학습 내용

- 스프링과 메이븐
- 메이븐의 특징

학습 목표

- 스프링 개발 시 필수로 사용되는 메이븐 빌드툴에 대해 알아봅니다.
- 메이븐의 개요와 특징에 대해 알아봅니다.

Maven (메이븐)

Spring 프로젝트의 기본적인 골격을 세울 때 Maven이라는 빌드 툴을 사용하여 골격을 만듭니다.

- 자바 개발의 사실상 표준 빌드 툴
  - 이전에는 ANT(최초 빌드 툴)를 많이 사용
  - Maven과 상호보완적인 빌드 툴이 아님
- XML 설정파일을 사용하여 빌드함
  - 최근에는 groovy라는 언어로 설정하는 gradle이라는 빌드 툴이 등장
  - maven과 상호보완적인 빌드 툴임

메이븐이 외부 라이브러리를 관리해줍니다.

Maven의 특징

Convention over Configuration(CoC)

우리가 개발할 때는 애플리케이션 개발 초기 설정 작업이 필요한데, 그 작업들이 대부분 비슷비슷하고 반복적으로 해야하는 경우가 있습니다. 그래서 이 작업을 할 필요 없도록 제공하는 것
이 **Maven의 특징**입니다. 즉, 미리 모든 것들이 설정되어 있기 때문에 개발자가 직접 설정해야하는 작업을 최소화할 수 있습니다.
CoC의 또다른 예시로는 Node.js의 Express 등을 볼 수 있습니다.

Java 기반 Web Application의 Best Practice를 따름

그렇기 때문에 Spring 기반의 대부분 시스템 개발 디렉토리 구조가 비슷합니다.
또한 빌드 단계도 미리 정의해뒀기 때문에(Compile - test - Package - install - diplay) 그대로 따라 사용하기만 하면 됩니다.

⭐ 의존성 관리를 자동으로 수행

사실상 ANT에서 Maven으로 넘어온 결정적인 Maven의 특징입니다.
개발 시 수백 개의 라이브러리를 관리해야합니다. 버전도 계속 업데이트 되고, 라이브러리끼리 의존성으로 인한 충돌 문제가 발생하면 개발자에게 많은 어려움을 줍니다. 하지만 Maven은 **중앙 저장소(Central Repository)**를 제공하여 자바 라이브러리에 대한 생태계를 조성해줍니다. 중앙 저장소를 통해 라이브러리를 한 곳에 모아 쉽게 의존성을 관리할 수 있게 하는 방법을 제공합니다.
또다른 예시로는 Node.js의 npm입니다.

POM.XML이라는 메이븐의 단 하나의 메인 설정파일 제공

이 파일은 프로젝트 루트에 위치해있으며, 하나이기 때문에 굉장히 심플해졌습니다.
이 프로젝트가 메이븐 기반인지 확인하고 싶으시면 루트에 PON.XML이 있는지만 보면 됩니다.
프로젝트를 IDE에서 불러오기나, 내보내기가 쉬워집니다.

메이븐 프로젝트 설정 시 필수사항

artifact ID 설정

- 일반적으로 프로젝트 명과 동일하게 설정합니다.

그룹 아이디 설정

- 주로 프로젝트 생성 조직이나 기관의 도메인 명 역순으로 표기(예 : kr.co.company)
- Top-level package명으로 사용됨

메이븐 중앙 저장소에는 수많은 라이브러리가 있습니다. 이 라이브러리들을 구분해야하는데 이를 위해 그룹 아이디+artifact ID 조합으로 구분합니다.

버전 설정

- 개발버전을 의미하는 **SNAPSHOT** 버전 사용
- cf) 배포버전 : **release** 버전

중앙저장소에는 같은 라이브러리일지라도 버전별로 저장되어 있습니다. 그래서 버전을 정확하게 기재해야 합니다. 일반적으로 세자리 수로 설정합니다.(예 : 1.2.3)

