1. 개발을 위한 준비
   - **JDK 1.8버전 이상 설치.**
   - **Tomcat 설치 및 연동**
   - **STS(Spring Tool Suite) 설치**
   - **오라클 데이터베이스 / SQL Developer 설치 및 설정**
   - **스프링 프로젝트 생성 및 라이브러리 추가**
   - **MyBatis / mybatis-spring 설정**
   - **스프링 MVC 개발 설정**

1.1.1 JDK 11버전 설치

1.1.2 Tomcat 9(8.5) 서버 설치

Tomcat을 사용할 때에는 항상 http://tomcat.apache.org/whichversion.html 문서를 통해서 자신의 환경에 맞는 버전을 이용해야 한다.

Apache Tomcat Versions

Apache Tomcat® is an open source software implementation of a subset of the Jakarta EE (formally Java EE) technologies. Different versions of Apache Tomcat are available for different versions of the specifications. The mapping between the specifications and the respective Apache Tomcat versions is:

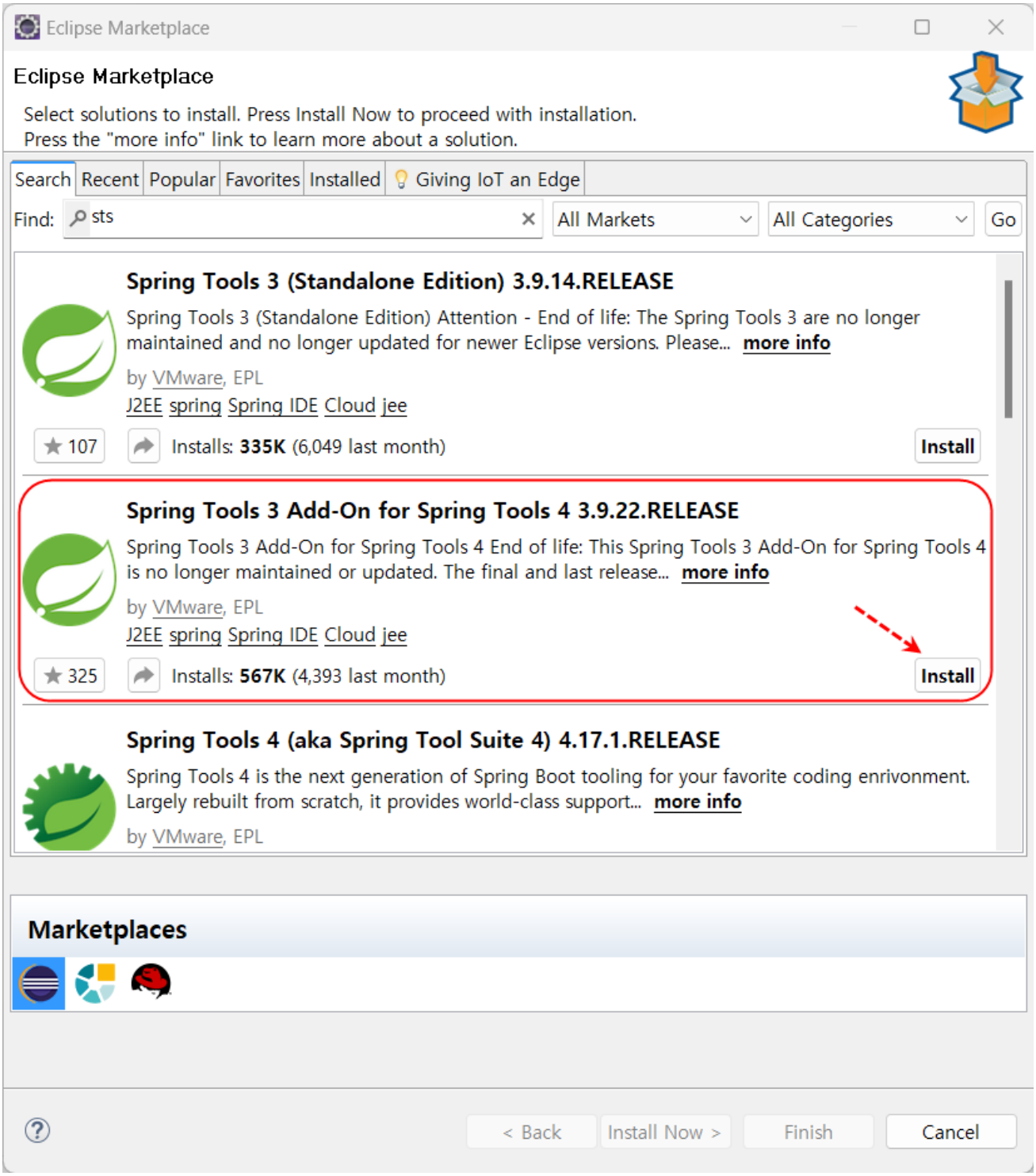
Servlet Spec	JSP Spec	EL Spec	WebSocket Spec	Authentication (JASPIC) Spec	Apache Tomcat Version	Latest Released Version	Supported Java Versions
6.1	4.0	6.0	TBD	TBD	11.0.x	11.0.0-M1 (alpha)	17 and later
6.0	3.1	5.0	2.1	3.0	10.1.x	10.1.5	11 and later
5.0	3.0	4.0	2.0	2.0	10.0.x (superseded)	10.0.27 (superseded)	8 and later
4.0	2.3	3.0	1.1	1.1	9.0.x	9.0.71	8 and later
3.1	2.3	3.0	1.1	1.1	8.5.x	8.5.85	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3.1	2.3	3.0	1.1	N/A	8.0.x (superseded)	8.0.53 (superseded)	7 and later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 (archived)	7.0.109 (archived)	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

1.1.3 Eclipse를 이용하는 경우 스프링 플러그인(STS) 설치

1) 이클립스에 스프링을 위한 STS 플러그인 설치

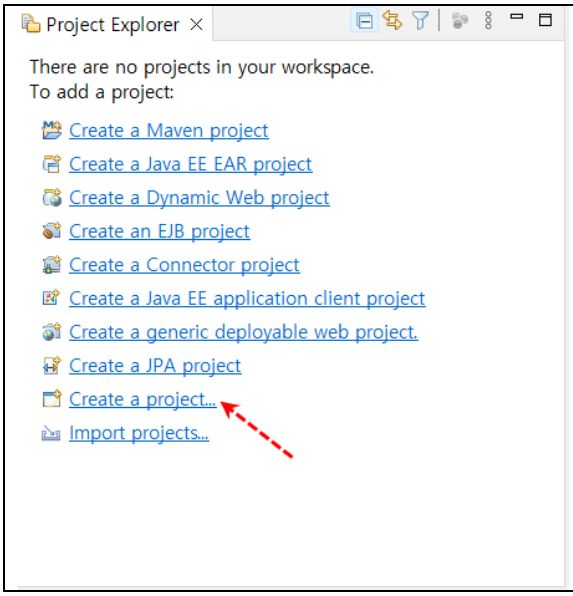
개발 도구에 STS 플러그인을 설치하는 방법은 다음과 같다.

① 이클립스 메뉴에 **[Help] - [Eclipse Marketplace]** 메뉴를 차례로 선택한다.



1.2 스프링 프로젝트 생성

이클립스(STS)에서 스프링 프로젝트를 생성하는 방식은 1) 처음부터 스프링 프로젝트를 지정하고 생성하는 방법, 2) Maven 또는 Gradle 프로젝트를 생성한 후 프레임워크를 추가하는 방식, 3) 직접 프레임워크 라이브러리를 추가하는 방식이 있다.



