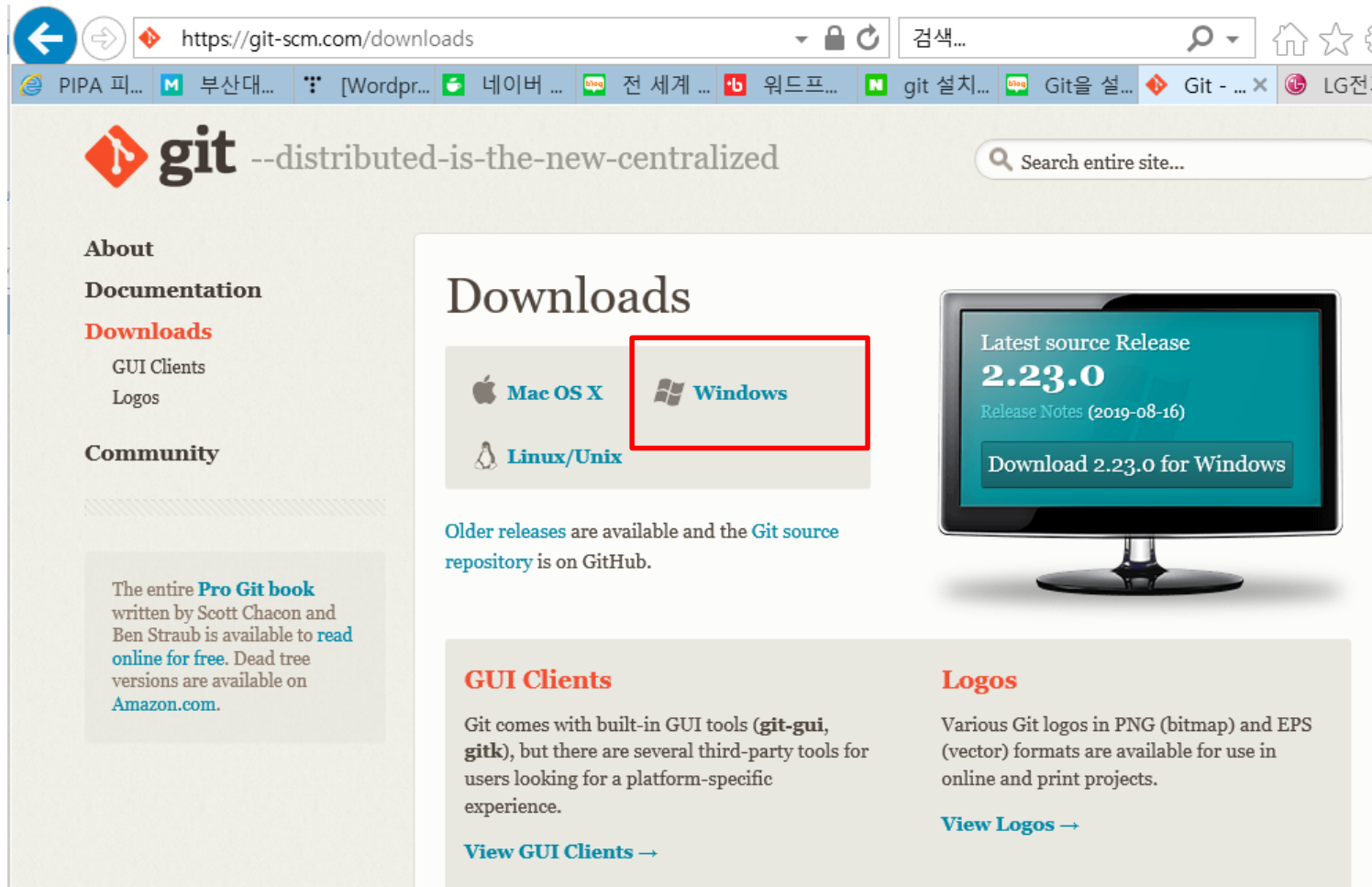


# 형상관리 / VO/DAO

---

# Git

✓ 다운로드 : <https://git-scm.com/downloads>



The screenshot shows the Git website's Downloads page. The browser's address bar displays <https://git-scm.com/downloads>. The page features the Git logo and the tagline "--distributed-is-the-new-centralized". A search bar is located in the top right corner. On the left sidebar, there are links for "About", "Documentation", "Downloads" (highlighted in red), "GUI Clients", "Logos", and "Community". The main content area is titled "Downloads" and includes three platform-specific download buttons: "Mac OS X", "Windows" (highlighted with a red rectangle), and "Linux/Unix". To the right of these buttons is a large monitor graphic displaying the "Latest source Release 2.23.0" and a button to "Download 2.23.0 for Windows". Below the platform buttons, a note states: "Older releases are available and the Git source repository is on GitHub." At the bottom, there are sections for "GUI Clients" and "Logos", each with a brief description and a link to view more options.

**About**

**Documentation**

**Downloads**


GUI Clients


Logos


**Community**

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## Downloads

 **Mac OS X**

 **Windows**

 **Linux/Unix**

Latest source Release  
**2.23.0**  
[Release Notes \(2019-08-16\)](#)  
[Download 2.23.0 for Windows](#)

Older releases are available and the [Git source repository](#) is on GitHub.

### GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

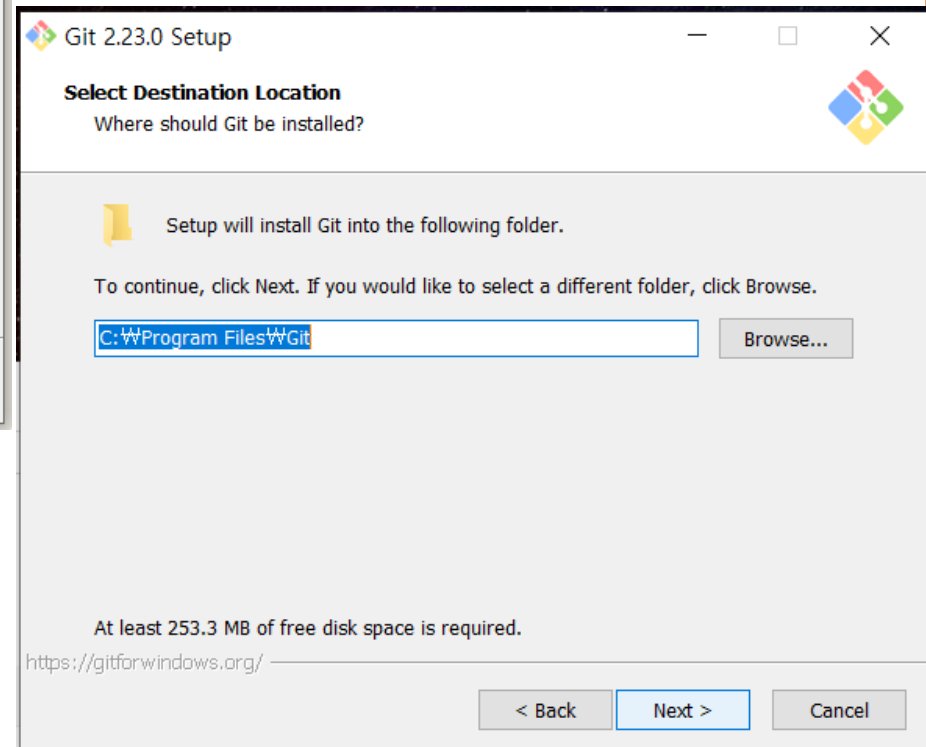
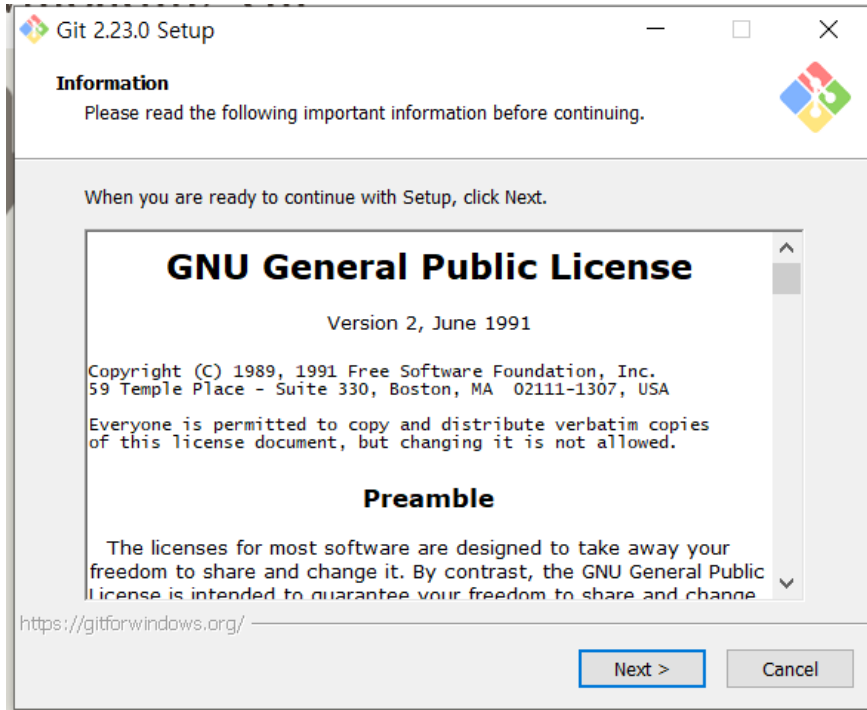
### Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

# Git

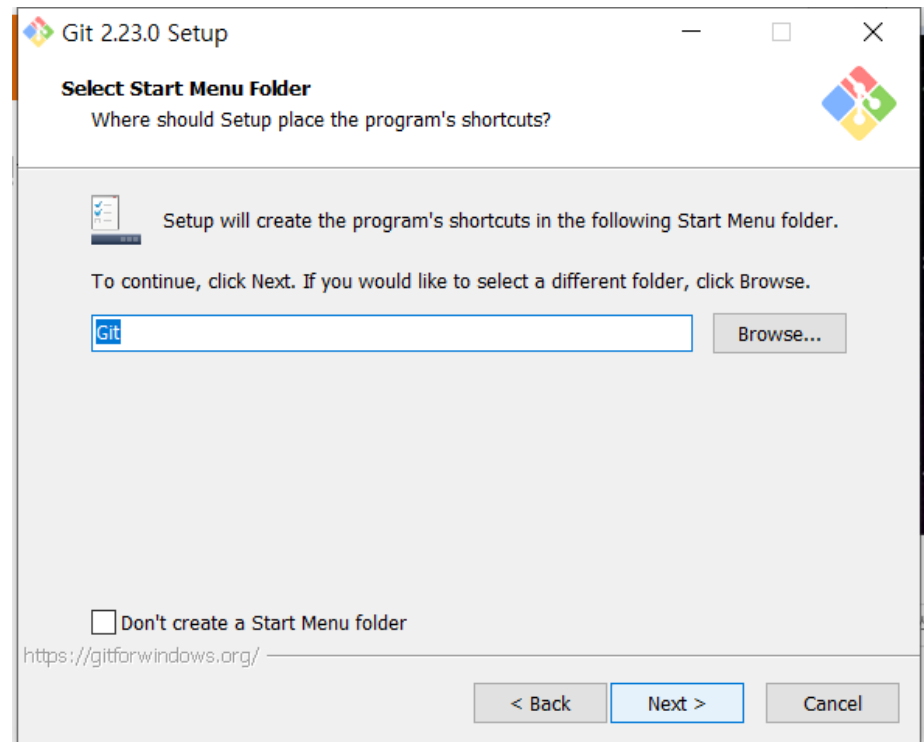
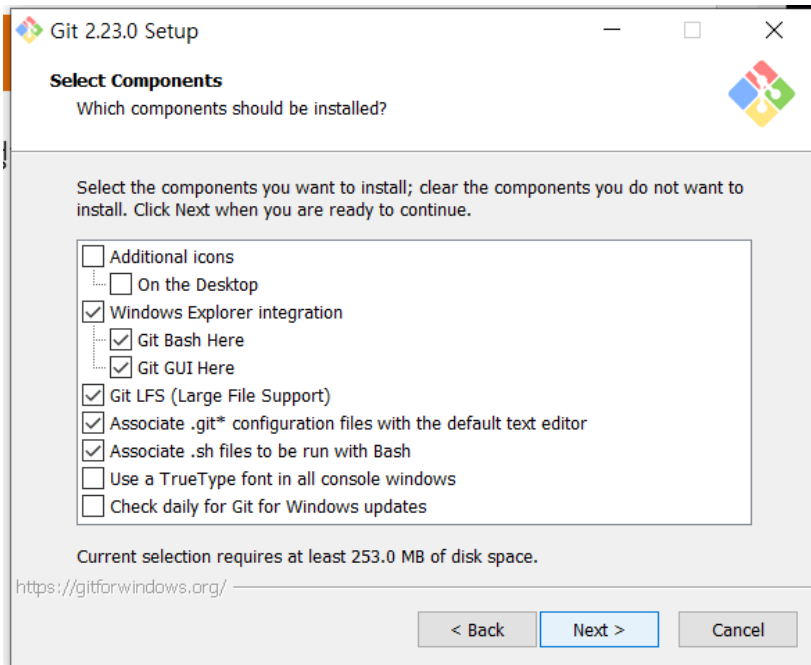
- ❖ 형상관리 도구를 설치한다.
- ✓ 다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.



