1. What is DevOps?
2. Suppose If you want to give demo about DevOps How you will start from scratch
3. What is mean Dev ? What is mean by OPS?
4. Can you please explain DevOps Architecture?
5. Can you Please explain Project Architecture?
6. With out Devops How we will release the Application ?
7. What are the version control system worked?
8. Can you pls explain version control?
9. Diff between Git And SVN
10. In Svn which strategy your following in our project ?
11. Diff between tag and branch
12. What is the use of tag, can you explain merging.
13. How you will provide the access SVN ?
14. Have you worked in SVN administration?
15. Can you explain how to push the code in GIT ?
16. Explain Merging in GIT ?
17. How to create the branches in GIT?
18. How to create the branch in SVN
19. How to rename the TAG in SVN?
20. Have you worked in ANT, can you explain bulid.xml
21. What is mean by manfeast file ?
22. What is the Diff stages in bulid.xml
23. what is mean by continues integration
24. what is the advantage of continues integration
25. What are the tools available continues integration
26. with out continues integration i can deploy why we need to use continues integration
27. what is mean by artifacts
28. what are the format of artifacts
29. In your project what kind of of package will be generate
30. How to configure the Job in Jenkins
31. what is the use of master slave configuration, can you explain node concept
32. what is the use of build parameterized how to provide complete Jenkins to the user

33. How to provide particular JOB access in Jenkins
34. what is use of string and test parameters in Jenkins
35. what is the use of plug-in
36. what is use of mulite job configuration
37. Have you worked in Jenkins administration
38. Have you worked jenkins administration,
39. Suppose my jenkins page working slowly how you will trouble shoot
40. why we required slave in jenkins
41. what is the use of master in jenkins
42. what type of permission you will provide for developer in jenkins job
43. How to automate the jenkins Job
44. Diff between POll SCM and bulid periodically
45. What is the use of quit period
46. Suppose if i asked with out using POLL SCM in jenkins how you will automated the build process
47. what are plugins installed in your current project
48. what is diff between ant and maven
49. what is structure of POM
50. what is mean by POM, can you explain maven process
51. what is the diff between clean and deploy
52. what is diff between clean and package
53. which plug -in using for releasing
54. what is the command in maven
55. can you expain types of repository in maven
56. what is the use of M2 repository in maven
57. what are the goals in maven
58. what is meant by setting.xml
59. have you worked in administration in maven
60. what is the use of dependency management in maven
61. what is the use of disruption management in maven
62. what is the use of plugins in maven
63. How to configure the maven in jenkins
64. what is mean by parent path in maven

65. have you worked in nexus administration
66. what is diff between snapshot and release
67. can you explain about your release management
68. which plug-in your using for release management
69. have you worked in pipeline can you pls explain pipeline Architecture in your project
70. what is use of CI and CD
71. what is diff between continues integration and continues delivery
72. how to implement conditional jobs in jenkins
73. What are the tools your worked for deployment
74. what is deployment
75. what is Architecture of chef
76. what is use of nodes in chef
77. what is use of server in chef
78. what is run list command
79. what is the bootstrap command in chef
80. what is use of knife command in chef
81. what is the use of SSL certification in chef
82. what is mean by recipe and cook book in chef
83. can you explain run book in chef
84. what is meant by mera.rb in chef
85. how to install one package with in chef development kit
86. what is mean by role in chef
87. what is use of rundeck? how to configure the rundeck
88. how you will integrate rundeck
89. how to create the jobs in rendeck
90. what is diff between virtualization and docker
91. what is the use of docker
92. what is mean by image in docker
93. what is mean by container in docker
94. Diff between image and container
95. what is mean by docker files
96. how to create the docker image

97. Diff between docker file and compose

98. How to delete the image in docker

99. how to automate the sonarqube in jenkins

100. what is the sonarqube administration

1. SVN
2. GIT } → version control system
3. ANT
4. Maven } → Build Tools
5. Jenkins → C.I continuous Integration
6. RSS → Red hat satilite server
7. Sonar qube → code quality
8. Nexus → Back up
9. vagrant
10. Docker
11. chef
12. Rundeck } → Deployment
13. Service Now
14. Linux.

1. Development Team
2. Devops
3. Environment Team
4. Testing
5. production

Linux commands

1. Mkdir → To create folder [or] directory
2. CD → To open the folder
3. CD .. → one step back
4. CD../.. → Two steps back
5. CD../../.. → Three steps back
6. ls → To check the list of folder [or] files
7. ls -last → To check list of folders [or] files with hidden files also
8. PWD → To check the present location [or] Path.
   Touch → To create the file
10. Vi → To Edit the file
11. ESC: wq! → To save the file and quit
12. ESC: q! → with out save file and quit
13. Sh(or) ./ → To run any script in Linux
14. rm -rf → To remove folder [or] file
15. chmod 777 → To give Access for files

SVN commands

1. svnadmin → create repository name
2. SVN co URL → To check out the code from the SVN server to Local desktop
   ↳ checkout
3. svn ci -m "message" → To check in the code from local desktop to svn server
   ↳ checkin
4. svn update (or) svn up → To update the folder in SVN
5. svn status [or] svn st → To check the current status
6. svn add → To add the files

7. svn delete → To delete the file [or] folder. (3)
8. svn move → To move the file [or] folda
9. svn copy → To copy the file [or] folder
10. svn mkdir → To create folder
11. svn merge → To combine the code.
12. svn log → To Track the history

Repository : It is nothing but empty box. to store the source code by version by version.

Local host:

1. Start the svn
2. sh ctlscript.sh restart
3. sh ctlscript.sh start
4. sh ctlscript.sh stop.

How to create the Repository

Go To putty
↓
Enter (or) select the IP Address [192.168.33.10]
↓
open putty
↓
Enter Login id : vagrant
↓
Password : vagrant
↓
Enter PWD
↓
Enter ls
↓
cd subversion-1.9.4-1/
↓
ls
↓
cd subversion/

↓

cd bin/

↓

ls

↓

sh svnadmin create /home/vagrant/repositary name

Eg : Test

How to Add new svn repositary to configuration

Go to putty

↓

Select IP Address

↓

open the putty

↓

Enter log in ID : vagrant

Password : vagrant

↓

PWD

↓

ls

↓

cd subversion -1.9.4.1/

↓

ls

↓

cd apache2/

↓

cd conf/

↓

vi http.conf

↓

Shift+G

↓

go to svn path

↓

inseat the repositary name

↓

Enter ESC: w9!

↓

cd ../..

↓

sh ctlscript.sh restart

How to create folder in repository

→1. First create the new empty folder in desktop

→2. save name in svnpractice. [Folder name]

→3. open the folder [svn practice]

→4. In this folder open the command prompt [cmd]

→

↓

Enter svn co URL [like http://192.168.33.10:8080/subversi

↓

cd subversion

↓

svn mkdir folder name [Eg test]

↓

svn ci -m "create folder"

↓

SVN Rules

A → Added already just check in the code from local to server using below commands.

svn ci -m "message"

? → you have to Add the folder [or] file then check in using below command

svn add folder [or] file name.
svn ci -m "message"

M → modified file just you have to check in

svn ci -m "message"

How to transmitting file data

→ open the folder svnpractice

→ right click on subversion

→ create one text document

→ click on Right [click]. → go to new → select text document
g:simple enter name and wrote some code → save.

Next go command prompt [cmd]

↓

svn co URL [http://192.168.33.10:8081/subversion/

↓

cd subversion

↓

svn st

↓

svn add simple [text document name enter]

↓

svn ci -m "message"

↓

Transmitting file data

create Repositary structure:

As per the developer requeriements i will create the inside the repositary i will create Trunk, Branch, tag.

How to copy the file from one folder to another folder

create new repository:

Go to putty

↓

Select IP Address [192.168.33.10]

↓

open putty

↓

login as: vagrant

Password (PWD): vagrant

↓

PWD

↓

cd subversion-1.9.4-1/subversion/bin/

sh svnadmin create /home/vagrant/repository name

↓

cd ../..

↓

cd apache2/conf/

↓

vi httpd.conf

↓

insert repository name

↓

:wq! ESC:wq!

↓

cd ../..