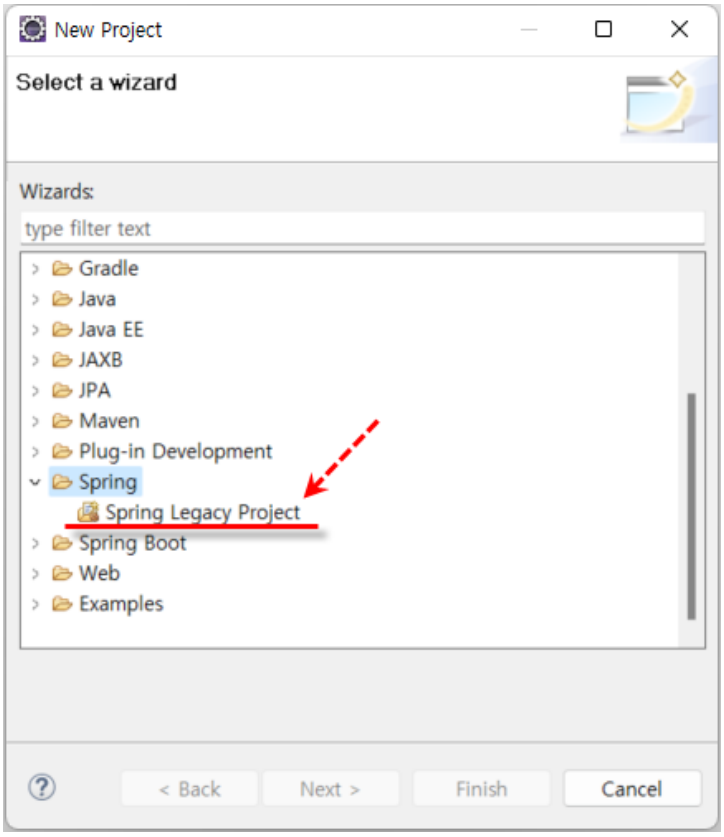
[프로젝트 생성 시 에러 발생 시]

※ **java.lang.ExceptionInInitializerError** 오류 발생 시 eclipse 설치 폴더 내에 eclipse.ini 파일을 열어 아래와 같이 수정한다.

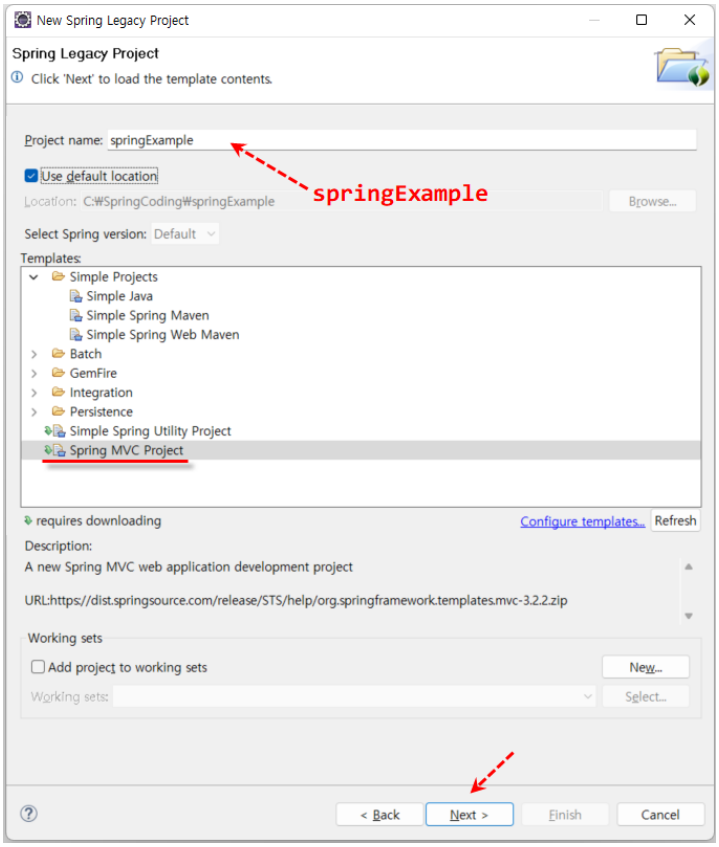
-vm

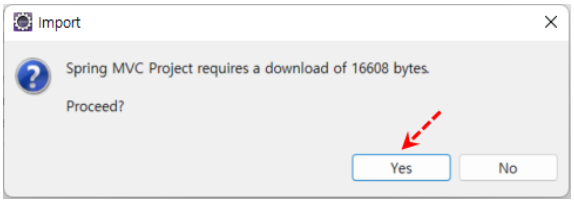
C:/webdeveloper/java/jdk-11/bin

-Dosgi.requiredJavaVersion=11 버전을 설정되어 있는지 확인한다.

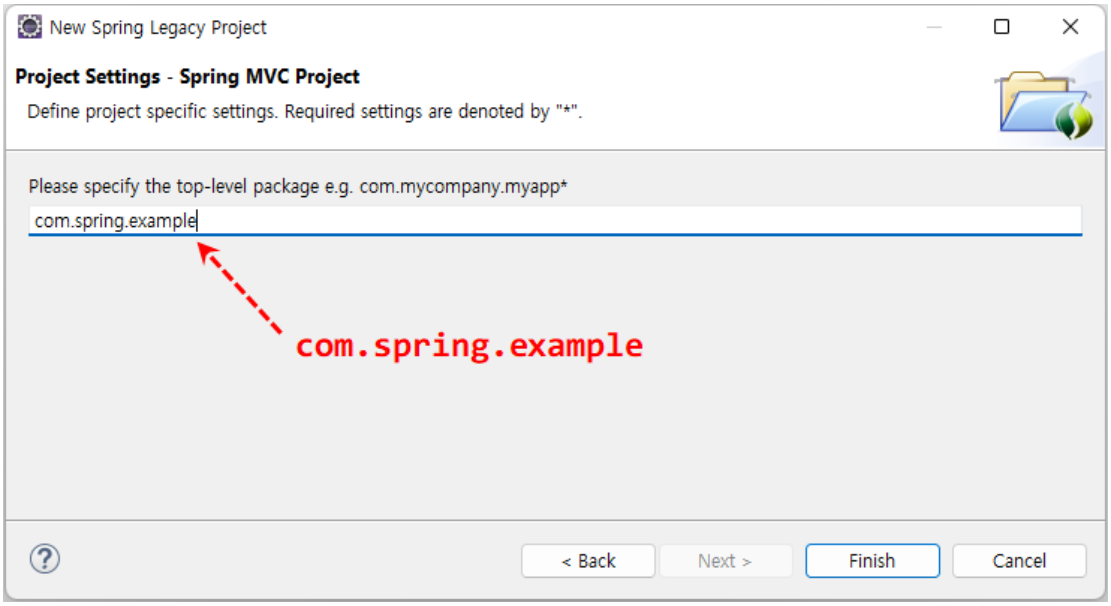


[New] - [Spring] - [Spring Legacy Project] 메뉴를 이용하면 아래 화면과 같이 여러 종류의 스프링 프로젝트를 Maven 기반으로 생성할 수 있다.



