\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Ansible\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ansible configuration file : /etc/ansible/ansible.cfg

Default inventory/hostfiles : /etc/ansible/ansible.hosts

host\_key\_checking = true/false : This will use to set host keys as known hosts.

Gather\_facts:no/yes: This variable is used to disable setup module. by default this module will enabled in ansible yaml file.This variable is in variable section.

Serial: This is a key word to set machines as a batch.

This is as a batch of machines. Taks are apply on batch.

To clubbing the group:

[newgroupname:children]

sub group 1

sub group 2

General Variables for All connections:

|  |
| --- |
| ansible\_host,ansible\_ssh\_key,ansible\_ssh\_pass,ansible\_ssh\_private\_keyfile,ansible\_port |

We can specify all these values in inventory file. for password we can us ansible vulture.

Modules:

ansible webservers -m service -a " name=httpd state=started" : To start the service

ansible webservers -m service -a " name=httpd state=stopped" : T o Stop the service

ansible webservers -m yum -a " name=httpd state=present" : T o Install the service

ansible webservers -m yum -a " name=httpd state=abesnt" : T o uninstall the service

ansible webservers -m yum -a " name=httpd state=latest" : T o update the service

ansible webservers -m copy -a " src=/etc/file.txt dest=/var/folder" : T o Copy the file from one location to another location.

ansible webservers -m file -a " dest=/var/folder mode=read oner=sanjay group =devops " : T o change the file permission and owner ship of the group.

ansible all -m setup --tree /tmp/facts : Setup module will give all information about target machines in the form of jsons.

ansible all -m setup " filter=facter\_\*" : This will displays specific details of all facts .

anisble 192.168.13.34 -m user -a " user=sanjay password=redhat" : This user module will create user with specified password in that machine.

|  |
| --- |
| Play Book stricture:  -----  Host Section: This section describes about host details.  Variables Sections: This section describes all variables using before.  Users:  Tasks:  Templates:  Handlers: |

1) Become =yes : This will describe that play book is running with sudo user.

2) {{ varable name}} : To use the variable values in the middle of the play book we use this pattern. Here varable name is replaced by variable values.

3) -include=playbookname : This will include another play book .

4): This shell module will use to execute shell command.

shell:/user/bin/whoami

register:username

This will register the out put of the above command to further use.

out put of above command will store in username variable.

5)Debug module will use for print messages and variable values while play book execution.

-debug: msg=" inventery hostnames are {{inventory\_hostnames}}"

Prompting for inputs:

While executing plabooks some values required like db password.

**vars\_prompt:**

**-name:"dbpasswd" //ur value will store in dbpassword**

**prompt: " Enter password for data base"**

**Handlers:**

We use handlers using " notify" systax . If task is successful completed this notify will work otherwise it will not work.

from notify control will go to handler section.

Example for handlers:

tasks:

-name: copy the website config file

copy: src=/tmp/httpd.conf dest=/etc/httpd/httpd/httpd.conf

notify:

-Apache Restart

handlers:

-name: Apache Restart

service : name=httpd status=restart

Conditiona execution:

Sometime we want to run tasks in a play book in certion condition.

tasks:

-name: Install apache in cent os

yum : service =httpd state=present

when: ansible\_os\_family=="redhat"

**note: This task will execute when varable values is redhat only**

**Templates:**

**While template modules gets executed it will read the template file and change all the variables to its value and copy the file in target place.**

**tasks:**

**-name: Coping file using template**

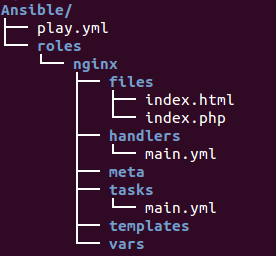
template: src=templates/index.j2 dest={{doc\_root}}/index.html mode=0644

This value come from play book.

To create role:

ansible-galaxy init role\_name

Ansible Role Streacture:



To call a ansible role in a play book:

roles:

-{name:rolaename ,http\_port:8080}

we can over ride the values when calling the role.

