JENKINS CHEAT SHEET

Jenkins

Jenkins is a software which allows you to do the continuous integration on your application/software lifecycle. It gets installed on the server where the central build will take place. Now let's understand its workflow.

Management

Configure system:

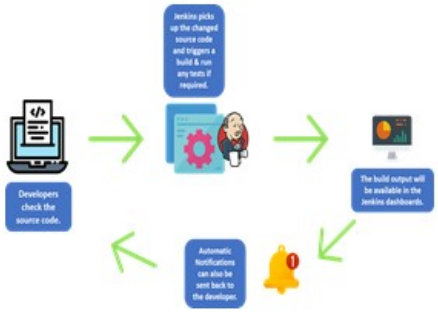
- Can be used to manage paths to various tools to use in builds.
- Jenkins dynamically adds the config fields after the plugins are installed.

Manage Plugins:

- Plugins can be removed, updated and installed by manage plugin screen.

System info:

- List all the current java properties and system environment variables.
- System log- view Jenkins log in real time.
- Script console- lets you run groovy scripts on the server.
- Manage nodes- configure the number of builds you want.
- Shutdown- click prepare to shutdown link to prevent any new builds from being started. After all current builds are finished, Jenkins will shut down cleanly.



Unit Testing

Testing unit in Jenkins:

- Step 1 : Go to the dashboard and choose an existing project and click configure
- Step 2 : browse to add a build step and invoke Ant
- Step 3: Click on advanced
- Step 4 : in the build file section enter the location of build.xml
- Step 5: add post build option and click Publish Junit test result report
- Step 6 : ensure the report is in the folder of the project workspace The "*.xml" basically tells Jenkins to pick up the result xml files which are produced by the running of the Junit test cases. Click Save after done.
- Step 7 : Click on build and check logs to see if successful or not with applications.

Automated Testing

- Step 1: go to plugins and choose selenium plugins and click to install.
- Step 2: go to configure system and select on selenium jar and save
- Step 3: Go to dashboard and select the config option for the project at hand
- Step 4: Click on add build step and choose SeleniumHQ htmlSuite Run"
- Step 5: Add the required details and click on save, execute and build. The test is executed, and a report is built.

Automated Deployment

- Head to the manage plugins and install the respective plugins
- It takes the war ear file and deploys that to the running remote application build
- Go-to build and configure and click on 'deploy to war/ear to container'
- In the war container section save details about the destination server and click save.

Backup Plugin

- To tweak backup settings via setup
- To backup Jenkins config
- To restore config from a previous backup.
    - Alternatively you can use SCM(sync config plugin) or ThinBackup for global and job configurations.

Notifications

- Jenkins comes with a feature to add email notifications to the build project
- Go-to Manage Jenkins → Configure System. In the email notification space enter the require STMP server and use email suffixes.
- Configure the recipients so that they would receive notification about broken or unstable builds
- Notification plugins such as Tikal Knowledge allows job status notification for JSON and XML formats.
- Options :
    - Format - either Json or XML types
    - Protocol- TCP, UDP or HTTP
    - Event - job event that triggers the notification
    - URL- destination to send notifications to.
    - Timeout - default timeout 30ms

Manage Plugins

- To uninstall plugins , go-to manage plugins and click on the installed tab and click on uninstall for the plugin
- In Case of need to install an older version of the plugin, download from the site and click on Upload option to do it manually.

Code Analysis

They provide utilities for static code analysis. Some tools are CheckStyle, FindBugs, PMD etc.
Provides details like :

- Total warning in a job
- Showing of new and fixed warning of a build
- Trend reports showing warnings per build
- Warnings per module, package or category
- Detailed reports of found warnings.

Security

- Ability to have secure config for different users in place.
- Click on manage Jenkins and Configure global security
- Set parameters in the enable security section
- Add the users and go-to 'manage users' for permissions
- To set authorizations, go-to configure global security, and click on matrix based security

Server Maintenance

Commands in Jenkins: (URLs)

- http://localhost:8080/jenkins/exit – shutdown Jenkins
- http://localhost:8080/jenkins/restart – restart Jenkins
- http://localhost:8080/jenkins/reload – to reload

To backup Jenkins Home:

- Go to configure system in manage Jenkins
- Select a partition that has to most free space.
- Perform automated clean-up options to avoid this.

Build Pipeline

- First go-to manage plugin and install build pipeline plugin.
- To see a build pipeline click the (+) on the dashboard
- Enter any name and click on the view. Choose build pipeline view.
- Accept the default settings and add the name of the project.
- A view of entire pipeline with statuses will be visible.

Remote Testing

Selenium tests can be run on remote slave machines via master slave and selenium suite plugin installation

- Step 1: go-to master Jenkins server and manage nodes
- Step 2: Click on configure for the slave machine
- Step 3 : put the launch method as 'Launch slave agents via Java Web Start'
- Step 4: open a browser instance of the master Jenkin on the slave machine, then manage nodes and select the DXBMEM30
- Step 5 : Scroll down and select the launch option and hit run
- Step 6: Configure tests to run on the slave
- Step 7 : make sure the selenium part of the job is configured. Make sure that the Sample.html file and the selenium-server.jar\

IntelliPaat

FURTHERMORE:
JenkinsTraining

edureka!

JENKINS CHEAT SHEET

Learn DevOps from experts at edureka.co

What is Continuous Integration?