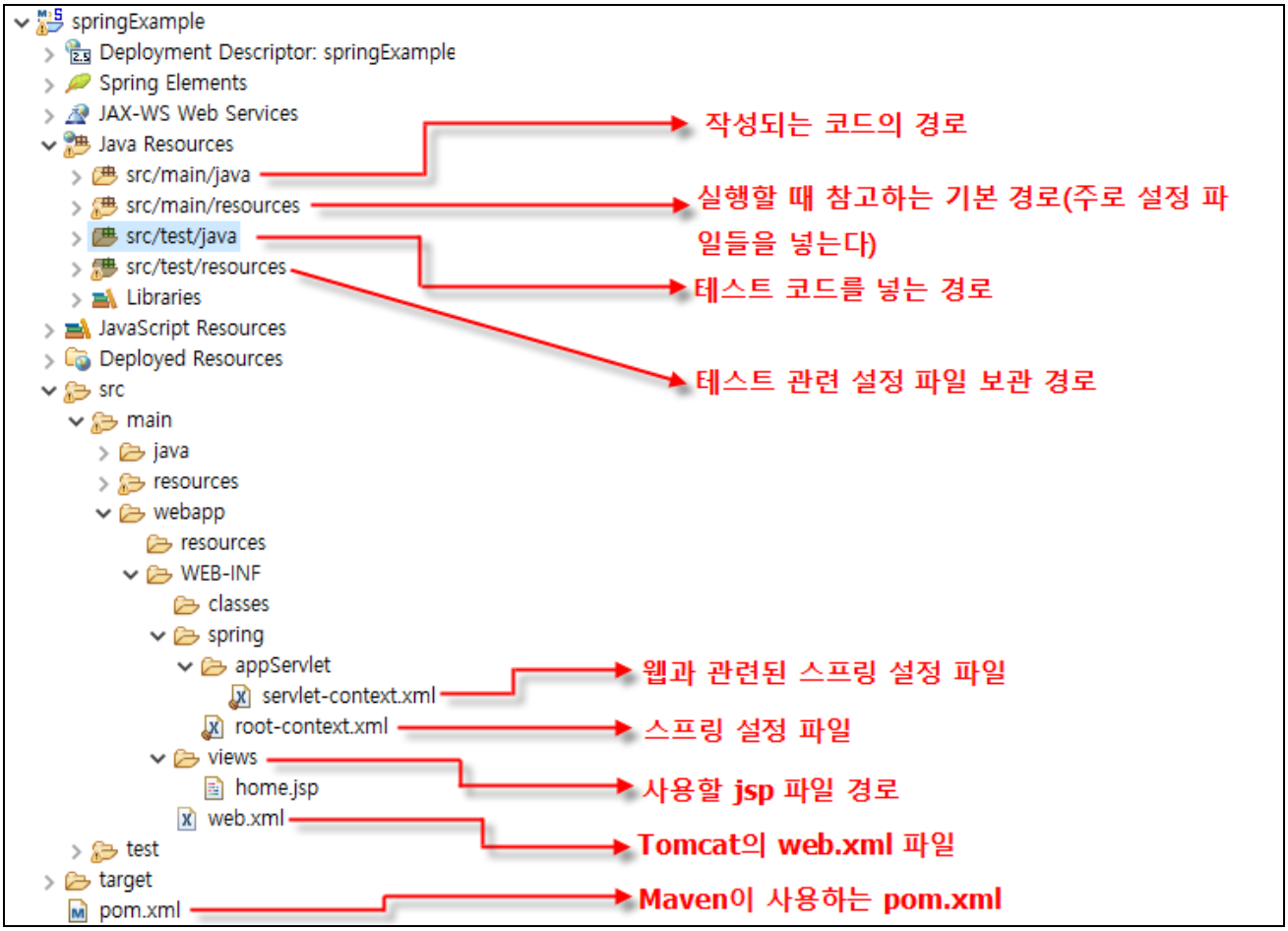


프로젝트는 'Spring MVC Project'를 이용해서 생성한다. 패키지명 'com.spring.example'로 지정한다.

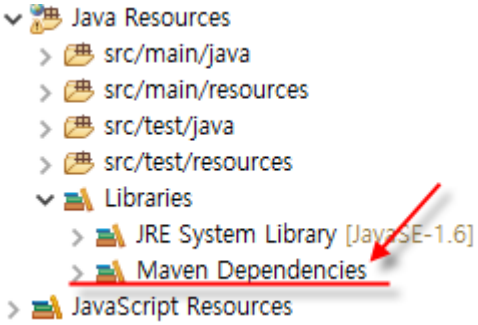


1.2.1 스프링 프레임워크 버전 변경

프로젝트를 최초로 생성하면 필요한 코드와 라이브러리를 다운로드하게 된다. 다운로드 하는 라이브러리들은 사용자 폴더 내 'C:₩Users₩각자의 사용자명₩.m2'라는 이름의 폴더를 이용한다. '.m2'폴더에 생성된 repository 폴더 안에는 프로젝트 생성 시 다운로드된 파일들이 추가된다.



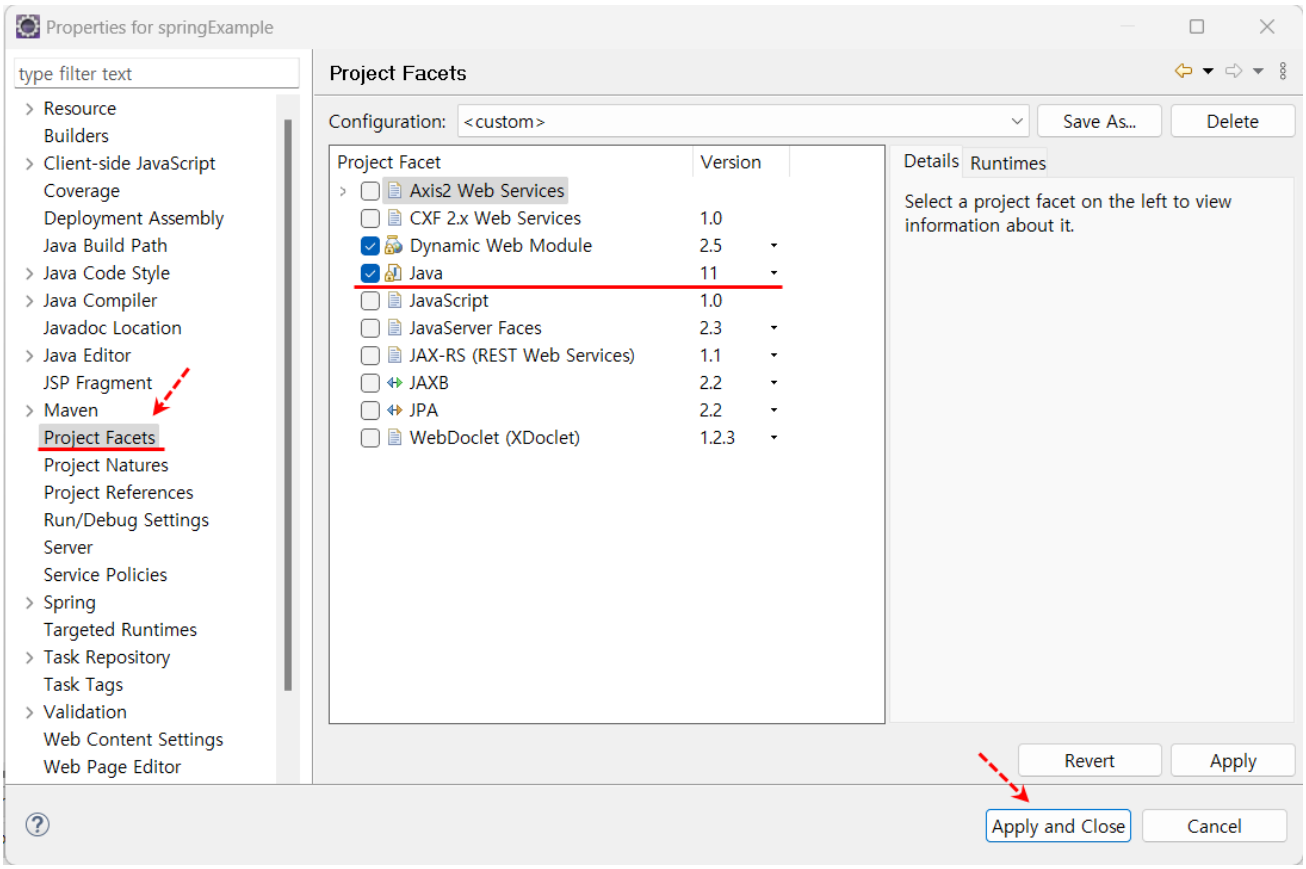
Spring Legacy Project 메뉴를 이용해서 생성하는 프로젝트는 스프링 버전은 3.x이고, JDK 환경 역시 1.6을 기준으로 작성되어 있다. 그래서 생성된 프로젝트의 라이브러리는 pom.xml 파일을 통해서 관리되므로 이를 수정한다.



