**Ansible vault:**

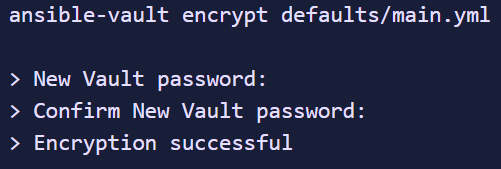
* If one of your tasks requires sensitive information (let’s say the database user and password), it’s a good practice to keep this information encrypted, instead of storing it in plain text.
* Ansible ships with a command line tool called ansible-vault, that allows you to create and manage encrypted files. This way you can commit the encrypted file to your source control and only users with the decryption password will be able to read it.

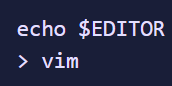
**Examples**:

Encrypt an existing file. You'll need to create an encryption password.

* **ansible-vault encrypt secrets.yml**

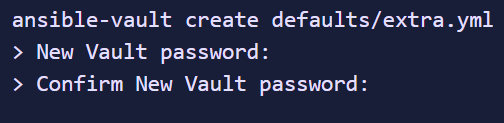
First you will be prompted for a password. After providing a password, the tool will launch whatever editor you have defined with $EDITOR, and defaults to vi. Once you are done with the editor session, the file will be saved as encrypted data.





* **ansible-vault create secrets.yml**

Creates a new, encrypted file. You'll need to create an encryption password.



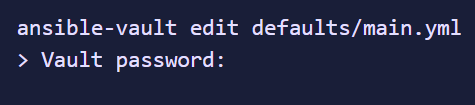
* **ansible-vault decrypt secrets.yml**

Decrypt a file. You'll have to enter password used for encryption.

Use it with caution! Don't leave your files unencrypted.

* **ansible-vault edit secrets.yml**

Edit an encrypted file (uses vim by default, can be overridden by the environment variable $EDITOR)



* **ansible-vault rekey foo.yml bar.yml baz.yml**

we can also change the password of the yaml file with ansible-vault above command.

We can also do it for multiple files at a time.

* **ansible-vault view foo.yml bar.yml baz.yml**

above is the command to view the encrypted yaml files.

**Providing the vault passwords:**

We can also store the password in a file and pass it in the command as below while executing the ansible playbook

* **ansible-playbook --vault-password-file /path/to/my/vault-password-file site.yml**

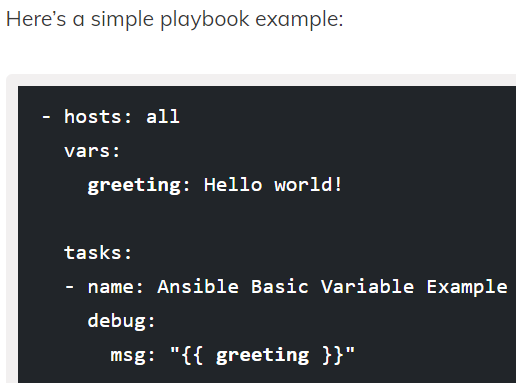
we can also make it ask for vault password while running a playbook as below.

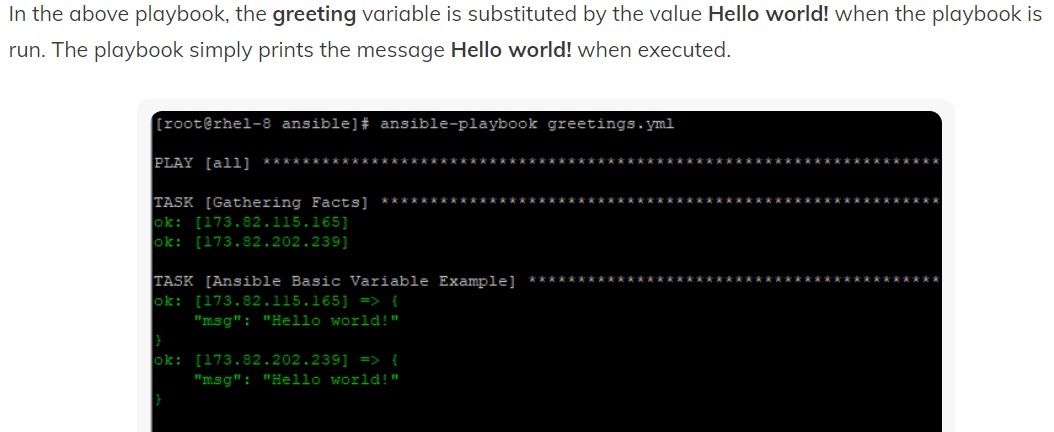
* **ansible-playbook --ask-vault-pass site.yml**

