Jenkins

Continuous Integration &

Continuous Delivery/Deploy

- Whenever developers write code, we integrate all that code of all developers at that point of time and we build, test and deliver/deploy to the client. This process is called CI & CD.
- Jenkins helps in achieving this.
- So instead of doing night builds, build as and when commit occurs by integrating all code in SCM tool, build, test and checking the quality of that code is what Continuous Integration.
- So in a day, there will be so many integrations, builds, tests & deliveries/deployments.
- So bugs will be reported fast and get rectified fast. So development happens fast.

Key Terminology

- **Integrate**: Combine all code written by developers till some point of time.
- **Build**: Compile the code and make a small executable package.
- **Test**: Test in all environments whether application is working properly or not.
- **Archived**: Stored in an artifactory so that in future we may use/deliver again.
- **Deliver** : Handing the product to Client

- **Deploy**: Installing product in client's machines.

Jenkins work flow

- We attach Git, Maven, Selenium & Artifactory plug-in's to Jenkins.
- Once Developers put code in Git, Jenkins pull that code & send to Maven for build.
- Once build is done, Jenkins pull that built code and send to selenium for testing
- Once testing is done, then Jenkins will pull that code and send to artifactory as per requirement and so on....
- We can also deploy with Jenkins

Ways of Continuous Integration

- Can do manually
- Can write scripts
- Can use tool like Jenkins
- Benefits of CI
- Detect bugs as soon as possible
- If you want you can stop the SDLC process at any stage
- Eg: You can stop at test stage only or you can continue till deployment.
- Maintains history (Logs) for reference

Why only Jenkins?

- It has so many plug-ins.
- You can write your own plug-in
- You can use community plug-ins
- Jenkins is not just a tool. It is a framework. i.e. You can do what ever
- You want. All you need is plug-ins.
- We can attach slaves (nodes) to Jenkins master. It instructs others (slaves) to do Job. If slaves are not available, Jenkins itself does the job.
- Jenkins also acts as crone server replacement. i.e. can do repeated tasks automatically
- Running some scripts regularly
    - Eg: Automatic daily alarm.
- Can create Labels (Can restrict where the project has to run)

Jenkins

- Jenkins Architecture
- Client-Server model
- When you install Jenkins it acts as master.
- Jenkins invoke clients to perform jobs.

Supported OS

- Can install in any OS
- You will be accessing Jenkins through web only.

- Choose Long Term Support release (to get support)
- Install Java
- Install web server

Jenkins

- Enterprise edition - Hudson
- Open source – Jenkins

Three types of configurations

- Global Configuration
- Node Configuration
- Job Configuration

Jenkins

- Get familiar with Jenkins dashboard
- Jenkins services on windows
- localhost:8080/restart
- localhost:8080/stop
- localhost:8080/start
- Create sample job & familiar with all options & build.
- Job/Project/Item: Projects are sometimes referred to as jobs or items. Jenkins appears to use these terms interchangeably.
- Workspace: The workspace is the location in your computer where Jenkins places all files related to the Jenkins project. By default each project or job is assigned a workspace location and it

contains Jenkins-specific project metadata, temporary files like logs and any build artifacts, including transient build files.

Plugin

- With Jenkins, nearly everything is a plugin and that nearly all functionality is provided by plugins. You can think of Jenkins as little more than an executor of plugins.
- Plugins are small libraries that add new abilities to Jenkins and can provide integration points to other tools.
- Since nearly everything Jenkins does is because of a plugin, Jenkins ships with a small set of default plugins -- some of which can be upgraded independently of Jenkins