웹 서버, 웹 UI 로컬 개발 환경
–

학습 목표

메이븐 빌드 도구 이해

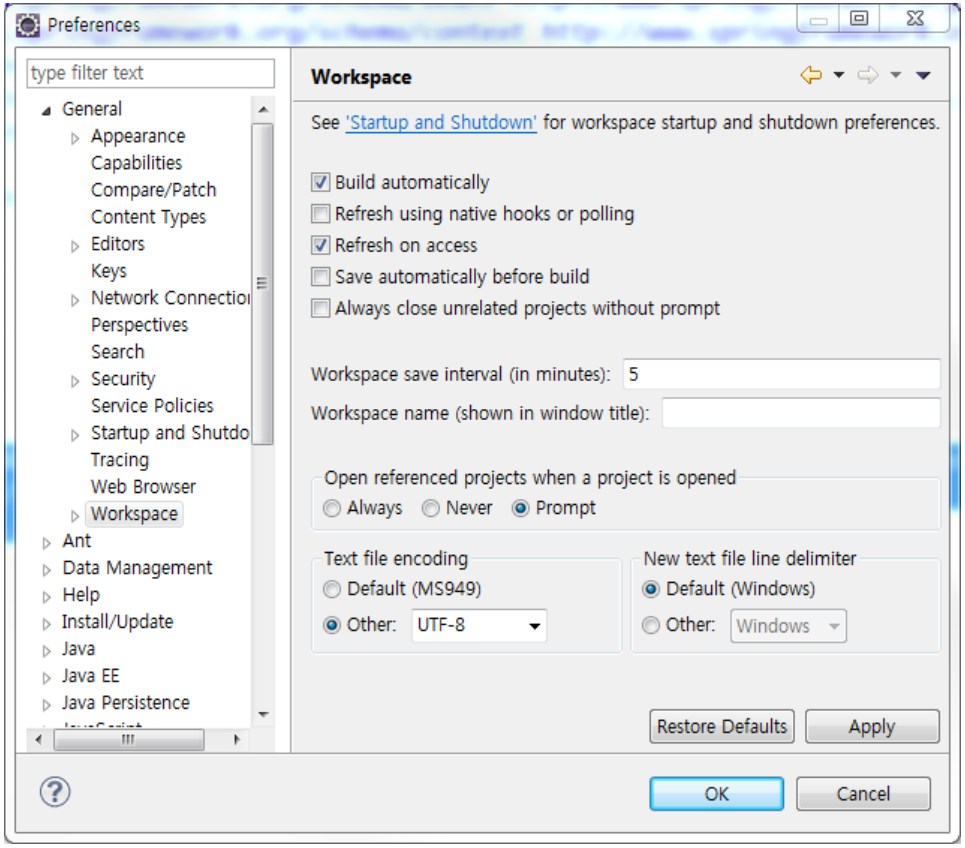
Eclipse + WTP를 활용한 웹 개발 환경 구축

Eclipse 설정

–

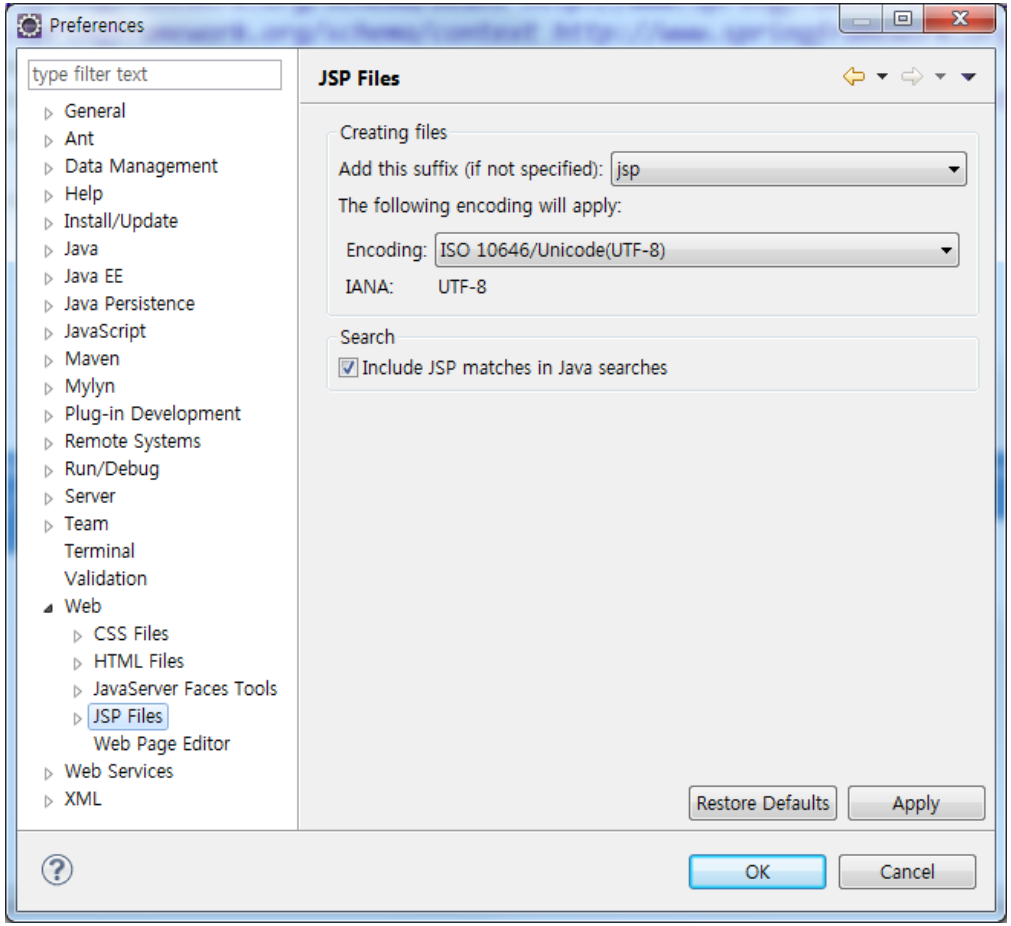
Workspace 인코딩 설정

Eclipse > 환경설정… (단축키 Cmd + ,) > General > Workspace 에서 Text file encoding을 UTF-8로 변경



