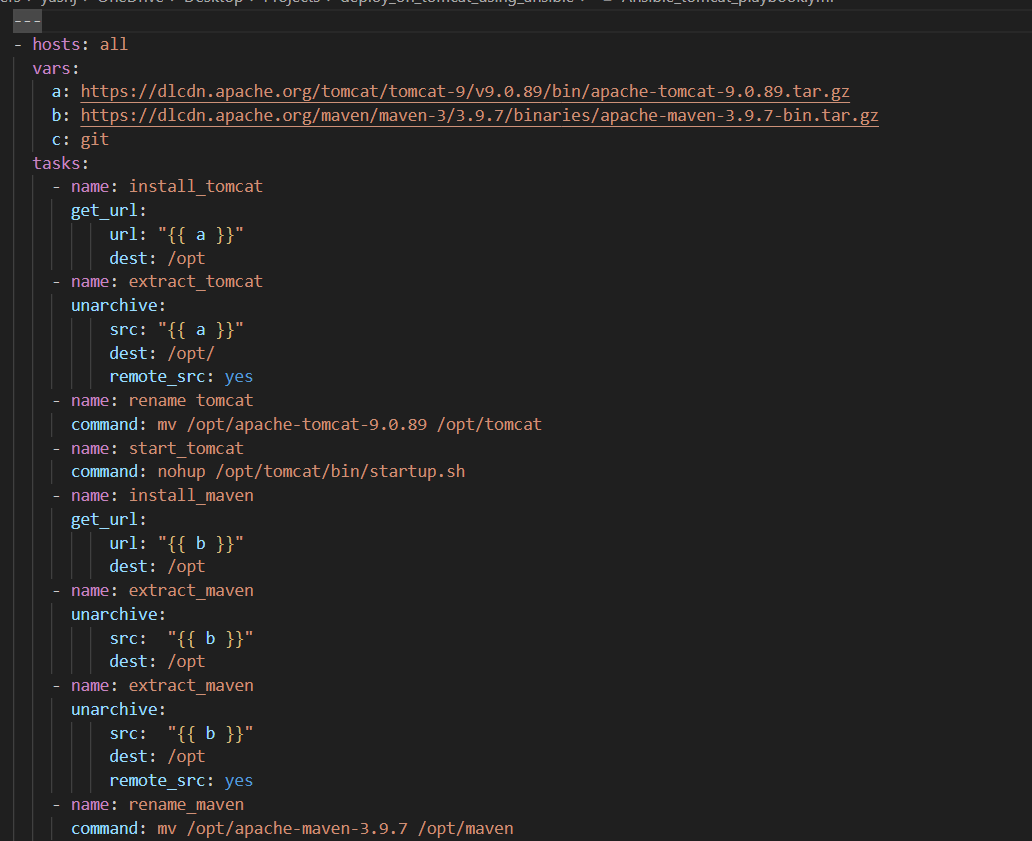
**Deployed WAR file on Tomcat using Ansible:**

Below is the Ansible-Playbook, for deploying war files on tomcat.





Ansible works on Master-Node architecture.

To communicate with Node, generate the keys using ssh-keygen and copy the public key from Master to Node.

Enable Passwordless authentication in Master, /etc/ssh/sshd\_config

Restart sshd.

Create a host file, which is basically an Inventory file, on Master and add the private IP of Node inside the host file. Save the file.

Once this is done, use command,

Ansible-playbook -i <inventory/host file name> -m ping all 🡪 to see if communication is success between Master and Node.

If ‘-i’ is not specified, then ansible will go and look the hosts in the default host/inventory file location, which is, /etc/ansible/hosts.

If inventory file is created and need to specify that file, so that ansible will look for this specific file for the hosts, then use -i option.

To Make this newly created host/inventory file as default inventory file, then go to /etc/ansible/ansible.cfg, under default section, change the inventory file location from /etc/ansible/hosts to newly created host file location.

