



zenika  
<animés par la passion>

# Maven Best Practices

Version 1.18-SNAPSHOT

2018-08-23

# Table of Contents

1. Config files location .....	2
2. Checkstyle : check javadoc .....	2
3. Add version and date to AsciiDoc PDFs .....	3
4. Javadoc generation with UML diagrams .....	4
5. Install provided dependencies in local repository .....	5
6. To generate AsciiDoc PDF files .....	6
7. SonarQube with Jacoco for coverage .....	6
8. Appendix .....	11
8.1. Revision marks .....	11

*Table 1. History*

Date	Author	Detail
2018-08-23	bcouetil	Initial commit

# 1. Config files location

Config files have to be put in the right folder in Eclipse.

- **src/main/resources/**
  - Only files that will is not likely to be modified, because it will be in the jar
- **config/**
  - Files that is likely to be modified on IS
  - Don't forget to put them manually on IS
- **src/test/resources/**
  - File used in JUnit tests only for this sub-module
- **../src/test/shared-resources**
  - Files used in JUnit tests accross multiple modules
  - requires some maven configuration



TODO give Maven details for shared-resources

## 2. Checkstyle : check javadoc

With Checkstyle, you can enforce continuous javadoc check

*pom.xml plugin*

```
<!-- checkstyle to fail the build on javadoc warnings -->
<!-- to skip : mvn install -Dcheckstyle.skip=true -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-checkstyle-plugin</artifactId>
  <version>2.17</version>
  <executions>
    <execution>
      <id>validate</id>
      <phase>validate</phase>
      <configuration>
        <configLocation>checkstyle-javadoc.xml</configLocation>
        <encoding>UTF-8</encoding>
        <consoleOutput>true</consoleOutput>
        <failsOnError>true</failsOnError>
        <linkXRef>false</linkXRef>
      </configuration>
      <goals>
        <goal>check</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

*checkstyle-javadoc.xml to be created in the root project*

```
<?xml version="1.0"?>
<!DOCTYPE module PUBLIC
    "-//Puppy Crawl//DTD Check Configuration 1.2//EN"
    "http://www.puppycrawl.com/dtds/configuration_1_2.dtd">
<module name="Checker">
    <module name="TreeWalker">
        <module name="JavadocMethod"/>
        <module name="JavadocType"/>
        <module name="JavadocVariable"/>
        <module name="JavadocStyle"/>
    </module>
</module>
```

### 3. Add version and date to AsciiDoc PDFs

```

<plugins>

  <plugin>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>buildnumber-maven-plugin</artifactId>
    <version>1.2</version>
    <executions>
      <execution>
        <phase>validate</phase>
        <goals>
          <goal>create-timestamp</goal>
        </goals>
      </execution>
    </executions>
    <configuration>
      <timestampFormat>yyyy-MM-dd</timestampFormat>
      <timestampPropertyName>build.date</timestampPropertyName>
    </configuration>
  </plugin>

  <!-- Ant tasks plugin -->
  <!-- single usage : mvn antrun:run -->
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-antrun-plugin</artifactId>
    <version>1.7</version>
    <inherited>true</inherited>
    <executions>
      <execution>
        <!-- add version to generated pdf filenames -->
        <id>pdfsAddVersion</id>
        <configuration>
          <failOnError>false</failOnError>
          <target name="add version and date to all generated pdf filenames">
            <move todir="${project.build.directory}/generated-docs" includeemptydirs="false">
              <fileset dir="${project.build.directory}/generated-docs" />
              <mapper type="glob" from="*.pdf" to="*_V${project.version}_${build.date}.pdf" />
            </move>
          </target>
        </configuration>
        <goals>
          <goal>run</goal>
        </goals>
      </execution>
    </executions>
  </plugin>

</plugins>

```

## 4. Javadoc generation with UML diagrams

```

<!-- javadoc html, fix or generate -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-javadoc-plugin</artifactId>
  <version>2.10.4</version>
  <configuration>
    <!-- usage : javadoc:javadoc or javadoc:jar -->
    <show>public</show>
    <reportOutputDirectory>${project.reporting.outputDirectory}</reportOutputDirectory>
    <destDir>javadoc</destDir>
    <!-- for UML diagram in javadoc:javadoc -->
    <!-- Locally : need http://www.graphviz.org/Download_windows.php to work -->
    <!-- and add "C:\Program Files (x86)\Graphviz\bin" to windows path -->
    <doclet>org.umlgraph.doclet.UmlGraphDoc</doclet>
    <docletArtifact>
      <groupId>org.umlgraph</groupId>
      <artifactId>umlgraph</artifactId>
      <version>5.6.6</version>
    </docletArtifact>
    <additionalparam>-views -attributes -visibility -types -enumerations -enumconstants</additionalparam>
    <useStandardDocletOptions>true</useStandardDocletOptions>
  </configuration>
</plugin>

```

## 5. Install provided dependencies in local repository