JSP 파일 인코딩 설정

Eclipse > 환경설정… (단축키 Cmd + ,) > Web > JSP Files 에서 Encoding을 Unicode(UTF-8)로 설정



개발 환경

–

JDK 6.0 이상

Eclipse Java EE Developers 4.3 버전

(Maven, Git 플러그인 포함되어 있음)

Apache Tomcat 7.0.x

웹 프로젝트 생성

–

자바 웹 프로젝트 생성에 필요한 디렉토리

배포할 자바 소스 코드(production code) : 보통 src 디렉토리에서 관리

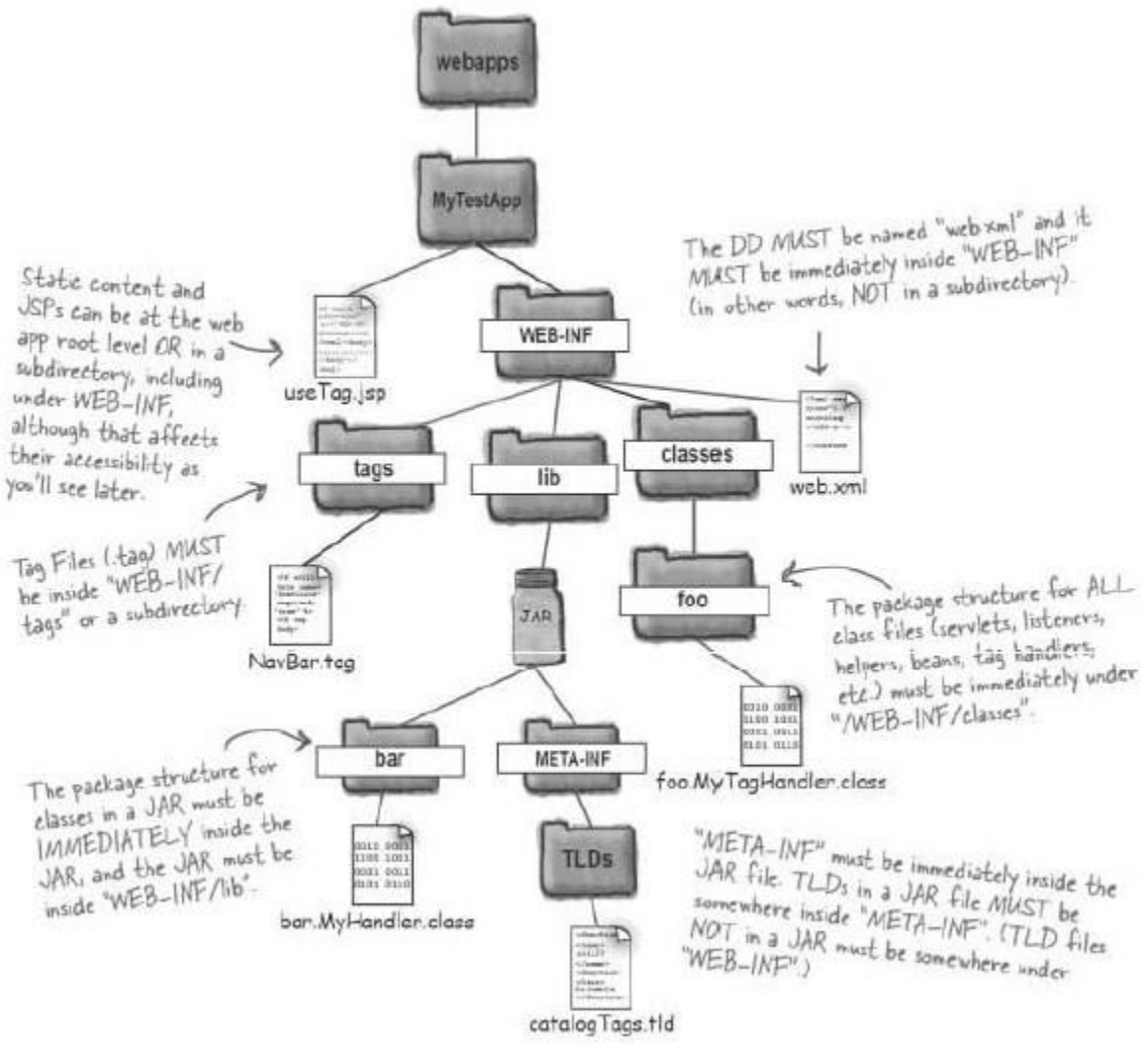
배포할 설정 파일 : 보통 resources 디렉토리에서 관리

테스트 자바 소스 코드(test code) : 보통 test 디렉토리에서 관리

웹 자원(html, css, javascript) : 보통 webapps 디렉토리에서 관리

기타 자바 웹 표준에 따르는 디렉토리

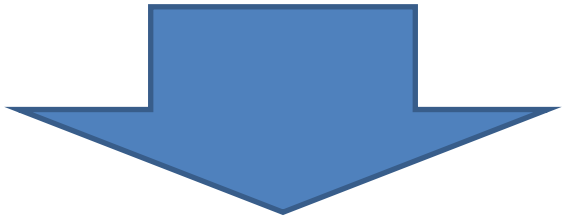
자바 표준 웹 프로젝트의 디렉토리 구조



이슈

프로젝트를 생성할 때마다 비슷한 구조의 디렉토리를 매번 생성해야 한다.

