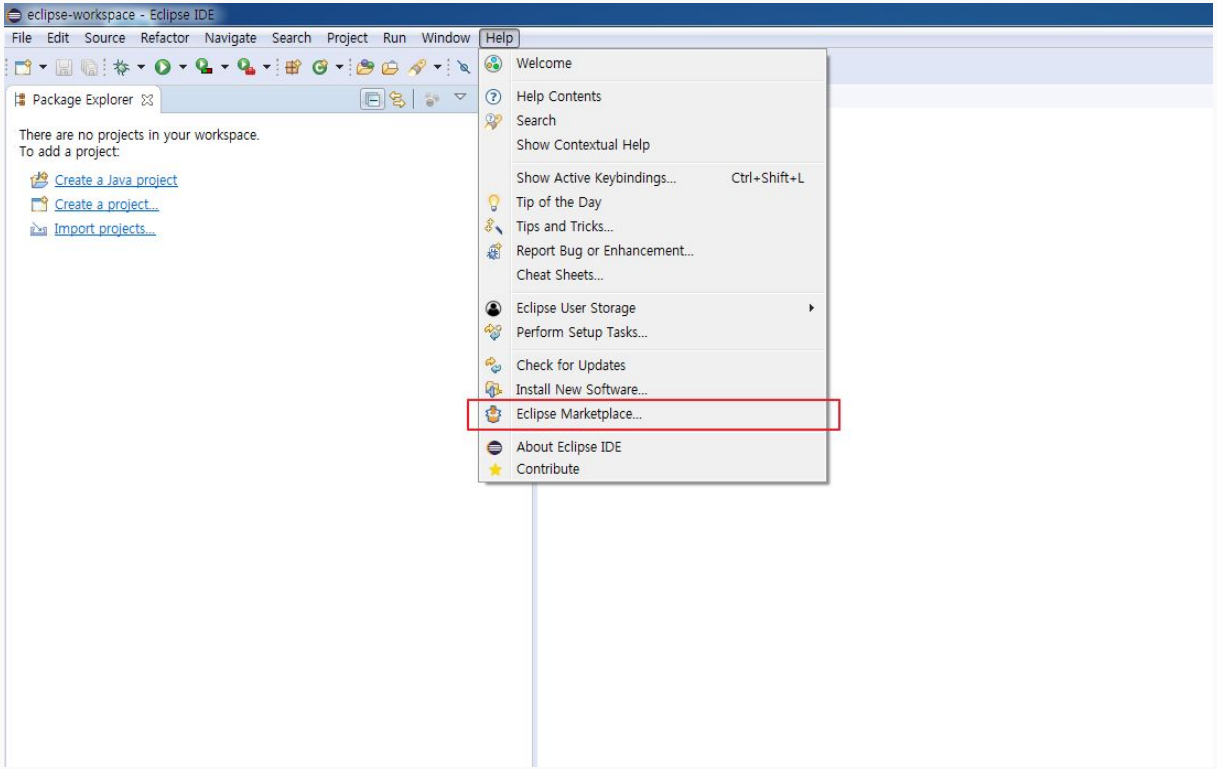
Spring 설치 및 환경 설정

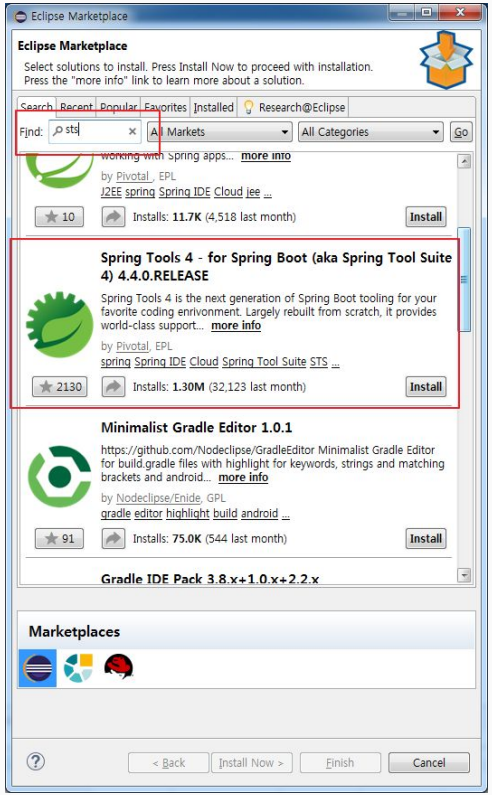
Spring 설치

- 이클립스 실행 -> [Help] -> [Eclipse Marketplace] 클릭



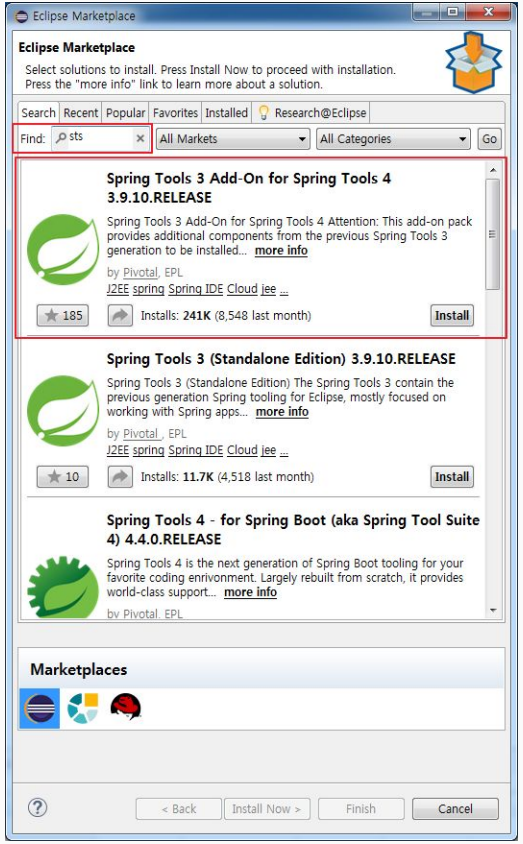
Spring 설치

- 검색창에 “sts”를 입력하고 검색