'Maven Dependencies' 항목을 통해서 스프링 프레임워크 라이브러리들이 제대로 변경되었는지를 확인한다.

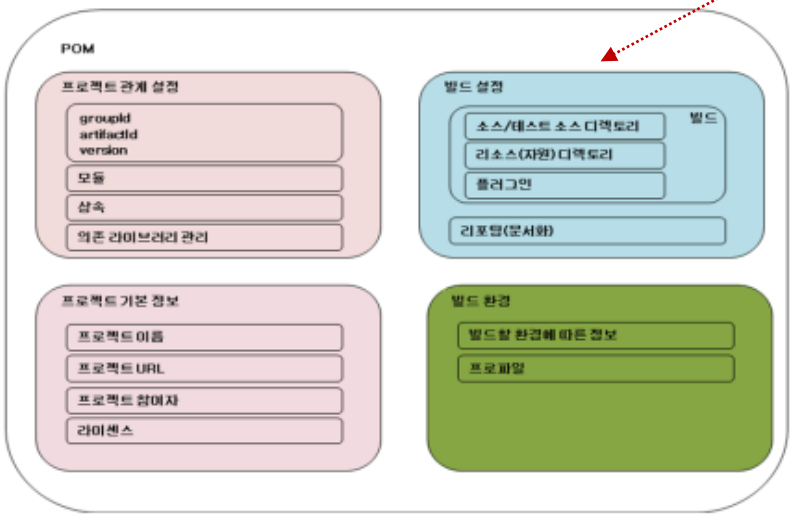
1.2.2 Java version 변경

생성된 프로젝트의 JRE System Library를 보면 'JavaSE-1.6'버전으로 생성되었으므로, 스프링 5.x 버전을 이용하고 싶다면 JDK 1.8을 사용하는 것이 좋다. maven-compiler-plugin의 코드 내용에서 1.6을 1.11로 변경한다.



Maven 관련 프로젝트를 생성하면 루트 디렉터리에 pom.xml 파일이 생성된다. **이 pom.xml 파일이 POM(Project Object Model) 즉, 프로젝트 객체 모델 정보를 담고 있는 파일로서, 여기에는 프로젝트 관리와 빌드에 필요한 환경 설정, 의존성 관리 등의 정보를 포함한다.**

다음 그림은 POM 파일의 구조를 보여준다.



빌드란?

- 소스코드 파일을 컴퓨터에서 실행할 수 있는 독립 소프트웨어 가공물로 변환하는 과정 또는 그에 대한 결과물 이다.
- 이를 좀더 쉽게 풀어 말하자면 우리가 작성한 소스코드(java), 프로젝트에서 쓰인 각각의 파일 및 자원 등(.xml, .jpg, .jar, .properties)을 JVM이나 톰캣 같은 WAS가 인식할 수 있는 구조로 패키징 하는 과정 및 결과물이라고 할 수 있다.

- 빌드 도구란 프로젝트 생성, 테스트 빌드, 배포 등의 작업을 위한 전용 프로그램.
- 단기간동안 계속해서 늘어나는 라이브러리 추가, 프로젝트를 진행하며 라이브러리의 버전 동기화의 어

려움을 해소하고자 등장.
- 초기의 java 빌드도구로 Ant를 많이 사용하였으나 최근 많은 빌드도구들이 생겨나 Maven이 많이 쓰였고, 현재는 Gradle이 많이 쓰인다.

· 의존성 설정

프로젝트가 의존하는 다른 프로젝트에 대한 정보는 <dependency>요소로 설정한다. <dependency> 요소 안에는 다음 서브 요소를 포함한다.

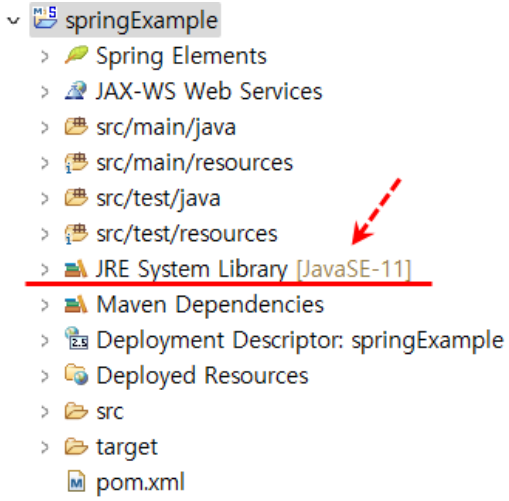
요소	설명
<groupId>	의존 프로젝트 그룹 ID
<artifactId>	의존 프로젝트 아티팩트 ID
<version>	버전

pom.xml에서 스프링 프레임워크 버전은 3.1.1로 생성되므로, 예제는 **5.3.18**버전으로 수정한다.

pom.xml

```
<project ...>
        <modelVersion>4.0.0</modelVersion>
        <groupId>com.spring</groupId>
        <artifactId>example</artifactId>
        <name>springExample</name>
        <packaging>war</packaging>
        <version>1.0.0-BUILD-SNAPSHOT</version>
        <properties>
                <java-version>11</java-version>
                <org.springframework-version>5.3.18</org.springframework-version>
                <org.aspectj-version>1.6.10</org.aspectj-version>
                <org.slf4j-version>1.6.6</org.slf4j-version>
        </properties>
        <dependencies>
```

실행된 뒤에는 프로젝트의 컴파일이나 실행 환경이 JDK 11를 기준으로 설정된 것을 확인할 수 있다.



pom.xml의 일부분

```
<plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.5.1</version>
        <configuration>
            <source>11</source>
            <target>11</target>
            <compilerArgument>-Xlint:all</compilerArgument>
            <showWarnings>true</showWarnings>
            <showDeprecation>true</showDeprecation>
        </configuration>
    </plugin>
```

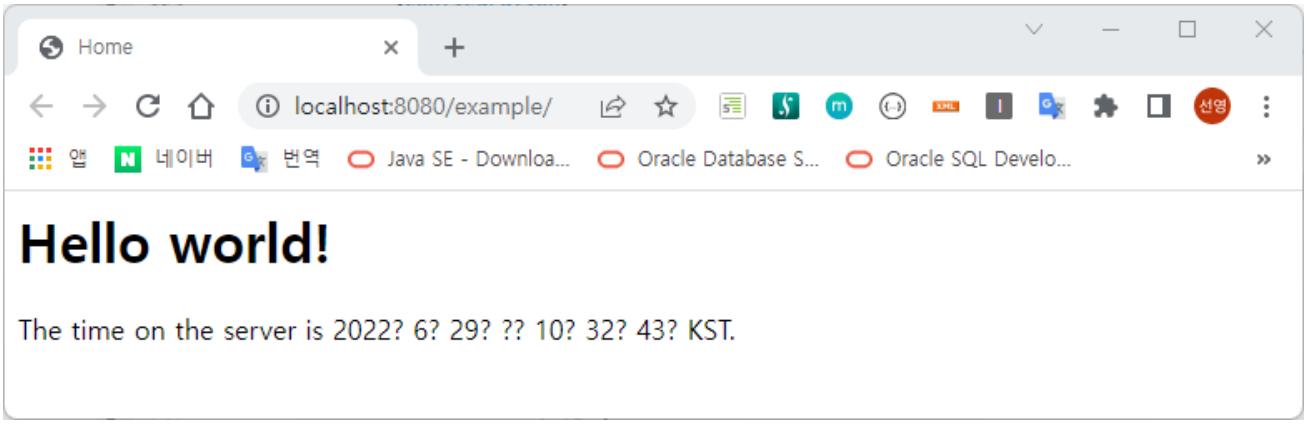
이 후 프로젝트를 선택한 상태에서 [Maven] - [Update Project]를 실행한다.

