







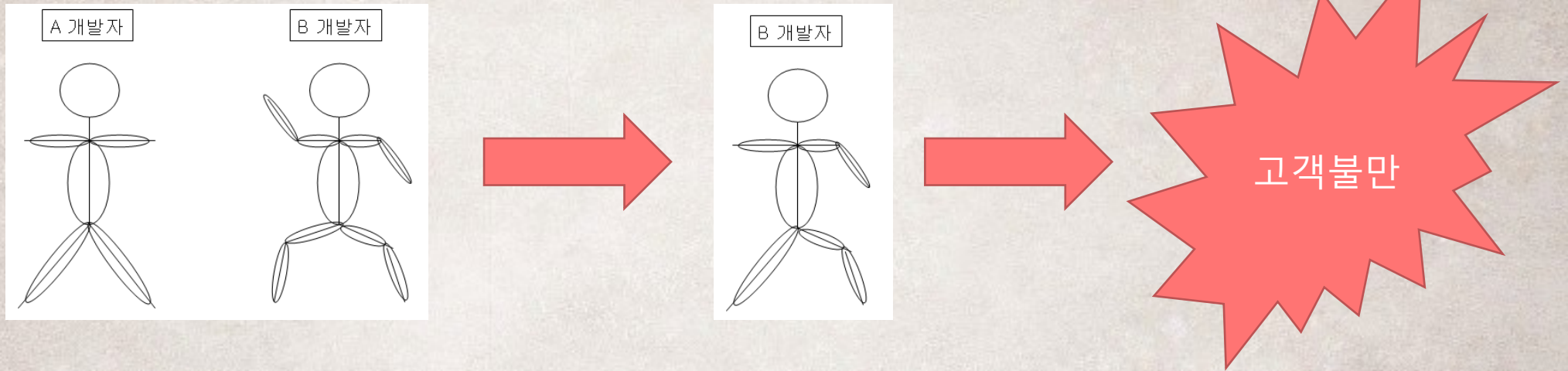
# 강의 내용

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 아키텍처 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>



# Framework 개념

- 두 명의 개발자가 모든 권한을 가지고 인형을 만들도록 지시 받아 만들었다.
- A개발자의 인형의 부러진 다리를 B개발자가 수리하면 ?



- 원인
  - 개발자들의 커뮤니케이션 부재
  - 동일한 골격을 사용하지 않았기 때문..
- 골격 = 프레임워크
  - 반제품



# Framework 구조

- 프레임워크

- 소프트웨어에 직면한 생산성 문제를 재사용을 통해 해결하려는 방편 중의 하나
- 반복되는 구조/기능을 추상화 시켜 놓음
- 완전한 소프트웨어는 아님
  - 완전한 애플리케이션이 되기 위해서는 프레임워크를 확장하여 비즈니스 니즈를 만족시키는 작업 필요
  - 확장이 프레임워크의 기본 구조에 영향을 끼치지 않음

- Cold Spot(콜드 스팟)

- 프레임워크 코어 : 변경되지 않고 반복적으로 재사용되는 부분

- Hook Point(확장 포인트)

- 추상 클래스나 인터페이스 형태로 제공

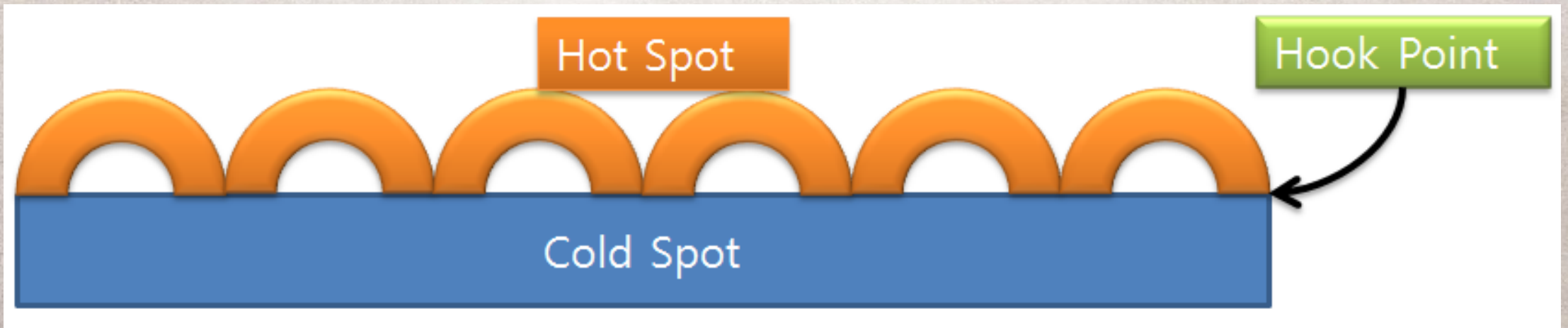
- Hot Spot(핫 스팟)

- 개발자에 의해 확장된 모듈



## • 프레임워크

- Open-Closed 원칙 (이하 OCP)을 그대로 따름.
  - 재사용되는 공통된 부분은 프레임워크로 구현되어 다른 사람이 그 내부를 가공 없이 이용하도록 제공(Closed). → Cold Spot (프레임워크 코어)
  - 확장이 필요한 부분은 사용자 요구사항에 맞게 정의하여 확장시키므로 문제영역에 최적화된 애플리케이션 설계가 완성(Open). → Hot Spot (확장 모듈)
- 이를 확장에는 열려 있고, 변경에는 닫혀 있다라고 말한다.
- 프레임워크 코어는 각 어플리케이션 마다 확장 모듈을 연결하는 확장 점을 제공하는 이클훅 포인트(Hook Point)라고 함





# Framework의 특징1

- 프레임워크의 장점
  - 빠른 구현 시간
    - 기반구조를 그대로 사용하므로 구현코드가 줄어들고 생산성이 향상
  - 관리의 용이성
    - 동일한 프레임워크가 적용된 애플리케이션들은 구조가 비슷하므로 관리가 용이해짐
  - 테스트
    - 기존에 사용된 테스트도 재사용 가능해짐
    - 새로 구축한 모듈에 대한 테스트만 수행해도 됨.
  - 안정성
    - 여러 번 사용된 프레임워크의 안정성은 새롭게 작성한 코드에 비해 안정성이 뛰어남



# Framework의 특징2-어플리케이션 처리 영역별

- 자바기반의 Framework

처리 영역	프레임워크	설명
Presentation	Struts	Struts 프레임워크는 UI Layer에 중점을 두고 개발된 MVC(Model-View-Controller)프레임워크다.  이미 전 세계적으로 가장 많이 사용되고 있는 프레임워크이며, 오픈 소스 사용을 꺼리고 있는 대부분의 국내 대형 SI 업체들도 Struts를 사용하고 있다.
	Spring (MVC)	Struts와 동일하게 MVC 아키텍처를 제공하는 UI Layer 프레임워크다. Spring 프레임워크를 구성하는 작은 모듈이다.



# Framework의 특징3-어플리케이션 처리 영역별

- 자바기반의 Framework

처리 영역	프레임워크	설명
Business	Spring (IoC, AOP)	<p>Spring 프레임워크는 컨테이너 성격을 가지는 프레임워크다. Spring은 아키텍처상에서 비즈니스 Layer에서부터 Persistence Layer, UI Layer 전체 적인 영역을 모두 포함하고 있다.</p> <p>물론 Spring이 비즈니스 Layer만을 전담하고, Persistence Layer, UI Layer는 다른 프레임워크를 사용하여 개발할 수도 있다.</p>



# Framework의 특징4-어플리케이션 처리 영역별

- 자바기반의 Framework

처리 영역	프레임워크	설명
Persistence	Hibernate	Hibernate 프레임워크는 Persistence Layer를 담당하는 ORM(Object Relation Mapping) 툴이다. 근래에 들어 ORM에 대한 관심도가 높아지면서 Hibernate 프레임워크에 대하여 많은 개발자들이 관심을 가지고 있다.
	iBatis	iBatis 프레임워크는 SQL Mapper로서 Hibernate와는 달리 완벽한 ORM 도구가 아니다. 개발자가 작성한 SQL 명령어와 Object를 매핑해 주는 기능을 제공하며, 기존의 SQL 명령어를 재사용하여 개발하는 애플리케이션에 유용하게 적용할 수 있다.

ORM(Object Rational Mapping): 자바 Object와 관계형 데이터베이스와의 맵핑.  
Query, Connection, transaction 처리 등을 Framework가 처리해 줌으로서 개발자로 하여금 business logic에 집중할 수 있도록 제공



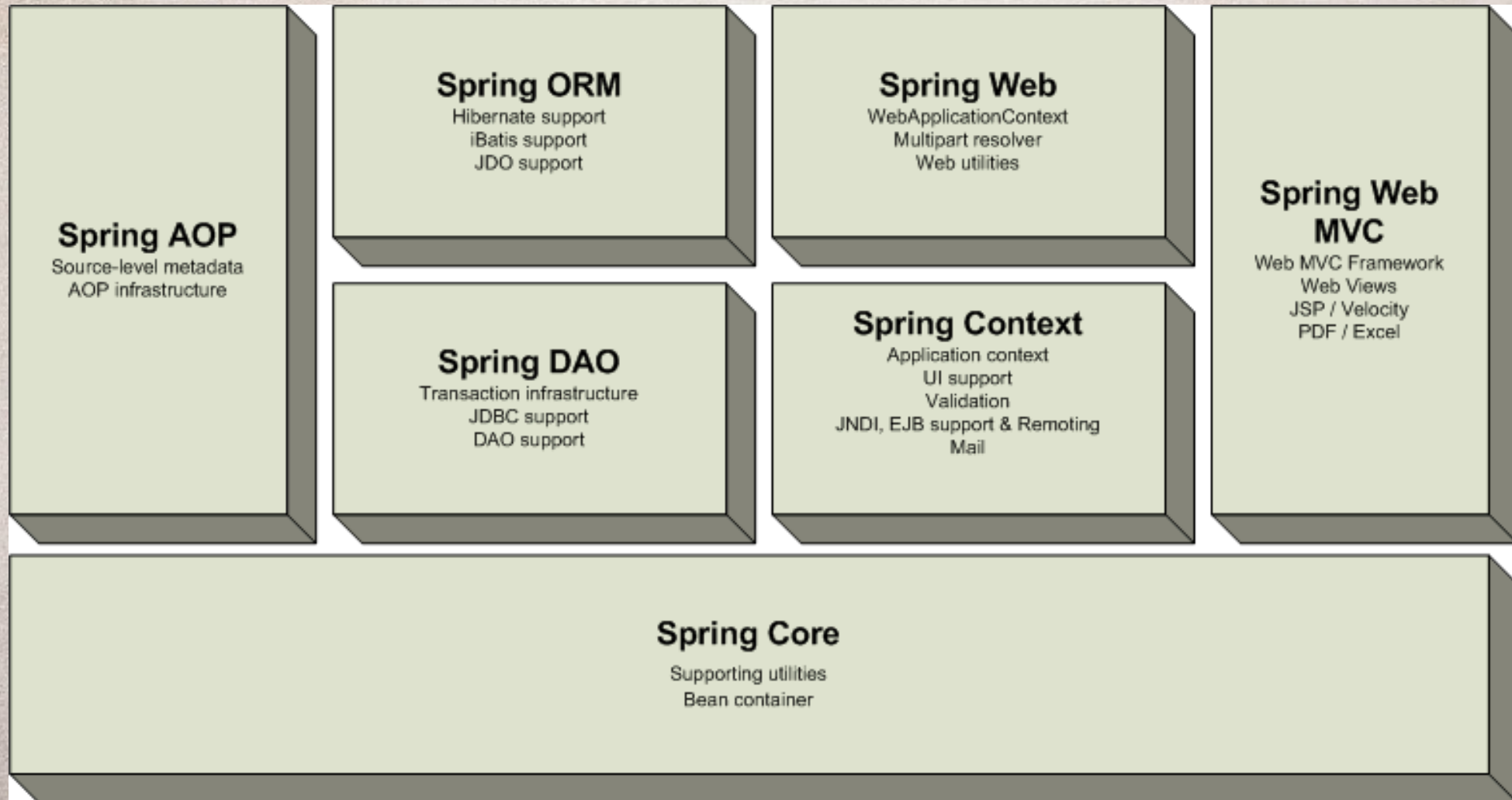
# Framework의 특징5

- 프레임워크 적용시 고려사항
  - 만들고자 하는 애플리케이션의 성격을 파악한다.
  - 개발하려는 애플리케이션의 아키텍처를 명확히 정의한다.
  - 프로젝트 구성원의 성향을 고려하여 프레임워크를 선택한다.
  - 프레임워크를 적절히 조합한다.