- Spring Tools 4 - for Spring Boot 를 확인하고 Install 클릭
- 설치 내용을 확인하고 절차에 따라 설치 진행



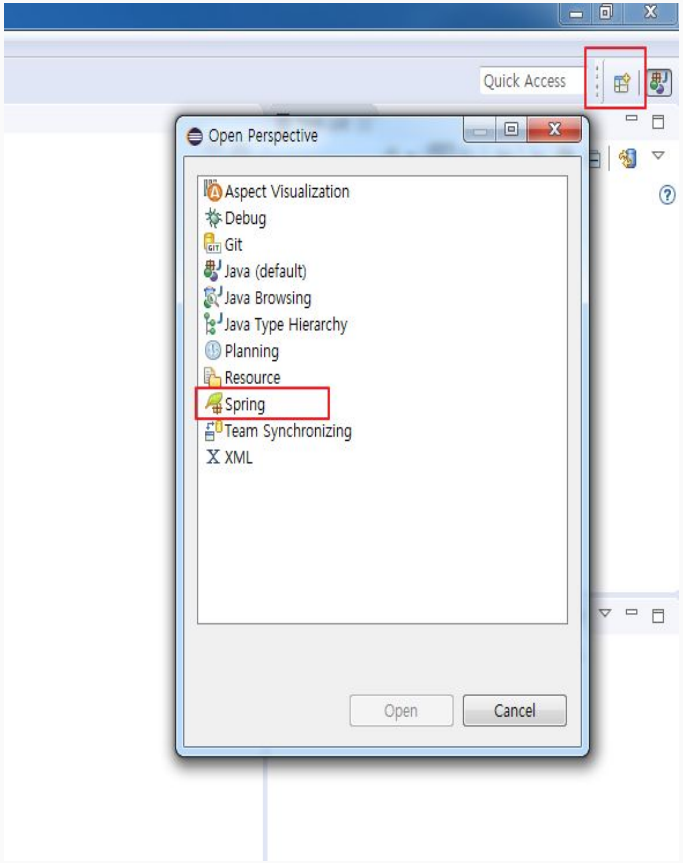
Spring 설치

- 검색창에 “sts”를 입력하고 검색
- Spring Tools 3 Add-on for Spring Tools 를 확인하고 Install 클릭
- 설치 내용을 확인하고 절차에 따라 설치 진행