※ 메이븐에 대한 참고 사이트

- http://search.maven.org/
- http://www.mvnrepository.com/

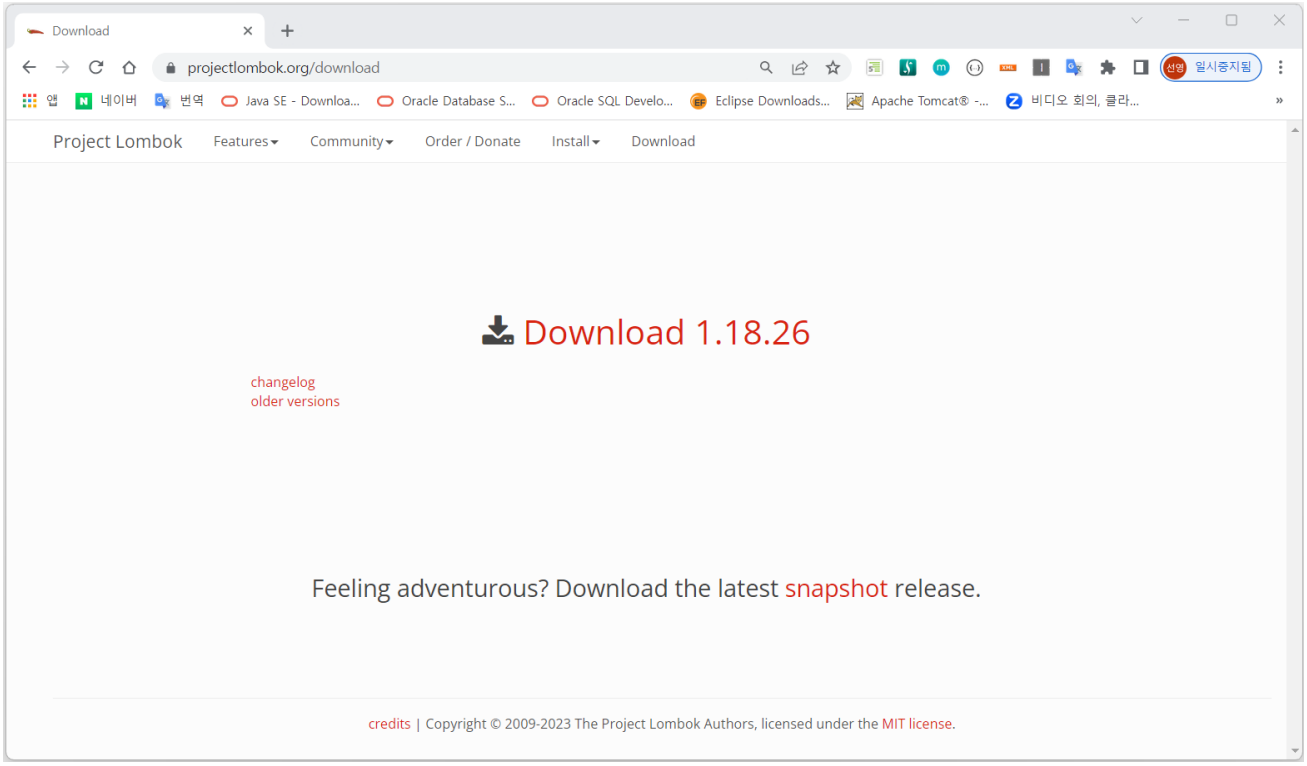
1.3 Tomcat을 이용한 프로젝트 실행 확인

작성된 프로젝트가 정상적으로 동작하는지 프로젝트를 실행해서 확인하는 과정은 프로젝트의 'Run > Run on Server'를 이용해서 처리한다.

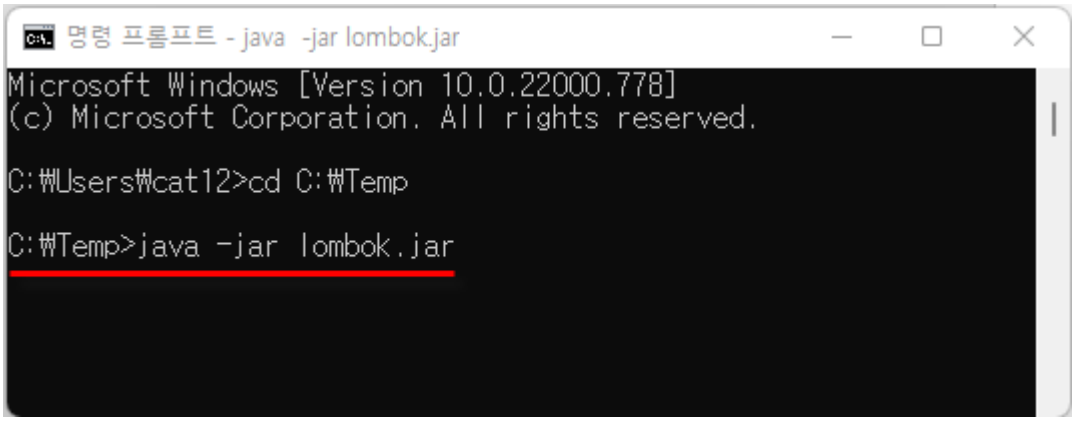


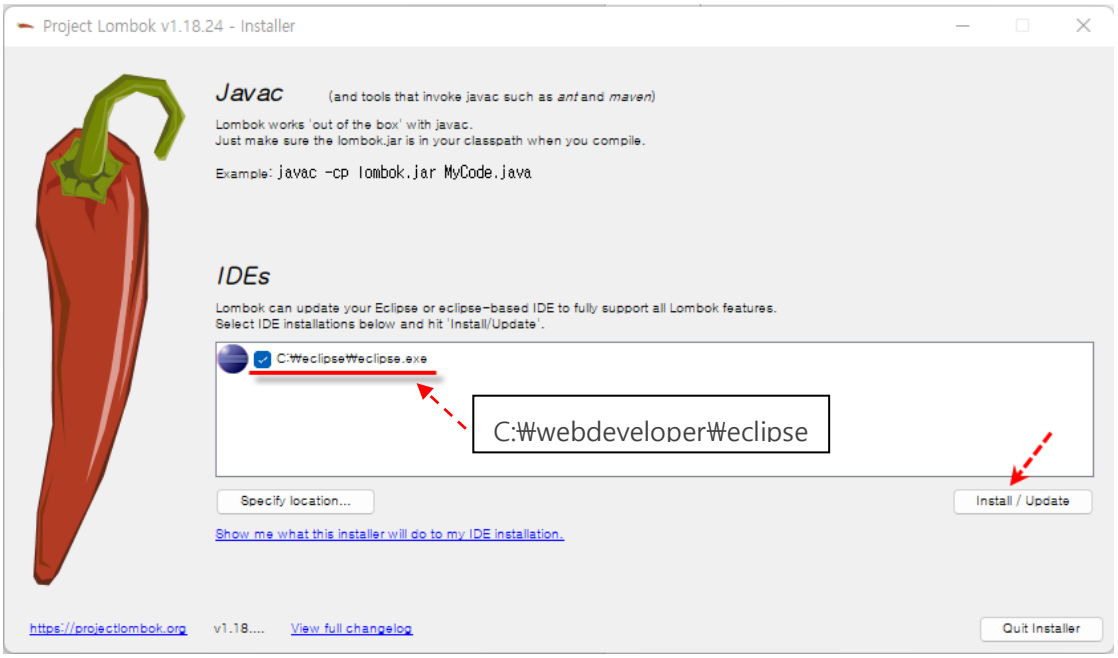
1.4 Lombok 라이브러리 설치

이클립스와 스프링 플러그인 만으로도 스프링 개발은 가능하지만 Lombok을 이용하면 Java 개발 시 자주 사용하는 getter / setter, toString(), 생성자 등을 자동으로 생성해주므로 약간의 코드만으로도 필요한 클래스를 설계할 때 유용하다. Lombok의 다운로드는 **https://projectlombok.org/**에서 [Download]에서 jar 파일 형태로 받을 수 있다.

