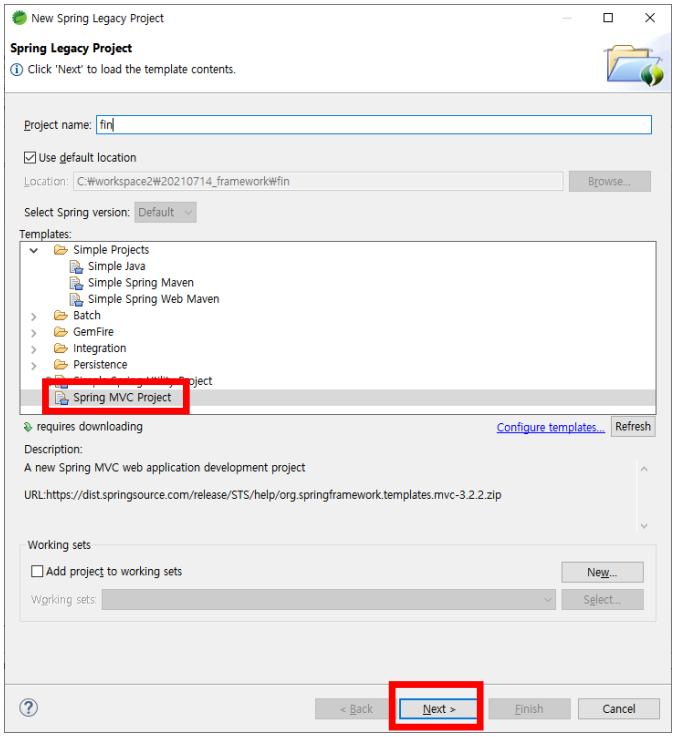
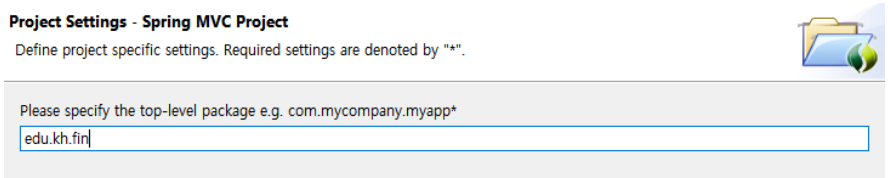
1. 프로젝트 생성

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

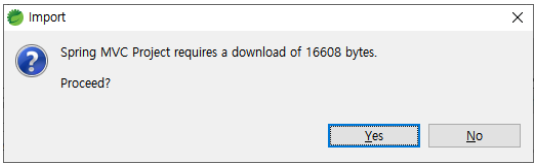
Templates : **'Spring MVC Project'** 템플릿 선택



[Next] > 최상위 패키지 지정 : edu.kh.fin(**최소 3개 이상의 레벨**로 지정할 것)

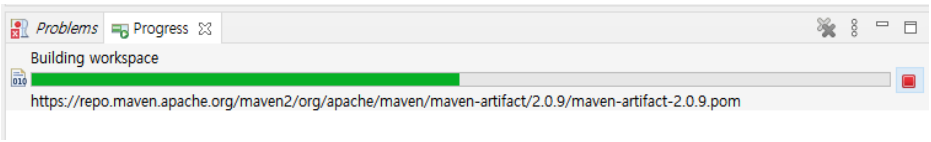


* 프로젝트 생성시 Maven이 필요한 라이브러리를 다운로드함. 인터넷 환경에 따라 오래 걸릴 수 있다.