↓

sh ctlscript.sh restart

create 3 folders and copy and move the one folder to another folder

- trunk
- branch
- tag

Go to svnpractice [create folder in desktop]

↓

open cmd

↓

svn co http://192.168.33.10:8080/subversion/

↓

svn mkdir trunk branch tag

↓

svn ci -m "message"

↓

create one text document in trunk [or] Any one like branch [or] tag [Eg: Java.text]

↓

svn st

↓

svn ci -m "message"

↓

cd trunk

↓

svn add java.txt [text document name]

8

↓

svn ci -m "message"

↓

cd ..

↓

opy the file from trunk to branch
[source] [Destination]

svn copy sourceURL[Path] DestinationURL[Path] -m "message"

nove the file from one folder to Another older

↓

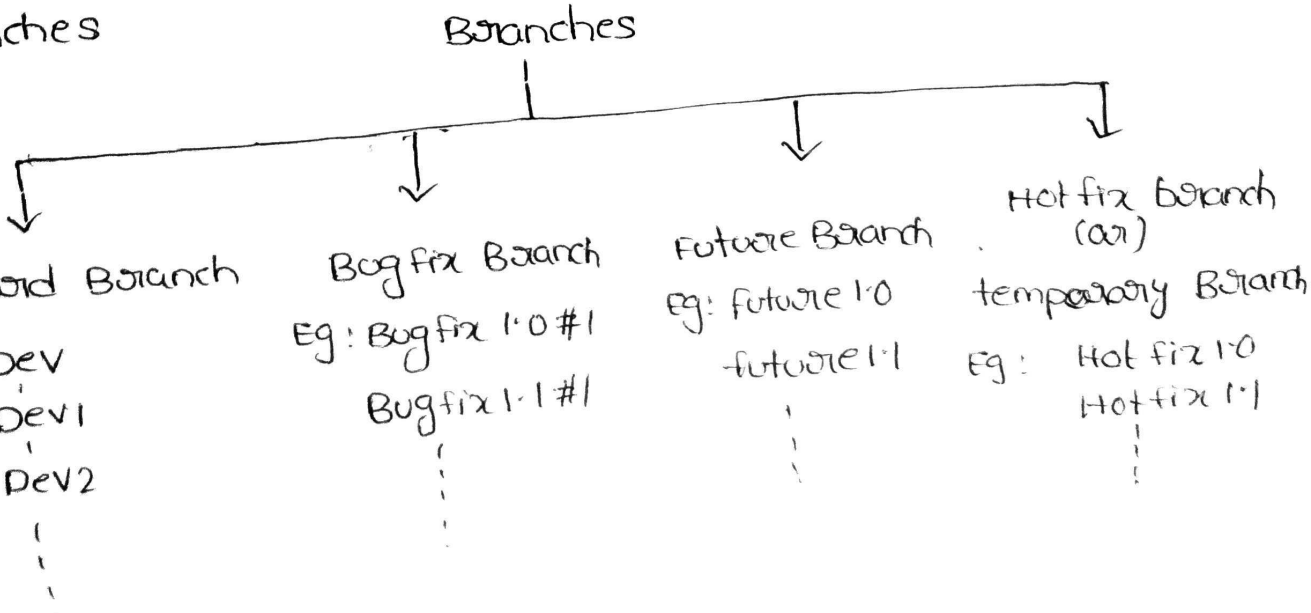
svn move sourceURL[Path] DestinationURL[Path] -m "message"

Branching Startegy [or] base line [or] what is mean by trunk Branch;tag[or] which startegy your following current organi-zation?

trunk: To maintain the original source code of the Particular version.

Branches: Branches are nothing but different different

Branches



Security

→ security means we will provide both

1) Authentication [having credintials, user name & pwd]
2) Authorisation [Access restrictions on svn]

→ First we need to create two empty files

1) touch svn_users
2) touch svn_access

] - a file create inside

→ Present we are in "home/vagrant" directory

home/vagrant

↓

cd subversion-1.9.4-1/apache2/bin/

↓

sh htpasswd -c/home/vagrant/svn_users Branch [like Eg]

↓

Enter New password

↓

Enter Re-Type new password

↓

cd

↓

cat svn_users → In this step Authentication is completed

↓

To provide Authorization → vi svn_access

↓

Add content

Eg:

```
[groups)

[subversion:/)
```

2)

→ In Above steps we provide Authentication & authorization seperately. But to provide security we combined these two both authentication & Authorization

→ To do this we need to add some content the file like presented in.

vi httpd.conf

Subv

cd subversion-1.9.4-1/apache2/bin/

↓

sh htpasswd -c /home/vagrant/svn_users Branch

↓

Enter New password

↓

Re-type new password

↓

cd

↓

cat svn_users

↓

vi svn_access

↓

cd subversion-1.9.4-1/apache2/conf

↓

vi httpd.conf

↓

Add content (insert)

↓

sh ctlscript.sh restcat

Svn Backup:

cd subversion-1.9.4-1

↓

cd subversion/

↓

cd bin/

↓

sh svnadmin dump /home/vagrant/repositary name > /home/vagrant/gow-ami.bkf

[Eg: merge]

↓

enter any name .bkp

↓

cd ../../..

↓

rm -rf merge [delete merge]

↓

ls

↓

cd subversion-1.9.4-1/subversion/bin

↓

sh svnadmin load /home/vagrant/Admin < /home/vagrant/gowthami.bkf

↓

cd ../..

↓

cd apache2/conf

↓

vi http.conf

↓

change the repositary name

↓

cd ../..

↓

sh ctlscript.sh restart

Svn merging

→1 create new repositary at as merge

→2. create 3 folders in repositary

1) Trunk
2) Branch
3) Tag

→3 create 2 folders in Trunk

1) Audio 1.0
2) video 1.0-

Go to putty

↓

create new repositary [Eg: merge]

↓

create one folder in desktop [old development]

↓

open cmd in that folder

↓

svn co http://192.168.33.10:8080/subversion/

↓

svn mkdir trunk Branch tag

↓

svn ci -m "message"

↓

cd trunk

↓

svn mkdir Audio1.0 video1.0

↓

svn ci -m "message"

open cmd in old development folder

↓

cd subversion

↓

copy the trunk folder in to Branch

↓

svn copy http://192.168.33.10:8080/subversion/trunk http://192.168.33.10:80
[source path] /subversion/branch/81 -m "messag
[destination path]

↓

create new folder in desktop name it as[new developme

↓

open cmd in new development folder

↓

svn co http://192.168.33.10:8080/subversion/branch/81

[create 2 folders] ↓

svn mkdir image1.1 pdf1.1

↓

svn ci -m "message"

↓

next again open the old development folder "cmd"

↓

go to cd subversion

↓

cd trunk

↓

svn up

↓

svn merge http://192.168.33.10:8080/subversion/branch/81

↓

svn ci -m "message"

SVN

Version control system is a software that helps software developers to work to gether and also maintaines complete history of their work.

Without VCS what are all Disadvantages

→ 1. Needs to maintain a source code in a shared directory

→ 2. Every one overwrites the source code

→ 3. No revert.

→ 4. Not possible to get who modified the file, what are the changes he made.

Advantages of VCS

→ A place to store your code

→ Historical record of what was done over time.

→ synchronization b/w developers

→ Enables the developers to work in parallel

Repository:

→ The repository is the location where all the data from all the various places is saved

→ In the subversion world, from the clients points points of view, it is the server which holds the database of the project

→ This database contains all the files that are part of the project with all of their past versions

Working copy: → working copy is the snapshot of the repository

→ The content of a project in the repository are usually copied into a local directory. this is known as working directory

→ The repository is shared by all the team, but people do not modify it directly. instead each developer checkout working copy.

→ The working copy is private workplace where developer can do their work within isolated from the rest of the team.

