

코드로 배우는 스프링 웹 프로젝트

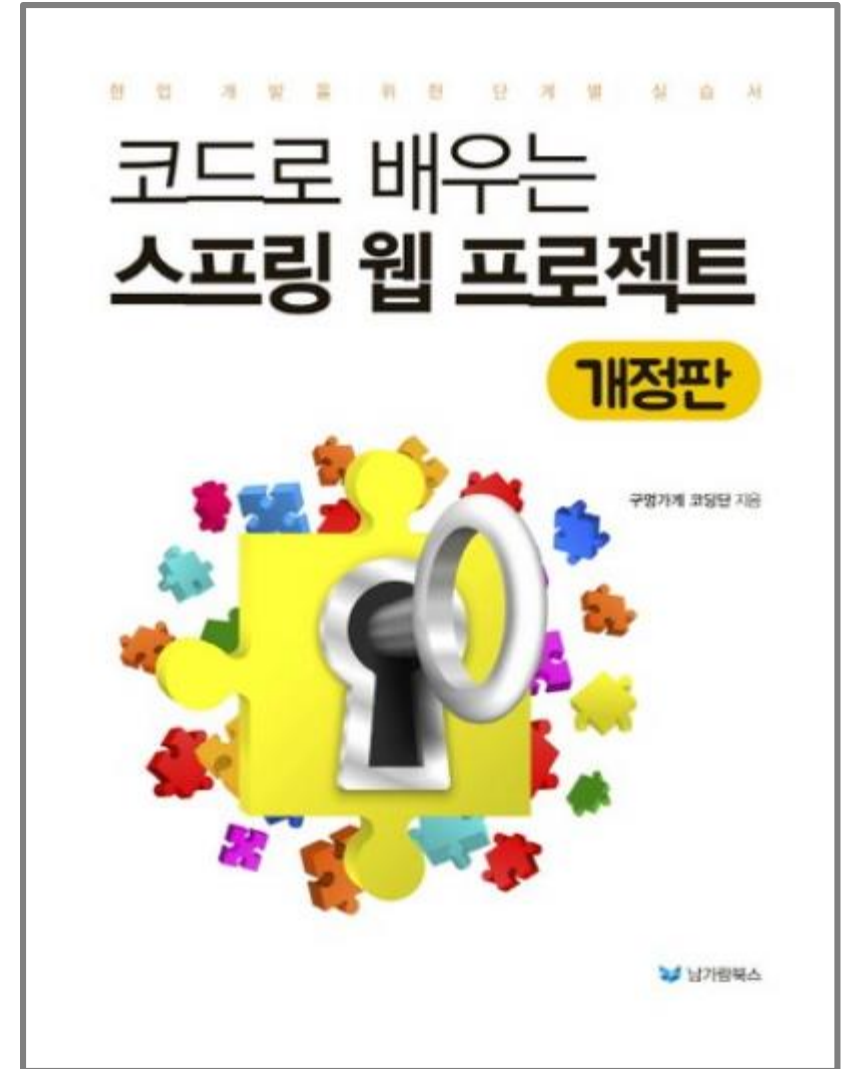
PART 1

스프링 개발 환경 구축

동의과학대학교 컴퓨터정보과
김진숙

학습 목표

- 스프링의 개발 환경 (STS(혹은 Eclipse), Lombok 등)
- 오라클 데이터베이스 설치 및 계정 설정
- 스프링과 MyBatis의 연동 설정
- 스프링 MVC의 구성 설정 및 테스트



01장. 개발을 위한 준비

PART 1

프레임워크

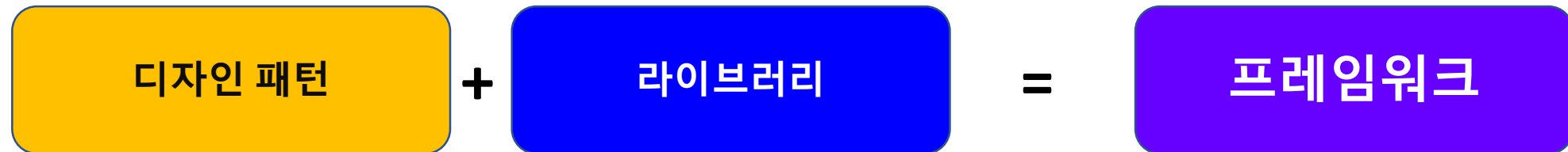
- 구현된 기능을 안정적으로 실행하도록 제어해주는 **구조**를 가진 **라이브러리**
- 비 기능적(업무적) 요구사항(성능, 보안, 확장성, 안정성 등)을 만족하는 구조
- 프레임워크는 애플리케이션들의 공통점을 찾아 **기반 구조를 제공**하여 개발자들이 기반 구조를 구현하는데 들어가는 노력 절감하도록 함

프레임워크를 사용하는 이유

- 비 기능적인 요소들을 초기 개발단계마다 구현해야 하는 불합리함을 극복
- **기능 요구사항에 집중**할 수 있음
- 디자인 패턴과 마찬가지로 반복되는 문제 해결을 위해 특화된 솔루션을 제공

디자인패턴과 프레임워크의 관련성

- **디자인 패턴**(예: MVC 패턴)은 응용을 설계할 때 필요한 구조적 가이드라인이 되어 줄 수는 있지만 구체적으로 구현된 기반 코드를 제공하지 않음
- **프레임워크**는 디자인 패턴과 함께 패턴이 적용된 기반 클래스 라이브러리를 제공하여 **구조적 틀**과 **구현 코드**를 함께 제공
- 개발자는 프레임워크의 기반 코드를 확장하여 사용하면 자연스럽게 프레임워크에 사용된 패턴을 적용할 수 있게 됨

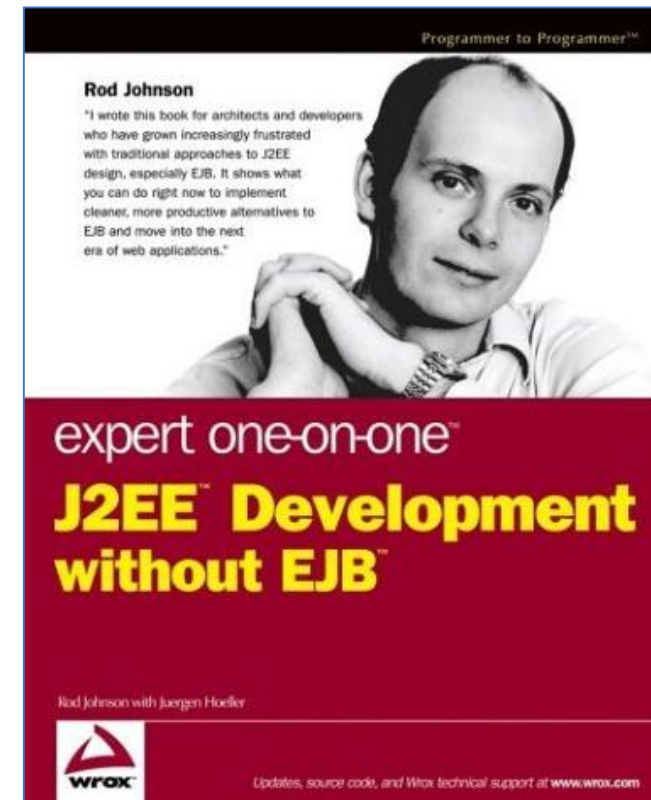


스프링 프레임워크

- Java 기반 응용 프로그램 개발을 위한 경량의 오픈소스 프레임워크
- 2003년 6월 Rod Johnson이 발표
- 모든 Java 애플리케이션 개발에 사용할 수 있으며 Java EE 위의 웹 애플리케이션 개발에 주로 사용됨

[참고 사항]

스프링 프레임워크의 역사 : <https://okky.kr/article/415474>



스프링 프레임워크 특징

- **경량**의 컨테이너로써 자바 객체를 직접 관리
- **POJO** 방식의 프레임워크(특정 개체에 독립적)
- IoC/DI 지원
- AOP 지원
- myBatis, Hibernate 등의 DB 라이브러리 지원
- 장단점

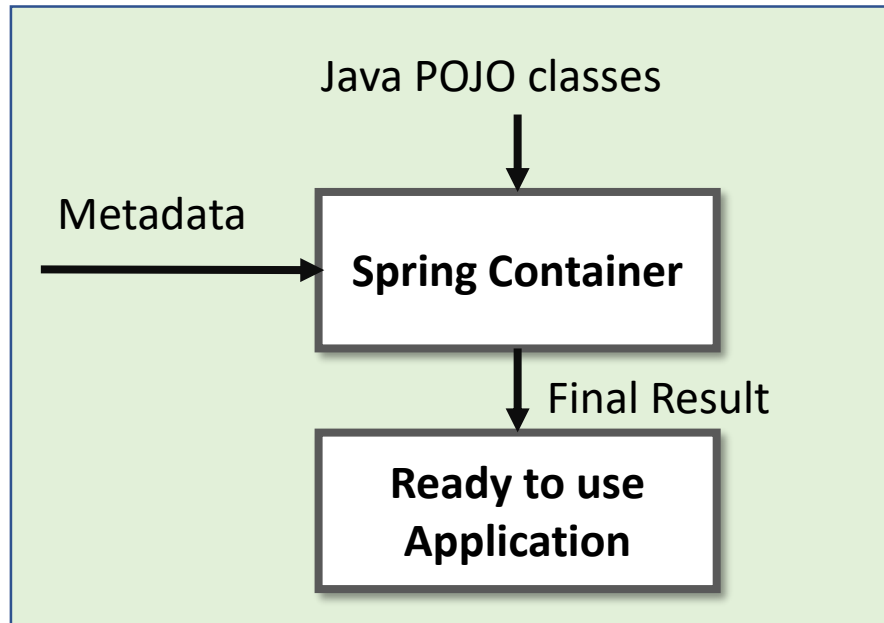
장점	단점
<ul style="list-style-type: none">• Java 코드를 줄일 수 있다.• 반복되는 작업을 줄일 수 있어 기능 개발에 집중할 수 있다.• 프로젝트 관리가 용이하다.• 다수의 개발자와 동시에 프로젝트를 수행하기가 용이하다.	<ul style="list-style-type: none">• 처음 프로젝트 세팅이 다소 복잡하다.• 개념을 제대로 숙지하지 못하면 코드 분석 조차 하기 어렵다.