# Git

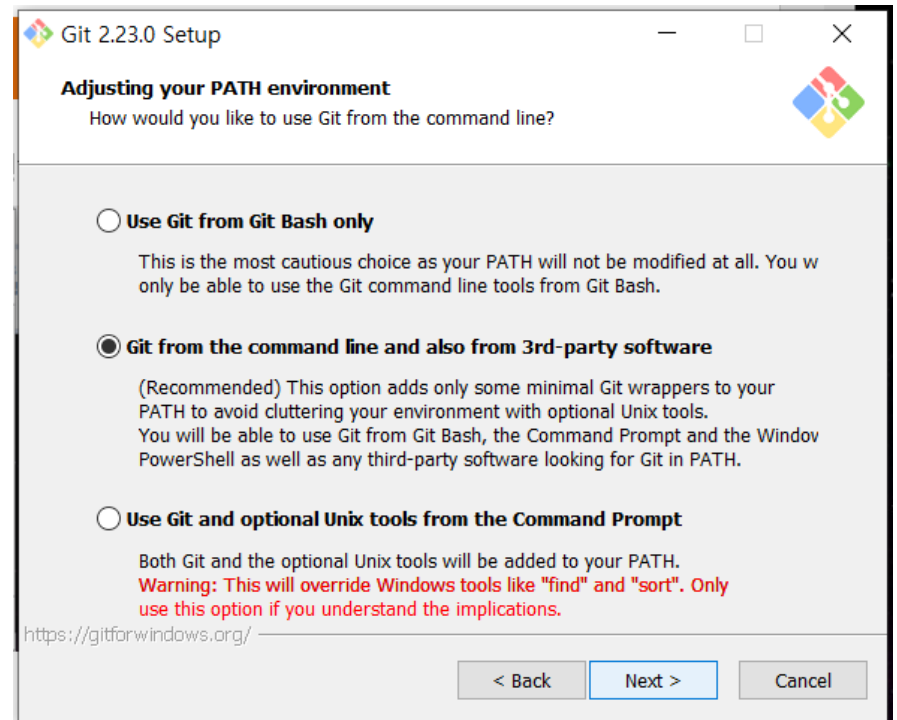
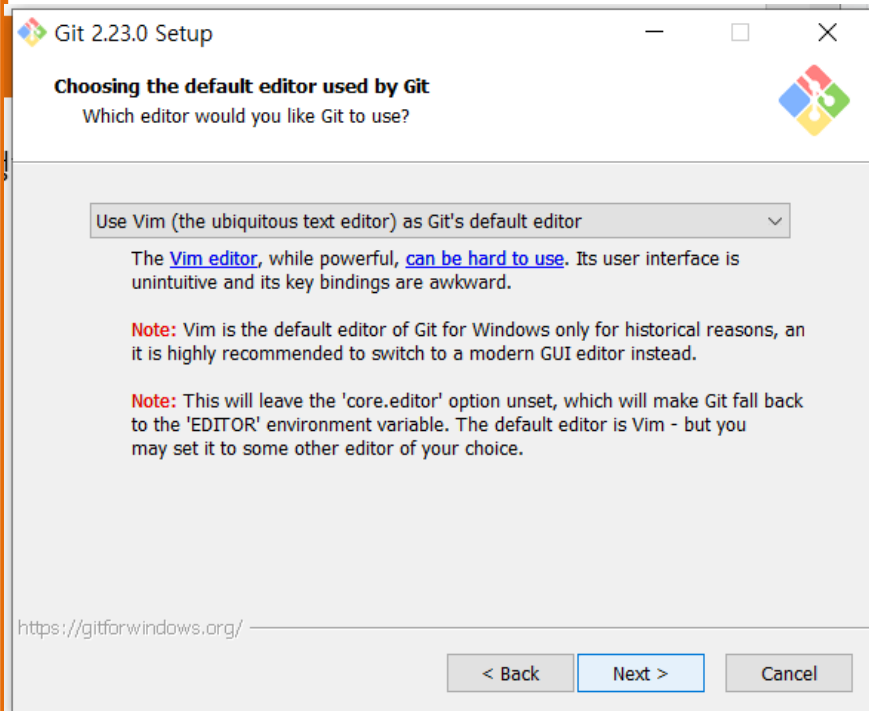
❖ 형상관리 도구를 설치한다.

✓ 다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.

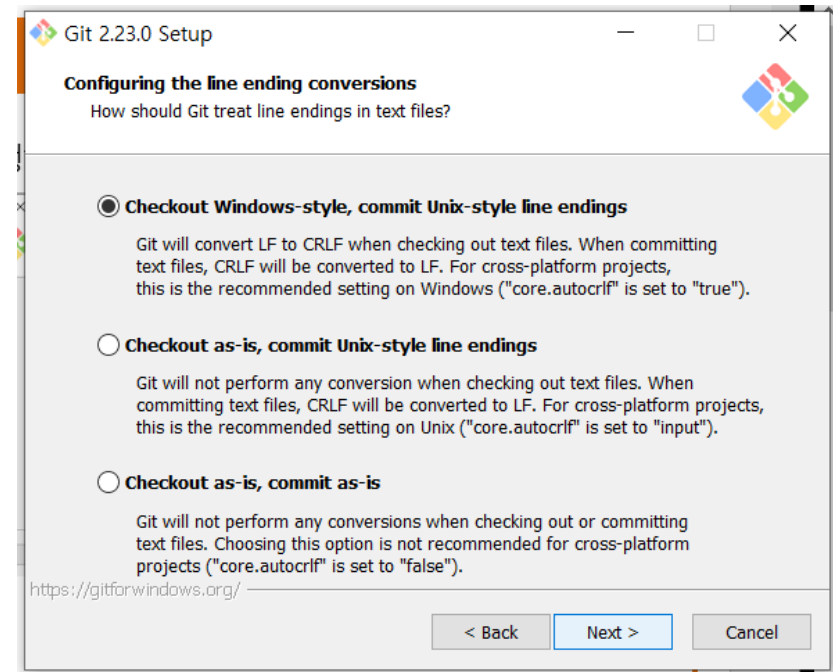
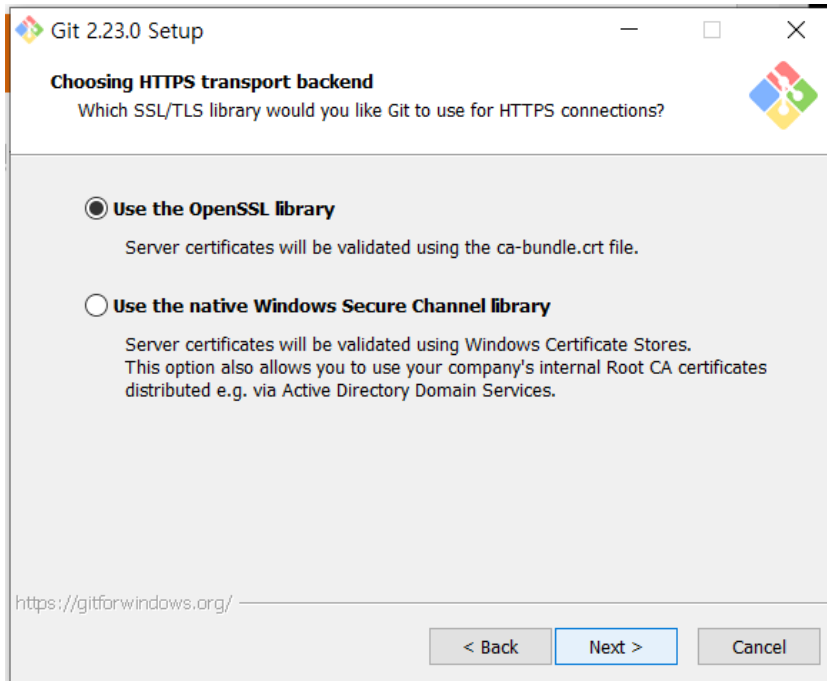


# Git

- ❖ 형상관리 도구를 설치한다.
- ✓ 다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.

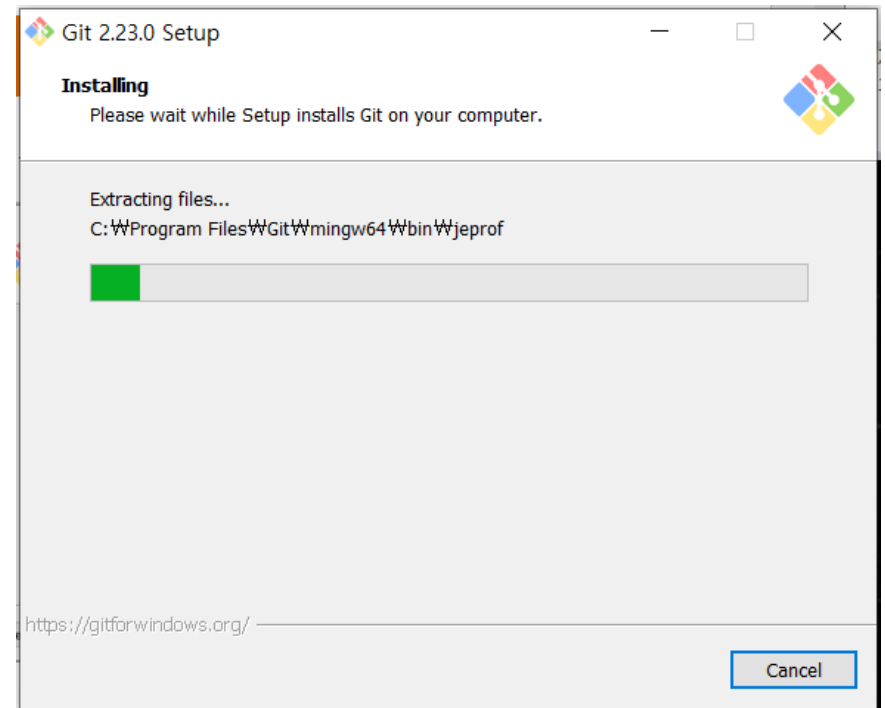
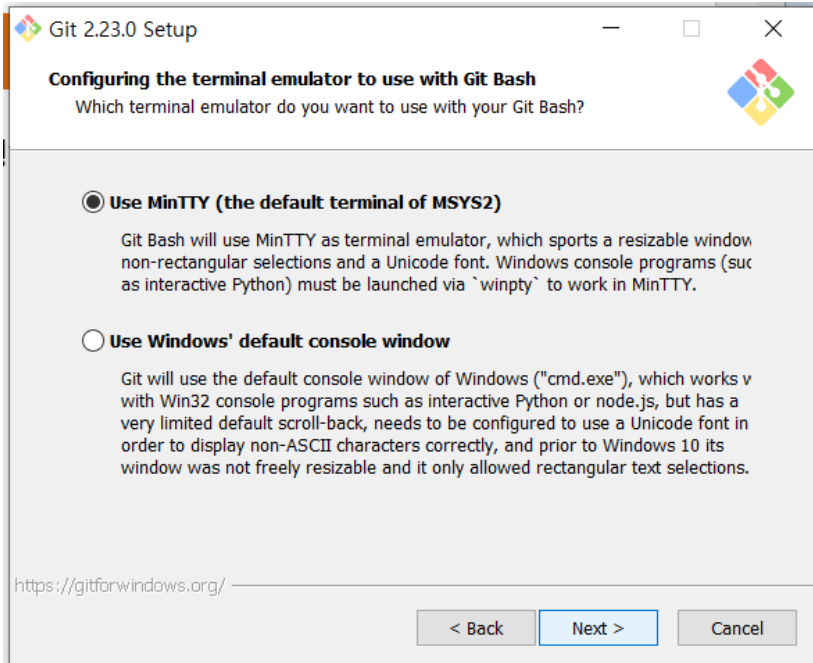


- ❖ 형상관리 도구를 설치한다.
- ✓ 다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.



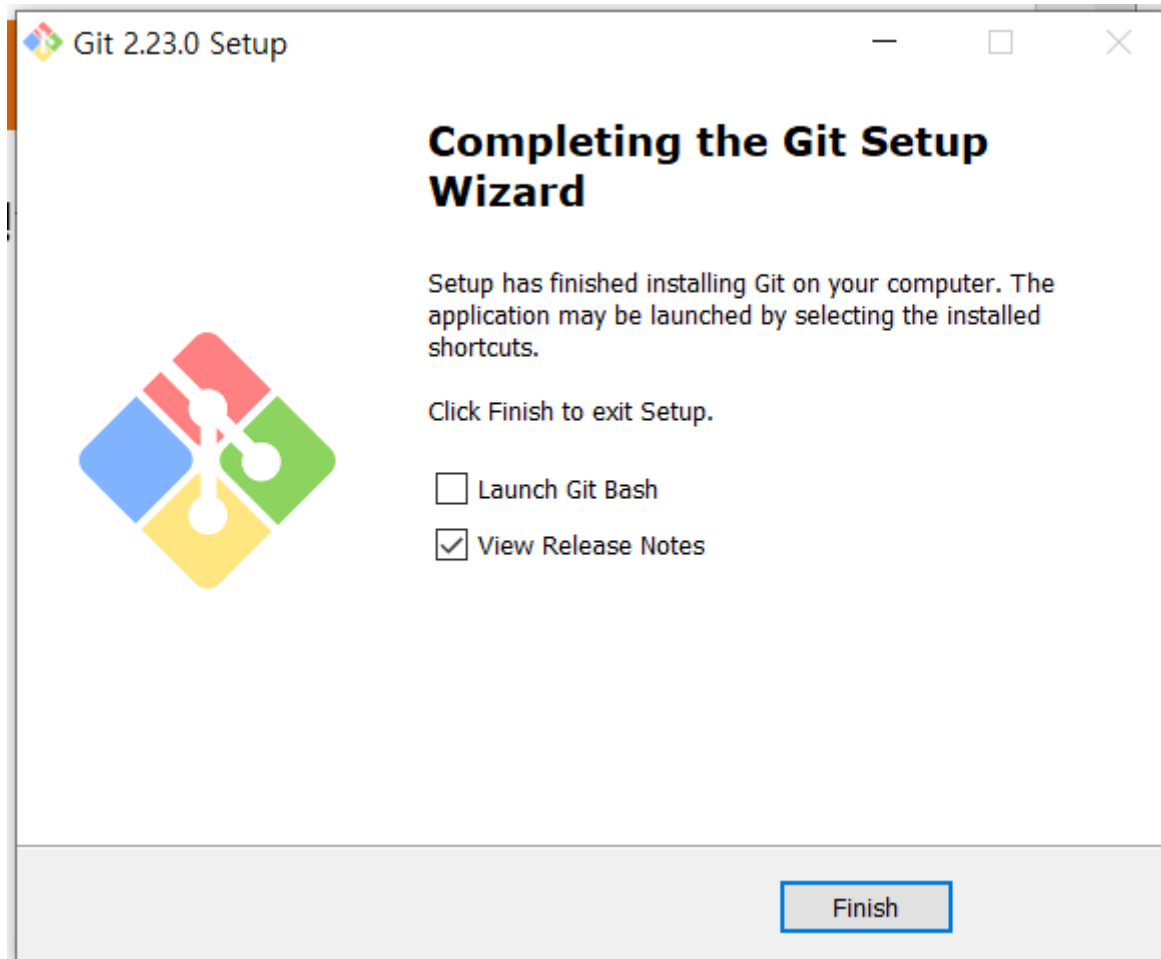
# Git

- ❖ 형상관리 도구를 설치한다.
- ✓ 다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.



