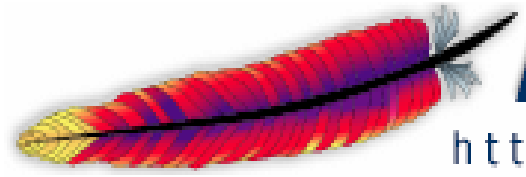


Apache Maven Project

http://maven.apache.org/

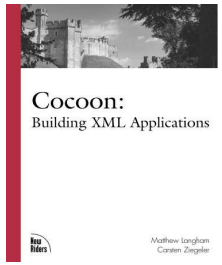
(A) Maven is Your Friend

Carsten Ziegeler
cziegeler@apache.org

Competence Center Open Source
S&N AG, Germany

About

- Member of the Apache Software Foundation
- Committer in some Apache Projects
  - Cocoon, Excalibur, Pluto, Incubator
- Chief Architect of the Competence Center Open Source, S&N AG, Germany
- Article and Book Author
- Technical Reviewer



A Maven is your friend?

- What does “maven“ mean?
  - “is not available in the general English dictionary“
  - “someone who is dazzlingly skilled in any field“
  - “an expert or connoisseur“
- So, am I a Maven maven?

- No, but...

The typical user!

- Neither (direct) Ant nor Maven committer
- Using both extensively
- For new projects
  - Use Maven if possible
- For old projects
  - Only convert if really **necessary**
- Currently TR for two books: Ant and Maven ☺

Agenda

- Motivation
- Basic Maven Concepts
- Using Maven - Build Management
- The Benefits - Project Comprehension
- Conclusion

This session is about Maven 1.x

Religious Wars are stupid and useless!

- Swing vs SWT
- Windows vs Linux vs MacOS X
- Eclipse vs “Some other IDE“
- Tabs vs Spaces
- Blue cars vs green cars
- **AND:** Maven vs Ant

Use the right tool for **your** work!

Use the Right Tool...

- To keep a manager/boss happy
  - Reduce costs
  - Reduce time to market
  - Frequent status reports
- To keep a developer happy
  - Concentrate on the „real“ work
  - Generate everything else (reports)

It all began...

- In the Jakarta Turbine Project
  - Each sub project with own build file/system (if any)
  - Jars somewhere in the project (CVS)

- Motivation
  - Use the same build process for all projects
  - Share jars between projects
    - Dependency management

What is Maven?

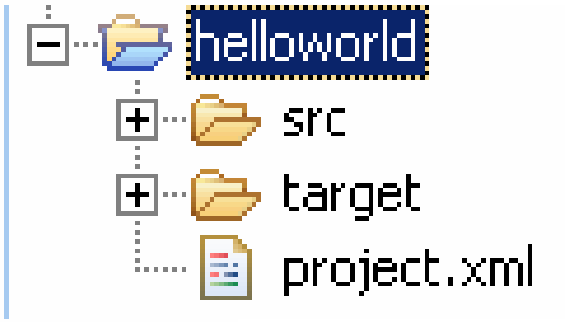
- Apache Top Level Project (maven.apache.org)
- Software Project Management and Comprehension Tool (for Java)
  - Build
  - Documentation
  - Reporting
  - Deployment
- Based on a single source of information:
  - The Project Object Model (POM)

Example: Hello Maven!

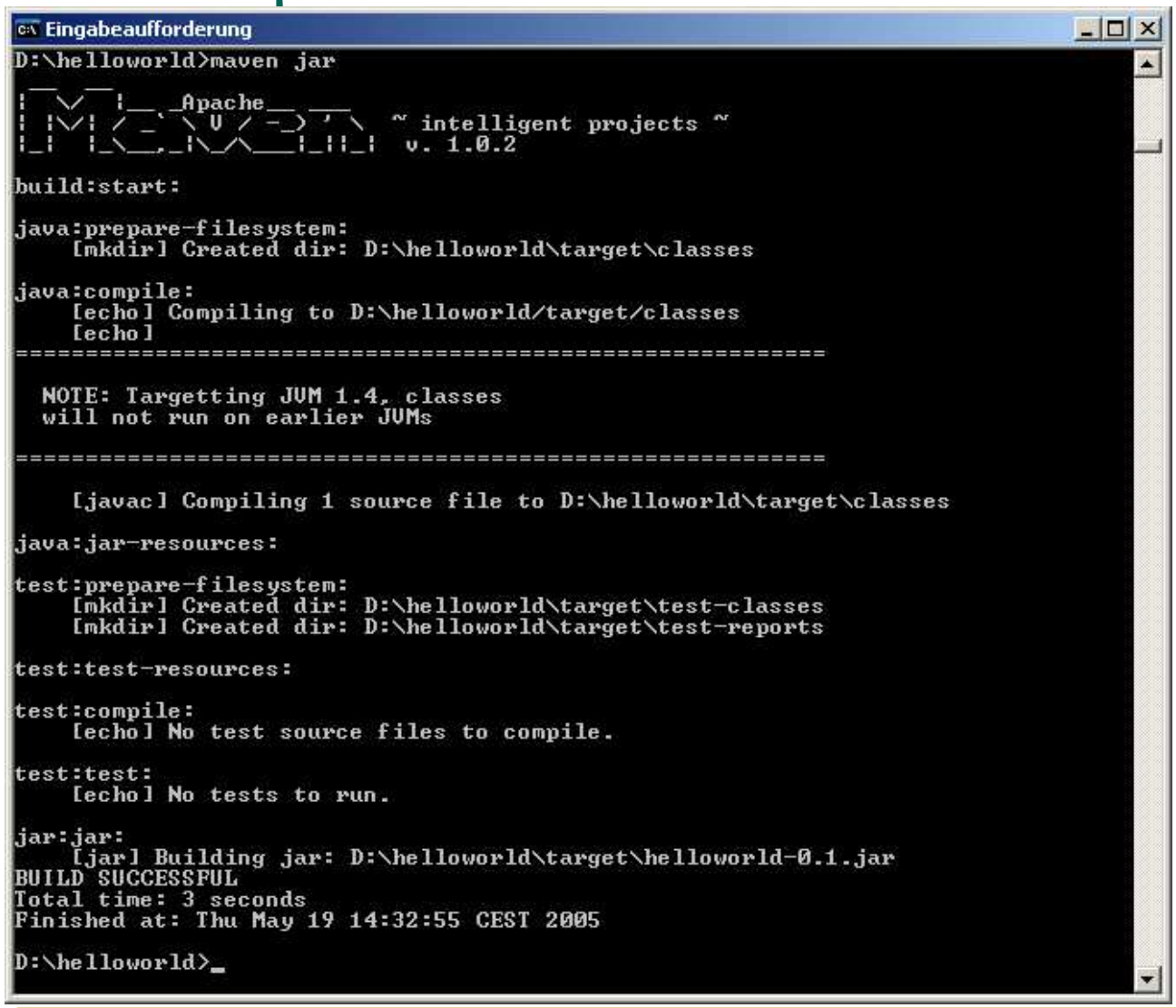
- POM:

```
<project>
  <pomVersion>3</pomVersion>
  <groupId>ApacheCon</groupId>
  <artifactId>helloworld</artifactId>
  <name>Hello World</name>
  <currentVersion>0.1</currentVersion>
  <build>
      <sourceDirectory>src/java</sourceDirectory>
  </build>
</project>
```

- Directory Layout:



Example: Hello Maven!



