The Stairway to Maven

The JAKARTA Project Build
And Comprehension Tool

DI Siegfried GÖSCHL
siegfried.goeschl@it2one.at
How do you build and maintain complex software projects written in Java?!

- The Chinese Approach ...
  - You can always find a student or two for the build

- Make or better not to Make – that’s the question ...
  - Ever tried to run a UNIX makefile on Windows?!

- All ANTs are equal and some are even more equal ...
  - Tired of copy/pasting ANT scripts?!
  - Need something like ANT with more bells and whistles?!
The intent of Maven is to make intra-project development highly manageable in the hopes of providing more time for cross-project development. You might call it cross-project pollination or the sharing of project development knowledge, this is what Maven attempts to encourage.

Jason van Zyl
Maven For Rookies

- Maven was initially developed for building Turbine
- Maven matured into an open source software engineering platform
- The core functionality is automated project building, distribution and website creation
- A project is described with a XML Project Object Model (POM)

- The POM defines how to build a project and the external dependencies
- The Maven functionality is implemented in terms of plugins
- The plugins are written in Jelly
- JARs are downloaded from a remote repository and stored into a local repository
Project Object Model (POM)

- Verbose project description
- Company information
- List of developer and their roles
- Mailing list support
- CVS server access
- Source code and unit test code path
- JAR dependencies
  - Versioning
  - How to download
The Core Plugins

Build Plugins

- The distribution plugin allows building a binary or a source distribution from the command line
- An **ANT** build file can be generated and distributed to allow **ANT** users to build the distribution
- Other plugins allows
  - Creating a WAR file
  - Creating a JAR file
  - Deploying the distribution automatically

No more thrills building your project!
Site Plugin

- Site creation based on Project Object Model
- Site generation uses XDOC
  - Maven generated XDOC
  - Manually written documentation
- DVSL transforms XDOC into HTML
- Site layout is defined through `xdoc/navigation.xml`
- Site appearance is customizable through properties defined in `project.properties`
### Site Plugin

#### Home
- Front Page
- IT20one Team
- Progress
- Todo

#### Design
- Goals
- Overview

#### ESD
- Overview
- Properties
- Metadata

#### XML-RPC
- Overview
- Functions
- Datatypes

### PartyCodeIdTypes

A PartyCodeIdType classifies an identifier for looking a party

<table>
<thead>
<tr>
<th>PartyCodeIdType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal</td>
<td>The internally used UUID for a party</td>
</tr>
<tr>
<td>external</td>
<td>A unique external id, usually the customer id of the target system</td>
</tr>
<tr>
<td>duns</td>
<td>A DUNS code of the party</td>
</tr>
<tr>
<td>iln</td>
<td>An ILN code of the party</td>
</tr>
<tr>
<td>government_id</td>
<td>A unique government id such as &quot;Firmenbuchnummer&quot;</td>
</tr>
</tbody>
</table>

### Language Codes

The [ISO 639-1 Codes](https://www.loc.gov/standards/iso639-2/iso639-2b.html) are used
### The Core Plugins

#### Site Plugin

**Project Reports**
- PMD Report

**Project Documentation**
- Front Page
- Project Info
- Project Reports
  - Change Log
  - Developer Activity
- File Activity
- Unit Tests
- Metric Results
- Checkstyle Report
- JavaDocs
- Source XReference
- Development Process

<table>
<thead>
<tr>
<th>Document</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change Log</td>
<td>This document provides a history of the most recent changes made to the project.</td>
</tr>
<tr>
<td>Developer Activity</td>
<td>This document provides a report of activity in terms of CVS commits and breaks it out by developer.</td>
</tr>
<tr>
<td>File Activity</td>
<td>This document provides a report of activity in terms of CVS commits and breaks it out by file.</td>
</tr>
<tr>
<td>Unit Tests</td>
<td>This document provides the results of the unit tests that are part of this project. Successes and failures are noted.</td>
</tr>
<tr>
<td>Metric Results</td>
<td>This document provides information on various source code metrics that have been computed. These metrics can provide useful information regarding the abstractness and total number of classes.</td>
</tr>
<tr>
<td>Checkstyle Report</td>
<td>This document provides the results of the Checkstyle report. This report provides an indication of how well this project complies with its coding conventions.</td>
</tr>
</tbody>
</table>
The Core Plugins

