1. 개발 환경 구축

관리의 편의를 위해 하나의 폴더를 정해서 다운 설치하는 것이 좋다. 프레임워크 사용 시 폴더 경로에 한글이 있을 경우 문제가 될 수 있는 경우가 있으므로 폴더 명에 한글이 포함되지 않도록 한다. 예를 들면, D:₩dev로 폴더를 지정하고 개발관련 프로그램들을 dev 폴더 안에 다운 받고 설치한다.

1) JDK 다운 설치

2) Tomcat 다운 설치

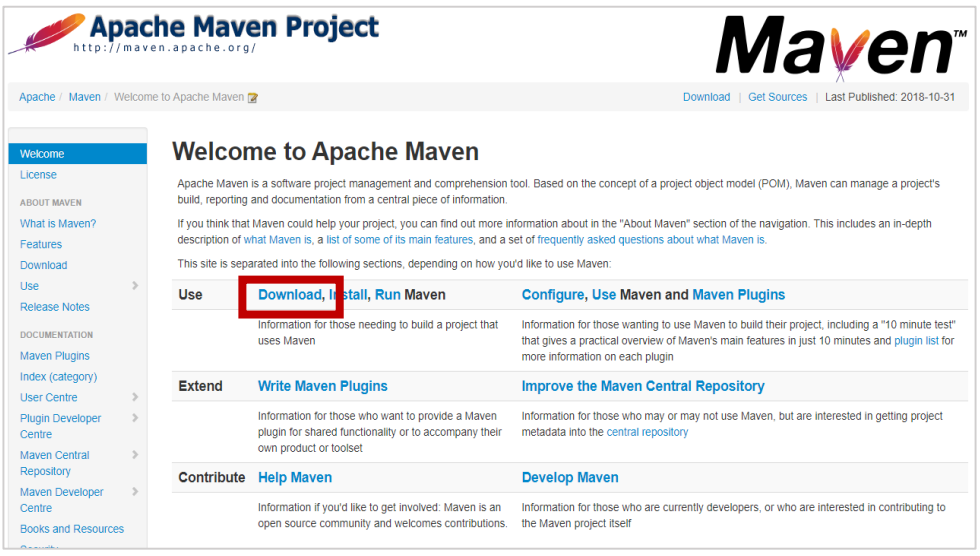
3) Maven 다운 설치

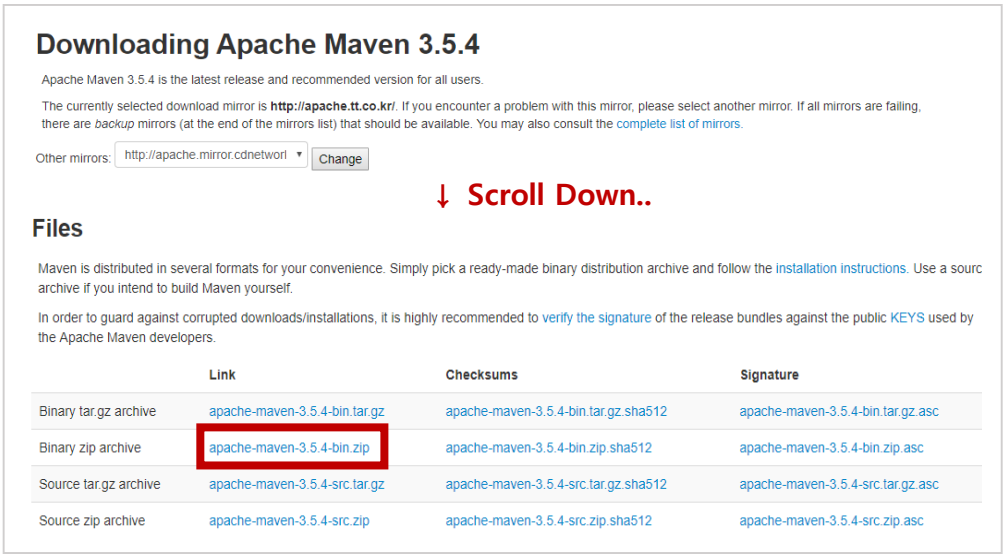
Maven이란 자바용 프로젝트 관리 도구로, **Project Object Model(POM) XML 문서**를 통해 **해당 프로젝트의 버전 정보 및 라이브러리 정보들을 통합하여 관리하는 프레임워크이다**. 일반적으로 프로젝트는 개발자가 필요한 라이브러리를 직접 찾아서 추가해야 하지만 Maven을 사용하면 pom.xml 문서에 사용하고자 하는 라이브러리를 등록하여 자동으로 프로젝트에 추가되게 하여 라이브러리 관리의 편리성을 제공해준다.