- Execute:
  - > maven jar
  - Compiles
  - Runs tests
  - Creates jar

Simple Comparison

- Maven
  - Description of the project
  - Invocation of defined goals (targets)
  - Project Knowledge
- Ant
  - Development of a build script per project
  - Invocation of project specific targets
  - Just the build process

Typical Problems I

- Setting up a build process is time/cost consuming
  - Each project might have a different one
- New team members have to learn the build process
  - Or they start to use their own mechanisms…
- How do I get a running version?
  - For development?
  - For testing?
  - For production?

Typical Problems II

- Dependencies
  - Which?
  - And what version?
  - Change during project lifetime
- Documentation
  - Noone writes it ☺
- How can I run my tests?
- How can I track the project status?

What is Maven and what it‘s not!

- Objectives
  - Making the build process easy
  - Providing a uniform build system
  - Providing quality project Information
  - Providing guidelines for best practice development
- Maven is not
  - The “next generation Ant“

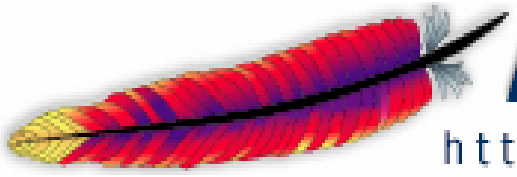
Terminology and Architecture

- Maven consists of a core and several *plugins*
- A *plugin* provides a set of functionality
  - Uses the POM
  - Provides *goals* (= targets)
- A project produces **one** *artifact* (jar, war etc.)
- Invocation:
  - `> maven plugin-name:goal`
  - `> maven jar:jar (or just jar=default)`

Available Plugins

- About
- Announcement
- Ant
- Antlr
- Appserver
- Artifact
- Ashkelon
- AspectJ
- AspectWerkz
- Caller
- Castor
- Changelog
- Changes
- Checkstyle
- Clean
- Clover
- Console
- Cruise Control
- Dashboard
- Developer Activity
- Distribution
- DocBook
- EAR
- Eclipse
- EJB
- FAQ
- File Activity
- Genapp
- Gump
- Hibernate
- Html2XDoc
- Idea
- J2EE
- Jalopy
- Jar
- Java
- Javacc
- Javadoc
- JBoss
- JBuilder
- JCoverage
- JDEE
- JDepend
- JDeveloper
- JDiff
- JellyDoc
- Jetty
- JIRA
- JNLP
- JUnit Report
- JUnitDoclet
- JXR
- Latex
- Latka
- License
- LinkCheck
- MultiChanges
- Multi-Project
- Native
- NSIS
- PDF
- Plugin
- PMD
- POM
- RAR
- Release
- Repository
- SCM
- Shell
- Simian
- Site
- Struts
- Tasklist
- Test
- TJDO
- Touchstone
- Uberjar
- VDoclet
- WAR
- Webserver
- Wizard
- XDoc

And Many more ...

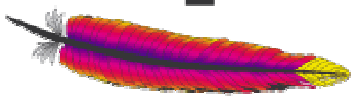


Apache Maven Project

http://maven.apache.org/

(A) Maven is your friend

The Project Object Model



ApacheCon Europe 05

Project Object Model (POM)

- Project Meta Data
  - General
    - Name, version, description
    - Owner, Developers, Contributers
    - SCMS, Issue Tracking, mailing lists, URLs
  - Dependencies
  - Directory Layout
    - Source, Tests, Configuration, Documentation
  - Build
    - Reports, properties

POM – Project Information

```
<project>
  <pomVersion>3</pomVersion>
  <name>Pluto Descriptors Subproject</name>
  <artifactId>descriptors</artifactId>
  <groupId>pluto</groupId>
  <currentVersion>1.0.1-rc3</currentVersion>
  <organization>
    <name>Apache Software Foundation</name>
    <url>http://portals.apache.org/</url>
    <logo>/images/apache-portals.gif</logo>
  </organization>
  <logo>/images/pluto.png</logo>
  <inceptionYear>2003</inceptionYear>
```

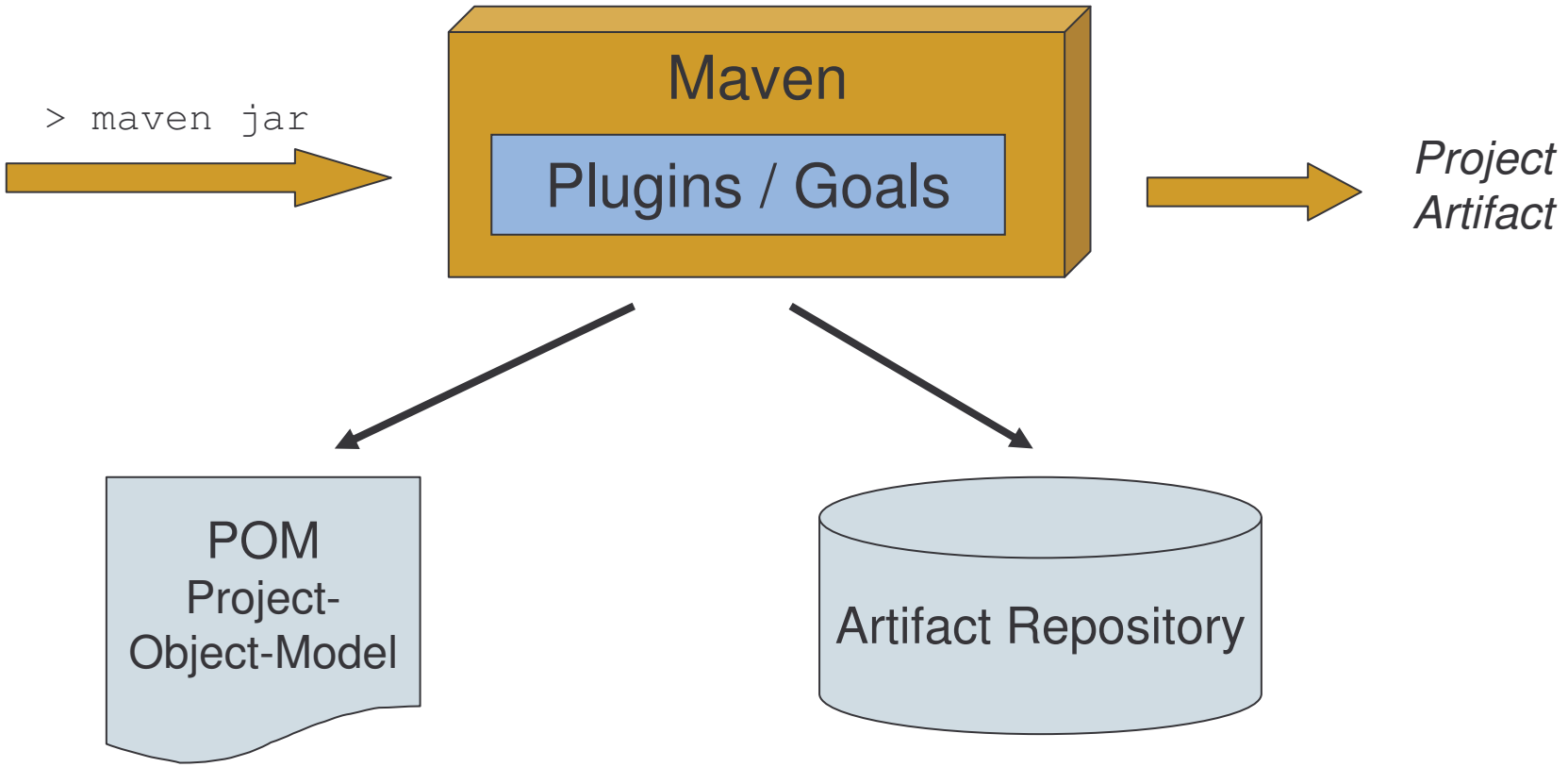
POM - Build

```
<build>
  <sourceDirectory>src/java</sourceDirectory>
  <unitTestSourceDirectory>src/test</unitTest
  <resources>
    <includes>
      <include>conf/**/*pipeline.xml</include
      <include>conf/**/*valve.xml</include>
    </includes>
  </resources>
</build>
```