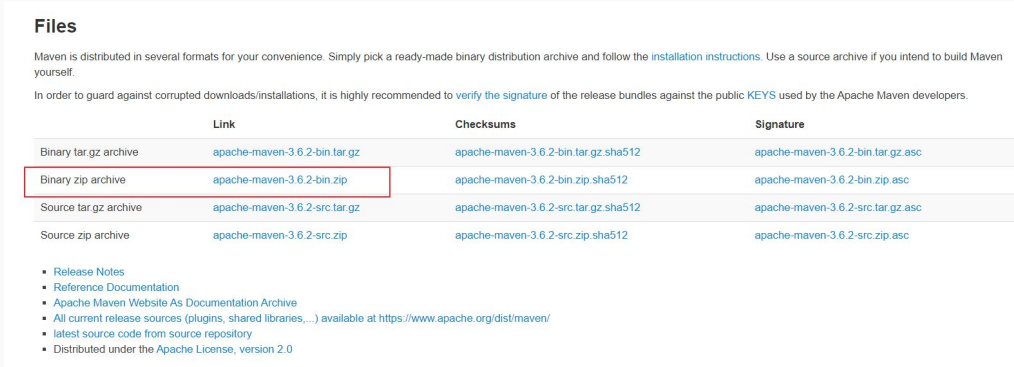
Spring 설치

- Open Perspective 클릭
- Spring 선택 및 Open
- 이클립스 작업 환경을 Spring으로 변경하는 작업
- 같은 탭에 Java를 눌러서 작업 환경 변경 가능



Maven 설치

- https://maven.apache.org/download.cgi 접속
- Files - Binary zip archive - apache-maven-3.6.2.bin.zip 다운(현재 버전)
- 압축 해제 후 apache-maven-3.6.2 폴더를 C:\Study 폴더로 복사(새로 생성된 폴더)
- 내 컴퓨터 - 속성 - 고급 시스템 설정 - 고급 - 환경 변수 - 시스템 변수 - Path 에서 'C:\Study\apache-maven-3.6.2\bin 경로 추가



Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the installation instructions. Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to verify the signature of the release bundles against the public KEYS used by the Apache Maven developers.

	Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.6.2-bin.tar.gz	apache-maven-3.6.2-bin.tar.gz.sha512	apache-maven-3.6.2-bin.tar.gz.asc
Binary zip archive	apache-maven-3.6.2-bin.zip	apache-maven-3.6.2-bin.zip.sha512	apache-maven-3.6.2-bin.zip.asc
Source tar.gz archive	apache-maven-3.6.2-src.tar.gz	apache-maven-3.6.2-src.tar.gz.sha512	apache-maven-3.6.2-src.tar.gz.asc
Source zip archive	apache-maven-3.6.2-src.zip	apache-maven-3.6.2-src.zip.sha512	apache-maven-3.6.2-src.zip.asc

- Release Notes
- Reference Documentation
- Apache Maven Website As Documentation Archive
- All current release sources (plugins, shared libraries,...) available at https://www.apache.org/dist/maven/
- latest source code from source repository
- Distributed under the Apache License, version 2.0

Maven - 오라클 설치

- Oracle 11g 경로 확인(오라클 설치된 경로)

