# Example

## Packaging according to the deploy environment using Profile

1. Create maven project

CODE

```
mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my
```

2. Add properties to pom.xml

HTML/XML

```
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEn
         <!-- 기본 프로파일 -->
         <env>local</env>
    </properties>
```

3. profiles technology

HTML/XML

```
<profiles>
        <profile>
            <id>local</id>
            <activation>
                <property>
                    <name>env</name>
                    <value>local</value>
                </property>
            </activation>
            <properties>
                <env>local</env>
            </properties>
        </profile>
```

```
        <profile>
            <id>dev</id>
            <activation>
                <property>
                    <name>env</name>
                    <value>dev</value>
                </property>
            </activation>
            <properties>
                <env>dev</env>
            </properties>
        </profile>
        <profile>
            <id>test</id>
            <activation>
                <property>
                    <name>env</name>
                    <value>test</value>
                </property>
            </activation>
            <properties>
                <env>test</env>
            </properties>
        </profile>
        <profile>
            <id>real</id>
            <activation>
                <property>
                    <name>env</name>
                    <value>real</value>
                </property>
            </activation>
            <properties>
                <env>real</env>
            </properties>
        </profile>
    </profiles>
```

4. Add to build item

HTML/XML

```
<build>
        <finalName>ArtifactId</finalName>
        <resources>
            <resource>
                <directory>src/main/resources/${env}</directory>
            </resource>
        </resources>
        <testResources>
            <testResource>
                <directory>src/test/resources/${env}</directory>
            </testResource>
        </testResources>
```

5. Set targetPath and resource path in webResources of war-plugin in pom.xml

CODE

```
<!-- aaaa -->
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-war-plugin</artifactId>
    <version>2.1.1</version>
    <configuration>
    <webResources>
        <resource>
            <directory>src/main/filters/${env}</directory>
            <includes>
                <include>*.properties</include>
                <include>*.xml</include>
            </includes>
            <targetPath>WEB-INF/config</targetPath>
            <filtering>true</filtering>
        </resource>
    </webResources>
    </configuration>
</plugin>
```

6. Create a folder structure like this:

Click here to expand...

- src
  - main
    - java
  - resources
    - dev
      - jdbc.properties
      - log4j.properties
      - database-context.xml
    - local
      - jdbc.properties
      - log4j.properties
      - database-context.xml
    - real
      - jdbc.properties
      - log4j.properties
      - database-context.xml
    - test
      - jdbc.properties
      - log4j.properties
      - database-context.xml
    - webapp
  - resources
    - default
      - css
      - images
      - js
      - html5
      - jqgrid

        - jquery
    - test
        - java
        - resources

7. mvn clean package -P profilename

# Used in jar plugin

1. profiles technology (item 3 above)
2. 33

```HTML/XML
<plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-jar-plugin</artifactId>
                <version>2.4</version>
                <configuration>
                    <archive>
                        <manifest>
                            <mainClass>MyMainClass</mainClass>
                            <packageName>
                            </packageName>
                        </manifest>
                        <manifestEntries>
                            <mode>development</mode>
                            <url>${project.url}</url>
                        </manifestEntries>
                    </archive>
                </configuration>
            </plugin>
```

3. Described in the resources section

HTML/XML

```
<build>
   <resources>
       <resource>
           <directory>src/main/resources/${env}</directory>
        </resource>
   </resources>
</build>
```

4. Create the following folder in src/main/resources and locate the resource file
    a. local
    b. dev
    c. test
    d. real
5. mvn package -P{local, dev, test, real}

# References