POM - Dependencies

```
<dependencies>
  <dependency>
    <groupId>castor</groupId>
    <artifactId>castor</artifactId>
    <version>0.9.5.3</version>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
  </dependency>
  <dependency>
    <groupId>commons-logging</groupId>
    <artifactId>commons-logging</artifactId>
    <version>1.0.4</version>
  </dependency>
```

Building

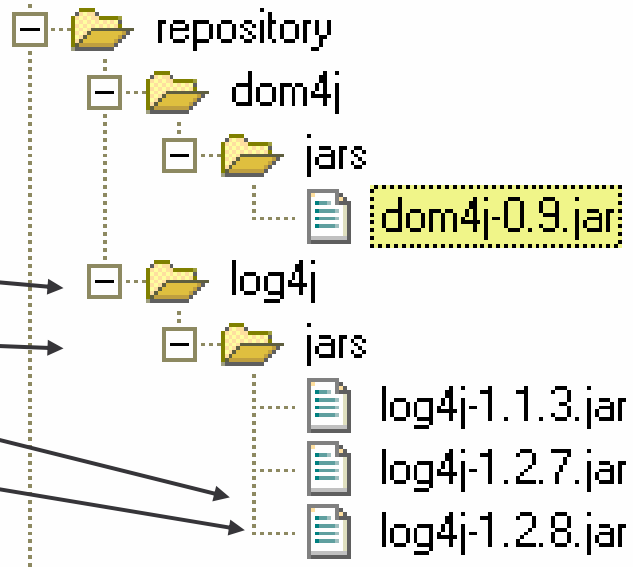

> maven jar
Maven
Plugins / Goals
Project
Artifact
POM
Project-
Object-Model
Artifact Repository

Artifact Repository

- Central repository for all artifacts
  - Includes own artifacts
  - Artifacts (e.g. jars) are not in the SCMS anymore
- Defined directory layout
- Versioning through naming conventions
- **All** dependencies are resolved through a repository

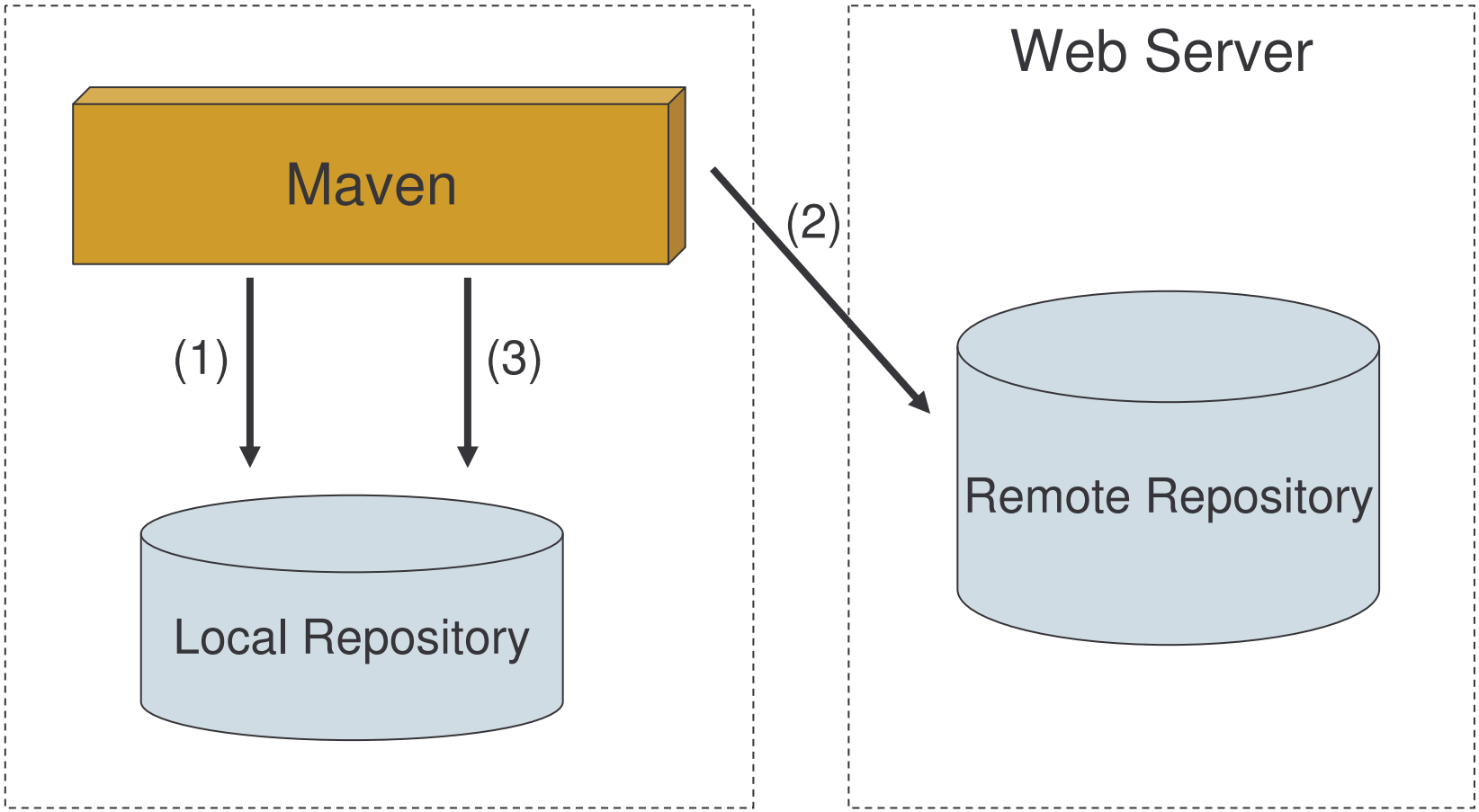
Artifact Repository Reference

```
<project>
   ...
   <dependencies>
     <dependency>
       <groupId>log4j</groupId>
       <type>jar</type>
       <artifactId>
           log4j
       </artifactId>
       <version>1.2.8</version>
     </dependency>
   ...
```



=> %MAVEN_REPO% / *[groupId]* / *[type]*s / *[artifactId]*-*[version]*.*[type]*

Artifact Repository



Artifact Repository

- Local Repository (per user)
  - all used external artifacts (cache)
  - Installed artifacts from build projects
  - (Use property to specifiy location)
- Available remote repositories
  - Ibiblio, Apache (over 8000 artifacts available)
- Install company repository
  - Exchange versions of company artifacts
  - Use Maven proxy

Only One Artifact Per Project!

- Projects with several artifacts (source directories) are common
  - Subprojects, API vs Implementation
- Maven supports only one artifact (source directory) per project
- Use separate projects
- Or: Maven Multi Project (Plugin)
- During build: artifacts are installed in local repository

Using Maven

- First steps with Maven are easy
  - Provide a POM and use available plugins
  - No need to write own build scripts
  - Requires a repository
- Plugins
  - Might use other plugins
  - Are used to extend Maven functionality
  - Can be added/removed

Five Maven Features you must love!