스프링 프레임워크의 용어

- **PSA**(Portable Service Abstraction)
 - 환경변화와 관계없이 일관된 방식의 기술 접근 환경을 제공하려는 추상화 구조 의미
 - 특정 기술에 직접적 영향을 받지 않게끔 객체를 POJO 기반으로 한번 더 추상화한 Layer를 갖고 있으며 이를 통해 일관성 있는 서비스 추상화를 만들어 냄
- **IoC**(Inversion of Container) : 제어의 역전
 - 객체 관리를 개발자가 하지 않고 스프링이 대신 맡아 처리해 주는 기술
- **DI**(Dependency Injection) : 의존성 주입
 - 유연하게 확장 가능한 객체를 만들고 그 관계는 외부에서 동적으로 설정
- **POJO**(Plain Old Java Object) : 순수 자바 객체
 - 자바모델이나 기능, 프레임워크의 규칙을 따르지 않고 독립적이며 단순한 기능만을 가진 객체
- **AOP**(Aspect Oriented Programming) : 관점 지향 프로그래밍
 - 업무 로직을 담당하는 코드(핵심관심)에 있는 기술관련 코드(횡단관심)를 분리하여 별도의 모듈로 관리(로깅, 보안, 트랜잭션)

IoC

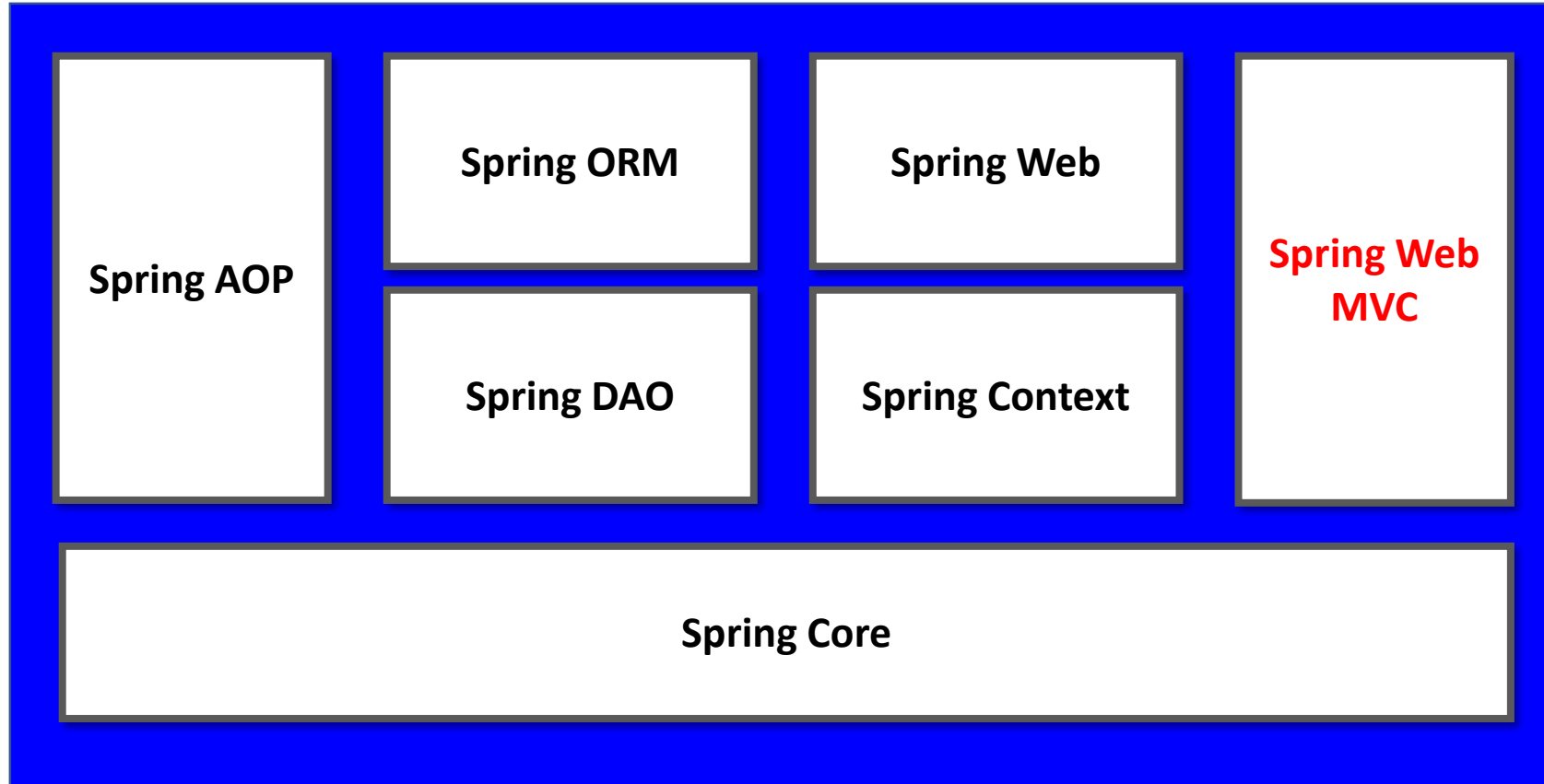
- IoC(Inversion of Control) : 제어 역전
 - 객체의 생성, 의존성 설정, 생명주기 관리까지 모든 객체에 대한 제어를 개발자가 하지 않고 대신 **프레임워크**가 맡아서 처리
 - 개발자가 코드의 흐름이나 객체 생성에 관련된 코드를 소스코드에 직접 작성하는 것이 아니라 프레임워크가 사용하는 파일에 정보를 주면 이를 토대로 프레임워크가 객체를 생성, 반환하고 코드 동작 순서를 결정하게 된다는 의미
 - POJO의 생성, 초기화, 서비스, 소멸에 대한 권한 가짐



DI(Dependency Injection)

- 의존성 주입이라는 OOP 방법 중 하나
- 객체들 간 부품 조립으로 생각하는 것이 좋음
- 스프링(ApplicationContext)이 관리하는 객체들을 빈(Been)이라고 함
- 각 클래스간의 의존관계를 **빈(bean) 설정 정보**를 바탕으로 컨테이너가 자동으로 연결해주는 것
- 개발자 역할
 - 빈(bean) 설정 파일에서 의존관계가 필요하다는 정보 추가
- 컨테이너의 역할
 - 프로그램 흐름의 주체가 되어 애플리케이션 코드에 의존관계를 주입
- 빈들 간의 의존관계를 처리하는 방식
 - annotation 설정(***) : 개발 시 사용
 - xml 설정 : 운영 시 사용
 - java 설정

스프링 프레임워크의 구성요소



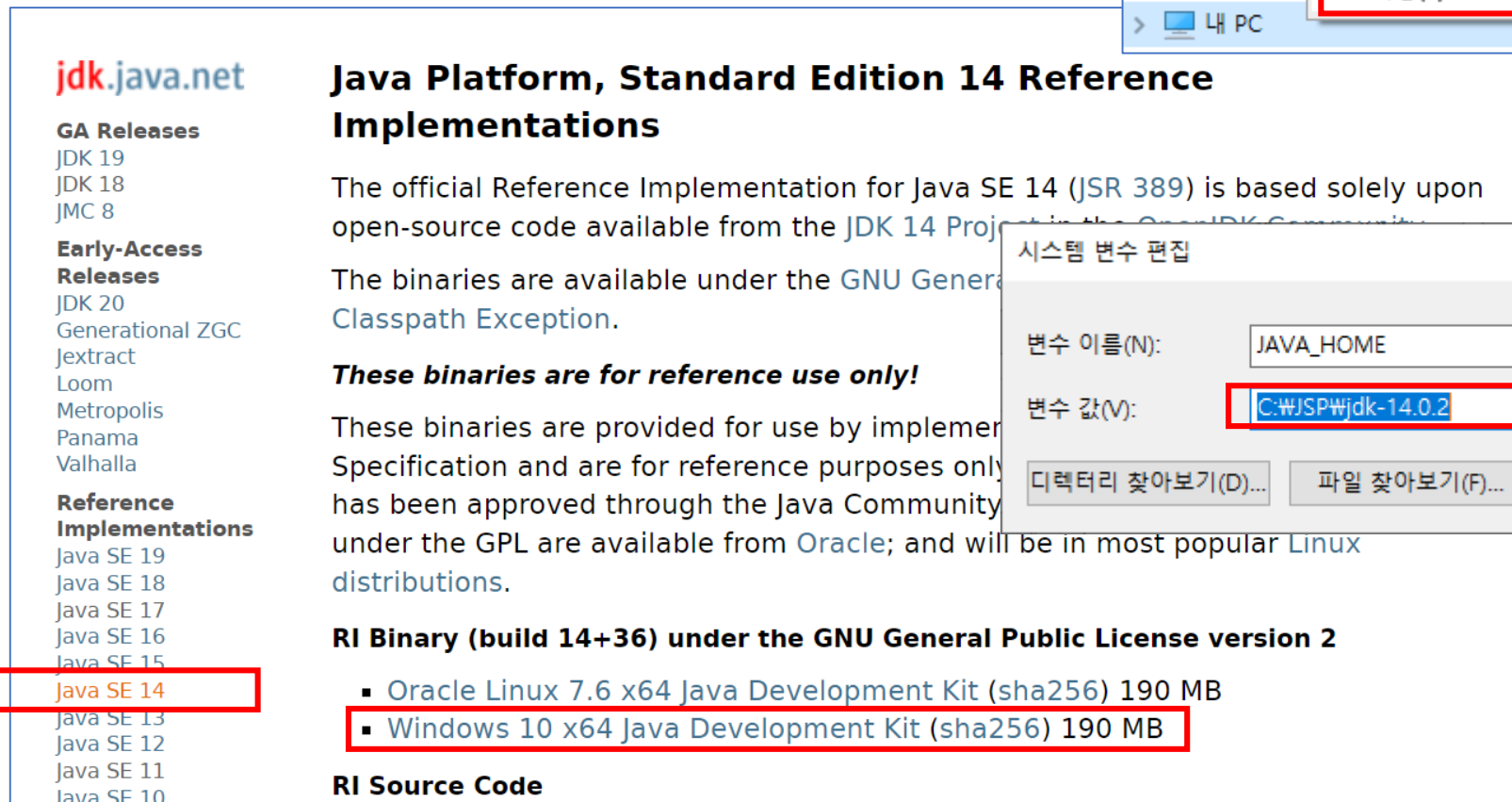
스프링 프레임워크의 구성요소

구성 요소	설명
Spring Core	<ul style="list-style-type: none">• Spring 프레임워크의 기본 기능 제공• BeanFactory는 Spring의 기본 컨테이너이면서 스프링 DI의 기반
Spring AOP	<ul style="list-style-type: none">• 개발자, 운영자 관점의 업무• 업무로직과 공통로직 분리• 로그, 보안, 트랜잭션
Spring ORM	<ul style="list-style-type: none">• MyBatis, Hibernate, JPA 등 널리 사용되는 ORM 프레임워크와의 연결고리 제공
Spring DAO	<ul style="list-style-type: none">• JDBC에 대한 추상화 계층으로 JDBC 코딩이나 예외처리를 간편화• AOP 모듈을 이용해 트랜잭션관리 서비스도 제공
Spring Web	<ul style="list-style-type: none">• 웹어플리케이션 개발에 필요한 기본 기능 제공• 다른 웹어플리케이션 프레임워크(스트러츠, 웹워크)와의 통합 지원
Spring Context	<ul style="list-style-type: none">• 기존의 Core부분인 BeanFactory 개념을 확장한 것• 국제화(i18N)메시지, 애플리케이션 생명주기 이벤트, 유효성 검증 등을 지원
Spring web MVC	<ul style="list-style-type: none">• 사용자 인터페이스가 애플리케이션 로직과 분리되는 웹애플리케이션을 만드는 경우 일반적으로 사용되는 패러다임

