**BUild Tools: -- MAVEN & ANT**

**Build:**

**The process of converting your source code into readly installable components.**

\* What is the o/p for your build process?

As we are working for java applications the o/p of build is war files.

\* What are the diff build tools?

JAVA -- ANT,Gradle,MAVEN

.Net -- NAnt, MS Build

C, C++ -- MAKE

**JAR: Java Archive**

**A** Jar file is just a zip file,usually it contains compiled java code (.classes)

Extension : .jar

**WAR: Web Archive**

A war file is a collection of web components like servlets,jsp,html,web.xml and war contains independent .jar files & .class files.

Extension : .war

**EAR: Enterprise Archive:**

Combination of jar files/war files and ejbs(enterprise java beans).

Extension : .ear

**MAVEN**

**pom.xml**

**-->Download Apache Maven s/w from apache maven org(http://maven.apache.org/download.cgi)**

**--> After completing the Download, By using WINSCP tool copy the maven s/w from your local system to your Linux server.**

**-->Now login to Linux server and copy maven s/w from your home directory to required location, for example /home/satya/apache-maven3.1 -->/opt/MAVEN/**

**-->Now login as sudo user , then extract apache-maven3.1.zip file**

**wget https://downloads.apache.org/maven/maven-3/3.8.1/binaries/apache-maven-3.8.1-bin.tar.gz**

**-->$sudo su -**

**satya password: \*\*\*\*\*\***

**#cd /opt/MAVEN**

**#unzip apache-maven3.1.zip**

**#ls**

**apache-maven3.1 apache-maven3.1.zip**

**#cd apache-maven3.1**

**#pwd**

**/opt/MAVEN/apache-maven3.1**

**--> Our maven home directory is /opt/MAVEN/apache-maven3.1**

**--> Now set environment variable for Maven.**

**-->#cd /etc/profile.d**

**#vi custom.sh**

**export MAVEN\_HOME=/opt/MAVEN/apache-maven3.1**

**export PATH=$PATH:$MAVEN\_HOME/bin**

**:wq!-->save it.**

**-->#source custom.sh**

**-->Now check maven is instaled or not in your system,**

**#mvn –version**

**apache-maven3.1.--------------we got the response like this.**

**-->#cd /opt/maven-project**

**# mvn archetype:generate**

**-DgroupId={project-packaging}**

**-DartifactId={project-name}**

**-DarchetypeArtifactId=maven-archetype-webapp**

**-DinteractiveMode=false**

**---->Above maven command is used to create a web appln project.**

**#mvn archetype:generate**

**-DgroupId=com.gspann**

**-DartifactId=safeway**

**-DarchetypeArtifactId=maven-archetype-webapp**

**-DinteractiveMode=false**

**#mvn archetype:generate -DgroupId=com.gspann -DartifactId=safeway -DarchetypeArtifactId=maven-archetype-webapp -DinteractiveMode=false**

**#ls**

**safeway**

**#cd safeway**

**#ls**

**pom.xml src**

**-->now run the pom.xml by using maven commands**

**#mvn package**

**#ls**

**pom.xml src target**

**#cd target**

**#ls**

**safeway.war**

**--------->Second method for maven project.<------------------------**

**#cd /opt/maven-project**

**#mvn archetype:generate**

**[INFO] Scanning for projects...**

**[INFO] Searching repository for plugin with prefix: 'archetype'.**

**[INFO] ------------------------------------------------------------------------**

**[INFO] Building Maven Default Project**

**[INFO] task-segment: [archetype:generate] (aggregator-style)**

**[INFO] ------------------------------------------------------------------------**

**[INFO] Preparing archetype:generate........................................**

**..........................................**

**...........................................**

**Choose a number or apply filter (format: [groupId:]artifactId, case sensitive contains): 399:**

**-->if u know the archetype number for web application enter the number or press enter**

**Choose org.apache.maven.archetypes:maven-archetype-quickstart version:**

**1: 1.0-alpha-1**

**2: 1.0-alpha-2**

**3: 1.0-alpha-3**

**4: 1.0-alpha-4**

**5: 1.0**

**6: 1.1**

**Choose a number: 6: 6**

**-->choose any one of the version , here i am choosing version 1.1 so the number is 6.**

**Define value for property 'groupId': : com.gspann**

**Define value for property 'artifactId': : safeway**

**Define value for property 'version': 1.0-SNAPSHOT:**

**Define value for property 'package': com.gspann:**

**Confirm properties configuration:**

**groupId: com.gspann**

**artifactId: safeway**

**version: 1.0-SNAPSHOT**

**package: com.gspann**

**Y: y**

**# ls**

**safeway**

**#cd safeway**

**pom.xml src**

**#mvn package**

**#ls**

**pom.xml src target**

**#cd target**

**#ls**

**safeway.1.0-SNAPSHOT.jar**

**POM.XML:------**

**T**he pom.xml contains **maven co-ordinates, meta data, build information, resources and dependencies.**

**maven co-ordinates: Gropuid,artifactid,version**

**matadata : what the project is and what the version is,**

**build information: which says weather the project is jar /war/ear**

**resource and dependencies: set of resource and dependencies are required for successful build**

MAVEN BUILD:-----

build life cycle is mainly having 2 phases like

mvn compile

mvn package

-->specify the build phase what you need.