프로젝트마다 디렉토리 구조가 달라 새로운 프로젝트를 이해하기 어렵다.

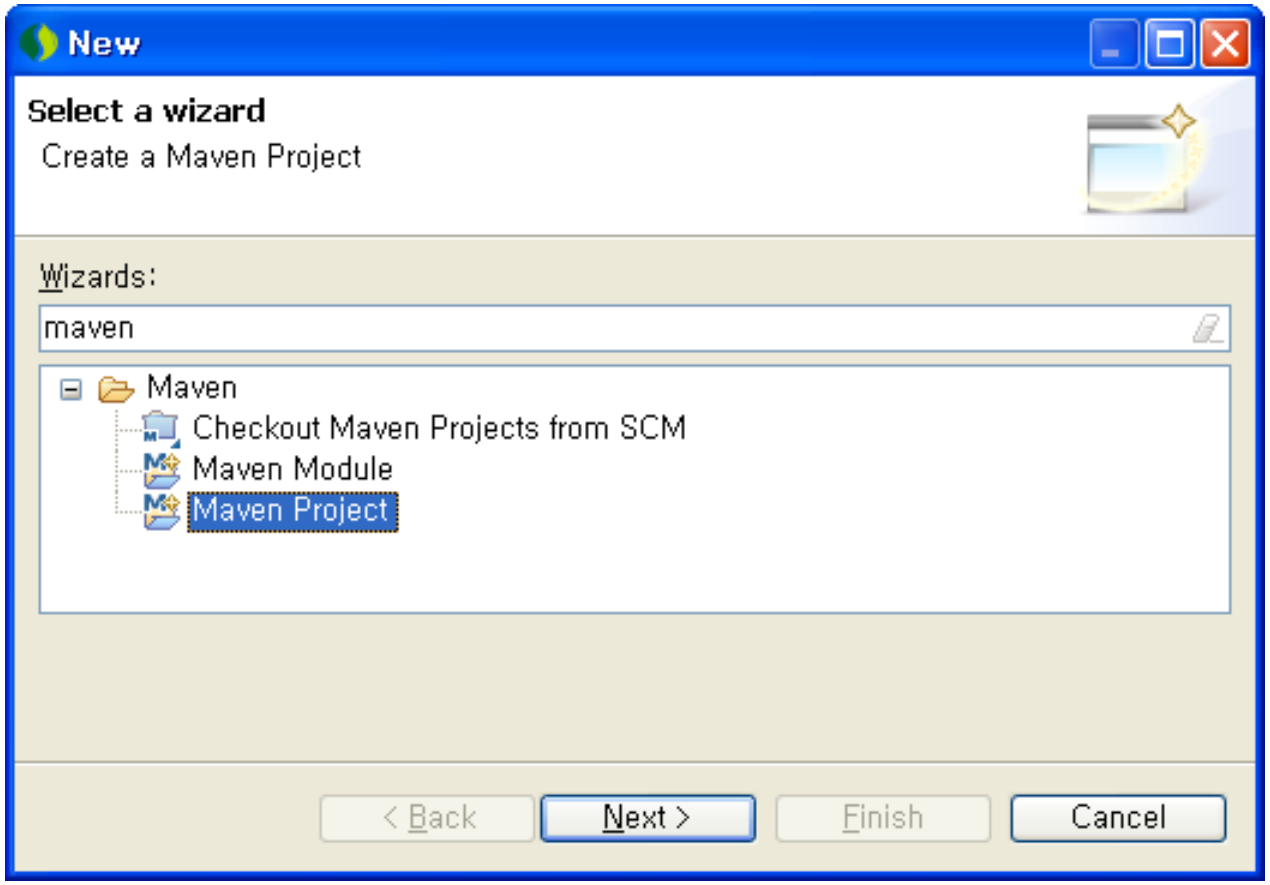


해결책

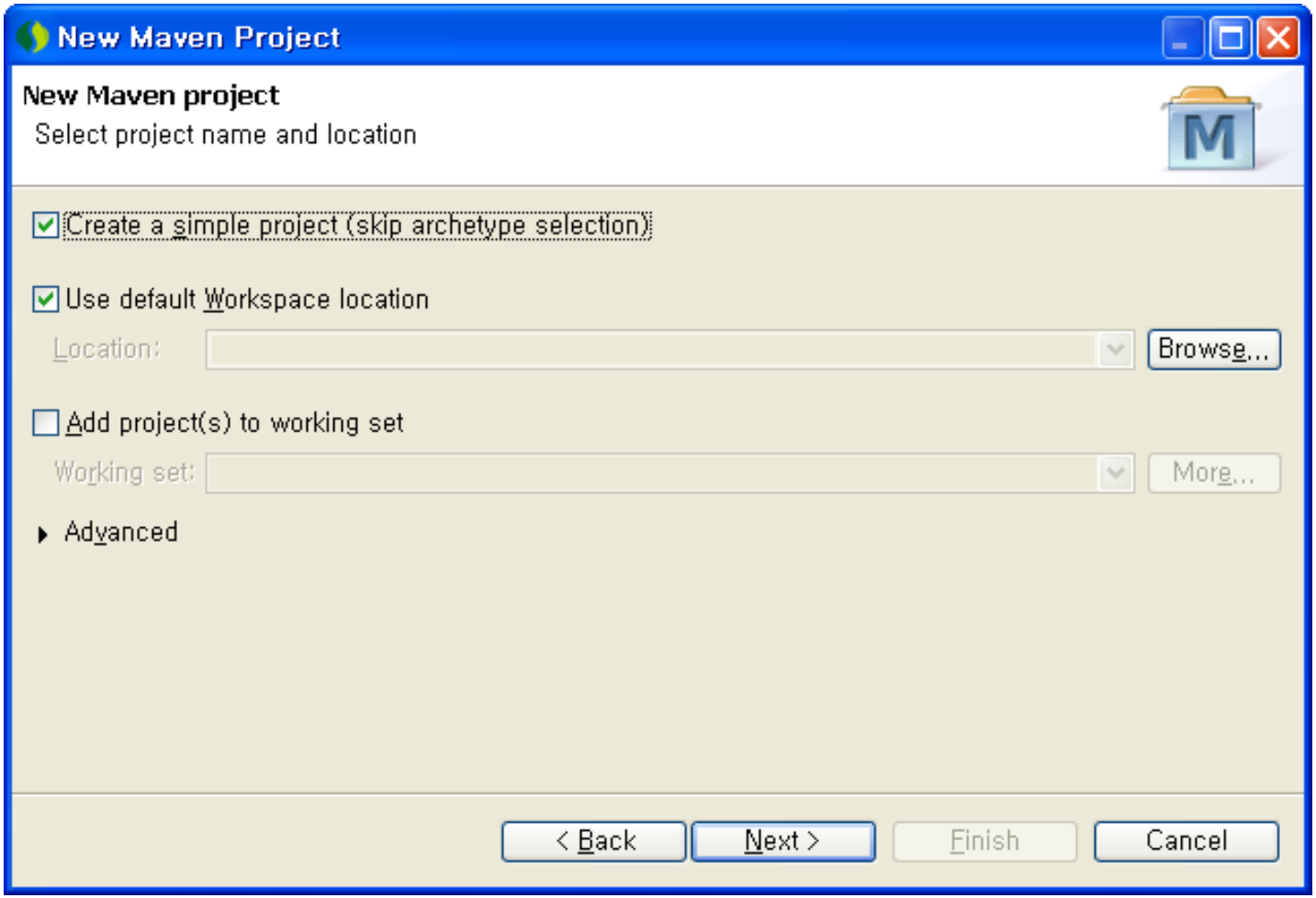