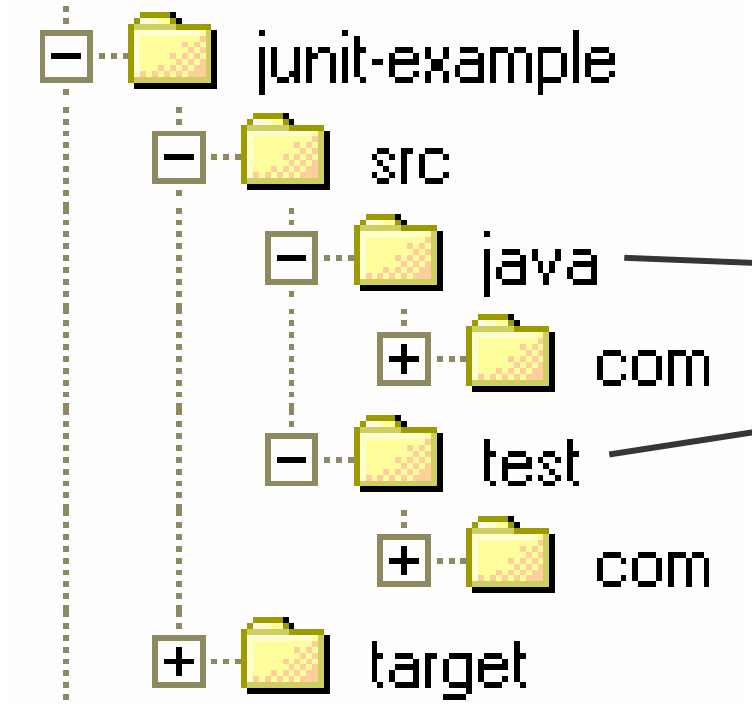
- Automatic downloading of dependencies
- A consistent and standarized directory layout
- A consistent naming of goals (targets): war, jar, javadoc,...
- AOP like chain of goals
- A declarative descriptor of dependencies and project settings

From Carlos Sanchez's Weblog

JUnit Tests

- Maven Test Plug-In
  - Compile JUnit Tests
  - Execute Tests
  - Generate Reports

JUnit Tests / Project Layout




```
<project>
  <pomVersion>3</pomVersion>
  <id>maven-junit-example</id>
- <build>
    <sourceDirectory> src/java
    </sourceDirectory>
    <unitTestSourceDirectory> src/test
    </unitTestSourceDirectory>
  </build>
</project>
```

JUnit Tests / POM Extensions

```
<build>
  <sourceDirectory>src/java</sourceDirectory>
  <unitTestSourceDirectory>src/test</unitTestSourceDirectory>
 - <unitTest>
   - <includes>
       <include>**/*Test*</include>
       <exclude>**/*ExtendedTest*</exclude>
       <resources/>
     </includes>
  </unitTest>
</build>
```

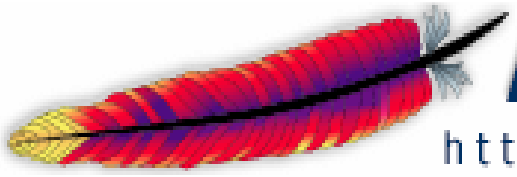
- Specify tests by patterns
- Define resources for the tests

JUnit Tests/Goals

- Compile
  - `maven test:compile`
- Execute All
  - `maven test:test`
- Other plugins use the test plugin
  - E.g. jar
- Single Test Case
  - `maven -Dtestmatch=*Test test:match`
  - `maven -Dtestcase=com.junit.Test test:single`

JUnit Tests / Reports

```
Testsuite: com.junitexample.ShoppingCartTest
Tests run: 4, Failures: 0, Errors: 0, Time
   elapsed: 0,057 sec
```

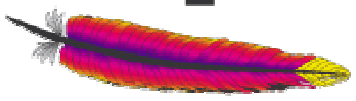


Apache Maven Project

http://maven.apache.org/

(A) Maven is your friend

Project Comprehension



ApacheCon Europe 05

Project Comprehension

- Project web site with
  - Project information
  - Own Documentation
  - Reports
    - Show the current state of the project
    - Source code checks/metrics (Checkstyle, PMD)
    - SCMS Reports
      - Last commits
      - Developer and file activity

Website

- Maven Site Plugin
  - Complete HTML website
  - Generate docs using the POM
  - Add own docs in XML
  - Reports are added by plugins
  - Navigation/Layout etc. can be adapted
  - `> maven site`

POM – Infrastructure I

```
<description>
Pluto is the Reference Implementation of the Java...
</description>
<url>http://portals.apache.org/pluto/</url>
<issueTrackingUrl>http://issues.apache.org/jira/secu
<siteAddress>portals.apache.org</siteAddress>
<siteDirectory>/www/portals.apache.org/pluto/</siteDi
<repository>
  <connection>scm:svn:http://svn.apache.org/repos/asf
  <developerConnection>scm:svn:https://svn.apache.org
  <url>http://svn.apache.org/repos/asf/portals/pluto/
</repository>
```

POM – Infrastructure II

```
<mailingLists>
  <mailingList>
    <name>Pluto User List</name>
    <subscribe>pluto-user-subscribe@portals.apache.org</s
    <unsubscribe>pluto-user-unsubscribe@portals.apache.or
    <archive>http://issues.apache.org/...</archive>
  </mailingList>
  <mailingList>
    <name>Pluto Developer List</name>
    <subscribe>pluto-dev-subscribe@portals.apache.org</su
    <unsubscribe>pluto-dev-unsubscribe@portals.apache.org
    <archive>http://issues.apache.org/...</archive>
  </mailingList>
```

POM - People

```
<developers>
  <developer>
    <name>Santiago Gala</name>
    <id/>
    <email>sgala{aT}hisitech{d0t}com</email>
    <organization/>
  </developer>
  ...
<contributors>
  <contributor>
    <name>Michael Westbay</name>
    <email>westbay{aT}users{d0T}sourceforge{d0T}net</email>
  </contributor>
  ...
```

Project Website

Documentation and Reports

- Reports for free!
  - Change Logs, Developer Activity...
  - Test Results
  - Metrics (incl. references to Java code)
  - Reports are Plugins!
- Read the reports!
- (Reports use locale of the system they‘re running on!)

POM - Reports

```
<reports>
  <report>maven-license-plugin</report>
  <report>maven-checkstyle-plugin</report>
  <report>maven-changes-plugin</report>
  <report>maven-changelog-plugin</report>
  <report>maven-developer-activity-plugin</report>
  <report>maven-file-activity-plugin</report>
  <report>maven-javadoc-plugin</report>
  <report>maven-jxr-plugin</report>
  <report>maven-tasklist-plugin</report>
</reports>
```

Reports: JUnit Tests

Summary

[summary] [package list] [test cases]

Tests	Errors	Failures	Success rate	Time(s)
4	0	0	100,00%	0,09

Note: *failures* are anticipated and checked for with assertions while *errors* are unanticipated.