# Git

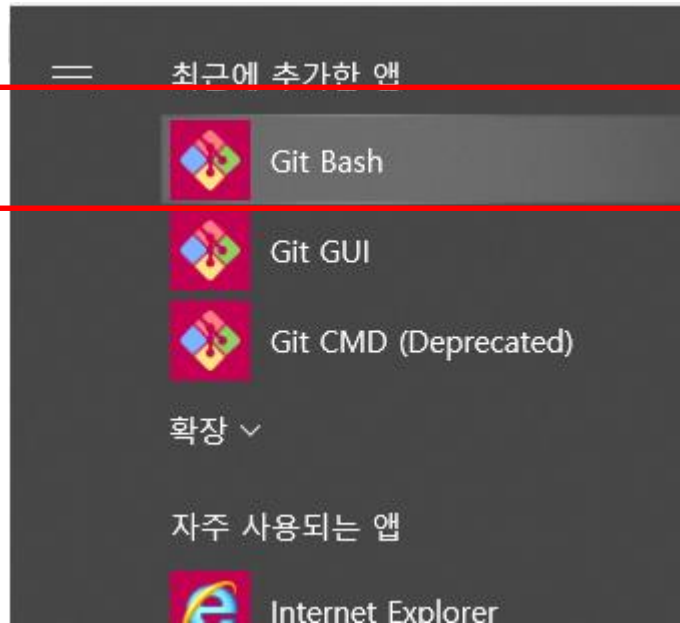
- ❖ 형상관리 도구를 설치한다.
- ✓ 다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.





# Git

## ✓ 설치 후 git 설정




MINGW64:/c/Users/jykim

```
jykim@DESKTOP-V3V85LD MINGW64 ~  
$
```

# Git

## ✓ 설치 후 git 설정

```
git config --global user.name "Your Name Here"
```

 MINGW64:/c/Users/jykim

```
jykim@DESKTOP-V3V85LD MINGW64 ~  
$ git config --global user.name "jykim61"  
jykim@DESKTOP-V3V85LD MINGW64 ~  
$
```

# Git

- ❖ 설치 후 git 설정
- ✓ 자신의 이메일을 등록한다.
- 본인의 이메일 주소를 입력한다.

```
git config --global user.email "your_email@youremail.com"
```

```
$ git config --global user.email "jykim61@hanmail.net"
```

```
jykim@DESKTOP-V3V85LD MINGW64 ~  
$
```

# Git

## ❖ 설치 후 git 설치 정보 확인.

- 본인의 이메일 주소를 입력한다.

```
jykim@DESKTOP-V3V85LD MINGW64 ~  
$ git config --list  
core.symlinks=false  
core.autocrlf=true  
core.fscache=true  
color.diff=auto  
color.status=auto  
color.branch=auto  
color.interactive=true  
help.format=html  
rebase.autosquash=true  
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt  
http.sslbackend=openssl  
diff.astextplain.textconv=astextplain  
filter.lfs.clean=git-lfs clean -- %f  
filter.lfs.smudge=git-lfs smudge --skip -- %f  
filter.lfs.process=git-lfs filter-process --skip  
filter.lfs.required=true  
credential.helper=manager  
difftool.sourcetree.cmd='' "$LOCAL" "$REMOTE"  
mergetool.sourcetree.cmd=''  
mergetool.sourcetree.trustexitcode=true  
user.name=jykim61  
user.email=jykim61@hanmail.net  
  
jykim@DESKTOP-V3V85LD MINGW64 ~  
$
```

## ❖ Project 생성(c:\project\test)

```
C:\> 명령 프롬프트
Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\jykim>cd c:/
c:\>mkdir project
c:\>cd project
c:\project>mkdir test
c:\project>cd test
c:\project\test>_
```

# Git

❖ Project 생성(c:\project\test)

✓ 깃 최초 실행 (로컬 저장소 생성)

```
git init
```

- git init은 initialize의 약자로 최초 초기화 명령어다.

\*최초 git을 사용하기위한 명령어다.

- 실행후에 폴더를 확인해도 아무런 변화가 없다.

\*실제로는 깃 설정파일이 숨김파일 형태로 저장되어 있다.

```
c:\project\test>git init
Initialized empty Git repository in c:/project/test/.git/

c:\project\test>
```

# Git

## ❖ Project 생성(c:\project\test)

✓ 현 프로젝트의 저장소 상태를 체크해보자.

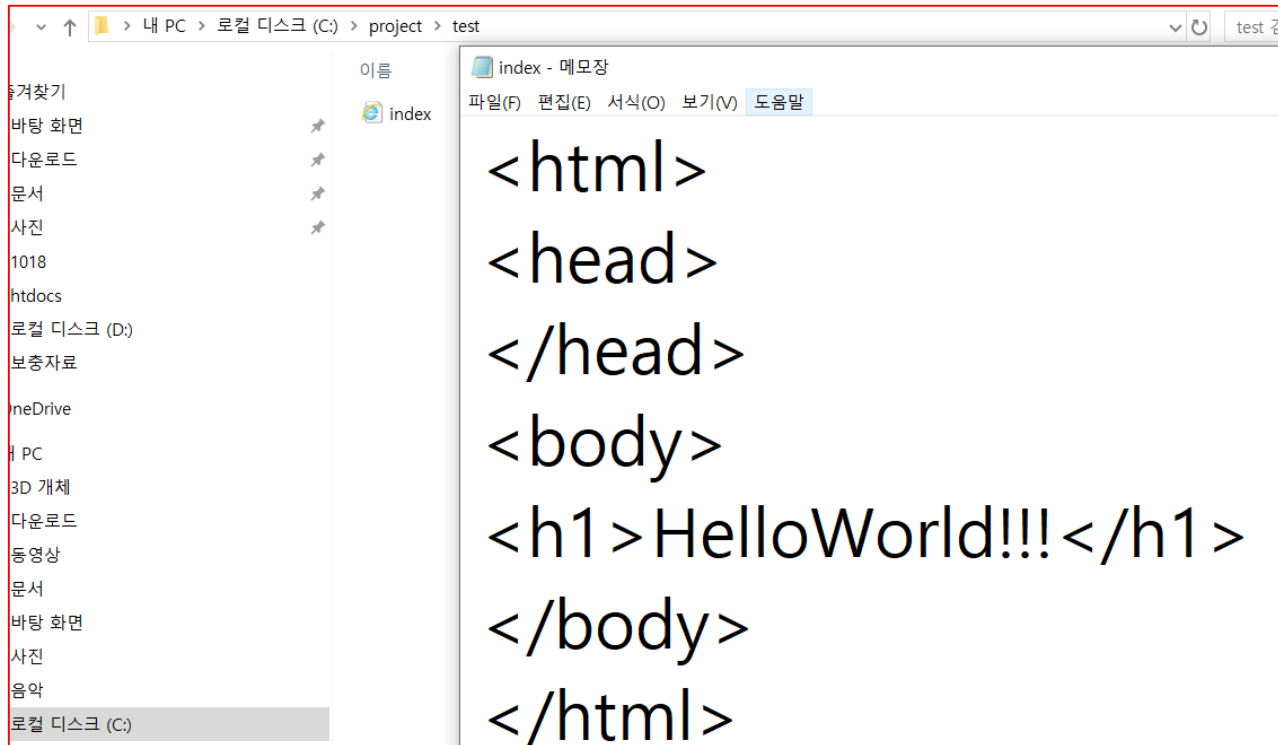
- git status를 실행하면 현재 저장소 상태를 파악할 수 있다.

```
c:\project\test>git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
c:\project\test>_
```

- ❖ Project 생성(c:\wproject\wtest)
- ✓ 현 프로젝트에 index.html 파일을 생성





## ❖ Project 생성(c:\project\test)

```
c:\project\test>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)
c:\project\test>
```

- \* 현재 프로젝트의 master branch 위치에 있다는 것을 알 수 있다.
- \* 현재 커밋된 내용이 없다는 것을 확인할 수 있다.
- \* 임의로 생성된 index.html을 추적할 수 없다고 출력된다. (이건 무슨소리일까?? 다음 내용을 보자)

# Git

## ❖ Project 생성(c:\project\test)

✓ 임의로 생성된 index.html을 Staging Area에 등록하자.

- git add를 사용한다.

- add를 활용하여 파일을 Staging Area에 추가할 수 있다.
- 실제 파일 자체가 복사되는 개념이 아니라 git이 해당 파일을 관리대상에 올린다는 의미다.

```
git add index.html
```

```
c:\project\test>git add index.html
```

```
c:\project\test>
```

## ❖ Project 생성(c:\project\test)

- ✓ Untracked 단계였던 index.html은 add 명령어를 통해 new file로 확인이 가능하다.

```
c:\project\test>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   index.html

c:\project\test>
```

## ❖ Project 생성(c:\project\test)

✓ 프로젝트를 커밋해보자.

```
git commit -m "Add index.html"
```

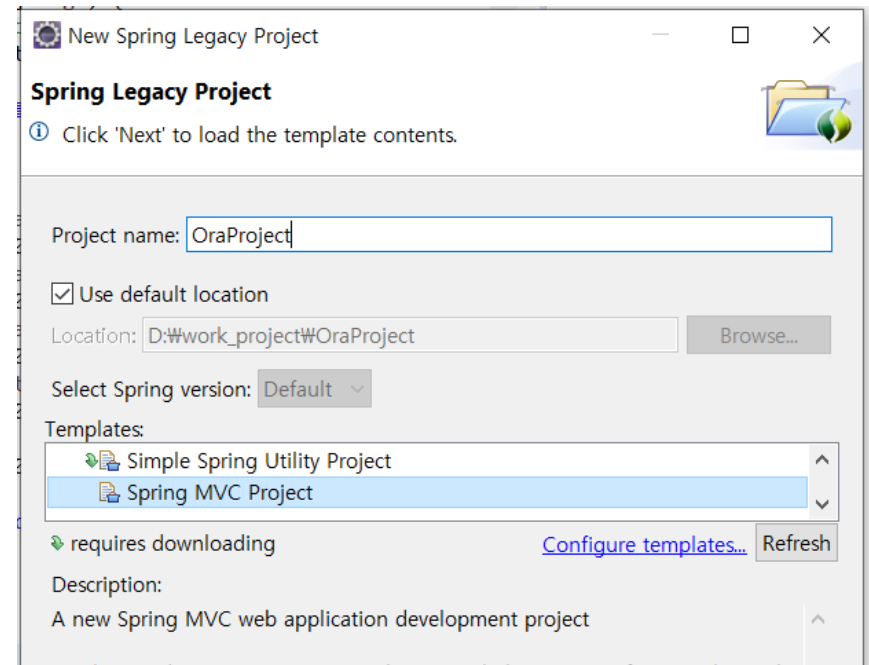
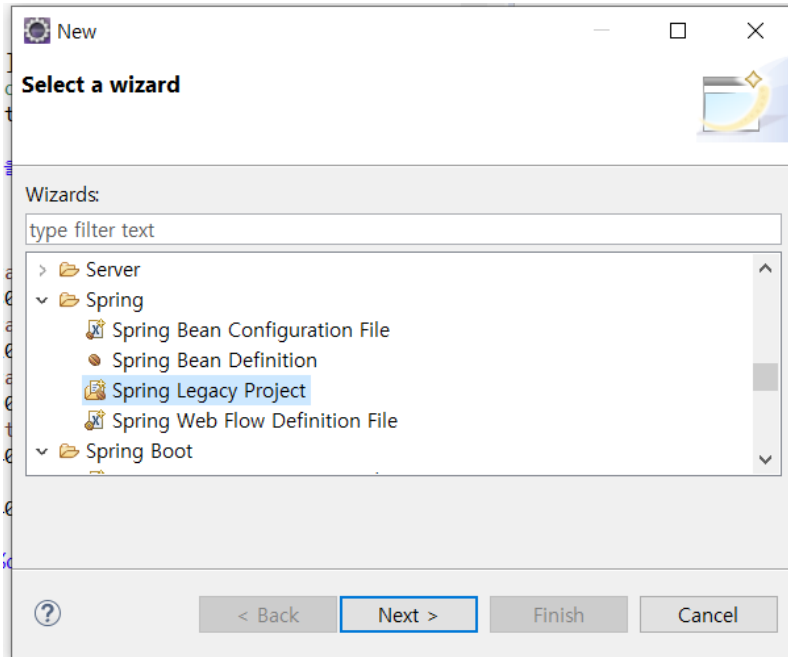
```
c:\project\test>git commit -m "Add index.html"
[master (root-commit) cfce0c7] Add index.html
1 file changed, 7 insertions(+)
create mode 100644 index.html

c:\project\test>
```