Checkout :-

svn checkout checks out (retrieves) a working copy of the repository into the specified folder

ANT

ANT installation

First install the notepad plus

↓

Next install the Jdk Java

↓

go to command prompt

↓

to run the command [Java -version]

↓

once Java is installed to display the Java version [Eg 1.7.0_79]

↓

go to program files

↓

open Java

↓

go to jdk1.7.0_79 → lib → copy the [tools] option

↓

go to JRE7 → lib → Paste the [tools] option

↓

go to properties [system]

↓

Advanced system settings

↓

Environment variables

↓

go to system variable

↓

Path select

↓

Paste Add the Ant installation apache path like

Add [C:\Users\Admin\Desktop\ANT installation\apache-ant-1.9.7~~\bin~~-bin\apache-ant-1.9.7\bin]

ok

ANT:

ANT is nothing but Another neat tool

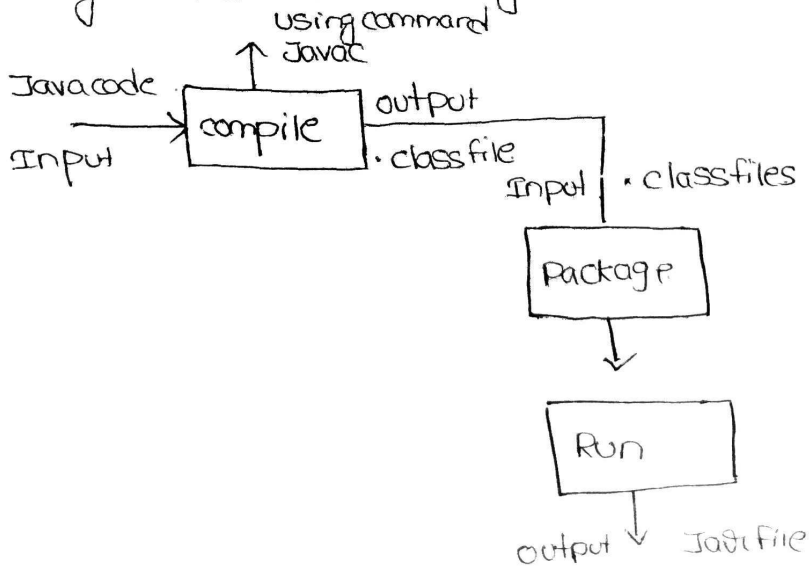
→ Build the code
  ↳ creating the packages

Rules:

→1. Always default name and target name should be same
→2. Always the file name should be build.xml
→3. Always when you open the <target><project> you have close that </target></project>

what is meant by build.xml

A) totally build.xml is 3 types



Default: The purpose of the default from where the execution will start

manifest: The purpose of manifest file is to indicate the main class in Java

→ manifest having 2 attributes

1) main class name

2) attribute value.

SRC directory:- The purpose of SRC directory is to indicate the source code.

dest directory: To indicate the destination directory

Base directory: The purpose of base directory to indicate the source code

.class files: It is nothing but after compaling the Java code we will get the output as .class files

Jaar files: It is nothing but collection of .class files we will call as jaar files.

create one folder in desktop name as
↓ [Eg: antmainprogram]

go to Notepad++

↓

go to file

↓

New

↓

Enter the build.xml code

↓

save as

↓

Seleck desktop File name as build.xml

↓

save

↓

go to google

copy the java code

↓

Paste the notepad

↓

Save as

↓

select folder → classname.java [like Eg. Hellowoorld.java]

↓

save.

↓

create subfolder [SRC name] in floder [Antmain Paogram]

↓

move the classname.java [Hellowoorld.java] in SRC folder

↓

open cmd in folder [Antmainpaogram)

↓

Run the command [: ant]

↓

BUILD Successful

Jenkins

Jenkins is a open source automation server which can be used to automate all of tasks such as building, testing, and deploying software.

winscp:-

winscp is for file transfer i.e secure file transfer between a local and a remote compute [using FTP, SFTP]

FTP :- File transfer protocol

SFTP : SSH file transfer protocol [SSH - secure shell]

`Java -Jar Jenkins.war` → command to start Jenkins

→ After executing this command, we should not do anything in putty, if we want to do any thing open duplicate session.

Configure Java

open configure Java in windows search & in security

Press medium → Ok

Imp

can you explain master slave concept [or] build machine configuration [or] node configuration [or] lable configuration

→ After executing this command

Java -Jar Jenkins.war

↓

open browser and search

192.168.33.10 : 8080

↓

open Jenkins

↓

manage Jenkins

↓

manage nodes

↓

New node

↓

Enter node name [Here we need to give node name Eg: node1]

↓

Select ⊙ Dumb Slave → [It is used whenever a foresh node created]

↓ ○ copy existing node → [copy from already created node]

or

↓

Enter name [node]

↓ description

↓

# of executors [5]

↓ → It is a home directory for the master on the slave mast

[Go to Desktop → create Folder → takepath → Paste.] ←

Remote FS Root [C:\Users\Gowthami\Desktop\jenkinlog] → Empty folder path

↓

Select Launch method [Launch slave agents via Java Web Start ▽] → [This is for windows]

↓

Save

↓

manage Jenkins

↓

configure System

↓

Add Jdk

↓

Jdk installations

JDK Name [Java_Home]

↓

Java_Home [ ] → Java path

disable the ☐ Install Automatically [unselect]

↓

Add Ant

↓

Ant Installations

Ant Name [Ant_Home]

ANT_HOME [ ]

disable the [ ] Install automatically (unselect) [If we select it will download latest version of ant, it will not support old version of developer we create using code old version]

Save

configure

click on tool Locations

Add Ant

Listof tool Locations

Name (Ant) Ant - Home

Home c:\Users\Admin\Desktop\ANTinstallation\apache-ant-1.9.7bin\apache-ant-1.9.7 → Ant Path

Add Java

Name (JDK) Java-Home

Home c:\Program files\java\jdk1.7.0_79 → Java Path

save.

Launch

Keep

open

click on I accept the risk and want to run this application

Run

configure Java
go to windows
Search configure Java
Security
Select medium
ok

Save code management

click on subversion

Repository URL http://192.168.33.10:8081/subversion/branch → Branch URL

✗ wrong

continuous Integration:

continuous integration is a process in which all development work done by different developers is integrated as easily as possible.

Job creation (or) New item creation

New item

↓

Enter item name [ci-src-branch]

branch or tag or trunk

↓

Select ⊙ Build a free-style software project

↓

ok

↓

Project Name [ci-src-branch]

↓

Description

↓

click on Restrict where this project can be run

↓

Label Expression [node1]

↓

Source code management

⊙ click on Subversion

↓

→ branch URL

Repository URL [http://192.168.33.10:8081/subversion/branch]

↓

Build

Add buit step [Select Invoke Ant]

Ant version [Default]

Targets [ ]

↓

Post Build Actions

↓

Add post build action select [Archive the artifacts]

Files to archive [**/*.jar]

↓

Save

↓

click on ~~Build~~ New ~~history~~

Build Triggers

(1) PollSCM:

* Whenever a particular change occur in svn and after doing checkin. build will done Automatically.
* It is used whenever we need to build automatically if any change occur in svn repository

Select Job
↓
Poll scm → Enable
↓
Enter * * * * *
↓
Save

open folder
↓

(2) Build periodically:

→ Build periodically builds the project pero 'periodically, here we need to give cron tab

Ex: * * * * * → every minute

If we give above expression build will happens four every minute.

(3) Quiet period:

→ To hold the build for particular duration Quiet period is used

→ For example we will give 120 sec then build will happen after 120 sec

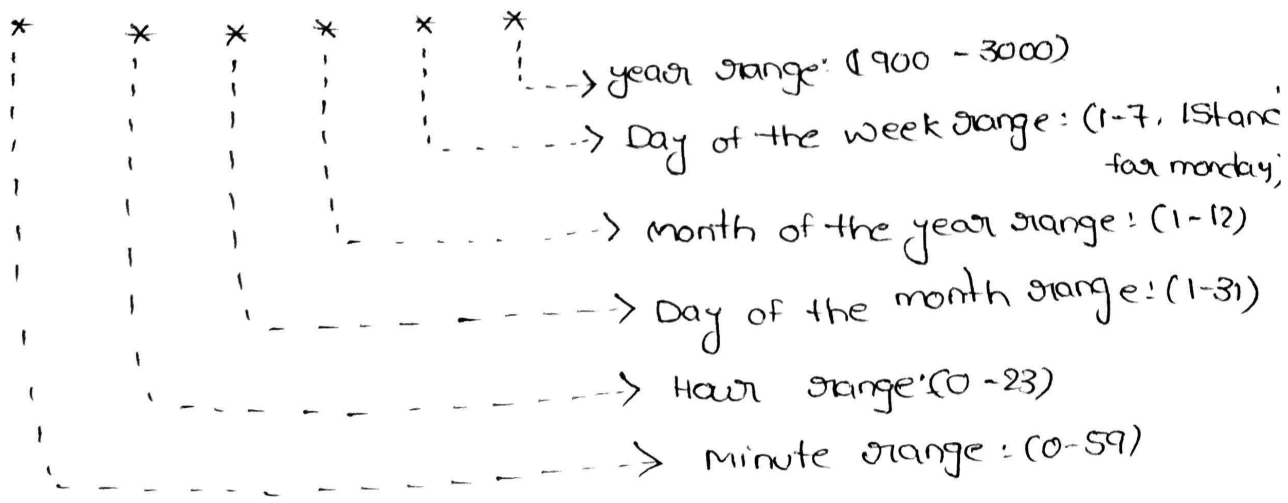
→ It is used in weekly release in real time