자바 웹 프로젝트에 대한 디렉토리 표준을 만들고 프로젝트를 자동으로 생성한다.

메이븐(Maven)이라는 도구가 등장함.

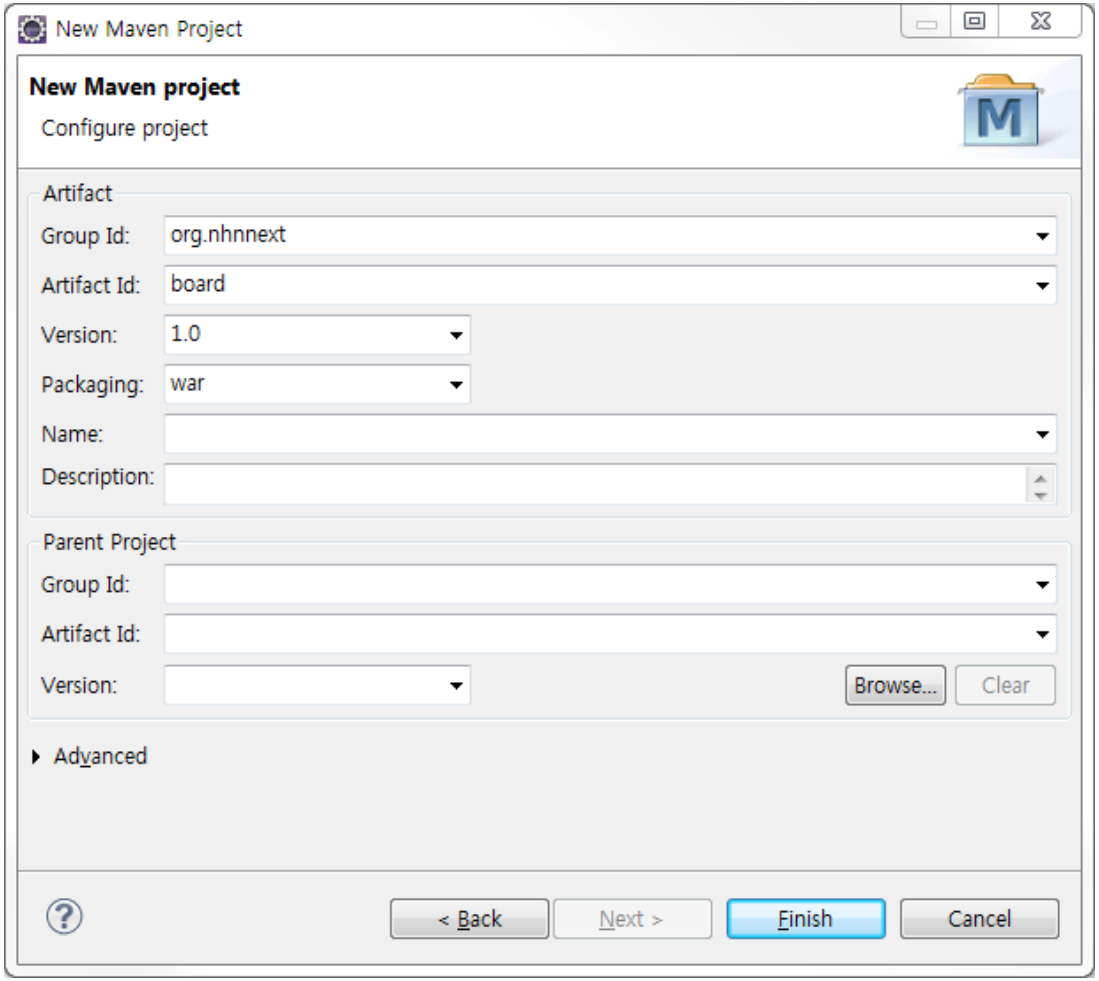
Cmd(or Ctrl) + N 단축키 > maven으로 검색 > Maven project 선택 > Next