메이븐 설치

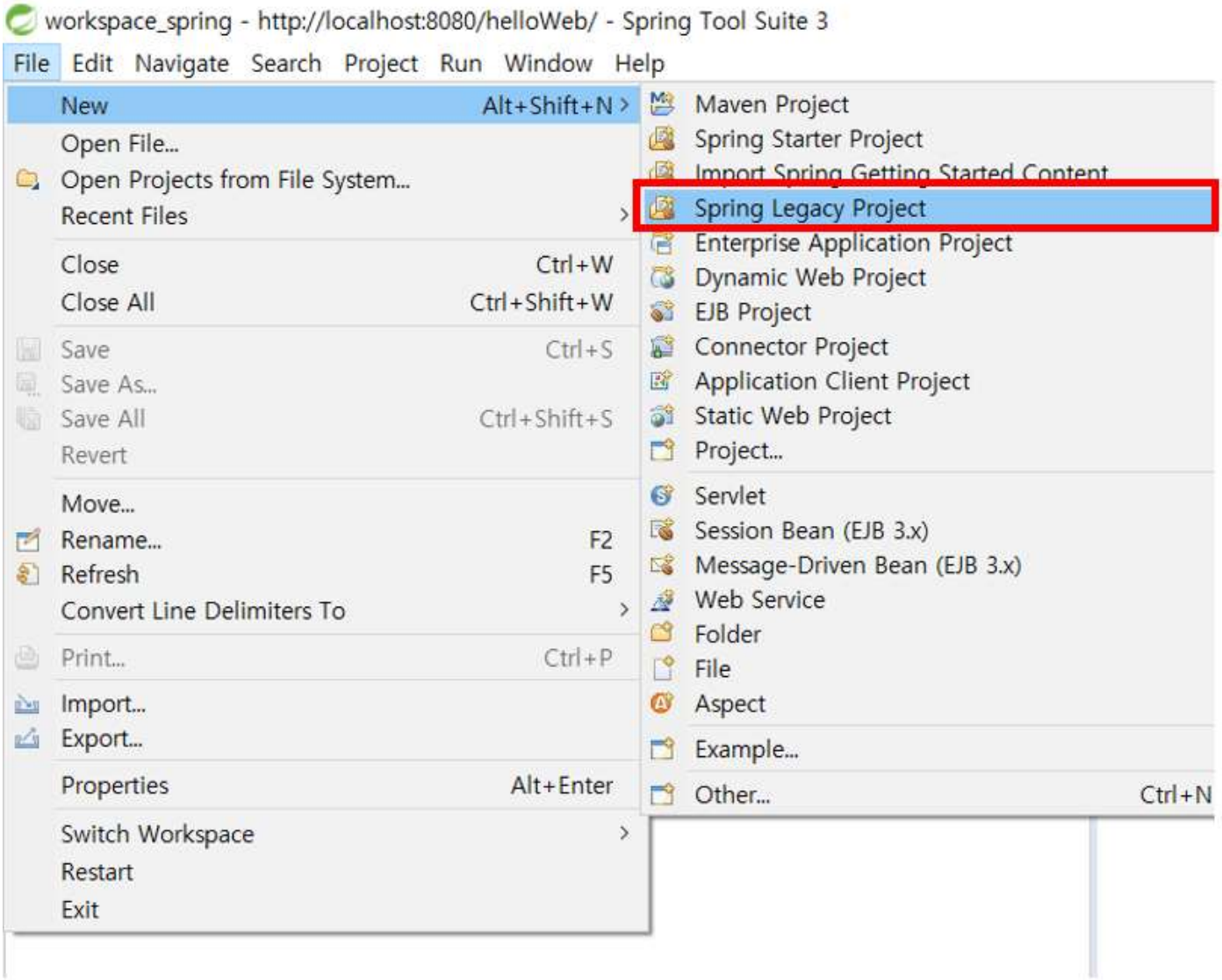
Maven 3 버전 사용

- 로컬에 설치 필요
- IDE에 포함된 경우에는 별도 설치 필요 없음
- STS 3.8.X IntelliJ IDEA 2017 버전 이상은 모두 메이븐 3.X 버전을 포함하고 있음