라이브러리를 다운 설치 받을 폴더를 명시적으로 지정하여 관리하는 것이 좋다.

➔ **Apache Maven 사이트에 접속하여 다운로드**

https://maven.apache.org/index.html 로 접속하여 다운로드

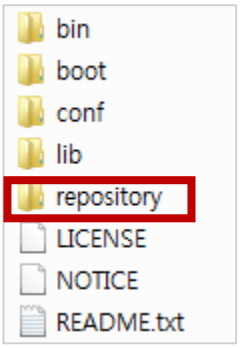




* 다운로드 받은 압축 파일을 dev 폴더에 압축 해제

➔ **라이브러리들이 저장될 폴더 생성**

압축해제 한 경로에 **'repository'**라는 저장소 역할의 새 폴더 생성



➔ **Settings.xml 파일 수정**

conf 폴더에 접근하여 **setting.xml** 파일을 열어 주석 처리된 **<localRepository>**란 부분을 찾아 복사하여 밖으로 뺀 뒤, 저장소로 이용하고자 하는 폴더(위에서 만든 repository경로)로 설정한다.

```
<!-- localRepository
 | The path to the local repository maven will use to store artifacts.
 |
 | Default: ${user.home}/.m2/repository
<localRepository>/path/to/local/repo</localRepository>
-->
<localRepository>D:₩dev₩apache-maven-3.5.4₩repository</localRepository>
```

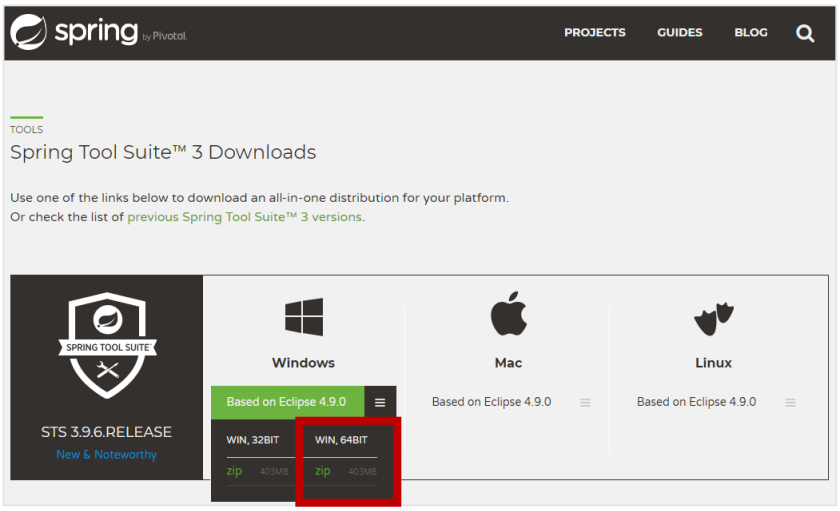
4) STS(Spring Tool Suite) 설치하기

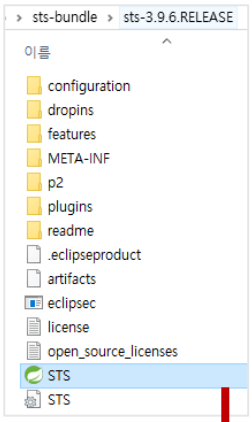
STS란 Spring Tool Suite의 약자로, **Spring Framework를 사용하기 위한 개발 툴**을 말한다. 일반적으로는 별도의 설치 도구를 통해 설치하여 사용가능하고, 이클립스 IDE에서 제공하는 STS 플러그 인을 통해 간단히 설치 할 수 있다.