# Spring Framework1

- Spring Framework의 구조






# 데이터베이스 구축



# H2-http://h2database.com



[Translate](#)

Search:

[Home](#)

[Download](#)

[Cheat Sheet](#)

**Documentation**

[Quickstart](#)

[Installation](#)

[Tutorial](#)

[Features](#)

[Performance](#)

[Advanced](#)

**Reference**

[SQL Grammar](#)

[Functions](#)

[Data Types](#)

[Javadoc](#)

[PDF \(1 MB\)](#)

**Support**

[FAQ](#)

[Error Analyzer](#)

[Google Group \(English\)](#)

[Google Group \(Japanese\)](#)

[Google Group \(Chinese\)](#)


## 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 \(zip, 8 MB\)](#)

[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](#).



# H2 서버 - 구동

The image shows a web browser window titled "H2 콘솔" (H2 Console) at the URL `192.168.0.8:8082/login.jsp?jsessionid=d01f700eb56c6eb9d577922910474b48`. The browser's address bar and bookmarks are visible. In the top left of the console interface, there is a language dropdown menu currently set to "English". A red box highlights this dropdown, and a red arrow points from it to a light blue text box on the right that says "“한국어”로 변경" (Change to "Korean"). Below the language menu, the "로그인" (Login) form is visible, containing fields for "저장한 설정:" (Saved settings: Generic H2 (Embedded)), "설정 이름:" (Setting name: Generic H2 (Embedded)), "드라이버 클래스:" (Driver class: org.h2.Driver), "JDBC URL:" (jdbc:h2:~/test), "사용자명:" (Username: sa), and "비밀번호:" (Password:). There are buttons for "저장" (Save), "삭제" (Delete), "연결" (Connect), and "연결 시험" (Test connection).

English ▼ 설정 도구 도움말

로그인

저장한 설정: Generic H2 (Embedded) ▼

설정 이름: Generic H2 (Embedded) 저장 삭제

드라이버 클래스: org.h2.Driver

JDBC URL: jdbc:h2:~/test

사용자명: sa

비밀번호:

연결 연결 시험

“한국어”로 변경



# H2 서버 - 연결

H2 콘솔

192.168.0.8:8082/login.do?sessionId=d01f700eb56c6eb9d577922910474b48

Bookmarkspulsero기 Windows10 듀얼모니터 All IT eBooks - Free MIT 6.00 컴퓨터 공학 일행행 기초영어 3강 안드로이드/Android 나의 북마크 AppInventor 가져온 북마크 라즈베리파이

자동 커밋 최대 행 수: 1000 자동 완성 안함 Auto select On

jdbc:h2:~/test

INFORMATION\_SCHEMA

사용자

H2 1.4.195 (2017-04-23)

실행 Run Selected 자동 완성 지우기 SQL 문:

중요 명령

?	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

샘플 SQL 스크립트

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

데이터베이스 드라이버 추가

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 USERS(  
    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 BOARD(  
    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 web interface in a browser window. The address bar shows the URL `192.168.0.8:8082/login.do?jsessionid=d01f700eb56c6eb9d577922910474b48`. The left sidebar shows the database structure: `jdbc:h2:~/test`, `BOARD`, `USERS`, `INFORMATION_SCHEMA`, `사용자`, and `H2 1.4.195 (2017-04-23)`. The main area displays the SQL queries and their results.

실행 Run Selected 자동 완성 지우기 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)

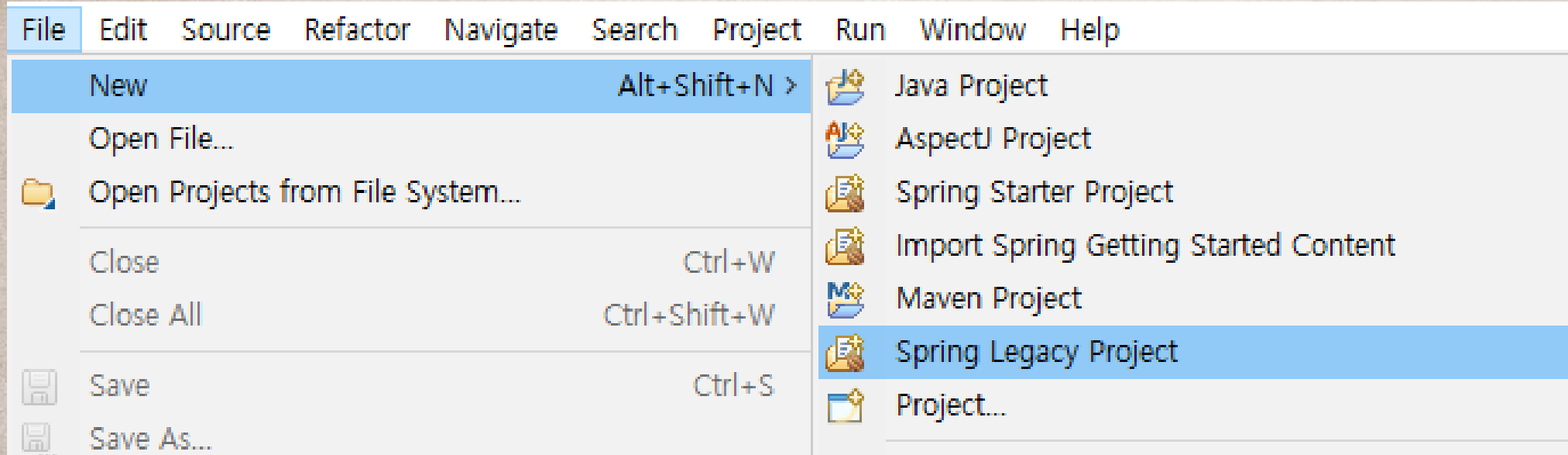


실습 프로젝트 생성



# 1. 프로젝트 생성

- [File]→[New] 메뉴로 이동한 후 [Spring Legacy Project] 선택





New Spring Legacy Project

**Spring Legacy Project**

Click 'Next' to load the template contents.

Project name: BoardWeb

☒ Use default location

Location: D:\workspace\java\2017\_1\BoardWeb [Browse...](#)

Select Spring version: Default

Templates:

- Simple Projects
  - Simple Java
  - Simple Spring Maven
  - Simple Spring Web Maven
- Batch
- GemFire
- Integration
- Persistence
- Simple Spring Utility Project
- Spring MVC Project

requires downloading [Configure templates...](#) [Refresh](#)

Description:  
A new Spring MVC web application development project

URL: <http://dist.springsource.com/release/STS/help/org.springframework.templates.mvc-3.2.2.zip>

Working sets

☐ Add project to working sets [New...](#)

Working sets: [Select...](#)

[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

BoardWeb

Import

Spring MVC Project requires a download of 16608 bytes.

Proceed?

[Yes](#) [No](#)



New Spring Legacy Project

Project Settings - Spring MVC Project

Define project specific settings. Required settings are denoted by "\*".

Please specify the top-level package e.g. com.mycompany.myapp\*

kr.ac.inje.comsi.ktpark

?