Package List

[summary] [package list] [test cases]

Package	Tests	Errors	Failures	Success Rate	Time
com.junitexample	4	0	0	100,00%	0,09

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

com.junitexample

Class	Tests	Errors	Failures	Success Rate	Time
ShoppingCartTest	4	0	0	100,00%	0,092

Test Cases

[summary] [package list] [test cases]

ShoppingCartTest

testProductAdd	0,01
testEmpty	0,00
testProductRemove	0,00
testProductNotFound	0,00

Reports: CheckStyle

- Different Java Coding Styles
  - Sun
  - Jakarta
  - Own (if you like...)
- Rules are configurable (XML)

Reports: CheckStyle

org/apache/maven/MavenUtils.java

Error	Line
Utility classes should not have a public or default constructor.	79
Line has trailing spaces.	715
';' is followed by whitespace.	880

org/apache/maven/UnknownGoalException.java

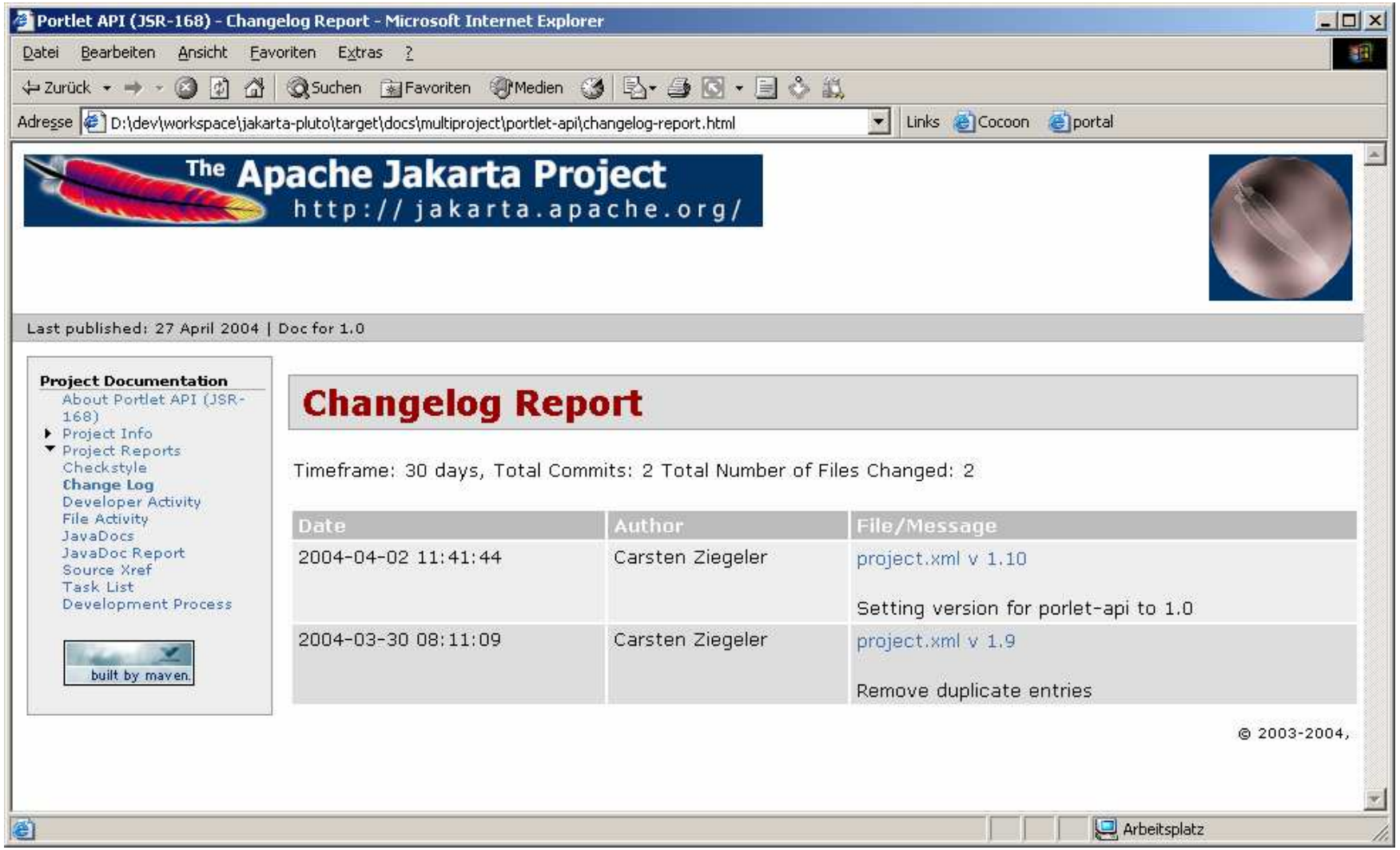
Error	Line
'goalName' hides a field.	35

org/apache/maven/cli/App.java

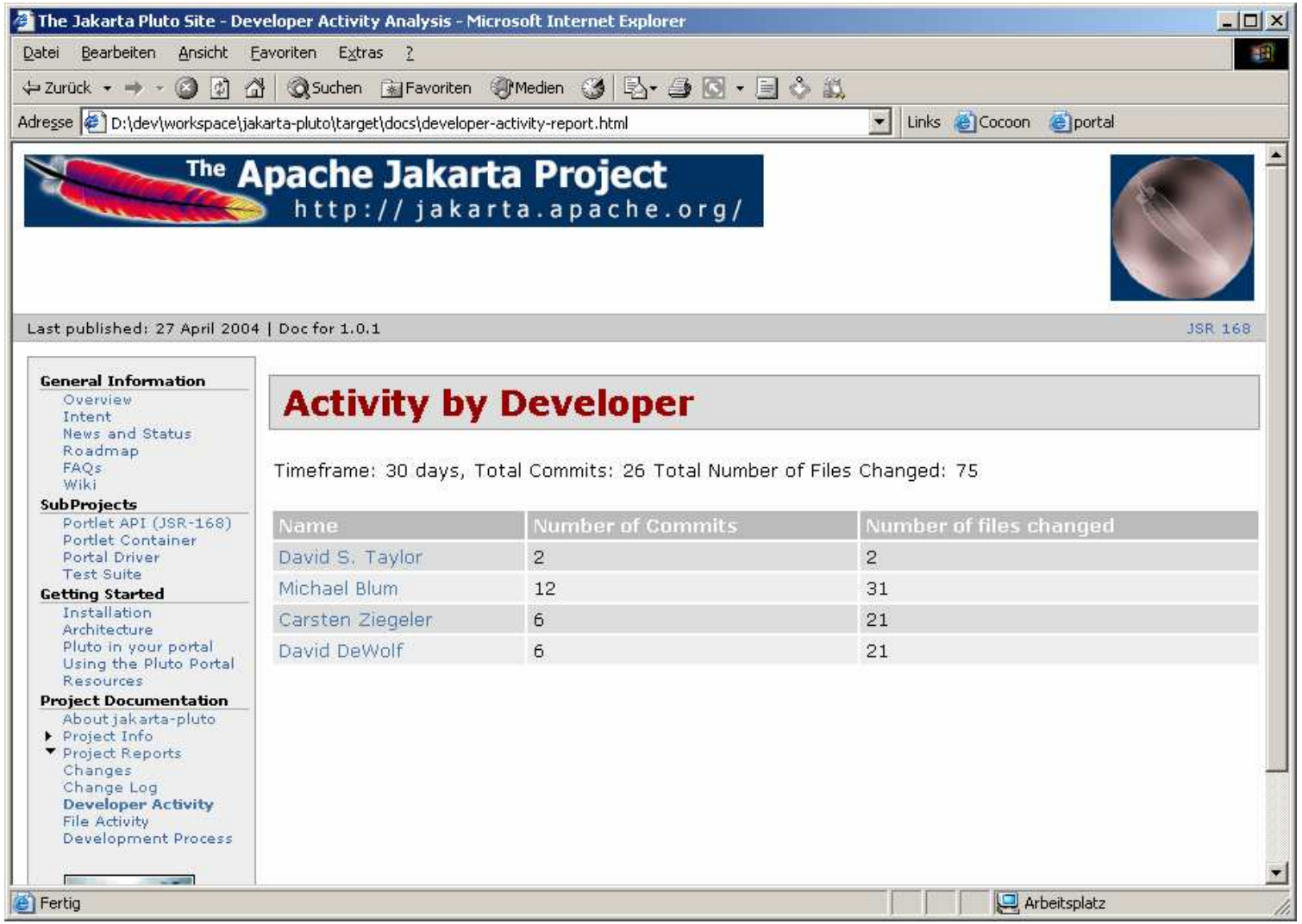
Reports: Activity

- Plugins directly use the SCMS (CVS, SVN)
  - Changes and Commits
  - Number of changes per developer
  - Number of changes per file