The previous phase automatically run when you choose the mvn package.

--> if you specify mvn package then mvn compile is automatically run.

**Some Phases:---**

**Validate --> it checks everything is in order, pom.xml is in the correct place and configuration is in proper and code is in proper way.**

**Compile: compiles the java file into .class files**

**Test : run the test cases**

**Package : package the code like combine all your .class files into a single file(jar)**

**----> if you run directly mvn package phase then it automatically runs the previous phases like validate, compile, test**

**----> if any phase fails then the package phase also fails automatically.**

**Install : install that package(artifact) into local maven repository, that is there in your system.**

**--> here install is not a deployment in appln servers.**

**Deploy: publishing your artifact to a remote repository for other developers to download that artifact.**

**deploy is not in the sense of app server deployment.**

**For web application dependencies are**

**<dependencies>**

**<dependency>**

**<groupId>javax.servlet</groupId>**

**<artifactId>servlet.api</artifactId>**

**<version>2.4</version>**

**<scope>provided</scope>**

**</dependency>**

**<dependency>**

**<groupId>javax.servlet.jsp</groupId>**

**<artifactId>jsp.api</artifactId>**

**<version>2.1</version>**

**<scope>provided</scope>**

**</dependency>**

**</dependencies>**

**<build>**

**<plugins>**

**<plugin>**

**<groupId>org.apache.maven.plugins</groupId>**

**<artifactId>maven.compiler-plugin</artifactId>**

**<version>2.0.2</version>**

**<configuration>**

**<source>1.4</source>**

**<target>1.4</target>**

**</configuration>**

**</plugin>**

**</plugins>**

**</build>**

**--->for example we choose j2ee1.4 web application archetype number(269),**

**---> for this web appln plugins configuration source and target version taken as 1.4.**

**---> if you did not specify the plugins in pom.xml then its automaticalyy compiles the code with appropriate plugins.**

**---> In maven 3.0 on words its uses plugins configuration source and target as 1.5.**

**Jetty server plugin :**

**-----> jetty is a server and when ever we add the jetty plugin in pom.xml, after generating the artifact, it automatically deploy in jetty server.**

**open browser and type like: http://10.210.0.5:8080/myweapp**

**here you can the content of that deployed artifact(.jar/.war) like Helloworld message.**

**<build>**

**<plugins>**

**<plugin>**

**<groupId>org.mortbay.jetty</groupId>**

**<artifactId>maven.jetty-plugin</artifactId>**

**<version>6.1.10</version>**

**</plugin>**

**</plugins>**

**</build>**

**Sample pom.xml file with jetty server plugin:**

**<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.Indigo</groupId>**

**<version>1.0-SNAPSHOT</version>**

**<artifactId>INDIGO</artifactId>**

**<packaging>jar</packaging>**

**<name>INDIGO</name>**

**<properties>**

**<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>**

**</properties>**

**<dependencies>**

**<dependency>**

**<groupId>javax.servlet</groupId>**

**<artifactId>servlet.api</artifactId>**

**<version>2.4</version>**

**<scope>provided</scope>**

**</dependency>**

**<dependency>**

**<groupId>javax.servlet.jsp</groupId>**

**<artifactId>jsp.api</artifactId>**

**<version>2.1</version>**

**<scope>provided</scope>**

**</dependency>**

**</dependencies>**

**<build>**

**<plugins>**

**<plugin>**

**<groupId>org.mortbay.jetty</groupId>**

**<artifactId>maven.jetty-plugin</artifactId>**

**<version>6.1.10</version>**

**<configuration>**

**<scanIntervalSeconds>5</scanIntervalSeconds>**

**</configuration>**

**</plugin>**

**</plugins>**

**</build>**

**</project>**

**---> the above pom.xml will compile the code and packing it and deploy the artifact in to jetty server.**

**---><configuration>**

**<scanIntervalSeconds>5</scanIntervalSeconds>**

**</configuration>**

**the above configuration will checks for every 5 mins and if any changes is there in class file then its automatically**