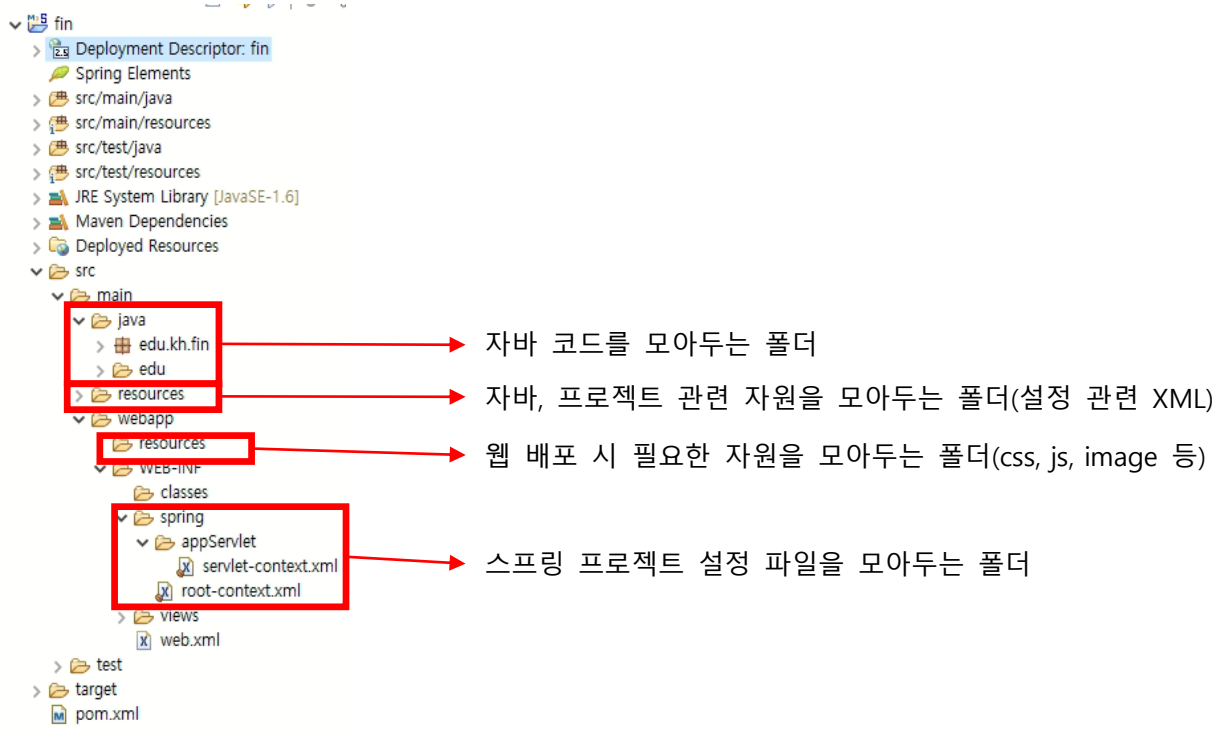


→ progress 바를 통해 확인 후 끝날 때까지 대기

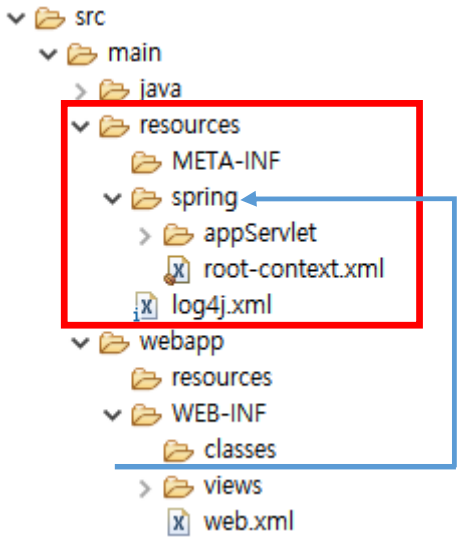
→ repository 경로에 필요한 라이브러리들이 다운로드 되는지 확인!



2. 프로젝트 구조 변경



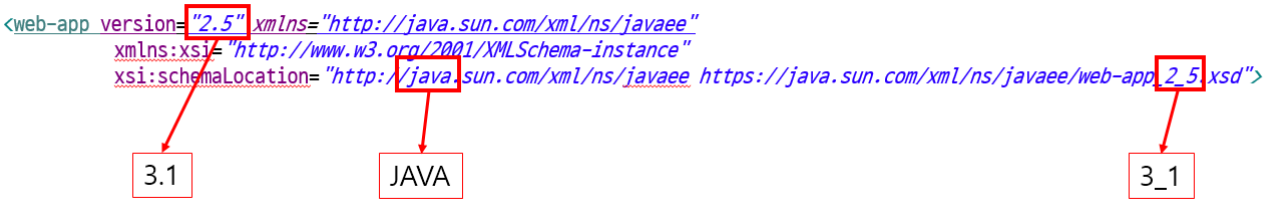
➔ spring 폴더는 프로젝트 설정 파일이 모이는 폴더 이므로
src/main/resources 폴더로 옮겨서 관리하는 것이 적절함.