  - c:\app\{사용자}\product\11.2.0\dbhome_1\jdbc\lib\ojdbc6.jar

- Oracle JDBC driver를 로컬 메이븐 저장소에 추가

  - mvn install:install-file -Dfile={Path/to/your/ojdbc.jar}

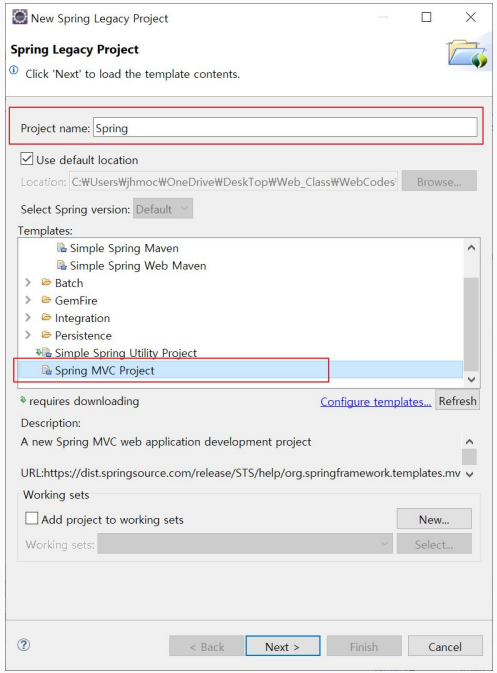
  - DgroupId=com.oracle -DartifactId=ojdbc6 -Dversion=11.2.0

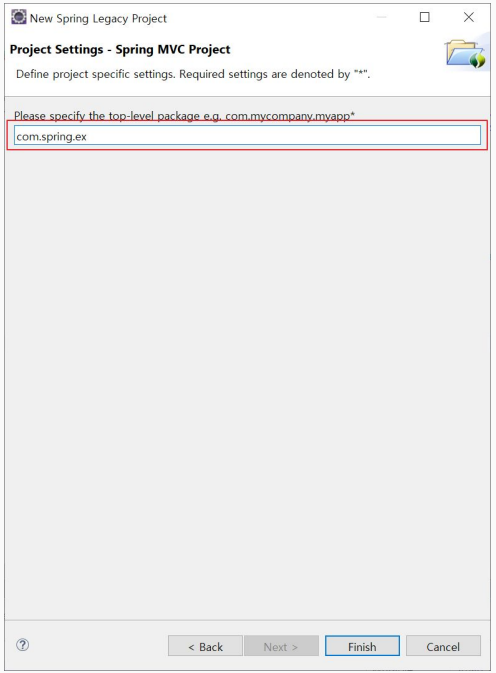
  - Dpackaging=jar

Spring 프로젝트 생성

- Spring MVC Project 생성

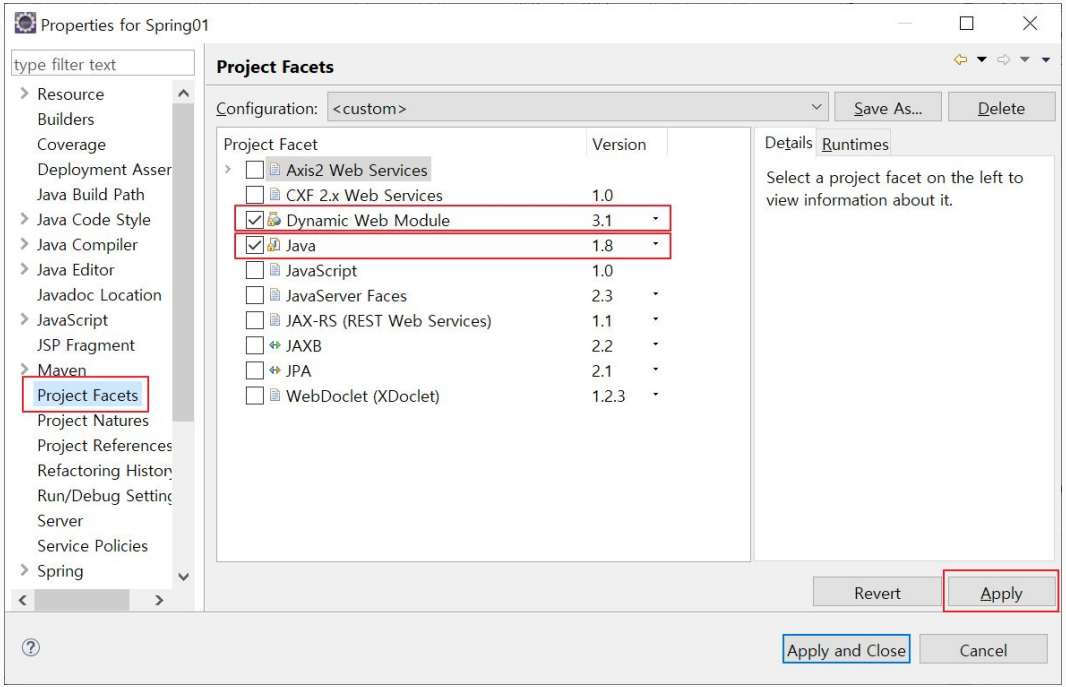
  - File → New → Spring Legacy Project → Spring MVC Project