**compiles and package it and deploy it in jetty server.**

**Sample POM.XML :----**

**<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/maven-v4\_0\_0.xsd">**

**<modelVersion>4.0.0</modelVersion>**

**<groupId>com.gspann</groupId>**

**<artifactId>mypage</artifactId>**

**<packaging>war</packaging>**

**<version>1.0-SNAPSHOT</version>**

**<name>mypage Maven Webapp</name>**

**<url>http://maven.apache.org</url>**

**<dependencies>**

**<dependency>**

**<groupId>junit</groupId>**

**<artifactId>junit</artifactId>**

**<version>3.8.1</version>**

**<scope>test</scope>**

**</dependency>**

**</dependencies>**

**<build>**

**<finalName>mypage</finalName>**

**</build>**

**</project>**

**------------------------------- -------------------------------- ----------------------**

**<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.gspann</groupId>**

**<artifactId>safeway</artifactId>**

**<version>1.0-SNAPSHOT</version>**

**<packaging>jar</packaging>**

**<name>safeway</name>**

**<url>http://maven.apache.org</url>**

**<properties>**

**<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>**

**</properties>**

**<dependencies>**

**<dependency>**

**<groupId>junit</groupId>**

**<artifactId>junit</artifactId>**

**<version>3.8.1</version>**

**<scope>test</scope>**

**</dependency>**

**</dependencies>**

**</project>**

**HTML CODE :--------(for index.jsp page)**

**<html>**

**<body>**

**<h2> Hello Friends! </h2>**

**<h3> !!!!!!!! WELCOME TO HYDERABAD !!!!!!!! </h3>**

**<h4> $@TY@ </h4>**

**<marquee behavior="alternate"> S@TY@</marquee>**

**<font color="WHITE"><marquee direction="left" style="background:RED">YSN</marquee></font>**

**<marquee behavior="scroll" direction="up">**

**<img src="http://www.html.am/images/html-codes/marquees/arrow-up.gif" width="32" height="32" alt="Arrow pointing up" />**

**</marquee>**

**</body>**

**</html>**

**==================================**

**<html>**

**<body>**

**<h2> Hello Friends! </h2>**

**<h3> !!!!!!!! WELCOME TO HYDERABAD !!!!!!!! </h3>**

**<marquee behavior="alternate"> S@TY@</marquee>**

**<font color="WHITE"><marquee direction="left" style="background:RED">YSN</marquee></font>**

**<marquee behavior="scroll" direction="up">**

**<marquee behavior="alternate">**

**<img src="http://www.html.am/images/html-codes/marquees/coffee.gif" width="150" height="108" alt="Cup of coffee on saucer" />**

**</marquee>**

**</marquee>**

**</body>**

**</html>**

[**http://mvnrepository.com/artifact/org.apache.activemq/activemq-stomp/5.12.1**](http://mvnrepository.com/artifact/org.apache.activemq/activemq-stomp/5.12.1)

**What do you mean by a Maven Repository?**

**Maven repository is the location of a directory where all the related project artifacts, jars, libraries, plugins are kept and can be utilized by Maven easily.**

## **What is POM?**

## **POM stands for Project Object Model. In Maven, it is a fundamental Unit of Work and it is an XML file. You can find it in the base directory of the project. It consists of information about the project and various configuration details used by Maven to build the project(s).**

**What do you mean by Maven Archetype?**

**Maven archetype is basically a plugin assigned with the job of generating the project structure according to a particular skeleton or template.**

**How to determine the version of Maven in our system?**

**To determine the version of Maven we are using in our system, we need to enter the below command in the console.**

**mvn –version**

**What are Maven Plugins?**

**Maven plugins are the basic component of a Maven framework. Each of the plugins has a specific task to be performed.**

**What does goal in Maven mean?**

**A collection of Maven goals constitute a phase. Each goal is defined for a particular job involved in project management and execution of the build.**

**What is the location where Maven dependencies are downloaded?**

**The project artifacts, dependencies, and jars downloaded by Maven are placed in the local repository of Maven. The folder .m2 is by default the location for the local repository.**

**What is the purpose of command mvn clean in Maven?**

**mvn clean aims to clean the project artifacts created by the previous Maven builds from the target directories. This is generally executed before initiating a new build process.**

===ANT============(**build.xml)**

In build.xml project task contain a set of targets and each target contains set of tasks.

<?xml version="1.0" ?>

<project name="HelloWorld" default="compress">

<target name="init">

<mkdir dir="build/classes" />

<mkdir dir="dist" />

</target>

<target name="compile" depends="init">

<javac srcdir="src" destdir="build/classes" />

</target>

<target name="compress" depends="compile">

<jar destfile="dist/Gspann.jar" basedir="build/classes" />

</target>

<target name="execute" depends="compile">

<java classname="com.vaannila.helloworld.HelloWorld" classpa

th="build/classes" />

</target>

<target name="clean">

<delete dir="build" />

<delete dir="dist" />

</target>

</project>

Create -- directories → src main

Empty file in main

src→

cd src/com/vaannila/helloworld/

[root@linux helloworld]# pwd

src/com/vaannila/helloworld

HelloWorld.java

package com.vaannila.helloworld;

public class HelloWorld

{

public static void main(String args[])

{

System.out.println("Hello Gspann!");

}

}