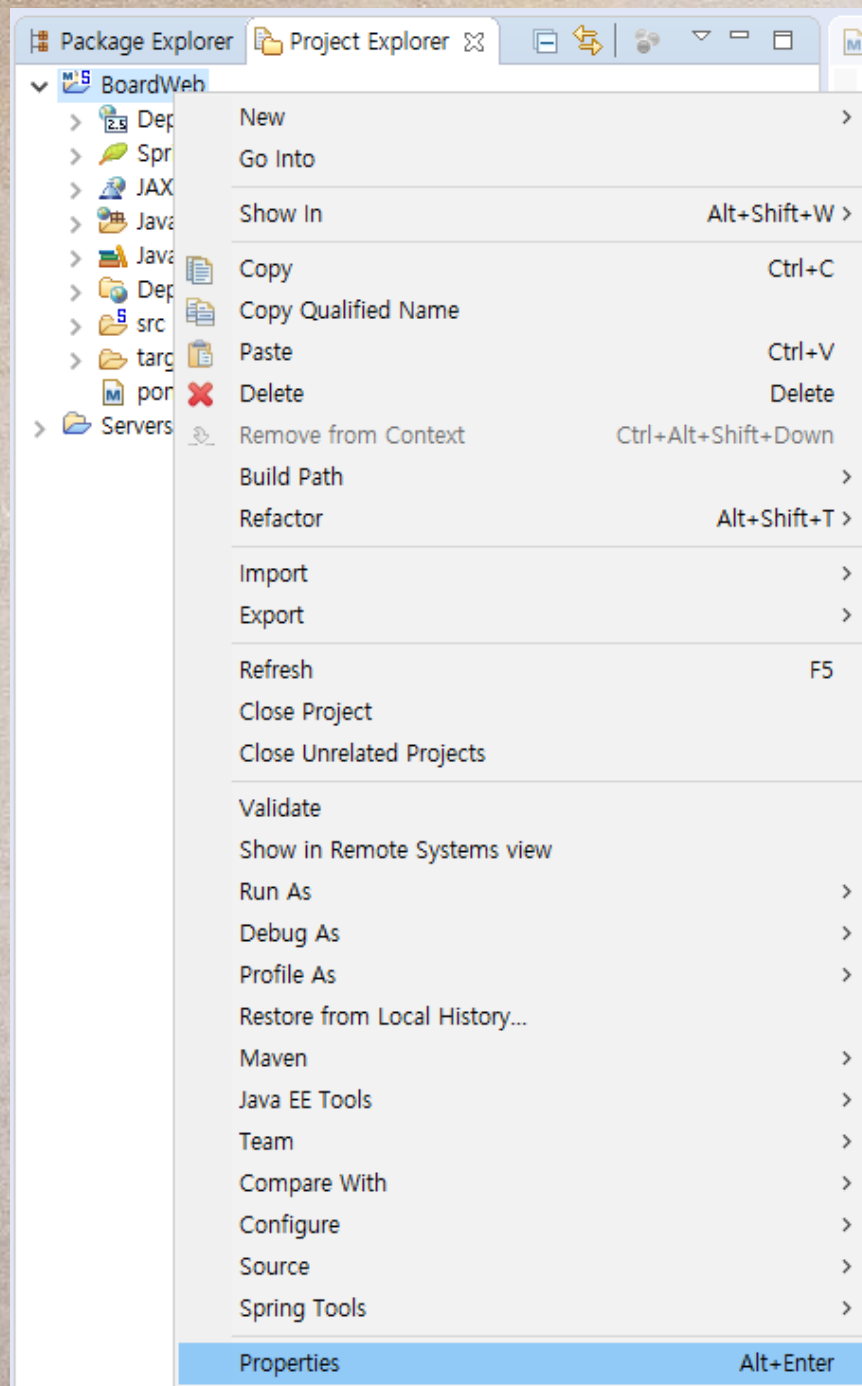
< Back

Next >

Finish

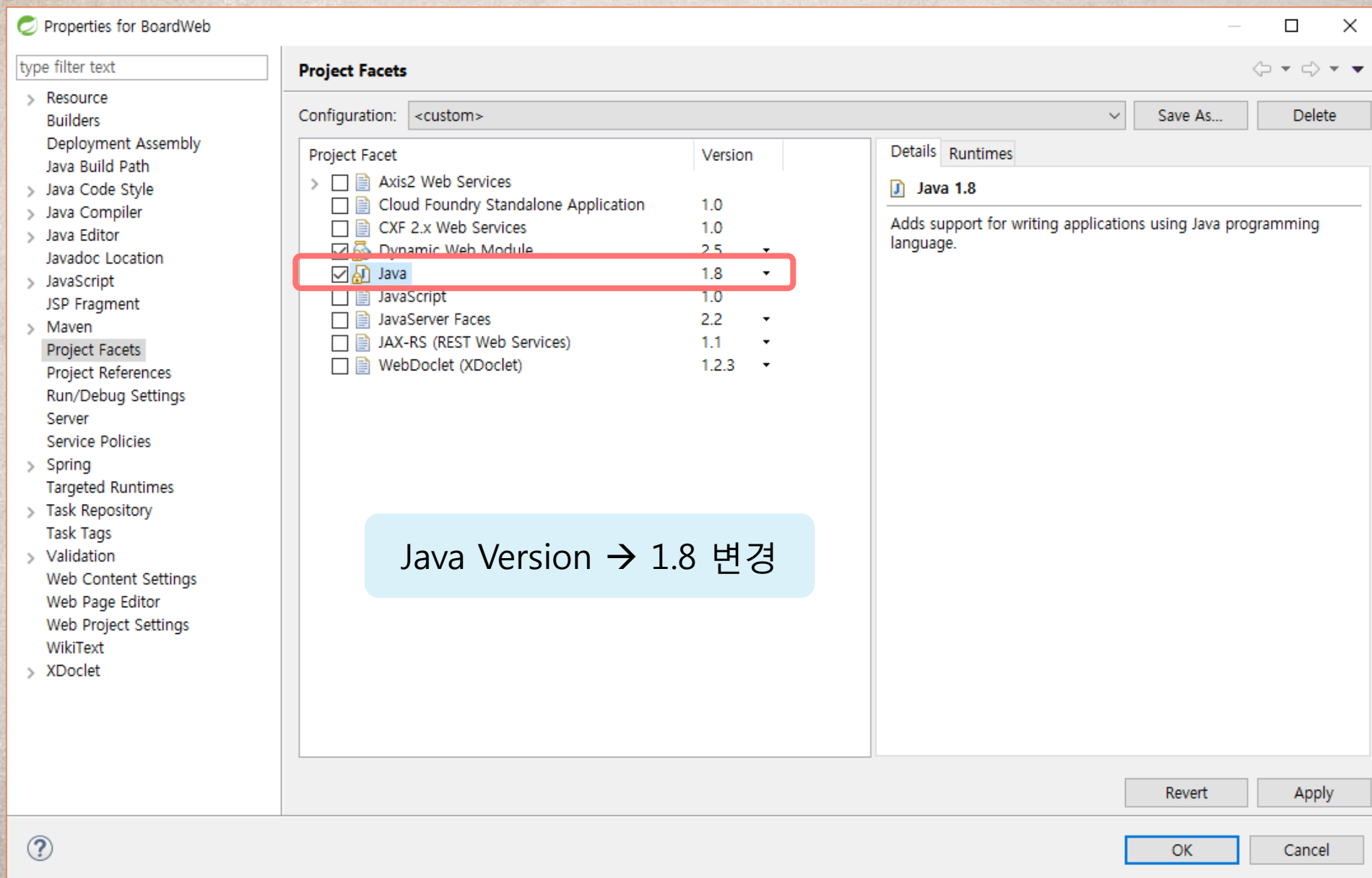
Cancel



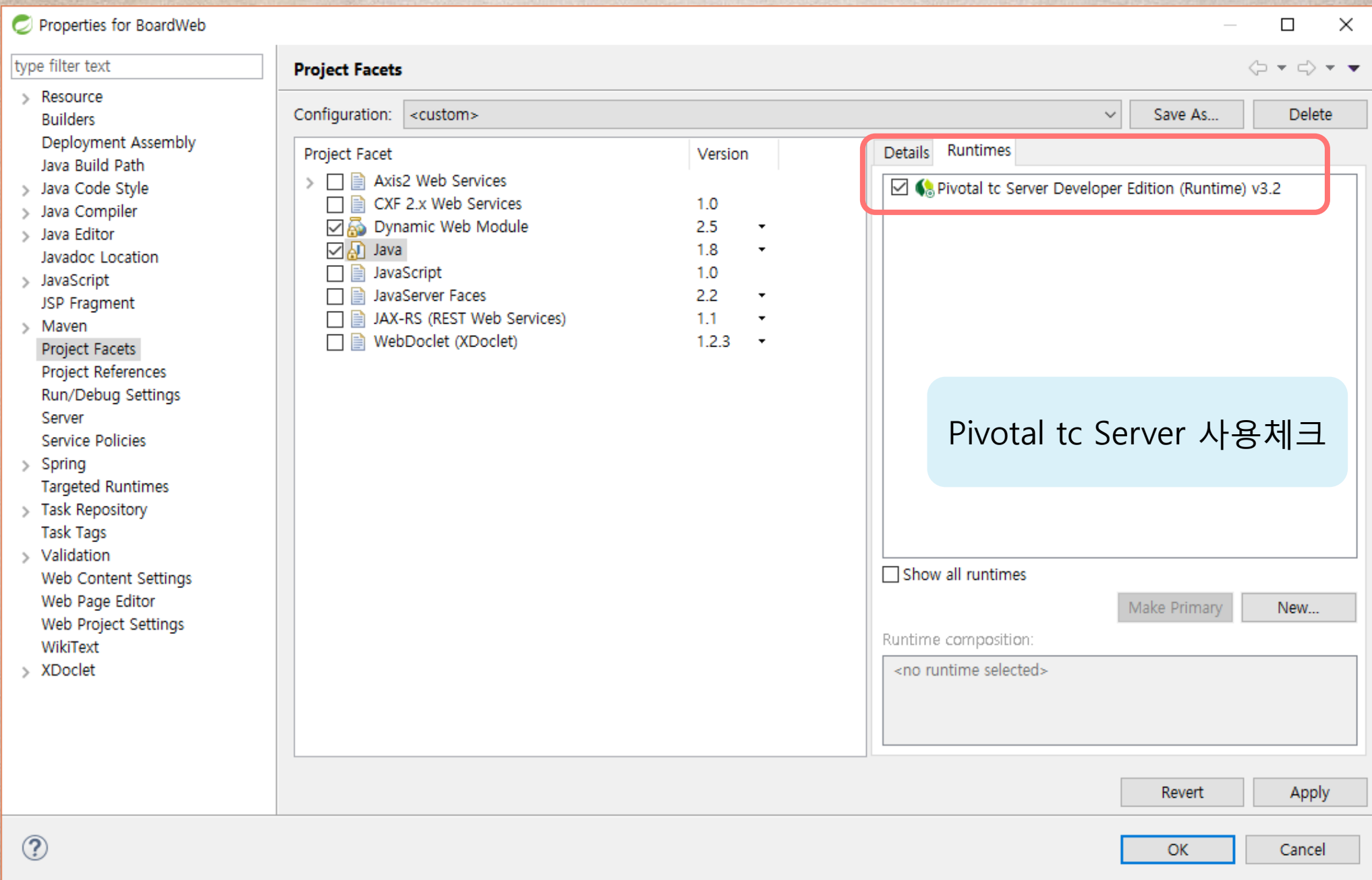




# Project의 Properties에서 Project Facets 설정

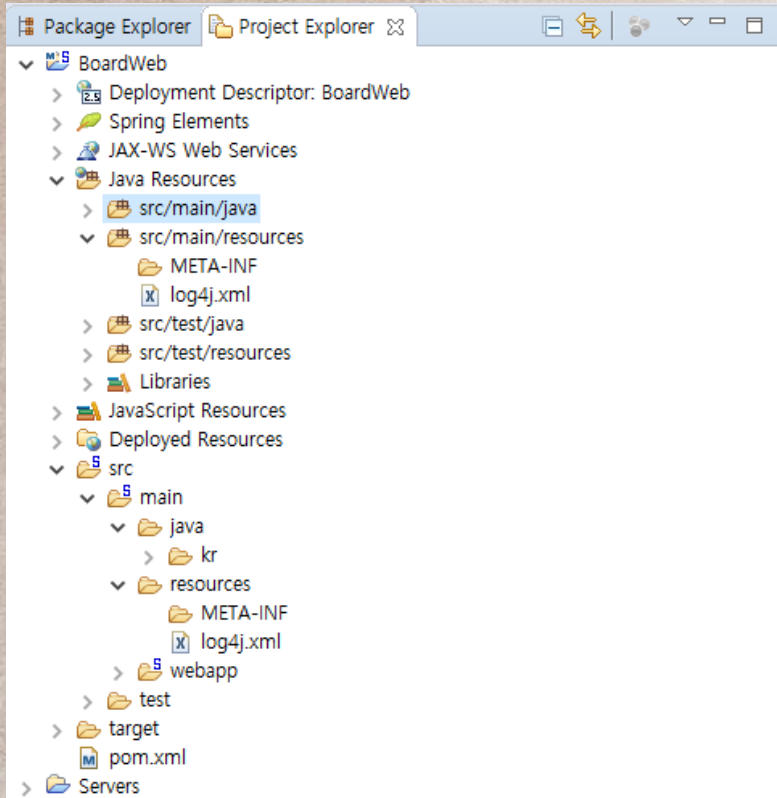




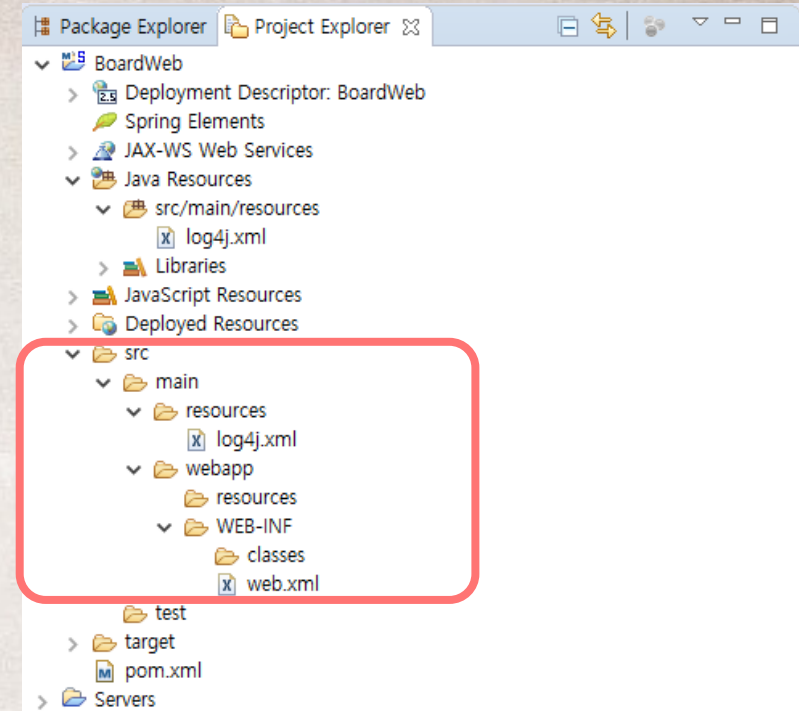
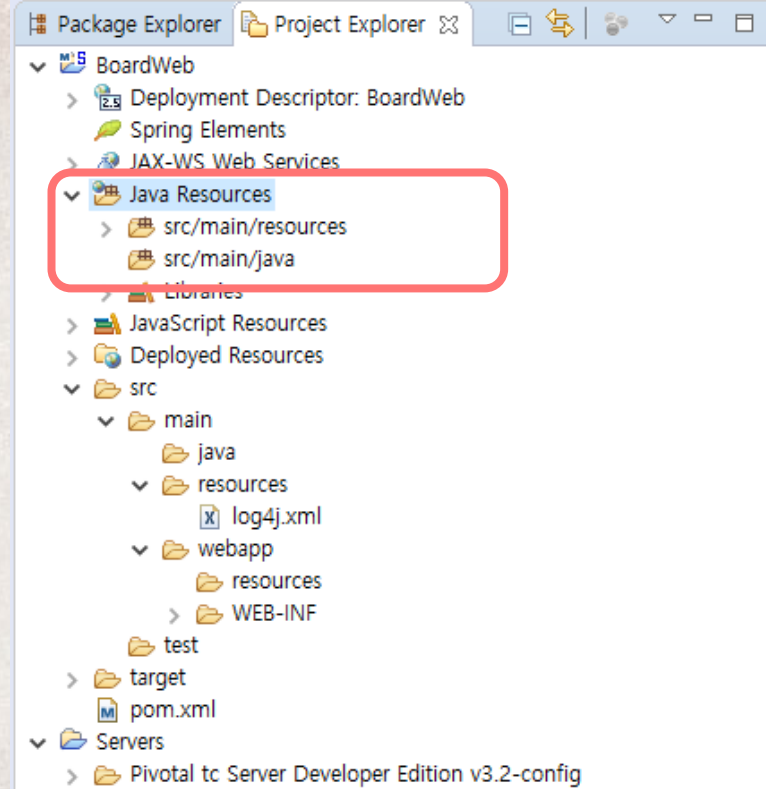




# 불필요한 폴더 및 파일 삭제



Java Resources에서  
src/main/resources/log4j.xml  
src/main/java 폴더 만 남겨  
놓고 모두 삭제



WEB-INF의 하위 폴더인  
spring과 views 폴더 삭제



# pom.xml 파일 - 사용할 Spring framework 버전 변경

The screenshot illustrates the steps to update the Spring Framework version in a Maven project's `pom.xml` file using the Eclipse IDE.

**Package Explorer:** Shows the project structure, including `src/main/java`, `src/main/resources`, `src/test/java`, `src/test/resources`, `Maven Dependencies`, `Pivotal tc Server v3.0 - v3.2 [Pivotal tc Server Developer Edition (R)]`, `JRE System Library [jre1.8.0_121]`, `src`, `main`, `test`, `target`, `pom.xml`, and `Servers`.

**Overview Tab:** Displays the project properties for `BoardWeb/pom.xml`.

- Artifact:** Group Id: `kr.ac`, Artifact Id: `*inje`, Version: `1.0.0-BUILD-SNAPSHOT`, Packaging: `war`.
- Parent:** (Empty)
- Properties:** A list of properties is shown, with `org.springframework-version : 4.3.8.RELEASE` highlighted by a red box.
- Modules:** (Empty)

A red arrow points from the highlighted property to a blue callout bubble with the text **더블클릭** (Double-click).

