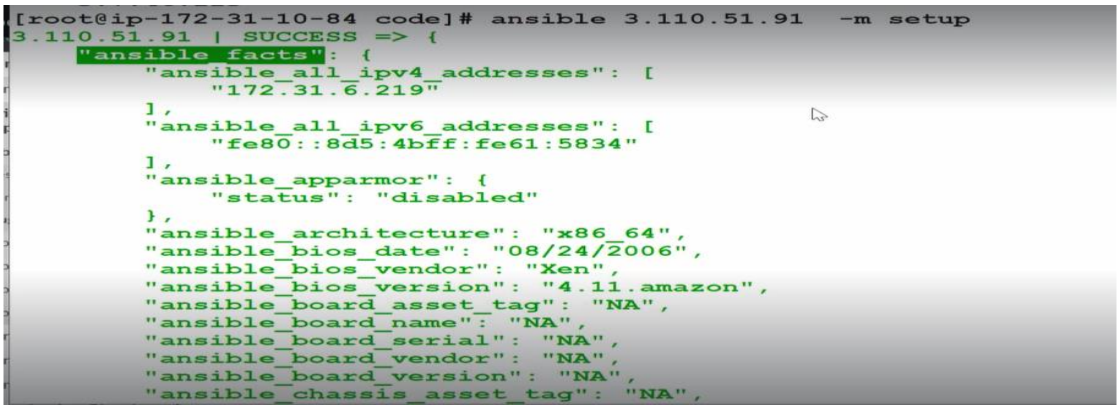
Ansible

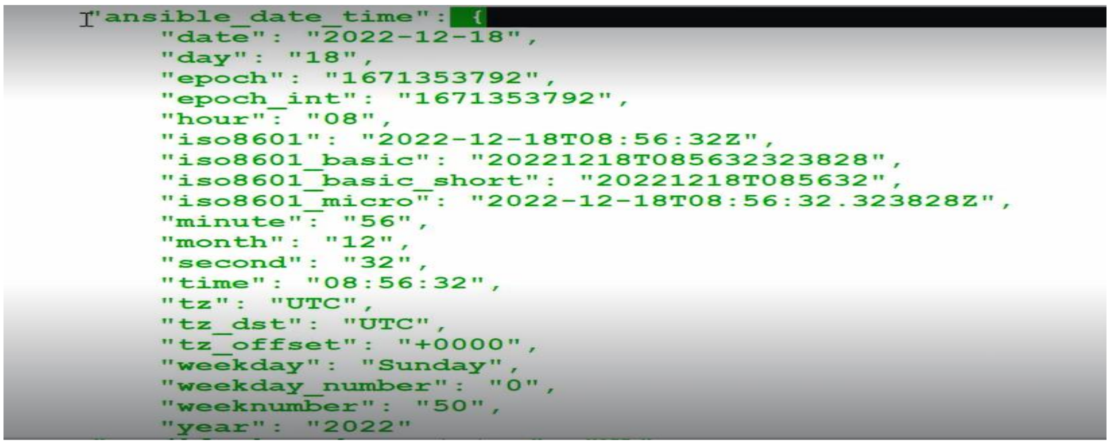
Session 4 – 18th Dec 2022 Summary

- Whenever the controller node hits the target node, ansible has the capability to retrieve the real time status of the target node.
- The command to retrieve all the facts of the node 3.110.51.91 is "**ansible 3.110.51.91 –m setup**". The "**setup**" module have the capability of retrieving all the real time status of a node, technically called as **Facts.** The information (value) retrieved is stored in a variable, here the variable is fact. When two systems exchange information they use a standard language called JSON



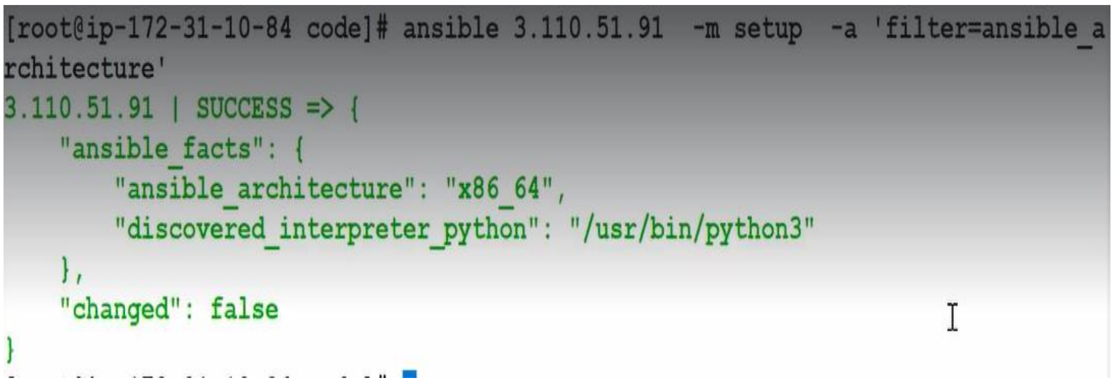
```
[root@ip-172-31-10-84 code]# ansible 3.110.51.91  -m setup
3.110.51.91 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "172.31.6.219"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::8d5:4bff:fe61:5834"
        ],
        "ansible_apparmor": {
            "status": "disabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "08/24/2006",
        "ansible_bios_vendor": "Xen",
        "ansible_bios_version": "4.11.amazon",
        "ansible_board_asset_tag": "NA",
        "ansible_board_name": "NA",
        "ansible_board_serial": "NA",
        "ansible_board_vendor": "NA",
        "ansible_board_version": "NA",
        "ansible_chassis_asset_tag": "NA",
```