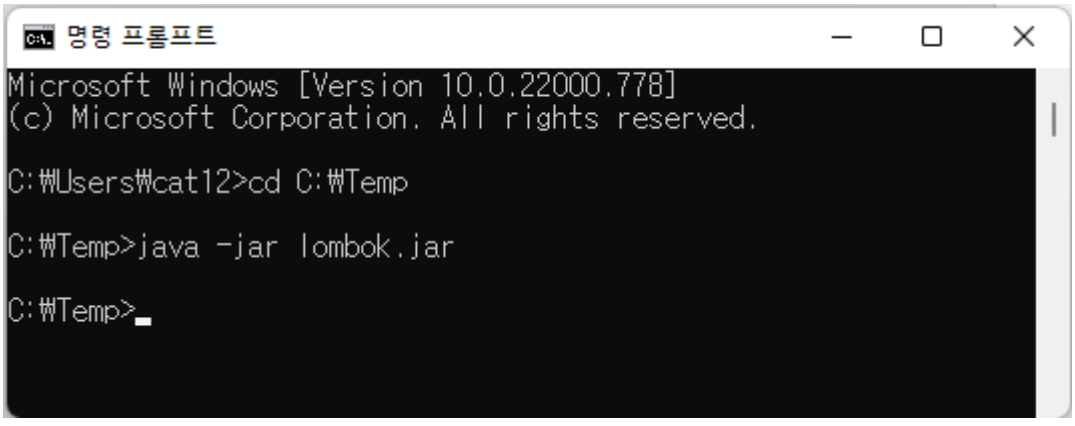


명령 프롬프트창에서 'java -jar lombok.jar'와 같은 명령어를 통해서 실행할 수 있다.

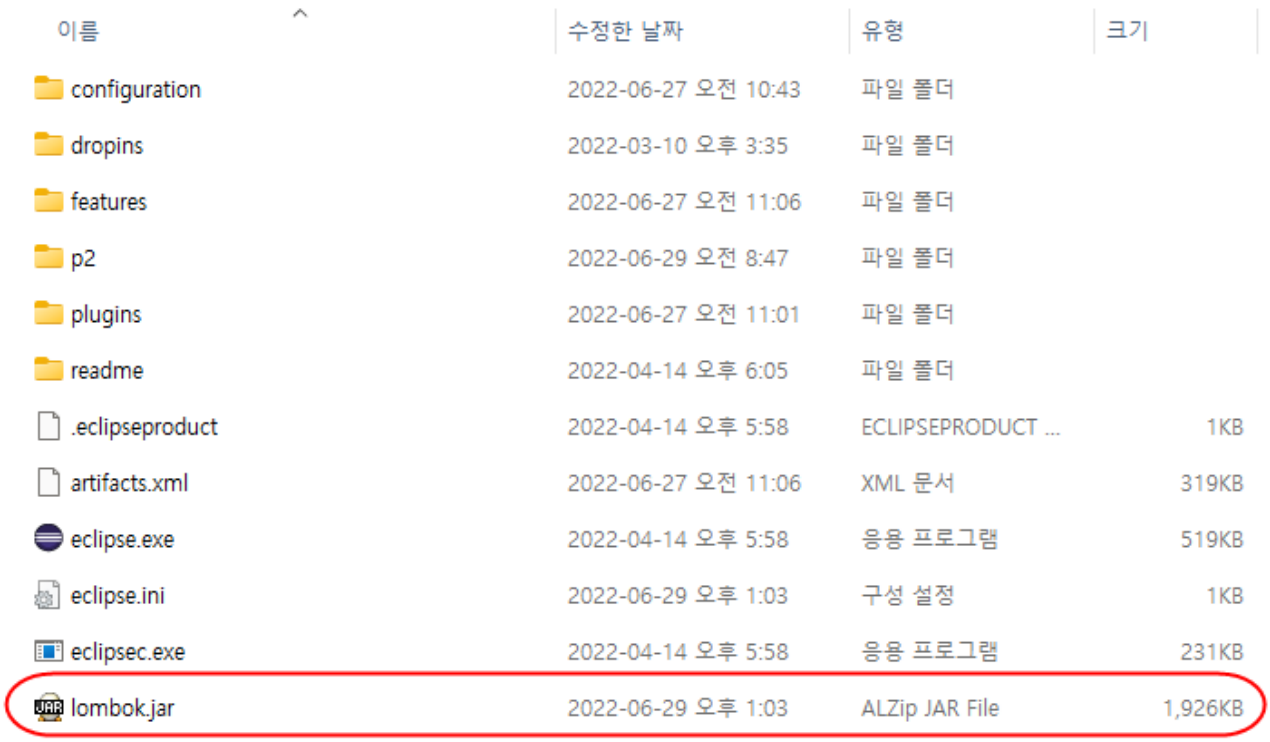



Project Lombok v1.18.24 - Installer
Javac
(and tools that invoke javac such as ant and maven)
Lombok works 'out of the box' with javac.
Just make sure the lombok.jar is in your classpath when you compile.
Example: javac -cp lombok.jar MyCode.java
IDEs
Lombok can update your Eclipse or eclipse-based IDE to fully support all Lombok features.
Select IDE installations below and hit 'Install/Update'.
C:₩eclipse₩eclipse.exe
C:₩webdeveloper₩eclipse
Specify location...
Install / Update
Show me what this installer will do to my IDE installation.
https://projectlombok.org
v1.18....
View full changelog
Quit Installer


Project Lombok v1.18.24 - Installer
Install successful
Lombok has been installed on the selected IDE installations.
Don't forget to:
add lombok.jar to your projects.
exit and start your IDE.
rebuild all projects!
If you start Eclipse with a custom -vm parameter, you'll need to add:
-vmargs -javaagent:lombok.jar
as parameter as well.
@Nullable annotations and thus generate a compiler error. . Thanks @JohnPaulTaylorII!
BUGFIX: Null analysis in eclipse was broken for incremental builds. .
BUGFIX VerifyError would show up in the latest eclipse release when using various refactor scripts. .
BUGFIX: The various @Log annotations can now be placed on inner enums and records. .
SECURITY: A widely reported security issue with log4j2 (CVE-2021-44228) has absolutely no effect on either lombok itself nor does usage of lombok on its own, or even the usage of lombok's @Log4j2, cause any issues whatsoever: You have to ship your own log4j2 dependency in your app - update that to 2.17 or otherwise mitigate this issue (see the CVE page). To avoid unneccessary warnings from dependency checkers, our dep on log4j2, which is used solely for testing, isn't shipped by us, and cannot be exploited in any way, has been updated to 2.17.1.
IMPROBABLE BREAKING CHANGE: Lombok now understands a few more annotations that imply "this field should not ever contain a null reference". Lombok will thus copy some of these new annotations e.g. to generated getters and the like. Pull Request #2904
https://projectlombok.org
v1.18.24
View full changelog
Quit Installer


명령 프롬프트
Microsoft Windows [Version 10.0.22000.778]
(c) Microsoft Corporation. All rights reserved.
C:₩Users₩cat12>cd C:₩Temp
C:₩Temp>java -jar lombok.jar
C:₩Temp>

C:₩webdeveloper₩eclipse 폴더에 아래와 같이 추가되어 있는지 확인한다.



