**Maven**

Auf Java basierendes Buildmanagement Tool. Zur Konfiguration wird die pom.xml verwendet.

Aufgaben: - Verwaltung von Abhängigkeiten (Libraries)

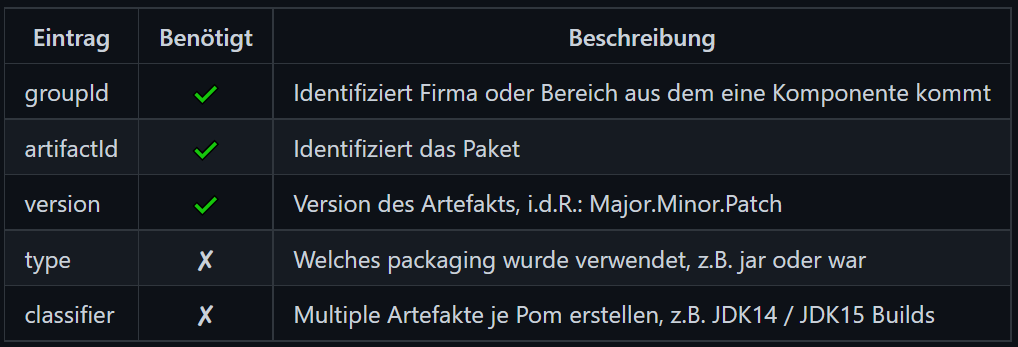
- Pflege von Metadaten (Lizenz Informationen, Entwickler, Webseite...)

- Versionierung

- Konfiguration des Build Prozesses

Koordinaten

Die eigenen Koordinaten werden wie folgt in der pom.xml innerhalb des project Knotens definiert.



Abhängigkeiten

Java Anwendungen haben eine Vielzahl von Abhängigkeiten. Diese werden in der dependency Sektion angegeben.



Default Lifecycle

Ein Bild, das Text, Screenshot, schwarz, Bildschirm enthält.

Automatisch generierte Beschreibung

Goals

Einzelne Plugins können Funktionen exponieren. Diese nennt man Goals. Goals können an Phasen gebunden werden, aber auch einzeln ausgeführt werden. Darüber können sich Maven Plugins am Lifecycle beteiligen.

Archetypes

Templates mit denen man Projekte ausliefern kann.

Maven Plugins

Mit Plugins kann Maven Funktionalität erweitert werden, wie z.B.:

Validierung Code Style (Checkstyle)

Security Scans (Common Vulnerabilities and Exposures - CVE)

Code Qualität / Static Code Analysis (PMD, SpotBugs, SonarQube)

Testing (Surefire)

Reporting (JavaDocs, Test Coverage, ..)

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Java Docs

Kommentarblöcke die mit "\*\*" beginnen, werden als JavaDocs interpretiert. Diese können an Packages, Klassen, Interfaces, Methoden, Feldern und Variablen genutzt werden.

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

Checkstyle/Codestyle

Checkstyle (maven) überprüft das Format des Codings. Regeln können durch eine xml Datei angepasst werden

Code style (IDE) formatiert den Code.

Unit Testing – JUnit

Tests für möglichst kleine funktionelle Einheiten

Test wird auch Unit Test genannt

Test in sich geschlossen.

Wird in der Test Phase ausgeführt Von Entwicklern, für Entwickler

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

**Arbeitsweisen**

Test-Driven-Deployment

1. Test wird entwickelt, schlägt aber fehl.

2. Programm wird minimal geändert, sodass Test erfolgreich ist. Copy Paste, oder ähnliches ist erlaubt und gewünscht. Der Aufwand muss minmal sein.

3. Refaktorierung sodass Qualität, Struktur und Architektur gut ist.

Pair-Programming

2 Entwickler, 1 Computer

Navigator: sagt was gemacht wird

Driver: bedient Computer

Rollen werden regelmäßig getauscht

Besonders geeignet bei unterschiedliche Skill Level wenn Software Qualität besonders hoch sein muss

Mob-Programming

3-5 Entwickler, 1 Computer

Driver: bedient Computer

Es muss Einigkeit herrschen was getan werden soll

Geteiltes Wissen im Team (Code Ownership)

Keine Handys oder ähnliche Ablenkungen

Zeitlich beschränkt, mit regelmäßigen Pausen (ca. alle 45 min)