Reports: Changelog



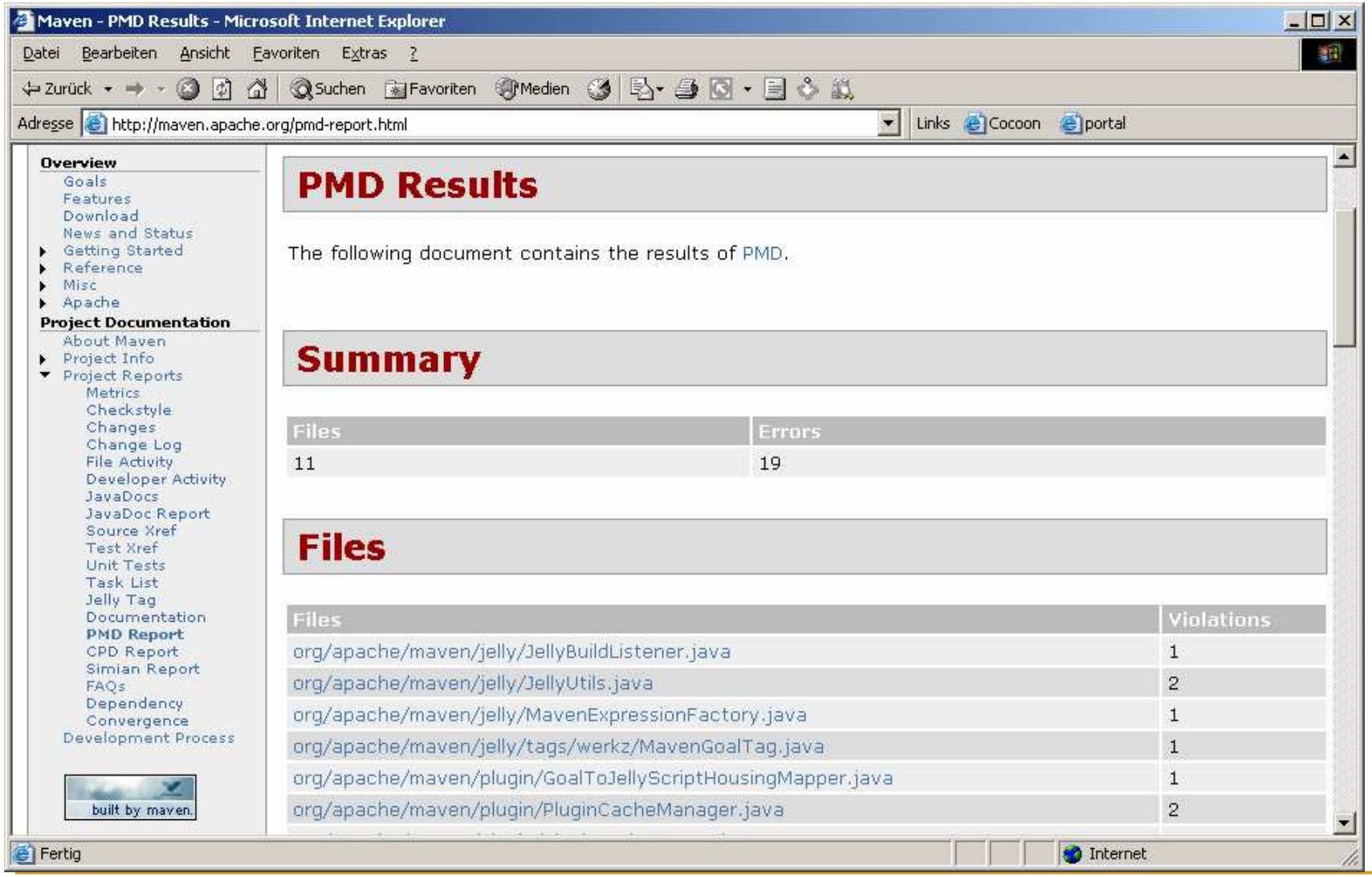
Reports: Developer Activity



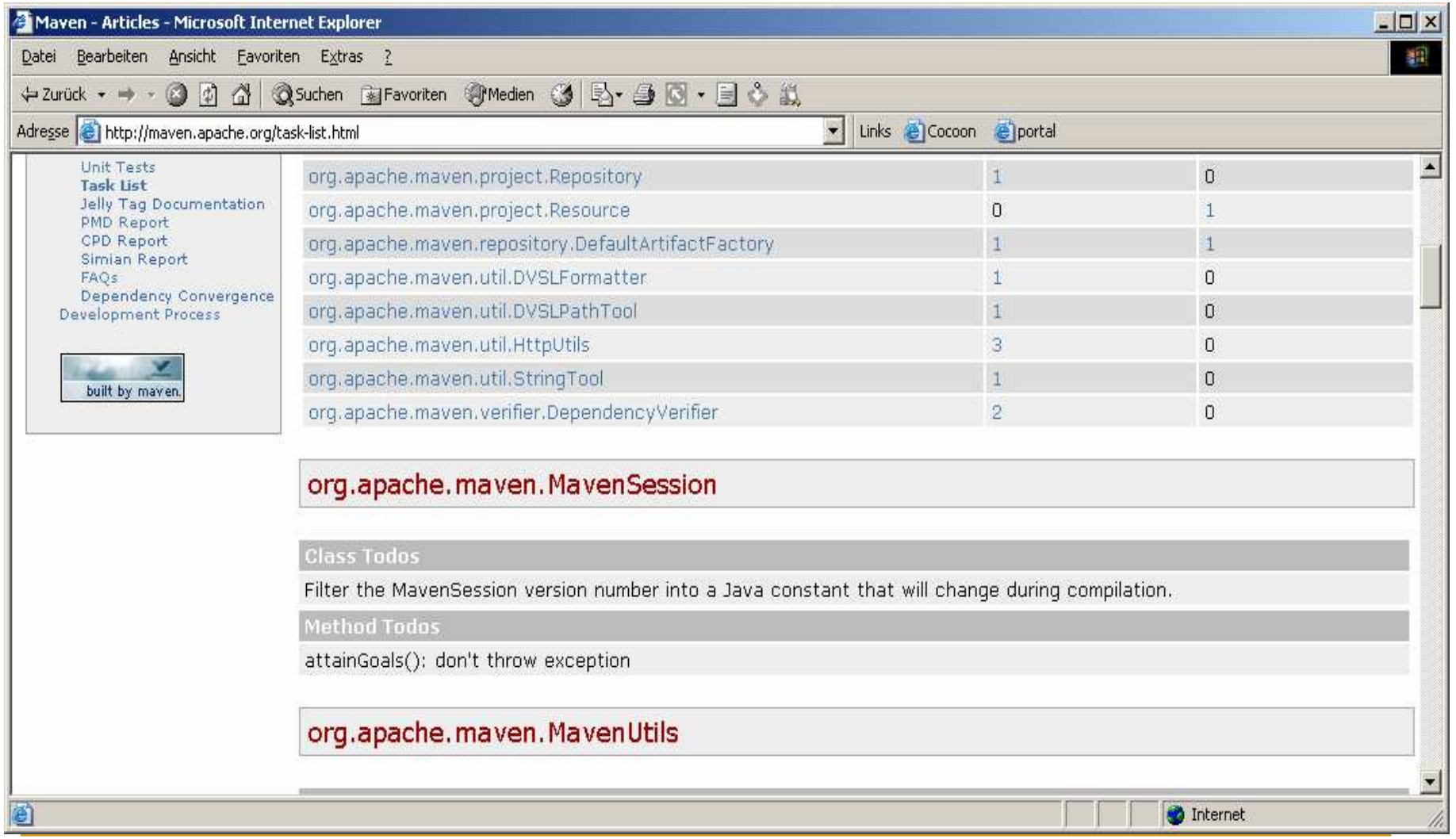
Further Reports

- PMD – Source Code Analyzer
  - Rule based
  - Check imports
  - Unused Code
  - Naming Conventions
  - …
- Tasklist Plugin
  - Analyzes Java Code
- And some more ☺