- The different facts are grouped together in one block, this entire block of code has also been given a name



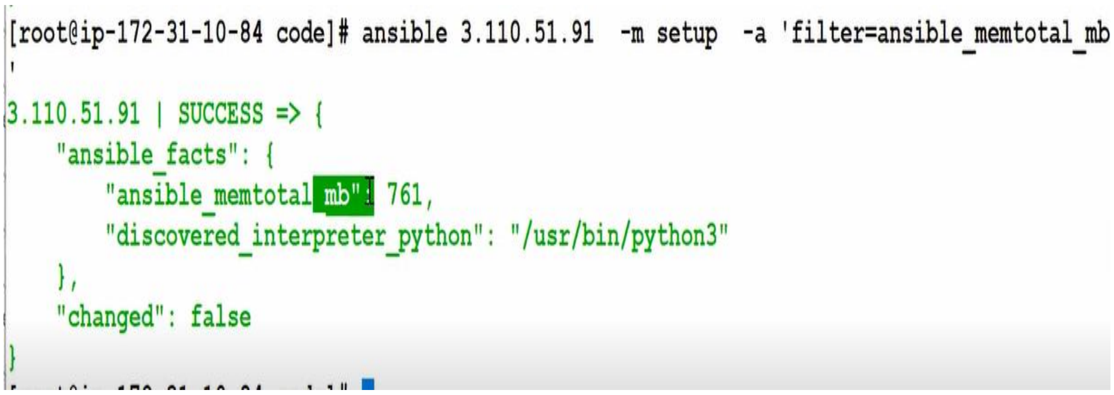
```
"ansible_date_time": {
    "date": "2022-12-18",
    "day": "18",
    "epoch": "1671353792",
    "epoch_int": "1671353792",
    "hour": "08",
    "iso8601": "2022-12-18T08:56:32Z",
    "iso8601_basic": "20221218T085632323828",
    "iso8601_basic_short": "20221218T085632",
    "iso8601_micro": "2022-12-18T08:56:32.323828Z",
    "minute": "56",
    "month": "12",
    "second": "32",
    "time": "08:56:32",
    "tz": "UTC",
    "tz_dst": "UTC",
    "tz_offset": "+0000",
    "weekday": "Sunday",
    "weekday_number": "0",
    "weeknumber": "50",
    "year": "2022"
```

➢ To retrieve a **particular block of data**, "**filter**" attribute can be used along with the information (fact) that is being retrieved. Here the **information about the architecture** of a node is retrieved.



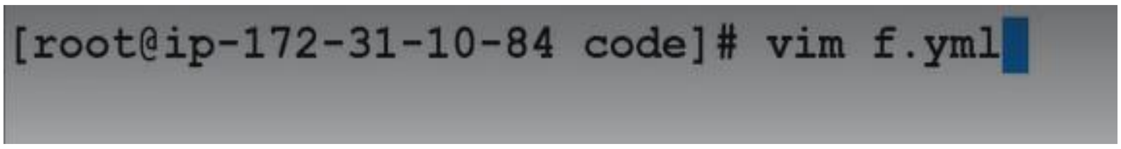
```
[root@ip-172-31-10-84 code]# ansible 3.110.51.91  -m setup  -a 'filter=ansible_a
rchitecture'
3.110.51.91 | SUCCESS => {
    "ansible_facts": {
        "ansible_architecture": "x86_64",
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

➢ The command used to retrieve the total memory information of a node

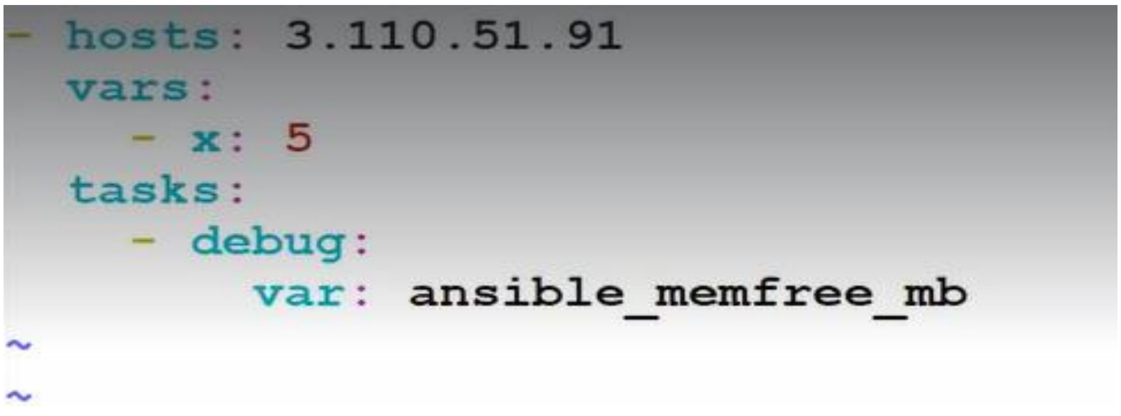


```
[root@ip-172-31-10-84 code]# ansible 3.110.51.91  -m setup  -a 'filter=ansible_memtotal_mb
'
3.110.51.91 | SUCCESS => {
    "ansible_facts": {
        "ansible_memtotal_mb": 761,
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

➢ The command to create a playbook "vim f.yml". The playbook is written to retrieve the information of the "**total free memory available**" of a node. The fact name is "**ansible_memfree_mb**".