→ If we set quiet period as 7am & 300sec then next build start at 7:05 Am

→ Files will build upto 7am then after 7am files start at next trigger i.e 7:05

Crontab:

1st



Build is parameterized:

→ To pass any parameters or any IP address from Jenkins we are using the option build is parameterized

→ There are different parameters avaliable but most of the time we will use 1) String parameter.
2) Test parameter

1) String parameter:

Here string name should pass

| Name : test
| Default : test
| Description : Job name

2. Test parameters

Here IP Address should pass
(or)
Deploy condition should pass

Jenkins
↓
ci-src-tag
↓
configuration
↓
☑ This build is parameterized
↓
Save.

| Name : IP Address
| default : 192.168.33.10, 192.168.33.11
| Description : font size: 2, seperated by comma

| Name : Deploy
| default : yes
| Description : font size: 2, separated by comma

check now build now will replace will build with parameters

→ As per requirement we can change ip address it can be changed outside (i.e. in Jenkins tab) or inside (i.e configuration)

Security

complete Jenkins access security

Jenkins access for Particular job.

manage Jenkins

↓

configure global security

↓

☑ Enable security

↓

⊙ project based matrix Authorization Strategy

↓

user/group to add : gowthami Add

↓

It will come in red colour as we didn't added http

Plugins:

→ Jenkin compatable for plugin, it contain some plugin

→ If we need to add extra functionality to existing software plugins are used

Some plugins are:

1. Green ball plugin
2. Embeddable build status
3. RHN PUSH

4. M2 Release plugin
5. Sonar qube
6. Rundeck
7. Multi job configuration
8. configuration matrix
9. Docker
10. Git
11. Pipe line
12. Docker build & publish

How install plugin

open google
↓
Search Green ball plugin
↓
open 1st one → wiki
↓
click on archives
↓
1.15 (version) → check version that is compatiable (or) suitable for our core jenkins version.
↓
will get in right both of jenkins page
↓
download .hpi file.
↓
Now open Jenkins
↓
manage Jenkins
↓
Advanced
↓
manage plugin
↓
upload .hpi file
↓
Restart (if required)

manage plugin

- update → update old versions of Jenkins
- Installed → It will show already installed plugin
- Available → It will show all plugin available in market
- Advanced → If you want to install new plugin the option use

chef:

chef installation:

→ create 3 machines with ip Address 192.168.33.10
192.168.33.11
192.168.33.12

Then follow installation steps in chef installation file

→ In Jenkins we generated .jar, .war etc files, then next step is deployment, to do deployment automatic-ally we will use chef.

→ chef is a configuration management tool.

→ By using code managing Nodes is called chef.

→ chef will support scalability i.e we can increase or (or) decrease nodes

→ chef invented in 2009 → 2013 in usage.

→ In an admin want to install particular package in 10 servers admin needs to login to every server & install. If 10,000 servers it will become hard task to overcome chef is used.

→ chef contains an workstation, there will be communicat-ion between workstation and nodes this is also called bootstrapped.

→ From workstation 10,000 servers are bootstrapped

→ Any changes made in workstation will effect in remaining servers connected

	Launch	started
Puppet -	2005	→ 2008
chef -	2009	→ 2013
Rundeck		
Ansible		

Deployment: Taking packages & move then install

→ For manually installing java we use sudo yum install Java H

Rundeck: Problem is if there is 10 servers

→ upto 4th server build successful & in 4th an issue found, it will stop to build till we kill job

1
2
3
4 — an issue occur
5
6
7

→ Rundeck used for small integration

chef configuration

192.168.33.11 chef.dk

192.168.33.10 chef-server chef-manage

192.168.33.12

chef server: used to do communication blw different nodes

chef manage: for managing complete process chef manage is used.

chef dk: It is workstation.

chef-Installation:

Step1:

* First check 1gb memory available (or) not

command `free -m`

Step2:

* If not increase, for increase go to vagrant text file remove comment for

```
config.vm.provider "virtual box" do |vb|
 vb.memory = "1024"
 end.
```

* Every server should have hostname
* check it using the command hostname. it will give localhost.local change it to to

chef.mycompany.com (shown in step4)

This is our wish, but it should be same in every server otherwise connectivity fails

step4:

copy chefserver from local using winscp

step5: → vi /etc/hosts

↳ Go to insertmode & type 192.168.33.10 chef.mycompany.com

save & quit

step6: Go to path vi /etc/sysconfig/network & change the file

HOSTNAME = chef.mycompany.com

step7: sudo reboot

step8: After this open browser & give 192.168.33.10 we will login as chef.

step9: Before we copied chef-server & chef-manage so now extract using the command

rpm -ivH package name

sudo rpm -ivh chef-server-core-12.3.0-1.el6.x86_64.rpm

step10: Then reconfigure sudo chef-server-ctl reconfigure

step11: Now extract chef-manage-2.2.0-1.el6.x86_64.rpm

sudo chef-manage-ctl reconfigure.

Step 12

As we already did browsing 192.168.33.10, it will ask userId & password so we need to configure for that run command → chef always use pem file & pemformat is

chef create username firstname lastname emaild. password --filename /home/vagrant/user.pem

It will create automatically like

Eg:-

chef-server-ctl user- create ramanji123 vempalli ramanjireddy ramanji.devops@gmail.com Ramanji123 --file name/home/vagrant/user.pe

Red colour → password Ramanji123

Green colour → username ramanji123

ramanji.devops@gmail.com → Email id

Vempalli → firstname

RamanjiReddy → Lastname

↓

once setup organization

↓

It will download zip file

↓

extract zip file so we will get chef repo then copy to 11th machine

↓

After download using winscp copy to 192.168.33.11

↓

Extract dk

↓

check chef --version

If we want to deploy from 11th machine to 12th machine server is required i.e 10th machine was server. so to establish connection b/w them we change hostname → as shown above.

↓

Now copy certificates from 10th machine to 11th machine Path is shown below open 10th machine in putty

* go to Path

↓

cd /var/opt/opscode/nginx/ca → path ssl

* gives → ls here we will get chef.mycompany.com.crt this is the certificate need to copy to 11th machine

what is meant by certificate?

→ certificates are used for authentication purpose of any website.

→ consider an example.
open google.com in internet explorer there we will see 🔒 symbol i.e certificates

→ If we need to access particular site we have to copy this certificates into backend server then only these is communication b/w server & backend.

→ In this only we will mention valid data, before the end of valid date they need to extend to work otherwise website will fail.

→ so chef server also will have some certificates, these certificates need to copy to development kit so that only we can access server (or) communicate b/w server & workstation

for copying we need to create folder for this go to path

cd /chef-repo/.chef

↳ After this give ls -lart it will show hidden files (.chef) go into .chef

↓

mkdir trusted_certs (In .chef folder)

↓

In 10th machine

↓

scp chef.mycompany.com.crt vagrant@192.168.33.11:

username IP Add ↳ home location in 192.168.33.11 11th machine

↓

Secure copy

↓

Go to 11th machine & move file & copied in above

mv chef.mycompany.com.crt /home/vagrant/chef-repo/.chef/trusted_cert

↓

go to trusted_certs location & check using ls whether chef.mycompany.com.crt is move or not

↓

knife SSL check → command to check certificates verified or not

↓

go to .chef & check folder

Bootstrapping: Establishing connection & install package in required server

Recipes: performing the action single action

cookbook: collection of actions or recipes

→ Here files will be saved in .rb format ruby

12th server

open 12th server

login : vagrant
↓
PWD : vagrant
↓
sudo su
↓
service httpd status

→ we will get unrecognized service → means there is no httpd package

→ usually to install package we will use sudo yum install httpd NO I don't want to install manually i want to install from chef server for this [open browser 192.168.33.10 we will get this site not working]

→ Deploying httpd (apache) from chef server to 12th server

we need to write recipe

11th machine

go to 11th machine
↓
ls
login : vagrant
Pwd : vagrant
↓
sudo su
↓
ls
↓
cd chef-repo [go to inside chef-repo we will get on file cookbooks]
↓
ls
↓
cd cookbooks
↓
ls

chef generate cookbook LeadAPP

↓

ls [LeadApp will come]

↓

cd LeadApp

↓

ls

↓

cd recipes

↓

ls

↓

vi default.rb → Here we need to write some code. press enter 5 times & wri

```
package 'httpd'
service 'httpd' do
        action [:enable, :start]
end
:wq!
```

↓

chef-client --local-mode -runlist 'recipe [LeadAPP]'

↓

cd ../..

