Build tools- ant, gradle, maven…

Maven - 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 is open source and it for Java based projects.

Build tool:

1.Get the required dependencies automatically from maven central repo using pom.xml-dependencies and add them to class path

Package the source code as jar/war or ear using maven plugins

It can execute automation test cases using surefire plugin in pom.xml

Pom.xml- It is main configuration file for maven project where all the required dependencies and plugins are configured.

Setup Maven :

Prerequisite:

Download java and java\_home should be set

1. Using cmd prompt

Download maven [apache-maven-3.3.9-bin.zip](http://mirrors.sonic.net/apache/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.zip) from <https://maven.apache.org/download.cgi>

Extract it to any local folder

Set env var-: add maven path till bin(C:\Program Files\apache-maven-3.3.9\bin) to existing path variable

Open a new cmd prompt window- mvn –version

1. Using IDE(eclipse,netbeans,intellij)

Install maven plugin in the ide.

In eclipse- install new software- give <http://download.eclipse.org/technology/m2e/releases> and complete the installation

Every maven Dependency has 3 main components:

groupId – package name

artifactId – project/jar/war name

version-

All dependencies can be searched from <https://mvnrepository.com/>.

.m2- is present in your local machine as maven local repository

Maven cmds:

mvn clean - cleans the prev build files/all existing files in target folder

mvn install - it gets all the required dependencies and build the project and give the specified packaged jar/war into target folder

mvn package

mvn test –

Execute automation using maven:

We need to specify testing.xml path in pom.xml using surefire plugin and view the reports under target-surefire-reports

Vcs: to maintain code at one place to be used by multiple team members at same time

centralized version control system – maintains only one central repo

distributed version control system – maintains individually –local repo and also global repo, merging and comparing the files for differences is easy.

**Git** : In real time source code is shared by many people and we cant share the Zip code files from our local system to others everytime, so to avoid these situations source code repositories are maintained.

* Which helps team members to share the code simultaneously
* It helps to keep track and provide changes in the files
* Every change / revisions are marked by timestamp and name of the persons
* It is also called as the version control system.

Two types :

1. Centralized version control systems : Always has only one main Repository Ex. CVS, SVN, Perforce etc…
2. Distributed Version control Systems: can have the several copies , maintains multiple repositories for the same source code: it maintains local repo and also global repo which makes easy when merging or comparing the files. Ex: Git

Refer ppt for more terminologies…

Commit – move changes to local repo

Push- move changes to central repo

Fork – copying the code from some other repo to your local repo

Clone – get the forked copy into your local system or ide using clone url

Pull – fetch+merge – getting updates from central repo and merging them with local repo changes

Fetch – getting updates from central repo

Merge- merging them with local repo changes

**Installation of git:**

1. install as eclipse plugin - egit

2. cmd prompt – download git <https://git-scm.com/download/win> and execute it

3. git stash, git easy (other UI tools we can use directly for installation , cmd prompt installation is suggestable and faster……)

Refer-txt file for commands

First we need to have a Repository, so we need to signup (for the first time) in the githud to create the repository.

**Git and github:**

Git – is mainly used for code changes and to track of files which it maintains

Github- it’s a online repository , it takes all the backup of the files and which can be shared to team memebers also.

Git commit- first push to local repo

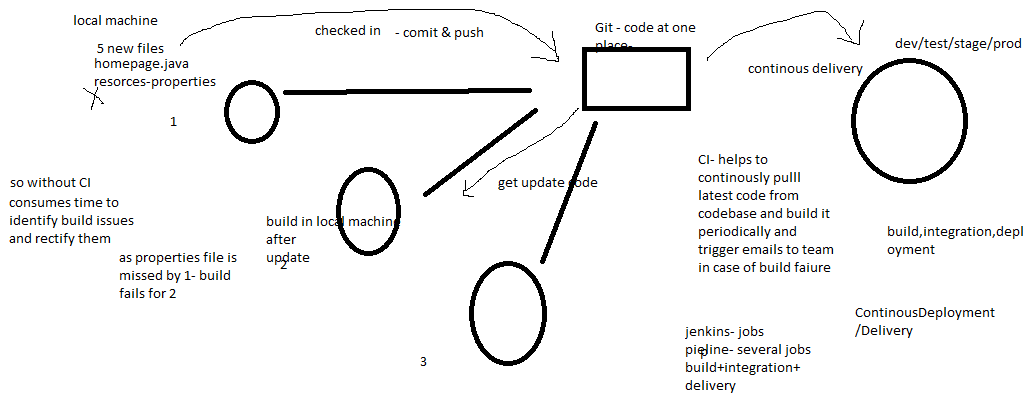
Push – which actually moves to central repo

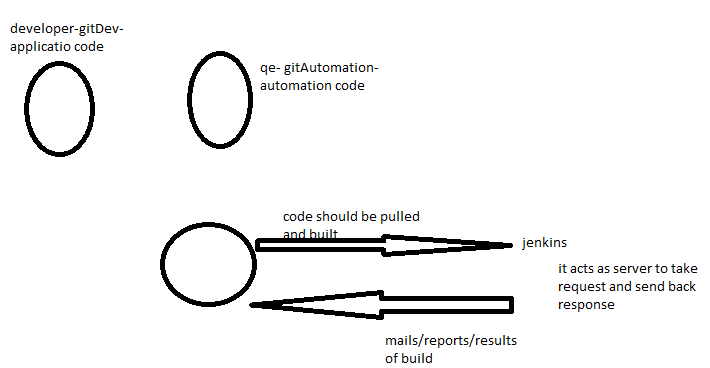
<https://github.com/latha-gudoor/EclipseWorkSpace.git>

Steps:

1. We need to create the repository and copy the url of the repository
2. Goto workspace and do git bash – which opens the cmd prompt for git
3. If you are using it for the first time then do “ git init” which initializes the git ..” .git” folder is created
4. Now add the projects which we want to push to repo(give the url which represents the address for the repo)

CI tools- bamboo, Hudson(old version of jenkins),Jenkins





**What are build tools?**

Build tools are programs that automate the creation of executable applications from source code(eg. .apk for android app). Building incorporates compiling,linking and packaging the code into a usable or executable form.

Basically build automation is the act of scripting or automating a wide variety of tasks that software developers do in their day-to-day activities like:

1. Downloading dependencies.
2. Compiling source code into binary code.
3. Packaging that binary code.
4. Running tests.
5. Deployment to production systems.

**Why do we use build tools or build automation?**

In small projects, developers will often manually invoke the build process. This is not practical for larger projects, where it is very hard to keep track of what needs to be built, in what sequence and what dependencies there are in the building process. Using an automation tool allows the build process to be more consistent.

**Various build tools available(Naming only few):**

1. For java - Ant,Maven,Gradle.
2. For .NET framework - NAnt
3. c# - MsBuild.