With this, no need to mention -i option explicitly.

Ansible all -m ping

-m -> this option is used to specify the Ansible modules.

Example: ping is one of the Ansible module.

all -> this is specified so that it will ping in all the hosts mentioned in inventory file. If need to ping in specific Node, then we can specify the specific Node private IP.

**Modules used inside the Playbook:**

Each Modules in Ansible has a Particular Functionality.

. get\_url 🡪 This module is used to download files from HTTP, HTTPS or FTP to Node.

. unarchive 🡪 this module is used to extract the file, means unpacks an archive.

. command 🡪 this module is used to run any commands or scripts in the Node server.

. yum 🡪 yum is a package manager, with this module we can install, remove, and update the packages.

. git 🡪 this module is used to clone the GitHub repo to Node.

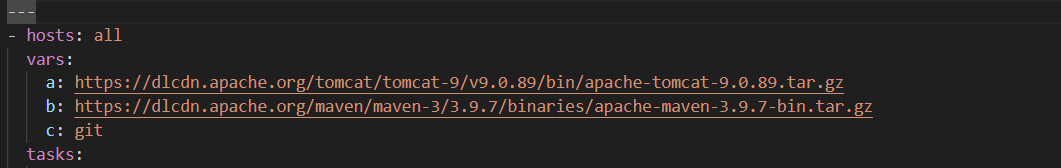
. file 🡪 this module is used to create/delete the files in the Node.

. copy 🡪 this module is used to copy files from Master to Node and even used to copy the contents inside the file.

. shell 🡪 this module is used to run shell commands on Node. The shell module does not execute directly on the target but in a shell environment ( /bin/sh ) on the target.

**Understanding Ansible-Playbook:**

Lets divide Playbook into small chunks and understand about it.



Every Ansible-playbook starts with ‘---’, but this is optional.

In Ansible-Playbook to specify lists, ‘-’ is used.

As hosts has lists, such as vars, tasks.

* hosts: all 🡪 with this, whatever tasks we are specifying in Playbook will be executed on all hosts, mentioned inside the inventory file.

To execute inside the specific Node/host, then specify the IP address of that specific Node.

vars 🡪 this is used to specify variables, in key-value format.

Variables are used to store the data inside it.

So that whenever data gets changes, then we can change the value of variable, instead of changing in all other places manually.

‘a’ variable is set to Tomcat install URL.

‘b’ variable is set to Maven install URL.

‘c’ variable is set to install Git.

tasks 🡪 this is used to specify the tasks which should be performed on the hosts.

Tasks has lists too. Means, tasks has multiple tasks.

Each tasks will be specified with ‘-’.

**Pre-requisites to deploy WAR file on the Tomcat:**

. Download tomcat Archive file

. Extract and rename the Archive file to tomcat.

. Download the Maven archive file, to build the code.

. Extract and rename the Archive file to maven

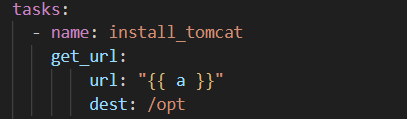
. Install git, to clone the repo which has code to local.

Explained below how to download tomcat, maven and git using Playbook**:**

* For deploying WAR file inside the tomcat, need to install tomcat first.

In Ansible variables are specified as,

“{{ <variable name> }}”

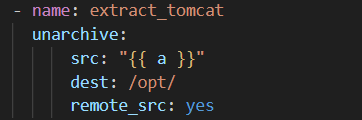


Given task name as ‘install\_tomcat’.

Used get\_url module to download the tomcat archive file.

Destination is provided as /opt, in this directory tomcat archive file will be downloaded.

* After archive file is downloaded, need to extract the archive file.



Given task name as ‘extract\_tomcat’.

Used unarchive module to extract the archive file.

Given source location as variable ‘a’, this variable specifies the url which contains the tomcat archive file.

Given destination location as /opt, in this directory the file will be extracted.