**Variables:**

* Stores the information that varies with other
* If we want to write the playbook which allow user to modify the value, changing in yml file and testing it again is not a good practice. we need to use variables for this dynamic data
* Variables can be defined in inventories and playbooks
* In inventory, we can have group variable and host variables

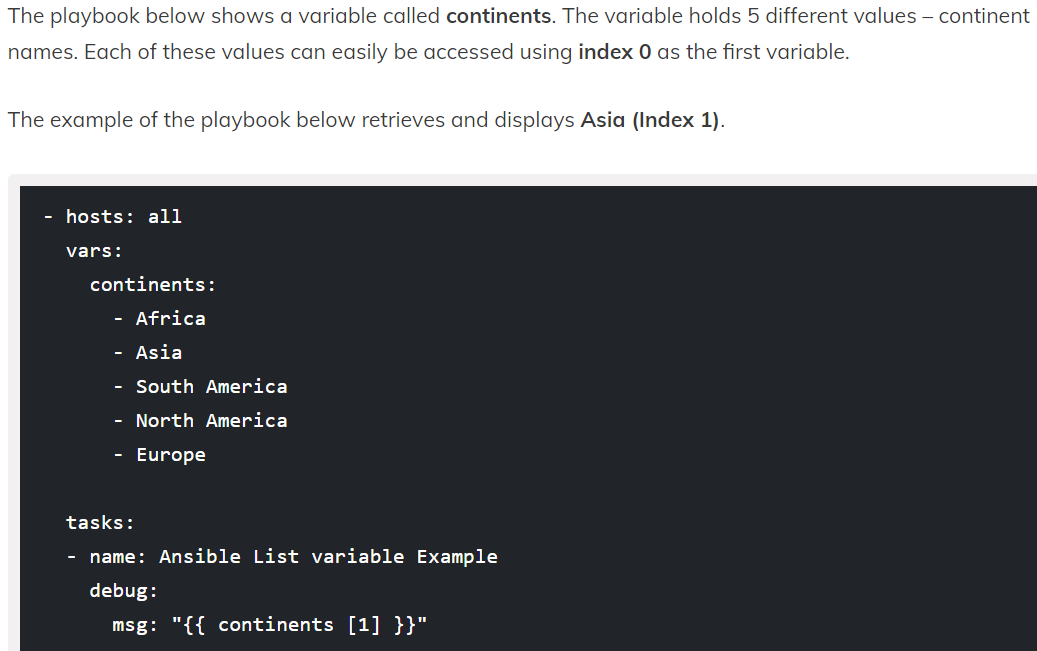
**Variables in playbook:**



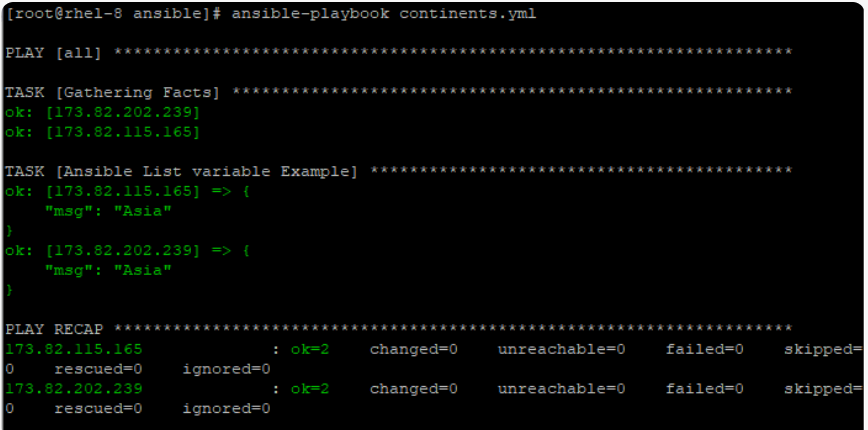


**List or array of variables:**

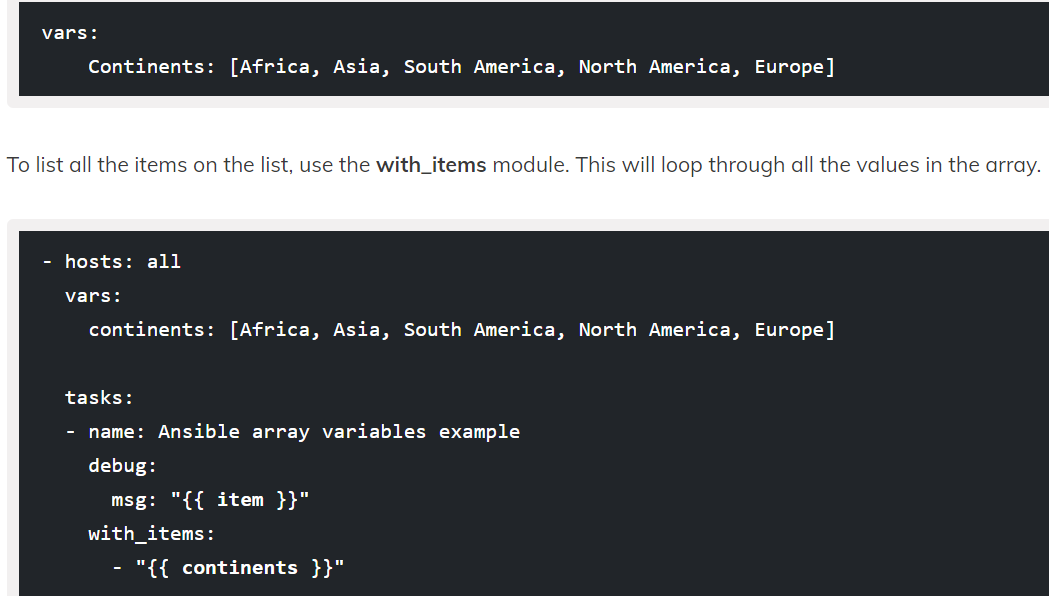
* We can have list or array of variables as below.



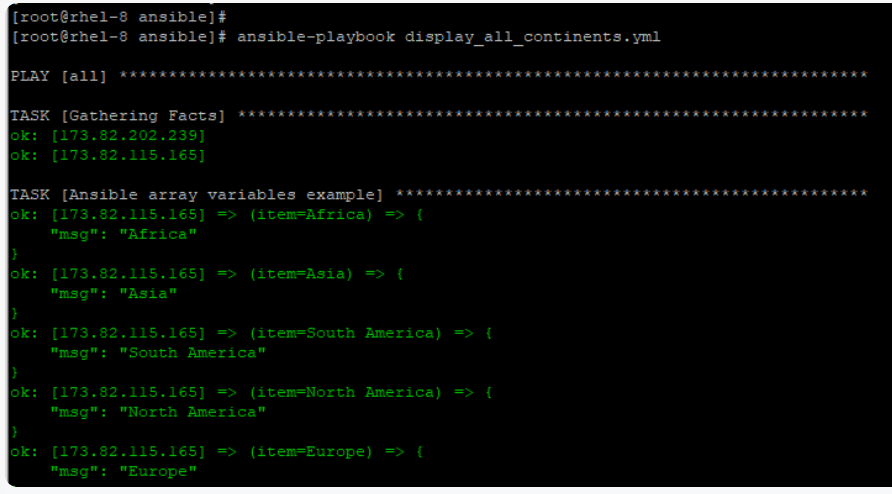
As index stars with 0. Below is the output.



