**Maven\_Notes**

**What is Maven?**

Maven is a project management and comprehension tool that provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

In case of multiple development teams environment, Maven can set-up the way to work as per standards in a very short time. As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

Maven provides developers ways to manage the following:

 Builds

 Documentation

 Reporting

 Dependencies

 SCMs

 Releases

 Distribution

 Mailing list

To summarize, Maven simplifies and standardizes the project build process. It handles compilation, distribution, documentation, team collaboration and other tasks seamlessly. Maven increases reusability and takes care of most of the build related tasks.

**Maven Evolution**

Maven was originally designed to simplify building processes in Jakarta Turbine project. There were several projects and each project contained slightly different ANT build files. JARs were checked into CVS.

Apache group then developed **Maven** which can build multiple projects together, publish projects information, deploy projects, share JARs across several projects and help in collaboration of teams.

**Objective**

The primary goal of Maven is to provide developer with the following:

 A comprehensive model for projects, which is reusable, maintainable, and easier to comprehend.

 Plugins or tools that interact with this declarative model.

Maven project structure and contents are declared in an xml file, pom.xml, referred as Project Object Model (POM), which is the fundamental unit of the entire Maven system. In later chapters, we will explain POM in detail.

**Convention over Configuration**

Maven uses **Convention** over **Configuration**, which means developers are not required to create build process themselves.

Developers do not have to mention each and every configuration detail. Maven provides sensible default behavior for projects. When a Maven project is created, Maven creates default project structure. Developer is only required to place files accordingly and he/she need not to define any configuration in pom.xml.

|  |  |
| --- | --- |
| As an example, following table shows the default values for project source code files, resource files and other configurations. Assuming, **${basedir}** denotes the project location: **Item** | **Default** |
| source code | ${basedir}/src/main/java |
| Resources | ${basedir}/src/main/resources |
| Tests | ${basedir}/src/test |
| Complied byte code | ${basedir}/target |
| distributable JAR | ${basedir}/target/classes |

**Features of Maven**

 Simple project setup that follows best practices.

 Consistent usage across all projects.

 Dependency management including automatic updating.

 A large and growing repository of libraries.

 Extensible, with the ability to easily write plugins in java or scripting languages.

 Instant access to new features with little or no extra configuration.

 **Model-based builds**: Maven is able to build any number of projects into predefined output types such as jar, war, metadata.

 **Coherent site of project information**: Using the same metadata as per the build process, maven is able to generate a website and a PDF including complete documentation.

 **Release management and distribution publication**: Without additional configuration, maven will integrate with your source control system such as CVS and manages the release of a project.

 **Backward Compatibility**: You can easily port the multiple modules of a project into Maven 3 from older versions of Maven. It can support the older versions also.

 **Automatic parent versioning**: No need to specify the parent in the sub module for maintenance.

 **Parallel builds**: It analyzes the project dependency graph and enables you to build schedule modules in parallel. Using this, you can achieve the performance improvements of 20-50%.

 **Better Error and Integrity Reporting**: Maven improved error reporting, and it provides you with a link to the Maven wiki page where you will get full description of the error.

Maven is a Java based tool, so the very first requirement is to have JDK installed on your machine.

**System Requirement**

|  |  |
| --- | --- |
| **JDK** | 1.7 or above. |
| **Memory** | No minimum requirement. |
| **Disk Space** | No minimum requirement. |
| **Operating System** | No minimum requirement. |

**POM**

POM stands for Project Object Model. It is fundamental unit of work in Maven. It is an XML file that resides in the base directory of the project as pom.xml.

The POM contains information about the project and various configuration detail used by Maven to build the project(s).

POM also contains the goals and plugins. While executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, and then executes the goal. Some of the configuration that can be specified in the POM are following:

 project dependencies

 plugins

 goals

 build profiles

 project version

 developers

 mailing list

Before creating a POM, we should first decide the project **group** (groupId), its **name** (artifactId) and its version as these attributes help in uniquely identifying the project in repository.

**POM Example**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.companyname.project-group</groupId>

<artifactId>project</artifactId>

<version>1.0</version>

</project>

It should be noted that there should be a single POM file for each project.

 All POM files require the **project** element and three mandatory fields: **groupId, artifactId, version.**

 Projects notation in repository is **groupId:artifactId:version.**

 Minimal requirements for a POM:

|  |  |
| --- | --- |
| **Node** | **Description** |
| Project root | This is project root tag. You need to specify the basic schema settings such as apache schema and w3.org specification. |
| Model version | Model version should be 4.0.0. |
| groupId | This is an Id of project's group. This is generally unique amongst an organization or a project. For example, a banking group com.company.bank has all bank related projects. |
| artifactId | This is an Id of the project. This is generally name of the project. For example, consumer-banking. Along with the groupId, the artifactId defines the artifact's location within the repository. |
| version | This is the version of the project. Along with the groupId, It is used within an artifact's repository to separate versions from each other. For example:  **com.company.bank:consumer-banking:1.0**  **com.company.bank:consumer-banking:1.1.** |