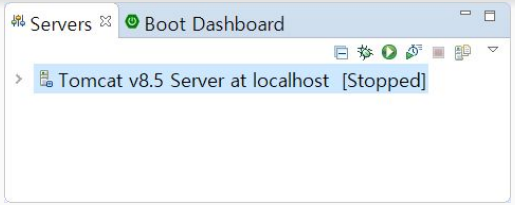


Spring 프로젝트 설정

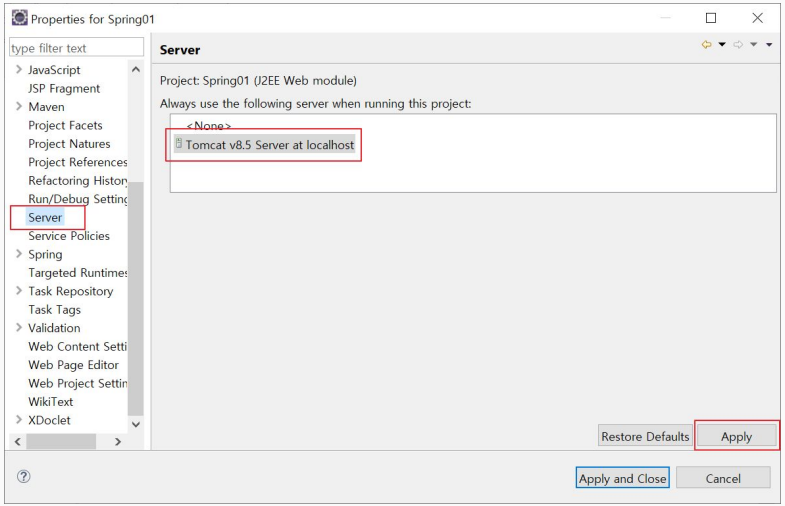
- Project → Properties → Project Facets → Java 버전 1.8 변경
- Dynamic Web Module 3.1로 변경