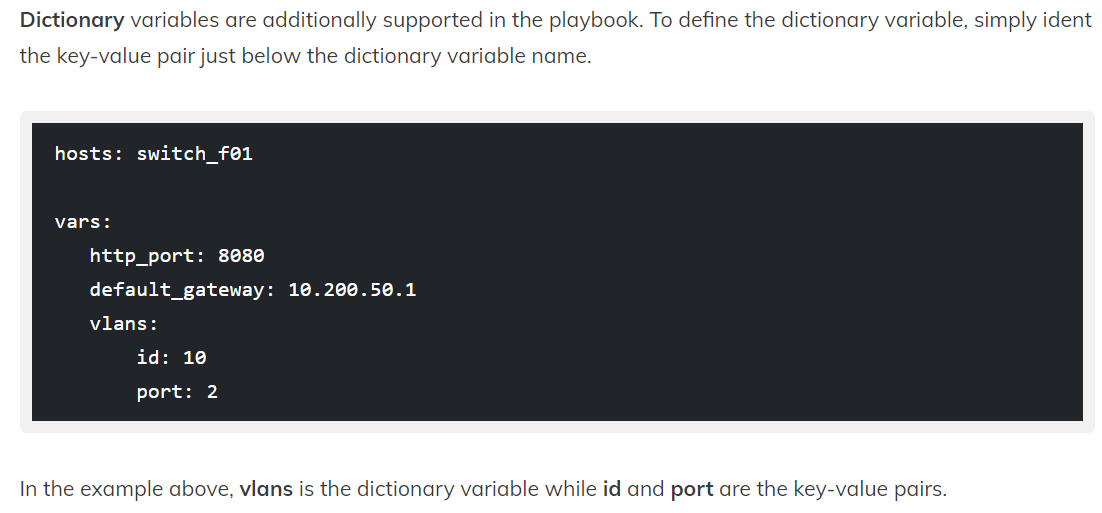
* List can also be added as below.



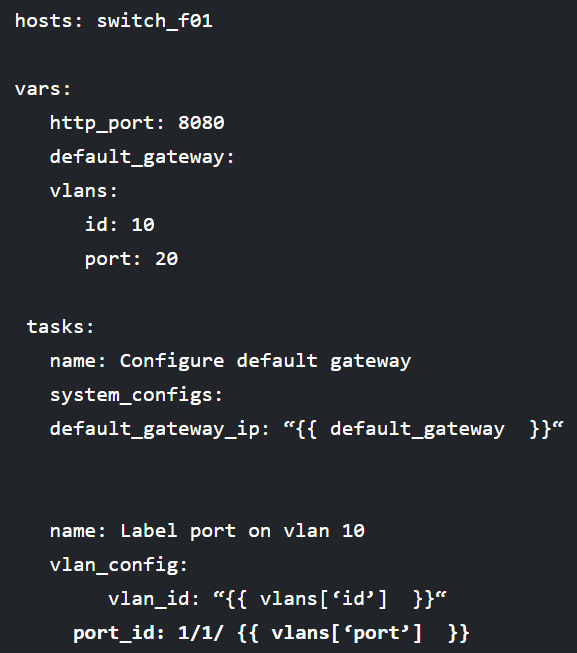
* Below is the output.



**Dictionary type variable:**

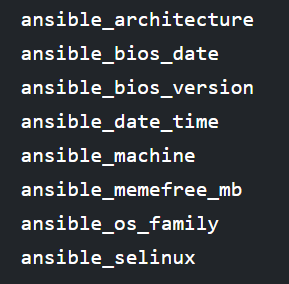


The same can be implemented in yaml file as below.