Loops:

For repeating tasks we use looping concept.

-name: add severals users

user:

name: {{item}}

state: Andhra

group: Civil

with\_items:

-rest\_user\_1

-rest\_user\_2

------------------------------------------------------------------------------------------------------------------------------------------Dictionary:

network\_interfaces:

- name: enp0s3

configure: true

This red color block is Dictionary

method: dhcp

parameters:

- param: pre-up sleep

val: 2

List of Disctionaryes

- name: enp0s8

configure: true

This blue color Block is Dictionary

method: static

address: 192.168.250.10

netmask: 255.255.255.0

----------------------------------------------------------------------------------------------------------------------------------------

List:

**---**

*# A list of tasty fruits*

**-** Apple

**-** Orange

**-** Strawberry

**-** Mango

**...**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DOCKER

1. **docker images - To see all images in current machine.**
2. **docker pull– To download images.**

docker run –it imagename:version /bin/bash – To run the container from image

1. **ps –elf – To check what process is running in docker container.**
2. **docker ps – To check running containers.**
3. **docker ps –a - To check all containers.**
4. **docker stop/start container name or container id – To stop/start the container.**
5. **docker rm <container name or container id> – To remove stopped container.**
6. **docker inspect <container name or container id> – To get container information in json format.**
7. **docker rmi image name or image id – To remove image.**
8. **docker run –d –-name container name -p 8080:80 imagename:version**

**note: p(caps)----🡪 will automatically assign the port number from host.Small (p) manual**

1. **docker run –p 8080:8080 –v /host machine path: /container path imagename:version – To map thevolumes with host machine.**
2. **docker push username/repositoryname:version**

**doker login --name <name of the user in docker> -- mail id mail id of the account – To login into docker.**

1. **docker tag image\_name username/repository:tag – to tag the image**
2. **docker push username/repository:tag – to upload images to docker hub.**
3. **Docker exec -it containername /bin/bash: To login to runing container:**
4. **Docker cp source destination : To copy the files from host machine to container.**
5. **Docker run -d --name=db redis:latest**

**Docker file instruction:**

1. **ADD – Too download the context from url.**
2. **CMD – it will run at the time of using container it is uniqe (if we add more cmd command last command will work)**
3. **ENTRY-POINT—it will run at runtime (If we add more enty points first one will work ,because it will not override)**
4. **ENV – this instruction can be used to set environment veriable.**
5. **EXPOSE – to open the ports when running the container.**
6. **FROM – to download the image from docker hub.(THis will use at te time of docker file starting)**
7. **MAINTAINER – author of the docker file.(This is not mandatory)**
8. **RUN – on the top of the existing layer this command will work.**
9. **USER–to set the user id or user name to use when running the image.**
10. **VOLUME – enable access to a location on the host system from container.**
11. **WORKDIRECTORY – to set current working directory.**
12. **ON BUILD – when one image is used as the base for same other image.(like down stream docker file).**
13. **Docker network – to display all networks.**
14. **docker network create --driver = brdge java-hari : To create docker customer bridge network.**
15. **docker inspect network name – gives all ips of that network.**
16. **docker logs—all logs.**
17. **Docker volume create volume name – to create a volume name.**
18. **Docker volume ps – to display all docker volumes.**

**VOLUMES**

|  |  |
| --- | --- |
| **NORMAL VOLUMES**  **This volumes are residing in the host machine**  **Managed by the docker**  **Non Docker process cannot modify this volumes** | **BINDED VOLUMES**  **This volumes are residing in the host machine**  **This can be managed by user.**  **Non Docker process will modify this volumes easily.** |

1. **docker import : Creates an image from a tarball.**
2. **docker load : load an image from tarfile. (image is inside tar file.)**
3. **docker build : To create a image from docker file.**
4. **docker commit – creating a image from a container, pausing. it is temporary if it is running.**
5. **docker container prune: To remove stopped containers**
6. **docker images prune : To remove untagged and unused images used by containers.**
7. **docker images prune -a: To remove all unused docker images**
8. **docker cp hostpat/filename containerid:containerpath: To copy the files from host to container**
9. **docker cp containerid:containerpath hostpat/filename : To copy files from container to hostmachine**