이름	수정한 날짜	유형	크기
configuration	2022-06-27 오전 10:43	파일 폴더	
dropins	2022-03-10 오후 3:35	파일 폴더	
features	2022-06-27 오전 11:06	파일 폴더	
p2	2022-06-29 오전 8:47	파일 폴더	
plugins	2022-06-27 오전 11:01	파일 폴더	
readme	2022-04-14 오후 6:05	파일 폴더	
.eclipseproduct	2022-04-14 오후 5:58	ECLIPSEPRODUCT ...	1KB
artifacts.xml	2022-06-27 오전 11:06	XML 문서	319KB
eclipse.exe	2022-04-14 오후 5:58	응용 프로그램	519KB
eclipse.ini	2022-06-29 오후 1:03	구성 설정	1KB
eclipsec.exe	2022-04-14 오후 5:58	응용 프로그램	231KB
lombok.jar	2022-06-29 오후 1:03	ALZip JAR File	1,926KB

혹시 바탕화면의 바로가기가 존재한다면 삭제 후 다시 생성한다.

2. 1 스프링 프레임워크의 간략한 역사

프레임워크는 말그대로 뼈대나 근간을 이루는 코드들의 묶음이라고 할 수 있다. 개발자는 각 개개인의 능력 차이가 큰 직종이고, 따라서 개발자 구성에 따라 프로젝트의 결과 역시 큰 차이를 낳을 수밖에 없다. 프레임워크는 바로 이런 상황을 극복하기 위한 코드의 결과물이다. 프레임워크를 이용한다는 의미는 프로그램의 기본 흐름이나 구조를 정하고 모든 팀원이 이 구조에 자신의 코드를 추가하는 방식으로 개발하게 된다.

프레임워크 최대의 장점은 개발에 필요한 구조를 이미 코드로 만들어 놓았기 때문에 실력이 부족한 개발자라 하더라도 반쯤 완성한 상태에서 필요한 부분을 조립하는 형태의 개발이 가능하다는 점이다.

- 복잡함에 반기를 들어서 만들어진 프레임워크
- 프로젝트의 전체 구조를 설계할 때 유용한 프레임워크
- 다른 프레임워크들의 포용
- 개발 생산성과 개발 도구의 지원

2. 2 자바 기반의 프레임워크

자바 기반의 프레임워크는 대부분 오픈소스 형태로 제공된다. 따라서 별도의 라이선스나 비용을 지불하지 않고 누구나 사용할 수 있으며, 기존의 프레임워크를 이용하여 자신만의 프레임워크를 구축할 수도 있다.

· 대표적인 자바 기반의 프레임워크로는 다음과 같은 것들이 있다.

처리 영역	프레임워크	설명
Presentation	Struts	Struts 프레임워크는 UI Layer에 중점을 두고 개발된 MVC (Model View Controller)프레임워크이다.
	Spring(MVC)	Struts와 동일하게 MVC 아키텍처를 제공하는 UI Layer 프레임워크이다. 하지만 Struts처럼 독립된 프레임워크는 아니고 Spring 프레임워크에 포함되어 있다.
Business	Spring(IoC, AOP)	Spring은 컨테이너 성격을 가지는 프레임워크이다. Spring의 IoC와 AOP 모듈을 이용하여 Spring 컨테이너에서 동작하는 엔터프라이즈 비즈니스 컴포넌트를 개발할 수 있다.
Persistence	Hibernate or JPA	Hibernate는 완벽한 ORM(Object Relational Mapping) 프레임워크이다. ORM 프레임워크는 SQL 명령어를 프레임워크가 자체적으로 생성하여 DB 연동을 처리한다. JPA는 Hibernate를 비롯한 모든 ORM의 공통 인터페이스를 제공하는 자바 표준 API이다.
	Ibatis or Mybatis	Ibatis 프레임워크는 개발자가 작성한 SQL 명령어와 자바 객체(VO 혹은 DTO)를 매핑해주는 기능을 제공하며, 기존에 SQL 명령어를 재사용하여 개발하는 프로젝트에 유용하게 적용할 수 있다. Mybatis는 Ibatis에서 파생된 상위 프레임워크이다.

2. 3 스프링 프레임워크

스프링 프레임워크는 로드 존슨(Rod Johnson)이 2004년에 만든 오픈소스 프레임워크다.

다시 말해 스프링 프레임워크(spring Framework)는 엔터프라이즈 자바 소프트웨어 개발을 단순화하기 위하여 만들어진 오픈 소스 응용프로그램 프레임워크이다. (간단히 스프링(spring)이라고도 불린다.)

이 프레임워크는 개발자들에게 컴포넌트 모델을 제공하고, 단순하면서도 일관성 있는 API 집합을 제공하여 개발자들이 복잡한 응용프로그램을 설계하는 동안 복잡하고 어려운 기반 코드에 연연하지 않도록 도움을 준다.

동적인 웹 사이트를 개발하기 위한 다양한 서비스를 제공하고 있고, 대한민국 공공기관의 전자정부 표준 프레임워크 기반 기술로서 사용되고 있다.

2.3.1 스프링의 주요 특징

스프링의 주요 특징이라고 하면 주로 다음과 같은 점을 들 수 있다.

- POJO 기반의 구성
- 의존성 주입(DI)을 통한 객체 간의 관계 구성
- AOP(Aspect-Oriented-Programming) 지원
- 편리한 MVC 구조
- WAS의 종속적이지 않은 개발 환경

POJO 기반의 구성

스프링은 내부에는 객체 간의 관계를 구성할 수 있는 특징을 가지고 있으며 다른 프레임워크들과 달리 이 관계를 구성할 때 별도의 API 등을 사용하지 않는 POJO(Plain Old Java Object)의 구성만으로 가능하도록 제작되어 있다. 다시 말해 일반적인 Java 코드를 이용해서 객체를 구성하는 방식을 그대로 스프링에서 사용할 수 있다는 것이다. 이것이 중요한 이유는 코드를 개발할 때 개발자가 특정한 라이브러리나 컨테이너의 기술에 종속적이지 않다는 것을 의미하기 때문이다. 개발자는 가장 일반적인 형태로 코드를 작성하고 실행할 수 있기 때문에 생산성에서도 유리하고, 코드에 대한 테스트 작업 역시 좀 더 유연하게 할 수 있다는 장점이 생긴다.

POJO(Plain Old Java Object)란?

POJO란 말 그대로 평범한 옛날 자바 객체를 의미한다. 다시 말해 객체 지향적인 원리에 충실하면서, 환경과 기술에 종속되지 않고 필요에 따라 재활용될 수 있는 방식으로 설계된 Object를 의미한다.

POJO를 좀 더 쉽게 이해하기 위해서 반대로 POJO가 아닌 클래스가 무엇인지 이해하면 된다. 대표적인 Not POJO 클래스가 Servlet 클래스이다.

Servlet 클래스는 우리 마음대로 만들 수 없으며, 반드시 Servlet에서 요구하는 규칙에 맞게 클래스를 만들어야 실행할 수 있다. 다음은 Servlet 클래스 작성 규칙이다.

- Javax.servlet, Javax.servlet.http 패키지를 import해야 한다.
- public 클래스로 선언되어야 한다.
- Servlet, GenericServlet, HttpServlet 중 하나를 상속해야 한다.
- 기본 생성자(Default Constructor)가 있어야 한다.
- 생명주기에 해당하는 메소드를 재정의(Overriding)한다.