CheckStyle Plugin

- Various coding styles can be defined
  - Sun coding convention
  - Turbine coding convention
  - Roll your own coding convention (if you really have to)
- Rules are customizable through *project.properties*
  - Maximum line length of 100 instead of 80 characters
  - Patterns for constants and variables
  - Setting a different tab width

Just in case you have a coding convention ...
## CheckStyle Plugin

A screenshot of the CheckStyle results showing various errors. The errors include:

- **Unused import - java.net.URL**
- **Unused import - java.net.URLClassLoader**
- **Unused import - java.util Enumeration**
- **Unused import - java.util.HashMap**
- **'static' modifier out of order with the JLS suggestions.**
- **'static' modifier out of order with the JLS suggestions.**
- **'static' modifier out of order with the JLS suggestions.**
- **'static' modifier out of order with the JLS suggestions.**
- **variable 'log' must match pattern '^[A-Z][\_\-]?[A-Z0-9]*'**
- **Cast needs to be followed by whitespace.**
- **Expected an @return tag.**
- **Expected @param tag for 'pluginName'.**
- **'.' needs to be followed by whitespace.**

The screenshot also shows the file path `org/apache/maven/app/PluginManager.java` with a line number of 130.
The Core Plugins

Activity Plugin

- Plugin accesses CVS directly to extract development activity information
- Extracts information about
  - Changes and commits
  - Number of commits per developer
  - Number of changes for a file
- Ongoing work to support VSS and ClearCase

No more digging through CVS changelog
The Core Plugins

Activity Plugin

<table>
<thead>
<tr>
<th>Date</th>
<th>Author</th>
<th>Files/Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002-09-04</td>
<td>Siegfried Goeschl</td>
<td>junitpp.jpr.local - v1.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>maven.xml - v1.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>project.properties - v1.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>project.xml - v1.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xdocs/commandline.xml - v1.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xdocs/index.xml - v1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xdocs/stylesheets/project.xml - v1.2</td>
</tr>
<tr>
<td>2002-09-03</td>
<td>Siegfried Goeschl</td>
<td>Cleaned up site and added documentation</td>
</tr>
<tr>
<td>20:18:20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2002-09-03</td>
<td>Siegfried Goeschl</td>
<td>maven.xml - v1.3</td>
</tr>
<tr>
<td>20:11:08</td>
<td></td>
<td>Commented out the PMD target</td>
</tr>
<tr>
<td>2002-09-03</td>
<td>Siegfried Goeschl</td>
<td>JUNIT.library - v1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>junitpp.jpr - v1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>junitpp.jpr.local - v1.3</td>
</tr>
</tbody>
</table>
The Core Plugins

Test Plugin

- Runs JUNIT test case
- The plugin creates a XDOC report
- The XDOC report is then transformed to HTML
- The JUNIT test report is then integrated within the generated project site

I hope you have some tests?!
The Core Plugins

Test Plugin

Test Cases

[summary] [package list] [test cases]

AllTests

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Type</th>
<th>Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>testGetKeyWithBaseNameKey</td>
<td>Success</td>
<td></td>
<td>0,000</td>
</tr>
<tr>
<td>testGetKeyWithBaseKey</td>
<td>Success</td>
<td></td>
<td>0,000</td>
</tr>
<tr>
<td>testGetKeyWithClassNameKey</td>
<td>Success</td>
<td></td>
<td>0,000</td>
</tr>
<tr>
<td>testGetKeyWithKey</td>
<td>Success</td>
<td></td>
<td>0,000</td>
</tr>
<tr>
<td>testGetInteger</td>
<td>Success</td>
<td></td>
<td>0,000</td>
</tr>
<tr>
<td>testGetUndefinedKey</td>
<td>Success</td>
<td></td>
<td>0,000</td>
</tr>
</tbody>
</table>
JDepend Plugin