**Naming convensions of docker :**

**DockerFile.azure.prod, DockerFile.azure.qa, azure.prod. DockerFile, azure.qa. DockerFile.**

**Docker build -f DockerFile.azure.prod -t dokcerimagename: This wiss specify exact docker file.**

**version: ‘3’**

**services:**

**database:**

**image: redis**

**web:**

**image :nginx**

**ports:**

**-56392: 80**

**version: ‘3’ --------------- version of the dockerfile.**

**services: ---------------services**

**database: ---------------service name**

**image: redis --------------- image name**

**web:**

**image :nginx ------------- image name**

**ports : ----------------To expose the ports**

**-56392: 80**

**Docker-compose will used to up more services at a time.**

**docker- compose up/down -d --scale database=4**

**docker-compose ps --status of the container in current stack.**

**docker-compose kill –to stop the container started with docker-compose up.**

Jenkins:

1) /var/lib/jenkins/jobs ---------------------> It will store all jobs

2) /var/lib/jenkins/config.xml---------------------> Basic configuration file

3) ./jenkins.sh-------------------------------------> To start jenkins

4) Major Plugins:

1) Git plugin -------Useful to connect git(scm)

2) SSh copy plugin -------------It will copy files to destinnation

3) Blueu Ocean ------------------It will change view of the jenkins

4) Git Parameter Plug-in------------It will provide git credentials as parameters.

5) Extended Mail Plug-in------------It will send mail to respected person when specific event occurse

6) RoleBack plugin------------------

7) Maven Plugin

9) Sonar Scanner

10) Backupplugin --------------To take a backup

11) AnsiblePlugin --------------TO contact ansible server

12) DockerPlugin -------------To connect docker container

13) PublishOverSSH -------------Sending and running a commands in target machine

14) EmbededBuildStatusPlugin-----It will display the job status.

15) BigBucket-------------------To integrate with bigbucket

16) GreenBalls -----------------To change color of succes blue to grean

17) Junit-----------------------To publish junit testcases.

18) Rebuild---------------------With out entering parameters again when build rerun this plugin will enter paremeters again

19) JobGeneratorPlugin--------This plugin gibves flexiblity to define templat4es,

Developers will create own jobs

20) Role Based authentication --------To give authrization to user

21) Configuration Slicing Plugging--------TO change bulk changes in multiple jobs

22) Delivery pipeline plugin: To display delivery pipeline as proper UI

23) Build pipeline plugin:

23) NOfification plugin: This will send email notification

24) POll mail trigger plugin.

CONFIGURE: This will describe all installed softwares/

GLOBAL TOOL CONFIGURATION: Tool of all with direct installation.

Build Triggers types:

1) Manual 2)Pollscm3) Buildperiodiacally 4)Build with token in url 5)Git huck

var/lib/jenkins

MAVEN:

Maven Build Phases:

Deploy

Install

test

Build

compile

Validate

Maven life cycles:

Site

Clean

Default

Maven Goals: These are all triggers to maven build phases. Goals will trigger maven phases.

These goles are based or the pom.xml.

Example: mvn compile:compile ---------> This will trigger compile phase.

Note: In all phases only test phase can skip.

**1) Explain what is Maven? How does it work?**

Maven is a project management tool. It provides the developer a complete build lifecycle framework. On executing Maven commands, it will look for POM file in Maven; it will run the command on the resources described in the POM.

**2) List out what are the aspects does Maven Manages?**

Maven handles following activities of a developer

• Build  
• Documentation  
• Reporting  
• Dependencies  
• SCMs  
• Releases  
• Distribution  
• Mailing list

**3) Mention the three build lifecycle of Maven?**

• Clean: Cleans up artifacts that are created by prior builds  
• Default (build): Used to create the application  
• Site: For the project generates site documentation

**4) Explain what is POM?**

In Maven, POM (Project Object Model) is the fundamental unit of work. It is an XML file which holds the information about the project and configuration details used to build a project by Maven.