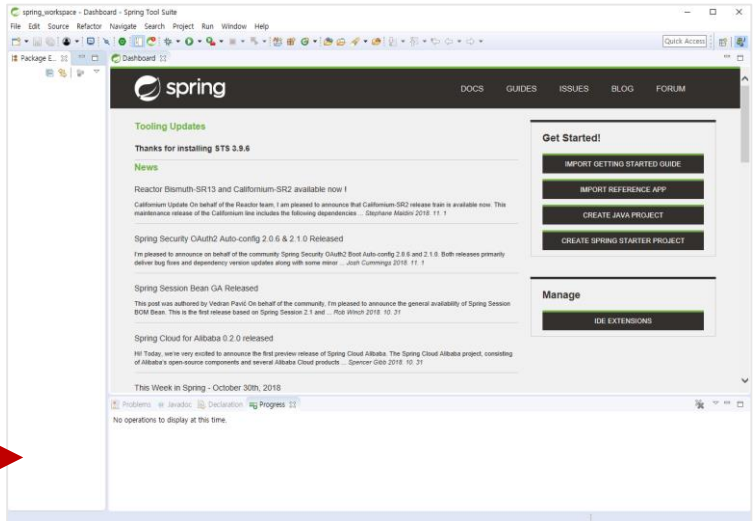
➔ 방법1: STS 직접 설치하기

STS 공식 설치 사이트 : https://spring.io/tools/sts/all

본인의 eclipse 버전에 맞춰서 STS 다운받기!! → 다운 완료 후 압축해제





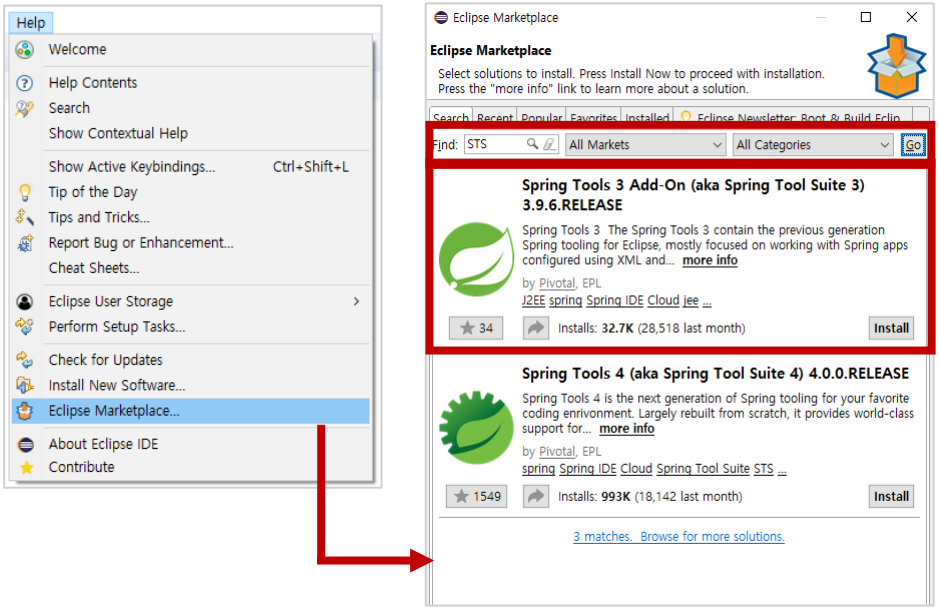


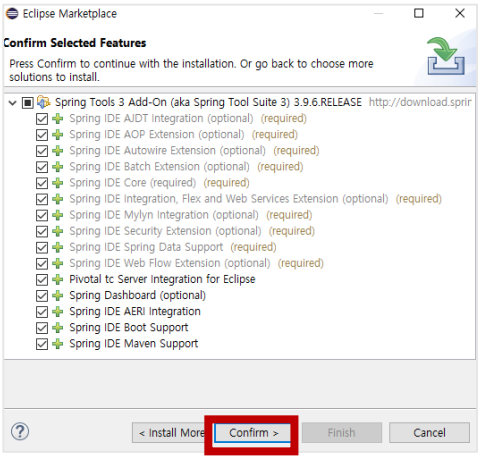
*

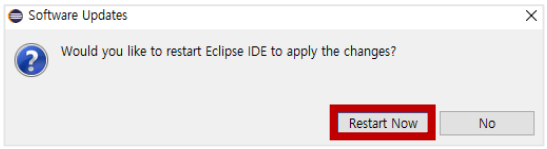
* STS 프로그램을 실행했을 때 위와 같은 화면이 보인다면 정상적으로 설치됨