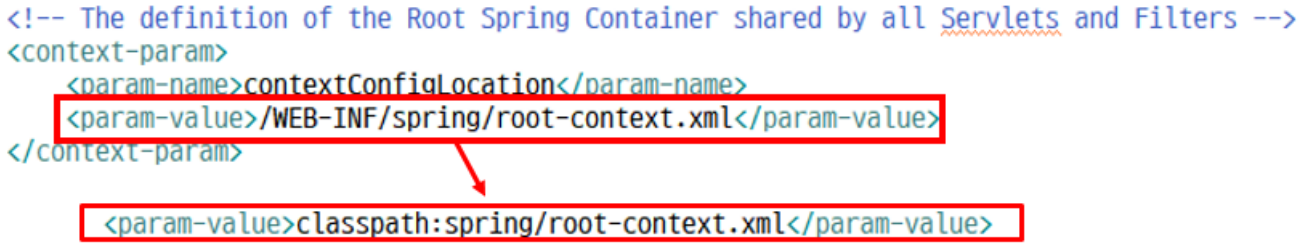
** 스프링 프로젝트 설정 파일의 경로가 변경되었기 때문에 현재 상태로는 프로젝트가 정상 수행되지 않음. -> **어느 위치에 설정파일이 존재하고 있음을 명시해야 함!**

➔ web.xml 파일 설정하기.

1) 사용하려는 웹 모듈 버전에 맞게 web.xml 버전도 변경



2) root-context.xml 파일 경로 변경 (classpath:spring/root-context.xml)



* classpath : JVM이 프로그램 실행 시 클래스 및 설정 파일을 찾는 기준이 되는 경로
Spring MVC Proejct는 src/main/java, src/main/resources 두 폴더가 포함되어 있음.

3) servlet-context.xml 파일 경로 변경 (classpath:spring/appServlet/servlet-context.xml)

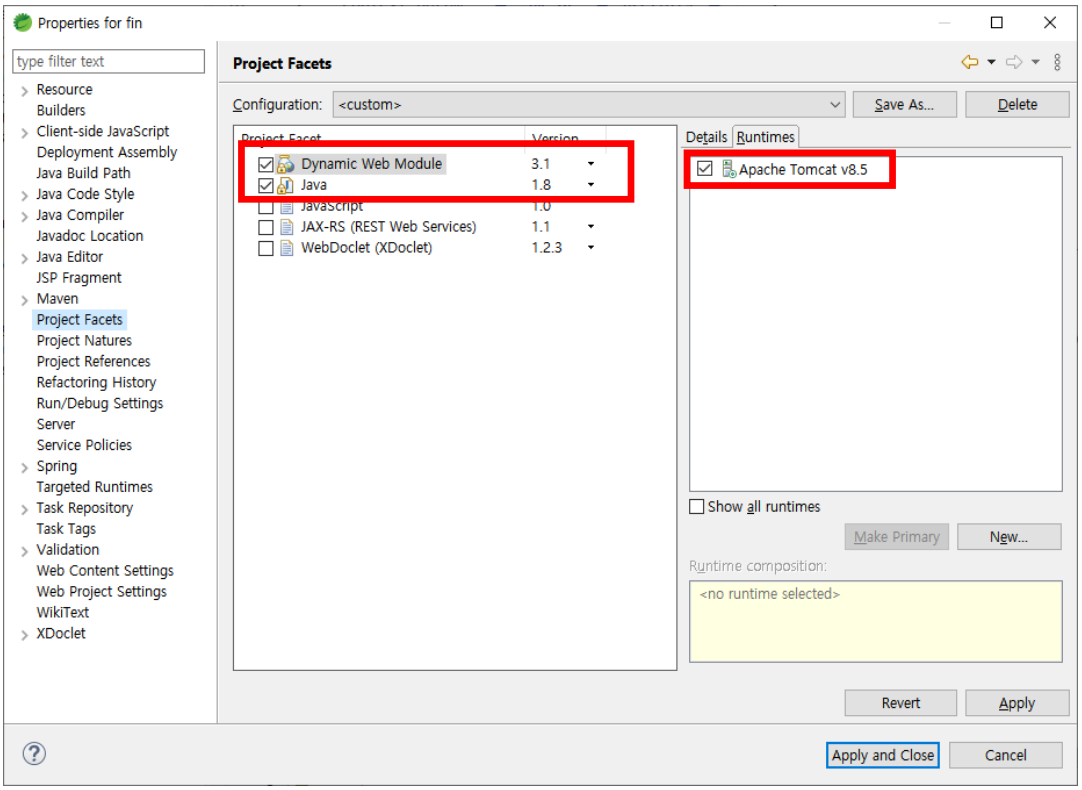
4) 요청/응답 시 한글 깨짐 방지 필터 추가 (스프링에서 제공하는 필터 사용)

```
<!-- 한글 깨짐 방지를 위한 Filter 추가 -->
<!-- 별도의 filter 클래스를 만들지 않고 스프링에서 제공하는 filter를 사용 -->
<filter>
	<filter-name>encodingFilter</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>encodingFilter</filter-name>
	<url-pattern>/*</url-pattern>
</filter-mapping>
```