The **Edit property** dialog box is open, showing the property name `org.springframework-version` and the value `4.3.8.RELEASE` (highlighted by a red box). The **OK** button is visible.

**Project Properties:** Name: `BoardWeb`, URL: (Empty), Description: (Empty).

**Organization:** (Empty)

**SCM:** (Empty)

**Issue Management:** (Empty)

**Continuous Integration:** (Empty)

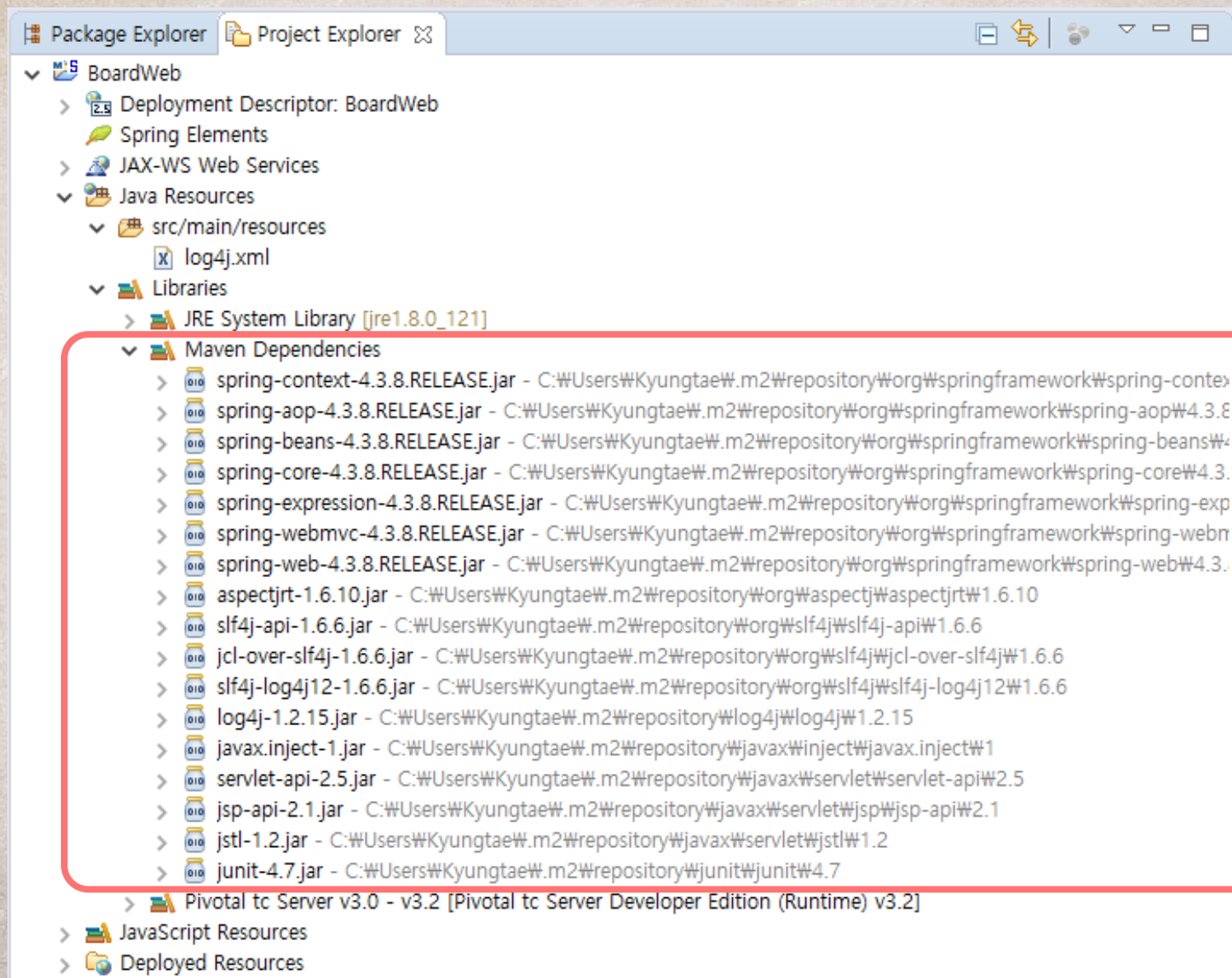
# web.xml 파일

```
*web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
5
6   <!-- The definition of the Root Spring Container shared by all Servlets and Filters -->
7   <context-param>
8     <param-name>contextConfigLocation</param-name>
9     <param-value>/WEB-INF/spring/root-context.xml</param-value>
10  </context-param>
11
12  <!-- Creates the Spring Container shared by all Servlets and Filters -->
13  <listener>
14    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
15  </listener>
16
17  <!-- Processes application requests -->
18  <servlet>
19    <servlet-name>appServlet</servlet-name>
20    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
21    <init-param>
22      <param-name>contextConfigLocation</param-name>
23      <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
24    </init-param>
25    <load-on-startup>1</load-on-startup>
26  </servlet>
27
28  <servlet-mapping>
29    <servlet-name>appServlet</servlet-name>
30    <url-pattern>/</url-pattern>
31  </servlet-mapping>
32
33 </web-app>
34
```

삭제



# [BoardWeb] → [Java Resources] → [Libraries] → [Maven Dependencies]





## 2. Spring Framework

- **경량(Light-weighted)**

- 크기와 부하 측면에서 경량임
- 몇 개의 jar 파일로 구성됨 → 설치 용이
- EJB와는 달리 Spring 자체의 부하는 무시해도 좋을 수준
- POJO(Plain Old Java Object) 사용

### 의존성 주입 방법

- 생성자를 이용하는 방법
- 설정 매서드를 이용하는 방법

- **IoC(Inversion Of Control)**

- 객체 사이의 의존 관계를 직접 자바코드로 구현하지 않으며 느슨한 결합을 도모
- 다른 객체를 생성하거나 찾는 대신 프레임워크에 의해 수동적으로 객체의 의존성을 주입 받음(Dependency Injection) → 의존성 주입이라고도 함
  - **의존성(Dependency)**이란? 어떤 클래스가 자신의 임무를 위해 필요한 값(필드값)이나 사용할 다른 클래스와의 관계를 말함.
  - **주입(Injection)**이란? 어떤 클래스의 인스턴스에 대해 다른 외부로부터 ‘의존성’, 즉 어떤 클래스나 필드값을 설정하는 것을 말함.



# Cont'd

## • AOP(Aspect Oriented Programming)

- 어플리케이션 객체는 원래해야 할 핵심 비즈니스 모듈만을 수행하는 것 외에는 아무것도 하지 않는다.
- 프로그래밍할 때 트랜잭션 지원이나 로깅같은 시스템적인 관심사에 신경 쓸 필요가 없음

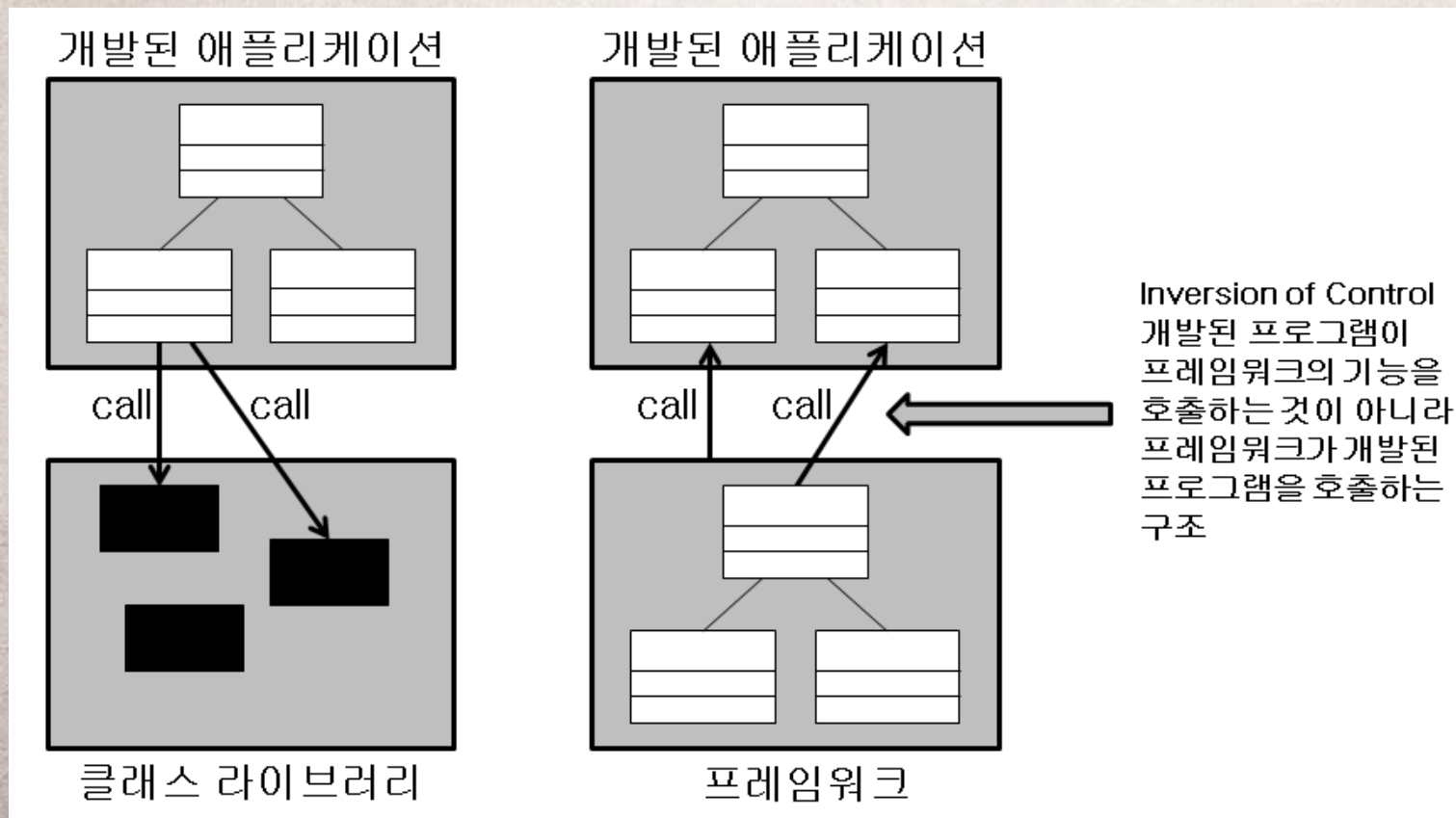
## • 컨테이너(Container)

- 객체의 생성 및 관리를 담당하는 일종의 서버와 같은 개념
- 애플리케이션 객체의 생명주기와 객체들 간의 의존관계를 관리한다는 점에서 Spring은 일종의 컨테이너임.
- 서블릿 컨테이너와 Spring 컨테이너의 차이점은 특정 WAS, 벤더에 종속적이지 않으면서 개발자가 자유롭게 확장가능하다는 점.
- 스프링의 가장 핵심적인 기능



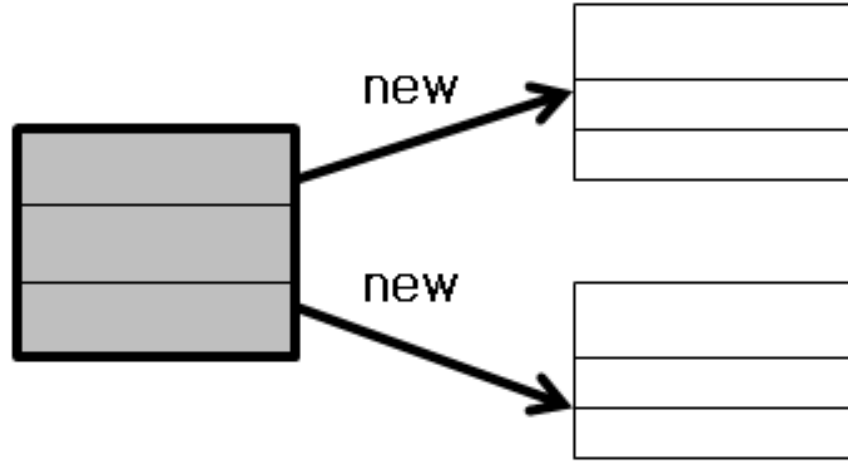
### 3. IoC(Inversion of Control) 컨테이너

- IOC(Inversion of Control): 제어의 역전
  - 개발자는 프레임워크의 확장 포인트를 상속받아 비즈니스 모듈 개발
  - 개발된 모듈은 프레임워크의 코어부분이 호출하는 구조로 바뀜
  - 클래스 라이브러리와 프레임워크의 차이점