**5) Explain what is Maven artifact?**

Usually an artifact is a JAR file which gets arrayed to a Maven repository. One or more artifacts a maven build produces such as compiled JAR and a sources JAR.

Each artifact includes a group ID, an artifact ID and a version string.

[Maven](https://career.guru99.com/wp-content/uploads/2014/09/Maven.jpg)

**6) Explain what is Maven Repository? What are their types?**

A Maven repository is a location where all the project jars, library jars, plugins or any other particular project related artifacts are stored and can be easily used by Maven.

Their types are local(.m2), central and remote(third party)

**7) Why Maven Plugins are used?**

Maven plugins are used to  
• Create a jar file  
• Create war file  
• Compile code files  
• Unit testing of code  
• Documenting projects  
• Reporting

**8) List out the dependency scope in Maven?**

The various dependency scope used in Maven are:

• Compile: It is the default scope, and it indicates what dependency is available in the classpath of the project  
• Provided: It indicates that the dependency is provided by JDK or web server or container at runtime  
• Runtime: This tells that the dependency is not needed for compilation but is required during execution  
• Test: It says dependency is available only for the test compilation and execution phases  
• System: It indicates you have to provide the system path  
• Import: This indicates that the identified or specified POM should be replaced with the dependencies in that POM’s section

**9) Mention how profiles are specified in Maven?**

Profiles are specified in Maven by using a subset of the elements existing in the POM itself.

**10) Explain how you can exclude dependency?**

By using the exclusion element, dependency can be excluded

**11) Mention the difference between Apache Ant and Maven?**

Apache Ant Maven  
• Ant is a toolbox – Maven is a framework  
• Ant does not have formal conventions like project directory structure – Maven has conventions  
• Ant is procedural; you have to tell to compile, copy and compress – Maven is declarative ( information on what to make & how to build)  
• Ant does not have lifecycle; you have to add sequence of tasks manually – Maven has a lifecycle  
• Ant scripts are not reusable – Maven plugins are reusable

**12) In Maven what are the two setting files called and what are their location?**

In Maven, the setting files are called settings.xml, and the two setting files are located at

• Maven installation directory: $M2\_Home/conf/settings.xml  
• User’s home directory: ${ user.home }/ .m2 / settings.xml

**13) List out what are the build phases in Maven?**

Build phases in Maven are

• Validate  
• Compile  
• Test  
• Package  
• Install  
• Deploy

**14) List out the build, source and test source directory for POM in Maven?**

• Build = Target  
• Source = src/main/java  
• Test = src/main/test

**15) Where do you find the class files when you compile a Maven project?**

You will find the class files ${basedir}/target/classes/.

**16) Explain what would the “jar: jar” goal do?**

jar: jar will not recompile sources; it will imply just create a JAR from the target/classes directory considering that everything else has been done

**17) List out what are the Maven’s order of inheritance?**

The maven’s order of inheritance is

• Parent Pom  
• Project Pom  
• Settings  
• CLI parameters

**18) For POM what are the minimum required elements?**

The minimum required elements for POM are project root, modelVersion, groupID, artifactID and version

**19) Explain how you can produce execution debug output or error messages?**

To produce execution debug output you could call Maven with X parameter or e parameter

**20) Explain how to run test classes in Maven?**

To run test classes in Maven, you need surefire plugin, check and configure your settings in setting.xml and pom.xml for a property named “test.”

GIT

Git status: Check status of the current.

Git init : Initialing git repository

Git config --Global user.name "sanjay"

Git config --global user.email sanjay\_kaumarreddy@rediffmail.com

git add file name: To add file from work space to staging /index are.\

Git commit -m "first commit" : To commit from index area to local repository.

Git log : To display all commit.

Git push: To send changes to remote repository.