```

<plugins>
  <!-- install WM jars in local repository -->
  <!-- part of mvn clean because maven check them early in the process -->
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-install-plugin</artifactId>
    <version>2.5.2</version>
    <!-- We do not want children attempting to install these jars to the repository -->
    <inherited>false</inherited>
    <executions>
      <execution>
        <id>wm-isclient95</id>
        <phase>clean</phase>
        <goals>
          <goal>install-file</goal>
        </goals>
        <configuration>
          <file>lib/wm9.5/wm-isclient-9.5.jar</file>
          <groupId>webmethods</groupId>
          <artifactId>wm-isclient</artifactId>
          <version>9.5</version>
          <packaging>jar</packaging>
        </configuration>
      </execution>
    </executions>
  </plugin>
</plugins>

```

## 6. To generate AsciiDoc PDF files

*pom.xml*

```
<plugins>
  <!-- to generate asciidoc pdf documents -->
  <!-- part of mvn install -->
  <!-- single usage : mvn asciidoctor:process-asciidoc -->
  <!-- We don't bind it to the official phase to choose the moment in Jenkins pipeline -->
  <plugin>
    <groupId>org.asciidoctor</groupId>
    <artifactId>asciidoctor-maven-plugin</artifactId>
    <version>1.5.5</version>
    <dependencies>
      <dependency>
        <groupId>org.asciidoctor</groupId>
        <artifactId>asciidoctorj-pdf</artifactId>
        <version>1.5.0-alpha.14</version>
      </dependency>
      <dependency>
        <groupId>org.asciidoctor</groupId>
        <artifactId>asciidoctorj-diagram</artifactId>
        <version>1.5.4</version>
      </dependency>
    </dependencies>
    <configuration>
      <backend>pdf</backend>
      <sourceDirectory>src/docs/asciidoc</sourceDirectory>
      <sourceHighlighter>rouge</sourceHighlighter>
      <requires>
        <require>asciidoctor-diagram</require>
      </requires>
      <!-- Attributes common to all output formats -->
      <attributes>
        <imagesdir>${project.build.directory}/generated-docs/images</imagesdir>
        <pdf-style>${user.dir}/src/docs/asciidoc/themes/cg-theme.yml</pdf-style>
        <icons>font</icons>
        <pagenums />
        <toc />
        <idprefix />
        <idseparator>-</idseparator>
        <!-- custom -->
        <source-dir>../../main/java</source-dir>
        <test-dir>../../test/java</test-dir>
        <project-version>${project.version}</project-version>
        <root-project-dir>${user.dir}</root-project-dir>
        <history-dir>${project.build.directory}/generated-docs/history</history-dir>
        <project-images-dir>${project.basedir}/src/main/resources/images</project-images-dir>
      </attributes>
    </configuration>
  </plugin>
</plugins>
```

## 7. SonarQube with Jacoco for coverage



<https://www.sonarqube.org>

SonarQube ensures code quality with static analysis and Jacoco checks code coverage.



## *pom.xml properties*

```
<properties>
  <custom.ut.skip>${skipTests}</custom.ut.skip>
  <sonar.java.coveragePlugin>jacoco</sonar.java.coveragePlugin>
  <jacoco.reportPath>../target/jacoco.exec</jacoco.reportPath>
  <jacoco.itReportPath>../target/jacoco-it.exec</jacoco.itReportPath>
  <sonar.jacoco.reportPaths>${jacoco.reportPath}, ${jacoco.itReportPath}</sonar.jacoco.reportPaths>

  <sonar.coverage.exclusions>**/WmCall.*,**/Broker.*,**/UniversalMessaging.*,**/MsgServerBroker.*,**/UmListener.*,**/PerfLogger.*,**/elastic/*DataSender.*</sonar.coverage.exclusions>
  <sonar.host.url>http://localhost:9000</sonar.host.url>
  <sonar.scm.disabled>true</sonar.scm.disabled>
  <sonar.scm.provider>git</sonar.scm.provider>
</properties>
```

## *pom.xml without powermock static*

```
<dependencies>

  <!-- For unit tests coverage in Sonar -->
  <dependency>
    <groupId>org.sonarsource.java</groupId>
    <artifactId>sonar-jacoco-listeners</artifactId>
    <version>4.9.0.9858</version>
    <scope>test</scope>
  </dependency>

</dependencies>

<plugins>

  <!-- SonarQube -->
  <plugin>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>sonar-maven-plugin</artifactId>
    <version>3.2</version>
  </plugin>

  <!-- handling unit tests coverage with Jacco -->
  <plugin>
    <groupId>org.jacoco</groupId>
    <artifactId>jacoco-maven-plugin</artifactId>
    <version>0.8.0</version>
    <executions>
      <execution>
        <id>pre-unit-test</id>
        <phase>test-compile</phase>
        <goals>
          <goal>prepare-agent</goal>
        </goals>
        <configuration>
          <destFile>${sonar.jacoco.reportPath}</destFile>
          <dataFile>${sonar.jacoco.reportPath}</dataFile>
          <append>true</append>
        </configuration>
      </execution>
      <execution>
        <id>prepare-jacoco-agent-it</id>
        <phase>pre-integration-test</phase>
        <goals>
          <goal>prepare-agent-integration</goal>
        </goals>
        <configuration>
          <destFile>${sonar.jacoco.itReportPath}</destFile>
          <dataFile>${sonar.jacoco.itReportPath}</dataFile>
          <append>true</append>
        </configuration>
      </execution>
    </executions>
  </plugin>