# [Spring] Spring + MyBatis + Oracle + JUnit 연동

## ❖ Project 생성

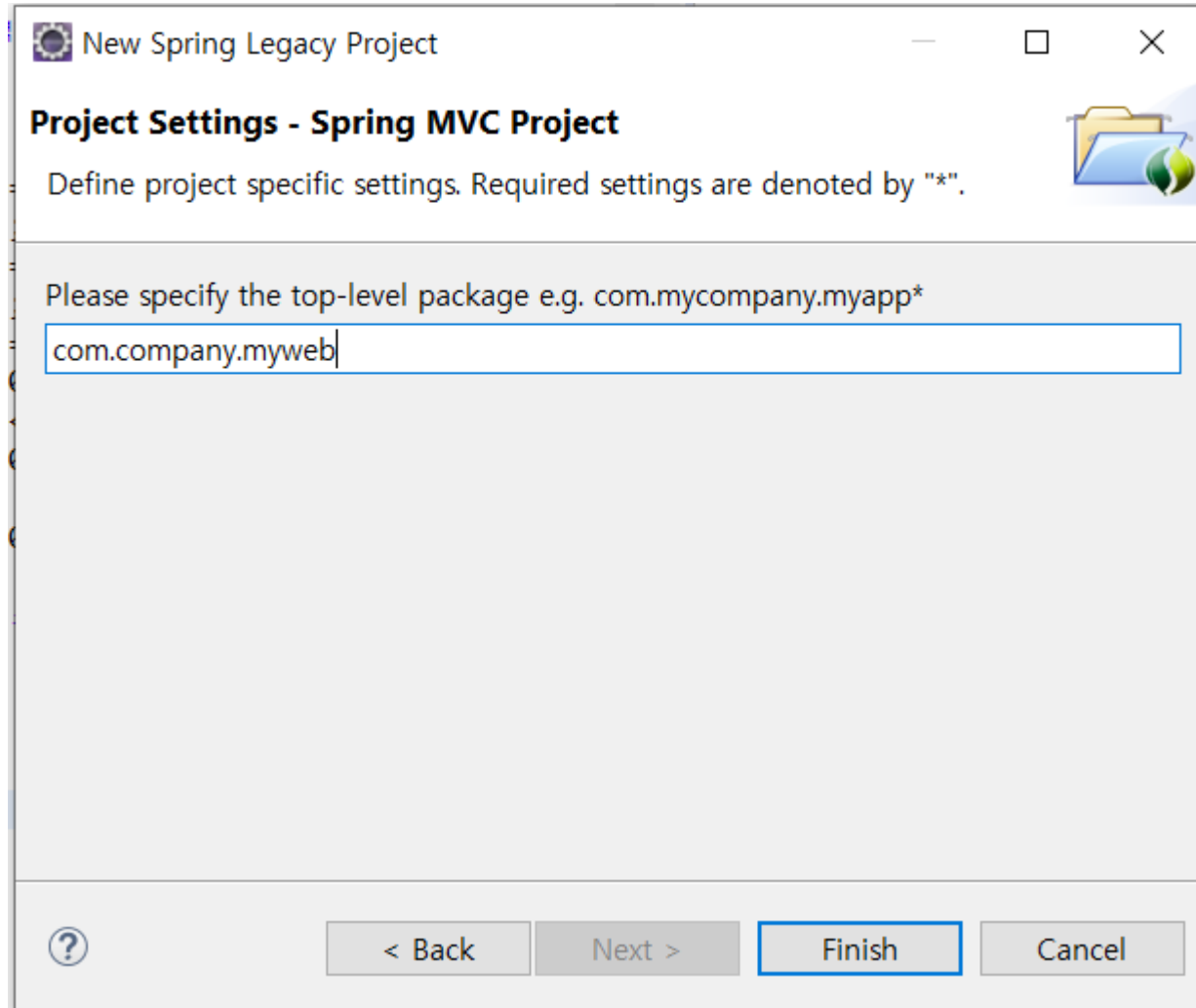
✓ new project -> Spring Legacy Project - Spring MVC 프로젝트 생성



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

### ❖ Project 생성

✓ new project -> Spring Legacy Project - Spring MVC 프로젝트 생성



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\*

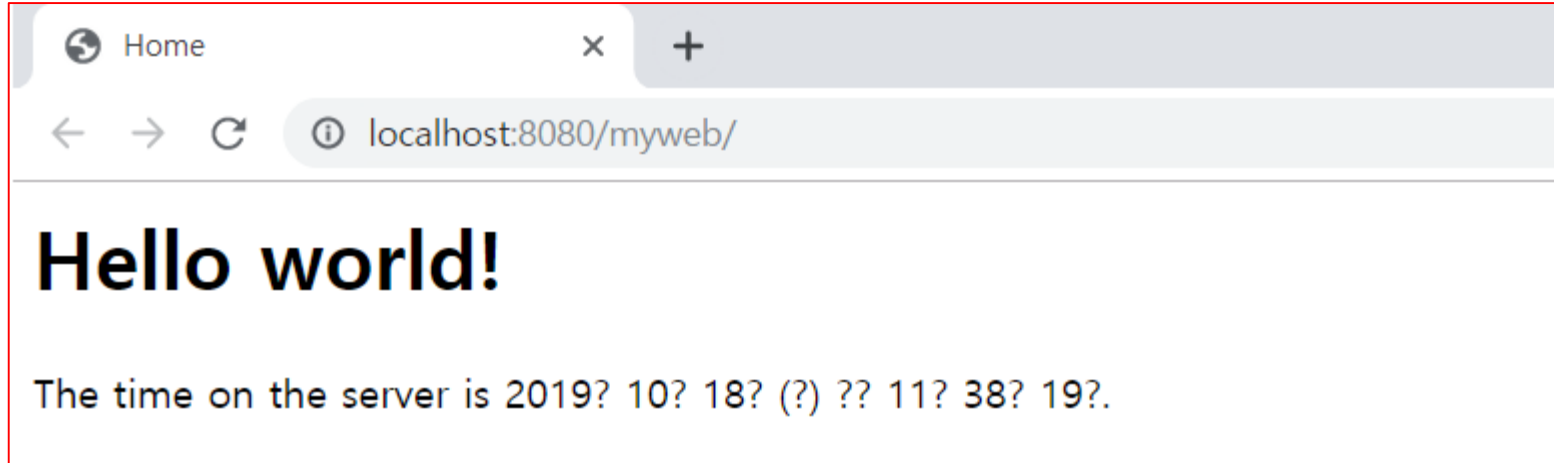
com.company.myweb

? < Back Next > Finish Cancel

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

### ❖ Project 생성

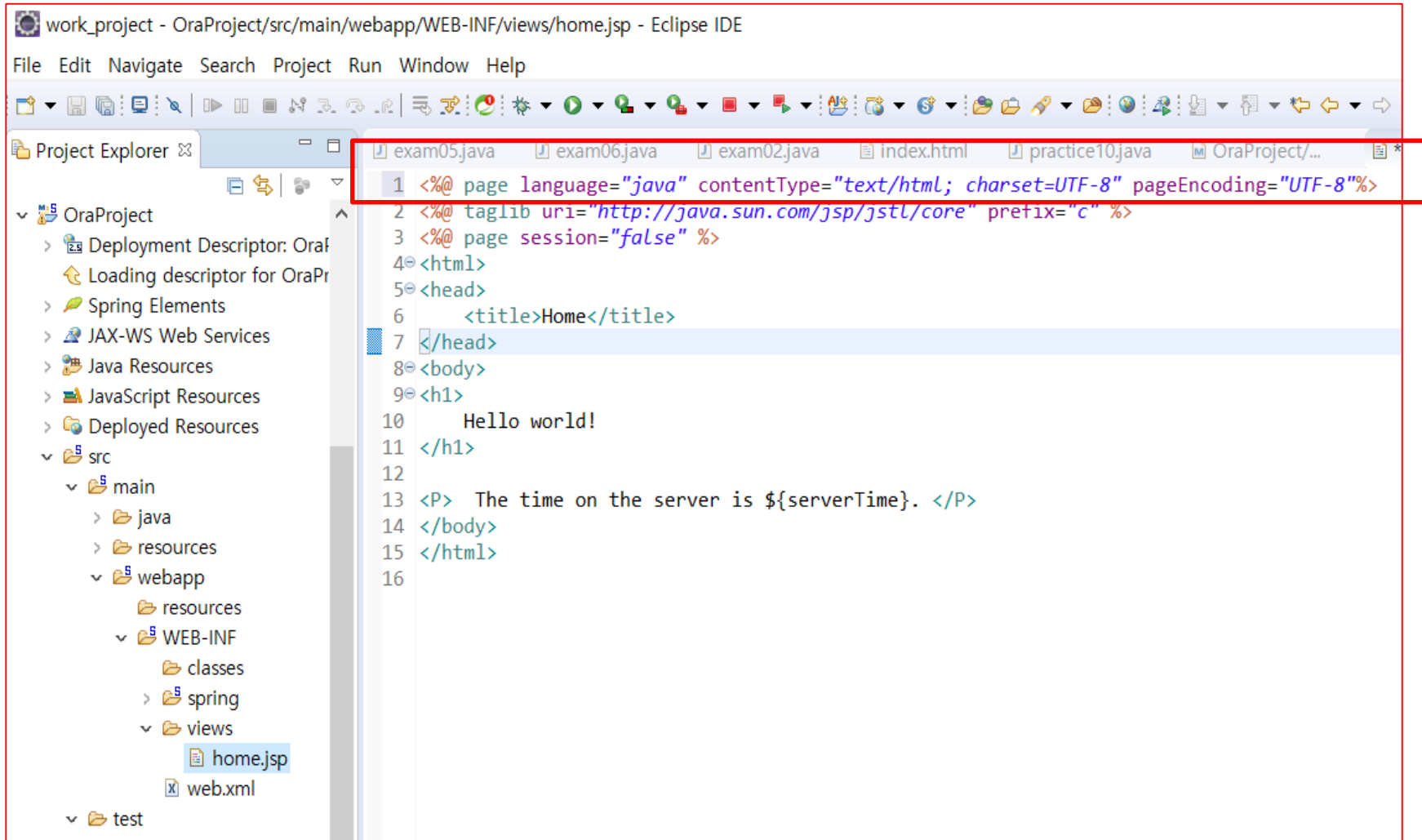
✓ 실행



# [Spring] Spring + MyBatis + Oracle + JUnit 연동

## ❖ Project 생성

✓ 실행(한글 깨지는 경우 처리)



```
work_project - OraProject/src/main/webapp/WEB-INF/views/home.jsp - Eclipse IDE
File Edit Navigate Search Project Run Window Help

Project Explorer
OraProject
  Deployment Descriptor: OraProject
  Loading descriptor for OraProject
  Spring Elements
  JAX-WS Web Services
  Java Resources
  JavaScript Resources
  Deployed Resources
  src
    main
      java
      resources
      webapp
        resources
        WEB-INF
          classes
          spring
          views
            home.jsp
            web.xml
    test

exam05.java exam06.java exam02.java index.html practice10.java OraProject/...
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
3 <%@ page session="false" %>
4 <html>
5 <head>
6   <title>Home</title>
7 </head>
8 <body>
9 <h1>
10   Hello world!
11 </h1>
12
13 <P> The time on the server is ${serverTime}. </P>
14 </body>
15 </html>
16
```



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

### ❖ Project 생성

#### [JDK 버전의 처리]

- ✓ STS를 이용해서 MVC 프로젝트를 생성하게 되면 Java 1.6 버전을 기준으로 생성
- ✓ 프로젝트 오른쪽 클릭 - Properties에서  
Java Compiler, Project Facets 에 가서 1.6 -> 1.8로 수정
- ✓ pom.xml에 `<java-version>1.8</java-version>`로 수정

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

### ❖ Project 생성

[Spring 프레임워크 버전 변경]

- ✓ pom.xml에 `<org.springframework-version>4.3.8. RELEASE</org.springframework-version>` 로 수정



```
3  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/mave
4  <modelVersion>4.0.0</modelVersion>
5  <groupId>com.company</groupId>
6  <artifactId>myweb</artifactId>
7  <name>OraProject</name>
8  <packaging>war</packaging>
9  <version>1.0.0-BUILD-SNAPSHOT</version>
10 <properties>
11     <java-version>1.8</java-version>
12     <org.springframework-version>4.3.8.RELEASE</org.springframework-version>
13     <org.aspectj-version>1.6.10</org.aspectj-version>
14     <org.slf4j-version>1.6.6</org.slf4j-version>
15 </properties>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ pom.xml에 추가

```
<!-- spring-jdbc -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
    <scope>test</scope>
</dependency>

<!-- spring-test -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-test</artifactId>
    <version>${org.springframework-version}</version>
    <scope>test</scope>
</dependency>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ pom.xml에 추가

```
<!-- log4jdbc-log4j2-jdbc4 -->
    <dependency>
        <groupId>org.bgee.log4jdbc-log4j2</groupId>
        <artifactId>log4jdbc-log4j2-jdbc4</artifactId>
        <version>1.16</version>
    </dependency>

<!-- org.mybatis/mybatis -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis</artifactId>
        <version>3.4.1</version>
    </dependency>

<!-- mybatis-spring -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>1.3.0</version>
    </dependency>

</dependencies>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ pom.xml에 추가
- ✓ Oracle Driver Maven 설정 시 repository를 별도로 지정

```
4      <modelVersion>4.0.0</modelVersion>
5      <groupId>com.company</groupId>
6      <artifactId>myweb</artifactId>
7      <name>OraProject</name>
8      <packaging>war</packaging>
9      <version>1.0.0-BUILD-SNAPSHOT</version>
10     <properties>
11         <java-version>1.8</java-version>
12         <org.springframework-version>4.3.7.RELEASE</org.springframework-version>
13         <org.aspectj-version>1.6.10</org.aspectj-version>
14         <org.slf4j-version>1.6.6</org.slf4j-version>
15     </properties>
16     <!--dependencies 위에 설정 -->
17     <repositories>
18         <repository>
19             <id>oracle</id>
20             <name>ORACLE JDBC Repository</name>
21             <url>https://code.lds.org/nexus/content/groups/main-repo</url>
22         </repository>
23     </repositories>
24
25     <dependencies>
26         <!-- Spring -->
27         <dependency>
28             <groupId>org.springframework</groupId>
29             <artifactId>spring-context</artifactId>
30             <version>${org.springframework-version}</version>
31             <exclusions>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