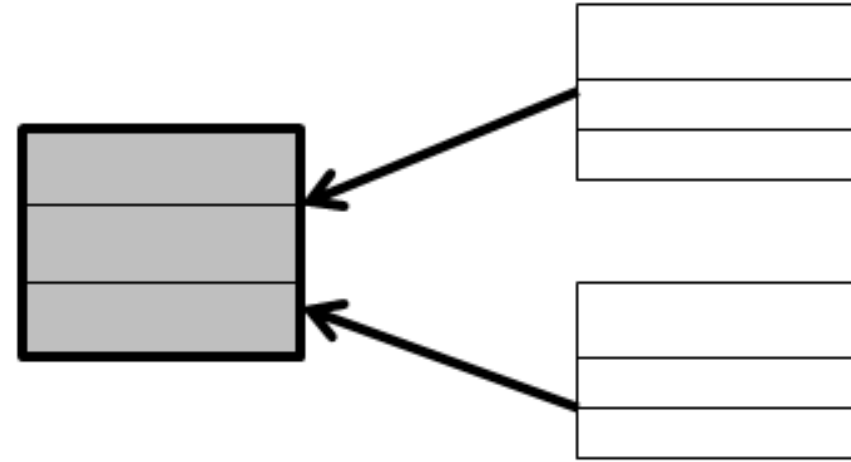




## Cont'd



[ Not IoC ]



[ IoC ]



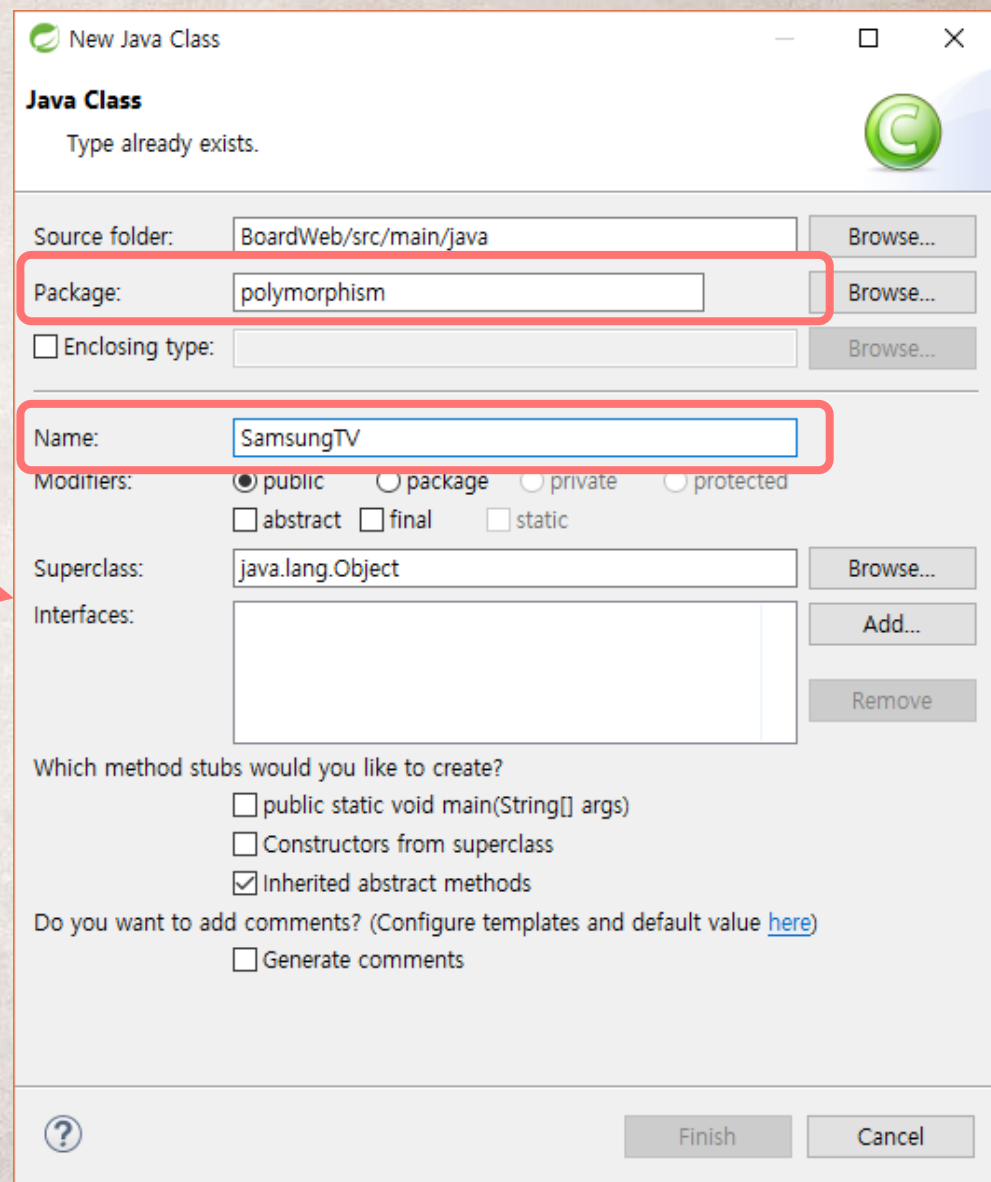
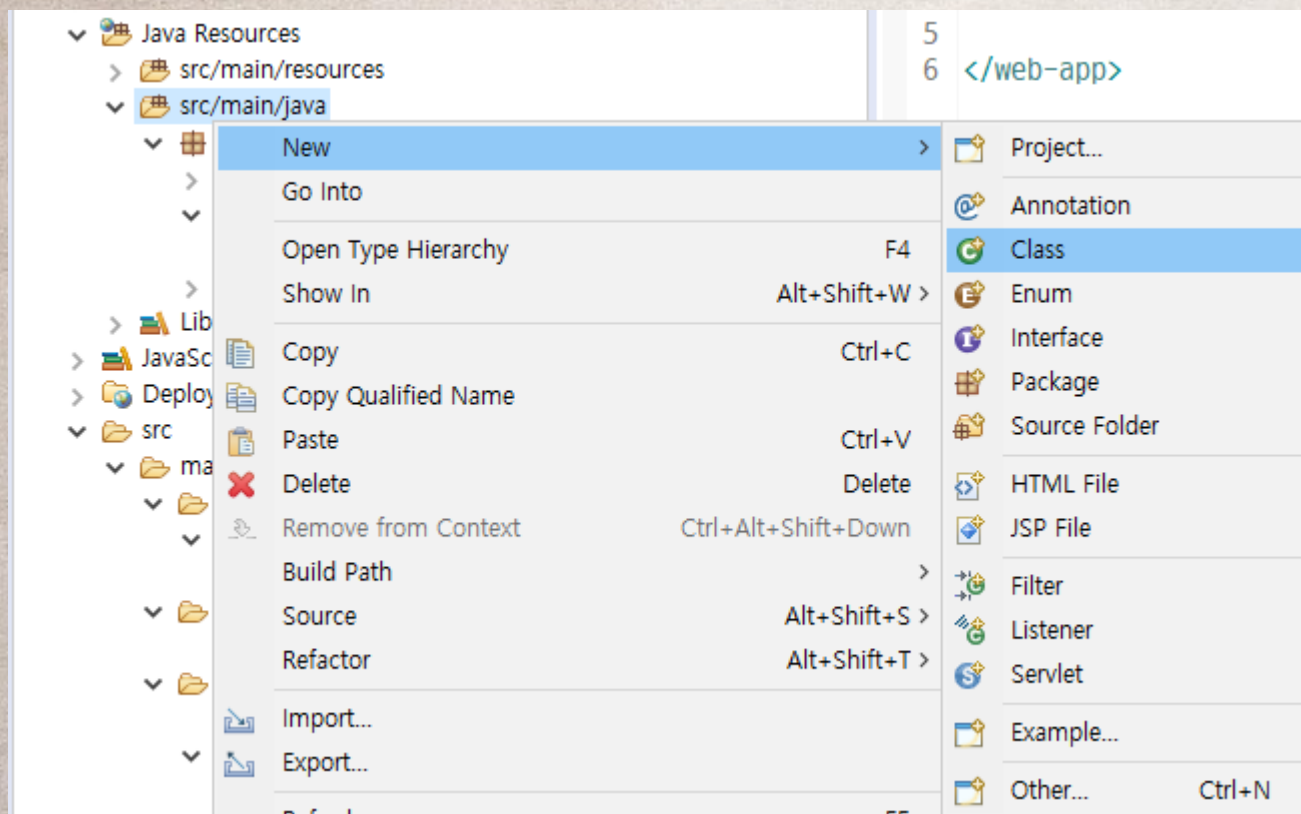
## 3.1 결합도(Coupling)가 높은 프로그램

- 결합도란 하나의 클래스가 다른 클래스와 얼마나 많이 연결되어 있는지를 나타내는 표현
- 결합도가 높은 프로그램은 유지보수가 어렵다. 유지 보수 관계를 확인하는 실습



# SamsungTV 클래스 작성

- src/main/java/main 소스 폴더에 작성



# SamsungTV.java 클래스

```
SamsungTV.java  LgTV.java  TVUser.java
1  package polymorphism;
2
3  public class SamsungTV {
4      public void powerOn(){
5          System.out.println("SamsungTV --- 전원을 켜다.");
6      }
7
8      public void powerOff(){
9          System.out.println("SamsungTV --- 전원을 끈다.");
10     }
11
12     public void volumeUp(){
13         System.out.println("SamsungTV --- 소리 올린다.");
14     }
15
16     public void volumeDown(){
17         System.out.println("SamsungTV --- 소리 내린다.");
18     }
19 }
20
```