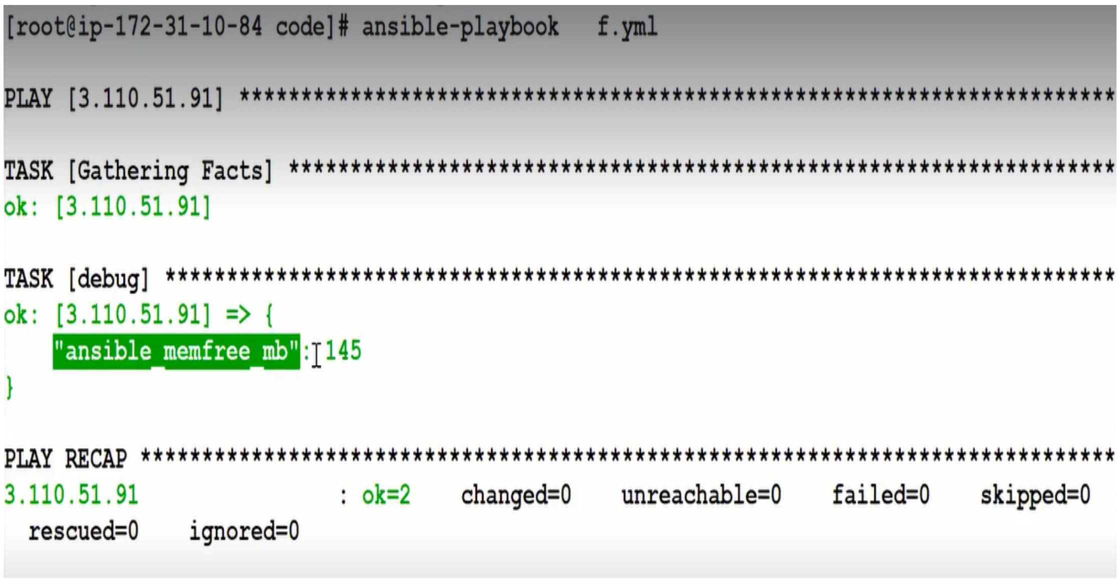


```
[root@ip-172-31-10-84 code]# vim f.yml
```



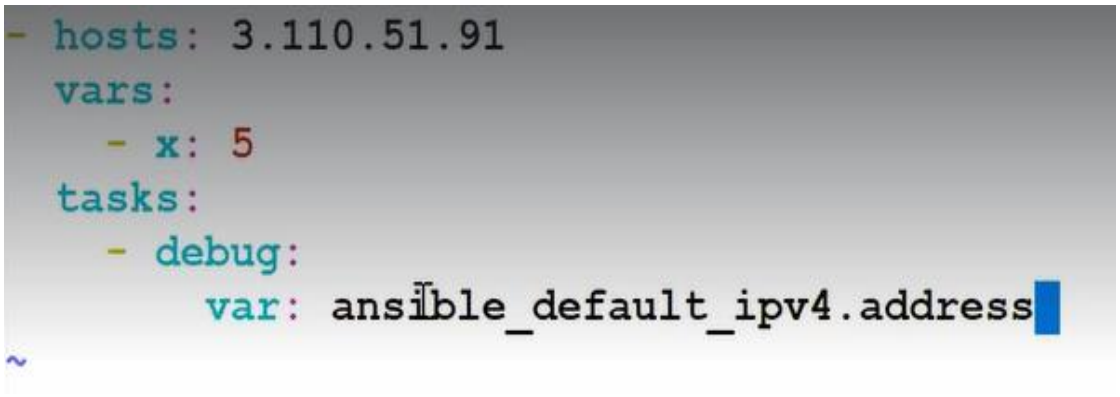
```
- hosts: 3.110.51.91
  vars:
    - x: 5
  tasks:
    - debug:
        var: ansible_memfree_mb
~
~
```

➢ The command to run the playbook “ansible-playbook f.yml”



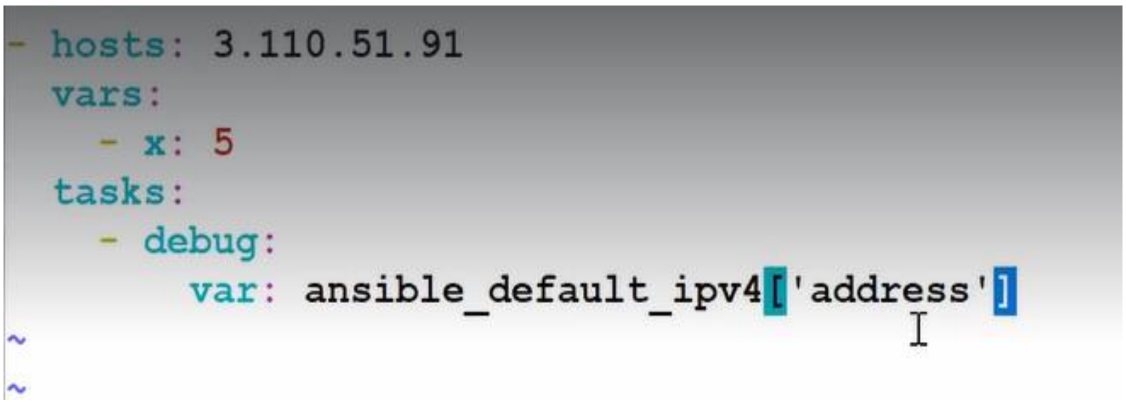
```
[root@ip-172-31-10-84 code]# ansible-playbook   f.yml

PLAY [3.110.51.91] *************************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [3.110.51.91]

TASK [debug] *******************************************************************************
ok: [3.110.51.91] => {
    "ansible_memfree_mb": 145
}

PLAY RECAP *********************************************************************************
3.110.51.91                : ok=2    changed=0    unreachable=0    failed=0    skipped=0
  rescued=0    ignored=0
```