- Is using **JDepend** from ClarkWare
- Reports package dependencies
- Finds cyclic dependencies
- Generates a HTML report to be included into project website
### Packages

- [summary]
- [packages]
- [cycles]
- [explanations]

#### junit.extensions

<table>
<thead>
<tr>
<th>Afferent Couplings</th>
<th>Efferent Couplings</th>
<th>Abstractness</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>17%</td>
</tr>
</tbody>
</table>

**Abstract Classes**

- ConfigurableTest

**Concrete Classes**

- ActiveTest
- ConfigurableTestCase
- ConfigurableTestSetup
- PPTestResult
- PPTestRunner

**Used by Packages**

- test.junit.extensions
- test.junit.extensions.util
The Core Plugins

Java Cross Reference Plugin

- Transforms source files into HTML files
- Provides syntax coloring and hypertext linking
- Allows linking between reports and actual source code as done with
  - CheckStyle plugin
  - PMD plugin
Java Cross Reference Plugin

```java
public class ConfigurableTestCase extends TestCase implements Configur

    /** Cached configuration. */
    protected Configuration conf;

    /**
     * Creates instance of configurable test case.
     *
     * @param name Name of this test case.
     */
    public ConfigurableTestCase(String name) {
        super(name);
        ConfigurationFactory.init(this.getClass());
    }

    /**
     * Retrieves configuration of this test case.
     */
```
JavaDoc Plugin

Package junit.extensions

Interface Summary

ConfigurableTest

Test that provides access to configuration parameters.

Class Summary

ActiveTest
A Decorator that runs a test in a separate thread.

ConfigurableTest
Base class for test cases which are configured using a configuration file instead of hardcoding test data in the fixture.

ConfigurableTestCase
A Decorator to set up and tear down additional fixture state using configuration.

PPTestResult
A PPTestResult collects the results of executing a test case and implements verbose output and delaying.
Custom Plugins

- Everything in **Maven** is a plugin
- Feel free to roll and contribute your own plugins
- Deployment of plugins as JAR files being dropped into the maven plugin directory
- There are many plugins available
  - CLOVER Code Coverage Analyzer
  - PMD Static Source Code Analyzer
  - JavaNCSS Java Source Code Metric Analyzer
  - Word2HTML
  - ...

Stairway To Maven
Custom Plugins

Clover Test Coverage Plugin

- Commercial software from *The Cortex*
- Instruments the JUNIT test case code
- Generates a HTML test coverage report
  - Conditional Coverage
  - Statement Coverage
  - Method Coverage
- Integrated into generated project site through Maven

Another good reason for unit tests !!
Stairway To Maven
Clover Test Coverage Plugin

```java
/**
 * Loads the contents of a file in a String array.
 * @return the lines in the file
 * @param aFileName the name of the file to load
 * @throws IOException error occurred
 */

private String[] getLines(String aFileName) throws IOException {
    final LineNumberReader lnr =
        new LineNumberReader(new FileReader(aFileName));
    final ArrayList lines = new ArrayList();
    while (true) {
        final String l = lnr.readLine();
        if (l == null) {
            break;
        }
        lines.add(l);
    }

    return (String[]) lines.toArray(new String[0]);
```
PMD Plugin

- PMD is a static source code analyzer
- PMD is rule driven
  - Basic rules
  - Import rules
  - Unused code rules
  - Naming rules
  - Design rules
- The plugin creates an HTML report with linking to the Source Cross Reference
### PMD Plugin

#### Avoid empty catch blocks

<table>
<thead>
<tr>
<th>Junit\extensions\PPTestRunner.java</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avoid unused local variables such as 'result'</td>
<td>103</td>
</tr>
<tr>
<td>Avoid using 'if...else' statements without curly braces</td>
<td>137</td>
</tr>
<tr>
<td>Avoid using 'if...else' statements without curly braces</td>
<td>139</td>
</tr>
<tr>
<td>Avoid using 'if...else' statements without curly braces</td>
<td>141</td>
</tr>
<tr>
<td>Avoid using 'if...else' statements without curly braces</td>
<td>143</td>
</tr>
</tbody>
</table>