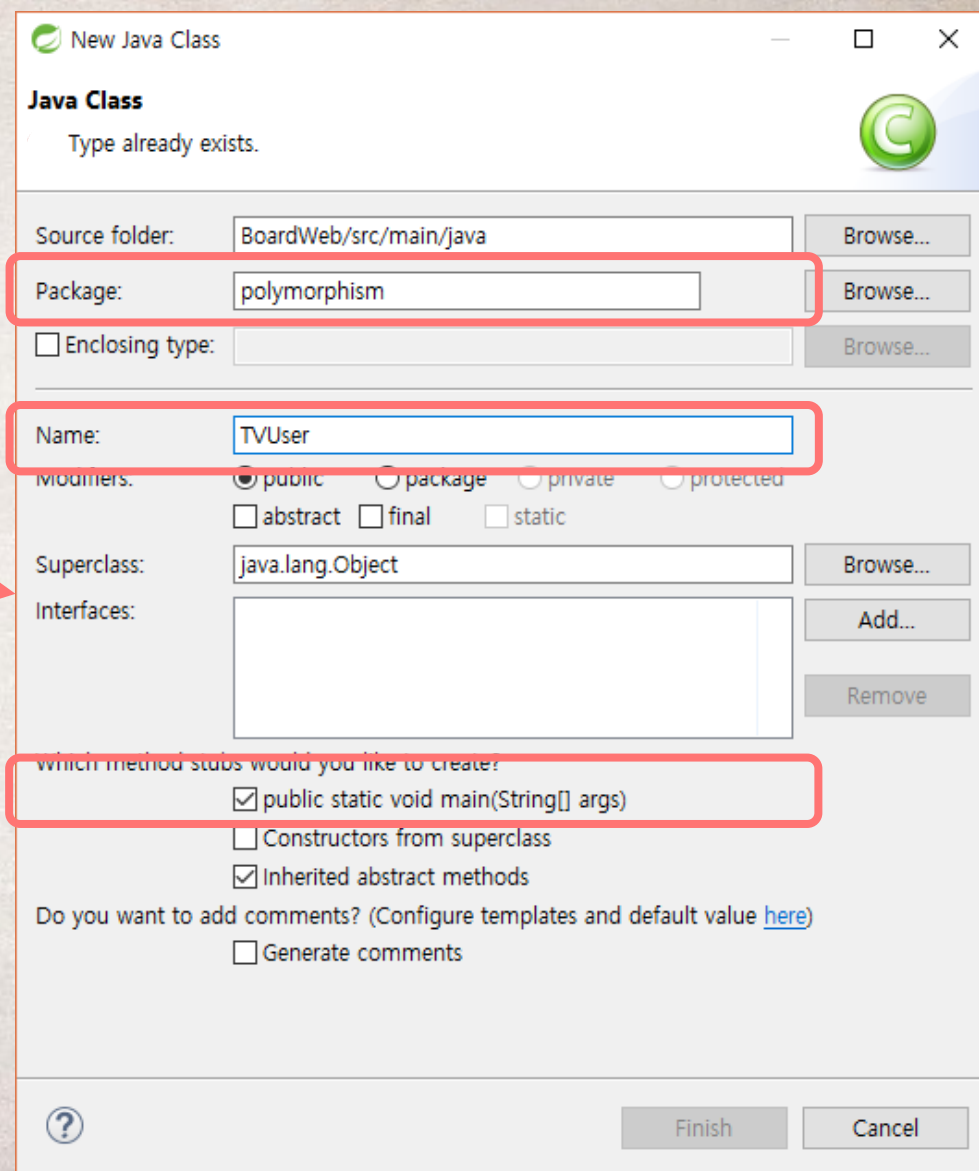
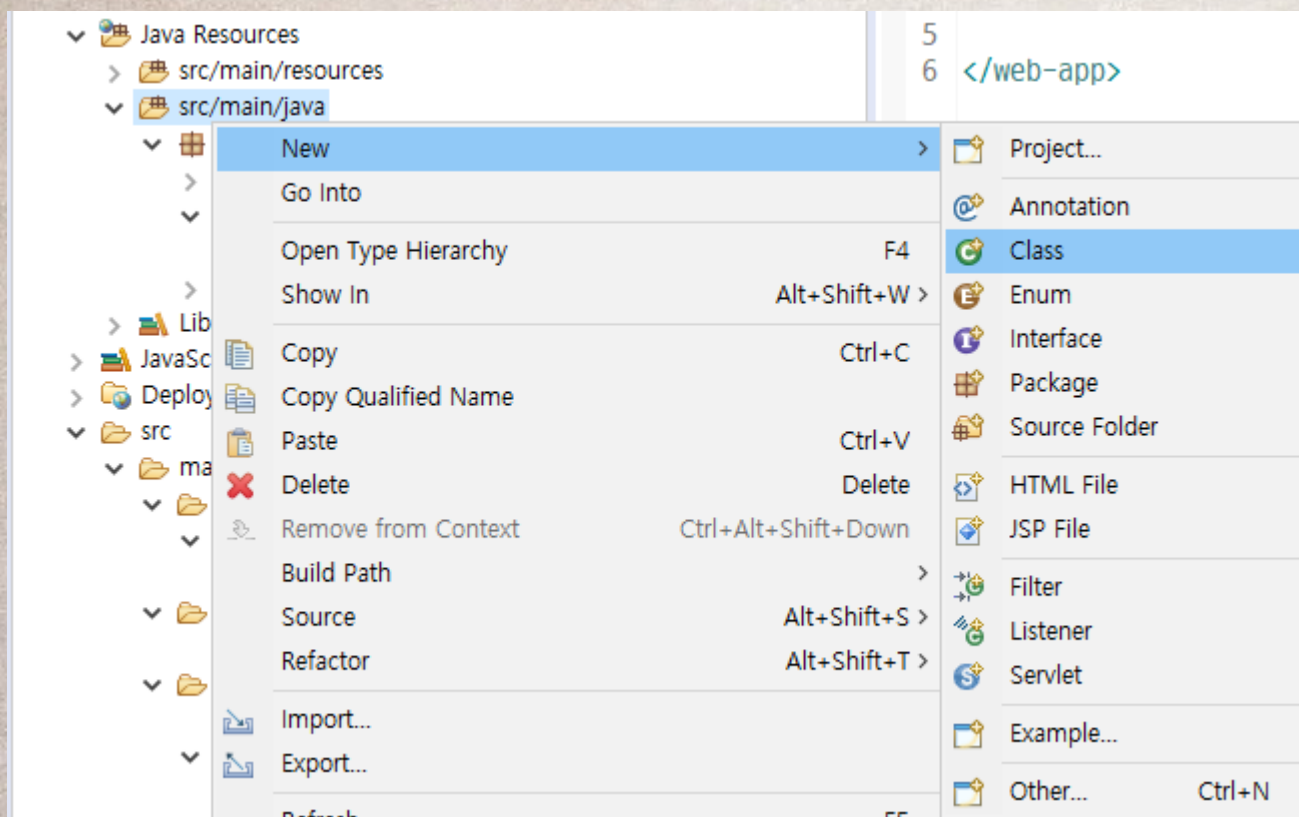


# LgTV.java 클래스

```
SamsungTV.java  LgTV.java  TVUser.java  web.xml
1 package polymorphism;
2
3 public class LgTV {
4     public void turnOn(){
5         System.out.println("Lg TV --- 전원을 켜다.");
6     }
7
8     public void turnOff(){
9         System.out.println("Lg TV --- 전원을 끈다.");
10    }
11
12    public void soundUp(){
13        System.out.println("Lg TV --- 소리를 올린다.");
14    }
15
16    public void soundDown(){
17        System.out.println("Lg TV --- 소리를 내린다.");
18    }
19 }
20
```

# SamsungTV를 시정하는 TVUser 프로그램 구현

- src/main/java/main 소스 폴더에 작성

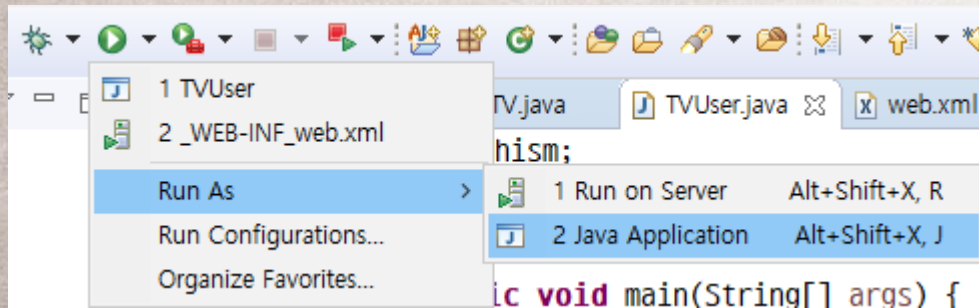




# TVUser.java 클래스 - SAMSUNG TV의 경우

```
SamsungTV.java  LgTV.java  TVUser.java  web.xml
1 package polymorphism;
2
3 public class TVUser {
4
5     public static void main(String[] args) {
6         SamsungTV tv = new SamsungTV();
7         tv.powerOn();
8         tv.volumeUp();
9         tv.volumeDown();
10        tv.powerOff();
11    }
12
13 }
14
```

## • 실행



Problems Javadoc Declaration Console Progress

```
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017.
SamsungTV --- 전원을 켜다.
SamsungTV --- 소리를 올린다.
SamsungTV --- 소리를 내린다.
SamsungTV --- 전원을 끈다.
```

# TVUser.java 클래스 - LG TV의 경우

```
SamsungTV.java  LgTV.java  TVUser.java  web.xml
1 package polymorphism;
2
3 public class TVUser {
4
5     public static void main(String[] args) {
6         LgTV tv = new LgTV();
7         tv.turnOn();
8         tv.soundUp();
9         tv.soundDown();
10        tv.turnOff();
11    }
12
13 }
14 |
```

## • 실행 결과

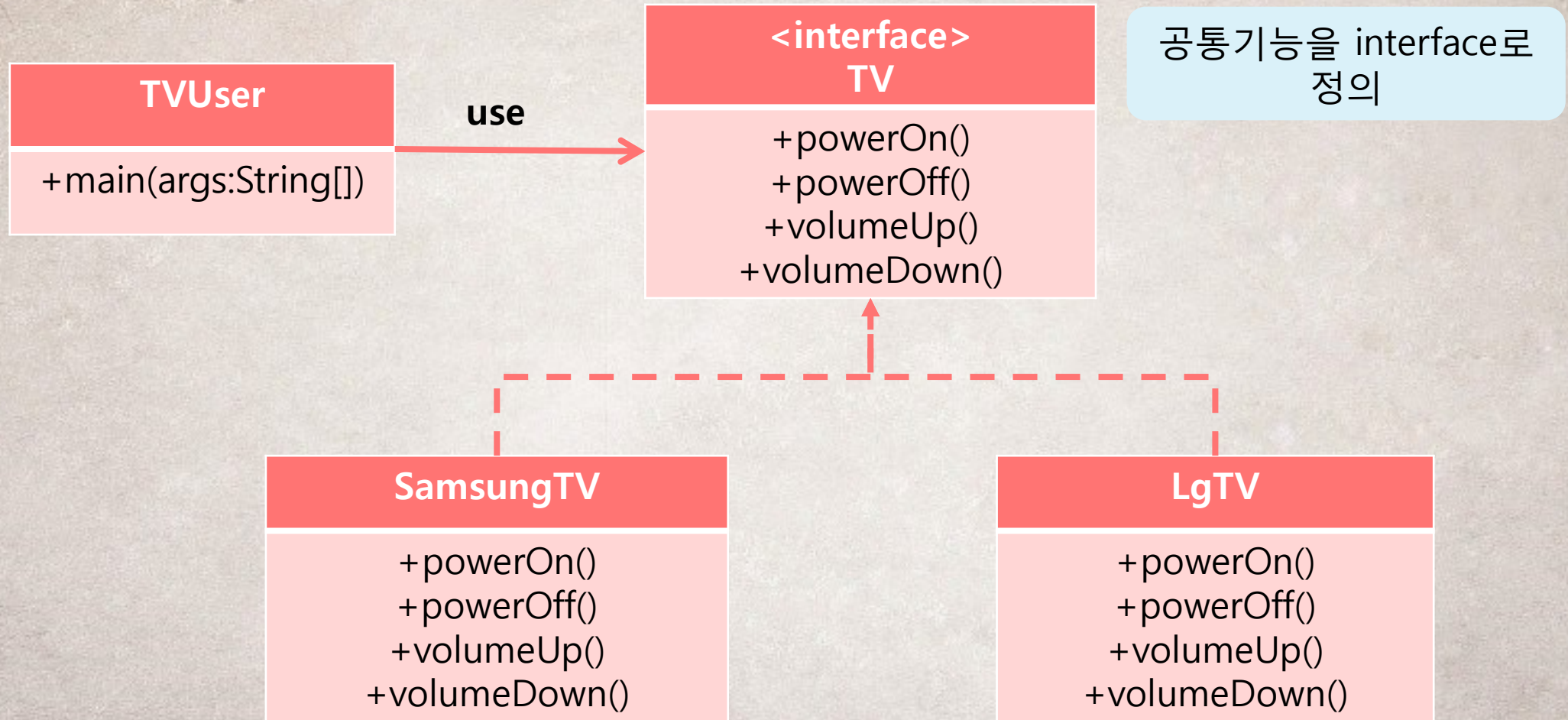
```
Problems  Javadoc  Declaration  Console  Progress
<terminated> TVUser [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (2017. 4. 26. 오후 6:08:5
Lg TV --- 전원을 켜다.
Lg TV --- 소리를 올린다.
Lg TV --- 소리를 내린다.
Lg TV --- 전원을 끈다.
```

