자의 요청에 대한 비즈니스 로직 처리 담당



프리젠테이션과 비즈니스 레이어로 구성된 시스템 구조

# 시스템의 전체 구조를 두 개의 레이아웃으로 표현

- 가장 핵심 요소는 Controller, ServiceImpl, DAO 클래스
  - Controller - 사용자의 요청을 제어
  - ServiceImpl - 여러 개의 DAO요소를 사용한 실제적인 비즈니스 로직을 처리
  - DAO(Data Access Object) - 데이터 베이스 연동을 담당
- 시스템 구성요소들을 스프링 컨테이너에서 해당 클래스 객체를 생성할 경우 클래스의 역할을 파악하기 어렵다.

어노테이션	위치	의미
@Service	XXXServiceImpl	비즈니스 로직을 처리하는 Service 클래스
@Repository	XXXDAO	데이터베이스 연동을 처리하는 DAO 클래스
@Controller	XXXController	사용자 요청을 제어하는 Controller 클래스

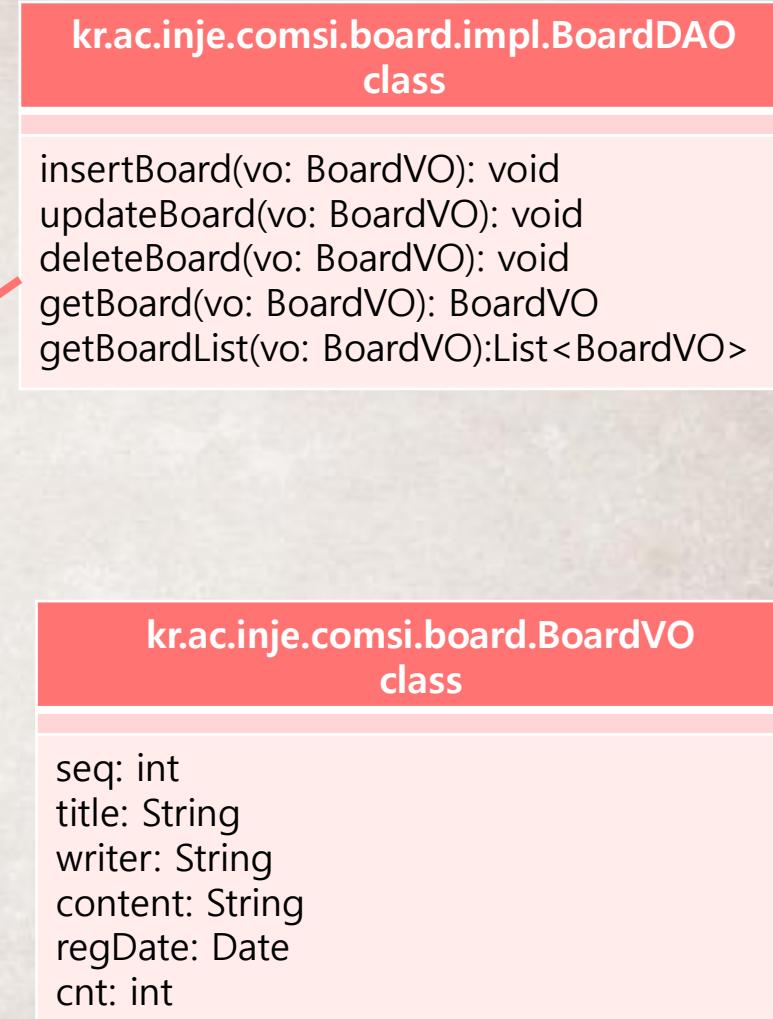
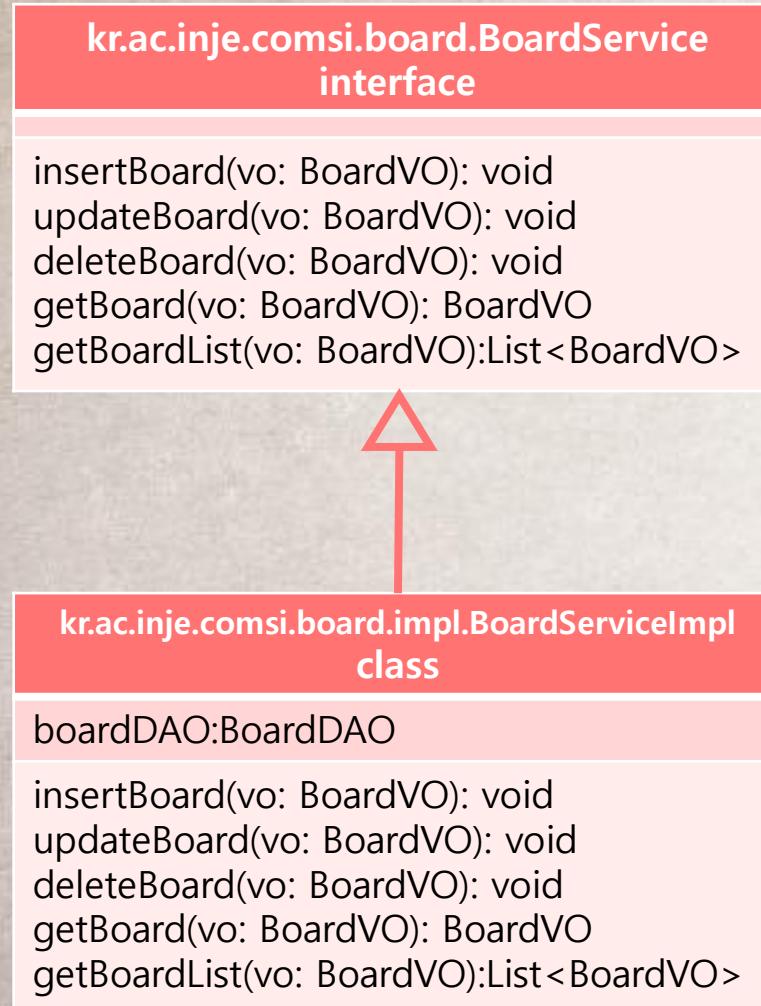
→ 각 클래스를 해당 객체의 기능으로 인식하도록 함

# 6. 비즈니스 컴포넌트 실습1

## 6.1 BoardService 컴포넌트 구조

- 비즈니스 컴포넌트는 4개의 자바 파일로 구성
- 각 자바 파일을 작성하는 순서와 이름 규칙도 어느 정도 정해져 있음
- 게시판 관련 컴포넌트를 구현하면서 비즈니스 컴포넌트의 작성 순서와 이름 규칙을 살펴볼 것임

# 6.1 BoardService 컴포넌트 구조



- BoardService 컴포넌트 클래스 다이어그램

# 실습 프로젝트의 구조와 파일 위치

## 6.2 Value Object 클래스 작성

- VO(Value Object) 클래스는 레이어와 레이어 사이에서 관련된 데이터를 한꺼번에 주고 받을 목적으로 사용하는 클래스
- DTO(Data Transfer Object)라고도 함

# 게시판의 테이블 구조(board.sql)

board.sql

```
=====

CREATE TABLE BOARD(
    SEQ NUMBER(5) PRIMARY KEY,
    TITLE VARCHAR2(200),
    WRITER VARCHAR2(20),
    CONTENT VARCHAR2(2000),
    REGDATE DATE DEFAULT SYSDATE,
    CNT NUMBER(5) DEFAULT 0
);
```