➢ To retrieve a particular fact “**IP Address of a node**” in the block of code “ **ansible_default_ipv4**”

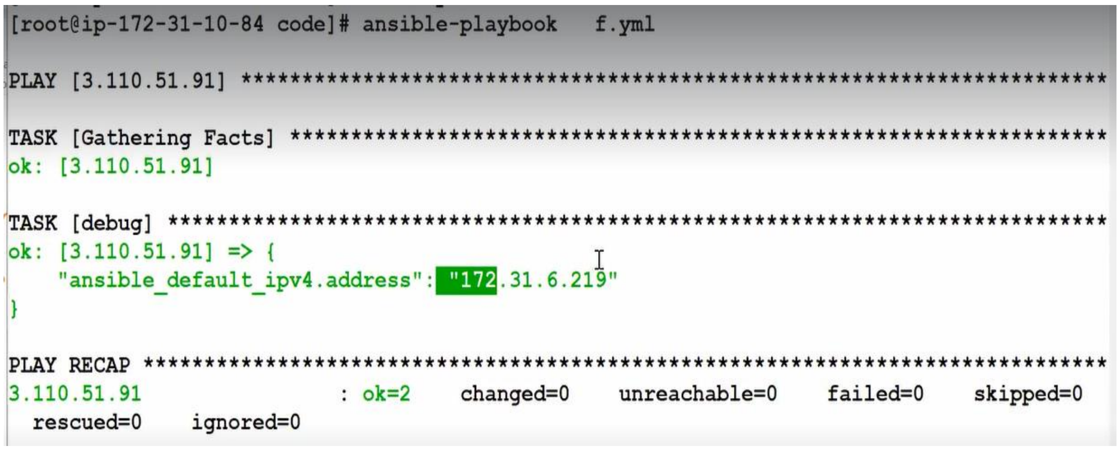


```
- hosts: 3.110.51.91
  vars:
    - x: 5
  tasks:
    - debug:
        var: ansible_default_ipv4.address
~
```

Or

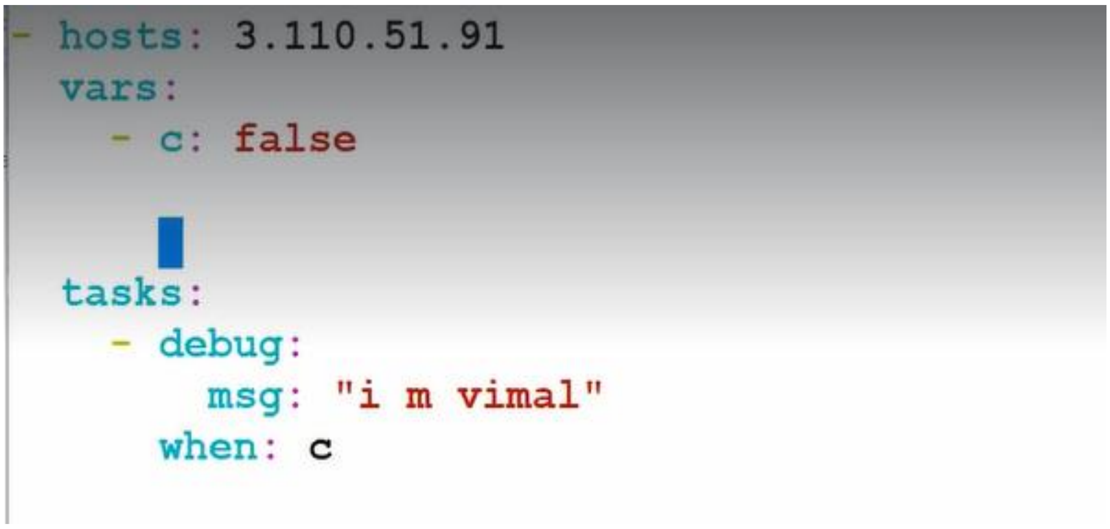


```
- hosts: 3.110.51.91
  vars:
    - x: 5
  tasks:
    - debug:
        var: ansible_default_ipv4['address']
~
~
```