↓

ls

For uploading cookbook default (LeadAPP)

knife cookbook upload LeadAPP

↓

cd ..

↓

knife bootstrap 192.168.33.12 --ssh-user vagrant --ssh-password 'vagrant' --sudo --use-sudo-password --node-name node1 --run-list 'recipe [LeadAPP]'

↓

go to browser [enter 192.168.33.10]

↓

go to chefserver [enter username 'kk']

↓

go to policy

↓

lead pol

Next goto 12th server run the command

sudo service httpd status.

File structure:

create → cookbook

↓

run → List

↓

upload ← upload cookbook to chef server

↓

Bootstrap

After uploading to chef server then only it will bootstrap to server

what ever file written in .rb compile then execute.

If anything wrong it will stop there only

192.168.33.11

workstation
[opt]

dk [development kit]

→ same host name provided here

1) Hostname → once we have to change the hostname.

↓

reboot

↓

rpm -ivh dk

↓

chef --version

192.168.33.10

server manage. [→ Browser → username → PWD]

It is a central server.

1) First to check 1gb memory available [opt] not using command free -m [opt] free -H

2) HostName

↳ vi /etc/hosts

→ 192.168.33.10 chef.mycompany.com

→ vi /etc/sysconfig/Network

→ HostName

→ reboot

→ rpm -ivh package name. [server]
re configure

→ rpm -ivh packagename [manage]
re configure

Pem: chef-server-ctl user-username firstname middlename emailid password -filename /home/vagrant/user.pem

→ Login ↳ organization ↳ repository ↳ starter kit ↓ repo

192.168.33.12.

Application [opt] client [opt] Node.

→ same hostname provided here.

Build pipeline plugin

install

Build pipeline plugin

→ First open the 192.168.33.11 machine

↓

install the subversion [svn]

↓

install the Jenkins new version 1.658 [Past copy and paste the winscp 11th machine.]

↓

install the Java 1.7 [sudo yum install javapa version]

↓

Inside the ~~repository~~ create

↓

create new repository name it as the pipeline

↓

inside the repositary create trunk, branch tag

↓

inside the trunk copy the helloworld code [Build.xml src]

↓

inside the Branch copy the factorial

↓

inside the tag copy the enhancement

↓

create node name it as the node1

↓

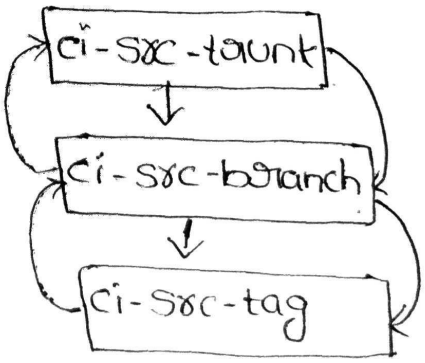
create 3 Jobs CI-SRC-Trunk, CI-SRC-Branch, CI-SRC-Tag } node1 [Label expression]

Install build pipeline:

Go to Jenkins
↓
manage Jenkins
↓
manage plugin
↓
click on Available
↓
search by typing Build Pipeline
↓
install without restart

Pipeline

create 3 Jobs
↓
configure same node1 for all 3 [individual also we can g
↓
for ci-src-trunk
downstream is ci-src-branch
and no upstream



for ci-src-branch
upstream → ci-src-trunk
downstream → ci-src-tag

for ci-src-tag
upstream → ci-src-branch
down stream → No downstream
↓
To do this go to Particular Job
↓
configure
↓

ADD post-build action → select trigger parameterized build on other projects

Projects to build : ci-src-tag [or] branch [or] trunk

Trigger when build is : stable.

Trigger build without parameters : [✓] → save

Now goto Jenkins

↓

Here we will get [+] symbol ~~present~~ press on '+'

[All] [+] enter view name [my pipeline] click

click on ◉ Build Pipeline View

ci-src-trunk

ci-src-tag

ci-src-Branch

↓

give name my pipeline

↓

select initial Job [ci-src-trunk]

↓

Number of Displayed builds : [100]

↳ our wish, it is just build history

↓

OK

↓

Run.

→ IF we need to do automatically we will use build triggers Like pollscm

→ If we configure pollscm in trunk & if any change occur build pipeline will start automatically.

→ Go to maven folder and copy the location (i.e D:\newversions\mavenpractice\apache-maven-3.3.9\bin)

Paste the path in environment variable Right click on on my computer

↳ Go to properties
↓
→ click on Advanced system settings
↓
→ In Advance click on environment variables
↓
In system variables
↓
Click on path and select edit
↓
variable name : Path.
variable value : ↓ Here go to the end of the line and give semicolon(;) and paste the maven path
↓
click on OK
↓
In user variables click on new.
↓
Give variable name : JAVA_HOME
variable value : C:\Program Files\Java\jdk1.7.0_79
↓
click on OK
↓
Again in User variables click on new
↓
variable Name : M2_Home
variable value : D:\newversion\maven-practice\apache-maven-3.3.9
↓
click on OK
↓
click on OK
↓
click on Apply and OK

create one folder on Desktop and name it as maven

Go to 10th machine and create a new repository as maven

↓

cd subversion-1.9.4-1/subversion/bin

↓

sh svnadmin create /home/vagrant/maven

↓

cd ../..

↓

cd apache2/conf

↓

vi httpd.conf

↓

change the repository name here

:wq!

↓

cd ../..

↓

sh ctlscript.sh restart:

↓

Go to browser and enter 192.168.33.10:8081

↓

click on Accesssubversion

↓

copy the URL (192.168.33.10:8081/subversion/)

↓

Go to maven folder in Desktop

↓

open command prompt here

↓

svn co <URL>

↓

cd subversion

Now go to my app folder in maven-practice folder

copy the folder and files like src

target

pom

And paste it in side subversion folder of maven folder

in Desktop

go to command prompt

```
svn st      ? src
  ↓         ? target
            ? pom
svn Add src target pom
  ↓
svn ci -m "message"
```

Now go to browser and refresh the page. Now you will see
- pom.xml
- src
- target

Go to 10th machine start Jenkins

```
java -jar Jenkins.war
  ↓
Go to browser [192.168.33.10:8080)
  ↓
click on manage Jenkins
  ↓
click on configure system
  ↓
In maven · click on Add maven
  ↓
Name : m2-home
uncheck ☐ Install Automatically
  ↓
click on Save
  ↓
click on Node1
  ↓
click on configure
  ↓
In node Properties ☑ Tool location
  ↓
click on Add
```

Name: select m2 home
Home: Give the maven path.

[D:\revisions\maven-practice\apache-maven-3.3.9)

↓

click on save

Go to Jenkins

↓

click on new item

Item Name : ci-maven-myapp

↓

Select ⊙ Build a maven 2/3 Project

↓

click on OK

↓

click on ☑ Restrict where the project can be run

↓

Label Expression : node1

↓

In source code management select ⊙ subversion

↓

Repository URL : http://192.168.33.10:8081/subversion/

↓

In pre steps

click on Add pre build step

↓

Select Invoke top-level maven targets

maven version : select M2-home

Goals : Package

In Post build Actions

click on Add Post-build Action

↓

Select Archive the .artifacts

↓

Files to archive : **/*.jar

↓

Click on save

↓

Click on Build Now

↓

Success

NEXUS Installation in windows machine:

Before going to install nexus repository we have to set maven path and JDK path in environment variables.

* Download nexus from
  https://www.sonatype.com/download-oss-sonatype
* Nexus Repository manager oss 2.x
  click on
  All platforms Nexus Repository Manager oss 2.x-bundle.tar.gz
  file will be downloaded.
* Unzip or Extract the file, you will get 2 folders
  i.e nexus-2.14.5-02 and sonatype-work.
  Copy the folders to the below location
  c:\programFiles (x86)
* Click on Start button and search for command prompt and run it as administrator.

c:\windows\System32>cd c:\program Files (x86)\nexus-2.14.5-02\nexus\jsw\windows-x86-64

windows-x86-64 > install-nexus.bat

o/p: installed nexus.