JDK14 설치

<https://jdk.java.net/java-se-ri/14>

- JDK14사용(JDK 1.8 이상)
- 환경 변수 설정도 같이 진행할 것



jdk.java.net

GA Releases

- JDK 19
- JDK 18
- JMC 8

Early-Access Releases

- JDK 20
- Generational ZGC
- Jextract
- Loom
- Metropolis
- Panama
- Valhalla

Reference Implementations

- Java SE 19
- Java SE 18
- Java SE 17
- Java SE 16
- Java SE 15
- Java SE 14**
- Java SE 13
- Java SE 12
- Java SE 11
- Java SE 10

Java Platform, Standard Edition 14 Reference Implementations

The official Reference Implementation for Java SE 14 (JSR 389) is based solely upon open-source code available from the [JDK 14 Project](#) in the [OpenJDK Community](#).

The binaries are available under the [GNU General Public License](#) with [Classpath Exception](#).

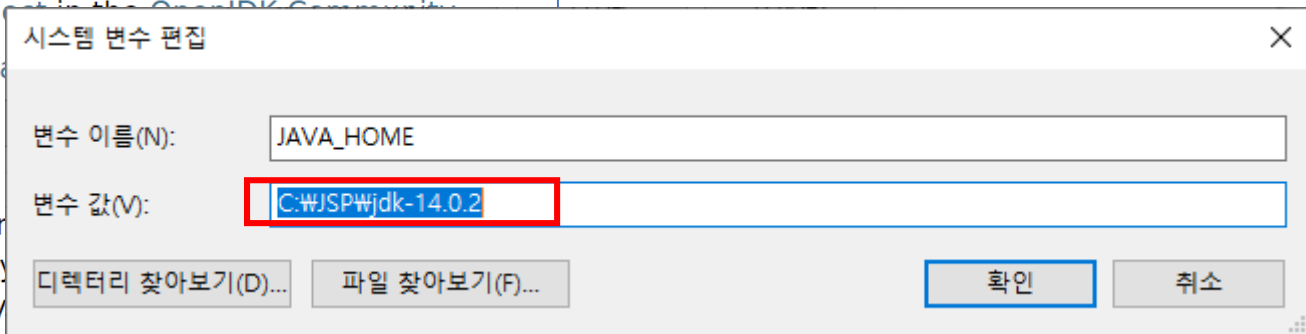
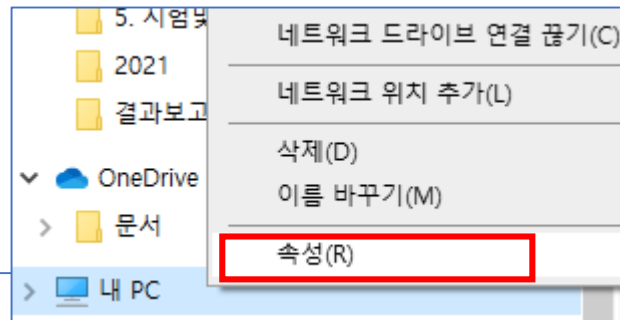
These binaries are for reference use only!

These binaries are provided for use by implementers of the Java Platform, Standard Edition 14 Specification and are for reference purposes only. The implementation has been approved through the Java Community Process and binaries under the GPL are available from [Oracle](#); and will be in most popular Linux distributions.

RI Binary (build 14+36) under the GNU General Public License version 2

- Oracle Linux 7.6 x64 Java Development Kit (sha256) 190 MB
- Windows 10 x64 Java Development Kit (sha256) 190 MB**

RI Source Code



[참고] JDK17은 사용안됨

STS 설치

컴퓨터 이름은 영문으로 되어
있어야 함

- 일반적으로_ <http://spring.io/tools> 에서 STS 다운로드하지만
- 간편한 POM.xml 설정을 위해 **3.9.14** RELEASE 사용
 - <https://github.com/spring-projects/toolsuite-distribution/wiki/Spring-Tool-Suite-3#latest-sts3-downloads>

Spring Tool Suite 3.9.14 (New and Noteworthy)

full distribution on Eclipse 4.17

STS4는 스프링부트를 고려하여
만들어진 IDE라서 세팅 등
학습하기에는 STS3가 적합

- https://download.springsource.com/release/STS/3.9.14.RELEASE/dist/e4.17/spring-tool-suite-3.9.14.RELEASE-e4.17.0-win32-x86_64.zip
- https://download.springsource.com/release/STS/3.9.14.RELEASE/dist/e4.17/spring-tool-suite-3.9.14.RELEASE-e4.17.0-macosx-cocoa-x86_64.dmg
- https://download.springsource.com/release/STS/3.9.14.RELEASE/dist/e4.17/spring-tool-suite-3.9.14.RELEASE-e4.17.0-linux-gtk-x86_64.tar.gz

- 윈도우 기본 압축푸는 앱은 긴
경로 처리에 어려움이 있으므로
반디집 같은 프로그램을
다운로드 받아서 압축을 풀어야
문제없이 설치됨
- C: 드라이브에 압축푸는 것이
 좋음

Tomcat 다운로드

- JDK버전에 맞게 다운로드 및 압축해제

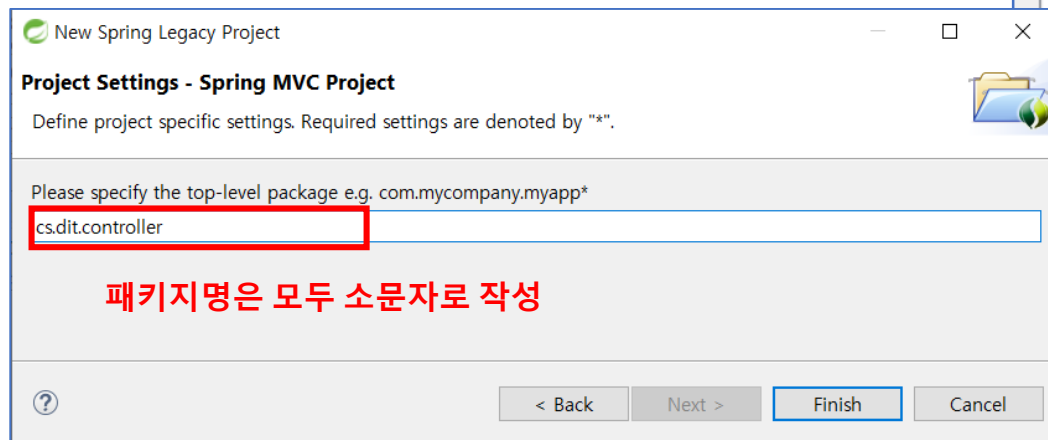
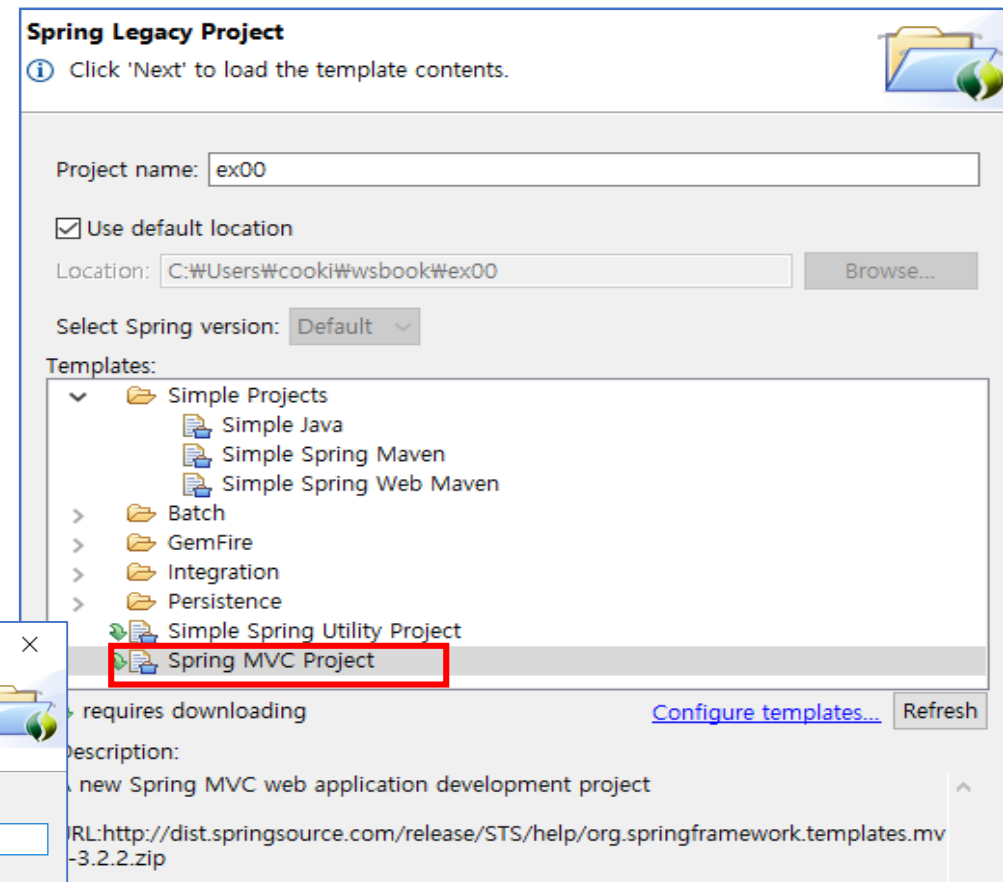
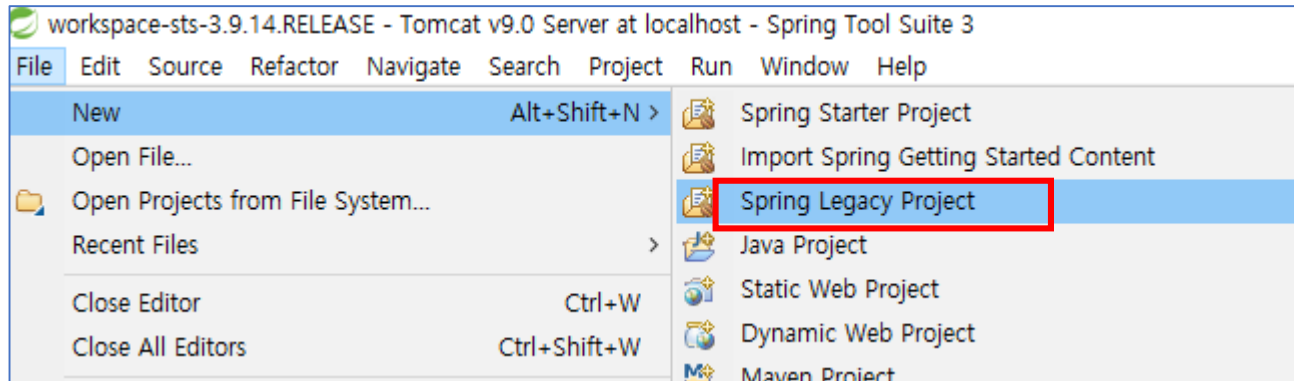
Apache Tomcat Versions