# VO - BoardVO.java

```
BoardVO.java ✘
1 package kr.ac.inje.comsi.board;
2
3 import java.sql.Date;
4
5 // VO(Value Object)
6 public class BoardVO {
7
8     private int seq;
9     private String title;
10    private String writer;
11    private String content;
12    private Date regDate;
13    private int cnt;
14
15    public int getSeq() {
16        return seq;
17    }
18    public void setSeq(int seq) {
19        this.seq = seq;
20    }
21    public String getTitle() {
22        return title;
23    }
24    public void setTitle(String title) {
25        this.title = title;
26    }
27
28    public String getWriter() {
29        return writer;
30    }
31    public void setWriter(String writer) {
32        this.writer = writer;
33    }
34    public String getContent() {
35        return content;
36    }
37    public void setContent(String content) {
38        this.content = content;
39    }
40    public Date getRegDate() {
41        return regDate;
42    }
43    public void setRegDate(Date regDate) {
44        this.regDate = regDate;
45    }
46    public int getCnt() {
47        return cnt;
48    }
49    public void setCnt(int cnt) {
50        this.cnt = cnt;
51    }
52
53    @Override
54    public String toString() {
55        return "BoardVO [seq=" + seq + ", title=" + title + ", writer=" +
56            writer + ", content=" + content + ", regDate=" + regDate +
57            ", cnt=" + cnt + "]";
58    }
59 }
```

다음 페이지

# @Override toString() method

The screenshot shows an IDE interface with a code editor and a tool dialog.

**Code Editor:** The file `*BoardVO.java` is open, displaying Java code for a `BoardVO` class. The code includes fields for seq, title, write, content, regDate, and cnt, along with methods for getting and setting these values.

**Context Menu:** A context menu is open over the code editor, with the "Source" option selected. Other visible options include Undo Typing, Revert File, Save, Open Declaration, Open Type Hierarchy, Open Call Hierarchy, Show in Breadcrumb, Quick Outline, Quick Type Hierarchy, Open With, Show In, Cut, Copy, Copy Qualified Name, Paste, Quick Fix, Toggle Comment, Refactor, Local History, References, Declarations, Add to Snippets..., AspectJ Refactoring, Run As, Debug As, Profile As, Validate, GitHub, and Override/Implement Methods... (which is highlighted).

**Override/Implement Methods Dialog:** This dialog lists methods to override or implement. It shows the `Object` class with its methods: clone(), equals(Object), finalize(), hashCode(), and `toString()`. The `toString()` method has a checked checkbox. Buttons for "Select All" and "Deselect All" are at the top right. Below the list are fields for "Insertion point" (set to "After 'setCnt(int)'") and checkboxes for "Generate method comments" and "Generate Getters and Setters...". A note at the bottom says, "The format of the method stubs may be configured on the [Code Templates](#) preference page." Buttons for "OK" and "Cancel" are at the bottom right.

```
1 package kr.ac.inje.comsi.board;
2
3 import java.sql.Date;
4
5 // VO(Value Object)
6 public class BoardVO {
7     private int seq;
8     private String title;
9     private String write;
10    private String content;
11    private Date regDate;
12    private int cnt;
13
14    public int getSeq() {
15        return seq;
16    }
17    public void setSeq(int seq) {
18        this.seq = seq;
19    }
20    public String getTitle() {
21        return title;
22    }
23    public void setTitle(String title) {
24        this.title = title;
25    }
26    public String getWrite() {
27        return write;
28    }
29    public void setWrite(String write) {
30        this.write = write;
31    }
32    public String getContent() {
33        return content;
34    }
35    public void setContent(String content) {
36        this.content = content;
37    }
38    public Date getRegDate() {
39        return regDate;
40    }
```

## 6.3 DAO 클래스 작성

- DAO(Data Access Object)클래스는 데이터 베이스 연동을 담당하는 클래스
- DAO 클래스에서는 CRUD(Create, Read, Update, Delete) 기능의 메소드가 구현되어야 함

# 1. 드라이버 내려받기

- build.gradle 파일 수정
  - dependencies{}에 내부에 Driver 정보 추가 후 Refresh Gradle Project

The screenshot shows an IDE interface with two tabs: 'BoardVO.java' and '\*build.gradle'. The '\*build.gradle' tab is active, displaying the following code:

```
1 /*
2 * This build file was generated by the Gradle 'init' task.
3 *
4 * This generated file contains a sample Java project to get
5 * more details take a look at the Java Quickstart chapter
6 * user guide available at https://docs.gradle.org/3.3/userguide/
7 */
8
9 // Apply the java plugin to add support for Java
10 apply plugin: 'application'
11
12 mainClassName = System.getProperty("mainClass");
13
14 compileJava.options.encoding = 'UTF-8'
15
16 // In this section you declare where to find the dependencies of your project
17 repositories {
18     mavenCentral()
19 }
20
21 dependencies {
22     compile 'org.springframework:spring-context:4.3.8.RELEASE'
23     compile 'com.h2database:h2:1.4.195' ←
24 }
25
```

A red arrow points from the line 'compile 'com.h2database:h2:1.4.195'' to the corresponding entry in the 'Project and External Dependencies' tree view on the right. The tree view shows the following structure:

- Project and External Dependencies
  - spring-context-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\files-2.1\spring\spring-context\4.3.8.RELEASE\spring-context-4.3.8.RELEASE.jar
  - h2-1.4.195.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\com\h2database\h2\1.4.195\h2-1.4.195.jar
  - spring-aop-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\files-2.1\spring\spring-aop\4.3.8.RELEASE\spring-aop-4.3.8.RELEASE.jar
  - spring-beans-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\files-2.1\spring\spring-beans\4.3.8.RELEASE\spring-beans-4.3.8.RELEASE.jar
  - spring-core-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\files-2.1\spring\spring-core\4.3.8.RELEASE\spring-core-4.3.8.RELEASE.jar
  - spring-expression-4.3.8.RELEASE.jar - C:\Users\Kyungtae\gradle\caches\files-2.1\spring\spring-expression\4.3.8.RELEASE\spring-expression-4.3.8.RELEASE.jar
  - commons-logging-1.2.jar - C:\Users\Kyungtae\gradle\caches\modules-2\files-2.1\commons\commons-logging\1.2\commons-logging-1.2.jar
- gradle
- src
- build.gradle

## 2. JDBC Utility 클래스 - Connection 획득과 해체 작업

```
BoardVO.java JDBCUtil.java BoardDAO.java
1 package kr.ac.inje.comsi.common;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7
8 public class JDBCUtil {
9     // get connection
10    public static Connection getConnection(){
11        try{
12            Class.forName("org.h2.Driver");
13            return DriverManager.getConnection("jdbc:h2:tcp://localhost/~test", "sa", "");
14        }catch(Exception e){
15            e.printStackTrace();
16        }
17        return null;
18    }
19    // close connection
20    public static void close(PreparedStatement stmt, Connection conn){
21        if(stmt != null){
22            try{
23                if(!stmt.isClosed()) stmt.close();
24            }catch(Exception e){
25                e.printStackTrace();
26            }finally{
27                stmt = null;
28            }
29        }
30    }
31
32    if(conn != null){
33        try{
34            if(!conn.isClosed()) conn.close();
35        }catch(Exception e){
36            e.printStackTrace();
37        }finally{
38            conn = null;
39        }
40    }
41 }
```

```
42     // close connection
43     public static void close(ResultSet rs, PreparedStatement stmt, Connection conn){
44         if(rs != null){
45             try{
46                 if(!rs.isClosed()) rs.close();
47             }catch(Exception e){
48                 e.printStackTrace();
49             }finally{
50                 rs = null;
51             }
52         }
53
54         if(stmt != null){
55             try{
56                 if(!stmt.isClosed()) stmt.close();
57             }catch(Exception e){
58                 e.printStackTrace();
59             }finally{
60                 stmt = null;
61             }
62         }
63
64         if(conn != null){
65             try{
66                 if(!conn.isClosed()) conn.close();
67             }catch(Exception e){
68                 e.printStackTrace();
69             }finally{
70                 conn = null;
71             }
72         }
73     }
74 }
75 }
```