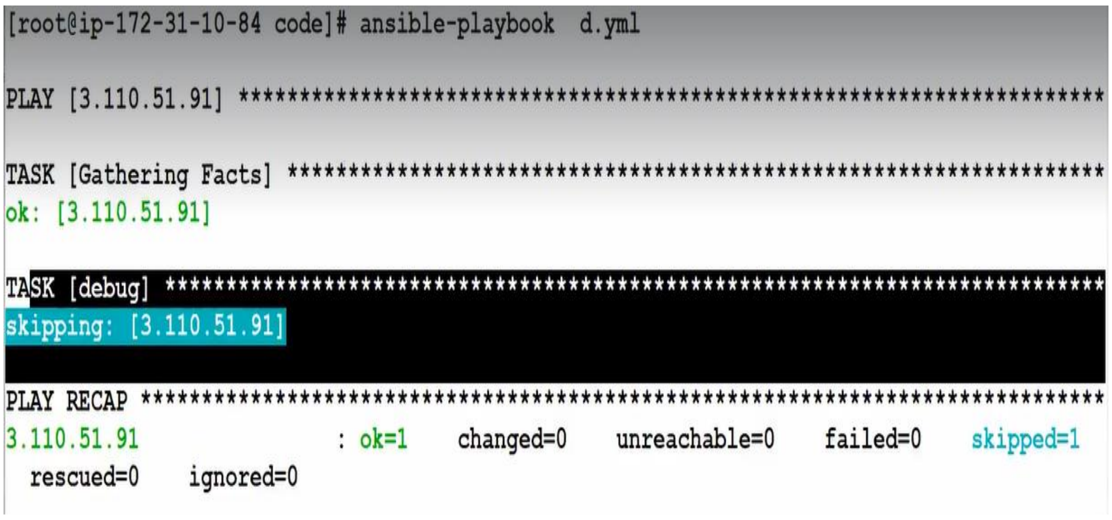


```
[root@ip-172-31-10-84 code]# ansible-playbook   f.yml

PLAY [3.110.51.91] ***************************************************************************

TASK [Gathering Facts] ***********************************************************************
ok: [3.110.51.91]

TASK [debug] *********************************************************************************
ok: [3.110.51.91] => {
    "ansible_default_ipv4.address": "172.31.6.219"
}

PLAY RECAP ***********************************************************************************
3.110.51.91                : ok=2    changed=0    unreachable=0    failed=0    skipped=0
  rescued=0    ignored=0
```

- The **Gathering Facts** automatically run in the playbook world and behind the scene use the **setup** module, to retrieve the information or real time status of a node
- To create a playbook to control the task using "**when**" keyword with boolean value "true" or "false"

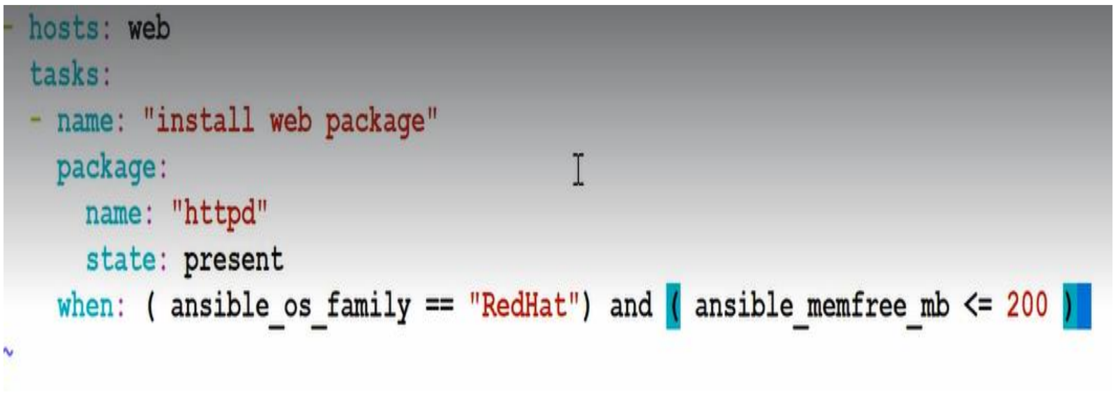


```
- hosts: 3.110.51.91
  vars:
    - c: false


  tasks:
    - debug:
        msg: "i m vimal"
      when: c
```



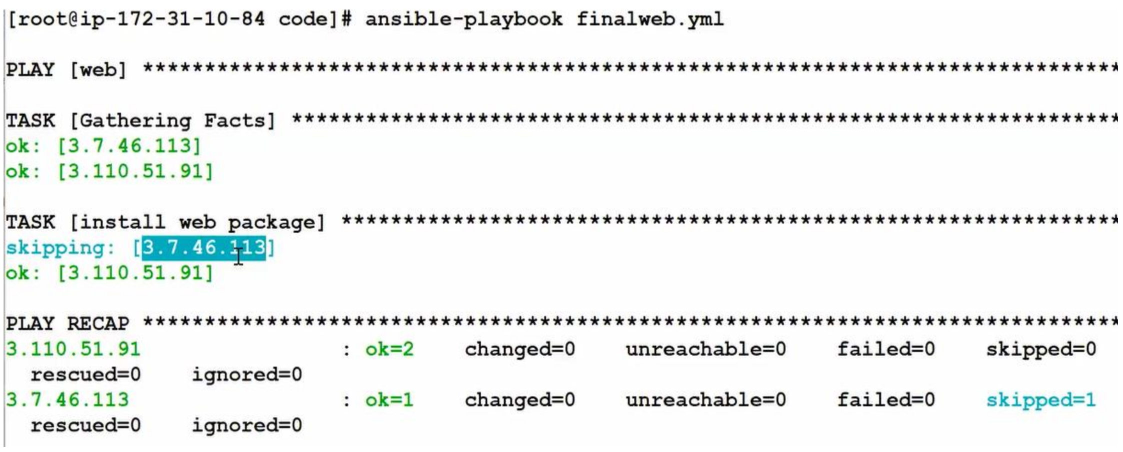
```
[root@ip-172-31-10-84 code]# ansible-playbook  d.yml

PLAY [3.110.51.91] ***************************************************************************

TASK [Gathering Facts] ***********************************************************************
ok: [3.110.51.91]

TASK [debug] *********************************************************************************
skipping: [3.110.51.91]

PLAY RECAP ***********************************************************************************
3.110.51.91                : ok=1    changed=0    unreachable=0    failed=0    skipped=1
  rescued=0    ignored=0
```