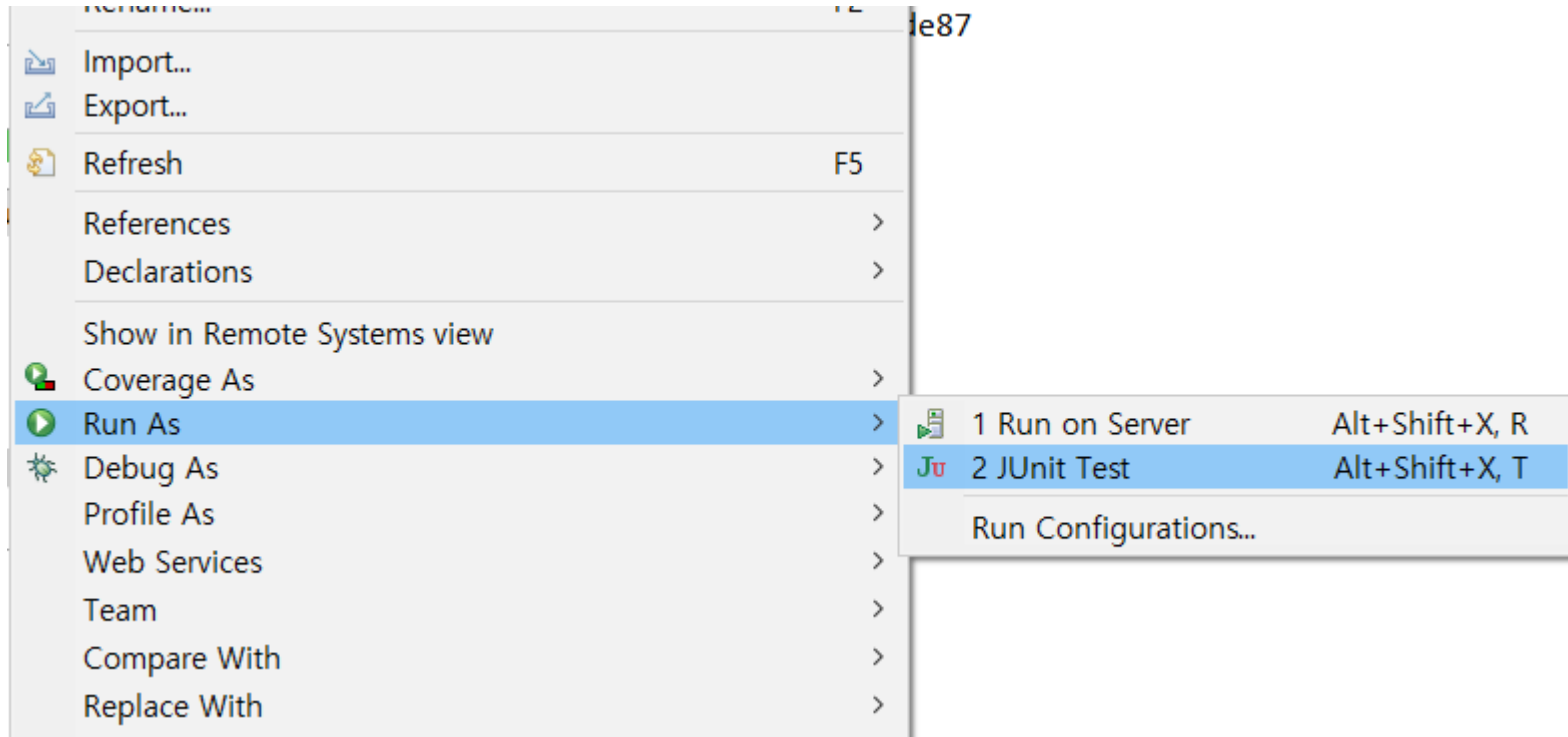
### [JUnit 설정]

- ✓ 이제 Oracle과 제대로 연동이 되었는지 테스트를 하기 위해 src/test/java 폴더 아래에 OracleConnectionTest.java 를 생성하고 아래의 코드를 입력

```
import java.sql.Connection;
import java.sql.DriverManager;
import org.junit.Test;
public class OracleConnectionTest {
    private static final String DRIVER = "oracle.jdbc.driver.OracleDriver";
    private static final String URL = "jdbc:oracle:thin:@localhost:1521:orcl";
    private static final String USER = "scott";
    private static final String PW = "1234";
    @Test
    public void testConnect() throws Exception{
        Class.forName(DRIVER);
        try(Connection con = DriverManager.getConnection(URL, USER, PW)){
            System.out.println(con);
            System.out.println("오라클 연결 성공");
        }catch(Exception e) {
            System.out.println("오라클 연결 실패");
            e.printStackTrace();
        }
    }
}
```

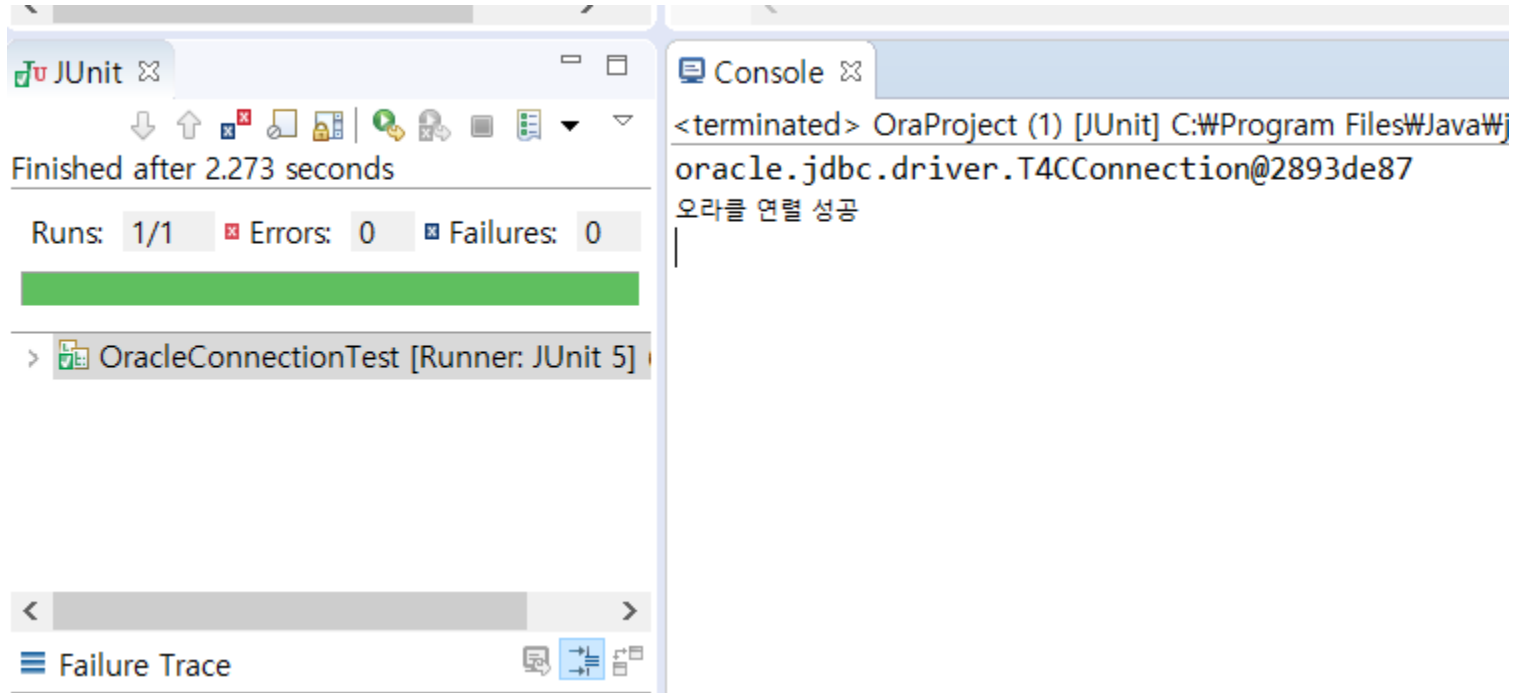
## [Spring] Spring + MyBatis + Oracle + JUnit 연동

[단위 테스트] Run As -> JUnit Test 하면 Console창에



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

[단위 테스트] Run As -> JUnit Test 하면 Console창에





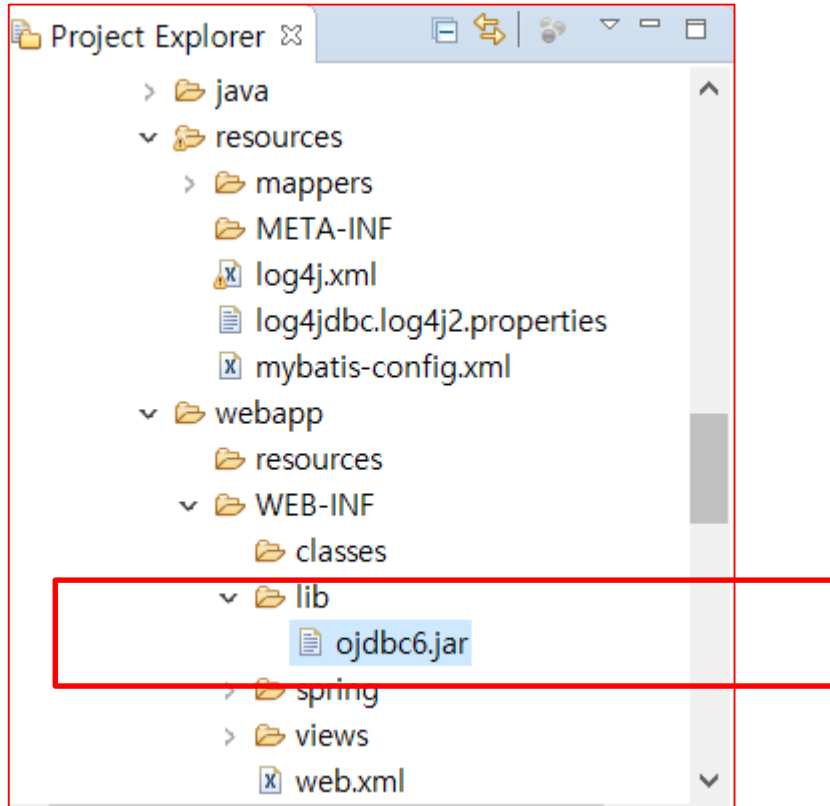
## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/resources에 “ log4jdbc.log4j2.properties ” 파일 생성



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/webapp/WEB-INF에 lib Folder를 하나 만들고 oracle 드라이버 로드



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/webapp/WEB-INF/spring/root-context.xml을 열고
- ✓ 아래에 Namespaces 클릭한 후 빨간박스에 있는 것들을 체크
- ✓ 그리고 Namespaces 왼쪽에 Source탭에 들어갑니다.

The screenshot shows an IDE interface with the following components:

- Left Sidebar (Project Explorer):** Displays the project structure. The path `src/main/webapp/WEB-INF/spring/root-context.xml` is selected.
- Main Window (Configure Namespaces):** A dialog titled "Namespaces 1 warning" with a sub-header "Configure Namespaces". It instructs to "Select XSD namespaces to use in the configuration file". A list of namespaces is shown with checkboxes:
  - ☐ aop - <http://www.springframework.org/schema/aop>
  - ☒ beans - <http://www.springframework.org/schema/beans>
  - ☐ c - <http://www.springframework.org/schema/c>
  - ☐ cache - <http://www.springframework.org/schema/cache>
  - ☒ context - <http://www.springframework.org/schema/context>
  - ☒ jdbc - <http://www.springframework.org/schema/jdbc>
  - ☐ jee - <http://www.springframework.org/schema/jee>
  - ☐ lang - <http://www.springframework.org/schema/lang>
  - ☐ mvc - <http://www.springframework.org/schema/mvc>
  - ☒ mybatis-spring - <http://mybatis.org/schema/mybatis-spring>
  - ☐ p - <http://www.springframework.org/schema/p>
  - ☐ task - <http://www.springframework.org/schema/task>
  - ☐ tx - <http://www.springframework.org/schema/tx>
- Bottom Status Bar:** Shows "Finished after 1.13 seconds" and a tab bar with "Source", "Namespaces", "Overview", "beans", "context", and "jdbc". The "Source" tab is currently selected.

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ 그리고 Namespaces 왼쪽에 Source탭에 들어갑니다.
- ✓ 자신의 오라클 아이디와 비밀번호를 입력

```
11
12 <!-- Root Context: defines shared resources visible to all other web components -->
13 <!-- 오라클 접속 -->
14 <bean class="org.springframework.jdbc.datasource.DriverManagerDataSource" id="dataSource">
15     <property name="driverClassName" value="net.sf.log4jdbc.sql.jdbcapi.DriverSpy"/>
16     <property name="url" value="jdbc:log4jdbc:oracle:thin:@localhost:1521/orcl"/>
17     <property name="username" value="scott"/>
18     <property name="password" value="1234"/>
19 </bean>
20
21 <!-- Mybatis 연동 -->
22 <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
23     <property name="dataSource" ref="dataSource"></property>
24     <property name="configLocation" value="classpath:/mybatis-config.xml"></property>
25     <property name="mapperLocations" value="classpath:mappers/**/*.xml"/>
26 </bean>
27 <bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate" destroy-method="clearCache">
28     <constructor-arg name="sqlSessionFactory" ref="sqlSessionFactory"></constructor-arg>
29 </bean>
30 </beans>
31
```

er 1.13 seconds

Source Namespaces Overview beans context jdbc

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ root-context.xml

<!-- 오라클 접속 -->

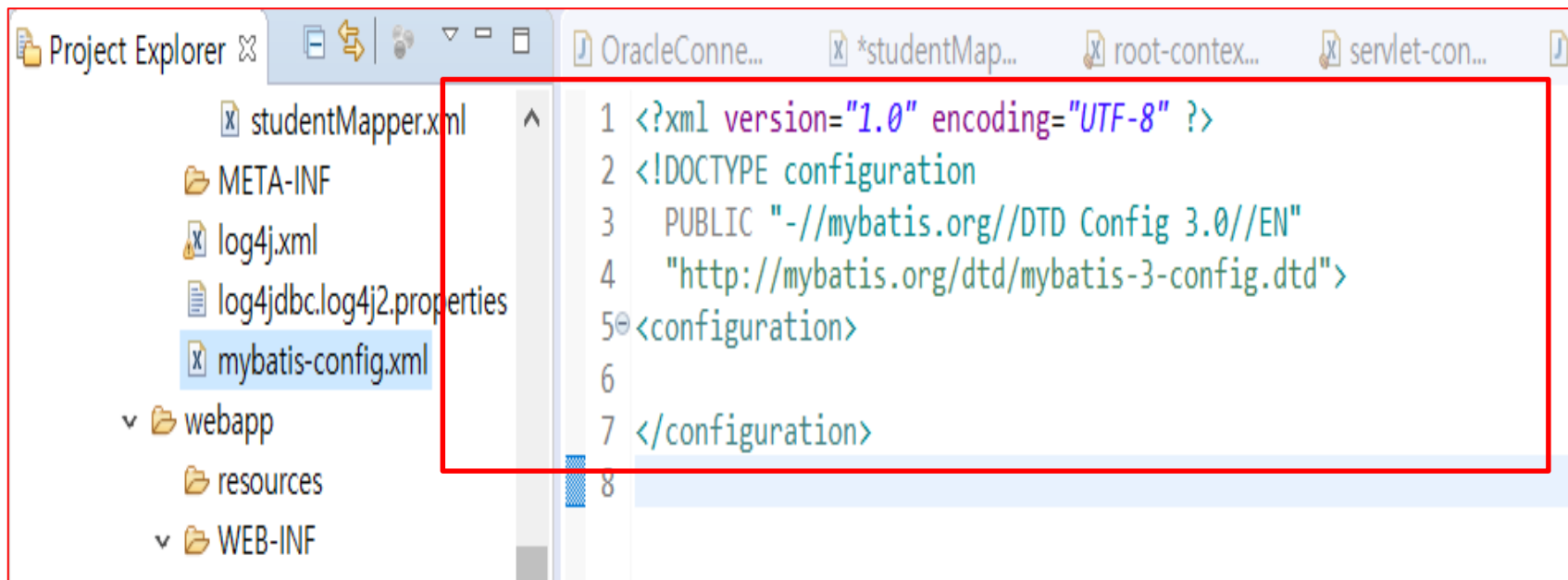
```
<bean class="org.springframework.jdbc.datasource.DriverManagerDataSource" id="dataSource">
    <property name="driverClassName" value="net.sf.log4jdbc.sql.jdbcapi.DriverSpy"/>
    <property name="url" value="jdbc:log4jdbc:oracle:thin:@localhost:1521/orcl"/>
    <property name="username" value="scott"/>
    <property name="password" value="1234"/>
</bean>
```