### 3. DAO 클래스 작성-BOARD테이블과 CRUD 기능 처리

The screenshot shows a Java code editor with two tabs at the top: 'JDBCUtil.java' and 'BoardDAO.java'. The 'BoardDAO.java' tab is active, indicating the current file being viewed.

```
1 package kr.ac.inje.comsi.board.impl;
2
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.util.ArrayList;
7 import java.util.List;
8
9 import org.springframework.stereotype.Repository;
10
11 import kr.ac.inje.comsi.board.BoardVO;
12 import kr.ac.inje.comsi.common.JDBCUtil;
13
14 // DAO(Data Access Object)
15 @Repository("boardDAO")
16 public class BoardDAO {
17     // JDBC 관련 변수
18     private Connection conn = null;
19     private PreparedStatement stmt = null;
20     private ResultSet rset = null;
21
22     // SQL 명령어
23     private final String BOARD_INSERT = "insert into board(seq, title, writer, "
24         + "content) values((select nvl(max(seq),0)+1 from board),?, ?, ?)";
25     private final String BOARD_UPDATE = "update board set title=?, content=? where seq=?";
26     private final String BOARD_DELETE = "delete board where seq=?";
27     private final String BOARD_GET = "select * from board where seq=?";
28     private final String BOARD_LIST = "select * from board order by seq desc";
29 }
```

```
30 // CRUD 기능의 메소드 구현
31 // 글 등록
32⊕ public void insertBoard(BoardVO vo){
33     System.out.println("==> JDBC로 insertBoard() 기능 처리");
34     try{
35         conn = JDBCUtil.getConnection();
36         stmt = conn.prepareStatement(BOARD_INSERT);
37         stmt.setString(1, vo.getTitle());
38         stmt.setString(2, vo.getWriter());
39         stmt.setString(3, vo.getContent());
40         stmt.executeUpdate();
41     }catch(Exception e){
42         e.printStackTrace();
43     }finally{
44         JDBCUtil.close(stmt, conn);
45     }
46 }
47
48 // 글 수정
49⊕ public void updateBoard(BoardVO vo){
50     System.out.println("==> JDBC로 updateBoard() 기능 처리");
51     try{
52         conn = JDBCUtil.getConnection();
53         stmt = conn.prepareStatement(BOARD_UPDATE);
54         stmt.setString(1, vo.getTitle());
55         stmt.setString(2, vo.getContent());
56         stmt.setInt(3, vo.getSeq());
57         stmt.executeUpdate();
58     }catch(Exception e){
59         e.printStackTrace();
60     }finally{
61         JDBCUtil.close(stmt, conn);
62     }
63 }
64 }
```

```
64 // 글 삭제
65 public void deleteBoard(BoardVO vo){
66     System.out.println("==> JDBC로 deleteBoard() 기능 처리");
67     try{
68         conn = JDBCUtil.getConnection();
69         stmt = conn.prepareStatement(BOARD_DELETE);
70         stmt.setInt(1, vo.getSeq());
71         stmt.executeUpdate();
72     }catch(Exception e){
73     e.printStackTrace();
74 }finally{
75     JDBCUtil.close(stmt, conn);
76 }
77 }
78 }

80 // 글 상세 조회
81 public void getBoard(BoardVO vo){
82     System.out.println("==> JDBC로 getBoard() 기능 처리");
83     BoardVO board = null;
84     try{
85         conn = JDBCUtil.getConnection();
86         stmt = conn.prepareStatement(BOARD_GET);
87         stmt.setInt(1, vo.getSeq());
88         rset = stmt.executeQuery();
89         if(rset.next()){
90             board = new BoardVO();
91             board.setSeq(rset.getInt("SEQ"));
92             board.setTitle(rset.getString("TITLE"));
93             board.setWriter(rset.getString("WRITER"));
94             board.setContent(rset.getString("CONTENT"));
95             board.setRegDate(rset.getDate("REGDATE"));
96             board.setCnt(rset.getInt("CNT"));
97         }
98     }catch(Exception e){
99     e.printStackTrace();
100 }finally{
101     JDBCUtil.close(rset, stmt, conn);
102 }
103 }
104 }
```

```
105 // 글 목록 조회
106 public List<BoardVO> getBoardList(BoardVO vo){
107     System.out.println("==> JDBC로 getBoardList() 기능 처리");
108     List<BoardVO> boardList = new ArrayList<BoardVO>();
109     try{
110         conn = JDBCUtil.getConnection();
111         stmt = conn.prepareStatement(BOARD_LIST);
112         rset = stmt.executeQuery();
113         while(rset.next()){
114             BoardVO board = new BoardVO();
115             board.setSeq(rset.getInt("SEQ"));
116             board.setTitle(rset.getString("TITLE"));
117             board.setWriter(rset.getString("WRITER"));
118             board.setContent(rset.getString("CONTENT"));
119             board.setRegDate(rset.getDate("REGDATE"));
120             board.setCnt(rset.getInt("CNT"));
121             boardList.add(board);
122         }
123     }catch(Exception e){
124         e.printStackTrace();
125     }finally{
126         JDBCUtil.close(rset, stmt, conn);
127     }
128     return boardList;
129 }
130 }
131 }
```

# BoardDAO 클래스

- 스프링 컨테이너가 생성할 수 있도록 @Repository 어노테이션 설정
  - @Component 를 사용해도 문제가 발생하지 않음
- DAO 기능의 클래스에는 @Component보다 @Repository가 적합
- CRUD 기능의 메소드 이름-일관성 유지

기능	메소드 이름
등록	insert테이블명
수정	update테이블명
삭제	delete테이블명
상세 조회	get테이블명 혹은 select테이블명
목록 검색	get테이블명List 또는 select테이블명List