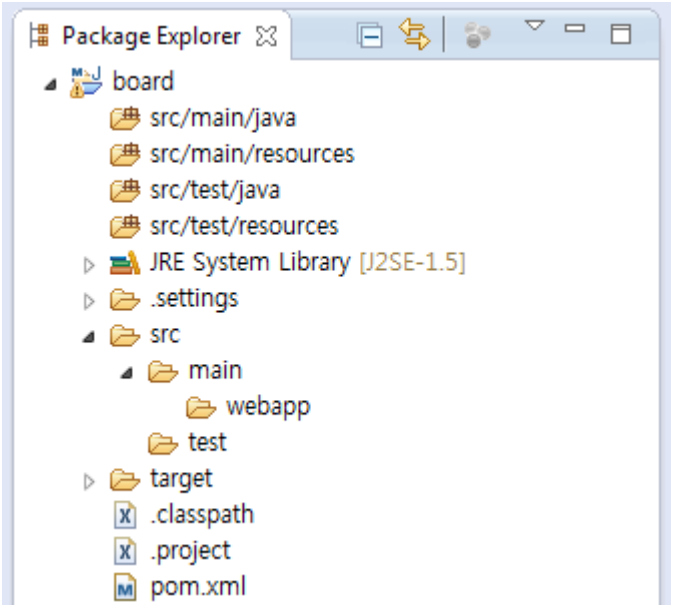
Create a simple project(….) 체크박스 선택 > Next



Group Id : 여러 개의 프로젝트를 대표하는 그룹 아이디(예. org.nhnnext)
Artifact Id : 프로젝트 고유 아이디(예. board)
Version : 프로젝트 버전
Packaging : war 선택(war는 web archive를 의미함)



메이븐 프로젝트의 기본 디렉토리 구조



src/main/java : 배포할 자바 소스 코드(production code) 디렉토리

src/main/resources : 배포할 설정 파일 디렉토리

src/test/java : 테스트 자바 소스 코드(test code) 디렉토리

src/test/resources : 테스트시 필요한 설정 파일 디렉토리

src/main/webapp : 웹 자원(html, css, javascript)을 관리하는 디렉토리

pom.xml : 메이븐 설정 파일

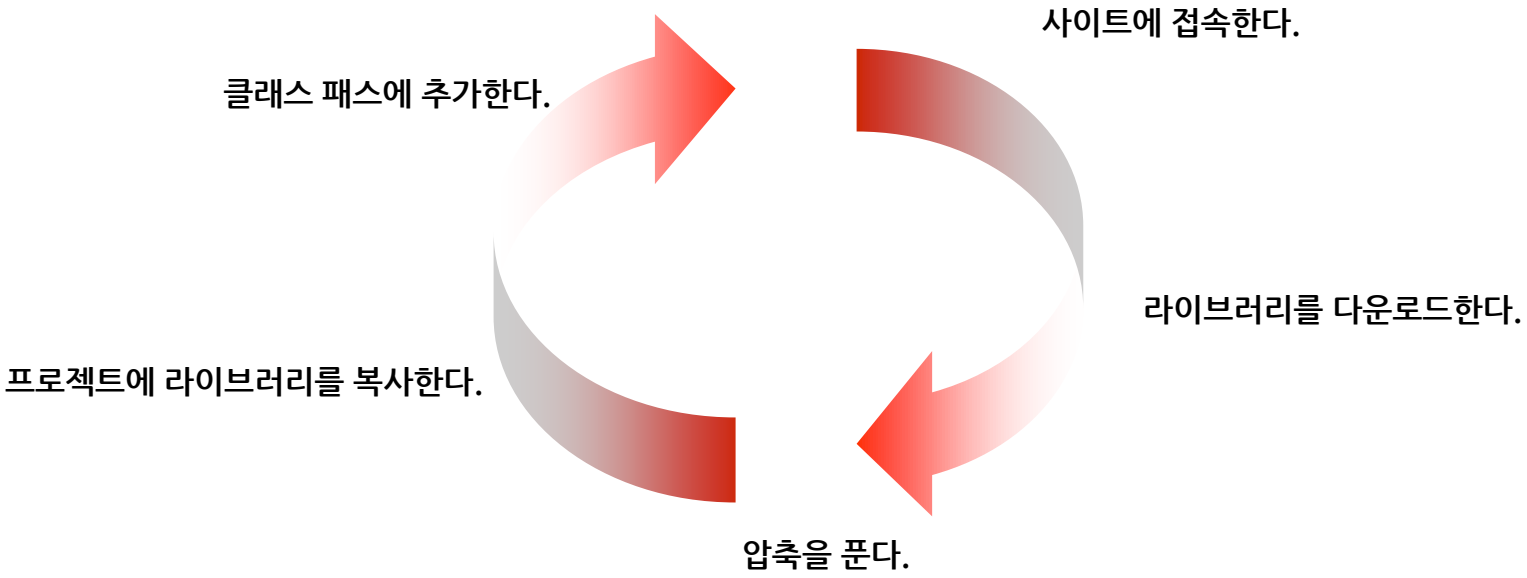
의존관계(dependency) 설정

–

자바 진영은 **오픈 소스 천국**이다. 오픈 소스가 없으면 아무 것도 할 수 없다.

모든 오픈 소스 클래스는 jar 압축 파일을 통해 배포

프로젝트에 오픈 소스 라이브러리 추가 Cycle



데모

Apache Commons Lang 라이브러리 추가

구글에서 apache commons lang 검색한다.

사이트 접속해 Download 메뉴 접근한다.

commons-lang-2.6-bin.zip 파일 다운로드한다.