* Open browser and type http://localhost:8081/nexus
  you will get an error like
  "This site can't be reached"
* you have to start nexus service in your local machine cli on Start button and search for services.
* Select nexus and start or restart the service. Now go to browser and refresh the page.
* Now you will get Nexus Repository Manage oss page.

* Click on login option by using below credentials.

username : admin
password : admin123

After login as admin, you will get with various options.

* In the left hand side click on repositories

Click on + Add

Click on Hosted Repository

Repository Id : ARTECH
Repository name : charan
Repository Policy : Select Release

Click on save.

* Create one more repository

Click on + Add

Click on Hosted Repository

Repository Id : ARTECH1
Repository name : charan1
Repository Policy : Select Snapshot

Click on Save

* After creating the repositories click on the created repo link, you will prompt to another page.

* Go to C-Drive location

C:\Users\charan\.m2.

→ In this location create a new file called settings.xml and save it as .xml.

* Open the file and write the below code.

```
<settings>
<servers>
        <server>
            <id> ARTECH </id>
            <username> admin </username>
            <password> admin123 </password>
        </server>
```

```
<Server>
    <id> ARTECH </id>
    <username> admin </username>
    <password> admin123 </password>
  </server>
</servers>
</settings>
```

* Create pom.xml file and write the code [The code will be there in the given document].
* Do the required changes in pom.xml file.
* Copy Release repository and Snapshot repository URL's along with their ID's from the nexus and paste them in pom.xml file.
* Create new repository in 11th machine and restart it. Go to browser and type 192.168.33.11:8081.
* Create one folder in Desktop and open command prompt here as checkout the URL.
* In that folder copy the new pom.xml file and copy src, tag folders from maven practice folder and paste it in the created folder. Add the files and folders in command prompt and commit it.
* Create a job in jenkins in 11th machine.

click on jenkins
↓
click on new item
↓
Item name: ci-maven-nexus
↓
Select ⊙ maven Project
↓
click on ok
↓
click on ☑ Restrict where this project can be run
↓
Label Expression: node1

* In Source Code Management
  click on ⊙ SubVersion
  ↓
  Repository URL: http://192.168.33.11:8080/svnrepo

* In post steps
  click on Add post-build Step
  Select Invoke top level Maven targets

* Maven Version: m2-home
  Goals: clean install deploy
  In post-build Actions
  click on Add post-build action
  Select Archive the artifacts
  files to archive: ***.jar
  click on Save
  click on Build now.

O/p: Success.

Difference between Ant and maven.

ANT	maven
1. We will write build.xml	1. We will write POM.xml
2. By using Ant we can build the code	2. By using maven we can build
3. Backup is not possible.	3. Backup is possible (nexus)
4. Not possible to reuse the code.	4. We can reuse the code.
5. No life cycle.	5. It has life cycle.
6. No plugin	6. It support plugin [M2-Release plugin]
7. Ant doesn't has formal convention so we need to provide information of the project structure in build.xml file.	7. maven has convention to place source code, compile code etc so we don't need to provide information about the project structure in pom.xml file
8. ANT is procedural, you need to provide information about what to do and when to do through code. you need to provide order	8. Maven is declarative, everything you define in the pom.xml file

Pom.xml project object model is an xml file which contains information about the project and configuration details used to build a project with maven along with its dependences

→ pom also contains the goals and plugins. While executing a task, maven look for the pom in the current directory. It reads the pom, gets the needed configuration information and then executes the goal. some of the configuration that can be specified in the pom are.

- project dependencies
- Plugins
- goals
- build profiles
- Project versions
- developers
- mailing.

Pom.xml structure

① < project > (Parent path)

- G → Group ID → is path
- A → Artifact ID → particular folder
- V → version → Always versioning mention in snapshot

</project

· Like 1.1 snapshot
1.2 snapshot

② < scm >
192.168.33.10:8080/subversion
</scm>

③ <Distribution Management> → we want to specify 3rd party repository we will use distribution management

<Nexus URL>

</Distribution Management>

Release:

Release is moving package to Redhat satelite & Deploying in required server

Repositories maven have 3 different Repositories

1) Local Repository
2) M2 Repository [or] central Repository [or] .M2 folder
3) Remote Repository [or] Backup Repository [or] artifactory [or] Third party Repository [nexus]

→ maven searches for the dependencies in the following order

→ Local repository then central repository then Remote Repository.

Local Repository.

→ It is a folder location on our machine. It gets created when you run maven command for the first time.

→ When we run a maven build, the maven automatically downloads all the dependency jars into local repository It helps to avoid references to dependencies stored on remote machine every time a project is build

→ maven local repository by default get created in users home directory

central repository: maven central repository provided by the maven community It contains a large number of commonly used libraries

→ When maven does not find any dependency in local repository it starts searching in central repository

Third party repository:

→ Also called as backup repository

→ i.e Nexus

→ To get 3rd party repository we need to install plug in.

→ maven does not find a mentioned dependency in central repository as well. It will then search in 3rd party repository. maven will download dependency from remote Repositories mentioned in the same pom.xml

* some times developers will write dependencies

* Whenever developer gives pom.xml

we need to check (1) source code management tag

↓

i.e scm url they should mention

How to check is copy URL from ticketing tool search in pom.xml it should reflect

(2) version → current version

Next development version

(iii) snapshot version

If any version mismatch & developer req - est to modify by us only then check out pom.xml modify & chec it

Snapshot :-

→ snapshot is a backup of any version

→ snapshot is created in 2 condition 1) Before upgrade

2) After upgrade

→ Indicated in -SNAPSHOT.

→ If current release version is 1.0 then next development version is 1.1

→ If current release version is 1.1 then next developme -nt version is 1.2

12 Release plugin :- Instal perform maven release plug in

↳ used to release packages

How I release is Once M2 release plug in then we will get,

↳ perform maven release we will get login Password

* How to Check current release version & next development version is by using perform maven release option

* If I release 1.0 in today. next development version is 1.1 [i.e Pom.xml]

* Always Jenkins release version < maven version

↓ ↓

This is current release version This is next development i.e in Pom.xml

Daily release:

Release Request:

1. Build URL : Not applicable
2. SCM URL : 192.168.33.10:8080
3. If branch required - branch name : Not required
4. Release version : 1.0
5. Next Development version : 1.1
6. Artifacts Required in the release : src-trunk-1.0-SNAPSHOT 20160720171217.no arch.rpm
7. Do you need Jenkins instance after release ? Yes.

AS we know that Pom.xml is reused only snapshot version & URL will changes in most of the case

Daily release → NO dependencies

Weekly release → only 1 dependencies

Monthly release → More dependencies

8. What is / are the RHN channel label as for the artifacts

Promotion post release

ceh - Bang - Test
ceh - Hyd - Test
ceh - che - Test
ceh - Mum - Test
ceh - NCR - Test

Daily release: Basically if developer wants to add small code to application then they will rise daily release request the duration daily release is 10-15 minutes

Weekly release: weekly release having the dependency job according to the developer requirement we will rise the. Weekly release request basically duration of weekly release is 5-7 hours.

Release Request:

1. Build URL:
   192.168.33.10:8080/Jenkins/Job/ci-src-trunk
   Helloworld.Java-2-6.81-199.jar
   192.168.33.10:8080/Jenkins/Job/ci-src-branch
   Factorial.java-1.4.2.22-92.jar
2. SCM URL
   192.168.33.10:8081/subversion/trunk
   192.168.33.10:8081/subversion/Branch
3. If Branch required - branch name: NA
4. Release version: 81
5. Next Development version: 82
6. Artifacts required in the release
   Helloworld.java.2.6.81-199.jar
   Factorial.java-1.4.2.22-92.jar

8. What is / are the RHN channel label as for the artifacts

Promotion post release

ceh - Bang - Test
ceh - Hyd - Test
ceh - che - Test
ceh - Mum - Test
ceh - NCR - Test

Daily release: Basically if developer wants to add small code to application then they will rise daily release request the duration daily release is 10-15 minutes

Weekly release: weekly release having the dependency job according to the developer requirement we will rise the. Weekly release request basically duration of weekly release is 5-7 hours.