## 6.4 Service 인터페이스 작성

DAO 클래스에서 BoardService 인터페이스를 작성

# BoardService 인터페이스 작성

- BoardDAO 소스에 생성

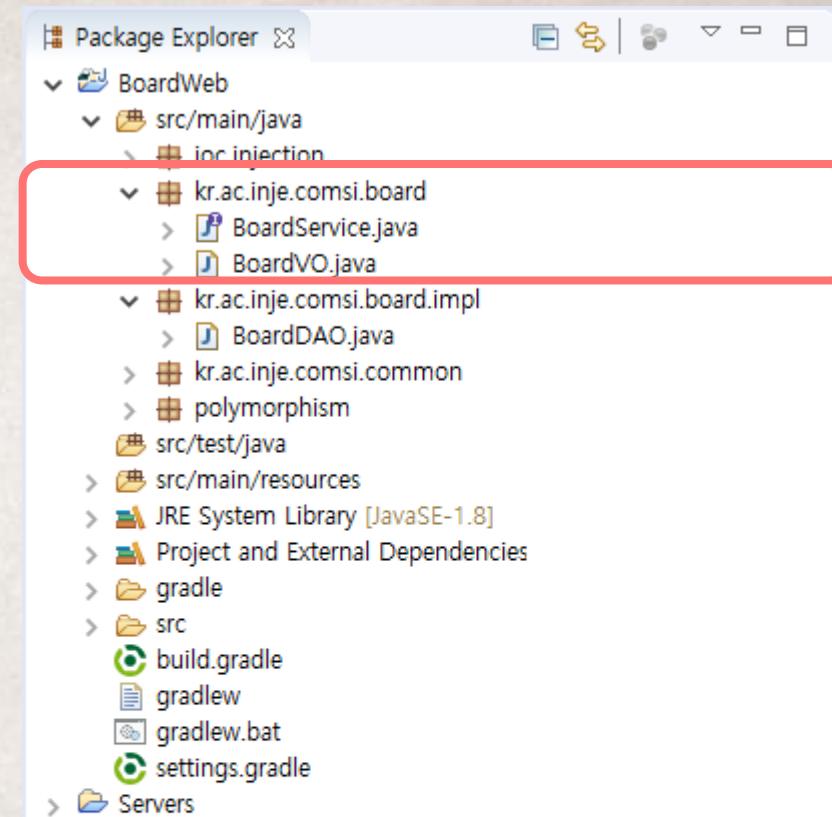
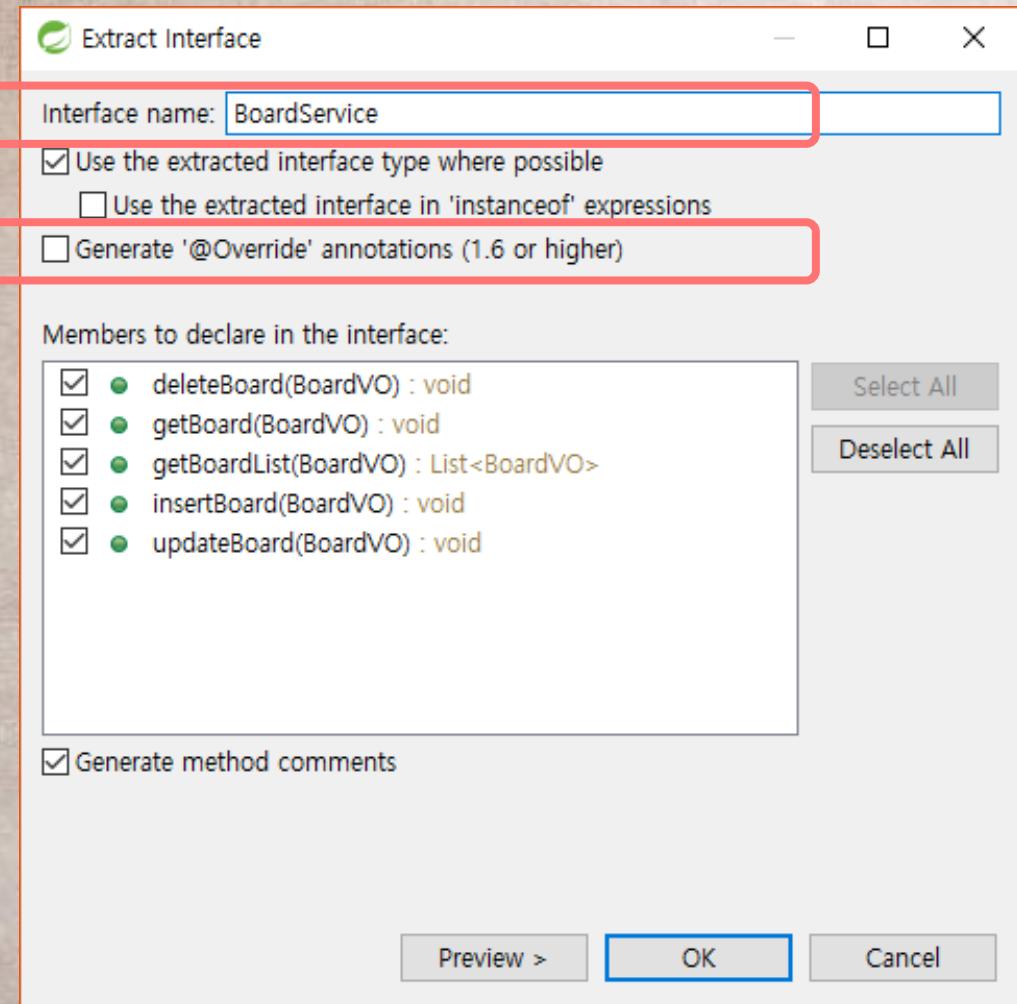
The screenshot shows a code editor with Java code for a BoardDAO class. A context menu is open over the code, specifically over the 'try' block. The menu path 'Refactor > Extract Interface...' is highlighted with a blue selection bar.

```
100     }finally{
101         JDBCUtil.close(rset,
102     }
103 }
104
105 // 글 목록 조회
106 public List<BoardVO> getBoar
107     System.out.println("==>">
108     List<BoardVO> boardList =
109     try{
110         conn = JDBCUtil.getConn();
111         stmt = conn.prepareStatement(
112         rset = stmt.executeQuery();
113         while(rset.next()){
114             BoardVO board =
115             board.setSeq(rse
116             board.setTitle(r
117             board.setWriter(r
118             board.setContent(r
119             board.setRegDate(r
120             board.setCnt(rse
121             boardList.add(board
122         }
123     }catch(Exception e){
124         e.printStackTrace();
125     }finally{
126         JDBCUtil.close(rset,
127     }
128     return boardList;
129 }
130 }
```

Context menu (Refactor submenu expanded):

- Undo (Ctrl+Z)
- Revert File
- Save (Ctrl+S)
- Open Declaration (F3)
- Open Type Hierarchy (F4)
- Open Call Hierarchy (Ctrl+Alt+H)
- Show in Breadcrumb (Alt+Shift+B)
- Quick Outline (Ctrl+O)
- Quick Type Hierarchy (Ctrl+T)
- Open With >
- Show In (Alt+Shift+W) >
- Cut (Ctrl+X)
- Copy (Ctrl+C)
- Copy Qualified Name
- Paste (Ctrl+V)
- Quick Fix (Ctrl+1)
- Source (Alt+Shift+S) >
- Refactor (Alt+Shift+T) >
- Local History >
- References >
- Declarations >
- Add to Snippets... (Alt+Shift+S) >
- AspectJ Refactoring >
- Run As >
- Move... (Alt+Shift+V)
- Change Method Signature... (Alt+Shift+C)
- Extract Interface... (highlighted)
- Extract Superclass...
- Use Supertype Where Possible...
- Pull Up...
- Push Down...

# BoardService 인터페이스 생성



kr.ac.inje.comsi.board 패키지로 이동

# BoardService 인터페이스 생성

```
JDBCUtil.java BoardDAO.java BoardService.java ✘  
1 package kr.ac.inje.comsi.board;  
2  
3 import java.util.List;  
4  
5 public interface BoardService {  
6  
7     // CRUD 기능의 메소드 구현  
8     // 글 등록  
9     void insertBoard(BoardVO vo);  
10  
11    // 글 수정  
12    void updateBoard(BoardVO vo);  
13  
14    // 글 삭제  
15    void deleteBoard(BoardVO vo);  
16  
17    // 글 상세 조회  
18    void getBoard(BoardVO vo);  
19  
20    // 글 목록 조회  
21    List<BoardVO> getBoardList(BoardVO vo);  
22  
23 }
```

## BoardDAO.java 소스 수정

```
14 // DAO(Data Access Object)  
15 @Repository("boardDAO")  
16 public class BoardDAO implements BoardService {  
17     // JDBC 관련 변수  
18     private Connection conn = null;  
19     private PreparedStatement stmt = null;  
20     private ResultSet rset = null;  
21
```

implements BoardService 삭제

## 6.5 Service 구현 클래스 작성

- BoardService 인터페이스를 구현한 BoardServiceImpl 클래스를 구현하면 비즈니스 컴포넌트는 마무리 됨
- BoardServiceImpl 클래스의 비즈니스 메소드를 구현할 때, 멤버변수로 선언된 BoardDAO를 이용함.