Apache Tomcat[®] is an open source software implementation of the Java Servlet and JavaServer Pages technologies. Different versions of Apache Tomcat are available for different versions of the Servlet and JSP specifications. The mapping between [the specifications](#) and the respective Apache Tomcat versions is:

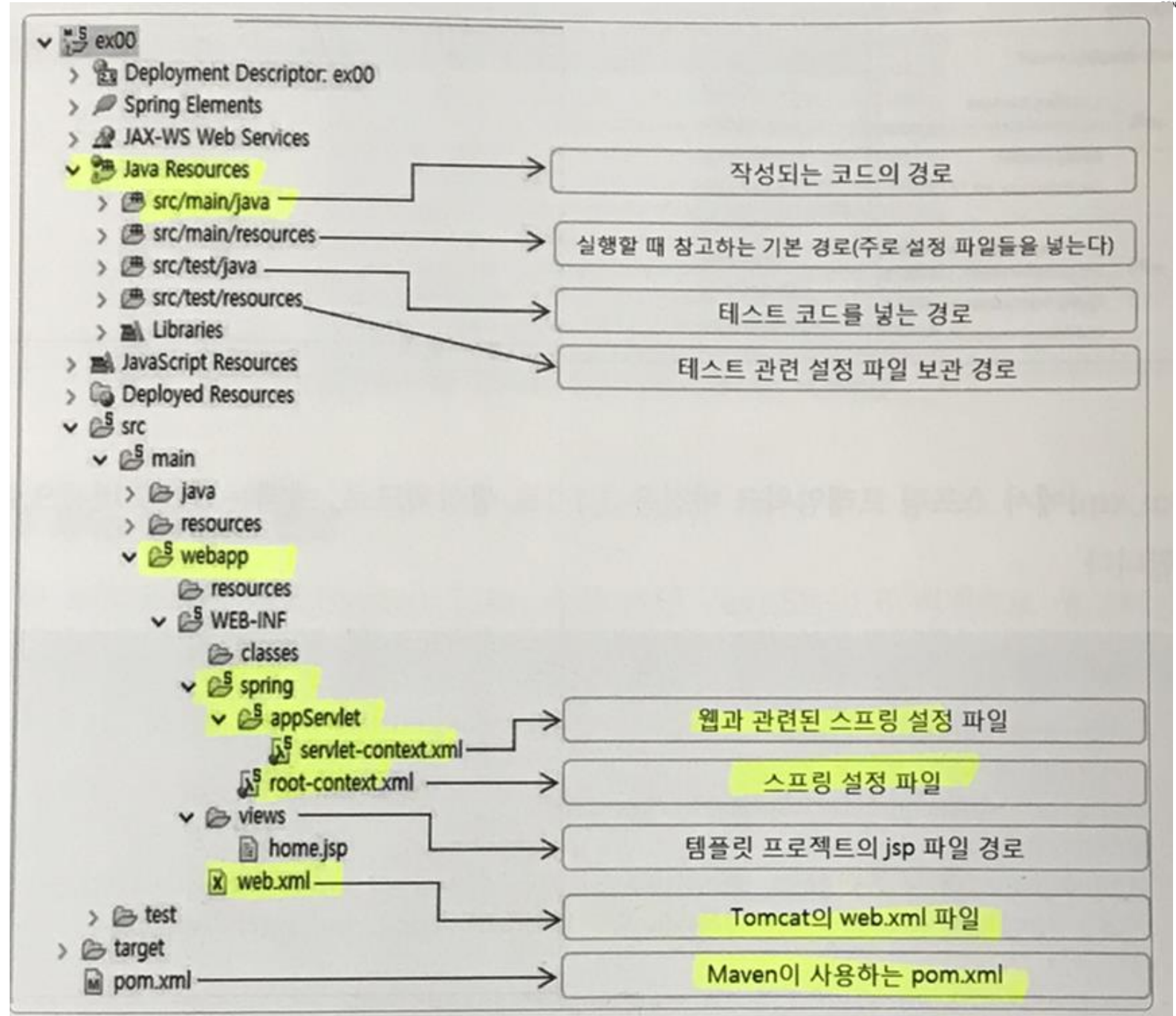
Servlet Spec	JSP Spec	EL Spec	WebSocket Spec	JASPIC Spec	Apache Tomcat Version	Latest Released Version	Supported Java Versions
4.0	2.3	3.0	1.1	1.1	9.0.x	9.0.10	8 and later
3.1	2.3	3.0	1.1	1.1	8.5.x	8.5.32	7 and later
3.1	2.3	3.0	1.1	N/A	8.0.x (superseded)	8.0.53 (superseded)	7 and later
3.0	2.2	2.2	1.1	N/A	7.0.x	7.0.90	6 and later (7 and later for WebSocket)
2.5	2.1	2.1	N/A	N/A	6.0.x (archived)	6.0.53 (archived)	5 and later
2.4	2.0	N/A	N/A	N/A	5.5.x (archived)	5.5.36 (archived)	1.4 and later
2.3	1.2	N/A	N/A	N/A	4.1.x (archived)	4.1.40 (archived)	1.3 and later
2.2	1.1	N/A	N/A	N/A	3.3.x (archived)	3.3.2 (archived)	1.1 and later