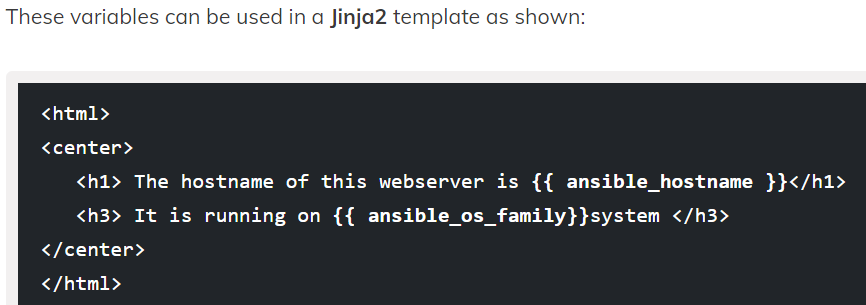


**Special variables:**

We can also use the variables with the gather facts. Below are the few example facts we can use as variables.

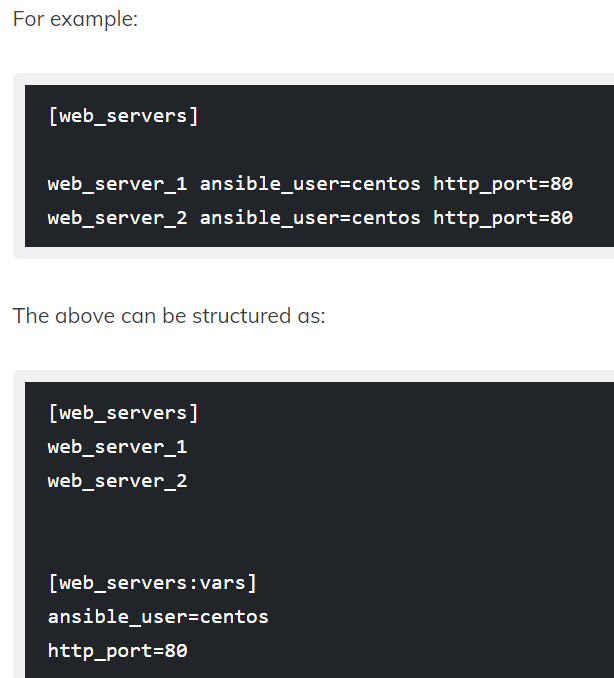


The same can be impleted in jinja2 template also. as below.

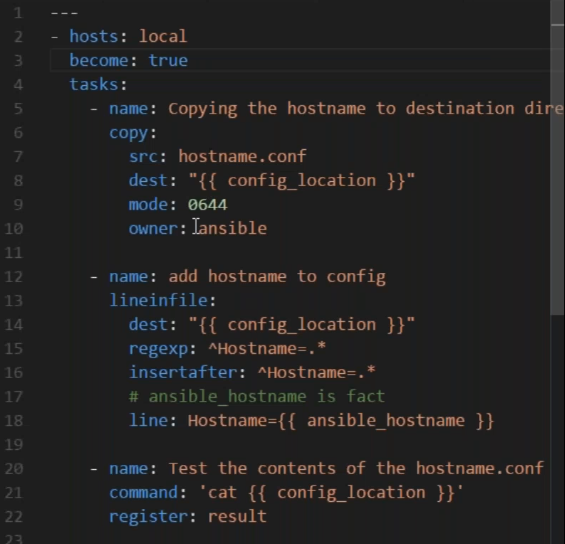


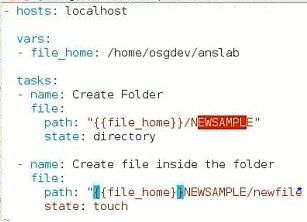
**Inventory variables:**

Below are the two ways we can add variables in inventory file.

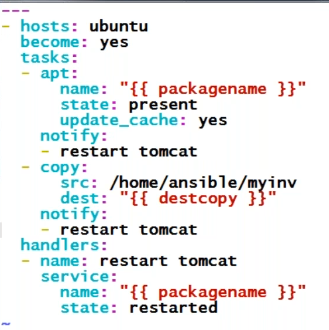


**Examples:**





* We have given name as “{{ packagename }}” where the packagename is tomcat7 in file
* Next time, if we want to install tomcat8, we no need to change yml file, we can simply change variables



Whenever we write a same variable in group\_vars and host\_vars with different values. It runs host\_vars as it has most priority than group