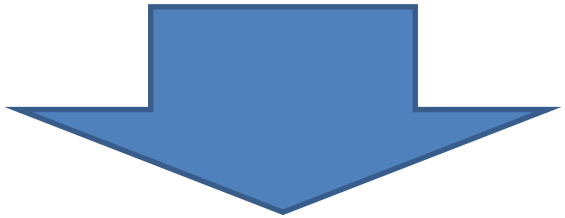
commons-lang-2.6-bin.zip 파일 압축을 푼다.

commons-lang-2.6.jar 파일을 프로젝트에 복사한다.(src/main/webapps/WEB-INF/lib가 표준)

Eclipse에서 commons-lang-2.6.jar 파일을 클래스 패스에 추가한다.

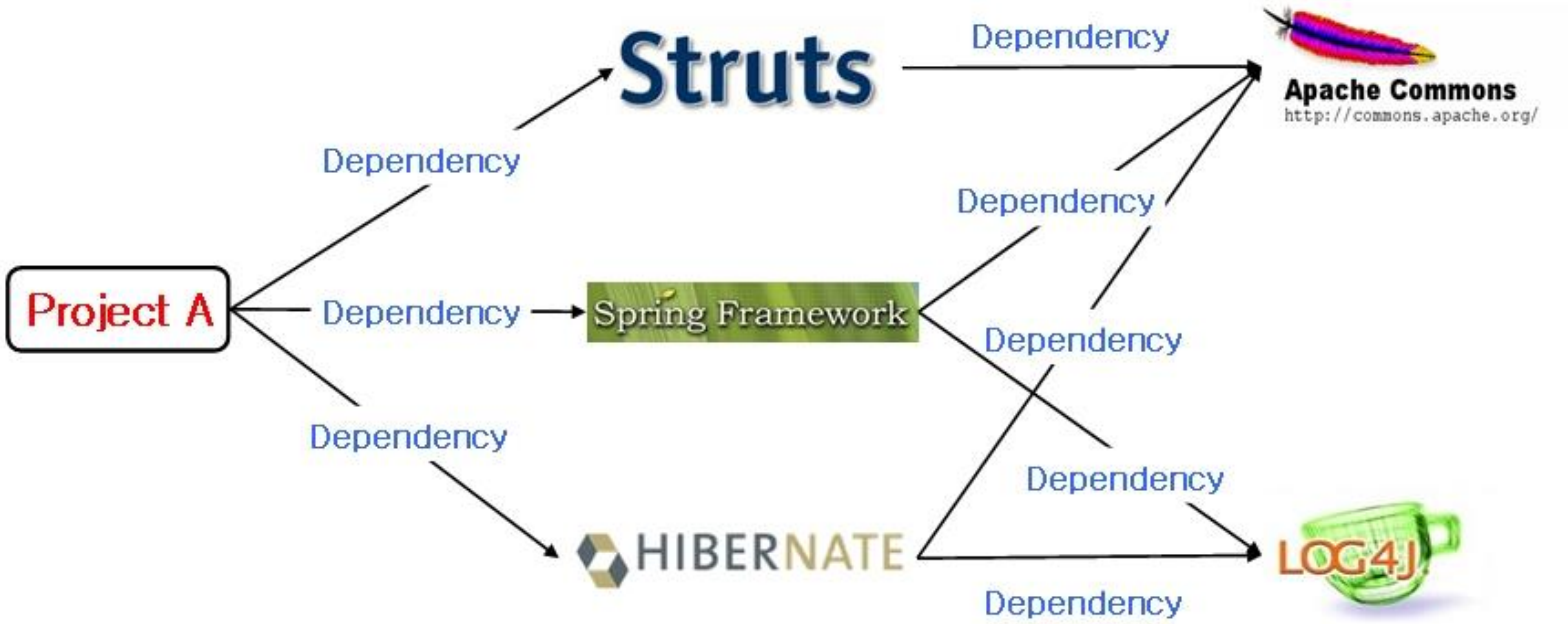
이슈

Apache Commons Lang 라이브러리의 버전이 2.6에서 3.1로 업그레이드 되었다.
자바 진영에서 기본으로 사용하는 오픈 소스 라이브러리는 수십 개이다. 많은 경우 100개도 넘는다.



해결책

모든 오픈 소스 라이브러리를 중앙 집중적으로 관리하는 저장소를 생성했다.
메이븐 설정을 통해 오픈 소스 라이브러리에 대한 의존성 관리를 할 수 있다.


Struts
Dependency
Apache Commons
http://commons.apache.org/
Dependency
Dependency
Project A
Dependency
Spring Framework
Dependency
Dependency
Dependency
HIBERNATE
LOG4J
Dependency

실습

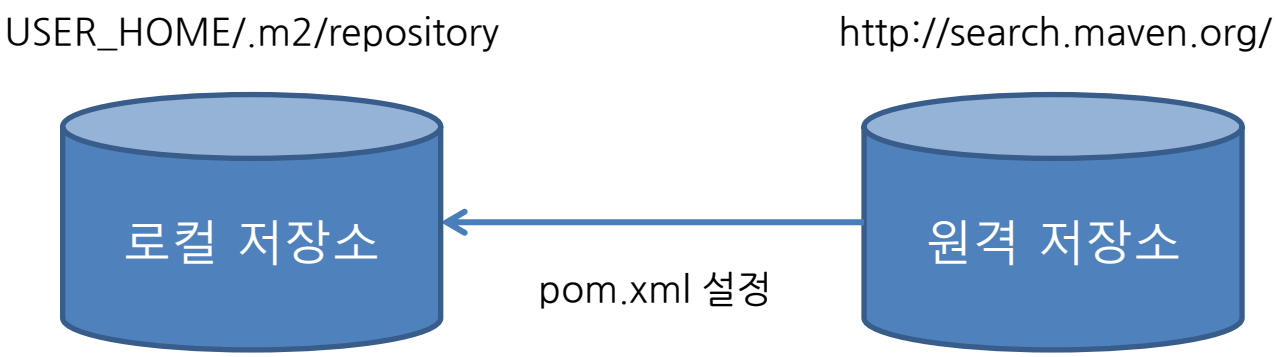
junit 라이브러리 추가(pom.xml)