# BoardServiceImpl.java 클래스 생성

The screenshot shows a Java code editor with two tabs: 'BoardService.java' and 'BoardServiceImpl.java'. The 'BoardServiceImpl.java' tab is active, displaying the following code:

```
1 package kr.ac.inje.comsi.board.impl;
2
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Service;
7
8 import kr.ac.inje.comsi.board.BoardService;
9 import kr.ac.inje.comsi.board.BoardVO;
10
11 @Service("boardService")
12 public class BoardServiceImpl implements BoardService {
13
14     @Autowired
15     private BoardDAO boardDAO;
16
17     @Override
18     public void insertBoard(BoardVO vo) {
19         boardDAO.insertBoard(vo);
20     }
21
22     @Override
23     public void updateBoard(BoardVO vo) {
24         boardDAO.updateBoard(vo);
25     }
26
27     @Override
28     public void deleteBoard(BoardVO vo) {
29         boardDAO.deleteBoard(vo);
30     }
}
```

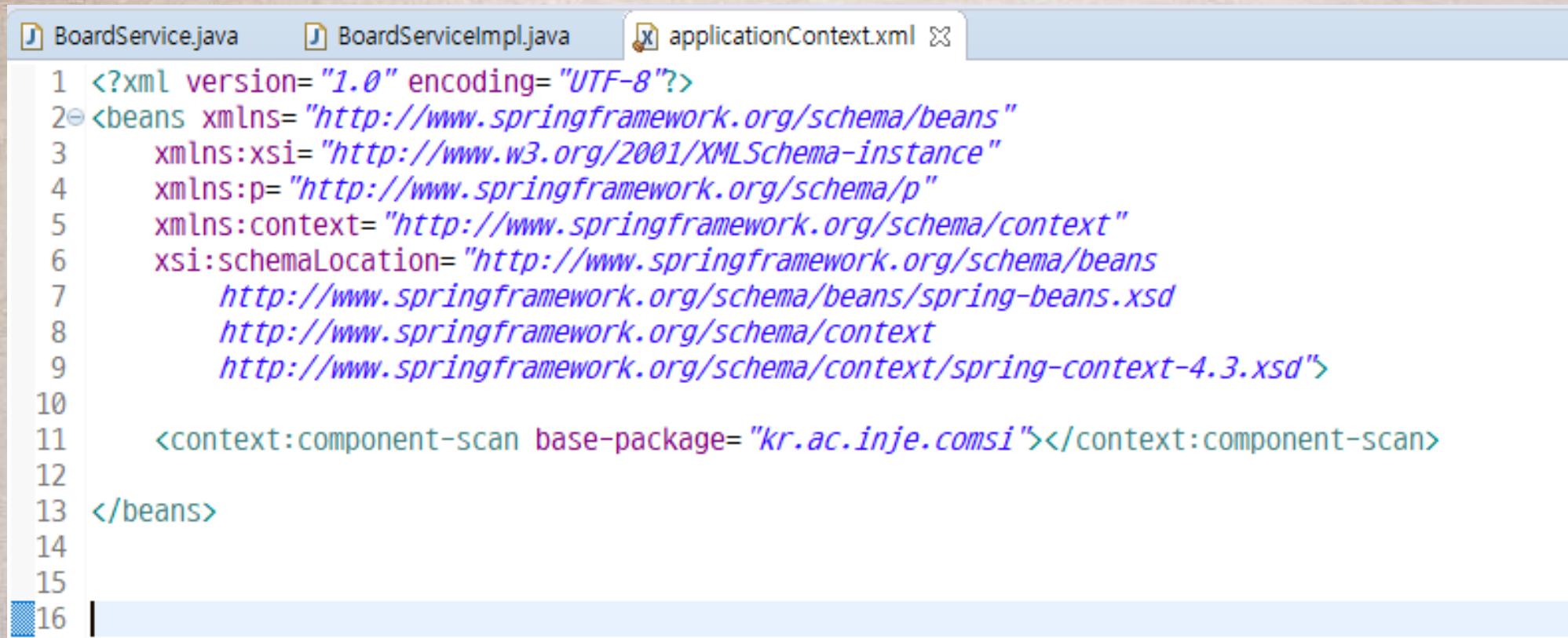
```
31  
32 @Override  
33 public void getBoard(BoardVO vo) {  
34     boardDAO.getBoard(vo);  
35 }  
36  
37 @Override  
38 public List<BoardVO> getBoardList(BoardVO vo) {  
39     return boardDAO.getBoardList(vo);  
40 }  
41  
42 }  
43
```

- BoardService 인터페이스의 모든 추상메소드를 Overriding 구현
- 클래스 선언부에 객체 생성을 위한 @Service가 선언
- 클라이언트 프로그램에서 boardService라는 이름으로 객체 요청
- BoardServiceImpl은 데이터베이스 연동을 포함된 비즈니스 로직 처리를 위해 BoardDAO 타입 객체를 멤버변수를 가짐
- BoardDAO 타입의 객체를 의존성 주입을 위해 @Autowired 설정

## 6.6 BoardService 컴포넌트 테스트

-

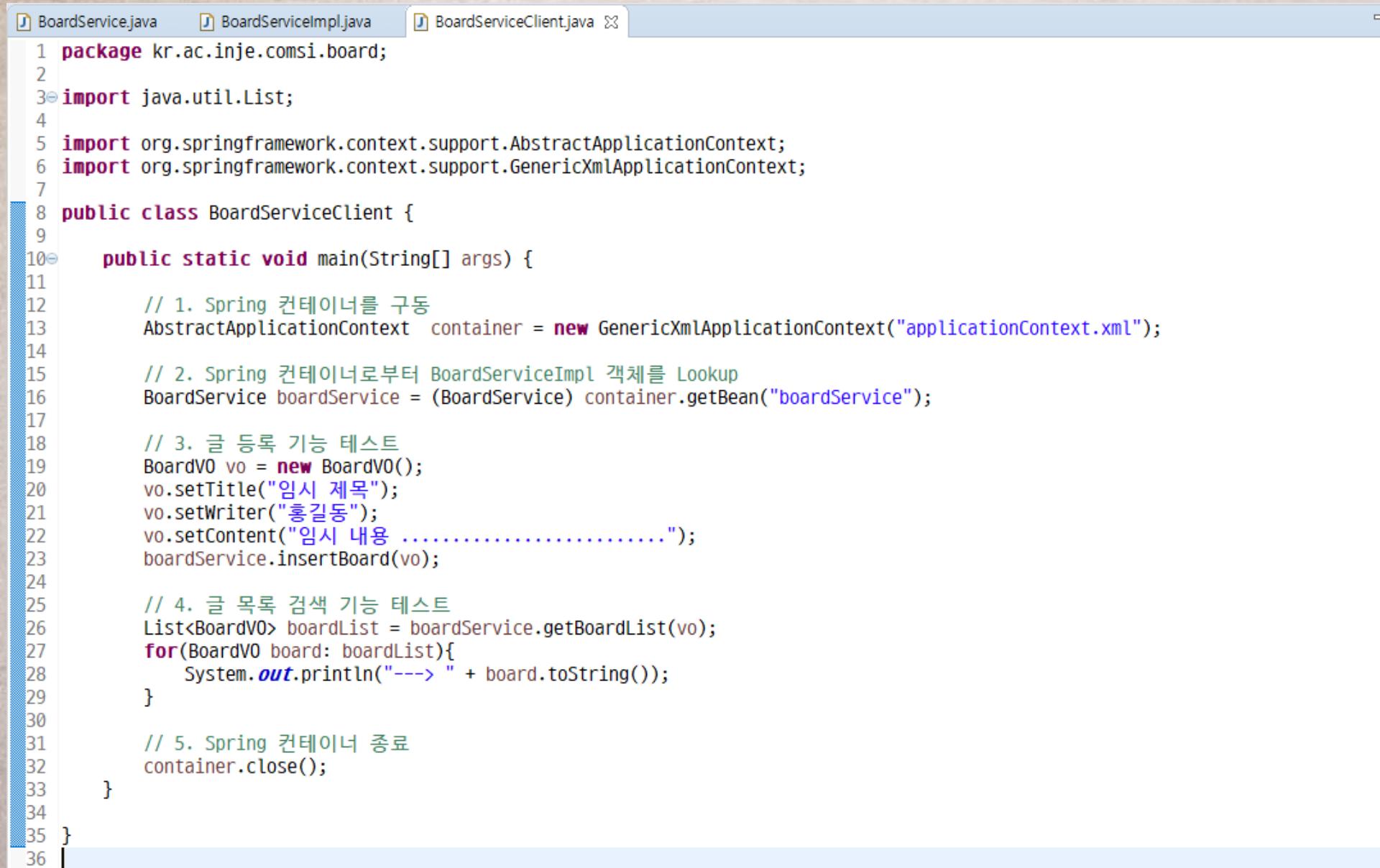
# 1. 스프링 설정 파일 수정



```
BoardService.java BoardServiceImpl.java applicationContext.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2<beans xmlns="http://www.springframework.org/schema/beans"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:p="http://www.springframework.org/schema/p"
5   xmlns:context="http://www.springframework.org/schema/context"
6   xsi:schemaLocation="http://www.springframework.org/schema/beans
7       http://www.springframework.org/schema/beans/spring-beans.xsd
8       http://www.springframework.org/schema/context
9       http://www.springframework.org/schema/context/spring-context-4.3.xsd">
10
11 <context:component-scan base-package="kr.ac.inje.comsi"></context:component-scan>
12
13 </beans>
14
15
16 |
```