➢ To create a playbook to install a package "**httpd**" only if the **OS is "RedHat"** and the **free memory space of node** is more or equal to 200MB



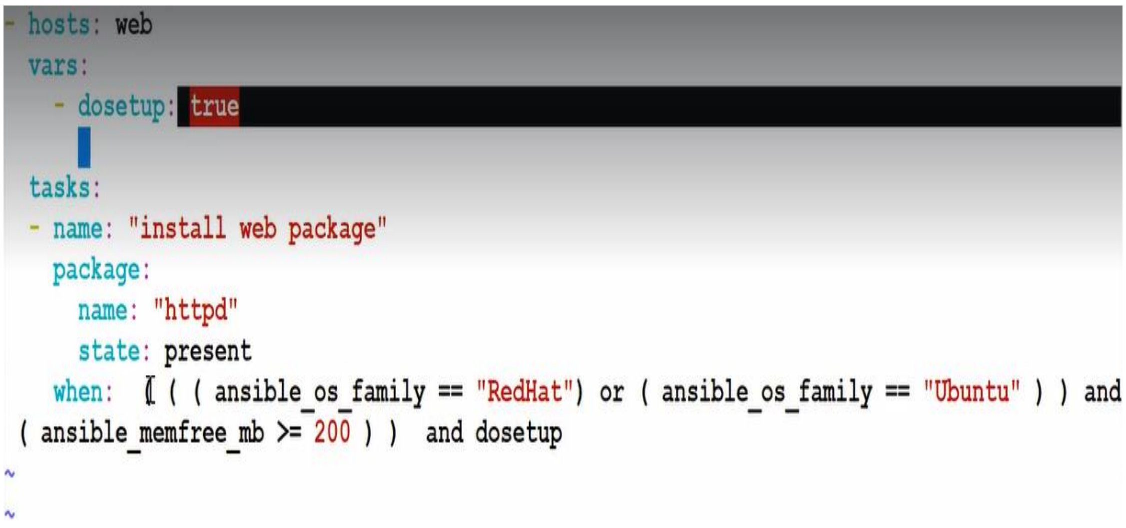
```
- hosts: web
  tasks:
  - name: "install web package"
    package:
      name: "httpd"
      state: present
    when: ( ansible_os_family == "RedHat") and ( ansible_memfree_mb <= 200 )
~
```

➢ When the playbook is run, the task is skipped on the node "3.7.46.113", since the OS is RedHat but the free memory space of this node is less.



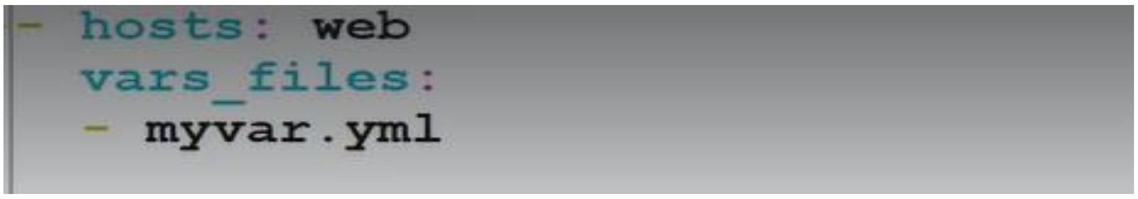
```
[root@ip-172-31-10-84 code]# ansible-playbook finalweb.yml

PLAY [web] ***************************************************************************

TASK [Gathering Facts] ***************************************************************
ok: [3.7.46.113]
ok: [3.110.51.91]

TASK [install web package] ***********************************************************
skipping: [3.7.46.113]
ok: [3.110.51.91]

PLAY RECAP ***************************************************************************
3.110.51.91                : ok=2    changed=0    unreachable=0    failed=0    skipped=0
  rescued=0    ignored=0
3.7.46.113                 : ok=1    changed=0    unreachable=0    failed=0    skipped=1
  rescued=0    ignored=0
```

➢ Controlling the task to install the package **"httpd"** using a variable "**dosetup**" with the boolean values "**true**" or "**false**"

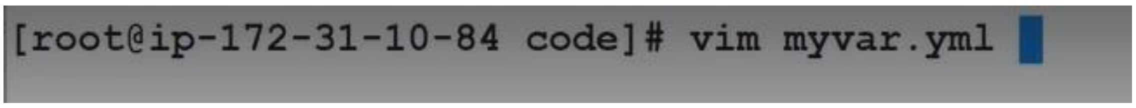


```
- hosts: web
  vars:
    - dosetup: true

  tasks:
  - name: "install web package"
    package:
      name: "httpd"
      state: present
    when:  ( ( ( ansible_os_family == "RedHat") or ( ansible_os_family == "Ubuntu" ) ) and
 ( ansible_memfree_mb >= 200 ) )  and dosetup
~
~
```