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>3.8.1</version>
    </dependency>
</dependencies>
```

실습

3.8.1에서 4.11로 업그레이드

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.11</version>
    </dependency>
</dependencies>
```


USER_HOME/.m2/repository
http://search.maven.org/
로컬 저장소
원격 저장소
pom.xml 설정

로컬 저장소 jar 파일 생성 규칙

USER_HOME/.m2/repository 디렉토리가 base 디렉토리

{groupId}/{artifactId}/{version}/{artifactId}-{version}.jar

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.11</version>
    </dependency>
</dependencies>
```

USER_HOME/.m2/repository/junit/junit/4.11/junit-4.11.jar

Apache Commons Lang 라이브러리 추가 과정

http://search.maven.org/에서 “apache commons lang”으로 검색

3.1 최신 버전 선택

<dependency /> 영역 복사한 후 pom.xml에 붙여 넣기

```
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-lang3</artifactId>
    <version>3.1</version>
</dependency>
```

USER_HOME/.m2/repository/org/apache/commons/commons-lang3/3.1/commons-lang3-3.1.jar

프로젝트 빌드

자바 웹 프로젝트 배포 과정

로컬 개발 환경에서는 Eclipse가 컴파일 및 배포와 같은 모든 작업을 해준다.

하지만, 실 서비스(대부분 리눅스 또는 유닉스)에서는 Eclipse를 사용할 수 없다.

자바 웹 프로젝트 배포 과정

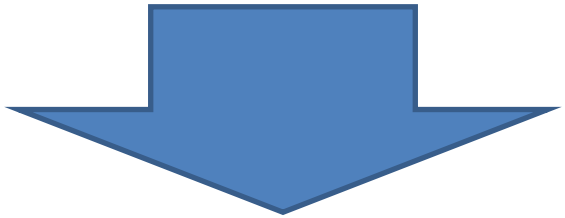
1. src/main/java 소스 코드를 컴파일 한다.
2. src/main/resources 설정 파일을 클래스 패스에 복사한다.
3. src/test/java 소스 코드를 컴파일 한다.
4. src/test/java 에서 컴파일한 클래스를 활용해 단위 테스트를 실행한다.
5. 단위 테스트가 성공하면 war(web archive) 파일로 압축한다.
6. 웹 애플리케이션 서버(Web Application Server, 이하 WAS)에 war 파일을 복사한다.
7. WAS를 시작한다.

이슈

자바 기반의 웹 프로젝트를 배포하기 위해 여러 단계를 거쳐야 한다.

여러 단계의 과정을 수동으로 진행하는 경우 일부 단계를 누락하는 경우 발생해 장애가 발생하는 경우가 종종 있다.

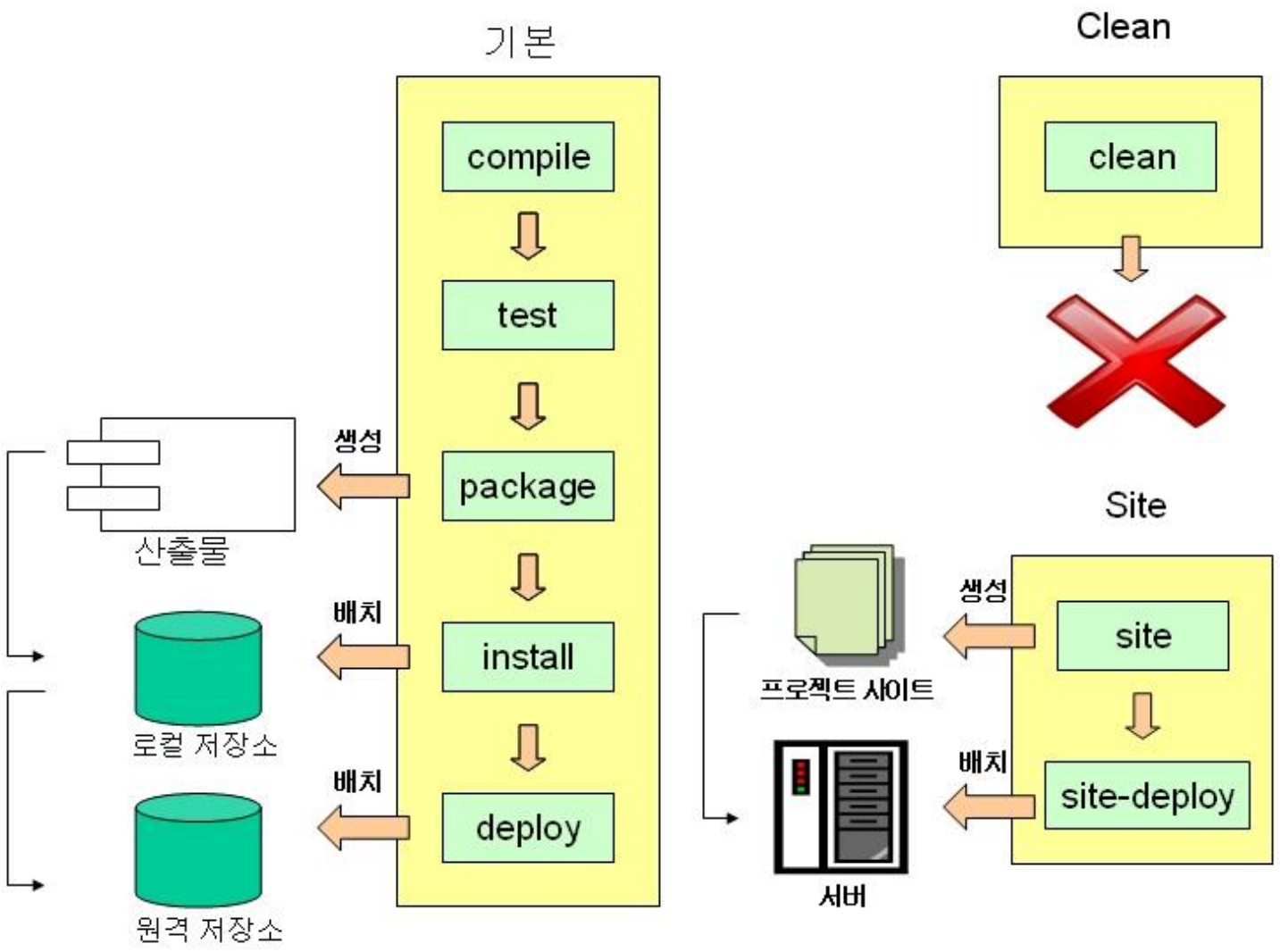
(사람이 개입해야 하는 모든 단계는 장애가 발생할 가능성이 있다.)

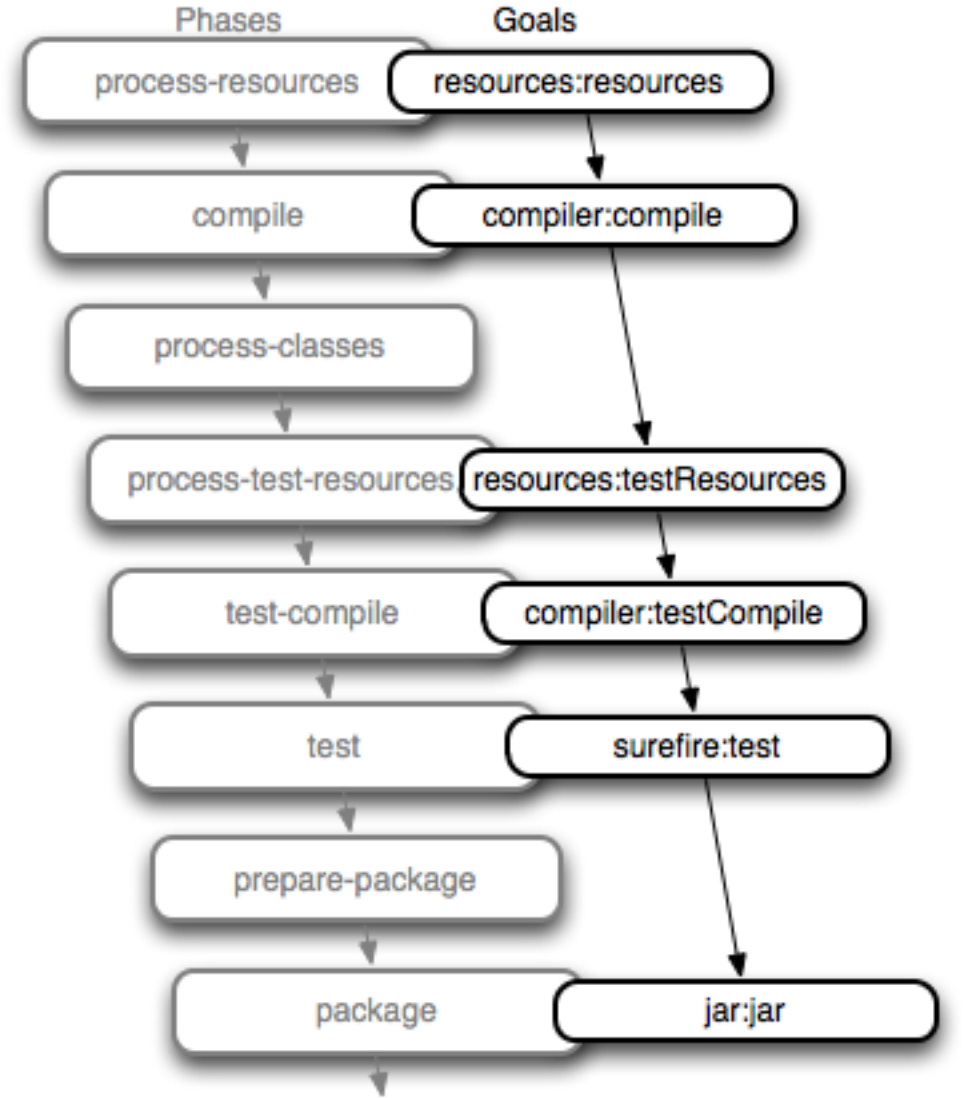


해결책

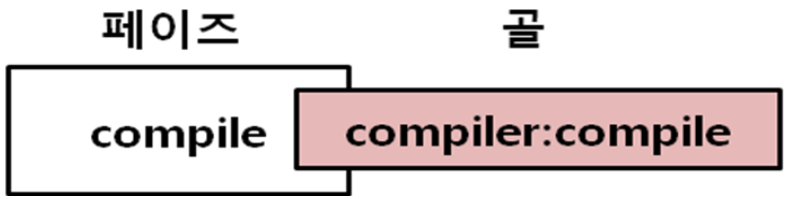
프로젝트 배포를 필요한 모든 단계를 자동화한다.

메이븐은 자바 프로젝트 배포를 위해 필요한 모든 과정을 표준화하고 있다.


기본
compile
test
package
install
deploy
생성
배치
배치
산출물
로컬 저장소
원격 저장소
Clean
clean
Site
site
site-deploy
생성
배치
프로젝트 사이트
서버



출처 : http://www.sonatype.com/books/mvnex-book/reference/simple-project-sect-lifecycle.html


페이즈
골
compile
compiler:compile

참고 서적 : 자바 세상의 빌드를 이끄는 메이븐, 박재성