<!-- Mybatis 연동 -->

```
<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="dataSource" ref="dataSource"></property>
    <property name="configLocation" value="classpath:/mybatis-config.xml"></property>
    <property name="mapperLocations" value="classpath:mappers/**/*.xml"/>
</bean>
<bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate" destroy-method="clearCache">
    <constructor-arg name="sqlSessionFactory" ref="sqlSessionFactory"></constructor-arg>
</bean>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/resources에 mybatis-config.xml 파일을 만들어주고 아래 코드를 추가



```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<!DOCTYPE configuration
```

```
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
```

```
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
```

```
<configuration>
```

```
</configuration>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ src/test/java에 Test.java파일을 생성 후 아래 코드를 추가

```
import java.sql.Connection;
```

```
import javax.inject.Inject;
```

```
import javax.sql.DataSource;
```

```
import org.apache.ibatis.session.SqlSession;
```

```
import org.apache.ibatis.session.SqlSessionFactory;
```

```
import org.junit.runner.RunWith;
```

```
import org.springframework.test.context.ContextConfiguration;
```

```
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
```

```
@RunWith(SpringJUnit4ClassRunner.class)
```

```
@ContextConfiguration(locations = {"file:src/main/webapp/WEB-INF/spring/**/root-  
context.xml"})
```

```
public class Test {
```

```
    @Inject
```

```
    private DataSource ds;
```

```
    @Inject
```

```
    private SqlSessionFactory sqlFactory;
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ src/test/java에 Test.java파일을 생성 후 아래 코드를 추가

```
@org.junit.Test
public void test() throws Exception{

    try(Connection conn = ds.getConnection()){
        System.out.println(conn);
    } catch(Exception e){
        e.printStackTrace();
    }
}

@org.junit.Test
public void factoryTest() {
    System.out.println(sqlFactory);
}

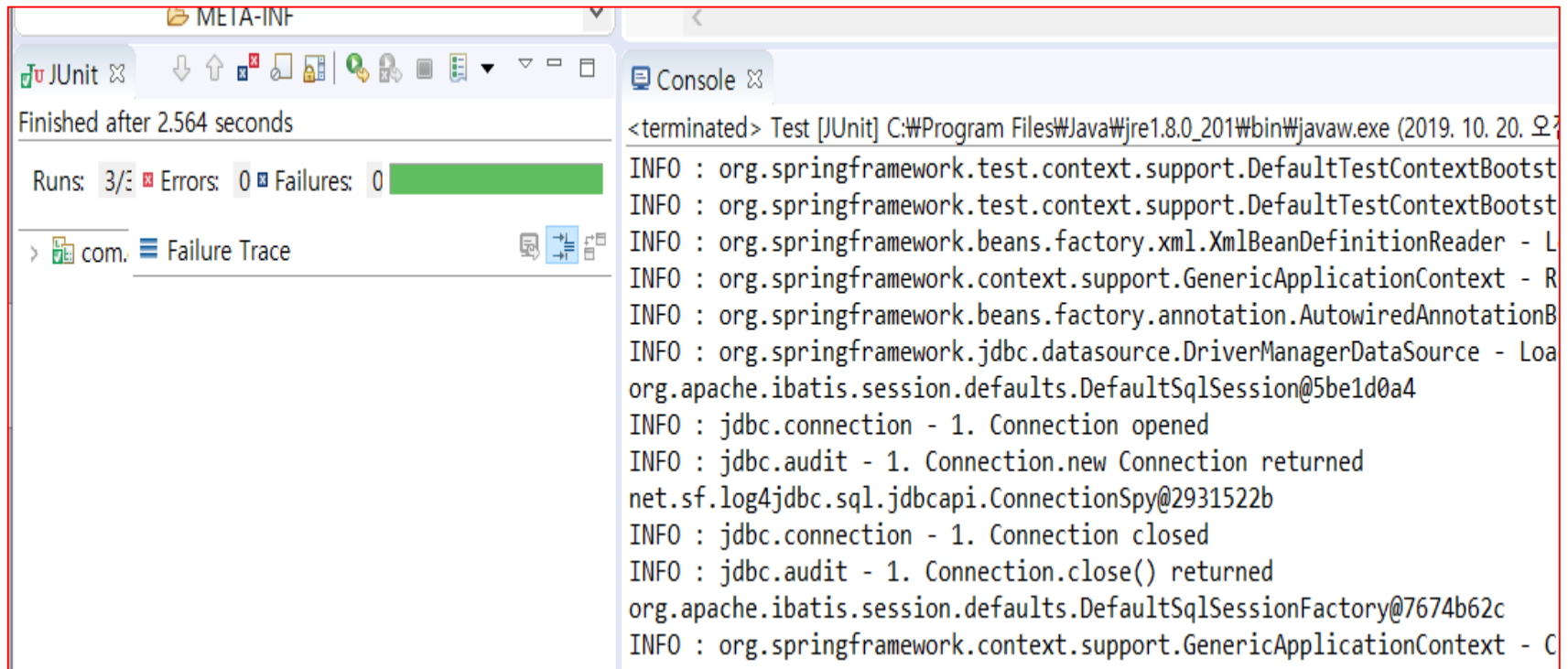
@org.junit.Test
public void sessionTest() throws Exception{

    try(SqlSession session = sqlFactory.openSession()) {
        System.out.println(session);
    }catch(Exception e) {
        e.printStackTrace();
    }
}
}
```



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ 이렇게 Junit가 녹색으로 나온다면 성공



The screenshot displays an IDE interface with two main panels. The left panel, titled 'JUnit', shows the test results: 'Finished after 2.564 seconds', 'Runs: 3/3', 'Errors: 0', and 'Failures: 0'. A green progress bar is visible next to the 'Failures' count. Below this, a tree view shows a package named 'com.' with a 'Failure Trace' icon. The right panel, titled 'Console', shows a log of messages from a Spring application context. The messages include information about the context bootstrapping, bean definition reading, and JDBC connection management. The log ends with the message 'INFO : org.springframework.context.support.GenericApplicationContext - C'.

```
<terminated> Test [JUnit] C:\Program Files\Java\jre1.8.0_201\bin\javaw.exe (2019. 10. 20. 오후 2:56:44)
INFO : org.springframework.test.context.support.DefaultTestContextBootstrapper - Loading
INFO : org.springframework.test.context.support.DefaultTestContextBootstrapper - Loading
INFO : org.springframework.beans.factory.xml.XmlBeanDefinitionReader - Loading
INFO : org.springframework.context.support.GenericApplicationContext - Refreshing
INFO : org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor -
INFO : org.springframework.jdbc.datasource.DriverManagerDataSource - Loading
org.apache.ibatis.session.defaults.DefaultSqlSession@5be1d0a4
INFO : jdbc.connection - 1. Connection opened
INFO : jdbc.audit - 1. Connection.new Connection returned
net.sf.log4jdbc.sql.jdbcapi.ConnectionSpy@2931522b
INFO : jdbc.connection - 1. Connection closed
INFO : jdbc.audit - 1. Connection.close() returned
org.apache.ibatis.session.defaults.DefaultSqlSessionFactory@7674b62c
INFO : org.springframework.context.support.GenericApplicationContext - C
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ Oracle에 Student table 생성

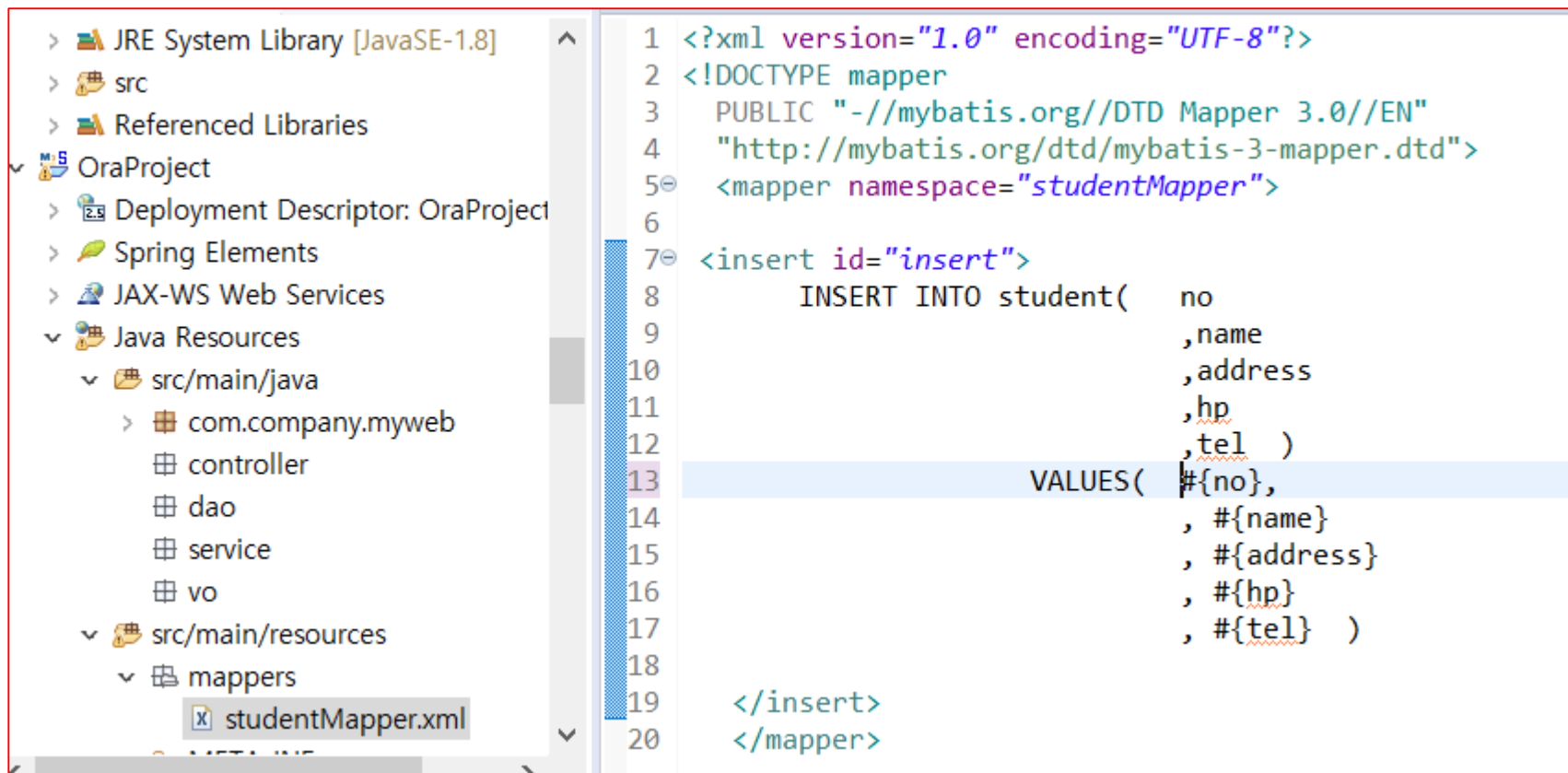
```
SQL> create table student(no varchar2(10) not null primary key,  
2  names varchar2(20) not null,  
3  address varchar2(80),  
4  hp varchar2(13),  
5  tel varchar2(13));
```