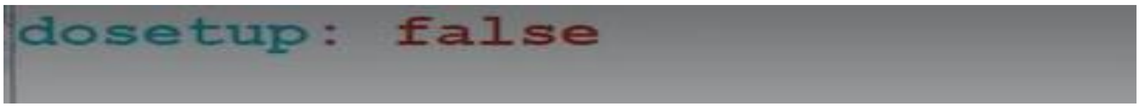
➢ In the playbook instead of specifying the variable, separate file can be created to specify the variable, then this file can be included in the playbook



```
- hosts: web
  vars_files:
  - myvar.yml
```

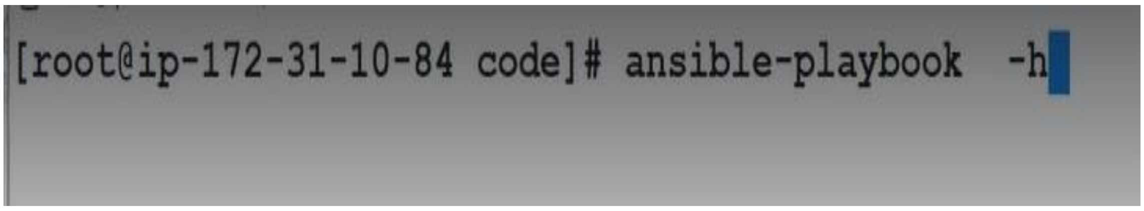


```
[root@ip-172-31-10-84 code]# vim myvar.yml
```

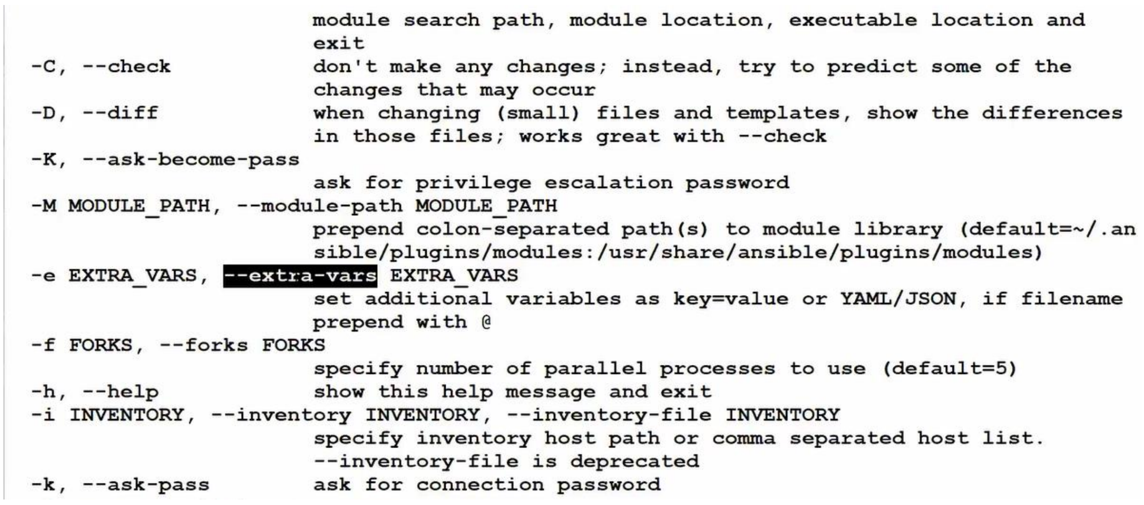


```
dosetup: false
```

➢ The command to see the help “**ansible-playbook –h**”

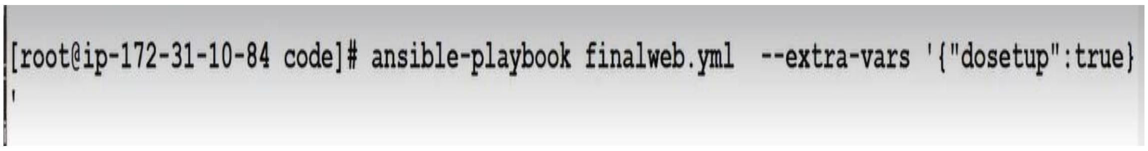


```
[root@ip-172-31-10-84 code]# ansible-playbook  -h
```