3. Project Facets 변경

설치된 JDK 버전, EL 구문 사용을 위한 웹 모듈 버전으로 프로젝트 설정 변경

+ 프로젝트 배포 시 사용할 server runtimes 지정



4. pom.xml 내용 수정

properties : 공통적으로 사용할 버전 또는 설정 값 정보를 작성하는 태그

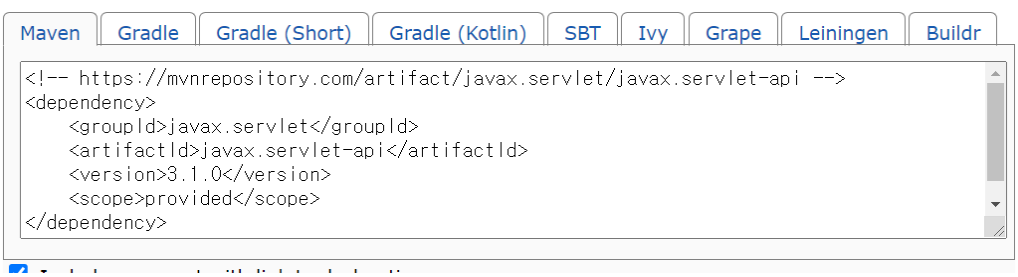
```
<properties>
	<java-version>1.8</java-version>
	<org.springframework-version>5.2.8.RELEASE</org.springframework-version>
	<org.aspectj-version>1.9.4</org.aspectj-version>
	<org.slf4j-version>1.7.25</org.slf4j-version>
</properties>
```

dependencies : Maven 프레임워크가 적용된 프로젝트는 외부 저장소(https://mvnrepository.com/)와 의존 관계를 맺고 있어 프로젝트에 필요한 파일을(라이브러리) 사용자가 직접 받을 필요 없이 해당 태그 내에 지정된 형식으로 작성하면 네트워크를 통해 외부 저장소에서 자동으로 얻어와 세팅함

1) MVN Repository -> Java Servlet API 검색 -> 첫 번째 검색 결과 선택

-> 사용하는 웹 모듈 버전과 같은 3.1.0 선택

-> 아래 Maven 탭에 있는 내용을 복사



2) pom.xml 파일의 <dependencies> 태그 내부 내용 중

2.5버전으로 설정된 servlet api 추가 구문을 지우고 MVN Repository에서 복사한 내용을 추가

```
<!-- Servlet -->
<!-- <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>servlet-api</artifactId>
        <version>2.5</version>
        <scope>provided</scope>
</dependency> -->

<!-- Servlet 버전을 3.1로 변경 -->
<!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
<dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>javax.servlet-api</artifactId>
    <version>3.1.0</version>
    <scope>provided</scope>
</dependency>
```

build : 프로젝트 빌드 시 사용되는 플러그인 추가 및 버전 정보 설정

➔ 작성된 플러그인 중 maven-compiler-plugin 내용 수정

```
<!-- 컴파일러 플러그인은 프로젝트의 소스(자바코드)를 컴파일하는 데 사용
jdk 1.6 이상 사용 시 3.0 이상 버전을 사용, source, taget에는 사용하는 jdk 버전을 작성 -->
          <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.8.1</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                    <compilerArgument>-Xlint:all</compilerArgument>
                    <showWarnings>true</showWarnings>
                    <showDeprecation>true</showDeprecation>
                </configuration>
            </plugin>
```

5. pom.xml에 작성된 내용과 이클립스 프로젝트간의 설정 동기화

➔ Maven은 프로젝트 관리 도구로써 프로젝트에 필요한 설정 내용을 pom.xml에 작성합니다.

만약에 pom.xml에 작성된 내용과 프로젝트 설정이 일치하지 않으면 오류가 발생합니다.

이때, 동기화를 진행하여 오류를 해결할 수 있습니다.

1) 프로젝트 우클릭 -> Maven -> Update Project... -> (업데이트 창에서) OK 클릭