Make sure to give remote\_src: yes.

If this is not specified then it will throw below error,

The error was: If you are using a module and expect the file to exist on the remote, see the remote\_src option.

And it will look for the file in local system instead of target. That’s the reason specify the remote\_src: yes

* Rename the file to tomcat.



Command module is used to run any commands in Linux.

* Once it is installed and renamed, need to start tomcat.



Here command module is used.

Shell module can be used too, to start tomcat.

Shell: sh /opt/tomcat/bin/startup.sh

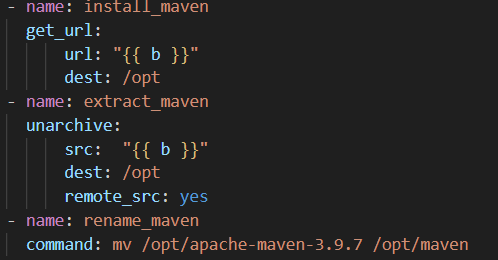
But with this, tomcat will be up only till the playbook execution. Once playbook is executed, tomcat will be shutdown.

To avoid this, used nohup command.

nohup stands for “No Hang up”.

With this nohup command, service will be in running state, even after the execution of playbook and even if the terminal is closed.

* To build the code, need to install Maven, then need to extract it inside /opt directory.



Used variable ‘b’, as this variable specifies the download URL of the maven archive file.

To download Maven, used get\_url module.

This module has url and dest arguments.

For url, mentioned variable b.

For destination, mentioned /opt directory, as file will be downloaded in opt directory.

To extract Archive file, used unarchive module.

This module has 3 arguments, source, destination and remote\_src.

For source, specified variable b.

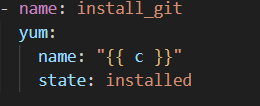
For destination, specified the directory in which it should be extracted.

Its mandatory to give remote\_src: yes, to say that, “yes archive file is located in /opt directory”.

If this remote\_src isn’t specified then it will throw error.

Lastly, need to rename the extracted file name to maven.

* Now, last pre-requisite is to install Git.



Given task name as ‘install\_git’.

To install, used yum module.

This module has 2 arguments, name and state.

For name, need to provide the package name which is Git. As variable ‘c’ contains the value as Git, specified variable ‘c’ as name.

For state, mentioned as installed. It can be mentioned as present too. This will install the Git.

If state is mentioned as absent, then it will uninstall.

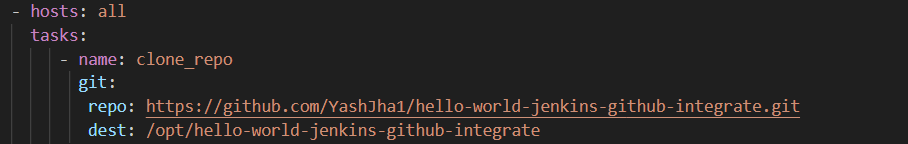
**Clone the repo which has code, build the code and deploy on Tomcat:**

* Once pre-requisites are done, actual thing starts here.

Create a new play in the same playbook by which pre-requisites are installed.

Need to clone the repo which has code.

The new play will again start with – hosts.



Git module is used to clone the repo.

This module has 2 arguments, repo and destination.

For repo, need to mention the GitHub repo URL.

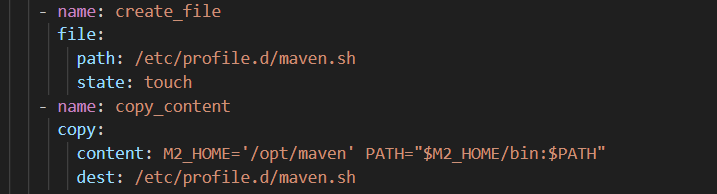
For destination, provide the directory where it should be cloned or copied.

Either provide as /opt or if need to copy the file name with other name, then mention as, /opt/<name>.