테이블이 생성되었습니다.

```
SQL>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/resources/mappers에 studentMapper.xml를 생성하여 코드를 작성



The screenshot shows an IDE with a project structure on the left and the content of a file named `studentMapper.xml` on the right.

**Project Structure (Left):**

- JRE System Library [JavaSE-1.8]
- src
- Referenced Libraries
- OraProject
  - Deployment Descriptor: OraProject
  - Spring Elements
  - JAX-WS Web Services
  - Java Resources
    - src/main/java
      - com.company.myweb
        - controller
        - dao
        - service
        - vo
      - src/main/resources
        - mappers
          - studentMapper.xml

**studentMapper.xml Content (Right):**

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE mapper
3   PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4   "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5 <mapper namespace="studentMapper">
6
7 <insert id="insert">
8     INSERT INTO student(    no
9                             ,name
10                            ,address
11                           ,hp
12                          ,tel )
13 VALUES( #{no},
14          , #{name}
15          , #{address}
16          , #{hp}
17          , #{tel} )
18
19 </insert>
20 </mapper>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ studentMapper.xml

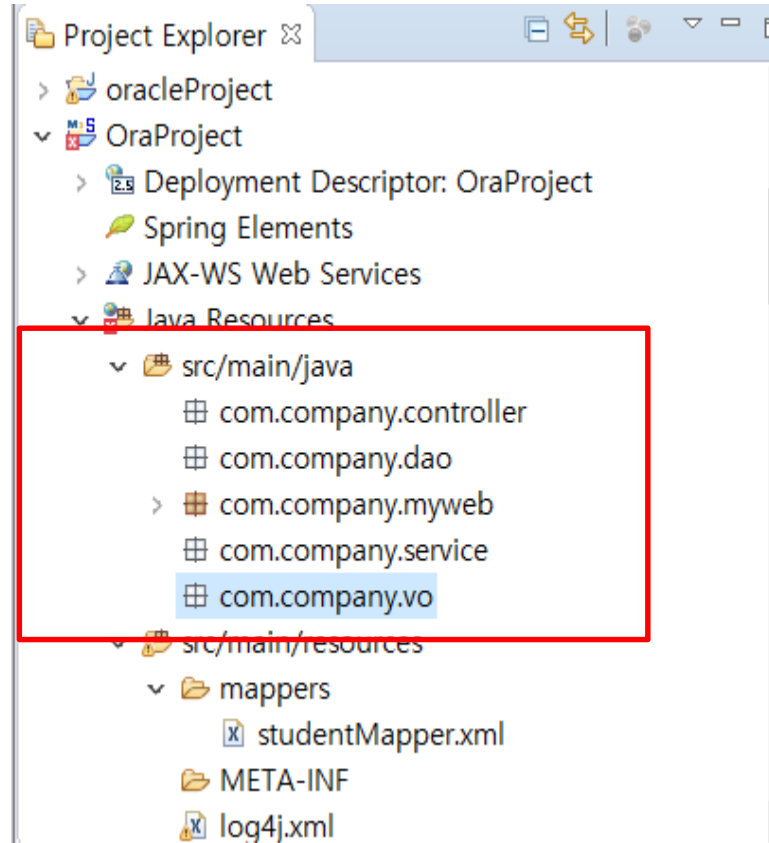
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="studentMapper">

  <insert id="insert">
    INSERT INTO student(  no
                        ,name
                        ,address
                        ,hp
                        ,tel  )
    VALUES(  #{no}
            , #{name}
            , #{address}
            , #{hp}
            , #{tel}  )

  </insert>
</mapper>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

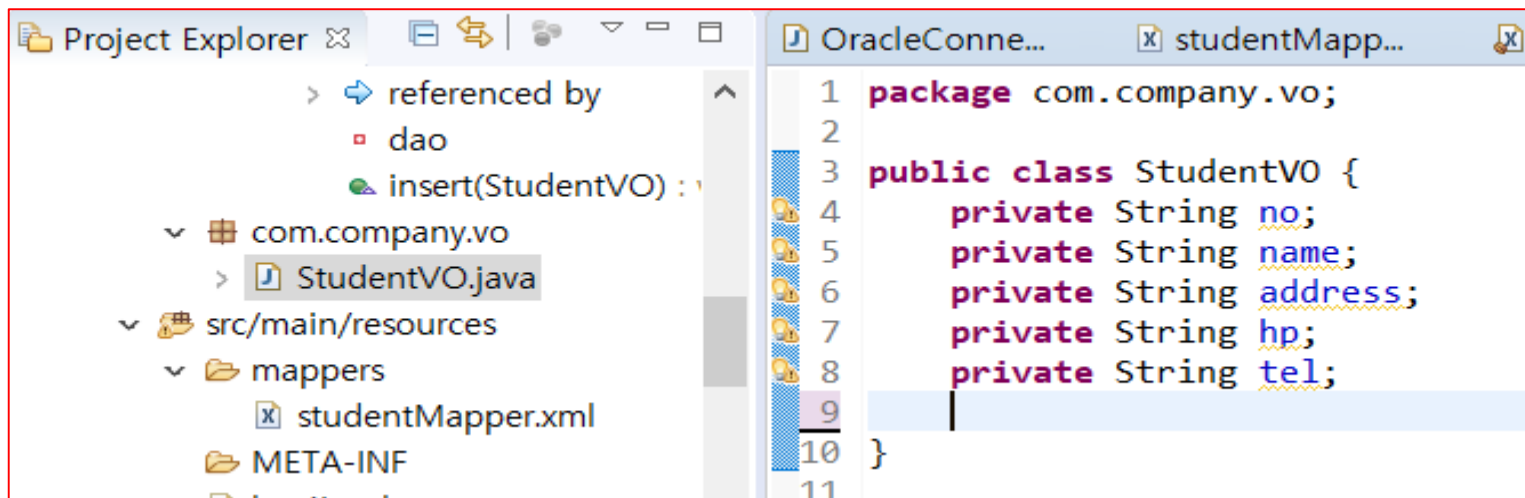
- ✓ src/main/java 밑에 다음과 같이 패키지 폴더를 생성



- ✓ 여기서 controller의 역할은 웹에서 처리해야 할 데이터를 받고, 이 데이터를 담당할 service를 선택하여 호출
- ✓ 그리고 처리한 데이터를 다음 페이지에서 볼 수 있게 셋팅하며 이동할 페이지를 리턴
- ✓ service는 데이터를 dao를 통해 넘겨주거나 받으면서 비즈니스 로직을 수행하는 역할
- ✓ dao는 DB를 통해 데이터를 조회하거나 수정 삭제 하는 역할을 합니다.
- ✓ vo는 DB에 있는 테이블 컬럼 값을 java에서 객체로 다루기 위해 사용

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

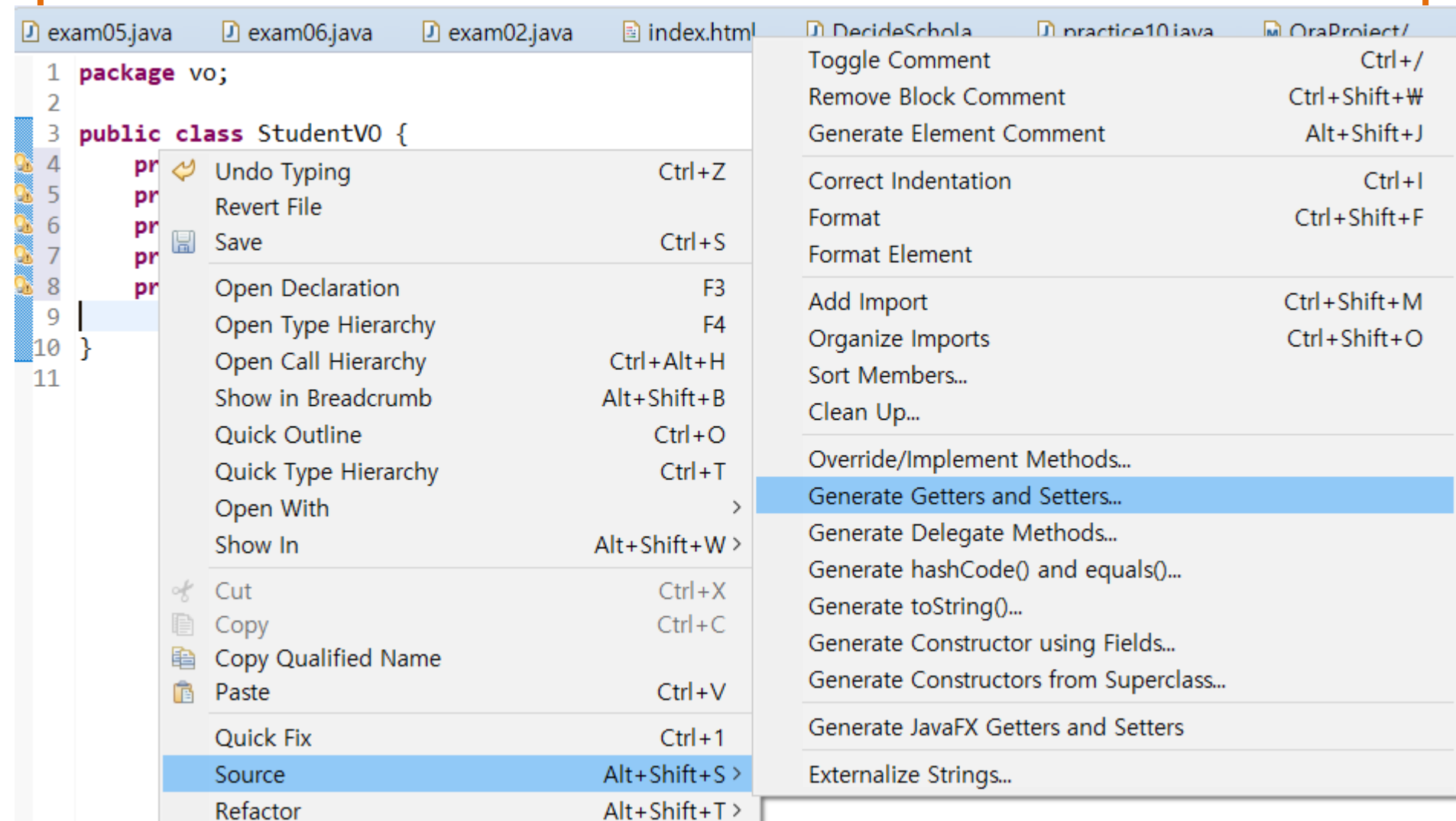
- ✓ vo에 StudentVO.java 파일을 만들어주고 다음과 같이 코드를 작성



```
public class StudentVO {
    private String no;
    private String name;
    private String address;
    private String hp;
    private String tel;
}
```

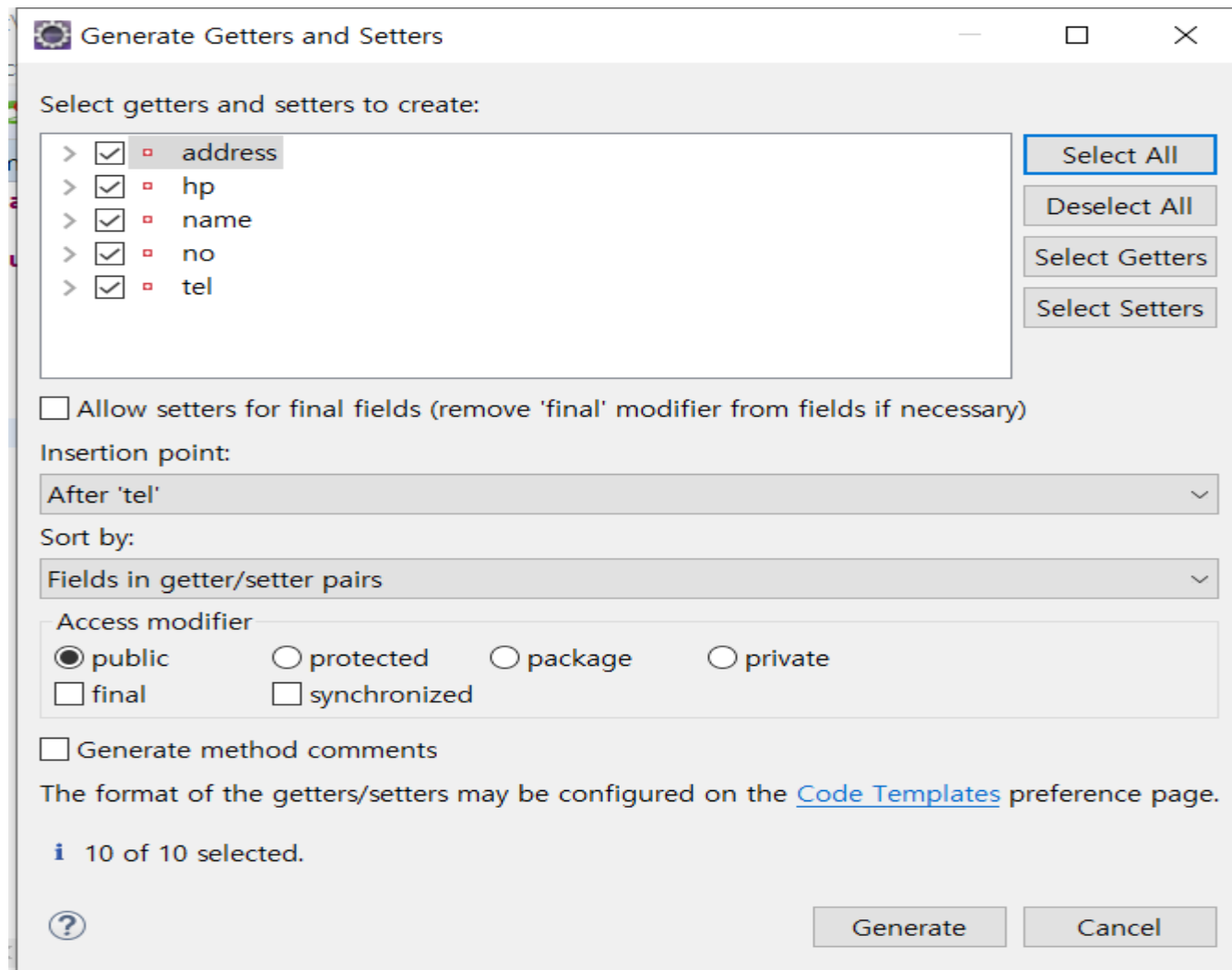
## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ 그리고 StudentV0안쪽 아래에 Alt + Shift + s를 누르거나 오른쪽 버튼 > Source에 Generate Getters and Setters에 들어가서 Setter & Getter를 만듦



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ 그리고 StudentVO안쪽 아래에 Alt + Shift + s를 누르거나 오른쪽 버튼 > Source에 Generate Getters and Setters에 들어가서 Setter & Getter를 만듦





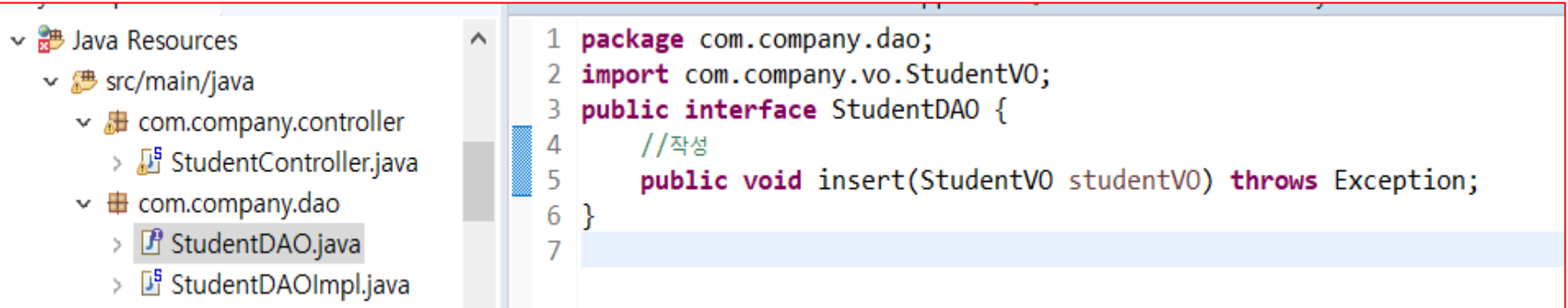
## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ 그리고 StudentVO안쪽 아래에 Alt + Shift + s를 누르거나 오른쪽 버튼 > Source에 Generate Getters and Setters에 들어가서 Setter & Getter를 만듦

```
3 public class StudentVO {
4     private String no;
5     private String name;
6     private String address;
7     private String hp;
8     private String tel;
9     public String getNo() {
10         return no;
11     }
12     public void setNo(String no) {
13         this.no = no;
14     }
15     public String getName() {
16         return name;
17     }
18     public void setName(String name) {
19         this.name = name;
20     }
21     public String getAddress() {
22         return address;
23     }
24     public void setAddress(String address) {
25         this.address = address;
26     }
27     public String getHp() {
28         return hp;
29     }
30     public void setHp(String hp) {
31         this.hp = hp;
32     }
33     public String getTel() {
34         return tel;
35     }
36     public void setTel(String tel) {
37         this.tel = tel;
38     }
39 }
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ dao에 StudentDAO.java만들어서 코드를 작성해 줍니다.



