Maven 저장소와 의존성 관리

Maven은 빌드 툴이지만 라이브러리 의존성 관리 툴로 더 많이 사용되고 있는것 같습니다. 스프링 프레임웍이나 전자정부 표준 프레임웍을 사용하게 되면 프로젝트의 기본 빌드 설정을 그대로 사용해서 별로 문제가 되는게 없으므로 손댈 것이 별로 없었던것 같습니다.

대부분 필요한 라이브러리를 추가하거나 하기 위해서 저장소(repository)와 의존성(dependency) 부분만을 많이 사용하게 됩니다.

이 글에서는 저장소(repository)와 의존성(dependency)을 사용하는 부분에 대해서 알아보겠습니다. Maven은 단독으로 설치해서 사용할 수도 있습니다. http://maven.apache.org 에서 다운로드 받아서 설치하면 됩니다. 여기서는 직접 설치하지 않고, Eclipse에 기본으로 포함되어 있는 Maven으로 알아 볼 것입니다.

예전의 자바 웹 프로젝트에서는 필요한 라이브러리 파일이 /WEB-INF/lib 폴더와 WAS에 있게 됩니다. WAS에서 기본 제공하거나 여러 Context 에서 공통적으로 사용되는 것은 WAS가 가지고 있고, 현재 Context 에서만 사용될 것은 /WEB-INF/lib 폴더에 두게 됩니다.

여기서 발생되는 문제로는 프로젝트가 오래될 경우 WAS에 의존적인 라이브러리들이나 구 버전의 라이브러리들을 잊어먹거나 구할 수가 없게되는 경우가 자주 발생합니다.

메이븐을 사용해서 라이브러리를 관리하면 이러한 경우에 좀더 잘 대처할 수 있습니다. 메이븐은 라이브러리를 프로젝트내에 포함하지 않습니다. Maven을 사용하는 프로젝트에는 설정 파일인 pom.xml 파일이 있습니다. POM은 Project Object Model의 약자 입니다. 이 설정에서 실제 라이브러리 파일이 있는 저장소 서버의 위치를 지정하고, 또한 이 프로젝트에서 사용할 라이브러리가 무엇인지를 지정하기만 하면 됩니다.

메이븐은 글로벌 저장소에서 필요한 라이브러리를 로컬 컴퓨터로 자동으로 다운로드하고, 로컬 저장소의 라이브러리들로 클래스 패스를 지정하여 프로젝트에서 사용하도록 합니다.

프로젝트를 배포하기 위해서 Maven build를 하면 의존성 설정에 따라 필요한 라이브러리를 /WEB-INF/lib 폴더로 복사해서 배포가능한 폴더 구조로 파일들을 모아주고, war 파일도 작성해 줍니다.

메이븐을 사용하여 프로젝트의 라이브러리 의존성을 관리하면 실제 .jar 파일들을 관리할 필요없이 pom.xml 파일만 잘 관리하며 되는 것입니다.

Eclipse에 설치된 Maven의 버전을 확인하는 방법입니다. 메뉴에서 Window -> Preferences를 선택해서 Preferences 창을 엽니다. 좌측 설정항목 트리에서 Maven -> Installations를 선택합니다. 현재 Maven버전을 확인할 수 있습니다. 필요하다면 새로운 버전을 다운로드 받아 설치할 다음에 여기에서 Add를 사용해서 새 버전의 Maven을 추가할 수도 있습니다.

---

pom.xml 파일의 구조를 알아 보겠습니다.

* <groupId> - 프로젝트를 구분하는 값으로 사용됩니다. 도메인을 주로사용합니다.
* <artifactId> - 역시 프로젝트를 구분하는 값으로 사용되며, 프로젝트 명을 사용합니다.
* <properties> - POM 내에서 공통적으로 사용되는 값들을 설정합니다. pom.xml 파일의 다른 곳에서 ${org.springframework-version} 처럼 사용합니다.
* <repositories> - 라이브러리를 받아올 저장소를 지정합니다.