SamsungTV와 LgTV는 메소드 시그니처 (signature)가 다르므로 코드 대부분을 수정

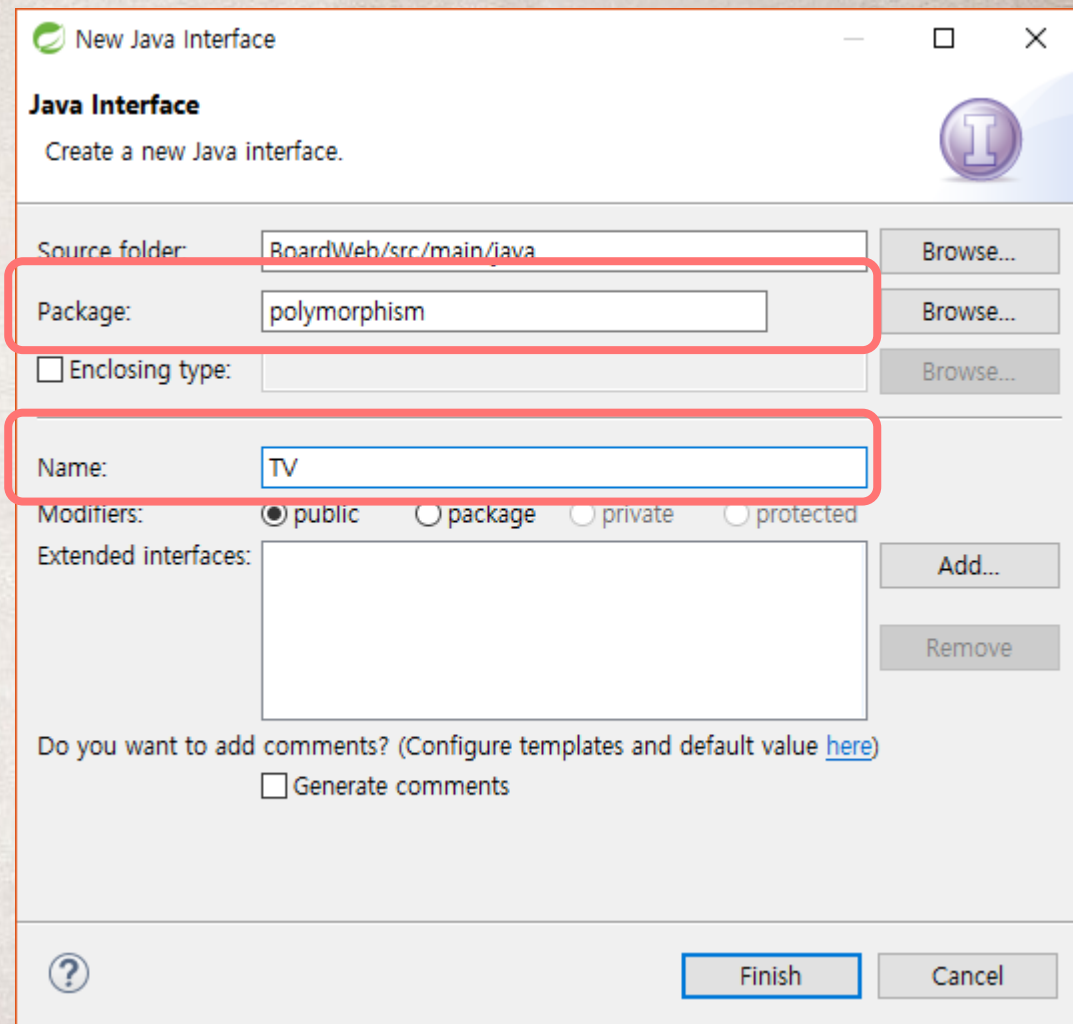
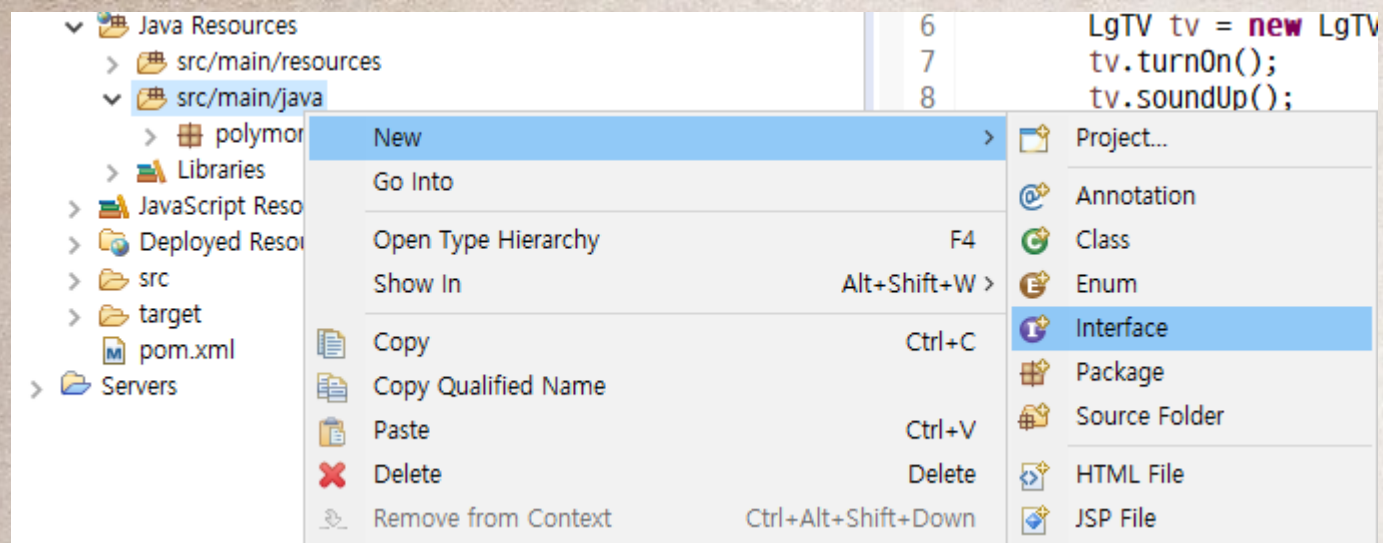
- TVUser와 같은 클라이언트 프로그램이 많으면 유지보수가 힘들 것
- TV 교체를 결정하기가 쉽지 않다.



## 3.2 결합도 낮추기 - 다형성 이용하기



# TV.java interface 만들기





# TV.java 인터페이스

```
TV.java SamsungTV.java LgTV.java TVUser.java
1 package polymorphism;
2
3 public interface TV {
4     public void powerOn();
5     public void powerOff();
6     public void volumeUp();
7     public void volumeDown();
8 }
9
10
```



# SamsungTV.java 클래스

- TV 인터페이스를 구현한 SamsungTV 클래스

```
TV.java SamsungTV.java LgTV.java TVUser.java
1 package polymorphism;
2
3 public class SamsungTV implements TV {
4     public void powerOn(){
5         System.out.println("SamsungTV --- 전원을 켜다.");
6     }
7
8     public void powerOff(){
9         System.out.println("SamsungTV --- 전원을 끈다.");
10    }
11
12    public void volumeUp(){
13        System.out.println("SamsungTV --- 소리 올린다.");
14    }
15
16    public void volumeDown(){
17        System.out.println("SamsungTV --- 소리 내린다.");
18    }
19 }
20
21
```



# LgTV.java 클래스

- TV 인터페이스를 구현한 LgTV 클래스

```
TV.java SamsungTV.java LgTV.java TVUser.java
1 package polymorphism;
2
3 public class LgTV implements TV {
4     public void powerOn(){
5         System.out.println("Lg TV --- 전원을 켜다.");
6     }
7
8     public void powerOff(){
9         System.out.println("Lg TV --- 전원을 끈다.");
10    }
11
12    public void volumeUp(){
13        System.out.println("Lg TV --- 소리를 올린다.");
14    }
15
16    public void volumeDown(){
17        System.out.println("Lg TV --- 소리를 내린다.");
18    }
19 }
20 |
```



# TVUser.java 클래스

```
TV.java SamsungTV.java LgTV.java TVUser.java ✖
1 package polymorphism;
2
3 public class TVUser {
4
5     public static void main(String[] args) {
6         TV tv = new SamsungTV();
7         tv.powerOn();
8         tv.volumeUp();
9         tv.volumeDown();
10        tv.powerOff();
11    }
12
13 }
14
```

- TVUser와 같은 클라이언트가 여러 개라도 최소한의 수정으로 TV를 교체할 수 있다. → 좀 더 편한 유지보수
- LgTV로 교체하고자 할 경우  
→ **TV tv = new LgTV();**
- TV 클래스 객체 생성 소스 수정 필요

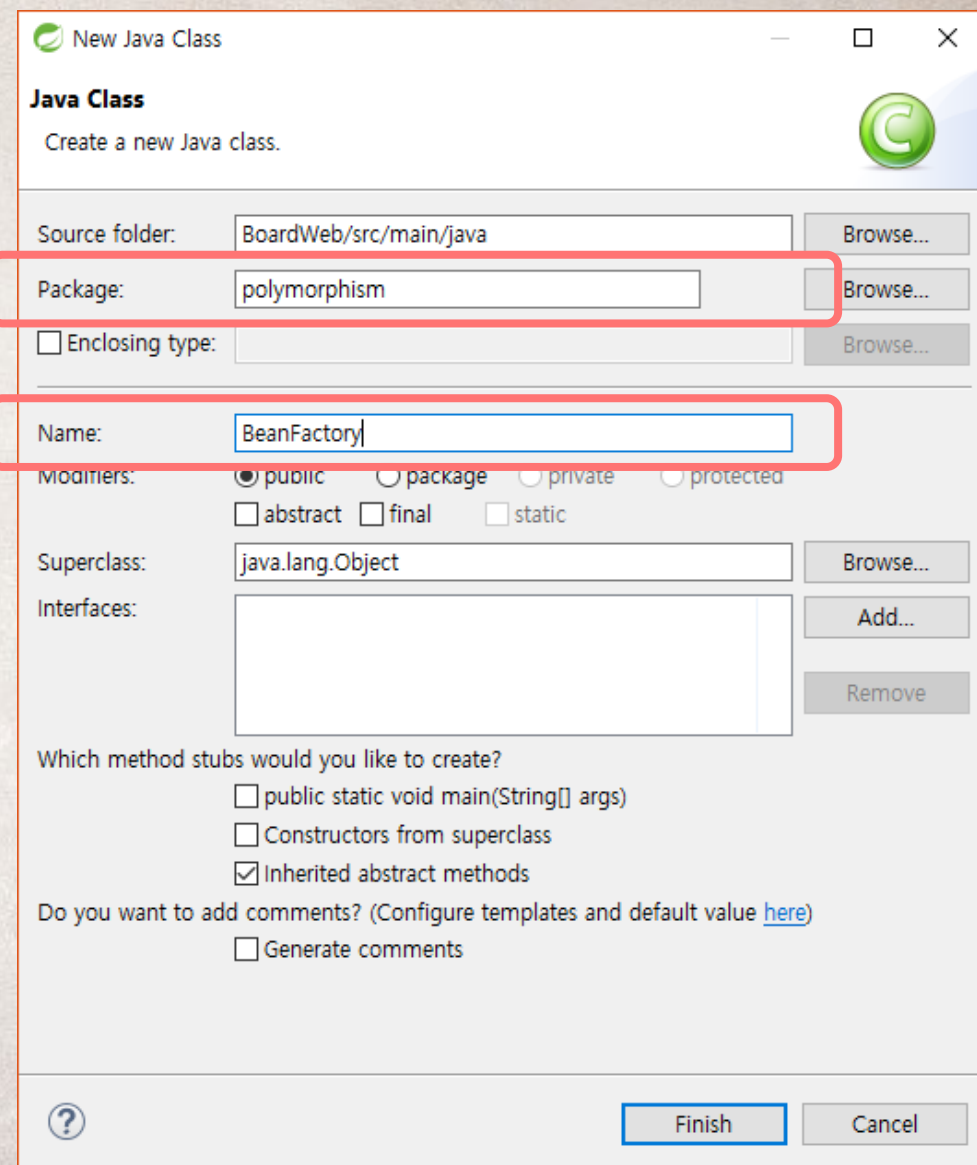
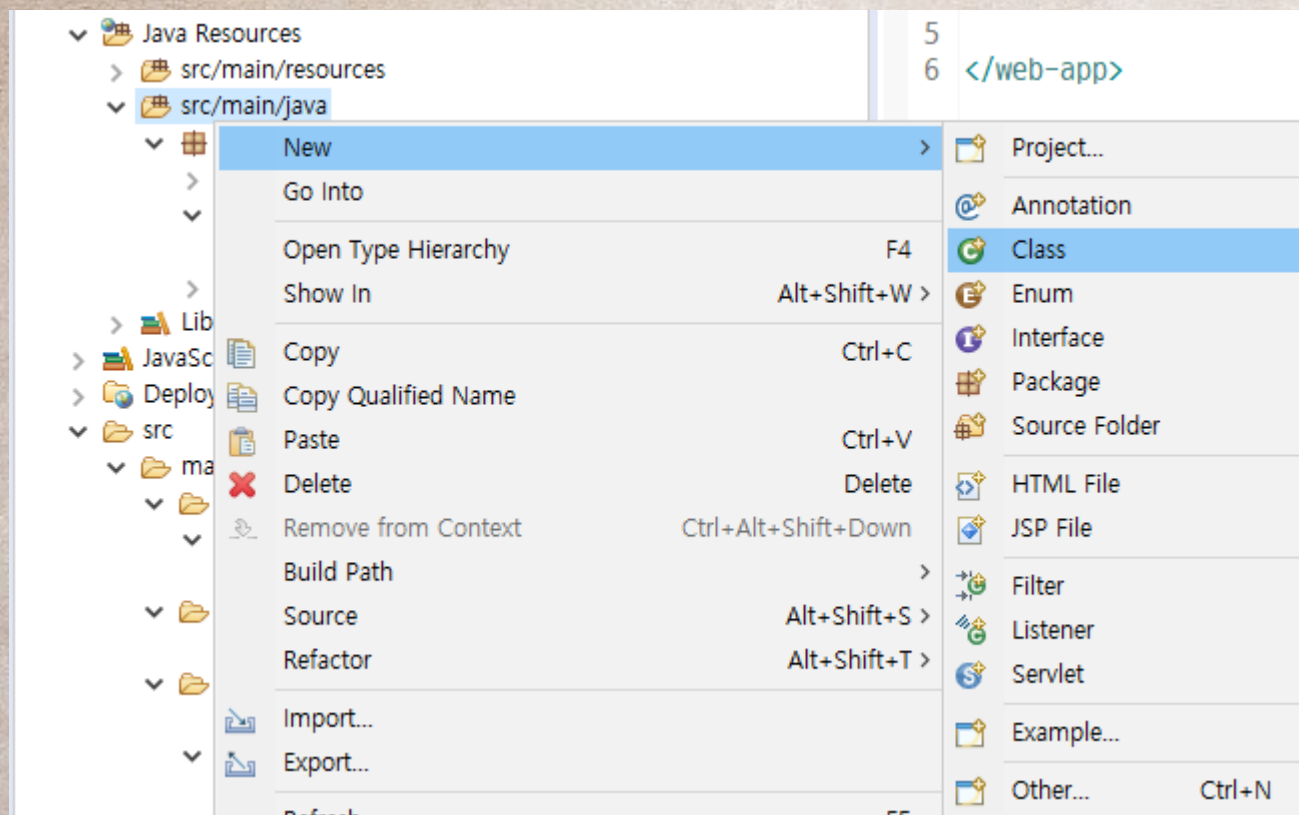


### 3.3 디자인 패턴 이용하기

- TV를 교체할 때 클라이언트 소스를 수정하지 않고 TV를 교체할 수 있다면!!!
- Factory 패턴을 적용하여 객체 생성의 캡슐화하여 TVUser와 TV 사이의 느슨한 결합 상태로 만들면 됨.