Release Request:

1. Build URL:
   192.168.33.10:8080/Jenkins/Job/ci-src-trunk
   Helloworld.Java-2.6.81-199.jar
   192.168.33.10:8080/Jenkins/Job/ci-src-branch
   Factorial.java-1.4.2.22-92.jar
2. SCM URL
   192.168.33.10:8081/subversion/trunk
   192.168.33.10:8081/subversion/Branch
3. If Branch required - branch name: NA
4. Release version: 81
5. Next Development version: 82
6. Artifacts required on the release.
   Helloworld.java.2.6.81-199.jar
   Factorial.java-1.4.2.22-92.jar

7. Do you need Jenkins instance after the release : NA

8. What is/are the RHN channel label(s) for the artifacts promotions post release?

ceh-32-bang-test

9. Release specific Requirements NA

SVN URL	Build URL	current release version	Next development version	Release state	duration
192.168.33.10:8081/subversion/Trunk	192.168.33.10:8080/Jenkins/Job/CI-SRC-Trunk	1.0	1.1	yes	2days
192.168.33.10:8081/subversion/branch	192.168.33.10:8080/Jenkins/Job/CI-SRC-branch	1.2	1.3	NO	2days
192.168.33.10:8081/subversion/Tag	192.168.33.10:8080/Jenkins/Job/CI-SRC-Tag	1.4	1.5	NO	2days.
192.168.33.10:8081/subversion/dev		1.6	1.7	yes	"
192.168.33.10:8081/subversion/dev1		1.8	1.9	yes	"
192.168.33.10:8081/subversion/dev2		2.0	2.1	NO	"

Monthly release: monthly release we will perform on production environment also

Basically in my project we are maintaining the release management sheet in that sheet develop will mention all dependency job nearly 20-30 Jobs in that sheet only developer will mention what and all jobs we have to release in current release.

In that sheet only developer will mention the jenkins url and also svn url and also svn revision number and build number and package ID the duration of monthly release is 2-3 days.

Monthly Release:

Release process is same but dependencies different, we are following the release structure.

$R_2 \cdot 5 \cdot (10)$

This is standard ↵ ↳ It will change for every release

$R_2 \cdot 5 \cdot 10$

$R_2 \cdot 5 \cdot 11$

$R_2 \cdot 5 \cdot 12$

$R_2 \cdot 5 \cdot 13$

$R_2 \cdot 5 \cdot 13 \#1$

↓

In 13th version it had bug so we have released another version

Dependency: NO → no need to release

yes → If they mention yes we need to release.

How to release

Developers will give sheet
↓
Perform maven release option
↓
check the version whether they mention properly or not
Here current release version 54 [once we release]
Next development version is 65 and then file the check-htals and then build

→ In ticket only they will give jenkins URL & SVN URL

Ant Life cycle:

compile
↓
Package
↓
Run

MAVEN Life cycle

validate
compile
test
Package
verify
Install
deploy

maven support to rundeck for deployment

validate: validate the project is correct and all necessary information is available.

compile: compile the source code of the project.

test: test the compiled code and package it in its distributable form

test: test the compiled code using a suitable unit testing frame work 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

Verify: run any checks on results of integration tests to ensure quality criteria are met

Install: Install the package into the local repository, for use. a dependency in other project locally

deploy: done in the build Environment, copies the final Package to the remote repository for sharing with other developers and projects

Nexus

Nexus version is 2.7.0

Nexus a open source project called proximity

→ First release made on 2006

→ Nexus used for back up of packages

→ Nexus is a third party repository

* How to upload packages to nexus repository:

↳ It has 2 ways 1. Manual upload.
2. Automation upload.

manual upload

first developer raise a request and they will give

Group ID
Artifact ID
Versioning
} parameter.

and also zip file

Hi.

could you please upload a zip file located in the link below for order status to nexus and provide us with the path

URL: - - - - -

The code has been uploaded in the following SVN repository for order status

URL : - - - - - -

Group ID
Artifact ID
versioning

→ First download zipfile from link

open Nexus
↓
Artifacts upload
↓
G
A
V } give parameters take from above

some times they will not give G A V parameters and they will give oldversion 1.2.0 & replace new 1.2.1 so open 1.2.0 by search & take GAV & do

Packaging : select zip
if zip is not there just type zip
↓
select Artifacts to upload
↓
select downloaded package

ADD artifacts
↓
upload

How to check file is uploaded or not:

Browse storage
↓
they will give groupID just copy & search

Copy if they gave com.specs.sis

Search: [ ]

Artifact ID is folder
↳ click on folder → open version
[or]

search folder by folder like they will give com.specs.sis
com → specs → sis

Automation upload.

How you will configure nexus in pom.xml:

```
</distribution management>
 < repository >
  <Id > release - profile - </id>
  < name > Internal Release Repository </Name>
   <url > http: // ---- release </url >
   </Repository >
   <snapshot repository>
   <id > Snapshot - profile </id >
    <name > Internal snapshot repository </names>
    <url > http: // ------ </url>
     </ snapshot repository>
     </distribution management>
```

→ configure URL in Jenkins & Perform maven release

Install distribution management plugin in Jenkins

Download git 2.13.0 for windows and install it

In the Installation steps

Select ⊙ use Git from Git Bash only

Click on next & Install it

open https://github.com/ link from browser

create an account with your details and verify email address

Click on start project

↓

Give a repository name click on create repository

↓

you will get a page like this

<> code issues pull requests projects wiki Insights settings

Quick setup

Setup in Desktop or HTTPS SSH: https://github.com/~~Polam~~Polamgowthami/devops.git

↓ username ↓ Repository name

open that folder and right click select Git bash here then it opens Git command prompt

$ mkdir gitpractice

↓

$ cd gitpractice

we have to checkout the repository

$ git clone https://github.com/Polamgowthami/devops.git

↓ copied this URL from Github

o/p cloning into git ...

warning: you appear to have cloned an empty repositary

$ cd git

we have to copy src folder and build.xml file and paste inside the git folder

↓

[gitpractice\git → Inside here you have to paste

In Git command prompt

$ git status

o/p on branch master

Initial commit

untracked files:

[use "git add <files> ..." to include in ...)

build.xml

src/

nothing added to commit but untracked files present

[use "git add" to track)

$ git add src build.xml

$ git commit -m "add"

*** please tell me who you are

Run

git config --global user.email "you@example.com"

git config --global user.name "your name"

to set your account's default identity

omit --global to set the identity only in this repositary

Fatal: unable to auto-detect email address (got ...)

$ git config --global user.email gowthamireddy499@gmail.com

$ git config --global user.name polamgowthami

Now push the code to server

$ git add commit -m "add"

$ git push origin master

Here you will o

Now refresh the browser. src and build.xml files reflects into the server

click on clone or download and copy the URL

In windows machine. (or) local go to the Jenkins.war file location and open command prompt

java -jar jenkins.war

Goto browser localhost:8080

Jenkins page will display

Install git plugin.

go to configure Java path give the Java path and git path and Ant path

click on manage jenkins

click on configure system

tool locations

JDK

click on JDK Installation

Name : Java-Home

Java-Home : C:\Program files\Java\Jdk1.7.0_79

uncheck ☐ Install Automatically

Git

Name : git_Home

Path to Git executable : C:\Program files\Git\bin\git.exe

Ant

Name : ant_Home.

ANT_HOME : D:\new versions\ANT Installation\apache-ant-1.9.7-bin\apache-ant-1.9.7

uncheck ☐ Install Automatically

click on Save.

↓

create a new Job

↓

click on new item

↓

item name : git_sample-job

↓

Select ⊙ free style project

↓

click on ok

In source code management

select ⊙ Git

Repository url : https://github.com/palamgowtham/devops-git

In Build

click Add build step and select Invoke Ant

Ant_version : select ant-home.

In post build actions

click Add post-build action and select archive the artifacts

files to archive : **/*.jar

↓

click on save

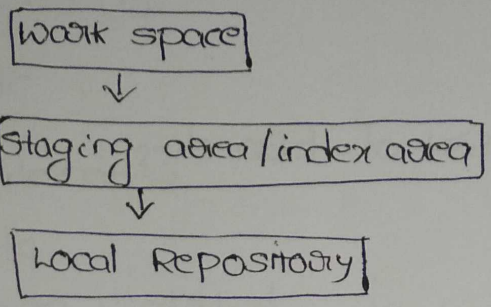
↓

click on Build Now.