Git show commitid: Ii will display all files with that commit id.

Git clone giturl : TO get entire repository from github.

git log --oneline: Display logs in single line.

Git reset filename: This will bring file from staging area to work space area.

Git reset --soft commitid : This will bring all file from local repository to index area with that commitid.

Git reset --mixed commid \_id : This will girectly take localrepository to workspacearea.

git branch : To Display all branches.

Note: If we cloned with git clone, No need to use git init commnd,To inetiate. Just navigate to that folder.

Git pull : To get all updates from current branch.

Git --aboart : To aboart the conflicts.

Stash Memory: This is temporary area to store files .

git stash list : This will display all stashed files.

Git stash save "name of the stash" : This store all files in to stash memory.

Git stash show stash[0] : This display corresponding file in stash memory.

Git stash pop stash[0]:This will bring file from corresponding stash memory to staging/index area. And removed file from stash memory.

Git stash apply stash[0]: This will bring all files from stash memory to index/staging area.

This will not delete file from stash memory.

Note: Here stash[0] is a memory location

Git stash drop stash[0]: TO drop the stash memory.

.ignorefile: This file contains all files names, That all files will not be added and committed. These files will ignored by git.

Git tag : It will display all tags in current branch.

Git tag commitid tag\_name: This will create tag with respective commit id.

Git push orign tagname: To update the tage name in centralized location.

Git tag -d tagname: This will remove the tag from local repository,

Git push orign -d tagname: To remove tag in git hub.

Git branch -d branchname: To delete the branch in local repo.

Note: If any changes in new branch it is not possible to delete new branch until unless these new changes will commit to main branch.

GitPull Request: While merging future branch in master branch ,Pull request will required. This pull request will approved by all collaborated access. This will happen from ui.

GitMergeToll: This tool will come by default from git. This tool will open ui and display conflicts automatically.We can directly move conflict code TO/From.

git config --mergetool: This command will open git ui with conflict code.

Apache Tomcat

server.xml,user.xml,catilina.sh,start.sh

user.xml:

Jconsole:

http,https(certificates self signed)

JVM monitoring with jconsole.

/webapp: This contains

Getting access log:

LogLoevel In tomcat :

1. Info
2. Debug
3. Warn
4. Notice
5. Crit
6. Alarm
7. Emerg
8. Error

**Status code 403:**It refers to a forbidden error like if a file misses some security context.

**Status code 404:**It refers to an error message that it is an http response and the client was not able to communicate with the given server.

NEXUS INSTALLATION

<https://devopscube.com/how-to-install-latest-sonatype-nexus-3-on-linux/>

yum update -y

yum install java-1.8.0-openjdk.x86\_64

mkdir /app && cd /app

wget https://sonatype-download.global.ssl.fastly.net/nexus/3/nexus-3.0.2-02-unix.tar.gz

tar -xvf nexus-3.0.2-02-unix.tar.gz

mv nexus-3.0.2-02 nexus

adduser nexus

chown -R nexus:nexus /app/nexus

Open /app/nexus/bin/nexus.rc file, uncomment run\_as\_user parameter and set it as following.

run\_as\_user="nexus"

vi /app/nexus/bin/nexus.vmoptions(MEMORY OPTIONS)

-Xms1200M

-Xmx1200M

-XX:+UnlockDiagnosticVMOptions

-XX:+UnsyncloadClass

-Djava.net.preferIPv4Stack=truer

-Dkaraf.home=.

-Dkaraf.base=.

-Dkaraf.etc=etc

-Djava.util.logging.config.file=etc/java.util.logging.properties

-Dkaraf.data=/nexus/nexus-data

-Djava.io.tmpdir=data/tmp

-Dkaraf.startLocalConsole=false

Creating soft link:

ln -s /app/nexus/bin/nexus /etc/init.d/nexus

Execute the following commands to add nexus service to boot.

chkconfig --add nexus

chkconfig --levels 345 nexus on

service nexus start

|  |  |
| --- | --- |
|  |  |