# BeanFactory.java 클래스





# BeanFactory.java 클래스

```
TVUser.java BeanFactory.java ✖
1 package polymorphism;
2
3 public class BeanFactory {
4     public Object getBean(String beanName){
5         if (beanName.equals("samsung")){
6             return new SamsungTV();
7         }else if (beanName.equals("lg")){
8             return new LgTV();
9         }
10        return null;
11    }
12 }
13
```

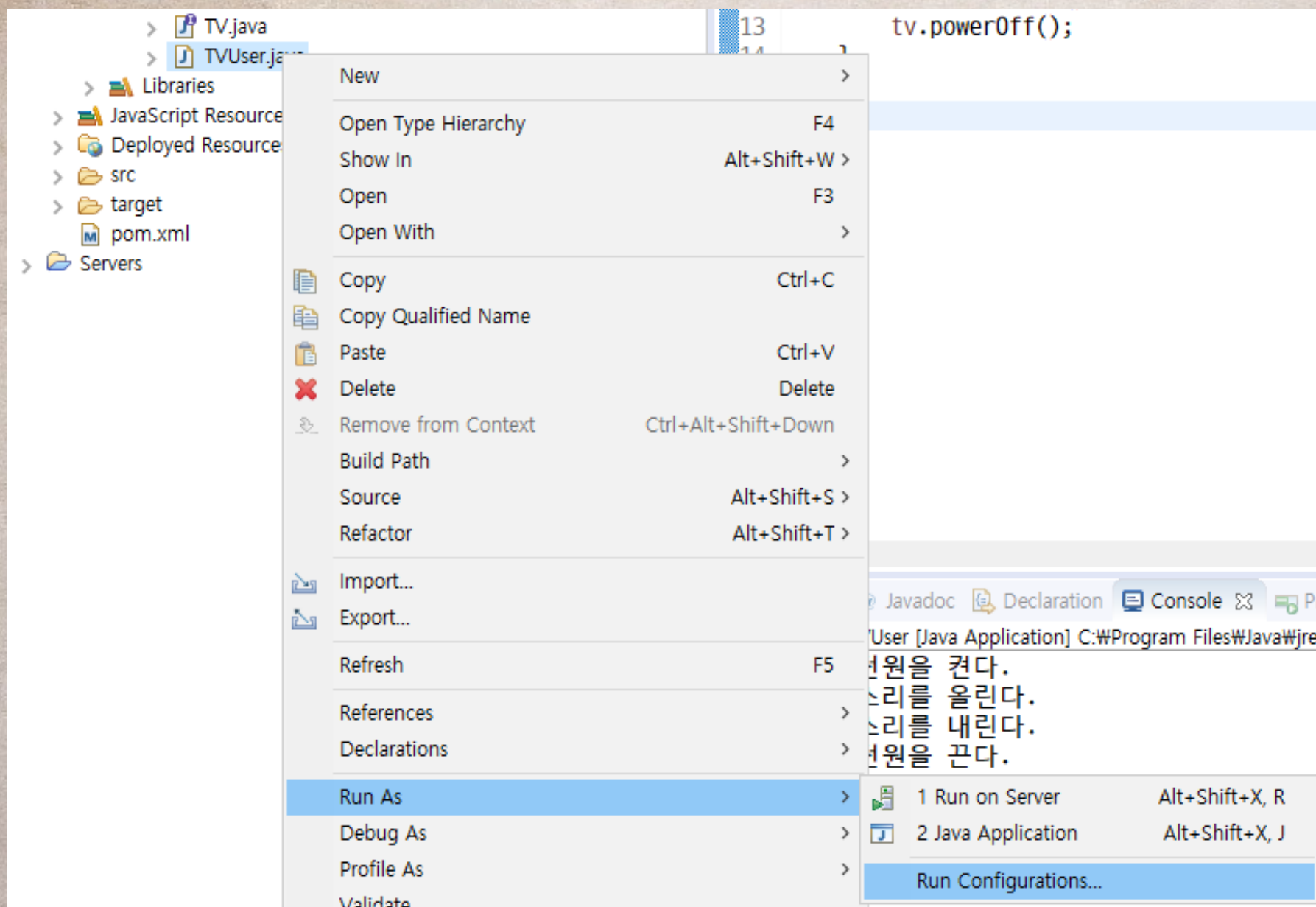


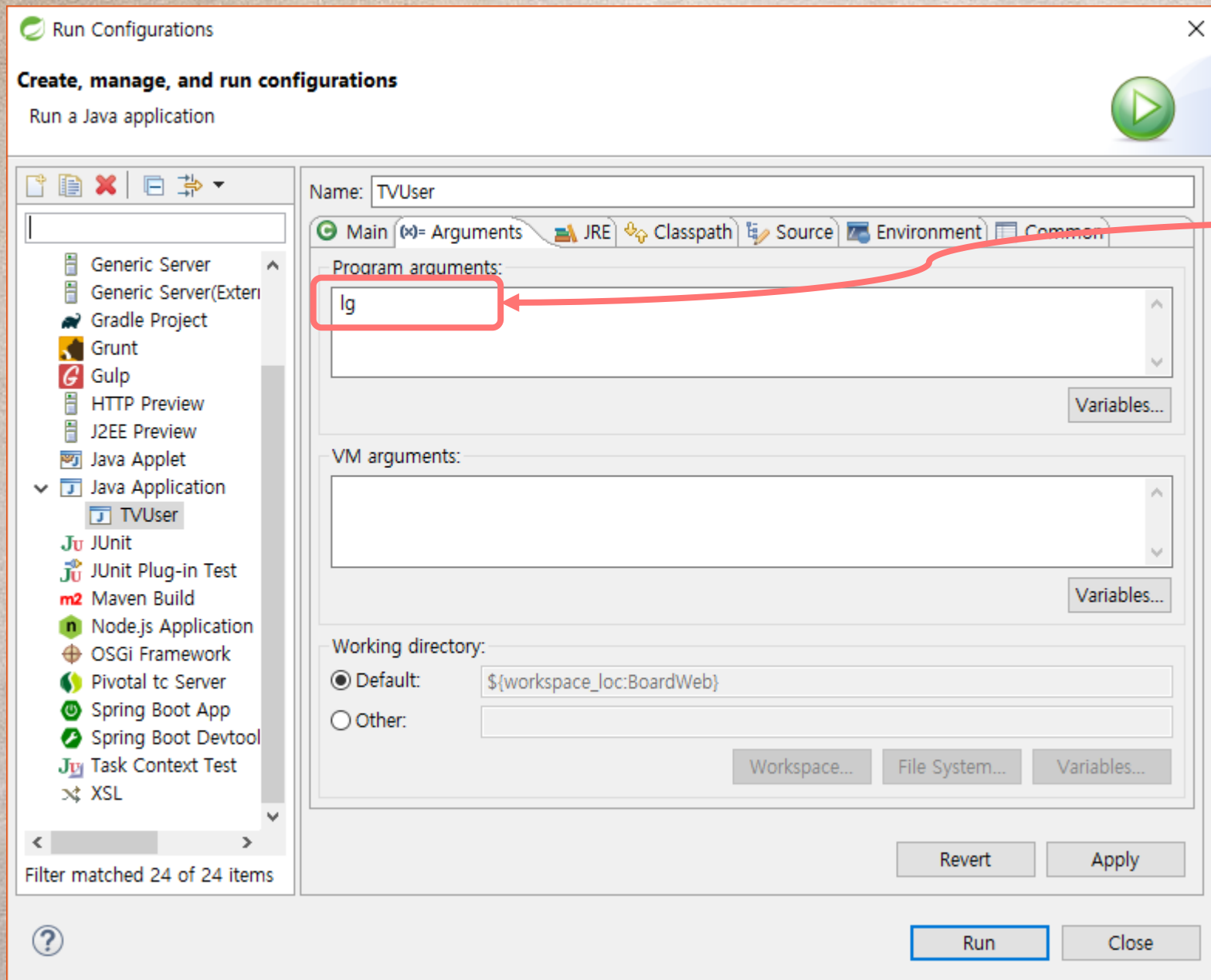
# TVUser.java 클래스

```
TVUser.java BeanFactory.java
1 package polymorphism;
2
3 public class TVUser {
4
5     public static void main(String[] args) {
6
7         BeanFactory factory = new BeanFactory();
8         TV tv = (TV) factory.getBean(args[0]);
9
10        tv.powerOn();
11        tv.volumeUp();
12        tv.volumeDown();
13        tv.powerOff();
14    }
15 }
16
```



# 실행 방법

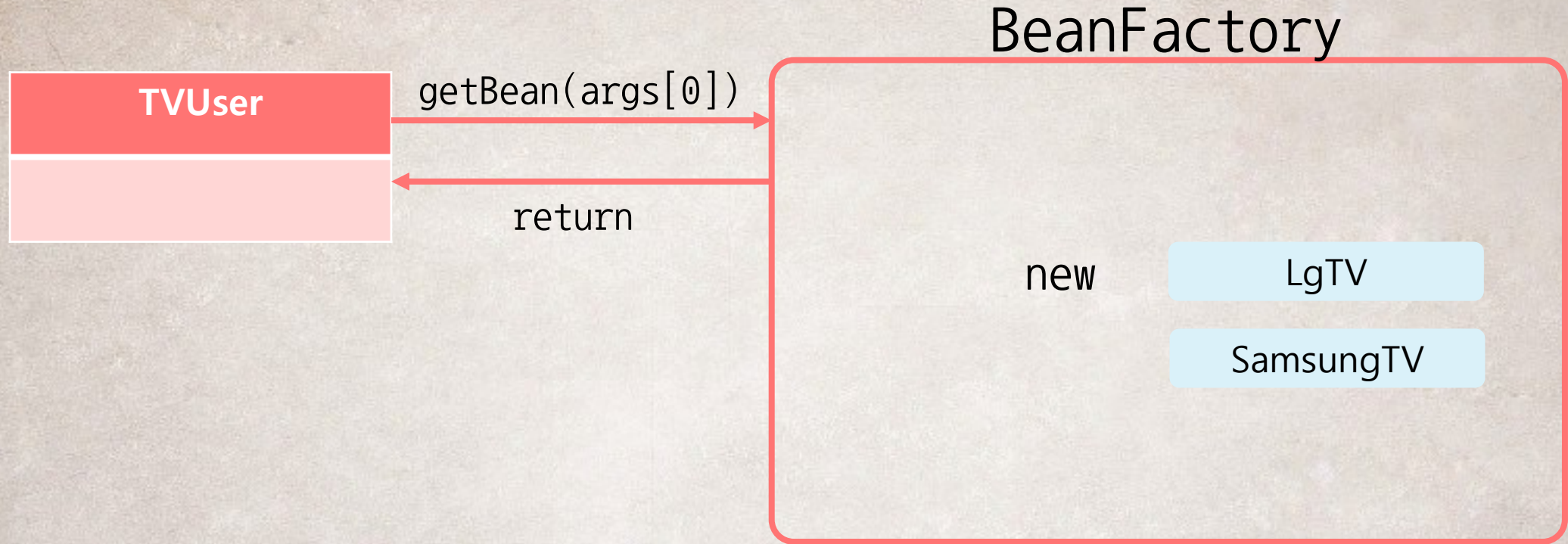




lg 또는 samsung



# TVUser 프로그램 실행 과정



- 명령행 매개변수만 수정하여 실행 → 소스 수정하지 않음
- TVUser는 자신이 필요한 객체를 직접 생산하지 않는다.