- 컴포넌트 스캔 범위를 ‘kr.ac.inje.comsi’로 지정
- BoardServiceImpl 클래스와 boardDAO 클래스가 스캔 범위에 포함되어 의존성 처리가 될 것임.

## 2. 클라이언트 작성 및 실행

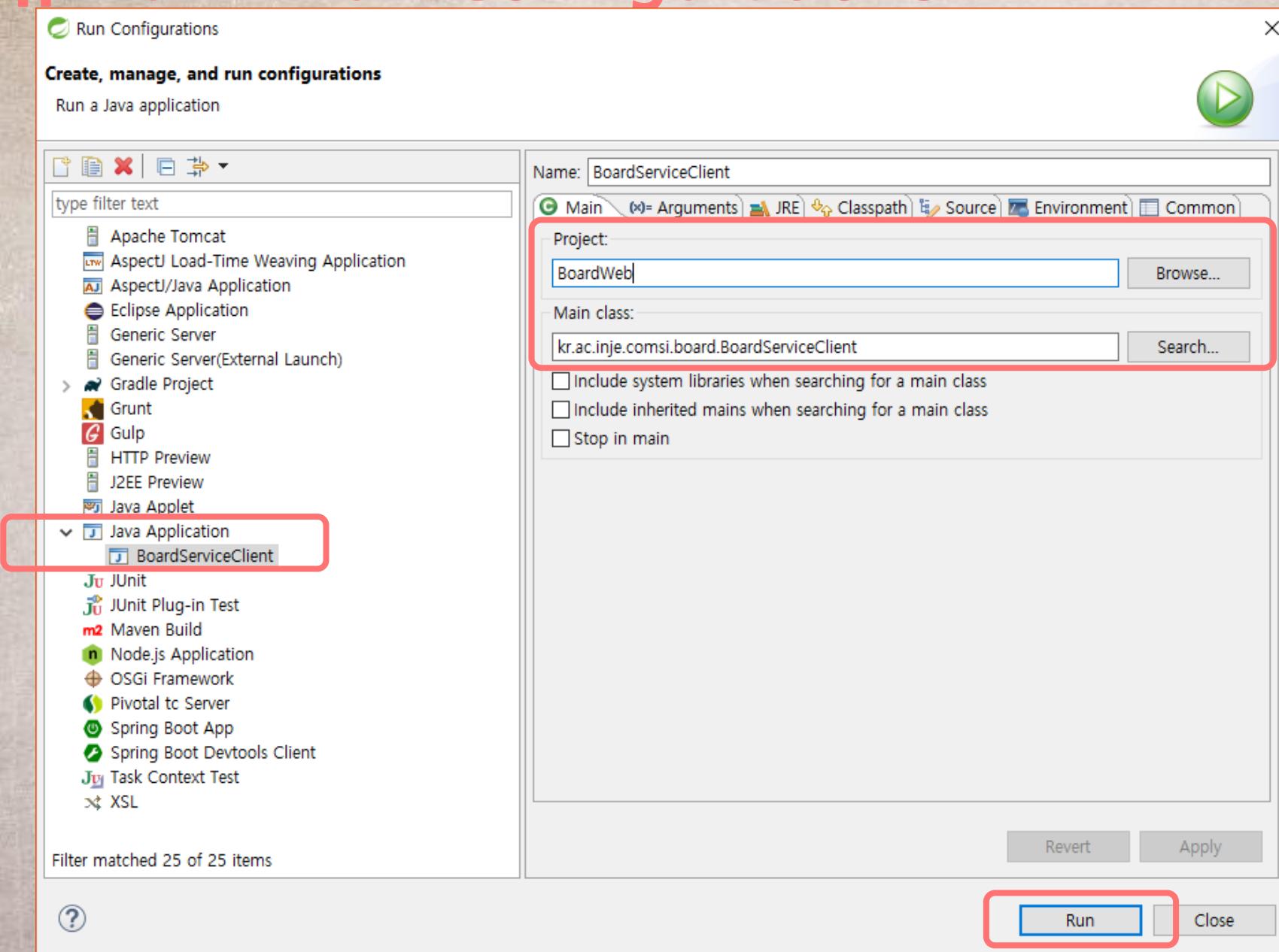


The screenshot shows a Java code editor with three tabs at the top: 'BoardService.java', 'BoardServiceImpl.java', and 'BoardServiceClient.java'. The 'BoardServiceClient.java' tab is active, displaying the following Java code:

```
1 package kr.ac.inje.comsi.board;
2
3 import java.util.List;
4
5 import org.springframework.context.support.AbstractApplicationContext;
6 import org.springframework.context.support.GenericXmlApplicationContext;
7
8 public class BoardServiceClient {
9
10    public static void main(String[] args) {
11
12        // 1. Spring 컨테이너를 구동
13        AbstractApplicationContext container = new GenericXmlApplicationContext("applicationContext.xml");
14
15        // 2. Spring 컨테이너로부터 BoardServiceImpl 객체를 Lookup
16        BoardService boardService = (BoardService) container.getBean("boardService");
17
18        // 3. 글 등록 기능 테스트
19        BoardVO vo = new BoardVO();
20        vo.setTitle("임시 제목");
21        vo.setWriter("홍길동");
22        vo.setContent("임시 내용 .....");
23        boardService.insertBoard(vo);
24
25        // 4. 글 목록 검색 기능 테스트
26        List<BoardVO> boardList = boardService.getBoardList(vo);
27        for(BoardVO board: boardList){
28            System.out.println("---> " + board.toString());
29        }
30
31        // 5. Spring 컨테이너 종료
32        container.close();
33    }
34
35}
36
```