➔ **방법2: Eclipse STS 플러그인을 통해 설치하기**

[Help] – [Eclipse MarketPlace..] 클릭하고 검색 창에 'STS' 입력 하여 검색 후 설치







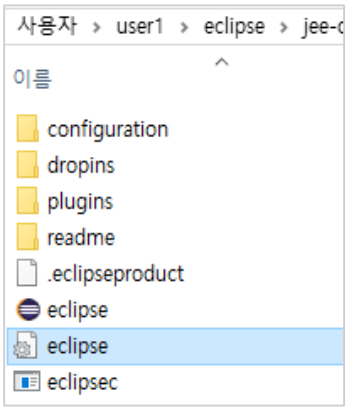
* STS 설치 완료 후 재실행한다.

➔ **보통 방법2를 통해 간단하게 설치 하지만 플러그인이 제대로 설치가 안되거나 호환이 맞지 않는 경우 방법1과 같은 방식으로 진행한다.**

5) Eclipse 및 STS 구성 설정 추가

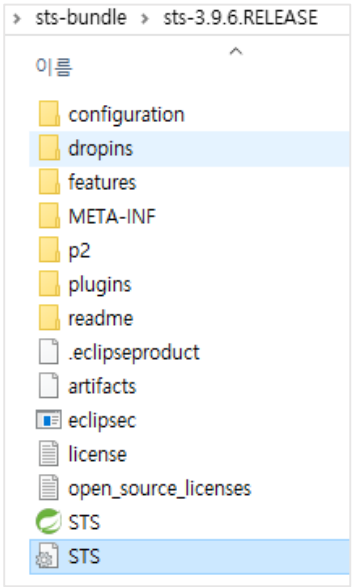
이클립스를 사용할 때 여러 JDK버전을 사용할 때가 있다. 하지만 이클립스 버전에 따라 **JDK 버전이 너무 낮으면 오류 및 실행 시 문제가 발생**할 수 있으므로 이클립스 내장 버전이 아닌 **현재 설치된 자바 버전을 인식 하도록 eclipse.ini 구성 설정 파일 (또는 STS 직접 설치 시 STS.ini 구성 설정 파일) 의 정보에 해당 내용을 추가**하고, 재실행 해야된다.

➔ Eclipse.ini 구성 설정 추가




```
-showsplash
org.eclipse.epp.package.common
--launcher.defaultAction
openFile
--launcher.appendVmargs
-vm
C:₩Program Files₩Java₩jdk1.8.0_65₩bin₩javaw.exe
-vmargs
-Dosgi.requiredJavaVersion=1.8
```

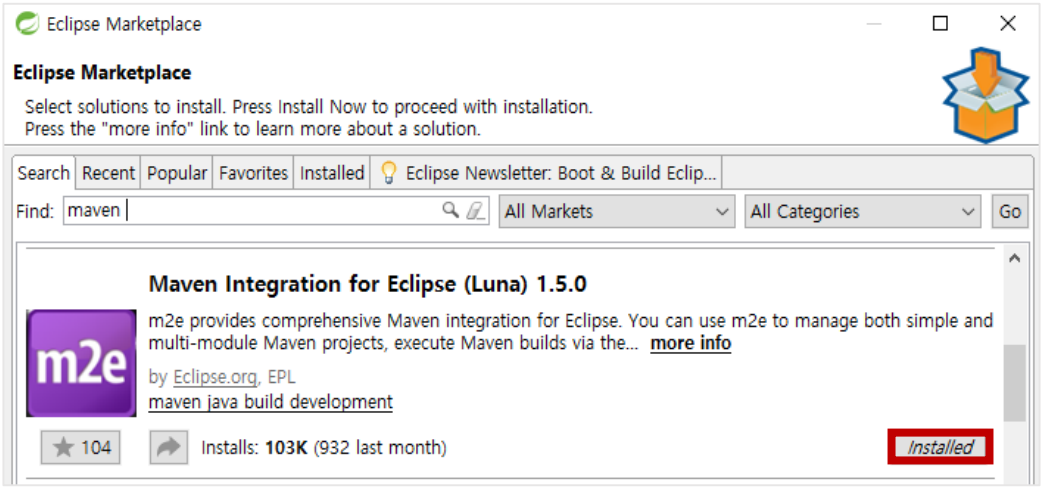
➔ STS.ini 구성 설정 추가




```
-product
org.springsource.sts.ide
--launcher.defaultAction
openFile
-vm
C:₩Program Files₩Java₩jdk1.8.0_65₩bin₩javaw.exe
-vmargs
-Dosgi.requiredJavaVersion=1.8
```

6) Maven Integration 설치

Maven을 사용하기 위한 플러그인으로, 이클립스가 버전업을 하면서 **자동으로 설치되어 있다.** 하지만 설치되어 있지 않는 경우 Maven 사용 시 오류가 나기 때문에 확인해보고 미 설치 시 [Help] – [Eclipse Marketplace..] 에 접근하여 검색을 통해 설치한다.