스프링 프로젝트 생성

- Spring 프로젝트 중에서 'Spring Legacy Project' 생성

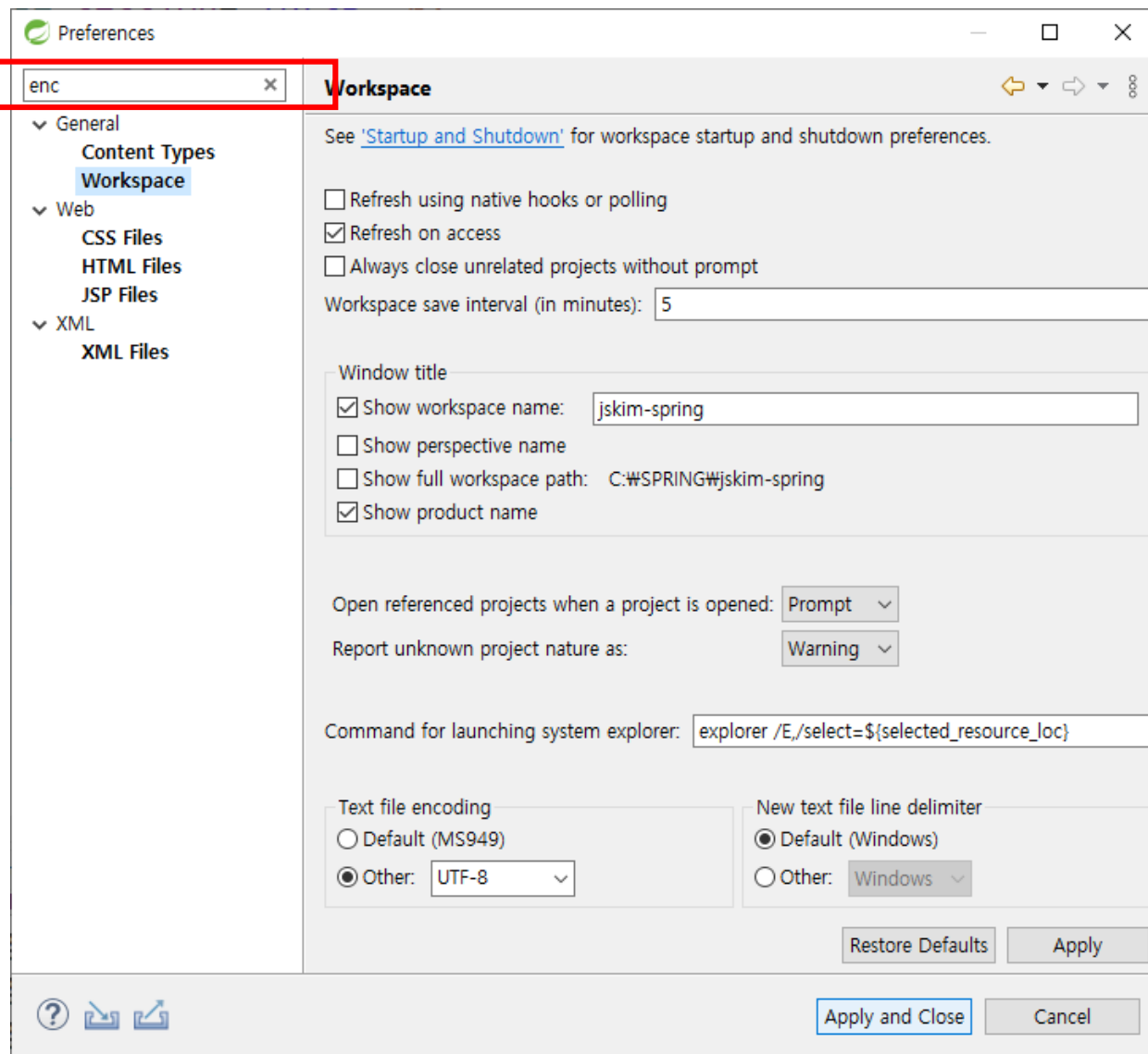


프로젝트 구조



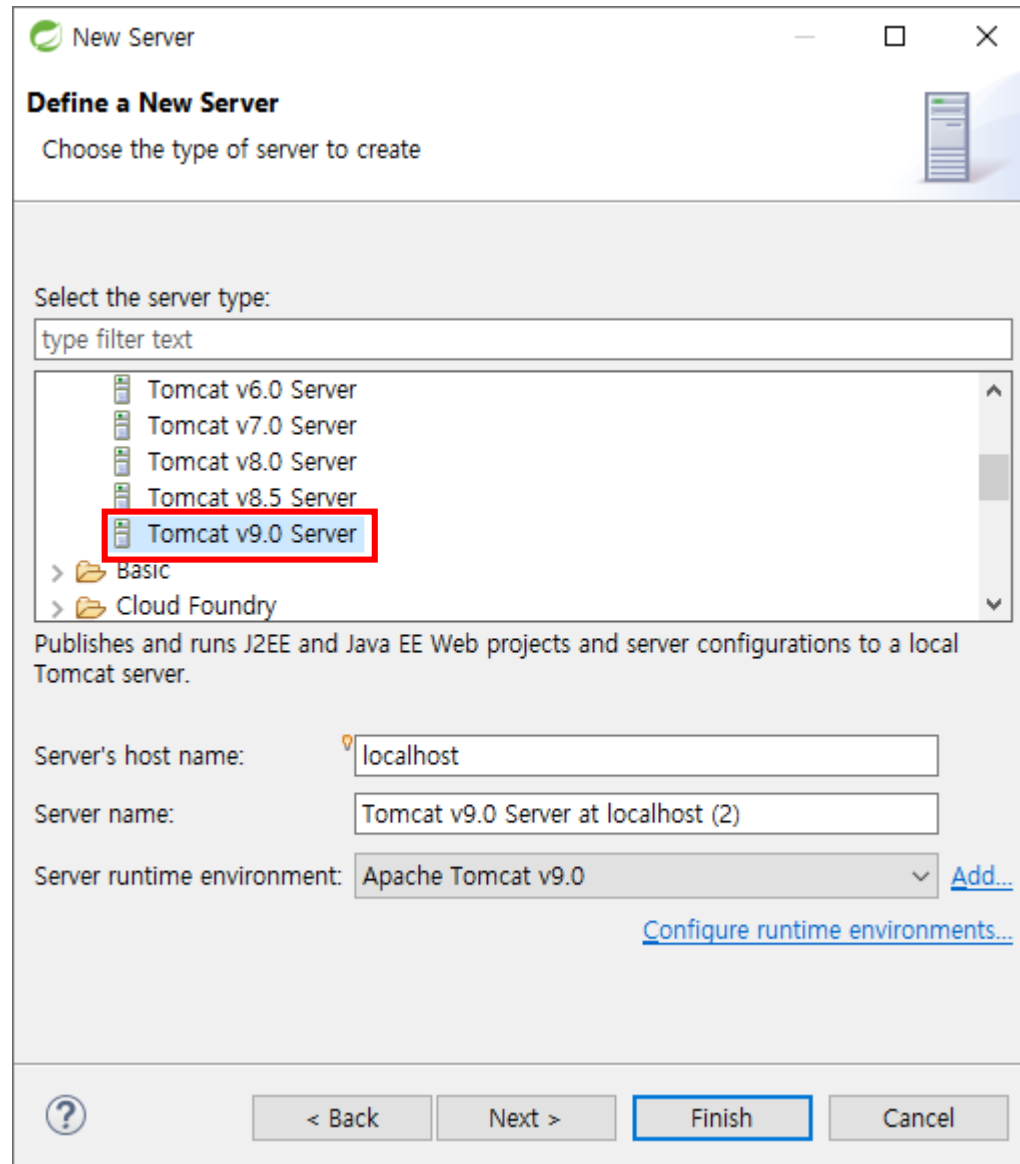
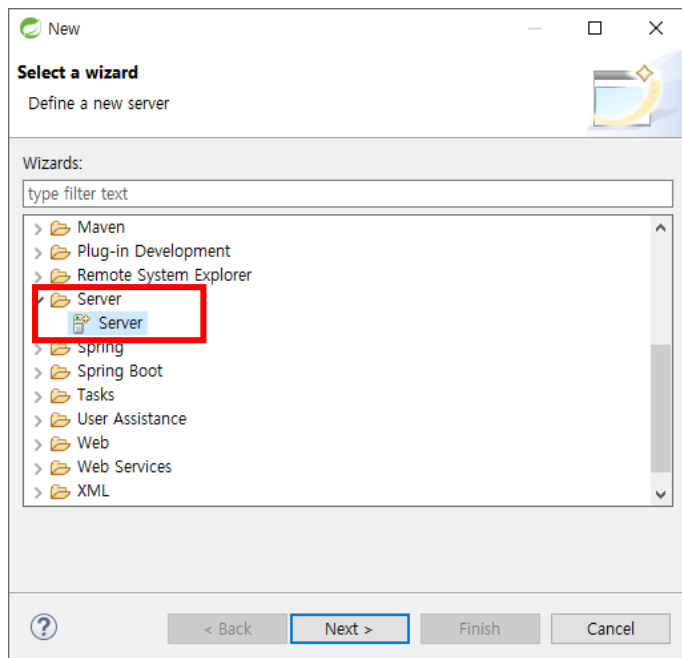
Preference 설정