실행

# 메뉴 Run → Run Configurations...



# 실행 결과

```
Console X Progress Problems
<terminated> BoardServiceClient [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 5. 24. 오전 10:51:14)
5월 24, 2017 10:51:15 오전 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
정보: Loading XML bean definitions from class path resource [applicationContext.xml]
5월 24, 2017 10:51:15 오전 org.springframework.context.support.GenericXmlApplicationContext prepareRefresh
정보: Refreshing org.springframework.context.support.GenericXmlApplicationContext@6e2c634b: startup date [Wed May 24 10:51:15
====> JDBC로 insertBoard() 기능 처리
====> JDBC로 getBoardList() 기능 처리
---> BoardVO [seq=2, title=임시 제목, writer=홍길동, content=임시 내용 ....., regDate=2017-05-24, cnt=0]
---> BoardVO [seq=1, title=가입인사, writer=관리자, content=잘 부탁드립니다....., regDate=2017-04-26, cnt=0]
5월 24, 2017 10:51:15 오전 org.springframework.context.support.GenericXmlApplicationContext doClose
정보: Closing org.springframework.context.support.GenericXmlApplicationContext@6e2c634b: startup date [Wed May 24 10:51:15 KST]
```

# 실행 결과 - H2Database

The screenshot shows the H2 Database browser interface. The left sidebar displays the database schema with tables: BOARD, SEQ, TITLE, WRITER, CONTENT, REGDATE, CNT, and INFORMATION\_SCHEMA. The main area shows the results of a SQL query: "SELECT \* FROM BOARD;". The results are displayed in a table:

SEQ	TITLE	WRITER	CONTENT	REGDATE	CNT
1	가입인사	관리자	잘 부탁드립니다.....	2017-04-26	0
2	임시 제목	홍길동	임시 내용 .....	2017-05-24	0

(2 행, 3 ms)

# 데이터베이스 구축

# H2-<http://h2database.com>

The screenshot shows a web browser window with the URL [h2database.com/html/main.html](http://h2database.com/html/main.html) in the address bar. The page content is as follows:

**H2 Database Engine**

Welcome to H2, the Java SQL database. The main features of H2 are:

- Very fast, open source, JDBC API
- Embedded and server modes; in-memory databases
- Browser based Console application
- Small footprint: around 1.5 MB jar file size

**Download**

Version 1.4.195 (2017-04-23)

[Windows Installer \(5 MB\)](#)  
[All Platforms \(zip, 8 MB\)](#) (This link is highlighted with a red box)

[All Downloads](#)

**Support**

[Stack Overflow \(tag H2\)](#)  
[Google Group English](#), [Japanese](#)

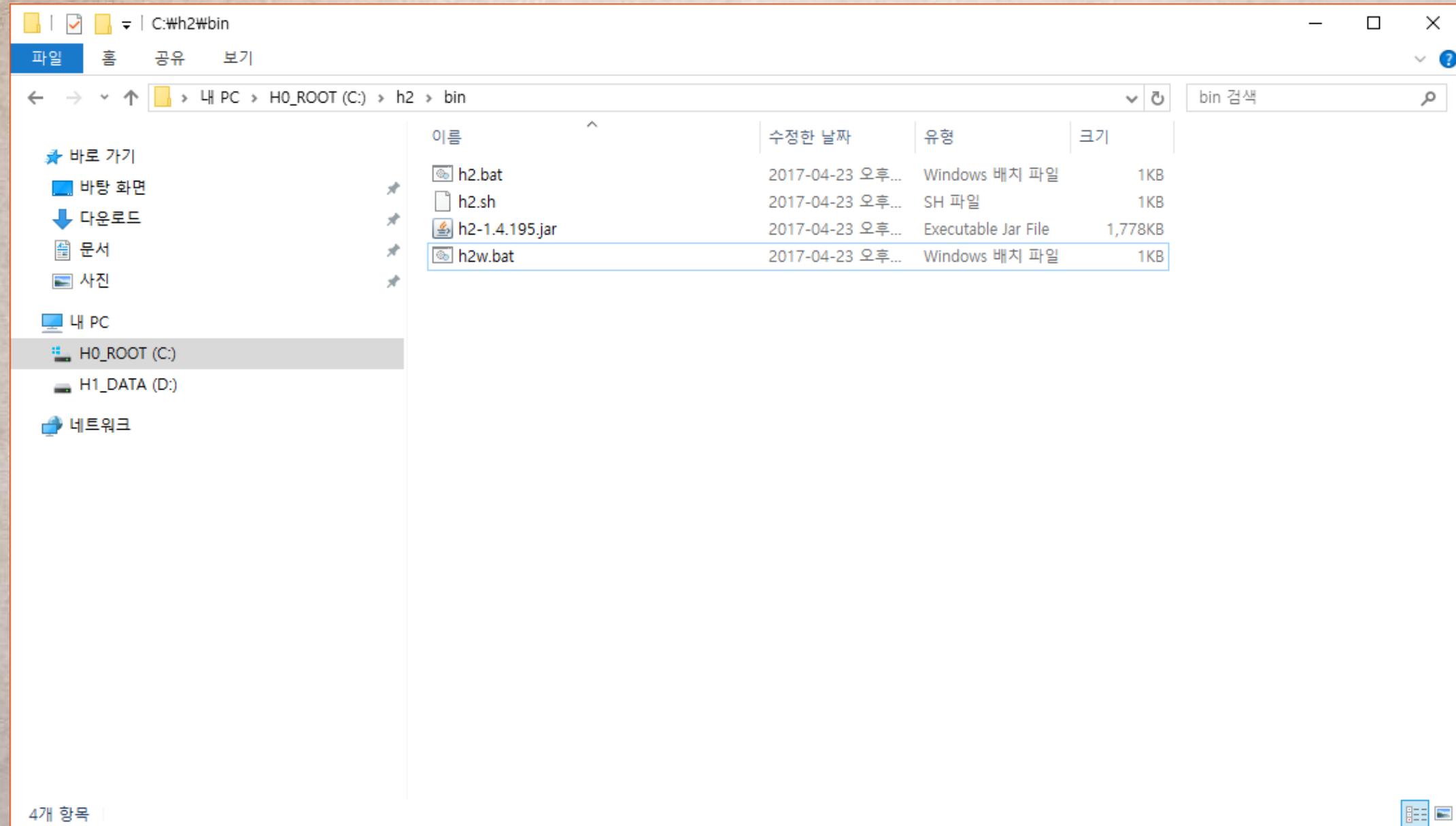
For non-technical issues, use:  
dbsupport at h2database.com