7) 데이터베이스 구축

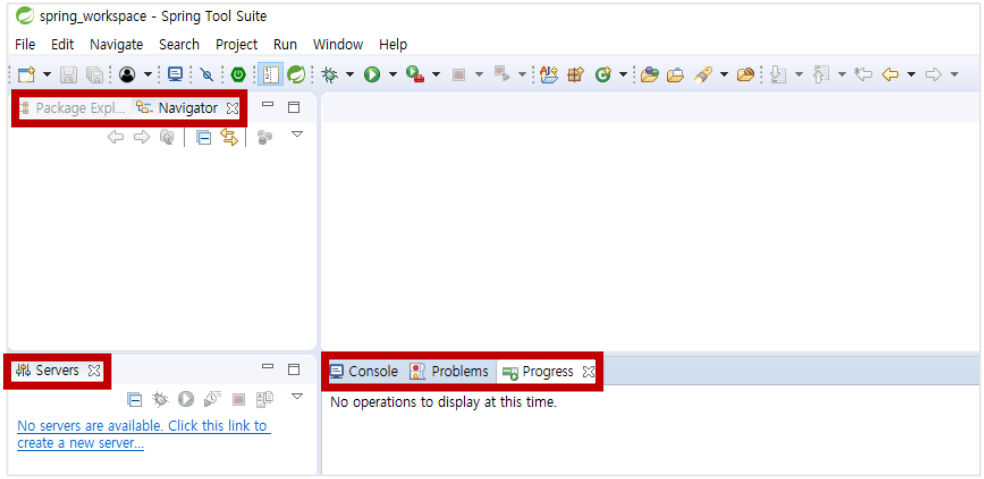
2. 실습 프로젝트

1) Workspace 환경 설정

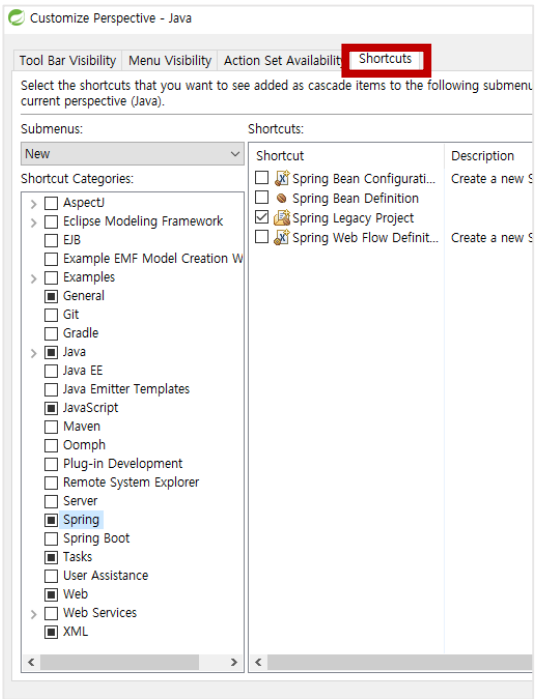
➔ Spring 프로젝트 작업 시 필요한 환경 셋팅

- [Window] – [Show View] 를 통해 아래와 같이 적용

'Package/Project Explorer', 'Navigator', 'Servers', 'Console', 'Problems', 'Progress'



- [Window] – [Perspective] – [Customize Perspective] – [Shortcuts] 탭을 통해 필요한 메뉴들 선택