검색키워드로 enc 입력
모든 encoding 방법을 utf-8로 변경

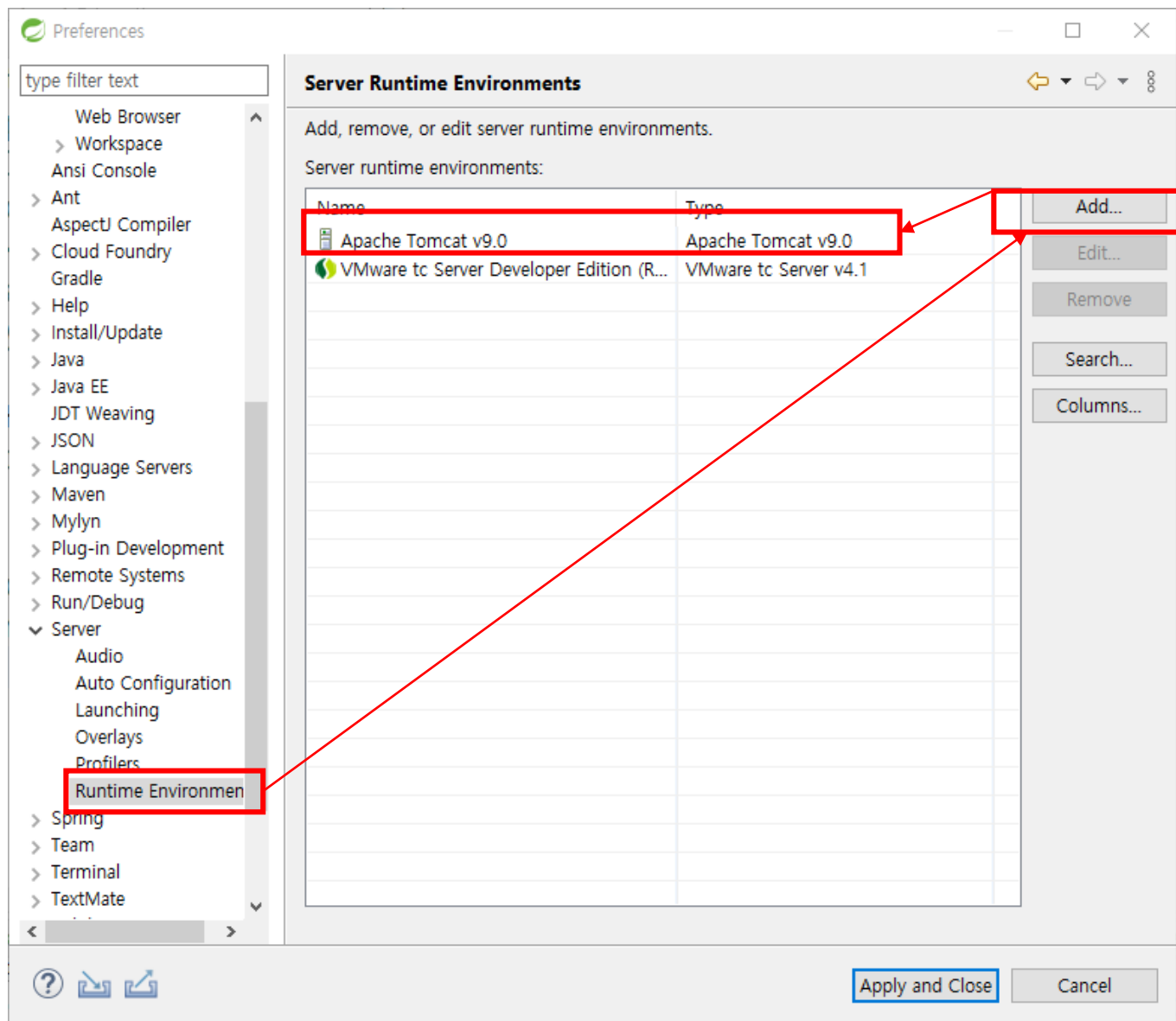


Server 설정

File – New – Other

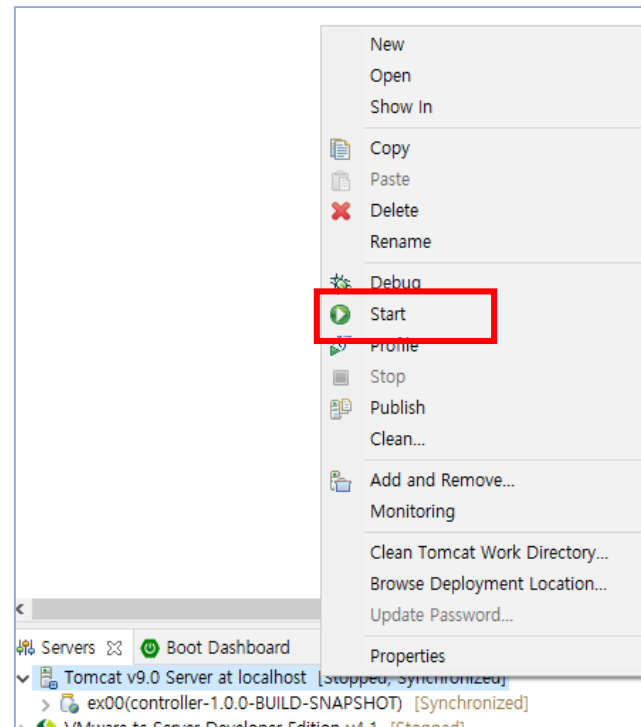
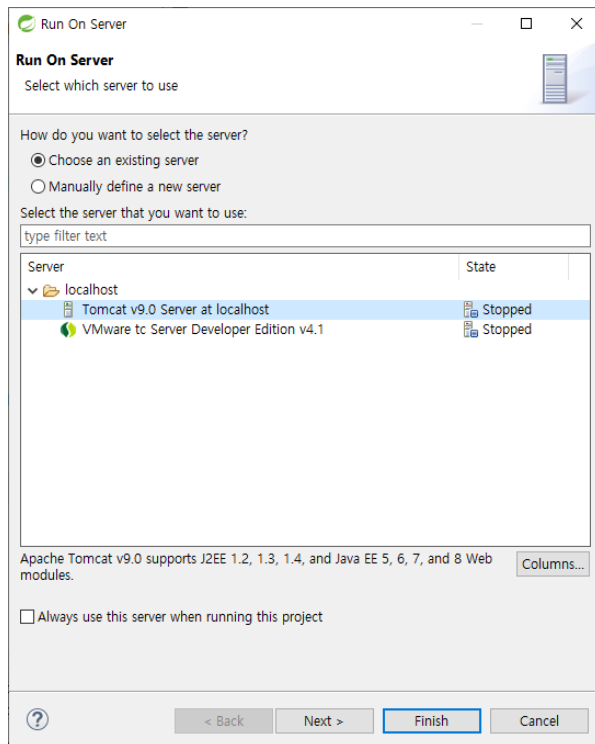


Server 설정



프로젝트 실행

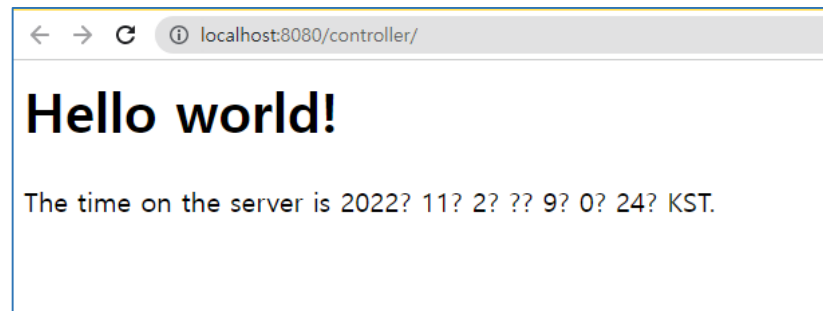
- 실행은 프로젝트 단위로 진행
- 프로젝트명을 선택한 후 오른쪽 마우스를 클릭하여 [Run As] – [Run on Server] – [Manually define a new server] – Apache – Tomcat9을 선택하여 실행



한번 실행한 뒤에는 이렇게 실행할 수 있다.

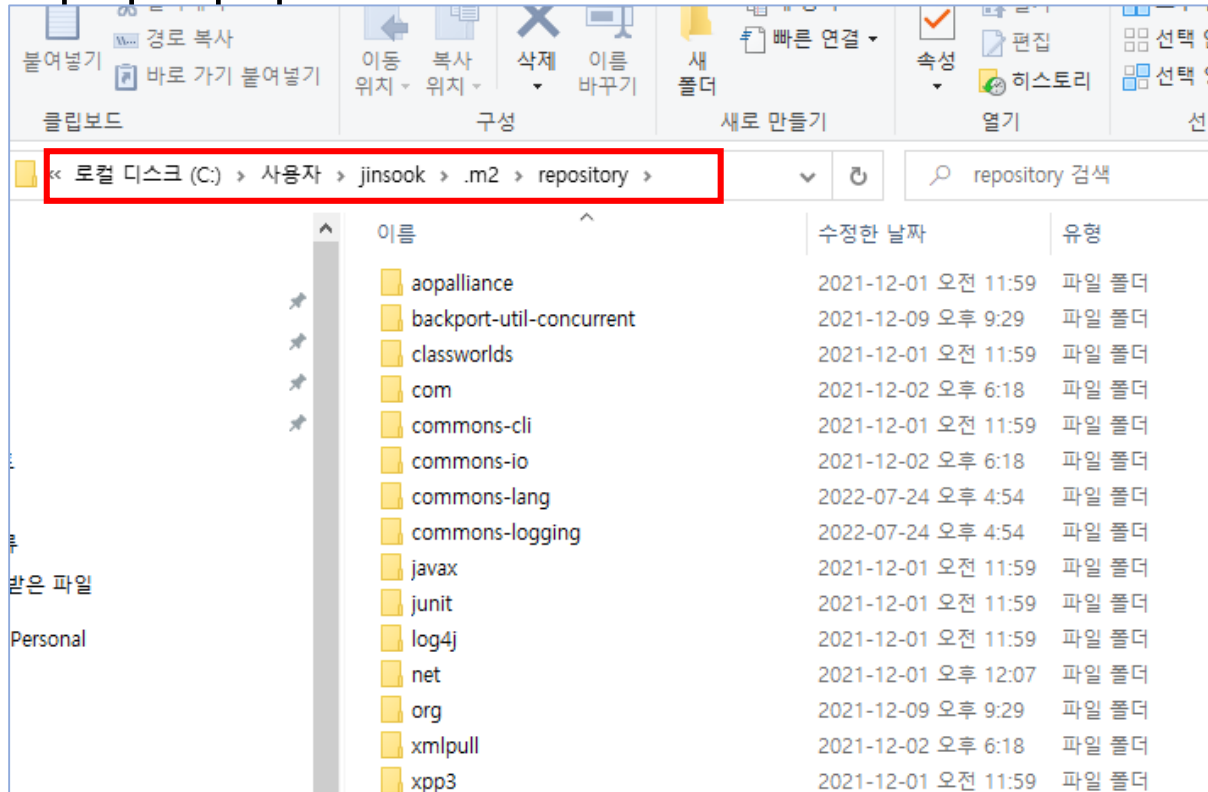
프로젝트 실행

- 방화벽 액세스 허용



실행 시 문제 발생

- Invalid loc header 등 문제발생 시에는 대부분의 경우 라이브러리의 문제
 1. STS를 종료
 2. 사용자-.m2 아래의 repository 폴더를 삭제
 3. STS 다시 시작



- 자동으로 다운로드되는 파일들이 저장되는 위치이다.
- 문제가 생기면 .m2 내의 repository를 삭제하고 다시 프로젝트를 생성하면 된다.

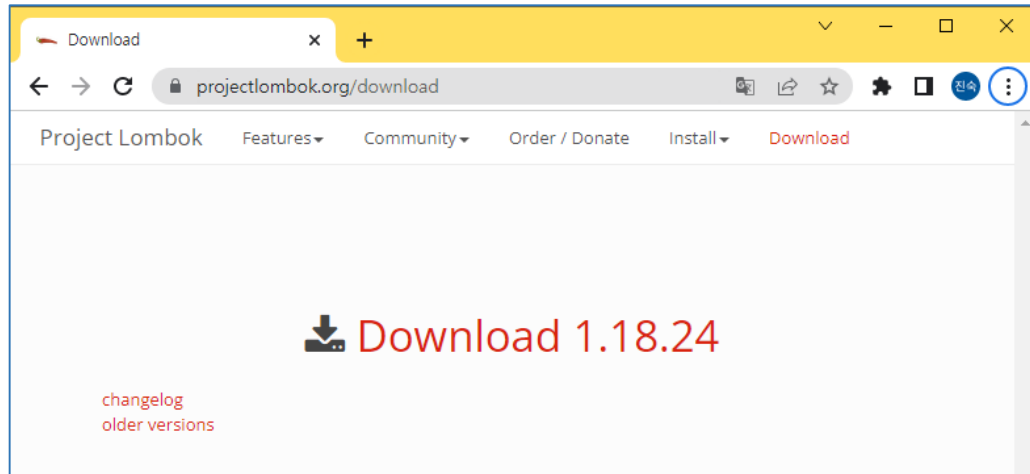
프로젝트의 실행 확인 및 경로 조정

- Tomcat을 이용한 프로젝트 실행
- 경로를 '/'로 조정
 - Tomcat의 'Modules'를 이용

```
정보: Initializing Spring FrameworkServlet 'appServlet'
INFO : org.springframework.web.servlet.DispatcherServlet - FrameworkServlet 'appServlet': initialization started
INFO : org.springframework.web.context.support.XmlWebApplicationContext - Refreshing WebApplicationContext for namespace 'appSe
INFO : org.springframework.beans.factory.xml.XmlBeanDefinitionReader - Loading XML bean definitions from ServletContext resourc
INFO : org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor - JSR-330 'javax.inject.Inject' annota
INFO : org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping - Mapped "{[/],methods=[GET]}" onto p
INFO : org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter - Looking for @ControllerAdvice: WebA
INFO : org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter - Looking for @ControllerAdvice: WebA
INFO : org.springframework.web.servlet.handler.SimpleUrlHandlerMapping - Mapped URL path [/resources/**] onto handler 'org.spr
INFO : org.springframework.web.servlet.DispatcherServlet - FrameworkServlet 'appServlet': initialization completed in 832 ms
6월 14, 2018 11:34:34 오전 org.apache.coyote.AbstractProtocol start
정보: Starting ProtocolHandler ["http-nio-8080"]
6월 14, 2018 11:34:34 오전 org.apache.coyote.AbstractProtocol start
정보: Starting ProtocolHandler ["ajp-nio-8009"]
6월 14, 2018 11:34:34 오전 org.apache.catalina.startup.Catalina start
정보: Server startup in 6577 ms
INFO : org.zerock.controller.HomeController - Welcome home! The client locale is ko_KR.
```