</plugins>
```

```

    </execution>
  </executions>
</plugin>

<!-- Unit Tests -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <!-- version 2.19.1 is broken on jenkins -->
  <version>2.18.1</version>
  <configuration>
    <testFailureIgnore>>false</testFailureIgnore>
    <runOrder>alphabetical</runOrder>
    <skipTests>${custom.ut.skip}</skipTests>
    <properties>
      <property>
        <name>listener</name>
        <value>org.sonar.java.jacoco.JUnitListener</value>
      </property>
    </properties>
  </configuration>
</plugin>

<!-- Integration Tests -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-failsafe-plugin</artifactId>
  <!-- version 2.19.1 is broken on jenkins -->
  <version>2.18.1</version>
  <configuration>
    <runOrder>alphabetical</runOrder>
    <properties>
      <property>
        <name>listener</name>
        <value>org.sonar.java.jacoco.JUnitListener</value>
      </property>
    </properties>
  </configuration>
  <executions>
    <execution>
      <id>integration-tests</id>
      <phase>integration-test</phase>
      <goals>
        <goal>integration-test</goal>
      </goals>
    </execution>
    <!-- to exit in error on test fail -->
    <execution>
      <id>verify</id>
      <phase>verify</phase>
      <goals>
        <goal>verify</goal>
      </goals>
    </execution>
  </executions>
</plugin>
</plugins>

```

*pom.xml with powermock : instrumentation in conflict, offline jacoco instrumentation is needed*

```

<dependencies>

  <!-- For unit tests coverage in Sonar -->
  <dependency>
    <groupId>org.jacoco</groupId>
    <artifactId>org.jacoco.agent</artifactId>
    <classifier>runtime</classifier>
    <version>0.8.0</version>
  </dependency>

```

```

        <scope>test</scope>
      </dependency>
    </dependencies>

    <plugins>

      <!-- SonarQube -->
      <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>sonar-maven-plugin</artifactId>
        <version>3.2</version>
      </plugin>

      <!-- handling unit tests coverage with Jacco -->
      <!-- offline instrumentation is mandatory when using other instrumentation framework such as PowerMock -->
      <!-- https://github.com/powermock/powermock/wiki/Code-coverage-with-JaCoCo -->
      <!-- to separate UT and IT : -->
      <!-- (1) mvn test jacoco:restore-instrumented-classes -->
      <!-- (2) mvn install -Dcustom.ut.skip=true -Dcheckstyle.skip=true -->
      <plugin>
        <groupId>org.jacoco</groupId>
        <artifactId>jacoco-maven-plugin</artifactId>
        <version>0.8.0</version>
        <executions>
          <execution>
            <id>jacoco-instrument</id>
            <phase>test-compile</phase>
            <goals>
              <goal>instrument</goal>
            </goals>
            <configuration>
              <skip>${skipTests}</skip>
            </configuration>
          </execution>
          <execution>
            <id>jacoco-restore-instrumented-classes</id>
            <phase>post-integration-test</phase>
            <goals>
              <goal>restore-instrumented-classes</goal>
            </goals>
            <configuration>
              <skip>${skipTests}</skip>
            </configuration>
          </execution>
        </executions>
      </plugin>

      <!-- Unit Tests -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-plugin</artifactId>
        <!-- version 2.19.1 is broken on jenkins -->
        <version>2.18.1</version>
        <configuration>
          <testFailureIgnore>>false</testFailureIgnore>
          <runOrder>alphabetical</runOrder>
          <skipTests>${custom.ut.skip}</skipTests>
          <systemPropertyVariables>
            <jacoco-agent.destfile>${jacoco.reportPath}</jacoco-agent.destfile>
          </systemPropertyVariables>
        </configuration>
      </plugin>

      <!-- Integration Tests -->
      <!-- usage full test : mvn integration-test -->
      <!-- usage only IT (but does not fill jacoco-it) : mvn test-compile failsafe:integration-test -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>

```

```

<artifactId>maven-failsafe-plugin</artifactId>
<!-- version 2.19.1 is broken on jenkins -->
<version>2.18.1</version>
<configuration>
  <runOrder>alphabetical</runOrder>
  <systemPropertyVariables>
    <jacoco-agent.destfile>${jacoco.itReportPath}</jacoco-agent.destfile>
  </systemPropertyVariables>
</configuration>
<executions>
  <execution>
    <id>integration-tests</id>
    <phase>integration-test</phase>
    <goals>
      <goal>integration-test</goal>
    </goals>
  </execution>
  <execution>
    <id>verify</id>
    <phase>verify</phase>
    <goals>
      <goal>verify</goal>
    </goals>
  </execution>
</executions>
</plugin>

```

## 8. Appendix

### 8.1. Revision marks

*Differences since last tag*