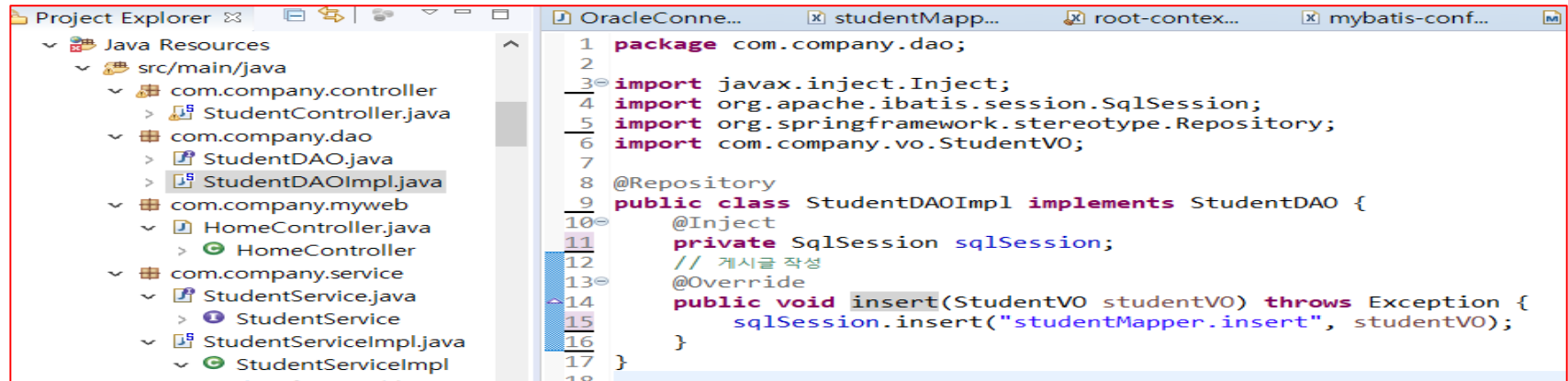
The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes 'Java Resources' with sub-packages 'src/main/java', 'com.company.controller', and 'com.company.dao'. The 'com.company.dao' package contains 'StudentDAO.java' and 'StudentDAOImpl.java'. The code editor shows the following code for 'StudentDAO.java':

```
1 package com.company.dao;
2 import com.company.vo.StudentVO;
3 public interface StudentDAO {
4     //작성
5     public void insert(StudentVO studentVO) throws Exception;
6 }
7
```

```
import com.company.vo.StudentVO;
public interface StudentDAO {
    //작성
    public void insert(StudentVO studentVO) throws Exception;
}
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ dao에 StudentDAOImpl.java 을 만들어서 코드를 작성해 줍니다.

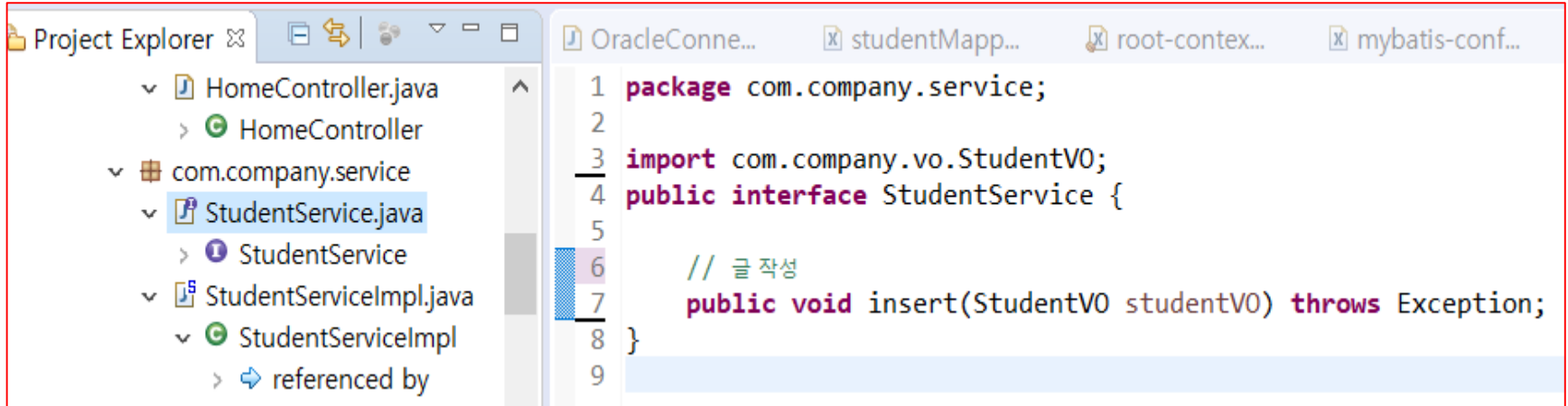


```
import javax.inject.Inject;
import org.apache.ibatis.session.SqlSession;
import org.springframework.stereotype.Repository;
import com.company.vo.StudentVO;
```

```
@Repository
public class StudentDAOImpl implements StudentDAO {
    @Inject
    private SqlSession sqlSession;
    // 게시글 작성
    @Override
    public void insert(StudentVO studentVO) throws Exception {
        sqlSession.insert("studentMapper.insert", studentVO);
    }
}
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ service에 StudentService와 StudentServiceImpl을 만들어서 코드를 작성



```
import com.company.vo.StudentVO;
public interface StudentService {

    // 글 작성
    public void insert(StudentVO studentVO) throws Exception;
}
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ service에 StudentServiceImpl.java을 만들어서 코드를 작성

```
import javax.inject.Inject;
```

```
import org.springframework.stereotype.Service;
```

```
import com.company.dao.StudentDAO;
```

```
import com.company.vo.StudentVO;
```

```
@Service
```

```
public class StudentServiceImpl implements StudentService {
```

```
@Inject
```

```
private StudentDAO dao;
```

```
// 글 작성
```

```
@Override
```

```
public void insert(StudentVO studentVO) throws Exception {
```

```
dao.insert(studentVO);
```

```
}
```

```
}
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ controller에 StudentController.java 파일을 만들고 코드를 작성

```
import javax.inject.Inject;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import com.company.service.StudentService;
import com.company.vo.StudentVO;

@Controller
@RequestMapping("/")
public class StudentController {
    private static final Logger logger = LoggerFactory.getLogger(StudentController.class);
    @Inject
    StudentService service;

    @RequestMapping(value = "insertView", method = RequestMethod.GET)
    public void insertView() throws Exception{
        logger.info("insertView");
    }
    // 글 작성
    @RequestMapping(value = "insert", method = RequestMethod.POST)
    public String insert(StudentVO studentVO) throws Exception{
        logger.info("insert");
        service.insert(studentVO);
        return "insertView";
    }
}
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ 이제 스프링이 스캔 할 수 있도록 root-context.xml파일에 코드를 추가



```
23 <property name="dataSource" ref="dataSource" /></property>
24 <property name="configLocation" value="classpath:/mybatis-config.xml"></property>
25 <property name="mapperLocations" value="classpath:mappers/**/*.xml"/>
26 </bean>
27 <bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate" destroy-method="clearCache"
28 <constructor-arg name="sqlSessionFactory" ref="sqlSessionFactory"></constructor-arg>
29 </bean>
30 <!-- scan -->
31 <context:component-scan base-package="com.company.service"></context:component-scan>
32 <context:component-scan base-package="com.company.dao"></context:component-scan>
33 <context:component-scan base-package="com.company.vo"></context:component-scan>
34 </beans>
35
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/webapp/spring/appServlet/servlet-context.xml로 들어갑니다.
- ✓ Base-package= “ com.company.controller ” 로 수정

```
18 <!-- Resolves views selected for rendering by @Controllers to .jsp
19 <beans:bean class="org.springframework.web.servlet.view.InternalResourceViewResolver"
20     <beans:property name="prefix" value="/WEB-INF/views/" />
21     <beans:property name="suffix" value=".jsp" />
22 </beans:bean>
23
24 <context:component-scan base-package="com.company.controller" />
25
26 </beans:beans>
```

564 seconds

Source Namespaces Overview beans context mvc



## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/webapp/WEB-INF/views에 insertView.jsp파일을 생성하여 코드를 추가

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
```

```
<html>
```

```
<head>
```

```
<title>Home</title>
```

```
<title>학생정보</title>
```

```
</head>
```

```
<body>
```

```
<div id="root">
```

```
<header>
```

```
<h1> 학생정보 </h1>
```

```
</header>
```

```
<hr />
```

```
<nav>
```

```
    홈 - 글 작성
```

```
</nav>
```

```
<hr />
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/webapp/WEB-INF/views에 insertView.jsp파일을 생성하여 코드를 추가

```
<section id="container">
<form role="form" method="post" action="insert">
<table>
<tbody>
<tr><td><label for="no">학 번</label><input type="text" id="no" name="no" /></td></tr>
<tr><td><label for="name">이 름</label><input type="text" id="name" name="name"
/></td></tr>
<tr><td><label for="address">주 소</label><input type="text" id="address" name="address"
/></td></tr>
<tr><td><label for="hp">휴 대 폰</label><input type="text" id="hp" name="hp" /></td></tr>
<tr><td><label for="tel">전 화</label><input type="text" id="tel" name="tel"
/></td></tr>
<tr><td><button type="submit">저장</button></td></tr>
</tbody>
</table>
</form>
</section>
<hr />
</div>
</body>
</html>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

- ✓ src/main/webapp/WEB-INF/views에 insertView.jsp파일을 생성하여 코드를 추가

```
<section id="container">
<form role="form" method="post" action="insert">
<table>
<tbody>
<tr><td><label for="no">학 번</label><input type="text" id="no" name="no" /></td></tr>
<tr><td><label for="name">이 름</label><input type="text" id="name" name="name"
/></td></tr>
<tr><td><label for="address">주 소</label><input type="text" id="address" name="address"
/></td></tr>
<tr><td><label for="hp">휴 대 폰</label><input type="text" id="hp" name="hp" /></td></tr>
<tr><td><label for="tel">전 화</label><input type="text" id="tel" name="tel"
/></td></tr>
<tr><td><button type="submit">저장</button></td></tr>
</tbody>
</table>
</form>
</section>
<hr />
</div>
</body>
</html>
```

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ <http://localhost:8080/myweb/insertView> 실행

← → http://localhost:8080/myweb/insertView

5.스프링 게시판 ... PIPA 피파링크 :: PIPA 피파링크 :: LG

### 학생정보

---

홍 - 글 작성

---

학 번

이 름

주 소

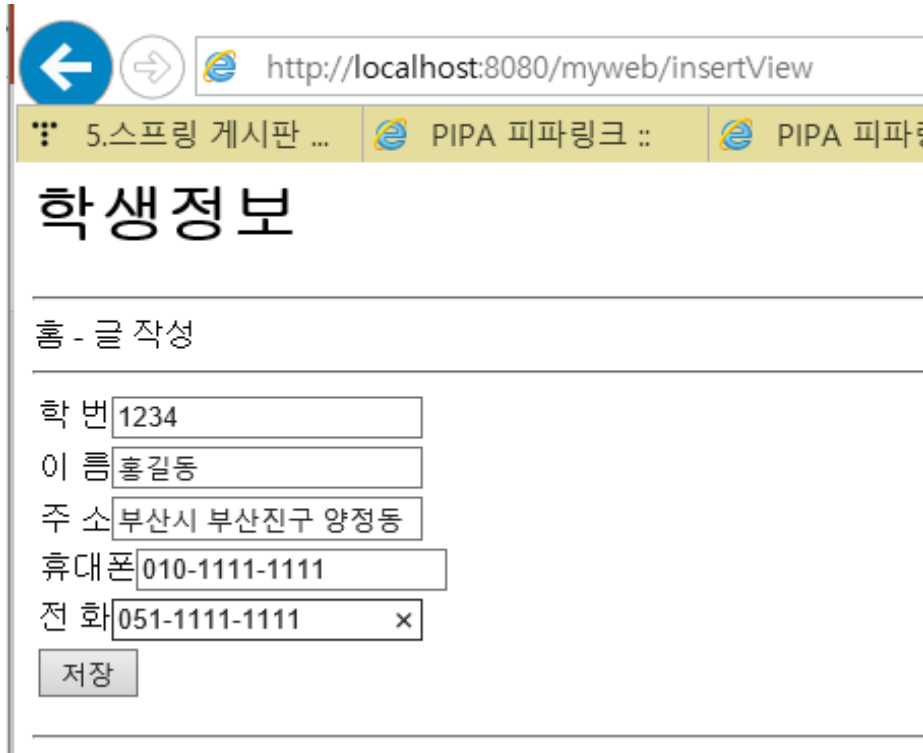
휴대폰


전 화

---

## [Spring] Spring + MyBatis + Oracle + JUnit 연동

✓ <http://localhost:8080/myweb/insertView> 실행



← →  http://localhost:8080/myweb/insertView

5.스프링 게시판 ... PIPA 피파링크 :: PIPA 피파링크

### 학생 정보

---

홍 - 글 작성

---

학 번

이름

주 소

휴대폰

전 화  ×

✓ 오라클에서 확인

```
SQL> select * from student;
```

NO	NAME	ADDRESS	HP	TEL
1234	홍길동	부산시 부산진구 양정동	010-1111-1111	051-1111-1111

```
SQL>
```