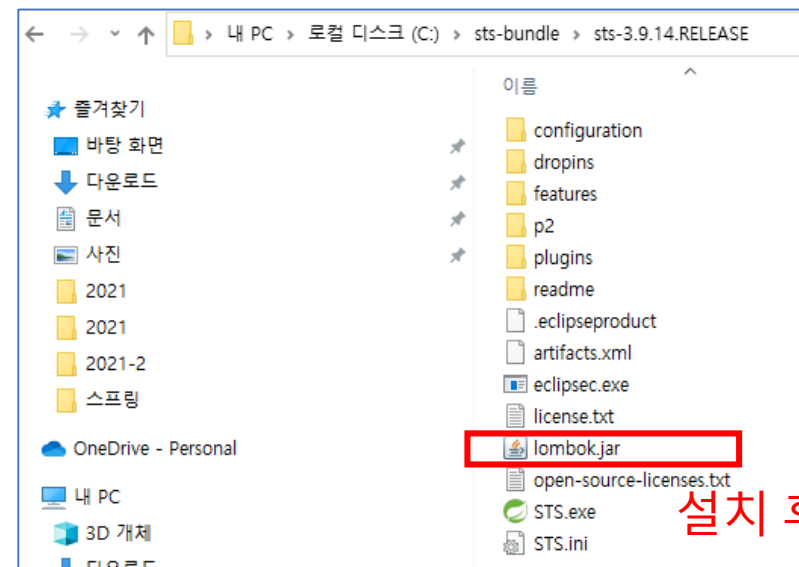
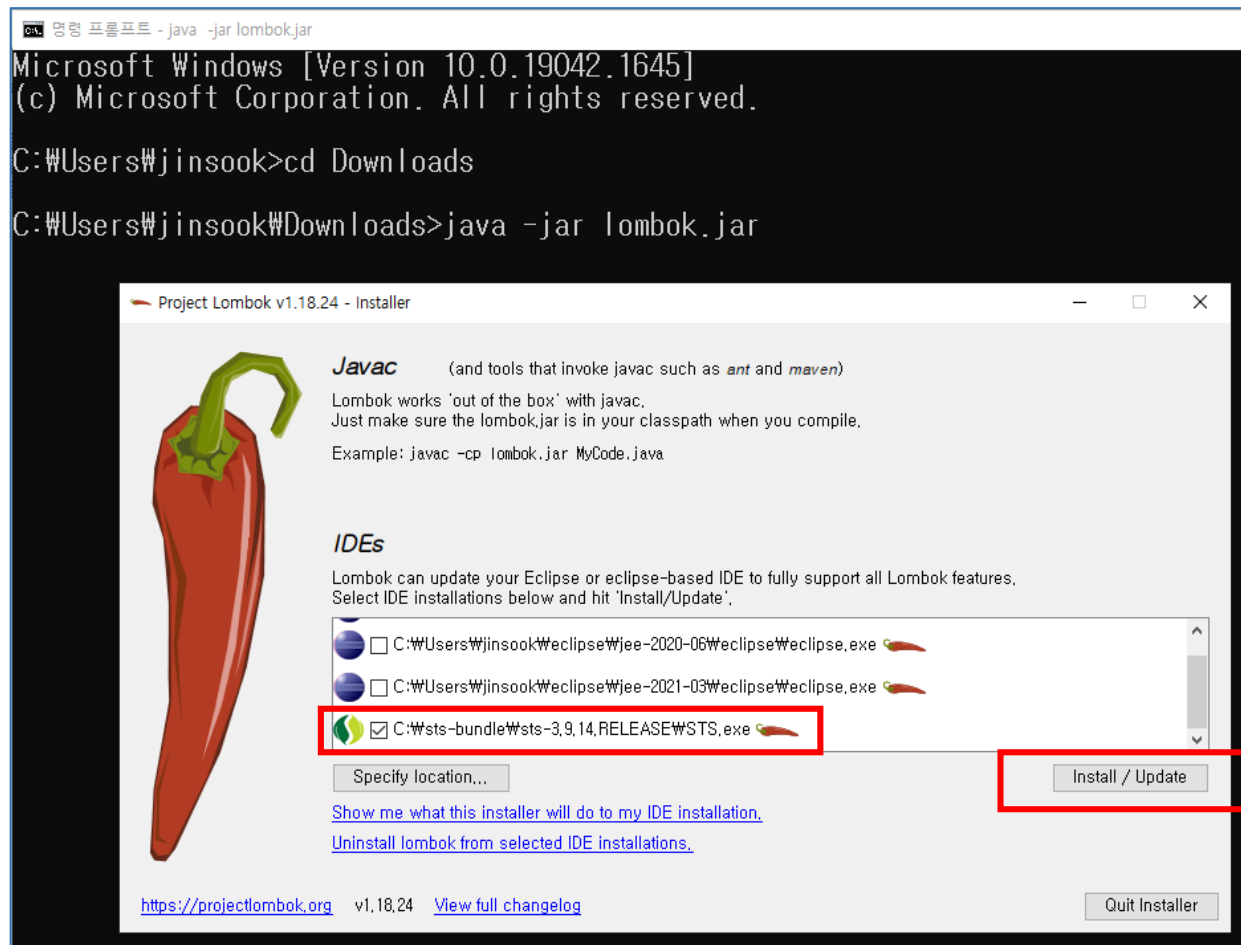
Lombok 라이브러리 설치

- Lombok 기능(v.1.8.24)
 - 컴파일 시점에 어노테이션으로 getter/setter, 생성자, toString()등을 자동으로 생성(편리함)
- 이클립스 전체에서 사용할 수 있도록 별도 설치
- <https://projectlombok.org/download>
- 다운로드 후에 Eclipse에 추가 설치 필요



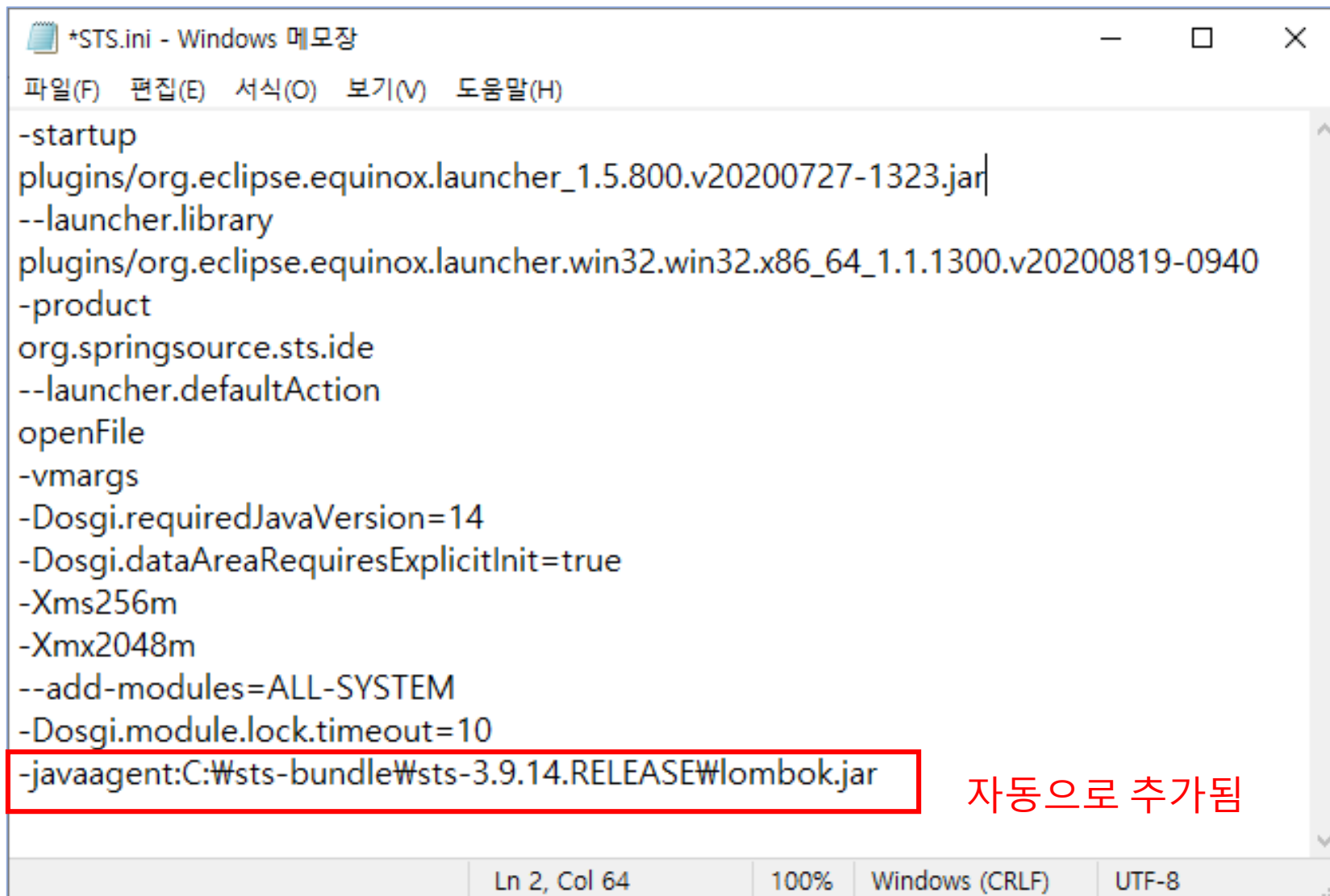
Lombok 라이브러리 설치

Lombok 설치 시에 STS는 반드시
종료한 뒤 작업하세요!!!