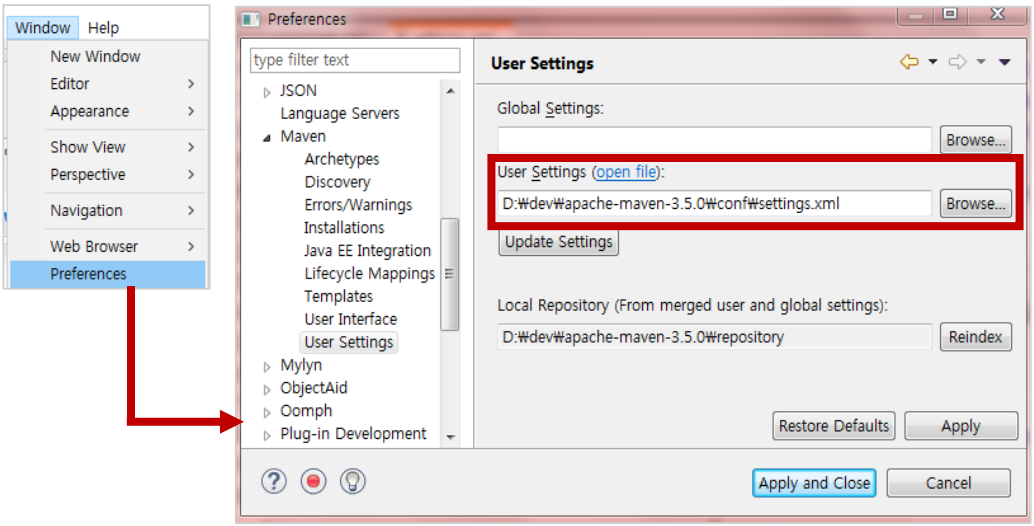
➔ 인코딩 설정

[Window] – [Preferences]에 접근하여 아래 부분들 **모두 'UTF-8'**로 적용

- [General] – [Workspace] – [Wokspace]
- [General] – [Editors] – [Text Editors] – [Spelling]
- [JSON] – [JSON Files]
- [Web] – [CSS Files] / [HTML Files] / [JSP Files]
- [XML] – [XML Files]

➔ Eclipse와 Maven 연동

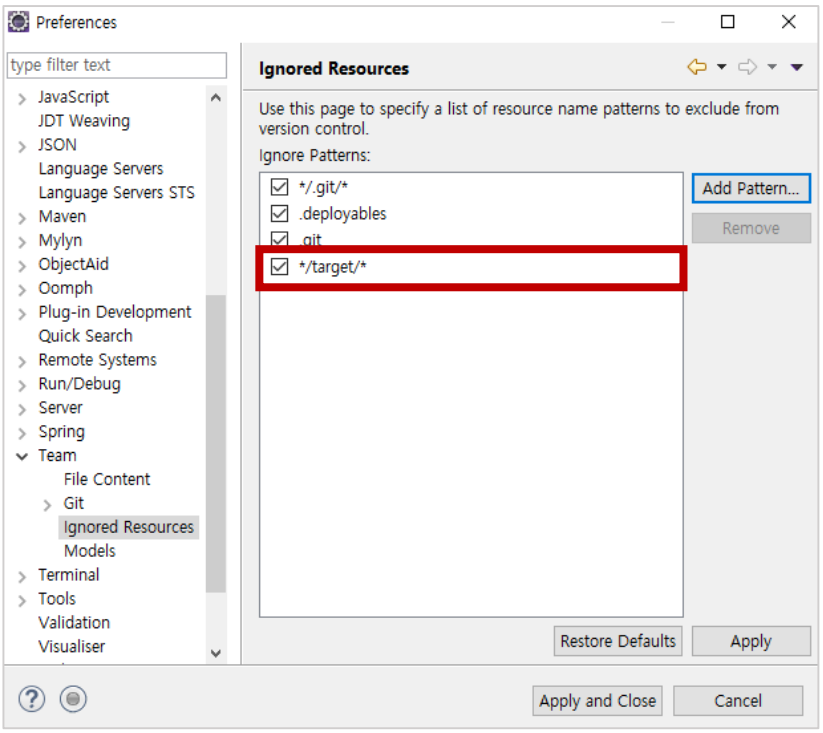
[Window] – [Preferences] ➔ [Maven] – [User Settings]에 접근하고 [User Settings]의 값을 <localRepository> 경로를 지정한 conf 폴더의 settings.xml의 위치로 변경한다.