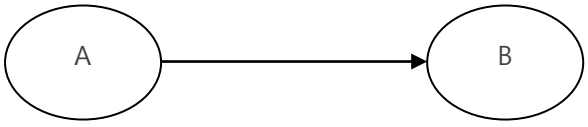
의존성 주입(DI)과 스프링

의존성(Dependency)이라는 것은 하나의 객체가 다른 객체 없이 제대로 된 역할을 할 수 없다는 것을 의미한다. 의존성은 이처럼 하나의 객체가 다른 객체의 상태에 따라 영향을 받는 것을 의미한다. **흔히 A 객체가 B객체 없이 불가능한 상황을 'A가 B에 의존적이다'라고 표현한다.**

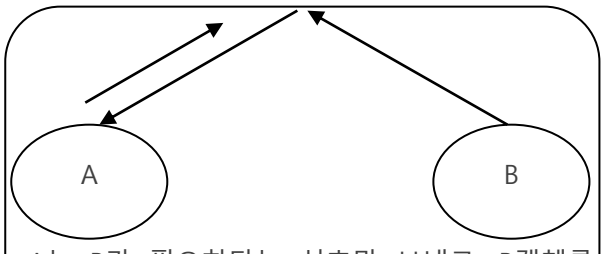
```
public class BoardController{
        private BoardService boardService;

        public void setBoardService(BoardService boardService){
                this.boardService = boardService;
        }
        public String boardList(){
                ArrayList<BoardVO> board = boardService.boardList();
                ...
        }
}
```

'주입(Injection)'은 말 그대로 외부에서 '밀어 넣는 것'을 의미한다.

의존성과 주입을 결합해서 생각해 보면 '어떤 객체가 필요한 객체를 외부에서 밀어 넣는다'는 의미가 된다. 이를 코드에 대입해서 살펴보면 '주입을 받는 입장에서는 어떤 객체인지 신경 쓸 필요가 없다.' '어떤 객체에 의존하든 자신의 역할을 변하지 않는다'와 같은 의미로 볼 수 있다. 이러한 변화를 그림으로 표현하면 다음과 같은 형태가 될 수 있다.



A객체에서 B객체를 직접 생성하는 방식



A는 B가 필요하다는 신호만 보내고, B객체를 주입하는 것은 외부에서 이루어지는 방식

'의존성 주입' 방식을 사용하려면 오른쪽 그림의 도형처럼 추가적인 하나의 존재가 필요하게 된다.
이 존재는 의존성이 필요한 객체에 필요한 객체를 찾아서 '주입'하는 역할을 하게 된다.

스프링은 이러한 구조를 만드는 데 적합한 구조로 설계되어 있다. 스프링에서는 'ApplicationContext'라는 존재가 필요한 객체들을 생성하고, 필요한 객체들을 주입하는 역할을 해 주는 구조이다. 따라서 스프링을 이용하면 개발자들은 기존의 프로그래밍과 달리 객체와 객체를 분리해서 생성하고 이러한 객체들을 엮는 작업을 하는 형태의 개발을 하게 된다. **스프링에서는 ApplicationContext기 관리하는 객체들을 '빈(Bean)'이라는 용어로 부르고, 빈과 빈 사이의 의존관계를 처리하는 방식으로 XML 설정, 어노테이션 설정, Java 설정 방식을 이용할 수 있다.**

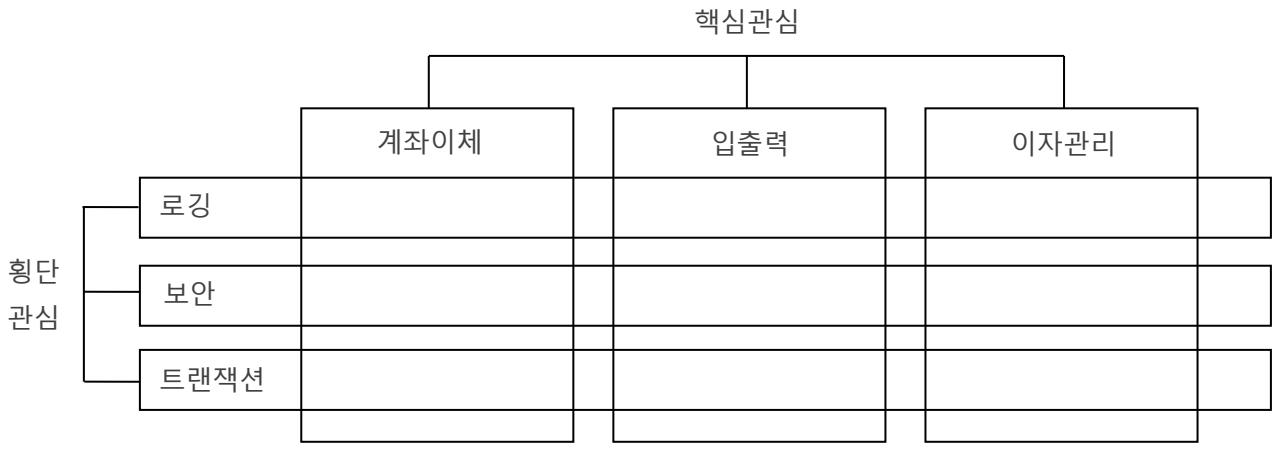
AOP(Aspect-Oriented-Programming) 지원

스프링은 프레임워크를 이용한 개발에도 이러한 반복적인 코드를 줄이고, 핵심 비즈니스 로직에만 집중할 수 있는 방법을 제공한다. 대부분의 시스템이 공통으로 가지고 있는 보안이나 로그, 트랜잭션과 같이 비즈니스 로직은 아니지만 반드시 처리가 필요한 부분을 스프링에서는 '횡단 관심사(cross-concern)'라고 한다. 스프링은 이러한 횡단 관심사를 분리해서 제작하는 것이 가능하다.

AOP(Aspect-Oriented-Programming)는 이러한 횡단 관심사를 모듈로 분리하는 프로그래밍의 패러다임이다. 스프링은 AOP를 AspectJ의 문법을 통해서 작성할 수 있는데, 개발자는 1) 핵심 비즈니스 로직에만 집중해서 코드를 개발할 수 있게 되었고, 2) 각 프로젝트마다 다른 관심사를 적용할 때 코드의 최소화시킬 수 있었으며, 3) 원하는 관심사의 유지보수가 수월한 코드를 구성할 수 있다.

→ AOP란, 기능을 핵심 비지니스 로직과 공통 모듈로 구분하고, 핵심 로직에 영향을 미치지 않고 사이 사이에 공통 모듈을 효과적으로 잘 끼워 넣도록 하는 개발 방법이다. 공통 모듈은 보안 인증, 로깅 같은 요소들이 해당된다

관계지향 프로그래밍은 비즈니스 메서드를 개발할 때 핵심 비즈니스 로직과 각 비즈니스 메소드마다 반복해서 등장하는 공통 로직을 분리함으로써 응집도가 높게 개발할 수 있도록 지원한다.



공통으로 사용하는 기능들을 외부의 독립된 클래스로 분리하고, 해당 기능을 프로그램 코드에 직접 명시하지 않고 선언적으로 처리하여 적용하는 것이 관점지향 프로그래밍의 기본 개념이다. 이렇게 되면 공통 기능을 분리하여 관리할 수 있으므로 응집도가 높은 비즈니스 컴포넌트를 만들 수 있을 뿐만 아니라 유지보수를 혁신적으로 향상시킬 수 있다.

트랜잭션의 지원

데이터베이스를 이용할 때 반드시 신경 써야 하는 부분은 하나의 업무가 여러 작업으로 이루어지는 경우의 트랜잭션 처리이다. 이 트랜잭션 처리는 상황에 따라서 복잡하게 구성될 수도 있고, 아닐 수도 있는데 그때마다 코드를 이용해서 처리하는 작업은 개발자에게는 상당히 피곤한 일이다. 스프링은 이런 트랜잭션의 관리를 어노테이션이나 XML로 설정할 수 있기 때문에 개발자가 매번 상황에 맞는 코드를 작성할 필요가 없도록 설계되었다.