Build Tools

A build tool is a tool that automates everything related to building the software project. Building a software project typically includes one or more of these activities:

- Generating source code (if auto-generated code is used in the project).
- Generating documentation from the source code.
- Compiling source code.
- Packaging compiled code into JAR files or ZIP files.
- Installing the packaged code on a server, in a repository or somewhere else.

In the beginning there was Make as the only build tool available. Later on it was improved with GNU Make. However, since then our needs increased and, as a result, build tools evolved.
JVM ecosystem is dominated with three build tools:

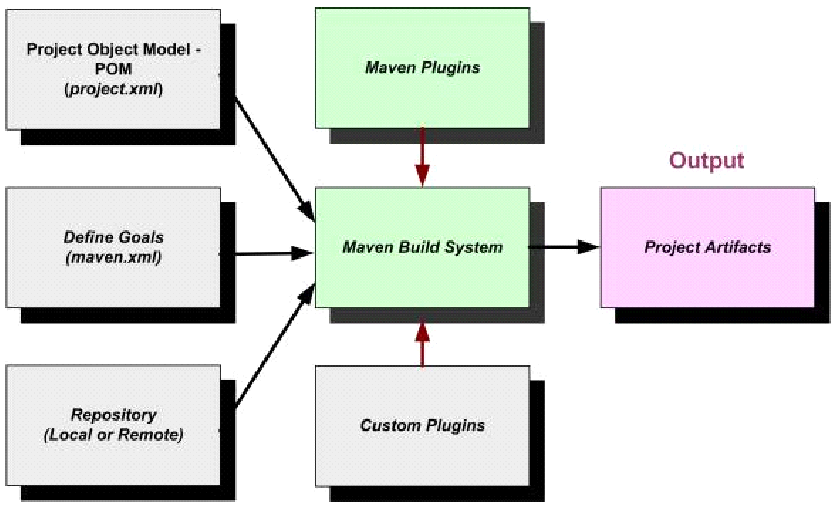
Apache Ant with Ivy	• Released in 2000; • XML based configuration (build.xml) • Adapted Apache Ivy for dependency management (ivy.xml) Ant jar # to run the ant task that creates a jar.	**<u>Advantages</u>**: • Control on the build **<u>Disadvantage</u>**: • XML is unmanageably big
Maven	Released in 2004 XML based build specification script	**<u>Advantages</u>**: • Provides targets (goals) • Can download dependencies over network • Maven Lifecycle • Maven goal that runs both unit tests and static analysis with CheckStyle, FindBugs and PMD (mvn verify) **<u>Disadvantage</u>**: Customization of goals is difficult Configuration written in XML is cumbersome
Gradle	Released in 2012 Google adapted Gradle as its build tool for Android OS DSL based on Groovy (JVM language)	**<u>Advantages:</u>** • Simpler build script • Life cycle (compilation - static analysis - test -package - deployment) • convention is good and so is flexibility

Maven Intro.

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information

Maven provides developers ways to manage the following ..

- Build Tool
- Dependency Management tool
  - Dependencies and versions
- Project Structure
- Building, Publishing and deploying



POM:

- A fundamental unit of work
- An XML file that contains information about project and configuration details
- Describes a project
- Provides info about Name, version, Artefact, Source code locations, dependencies etc.,

Installation:

1. Download from Maven.apache.org
2. Setup path
   a. export M2_HOME= <The folder path where Maven has been extracted to>
   b. export PATH=<provide path of Maven's bin folder>: ${PATH}
   c.

mvn --version	To check the version

Archetype is a model how you want your project structure should look like. It contains..

1. Folder Structure
2. pom.xml

Clean	deletes all artifacts and targets which are created already.
Compile	used to compile the source code of the project.
Test	test the compiled code and these tests do not require to be packaged or deployed.
Package	package is used to convert your project into a jar or war etc.
Install	install the package into local repository for use of other project.

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

NOTE: ${basedir} denotes the project location

Maven Setup

sudo add-apt-repository ppa:webupd8team/java -y sudo apt-get update sudo apt-get install oracle-java8-installer	echo $JAVA_HOME Display JAVA_HOME variable path. IF NOTHING APPEARS THEN SET IT WITH THIS export JAVA_HOME=/usr/lib/jvm/java-8-oracle
apt-cache search maven	To get all available Maven package
sudo apt-get install maven	installs the Maven in /usr/share/maven The Maven configuration files are stored in /etc/maven
mvn -version	To verify the installation of maven
Set Maven Path	export M2_HOME =
mkdir mavenproj	Create a new directory "mavenproj"
cd mavenproj	
mvn archetype:generate	Downloads all the required plugins.
Choose a number or apply filter	Default is 984
Choose version	Choose 6 which is the latest version
Define value for property 'goupId'	Asks you to give a groupId, it's like giving a package name e.g. com.qshore
Define value for property 'artefactId'	MavenFirstApp
Define value for property 'version'	Press enter to take the default value 1.0
Define value for property 'package' org.raghu.qshore	It automatically picks the groupId, Press ENTER
COMPILE THE CODE	Make sure that you are in Maven application directory where you see pom.xml e.g. MavenFirstApp
mvn compile	It downloads all the dependencies and compiles the entire code;
mvn package	Packages the app into jar file /home/Home/mavenproj/MavenFirstApp/target/MavenFirstApp-1.0-SNAPSHOT.jar Also runs the test cases
java -cp target/MavenFirstApp-1.0-SNAPSHOT.jar com.qshore.App	

Maven build cycle

Build Life Cycles

Maven has 3 built-in build life cycles. These are:

1. default
2. clean
3. site

Phase	Handles	Description
prepare-resources	resource	Resource copying can be customized in this phase.
compile	compilation	Source code compilation is done in this phase.
package	packaging	This phase creates the JAR / WAR package as mentioned in packaging in POM.xml.
install	installation	This phase installs the package in local / remote maven repository.

Clean Lifecycle:

handles everything related to removing temporary files from the output directory, including generated source files, compiled classes, previous JAR files etc.

Build Life Cycle

Lifecycle Phase	Description
validate	Validates whether project is correct and all necessary information is available to complete the build process.
compile	Compile the source code of the project.
test	Run tests using a suitable unit testing framework(Junit is one). test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
package	Take the compiled code and package it in its distributable format, such as a JAR, WAR, or EAR file.
verify	Run any check-ups to verify the package is valid and meets quality criteria.
install	Install the package into the local repository, which can be used as a dependency in other projects locally.
deploy	done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

Site Lifecycle

Maven Site plugin is generally used to create fresh documentation to create reports, deploy site etc.

pre-site	execute processes needed prior to the actual project site generation
Site	generate the project's site documentation
post-site	execute processes needed to finalize the site generation, and to prepare for site deployment
site-deploy	deploy the generated site documentation to the specified web server