Continuous Integration is a software development practice in which developers are required to frequently commit changes to the source code in a shared repository. Each commit is then continuously pulled & built. Jenkins is an open source, Continuous Integration (CI) tool, written in Java. It continuously pulls, builds and tests any code commits made by a developer with the help of plugins.

Install Jenkins On Ubuntu

This installation is specific to systems operating on Ubuntu. Follow the below steps:

```
Step 1: Install Java
$ sudo apt update
$ sudo apt install openjdk-8-jdk
Step 2: Add Jenkins Repository
$ wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key
| sudo apt-key add –
Step 3: Add Jenkins repo to the system
$ sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable
binary/ > /etc/apt/sources.list.d/jenkins.list’
Step 4: Install Jenkins
$ sudo apt update
$ sudo apt install Jenkins
Step 5: Verify installation
$ systemctl status Jenkins
Step 6: Once Jenkins is up and running, access it from the
link:
http://localhost:8080
```

Build Pipeline

Build pipeline can be used to chain several jobs together and run them in a sequence. Let's see how to install Build Pipeline:

Jenkins Dashboard-> Manage Jenkins-> Manage Plugins-> Available-> Build Pipeline

Build Pipeline example

Step 1: Create 3 freestyle Jobs (Job1, Job2, Job3)
Step 2: Chain the 3 Jobs together
Job1 ->configure ->Post Build ->Build other projects ->Job2
Job2 ->configure ->Post Build ->Build other projects ->Job3
Step 3: Create a build pipeline view
Jenkins Dashboard ->Add view ->Enter a name ->Build pipeline view ->ok ->configure ->Pipeline flow ->Select Initial job ->Job1 ->ok
Step 4: Run the Build Pipeline

Most Commonly Used Jenkins Plugins

Jenkins comes with over 2000 plugins and each plugin has a unique functionality. But when it comes to software development most developers use a set of plugins, such as, Maven, Git, Ant, Docker, Amazon EC2, HTML publisher, Copy artefact, etc.

Follow the below step to install the above plugins or any other Jenkins plugin.

Jenkins Dashboard-> Manage Jenkins-> Manage Plugins-> Available

In the filter text field enter the name of the plugin you want to install.

Different Types of Jenkins Jobs

Jenkins provides the option of choosing from different types of jobs to build your project.

Freestyle Job

Freestyle build jobs are general-purpose build jobs, which provides maximum flexibility. It can be used for any type of project.

Pipeline

This project runs the entire software development workflow as code. Instead of creating several jobs for each stage of software development, you can now run the entire workflow as one code.

Multiconfiguration

The multiconfiguration project allows you to run the same build job on different environments. It is used for testing an application in different environments.

Folder

This project allows users to create folders to organize and categorize similar jobs in one folder or sub folder.

GitHub Organisation

This project scans your entire GitHub organization and creates Pipeline jobs for each repository containing a Jenkinsfile

Multibranch Pipeline

This project type lets you implement different Jenkinsfiles for different branches of the same project.

Jenkins Pipeline

Jenkins pipeline is a single platform that runs the entire pipeline as code. Instead of building several jobs for each phase, you can now code the entire workflow and put it in a Jenkinsfile. Jenkinsfile is a text file that stores the pipeline as code. It is written using the Groovy DSL. It can be written based on two syntaxes:

- Scripted pipeline: Code is written on the Jenkins UI instance and is enclosed within the node block

```
node {
    scripted pipeline code
}
```

- Declarative pipeline: Code is written locally in a file and is checked into a SCM and is enclosed within the pipeline block

```
node {
    declarative pipeline code
}
```

Jenkins Pipeline Syntax Example

```
node {
 stage(‘SCM checkout’) {
 //Checkout from your SCM(Source Control Management)
 //For eg: Git Checkout
 }
 stage(‘Build’) {
 //Compile code
 //Install dependencies
 //Perform Unit Test, Integration Test
 }
 stage(‘Test’) {
 //Resolve test server dependencies
 //Perform UAT
 }
 stage(‘Deploy’) {
 //Deploy code to prod server
 //Solve dependency issues
 }
}
```

Jenkins Tips and Tricks

Start, Stop & Restart Jenkins

```
$ sudo service jenkins restart
$ sudo service jenkins stop
$ sudo service jenkins start
```

Schedule a build Periodically

Jenkins uses a cron expressions to schedule a job. Each line consists of 5 fields separated by TAB or whitespace:

Syntax: (Minute Hour DOM Month DOW)
MINUTE: Minutes in one hour (0-59)
HOURS: Hours in one day (0-23)
DAYMONTH: Day in a month (1-31)
MONTH: Month in a year (1-12)
DAYWEEK: Day of the week (0-7) where 0 and 7 are Sunday
Example: H/2 * * * * (schedule your build for every 2 minutes)

Snippet Generator

Step 1: Create a pipeline job > configure

Step 2: Select pipeline script from pipeline definition

Step 3: Click on Pipeline syntax > snippet generator

Step 4: Step > select Git > enter repo URL

Step 5: Scroll down > Generate pipeline script

Step 6: Copy the script into your pipeline script UI

Deploy a custom build of a core plugin

Step 1: Stop Jenkins.
Step 2: Copy the custom HPI to **$Jenkins_Home/plugins**.
Step 3: Delete the previously expanded plugin directory.
Step 4: Make an empty file called **<plugin>.hpi.pinned**.
Step 5: Start Jenkins.