#### Avoid unused local variables such as 'result'

<table>
<thead>
<tr>
<th>ConfigurableTestCaseTest.java</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avoid unused local variables such as 'result'</td>
<td>112</td>
</tr>
</tbody>
</table>
JavaNCSS Plugin

- **JavaNCSS** is a source code metric analyzer
- Determines simple metrics
  - Number of packages
  - Number of classes
  - Number of functions
  - Number of non-commented lines of code
  - Number of JavaDoc lines
- Generates an XML report which is transformed to HTML using a XSL stylesheet
### JavaNCSS Plugin

#### Packages

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Classes</th>
<th>Functions</th>
<th>NCSS</th>
<th>Javadocs</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>7</td>
<td>3</td>
<td>javax.mail.internet</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>29</td>
<td>307</td>
<td>32</td>
<td>org.webdocwf.util.smime.activation</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>75</td>
<td>591</td>
<td>105</td>
<td>org.webdocwf.util.smime.cms</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>22</td>
<td>173</td>
<td>26</td>
<td>org.webdocwf.util.smime.crypto</td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>50</td>
<td>658</td>
<td>67</td>
<td>org.webdocwf.util.smime.der</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>15</td>
<td>318</td>
<td>18</td>
<td>org.webdocwf.util.smime.exception</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>2</td>
<td>66</td>
<td>3</td>
<td>org.webdocwf.util.smime.mail</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>59</td>
<td>857</td>
<td>62</td>
<td>org.webdocwf.util.smime.smime</td>
</tr>
<tr>
<td>9</td>
<td>13</td>
<td>17</td>
<td>931</td>
<td>18</td>
<td>org.webdocwf.util.smime.test</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>41</td>
<td>622</td>
<td>48</td>
<td>org.webdocwf.util.smime.util</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Classes</th>
<th>Functions</th>
<th>NCSS</th>
<th>Javadocs</th>
<th>(\text{per})</th>
</tr>
</thead>
<tbody>
<tr>
<td>10,00</td>
<td>82,00</td>
<td>312,00</td>
<td>4,530,00</td>
<td>382,00</td>
<td>Project</td>
</tr>
<tr>
<td>8,20</td>
<td>31,20</td>
<td>31,20</td>
<td>453,00</td>
<td>38,20</td>
<td>Package</td>
</tr>
<tr>
<td>3,80</td>
<td>55,24</td>
<td>4,66</td>
<td>14,52</td>
<td>14,52</td>
<td>Class</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Function</td>
</tr>
</tbody>
</table>
The Future

- Support of other version control systems for generating the activity report
  - Microsoft’s Visual Source Safe
  - Rational’s ClearCase
  - Perforce
- POM Inheritance Mechanism to simplify mavenizing multiple projects
- Quilt integration as Open Source alternative to Clover
- There is a Source Forge Maven Plugin-in project
Maven is an Open Source project build and comprehension tool developed by the JAKARTA community.

Maven is an integration platform for software engineering tools using a plug-in mechanism.

Maven provides a remote and local JAR repository to simplify upgrading of JARs.

Maven could be the answer to a lot of your software engineering problems.

And last but not least, Maven rocks …
Any questions !?
### Related Links

<table>
<thead>
<tr>
<th>Tool</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clover</td>
<td><a href="http://www.thecortex.net/clover/">http://www.thecortex.net/clover/</a></td>
</tr>
<tr>
<td>JDepend</td>
<td><a href="http://www.clarkware.com/software/JDepend.html">http://www.clarkware.com/software/JDepend.html</a></td>
</tr>
</tbody>
</table>
The plugins are implemented in Jelly
- Jelly is a XML scripting language similar to ANT
- Allows conditionals and loops
- Compatible to ANT scrips

A plugin exports goals which are invoked
- Directly by a script
- As pregoal or postgoal within Maven

A plugin can
- Start external application
- Fork a JVM to execute Java code
- Execute a Java code in-process