**Features**

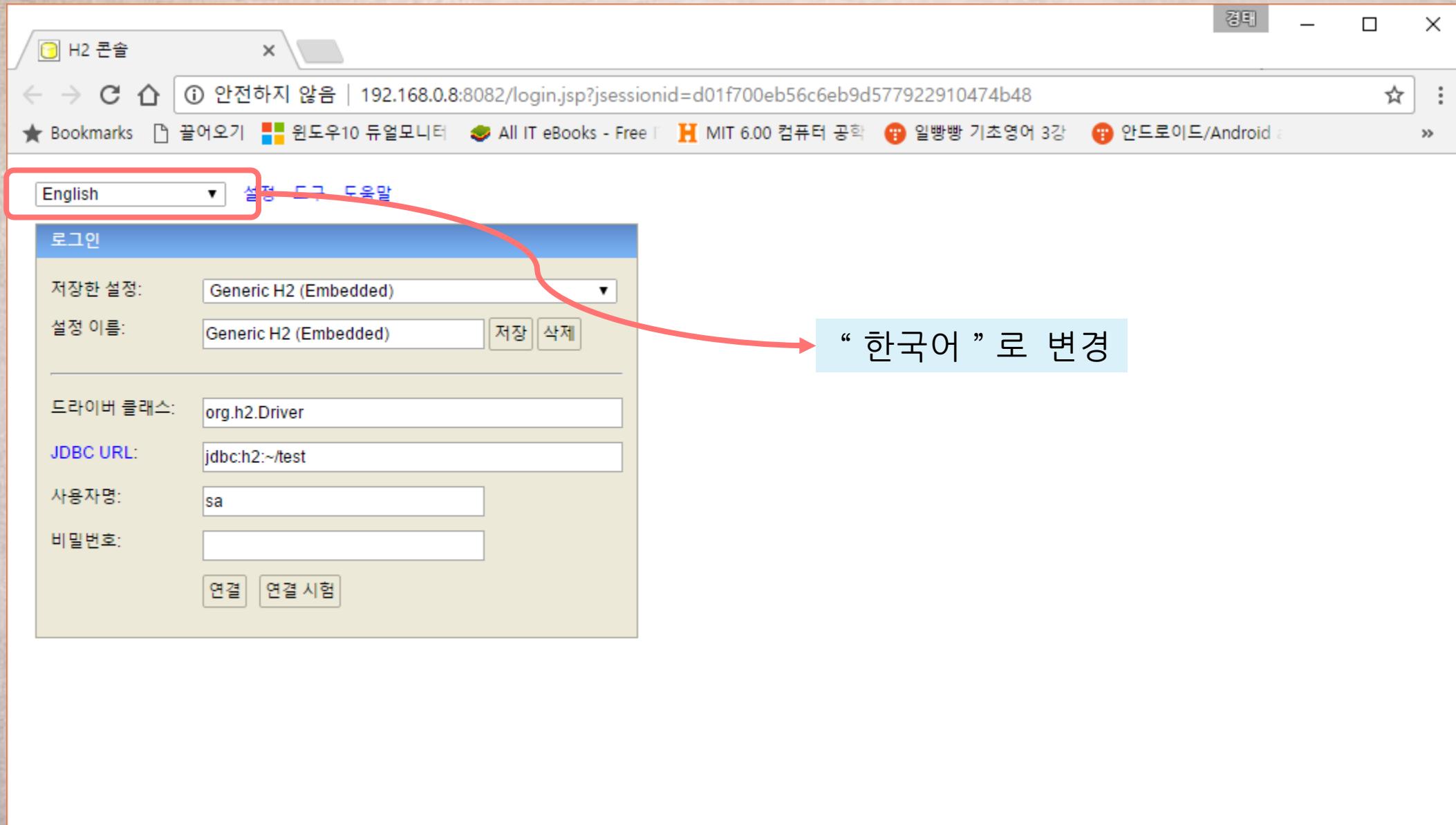
	H2	Derby	HSQldb	MySQL	PostgreSQL
Pure Java	Yes	Yes	Yes	No	No
Memory Mode	Yes	Yes	Yes	No	No
Encrypted Database	Yes	Yes	Yes	No	No
ODBC Driver	Yes	No	No	Yes	Yes
Fulltext Search	Yes	No	No	Yes	Yes
Multi Version Concurrency	Yes	No	Yes	Yes	Yes
Footprint (jar/dll size)	~1 MB	~2 MB	~1 MB	~4 MB	~6 MB

See also the [detailed comparison](#).

# H2Database 실행 - h2→bin>h2w.bat 실행



# H2 서버 - 구동



# H2 서버 - 연결

The screenshot shows the H2 Console application window. The title bar reads "H2 콘솔". The address bar shows the URL "192.168.0.8:8082/login.do?jsessionid=d01f700eb56c6eb9d577922910474b48". The left sidebar displays the database connection details: "jdbc:h2:~/test", "INFORMATION\_SCHEMA", "사용자", and "H2 1.4.195 (2017-04-23)". The main area contains a toolbar with various icons, a dropdown for "최대 행 수: 1000", and a "SQL 문:" input field with tabs for "실행" (Run Selected), "자동 완성" (Auto Complete), and "지우기" (Clear). Below the SQL input is a section titled "중요 명령" (Important Commands) with a table:

①	이 도움말 페이지 보기
₩	명령 이력 보기
Ctrl+엔터	현재의 SQL 문 실행
Shift+엔터	Executes the SQL statement defined by the text selection
Ctrl+Space	자동 완성
데이터베이스 연결 끊기	데이터베이스 연결 끊기

Below this is a section titled "샘플 SQL 스크립트" (Sample SQL Scripts) with a table:

테이블이 존재하는 경우 삭제하기	DROP TABLE IF EXISTS TEST;
새 테이블 만들기	CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
컬럼을 ID와 NAME	INSERT INTO TEST VALUES(1, 'Hello');
행 추가	INSERT INTO TEST VALUES(2, 'World');
행 추가	SELECT * FROM TEST ORDER BY ID;
테이블 질의	UPDATE TEST SET NAME='Hi' WHERE ID=1;
행 데이터 변경	DELETE FROM TEST WHERE ID=2;
행 삭제	
도움말	HELP ...

At the bottom, there is a section titled "데이터베이스 드라이버 추가" (Add Database Driver) with the following text:

데이터베이스 드라이버를 추가로 등록하려면 H2DRIVERS나 CLASSPATH 환경 변수에 드라이버의 Jar 파일 위치를 추가하면 됩니다. 예 (Windows): 데이터베이스 드라이버 라이브러리로 C:/Programs/hsqldb/lib/hsqldb.jar를 추가하려면 H2DRIVERS 환경 변수를 C:/Programs/hsqldb/lib/hsqldb.jar로 설정합니다.

## H2 서버 - 테이블 생성 및 데이터 입력코드(sql문)

```
CREATE TABLE USERSC  
    ID VARCHAR2(8) PRIMARY KEY,  
    PASSWORD VARCHAR2(8),  
    NAME VARCHAR2(20),  
    ROLE VARCHAR2(5)  
);
```

USER 테이블

```
INSERT INTO USERS VALUES('test','test123','관리자','Admin');  
INSERT INTO USERS VALUES('user1','user1','홍길동','user');
```

```
CREATE TABLE BOARDG  
    SEQ NUMBER(5) PRIMARY KEY,  
    TITLE VARCHAR2(200),  
    WRITER VARCHAR2(20),  
    CONTENT VARCHAR2(2000),  
    REGDATE DATE DEFAULT SYSDATE,  
    CNT NUMBER(5) DEFAULT 0  
);
```

BOARD 테이블

```
INSERT INTO BOARD(SEQ, TITLE, WRITER, CONTENT) VALUES(1,'가입인사','관리자','잘 부탁드립니다.....');
```

# H2 서버 - 입력 데이터 확인

The screenshot shows the H2 Console interface running in a browser window. The URL is 192.168.0.8:8082/login.do?jsessionid=d01f700eb56c6eb9d577922910474b48. The left sidebar lists database schemas: BOARD, USERS, INFORMATION\_SCHEMA, and 사용자 (User). The main area contains two SQL queries and their results.

**SQL 문:**

```
select * from users;
select * from board;
```

**select \* from users;**

ID	PASSWORD	NAME	ROLE
test	test123	관리자	Admin
user1	user1	홍길동	user

(2 행, 0 ms)

**select \* from board;**

SEQ	TITLE	WRITER	CONTENT	REGDATE	CNT
1	가입인사	관리자	잘 부탁드립니다.....	2017-04-26	0

(1 row, 0 ms)