* <dependencies> - 프로젝트에서 사용되는 라이브러리들을 지정합니다.
* <build> - 프로젝트의 빌드방법을 지정합니다.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/.
3      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.o
4      <modelVersion>4.0.0</modelVersion>
5      <groupId>com.poe</groupId>
6      <artifactId>test</artifactId>
7      <name>test01</name>
8      <packaging>war</packaging>
9      <version>1.0.0-BUILD-SNAPSHOT</version>
10     <properties>
11         <java-version>1.8</java-version>
12         <org.springframework-version>4.3.3.RELEASE</org.springframework-version>
13         <org.aspectj-version>1.6.10</org.aspectj-version>
14         <org.slf4j-version>1.6.6</org.slf4j-version>
15     </properties>
16     <dependencies>
17         <!-- Spring -->
18         <dependency>
19             <groupId>org.springframework</groupId>
20             <artifactId>spring-context</artifactId>
21             <version>${org.springframework-version}</version>
22             <exclusions>
23                 <!-- Exclude Commons Logging in favor of SLF4j -->
24                 <exclusion>
25                     <groupId>commons-logging</groupId>
26                     <artifactId>commons-logging</artifactId>
27                  </exclusion>
28             </exclusions>
29         </dependency>
30         <dependency>
```

위에서 우리가 직접 지정하는 pom.xml 파일의 구조에 대해서 알아 보았습니다.

위의 <repositories>안의 내용을 보면 http://maven.jahia.org/maven2 가 있는데, 이것은 jdbc 드라이버를 사용하기 위해서 직접 추가한 저장소 입니다. 그러면 Maven이 가지고 있는 기본 글로벌 저장소는 어디에 정의되어 있을까요? 기본 글로벌 저장소는 https://repo.maven.apache.org/maven2 입니다.

Maven은 자신이 가진 기본설정과 사용자가 지정한 설정을 가진 pom.xml 파일의 내용을 합한 Effective POM을 가지고 동작을 합니다. **Eclise에서는 Effective POM을 간단히 확인할 수 있습니다. pom.xml 파일을 열어서 하단의 Effective POM 탭을 클릭**하면 실제 사용되는 전체 내용을 확인할 수 있습니다.

메이븐의 공통 설정은 메이븐을 직접 설치했다면 MAVEN_HOME/conf 폴더 아래에 settings.xml 파일에 있을것 입니다. Eclipse 를 사용한다면 메뉴에서 Window -> Preferences 를 선택하여 Preferences창을 열어서 Maven -> User Settings 를 보면 있습니다.

개인설정을 사용하고자 한다면 USER_HOME/.m2 폴더 아래에 settings.xml 파일을 생성합니다. 위 예에서는 로그인 아이디가 Administrator 일 경우 C:₩Users₩Administrator₩.m2₩settings.xml 파일 입니다.

로컬 저장소의 위치는 USER_HOME/.m2/repository 입니다. 이 아래에 글로벌 저장소에서 가져온 라이브러리가 groupId 별로 저장되어 있습니다.

다음은 의존성 설정 부분입니다. 이 부분이 실제 프로젝트에서 사용되는 라이브러리를 지정하는 부분 입니다. 메이븐은 이 라이브러리를 저장소에서 찾아서 사용하게 되는 것입니다. 다음 예를 보겠습니다.

```
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
     <version>4.7</version>
     <scope>test</scope>
</dependency>
```

다른 부분은 그대로 사용하면 되는데 <scope> 부분은 사용자가 필요한 형태도 지정하여 사용하면 되겠습니다. junit는 테스트시에만 필요하고, 배포할 때는 빠져야 하는 라이브러리 입니다. 이럴경우 <scope>test</scope>로 지정을 하면 됩니다.

메이븐에서 사용할 수 있는 scope 는 다음과 같습니다.

* compile - 컴파일 및 배포시 같이 제공되어 집니다. scope를 설정하지 않았을 때의 **기본값**입니다.
* provided - WAS에서 제공되어 지므로 컴파일 시에는 필요하지만, 배포시에는 빠지는 라이브러리 입니다.
* runtime - 컴파일 때는 사용되지 않고, 실행시에만 사용되어지는 라이브러리 입니다.
* test - 테스트 할때만 사용하는 라이브러리 입니다.
* system - 저장소에서 관리하지 않고 직접 관리하는 jar 파일을 지정합니다.
* import - 다른 pom 설정파일에서 정의되어 있는 의존 관계를 이 프로젝트로 가져옵니다.

메이븐 기본 글로벌 저장소에 필요한 라이브러리가 없을 경우에는 라이브러리가 존재하는 저장소를 pom.xml 파일의 <repositories> 에 추가하면 됩니다. 메이븐 저장소로 라이브러리가 제공되지 않는 경우에는 .jar 파일을 다운 받아 의존성을 system 스코프로 관리하던가 아니면, 직접 /WEB-INF/lib 폴더에 넣고 마우스 오른쪽 키를 눌러 **add build path 를 선택해서 이클립스 빌드패스에 등록하여 사용하면 됩니다.**