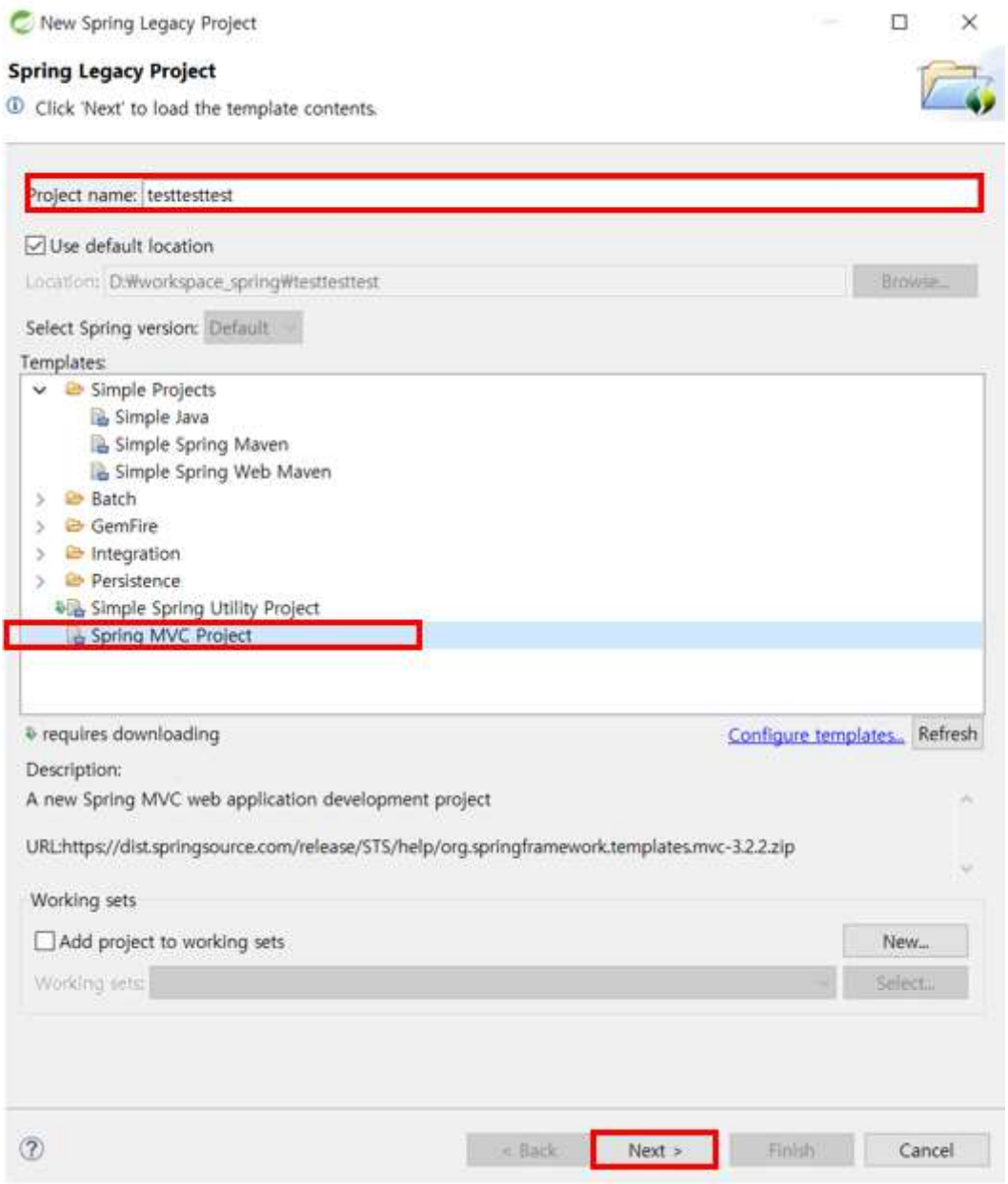
Spring Maven 프로젝트 생성 실습

STS에서 Spring 프로젝트 생성

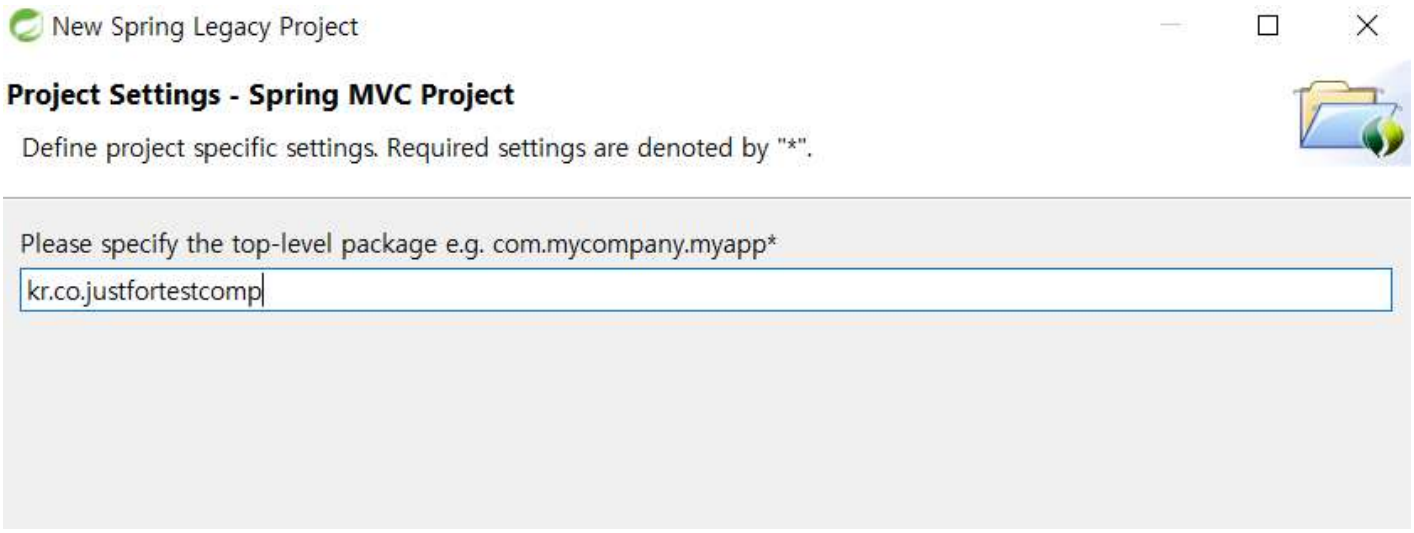
미리 말씀드리자면 환경 세팅하는 포스팅은 따로 재업로드 할 예정입니다.



File -> New -> Spring Legacy Project 클릭

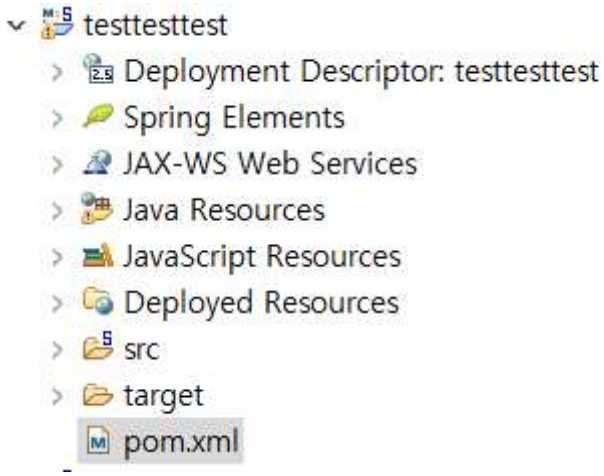


Project Name 설정 -> Spring MVC Project 선택 -> Next 클릭



그룹 아이디가 될 닉네임을 정해주세요.
위에서 언급했듯이 일반적으로 회사 사이트 도메인을 거꾸로 해서 설정한다고 합니다.(강사님이 그러셨음)

Spring MVC Project 생성 후



이렇게 디렉토리 구조가 생성됩니다.(휴;;;)
여기서 **pom.xml**을 열어봅시다.

pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/maven-v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>kr.co</groupId>
    <artifactId>justfortestcomp</artifactId>
    <name>testtesttest</name>
    <packaging>war</packaging>
    <version>1.0.0-BUILD-SNAPSHOT</version>
    <properties>
        <java-version>1.6</java-version>
        <org.springframework-version>3.1.1.RELEASE</org.springframework-version>
        <org.aspectj-version>1.6.10</org.aspectj-version>
        <org.slf4j-version>1.6.6</org.slf4j-version>
    </properties>
    <dependencies>
        <!-- Spring -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>${org.springframework-version}</version>
            <exclusions>
                <!-- Exclude Commons Logging in favor of SLF4j -->
                <exclusion>
                    <groupId>commons-logging</groupId>
                    <artifactId>commons-logging</artifactId>
                 </exclusion>
            </exclusions>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>${org.springframework-version}</version>
        </dependency>

        <!-- AspectJ -->
        <dependency>
```