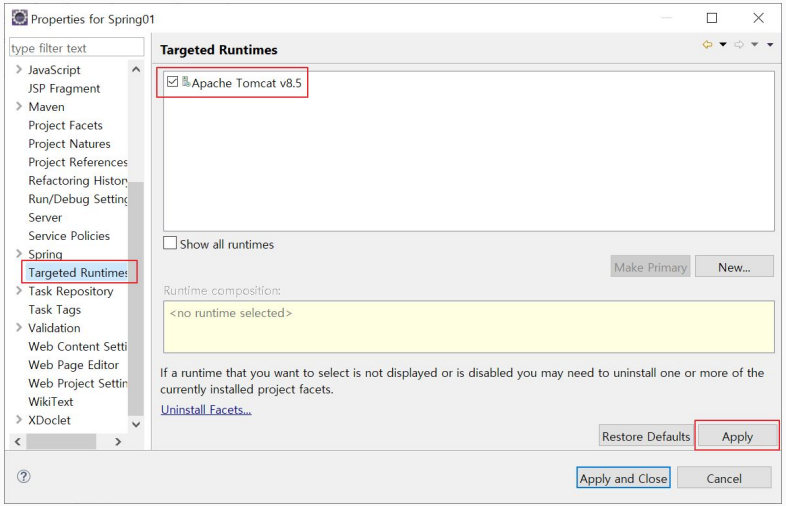


Spring 프로젝트 설정

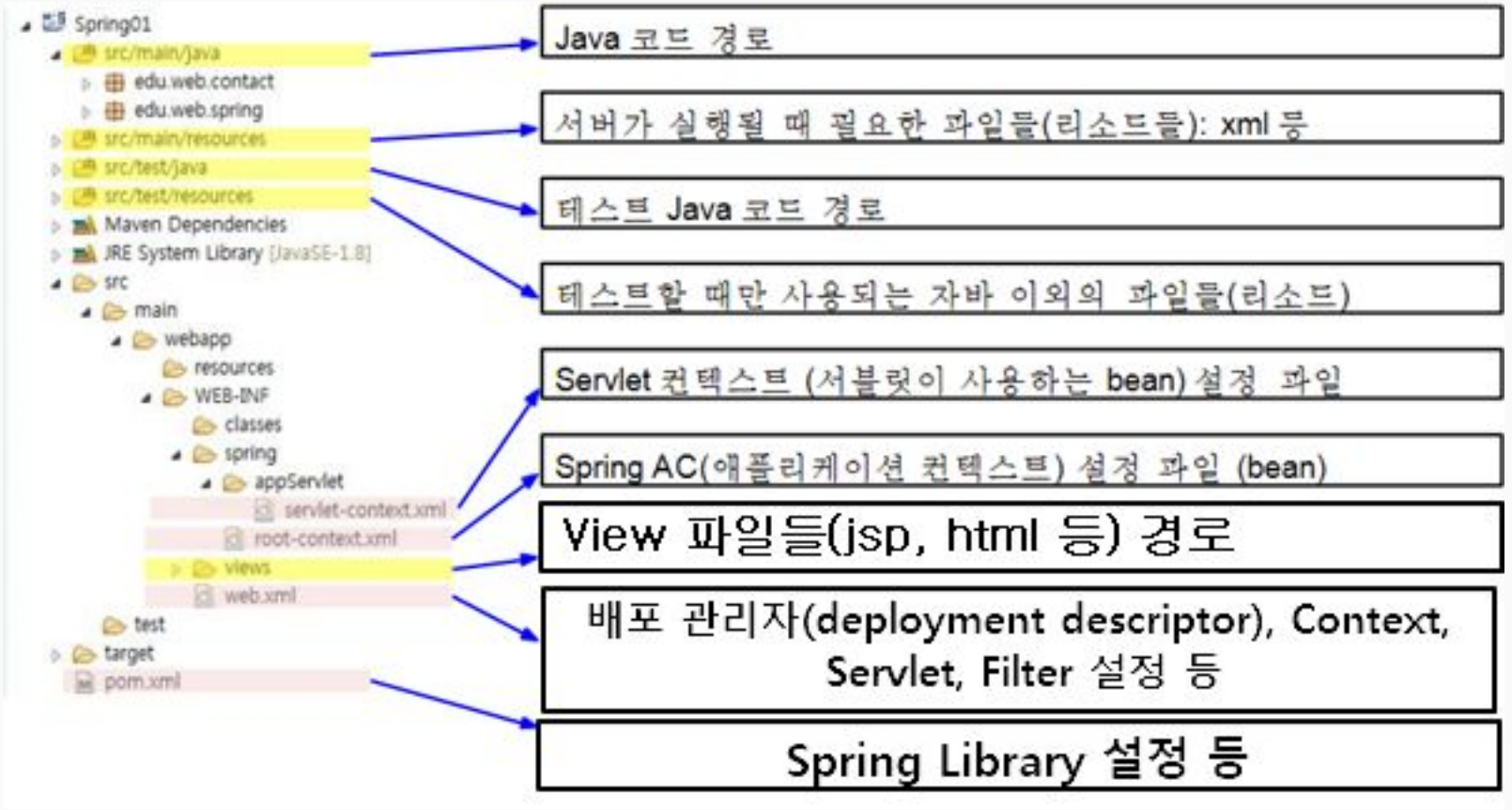


- 기존 서버가 존재하면 Spring 프로젝트와 연결
- Project → Properties → Server → 서버 선택
- Project → Properties → Targeted Runtimes → Apache Tomcat v8.5 선택





Spring 프로젝트 설정 - 프로젝트 구조



Spring 프로젝트 설정 - pom.xml

- <properties>

  - java-version
  - springframework-version (http://projects.spring.io/spring-framework/)
  - aspectj-version (https://eclipse.org/aspectj/)
  - slf4j-version (http://www.slf4j.org/download.html)

```
<properties>
        <java-version>1.8</java-version>
        <org.springframework-version>4.3.8.RELEASE</org.springframework-version>
        <org.aspectj-version>1.8.10</org.aspectj-version>
        <org.slf4j-version>1.7.25</org.slf4j-version>
</properties>
```

Spring 프로젝트 설정 - pom.xml

- <dependencies>

  <dependency>

  <groupId>log4j</groupId>

  <artifactId>log4j</artifactId>

  <version>1.2.17</version>

  ….

  </dependency>

Spring 프로젝트 설정 - pom.xml

```
<dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.1.0</version>
</dependency>
<dependency>
        <groupId>javax.servlet.jsp</groupId>
        <artifactId>javax.servlet.jsp-api</artifactId>
        <version>2.3.1</version>
        <scope>provided</scope>
</dependency>
```

Spring 프로젝트 설정 - pom.xml

```
<dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
</dependency>
```

Spring 프로젝트 설정 - pom.xml

- <build>
  <plugins>
    <plugin>
      <artifactId>maven-eclipse-plugin</artifactId>
      <version>2.10</version>
      ….
    </plugin>