Difference between SVN and ~~ANT~~ GIT

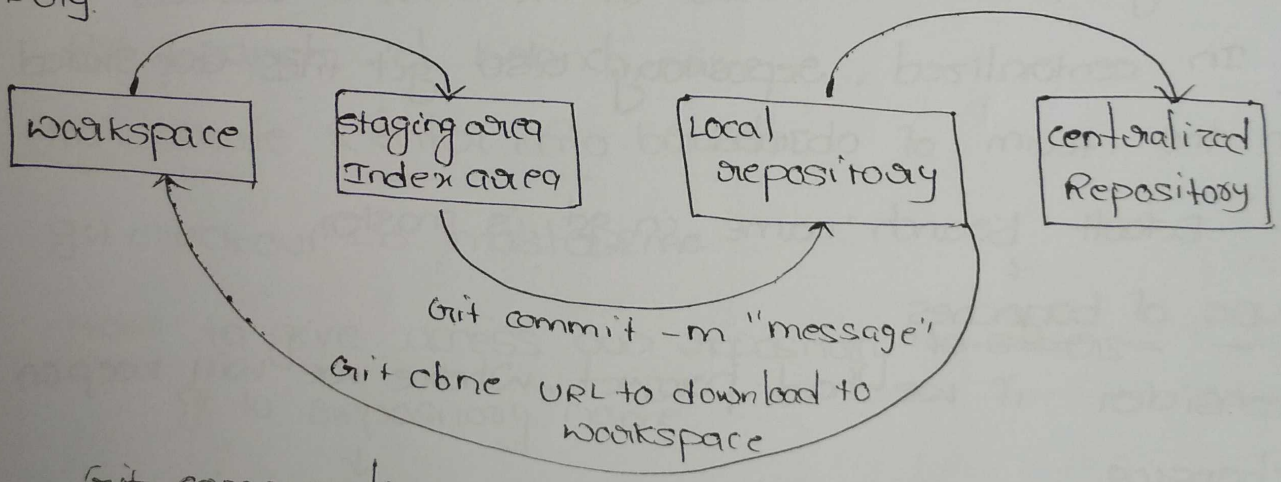
SVN	GIT ~~ANT~~
1) SVN is a centralized repository we are directly involved in the centralized repository	1) GIT is a distributed repository Here first we are working on laptop & then transfered to centralized repository
2) As we are connected directly to centralized repository, if any network issue (or) N/W failure happens then we can't work	2) Git has 3 phases 1) work space 2) staging area / index area 3) Local repository
3) Developers directly interact with centralized repository can create some issues	
) There is no reviews direct check in	

Git



* Here we develop the code in work space then we need to transfer to staging / Index area then to local repository.

* All these stages in our local machine only after these stages we need to push to centralized reposito -ry.



Git commands

1. Git init → for initializing git. after this it will show master. When we install git it will automatically create 1 branch. i.e master
2. git clone URL
3. git push origin master
4. git commit -m "message"
5. git log → to check committed files Ex: git log -n 10 to see last 10 comments
6. git pull → It will like svn update.

7) git add

8. git status

9. git branch → gives list of branches

use git status

If the file is red colour → means it is in workspace

Eg: vi test

write some code & save.

check git status

If the file is in green colour → file is in index area

* In git files are stored in the form of objects
* In centralized repository also git files are stored in the form of objects
* Default branch name in git is master

use of branches

consider if we had project, where we will keep on changing

git branch master1

creating new branch

How to change branch

git checkout (branch name)

Merge:

consider if we had 2 branch

master
audio 1.0

master1
audio 1.0
video 1.0

↓

Here do git merge master1

so we will git video 1.0 also in master

Launch file

(i) cat > file name.
add data
then press ctrl+d
save.

(ii) cat >> file name
add data
ctrl+d
add data to above line we added using > file name

TO remove branch

git branch -d branch name

TO create & enter into branch

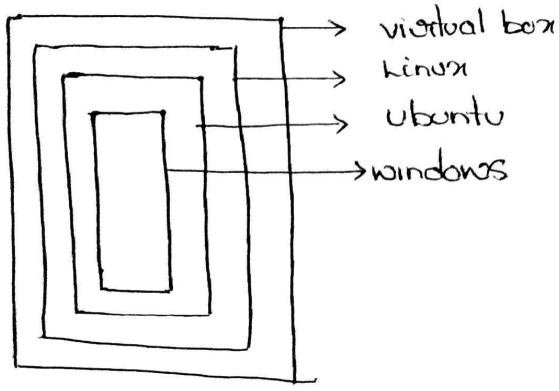
git checkout -b master-2nd

* How to give access our repository to others

go to repository name
↓
settings
↓
colaborations
↓
give user name of required person
↓
ADD colaborators
↓
A verification email will send to us
after verify you can push

Problems before docker

1. There is a developer who developed an application that works fine in his own environment but when it reach production, it will get some issues, this is because of difference in computing environment b/w development & production.
2. virtual machine / virtual box used to operate multiple os.



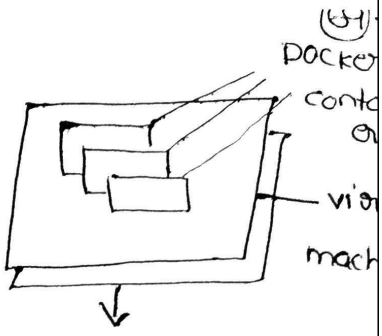
consider an application which is broken into also called as micro services smaller services. because smaller services are each to build & maintain & also if any errors occur it will not reflect whole app.

→ There micro services are using 3 vms where we need to allocate RAM, Disk space to each and every machine, suppose if one machine is serving & other 2 are sleep state at this time we will get disadvantage in resource allocation i.e waste of ram & diskspace.

* Let us see how docker solve this problem

There will be one host machine on the top of the host machine there will be virtual machine and on the top

of vm docker containers are exist



→ Docker containers are light Light weight alternatives of VM.

→ In docker containers are ↳ we dont need to preall -cate any ram or diskspace. It will take the ram & diskspace according to the requirement of applicaton

→ Why

AR TECHNOLOGIES

DEVOPS

Introduction: Overview of Build and Release process

Version control system SVN (Subversion)

1. Installation of SVN
2. How to create SVN Repository
3. SVN commands
4. How to create Trunk,branch,Tags
5. SVN Hooks
6. Branching strategy
7. SVN access
8. SVN Merging
9. SVN workflow
10. SVN backup

ANT: Another Neat Tool

11. Installation
12. Overview of ANT
13. ANT commands
14. Simple build.xml
15. Build.xml for creating, deleting directory
16. Build.xml for deleting and copying and Moving the directory
17. Ant properties
18. Build.xml for Jar file creation

Jenkins

19. Jenkins Installation
20. Jenkins overview
21. How we will configure the SVN and ANT in Jenkins
22. Crating the Jenkins Job
23. How to copy the Job in Jenkins
24. How to give the access to new user in Jenkins
25. How to access to user for particular user
26. Manage Jenkins
27. Manage Nodes
28. Manage plugins

29. How to check the logs in Jenkins
30. Rundeck Configuration in Jenkins
31. Sonar Configuration in Jenkins

Nexus Repository

32. How to upload artifact to nexus
33. How to upload zip ,jar, war file to nexus
34. How to delete the artifact in nexus
35. How to do the dependency release process
36. How to upload dependency artifacts in same request.

Red Hat Server

37. How to upload rpm in to Red hat server
38. Channels in Red hat server
39. How to upload the RPMs into channel using Jenkins
40. How to upload the RPMs force way in to channel

MAVEN

41. Maven Installation
42. Maven Lifecycle
43. Maven Pom.xml
44. Maven Installation in Jenkin
45. Nexus configuration in Nexus
46. Maven Release Process
47. Maven Release types
48. Maven Profile and settings.xml

GIT/GERRIT

49. GIT Installation
50. Repository creation in GIT
51. GIT commands
52. Branches creation
53. Access Management
54. Git workflow

Docker

55. Docker Installation
56. Virtual Machines
57. Docker
58. Basic commands in Docker
59. Container
60. Containerization vs Virtualization

61. Image
62. Registry
63. How we will configure Docker in Jenkins

Chef

64. Chef installation
65. Chef basics
66. Recipe
67. Cookbooks
68. Templates
69. Managing nodes
70. Managing nodes

Service Now ticketing tool

Real Time Work flow

Real time Deployment process

Mock Interview

Resume Preparation