Reports: PMD

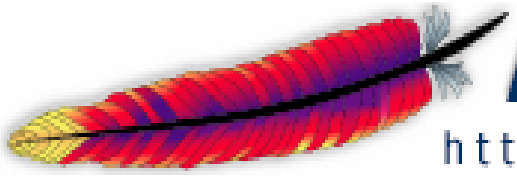


Reports: Tasklist



Project Comprehension

- Project Comprehension comes with zero costs
  - `> maven site`
  - Information from POM
  - Provided reports
- Simply *use* the report results
- Use together with Continuous Integration
  - CruiseControl
  - Maven Continuum (alpha)

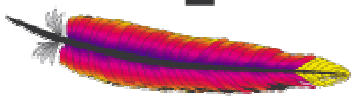


Apache Maven Project

http://maven.apache.org/

(A) Maven is your friend

Configuration / Adapting



ApacheCon Europe 05

Configuration / Adapting

- Configuration of plugins through properties
  - Will change in 2.0
- Plugins can be extended per project through „maven.xml“
  - Will not be supported starting with 2.0
- Write your own plugins

Properties

- Each plugin has a defined set of properties
- Location of properties:
  1. [project.home] / project.properties
  2. [project.home] / build.properties
  3. [user.home] / build.properties
  4. system properties

Last definition wins!

Adaption through maven.xml

- Add own goals per project
- Adapt existing goals (preGoal, postGoal)

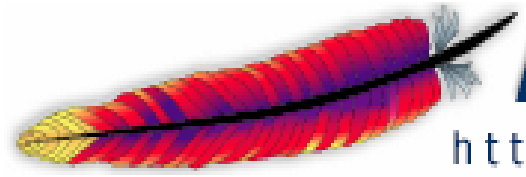
```
<project>
 - <postGoal name="plugin:deploy">
     <ant:echo message="deleting cache files"/>
   - <ant:delete>
       <ant:fileset dir="${maven.plugin.unpacked.dir}" includes="*.cache"/>
     </ant:delete>
   </postGoal>
</project>
```

Write your own plugin

- Plugins are Maven projects, too
- Use the “plugin“ plugin
- Requires (1.x):
  - `project.xml`
  - `plugin.jelly`
  - `plugin.properties`
- Will change with 2.0

IDE Integration

- Plugins for IDEs
  - Create/Setup IDE specific project files
- Eclipse plugin
  - Generate classpath and project file
    - `> maven eclipse`
  - Add classpath variable for Maven repository
    - `> maven eclipse:add-maven-repo`
- Mevenide (Eclipse Plugin)

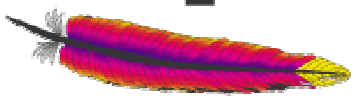


Apache Maven Project

http://maven.apache.org/

(A) Maven is your friend

Multi Project



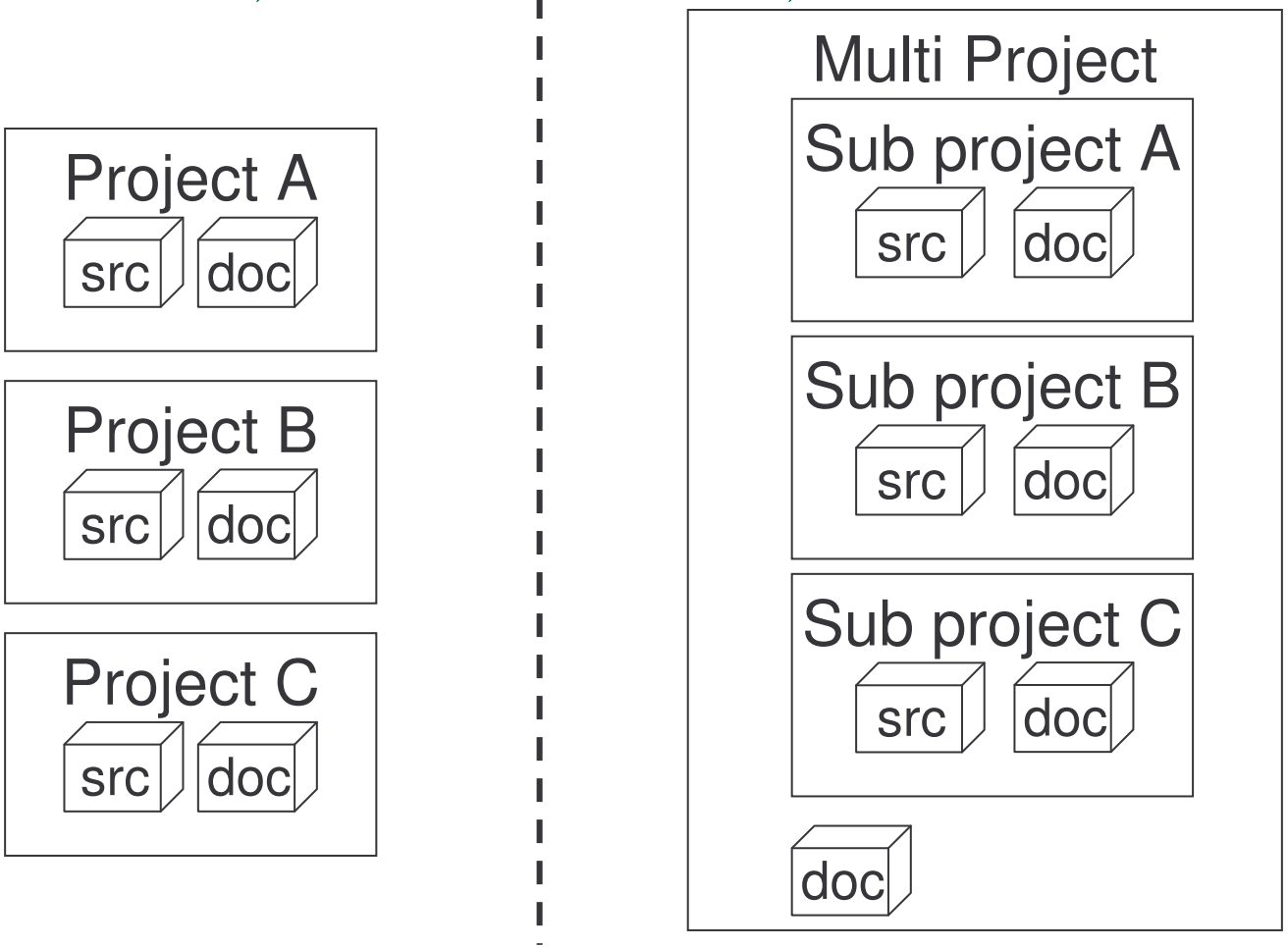
ApacheCon Europe 05

Multi Project - Motivation

- Projects with several artifacts (source directories)
  - Subprojects, API vs Implementation
- Maven supports only one artifact (source directory) per project
- Use separate projects
- Or: Maven Multi Project (Plugin)