    <plugin>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.6.1</version>
      <configuration>
        <source>${java-version}</source>
        <target>${java-version}</target>
      </configuration>
    </plugin>

Spring 프로젝트 설정 - pom.xml

```
<plugins>
    <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>exec-maven-plugin</artifactId>
        <version>1.6.0</version>
    </plugin>
</plugins>
```

Spring 프로젝트 설정 - web.xml

```
<web-app version="3.1"
        xmlns="http://xmlns.jcp.org/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
        http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd">

</web-app>
```

Spring 프로젝트 설정 - web.xml

- 요청 파라미터 한글 처리

```
<filter>
        <filter-name>encoding</filter-name>
        <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
        <init-param>
                <param-name>encoding</param-name>
                <param-value>UTF-8</param-value>
        </init-param>
</filter>
<filter-mapping>
        <filter-name>encoding</filter-name>
        <url-pattern>/*</url-pattern>
</filter-mapping>
```

Spring 주요 특징

- 가벼운(ligth-weight) 프레임워크
- POJO(Plain Old Java Object) 기반의 구성
  - 일반적인 Java 코드를 이용
- 의존성 주입(DI: Dependency Injection)을 통한 객체 간의 관계 구성
  - 제어의 역전(IoC: Inversion of Control): 객체를 사용하는 주체가 직접 객체를 생성하고 제어하는 것이 아니라, 외부(시스템, 프레임워크)에서 생성된 객체를 주입받아서 쓰는 방법
- AOP(Aspect-Oriented Programming) 지원
  - 비즈니스 로직은 아니지만, 대부분의 시스템이 공통으로 가져야 하는 보안, 로그, 트랜잭션과 같은 횡단 관심사(cross-cutting concerns)를 모듈로 분리하는 프로그램 패러다임
- 편리한 MVC 구조
- WAS(Web Application Server)에 종속적이지 않은 개발 환경