➔ **target 제외**

Maven을 이용하여 프로젝트를 진행할 경우 target이라는 폴더가 나온다. 프로젝트를 컴파일 하게 되면 target/classes 디렉토리에 컴파일 된 클래스 파일들이 생긴다. 일반적으로 maven clean 옵션을 사용하면 제거되어 문제는 없지만, 이후 SVN, GIT 등을 이용하여 **프로젝트의 형상관리를 할 경우 프로젝트 공유 시 컴파일 된 결과까지 공유할 필요는 없기 때문에 target 폴더를 공유 목록에서 제외하는 ignore 작업**을 설정해야 된다.

[Window] – [Preferences] → [Team] – [Ignored Resources] 접근 후 [Add Patterns..] 를 통해 ' */target/* ' 를 추가해준다.



➔ **Server Runtime Environments 설정**

[Window] – [Preferences] → [Server] – [Runtime Environments] 을 통해 톰캣 환경을 구축해준다.

➔ **Local Server 구축**

- Port 변경
- Server modules without publishing 체크

2) 프로젝트 생성

➔ Spring Legacy Project 생성

[New] – **[Spring Legacy Project]** → 프로젝트 이름 입력

Templates : **'Spring MVC Project'** 템플릿 선택

[Next] > 최상위 패키지 지정 : **최소 3개 이상의 레벨**로 지정할 것

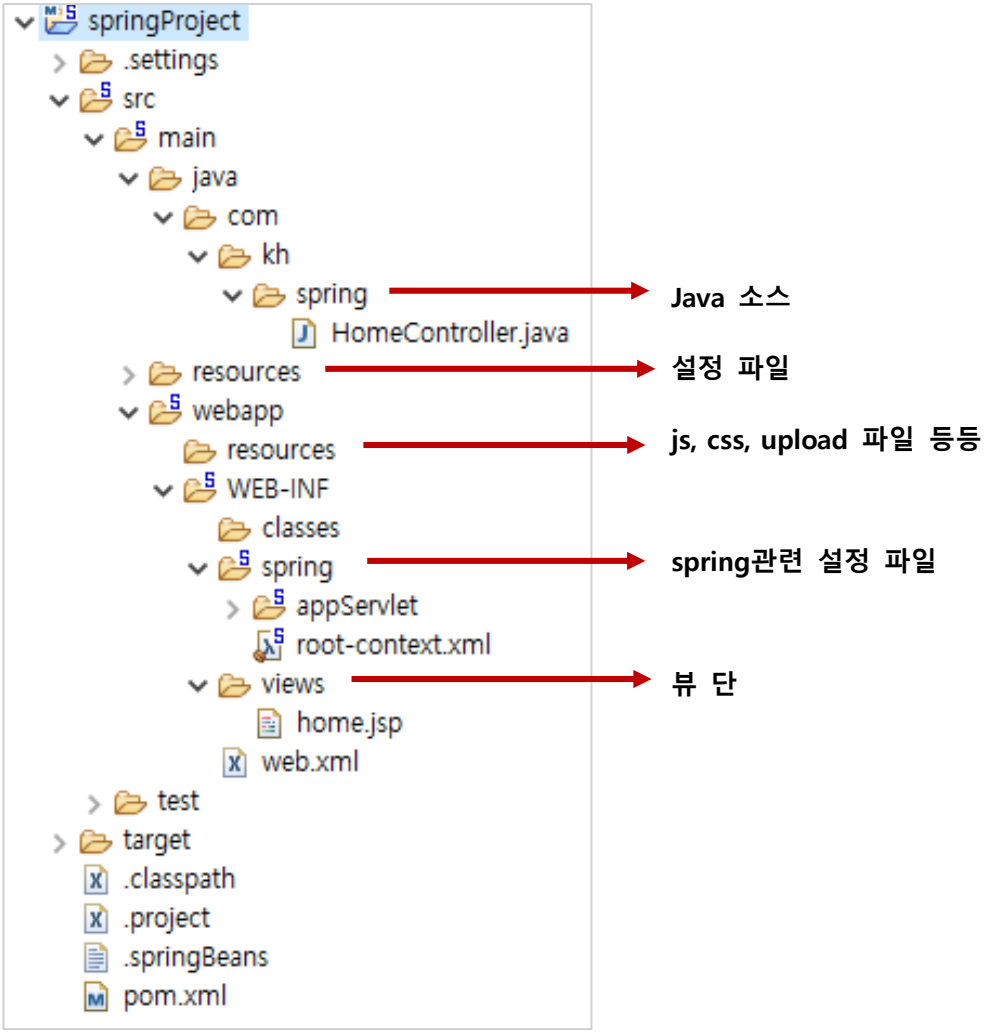
3번째 레벨이 ContextPath가 된다.