* Once repo is cloned, need to build the code which is in that repo. But to run maven commands, firstly need to add variables inside the /etc/profile.d

Inside /etc/profile.d need to create a file with name,

maven.sh and then need to add variables inside the file.



File module is used to create file. This module has 2 arguments, path and state.

For path, we are providing the path where the file should get created along with file name.

For state, used touch. With this file will be created.

Once it is created, need to add contents/variables into it.

Copy module is used to copy the contents inside the file.

This module has 2 args, content and destination.

For content, need to provide the contents which should be added to the file.

Contents are:

M2\_HOME='/opt/maven' PATH="$M2\_HOME/bin:$PATH"

For destination, provided the file path, so that contents will be added to the file.

* Once specific variables are added, then maven commands can run anywhere in any location.

Used shell module, as mvn is a shell command.

First navigated to repo directory which we cloned using git module.

Then, ran the command, mvn clean install.

This command will build the code and it will create a war file inside the target directory.

/webapp/target/<.war file >

* Once code is build, need to deploy it on the Tomcat. 

used command module, and gave the above command to copy war file inside the tomcat/webapps/

Run the playbook, using below command,

ansible-playbook -i hosts <playbook name>

Once the playbook is ran, it will execute all the tasks mentioned inside the playbook on the Node/target server.

Lastly war file will be deployed on tomcat.

Tomcat runs on port 8080.

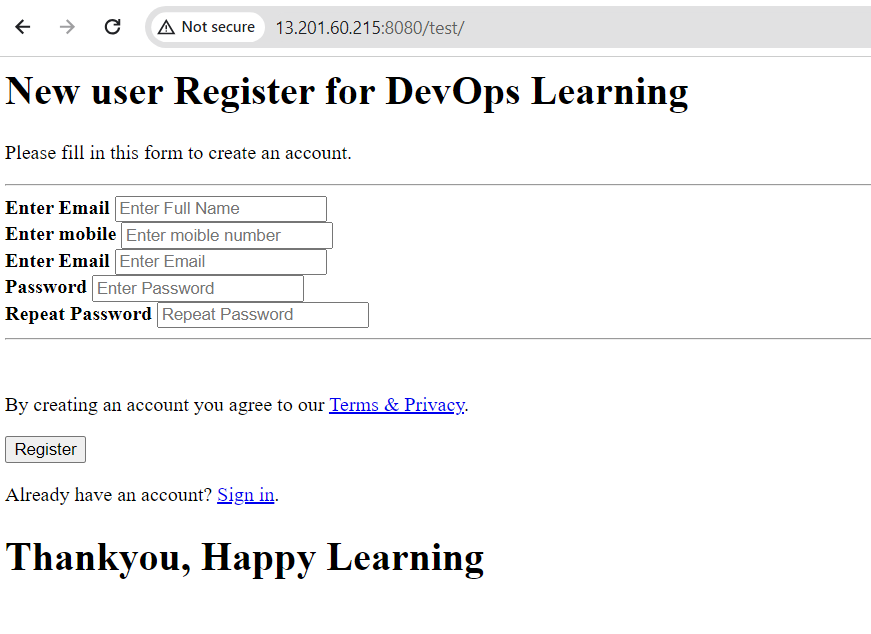
To see the application is deployed or not, go to browser,

http://<instance public IP>:8080/<war file name or app name>

In my case, URL was,

http://13.201.60.215:8080/test

When I hit the above URL, this was my output,



**Note:** Once tomcat file is unarchived, give execution permission to startup.sh and shutdown.sh

File module is used.

File:

Path: /opt/tomcat/bin/startup.sh

Mode: 0777

Once execution permission is given, start the services.

Shell module is used.

Shell: nohup ./startup.sh

Args:

Chdir: /opt/tomcat/bin

If nohup isn’t given and the command is executed, once executed, this script(startup.sh) wont work.

To execute in the background continuously, use nohup command.