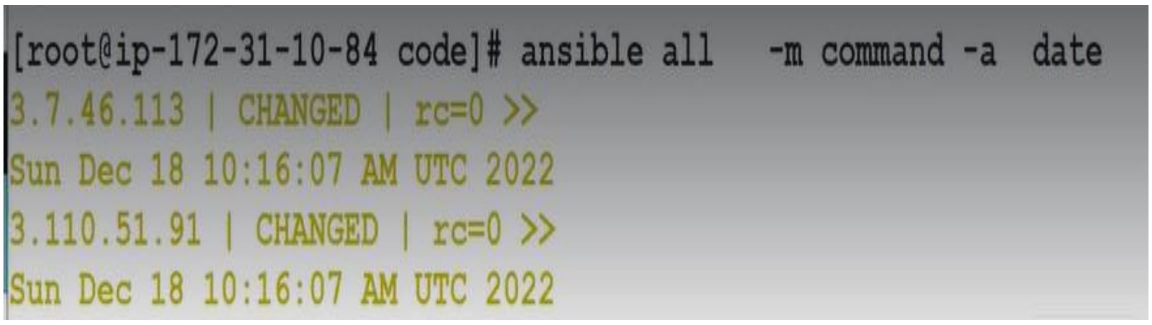
```
                        module search path, module location, executable location and
                        exit
  -C, --check           don't make any changes; instead, try to predict some of the
                        changes that may occur
  -D, --diff            when changing (small) files and templates, show the differences
                        in those files; works great with --check
  -K, --ask-become-pass
                        ask for privilege escalation password
  -M MODULE_PATH, --module-path MODULE_PATH
                        prepend colon-separated path(s) to module library (default=~/.an
                        sible/plugins/modules:/usr/share/ansible/plugins/modules)
  -e EXTRA_VARS, --extra-vars EXTRA_VARS
                        set additional variables as key=value or YAML/JSON, if filename
                        prepend with @
  -f FORKS, --forks FORKS
                        specify number of parallel processes to use (default=5)
  -h, --help            show this help message and exit
  -i INVENTORY, --inventory INVENTORY, --inventory-file INVENTORY
                        specify inventory host path or comma separated host list.
                        --inventory-file is deprecated
  -k, --ask-pass        ask for connection password
```

➢ The keyword **“ - - extra - vars** “ can be used to pass the variables while running the playbook



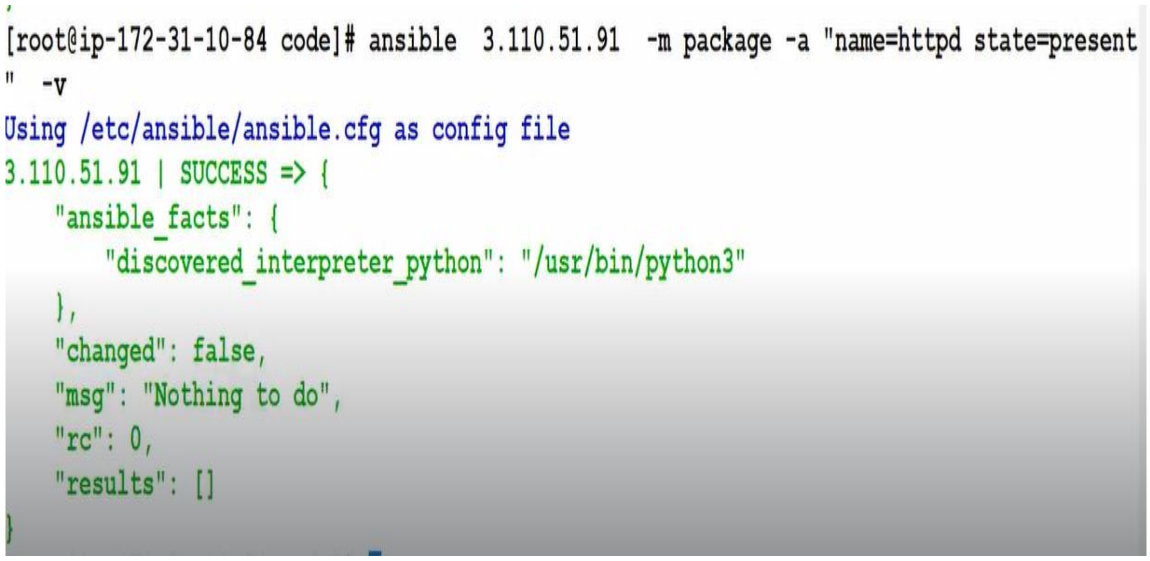
```
[root@ip-172-31-10-84 code]# ansible-playbook finalweb.yml  --extra-vars '{"dosetup":true}
'
```

- The command to run the **date command** on **all** the nodes "**ansible all –m command –a date**". The ansible intelligence comes from the modules, the modules takes input from the facts



```
[root@ip-172-31-10-84 code]# ansible all   -m command -a  date
3.7.46.113 | CHANGED | rc=0 >>
Sun Dec 18 10:16:07 AM UTC 2022
3.110.51.91 | CHANGED | rc=0 >>
Sun Dec 18 10:16:07 AM UTC 2022
```

- The command to see what is happening behind the scene, it gives more information. The "-v" or "-vv" or "-vvv" or "-vvvv" is the verbosity.



```
[root@ip-172-31-10-84 code]# ansible  3.110.51.91  -m package -a "name=httpd state=present
"  -v
Using /etc/ansible/ansible.cfg as config file
3.110.51.91 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "msg": "Nothing to do",
    "rc": 0,
    "results": []
}
```

Important Links –

Hash13 link for Sessions and extra sessions recordings –
https://learning.hash13.com/

How to build Inventory:-
https://docs.ansible.com/ansible/latest/inventory_guide/intro_inventory.html

Ansible Documentation- Copy Module -
https://docs.ansible.com/ansible/latest/collections/ansible/builtin/copy_module.html

Ansible Documentation- Package Module -
https://docs.ansible.com/ansible/latest/collections/ansible/builtin/package_module.html

Ansible Documentation- Service Module –
https://docs.ansible.com/ansible/latest/collections/ansible/builtin/service_module.html

Ansible Documentation – Debug Module –
https://docs.ansible.com/ansible/latest/collections/ansible/builtin/debug_module.html