Multi Project

Several Projects vs. Multi Project



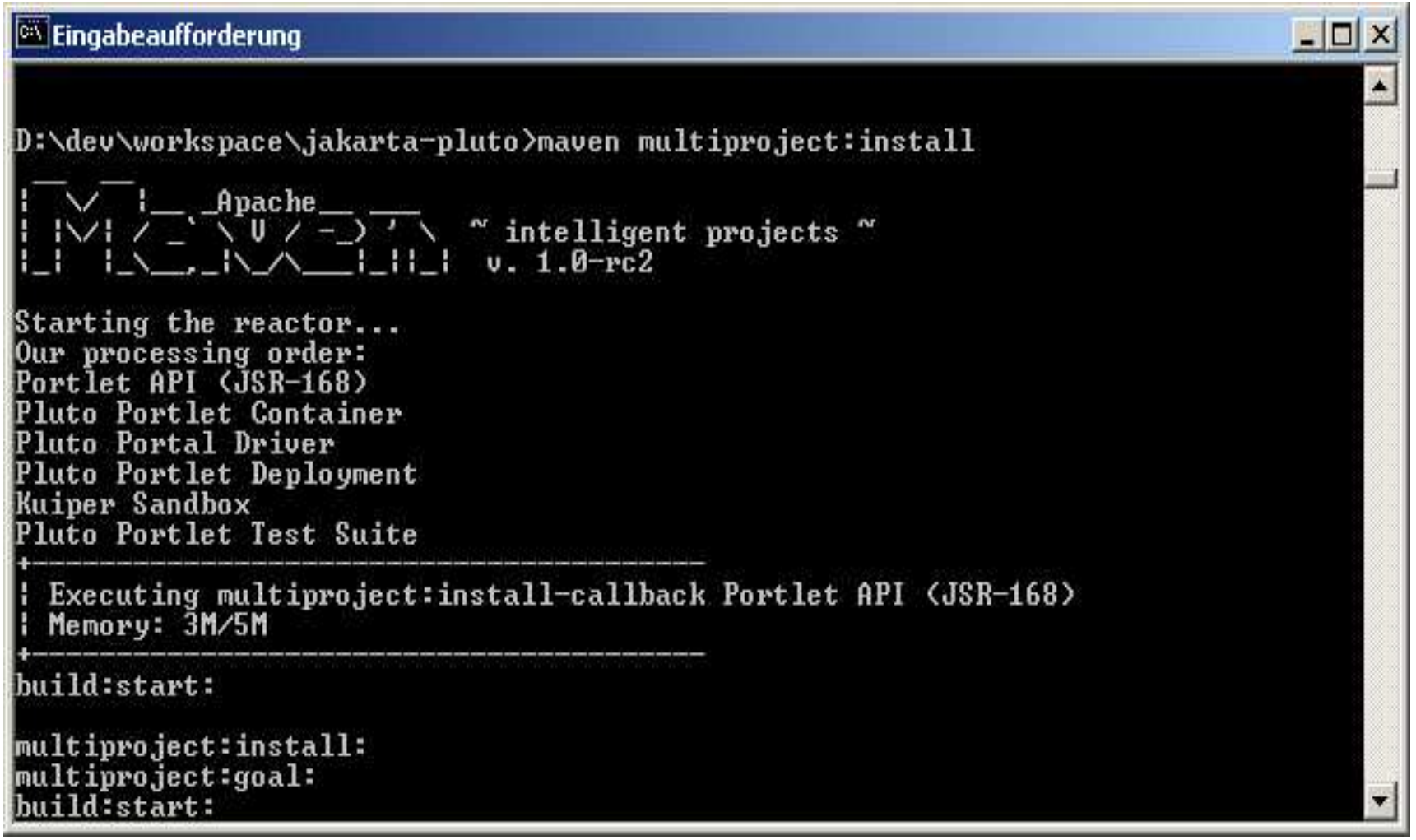
Multi Project – Project Layout

- Project envelope (= the multi project)
- Contains several subprojects
- Subprojects inherit POM
  - For extension/adaption
- Maven recognizes dependencies between subprojects
  - Build order

Multi Project – Apache Pluto

- Apache Pluto (http://portals.apache.org/pluto)
  - Portlet API (JSR 168)
  - Portlet Container
  - Demo Portal
  - Testsuite
  - Portlet Deployment Tool

Multi Project – Apache Pluto



Multi Project – Apache Pluto

- Before: Complicated 80 KB Ant Script + Batch files
  - Neither transparent nor maintainable
  - Did not run everywhere
  - Jars were in CVS, some more than once
- Now: Maven Multi Project
  - Manageable
  - Runs everywhere
  - Jars only once in repository
- Additional Advantages
  - More project information is in POM
  - Reports, Documentation etc.

Multi Project – Apache Pluto

- Multi Project POM
  - Projectinfo
  - Version
  - Developers
  - Resources
- Sub project POMs
  - Dependencies
  - Source directory, Test directory

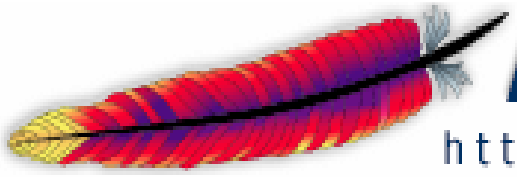
Multi Project - Problems

- Multi Project does not support all Goals
  - Own Plugin
- All dependencies are resolved by the repository
  - Install/Deploy instead of jar...
  - Own goals/plugins for specific tasks
    - E.g. releases/distributions
    - But this will get better!
- Sub projects are independent
  - Own docs

Multi Project – Website

Multi Project

- Multi Project
  - Separate a project into several parts
  - Is treated as one unit
  - But reduced Maven functionality
- Check what is best for you!
  - Multi project vs. several projects
  - Ant?
- Will be improved!

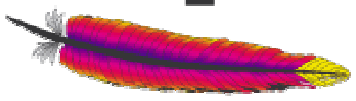


Apache Maven Project

http://maven.apache.org/

(A) Maven is your friend

Conclusion



ApacheCon Europe 05

Conclusion I

- Maven uses a project description
  - No need to write own build files
- Artifact repository (independent of projects)
- Support for documentation and reports
- Configurable and extendable
- Teaser:

“Maven can do in 0 lines what ANT takes 50 lines for a simple project.“

Conclusion II

- Maven is **not** suited for every task
- Believe it: it is **NOT** the „golden hammer“
- And it is not an IDE !
  - But can be integrated
- But it can make **YOUR** life easier

Conclusion III

- Maven is not Ant
  - Use the provided functionality/approach
  - Try to avoid own goals (maven.xml)
  - If something is missing: own plugin
  - If you have special needs: **use Ant**!
  - Maven can be another "Start fast fail later" trap

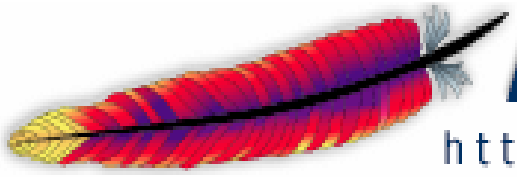
Maven Summary

- + Rich function set through plugins
- + Reusability between projects
- + Good chance to standardize your project work

- - Only one source directory per project
- - Only one artifact per project
- - Finding the right plugin is not that easy
- - Not as flexible as Ant

The Future

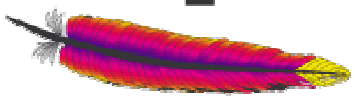
- Maven 2.0 is comming soon(?)
  - Provides long awaited features
  - Multi project support
  - Even simpler
  - Faster
  - Better architecture
  - Transitive dependencies and scope



Apache Maven Project

http://maven.apache.org/

Thanks for your attention!



ApacheCon Europe 05