* 프로젝트 생성시 Maven이 필요한 라이브러리를 인터넷에서 다운받게 되는데 꽤 시간이 소요된다.

→ progress 바를 통해 확인 후 끝날 때 까지 대기

(제어권이 우리한테 있는게 아니라 프레임워크한테 있다.)

→ repository 경로에 필요한 라이브러리들이 다운로드 되는지 확인!



➔ **프로젝트 설정 변경**

STS를 이용하여 'Spring MVC Project' 프로젝트를 생성하면 JRE 버전도 맞지 않고 서버 라이브러리도 등록되어 있지 않다. → 설정을 수정해 줘야 된다.

프로젝트 [Properties] – [Project Facets] 탭

> Java 버전 1.8 또는 설치된 버전으로 수정

> Runtimes 탭 – 등록된 서버로 체크 > Apply

[Java Build Path] 탭 – [Librarie] 탭

> Tomcat, JRE, Maven 등록 확인 함 > Apply and Close

➔ **pom.xml 파일에서 자바 버전 변경 및 Spring 버전을 가장 최신 버전으로 변경**

스프링 버전 확인 : http://projects.spring.io/spring-framework

```
<properties>
    <java-version>1.8</java-version>
    <org.springframework-version>5.1.2.RELEASE</org.springframework-version>
    <org.aspectj-version>1.6.10</org.aspectj-version>
    <org.slf4j-version>1.6.6</org.slf4j-version>
</properties>
```

pom.xml을 수정하고 나서 어느 정도 시간이 지나면, Package Explorer 뷰에서 Java Resources/Libraries/Maven Dependencies에 스프링 버전이 일괄적으로 변경된 것을 확인 할 수 있다.

➔ **서버에 프로젝트 add 하고 실행 확인**

- jstl 관련 에러 발생 시 jstl 관련 라이브러리를 수동으로 WEB-INF/lib 폴더에 추가할 것
- 한글이 깨지는 경우 'views/home.jsp'에 jsp 파일 생성시 맨 위에 제시되는 인코딩 관련 구문을 추가할 것

3) 추가적인 라이브러리 등록

기본적으로 필요한 라이브러리들은 프로젝트 생성시 Maven이 자동으로 다운받아 준다. 하지만 **추가적으로 다뤄야 할 라이브러리들은 pom.xml을 통해 우리가 등록을 해줘야 된다.**

* 라이브러리 참조 위치 사이트 : https://mvnrepository.com/

➔ Oracle 드라이버와 MyBatis 라이브러리 내려 받기

<dependencies> 위에 **<repositories>를 이용해 사설저장소** 등록

```
<!-- * repositories -->
<!-- 기본 저장소에서 다운 받지 못하는 경우 직접 다운 받고자하는 사설 저장소 경로를 등록 -->
<repositories>
    <repository>
        <id>codelds</id>
        <url>https://code.lds.org/nexus/content/groups/main-repo</url>
    </repository>
</repositories>
```

<dependency> 이용해서 오라클, 마이바티스 관련 라이브러리들 추가

```
<!-- DataBase 오라클 사용을 위한 라이브러리 -->
<dependency>
    <groupId>com.oracle</groupId>
    <artifactId>ojdbc6</artifactId>
    <version>11.2.0.3</version>
</dependency>
<!-- Mybatis 라이브러리 -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.4.6</version>
</dependency>
<!-- 스프링에서 Mybatis를 사용하기 위한 라이브러리 -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis-spring</artifactId>
    <version>1.3.1</version>
</dependency>
<!-- 스프링 데이터베이스 기능을 사용하기 위한 라이브러리 -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <!-- <version>5.1.2.RELEASE</version> -->
    <version>${org.springframework-version}</version><!-- 위의 properties에 지정한 변수
</dependency>
<!-- 커넥션 풀 기능을 사용하기 위한 라이브러리 -->
<dependency>
    <groupId>commons-dbcp</groupId>
    <artifactId>commons-dbcp</artifactId>
    <version>1.4</version>
</dependency>
```