설치 후 추가됨

STS.ini 파일



```
*STS.ini - Windows 메모장
파일(F)  편집(E)  서식(O)  보기(V)  도움말(H)

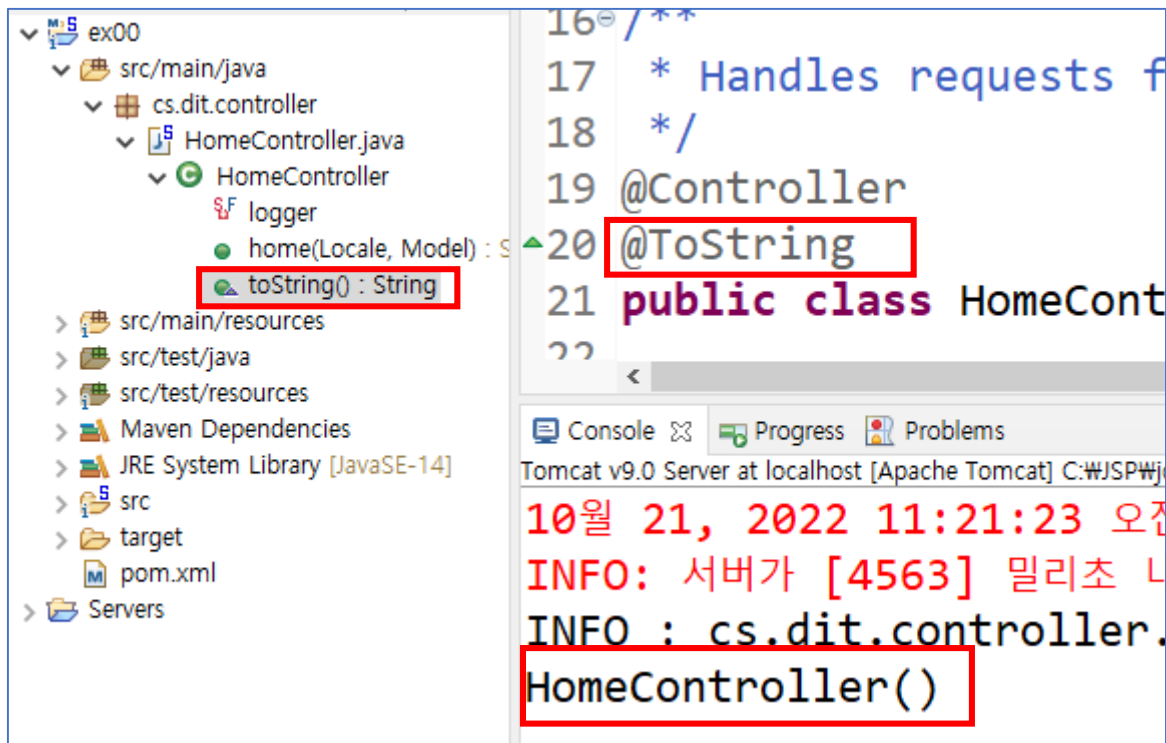
-startup
plugins/org.eclipse.equinox.launcher_1.5.800.v20200727-1323.jar|
--launcher.library
plugins/org.eclipse.equinox.launcher.win32.win32.x86_64_1.1.1300.v20200819-0940
-product
org.springframework.sts.ide
--launcher.defaultAction
openFile
-vmargs
-Dosgi.requiredJavaVersion=14
-Dosgi.dataAreaRequiresExplicitInit=true
-Xms256m
-Xmx2048m
--add-modules=ALL-SYSTEM
-Dosgi.module.lock.timeout=10
-javaagent:C:\wsts-bundle\wsts-3.9.14.RELEASE\lombok.jar

Ln 2, Col 64    100%    Windows (CRLF)    UTF-8
```

자동으로 추가됨

Lombok 기능

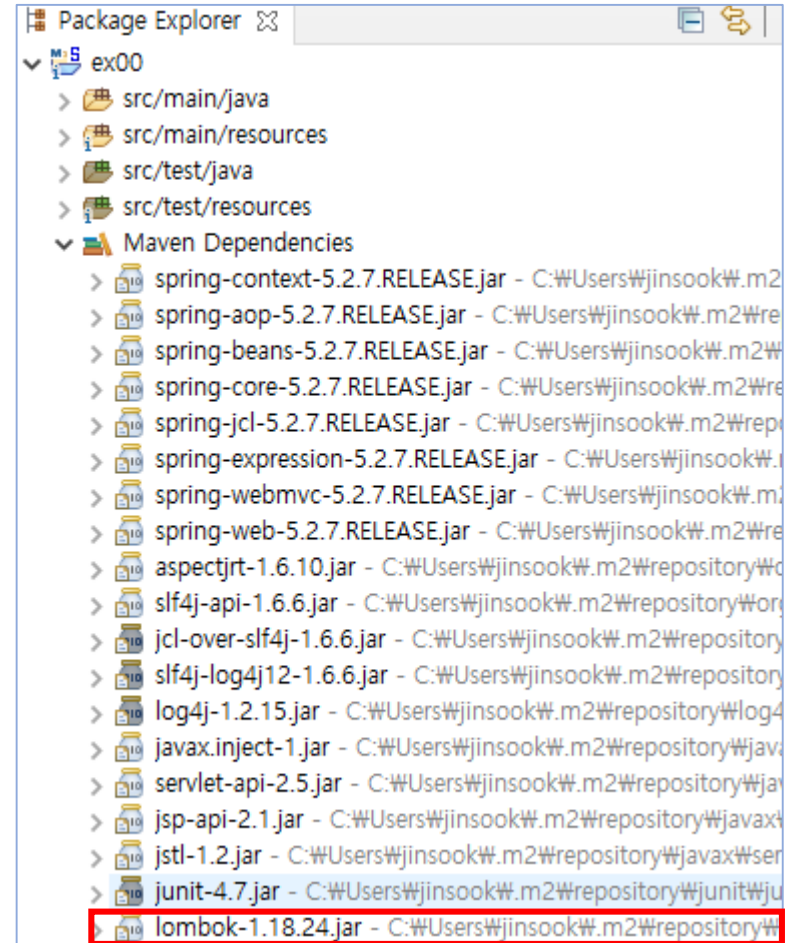
- 소스코드 완성 기능
- 생성자 관련 기능



pom.xml 파일 추가

- Maven Repository에서 다음과 같은 의존성 태그를 복사
- pom.xml 파일에 붙여넣기
- 오른쪽 마우스의 update project 메뉴로 업데이트 할 것

```
<dependency>  
  <groupId>org.projectlombok</groupId>  
  <artifactId>lombok</artifactId>  
  <version>1.18.24</version>  
  <scope>provided</scope>  
</dependency>
```



다운로드 확인

pom.xml 수정

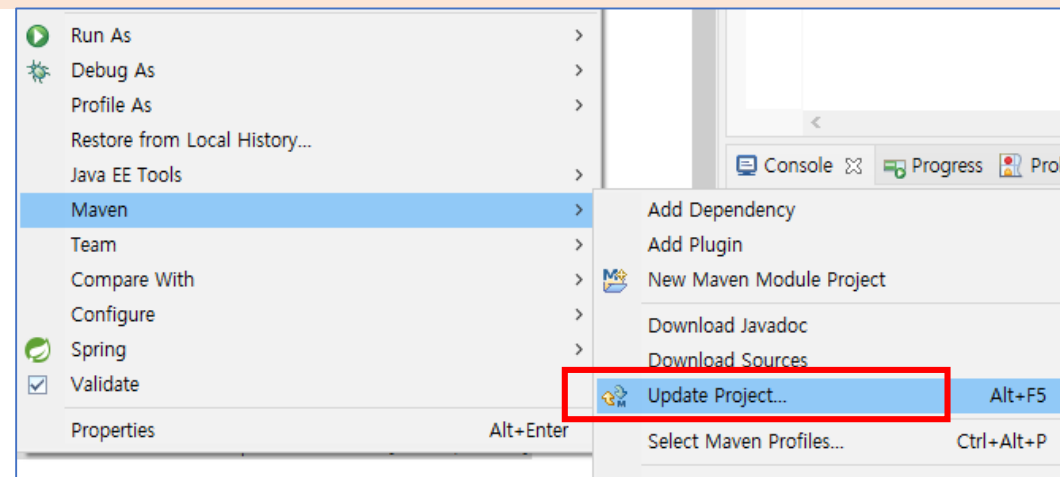
- 스프링 프로젝트의 버전 변경 => **5.2.7**
- Java 버전 변경 => **14**
- Junit 버전 => **4.12**
- Log4j 버전 => **1.2.17**
- Lombok 버전 => **1.18.24**
- servlet 버전 => **4.0.1**
- Jsp 버전 => **2.3.3**

애플도 당했다! 최악의 보안사태 "Log4J" 설명

<https://www.youtube.com/watch?v=kwS3twdVsko>

```
136<plugin>
137    <groupId>org.apache.maven.plugins</groupId>
138    <artifactId>maven-compiler-plugin</artifactId>
139    <version>2.5.1</version>
140<configuration>
141    <source>14</source>
142    <target>14</target>
143    <compilerArgument>-Xlint:all</compilerArgument>
144    <showWarnings>true</showWarnings>
145    <showDeprecation>true</showDeprecation>
146</configuration>
147</plugin>
```

pom.xml 파일을 수정하고 난 후에는 update project를 해야 함



프로젝트 실행

```
1 package cs.dit.controller;
2
3 import java.text.DateFormat;
4
5
6
7
8
9
10
11
12
13
14 @Controller
15 public class HomeController {
16
17     private static final Logger logger = LoggerFactory.getLogger(HomeController.class);
18
19     /**
20      * Simply selects the home view to render by returning its name.
21      */
22     @RequestMapping(value = "/", method = RequestMethod.GET)
23     public String home(Locale locale, Model model) {
24         logger.info("Welcome home! The client locale is {}.", locale);
25
26         Date date = new Date();
27         DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.LONG, locale);
28
29         String formattedDate = dateFormat.format(date);
30
31         model.addAttribute("serverTime", formattedDate );
32
33         return "home";
34     }
35 }
```


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 https://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 </web-app>
```

클라이언트의 모든 요청을 받는 서블릿

servlet-context.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans:beans xmlns="http://www.springframework.org/schema/mvc"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:beans="http://www.springframework.org/schema/beans"
5   xmlns:context="http://www.springframework.org/schema/context"
6   xsi:schemaLocation="http://www.springframework.org/schema/mvc https://www.springframework.org/schema/mvc/spring-mvc.xsd
7     http://www.springframework.org/schema/beans https://www.springframework.org/schema/beans/spring-beans.xsd
8     http://www.springframework.org/schema/context https://www.springframework.org/schema/context/spring-context.xsd">
9
10  <!-- DispatcherServlet Context: defines this servlet's request-processing infrastructure -->
11
12  <!-- Enables the Spring MVC @Controller programming model -->
13  <annotation-driven />
14
15  <!-- Handles HTTP GET requests for /resources/** by efficiently serving up static resources in the ${webappRoot}/resources directory -->
16  <resources mapping="/resources/**" location="/resources/" />
17
18  <!-- Resolves views selected for rendering by @Controllers to .jsp resources in the /WEB-INF/views directory -->
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="cs.dit.controller" />
25
26 </beans:beans>
```

servlet-context.xml

- <context:component-scan>
 - **@Component**를 통해 자동으로 bean을 등록하고 **@Autowired**로 의존관계를 주입 받는 어노테이션을 클래스에서 선언하고 사용했을 경우, 해당 클래스가 위치한 패키지를 scan하기 위한 설정에 필요

```
<context:component-scan base-package="cs.dit.controller" />
```

Spring Web MVC 처리 과정

