Managing configurations of multiple servers and environments are one of the significant uses of Ansible. But these configuration files may vary for each remote servers or each cluster. But apart from some few parameters, all other settings will be same.

Creating static files for each of these configurations is not an efficient solution. And It will take a lot more time and every time a new cluster is added you will have to add more files. So if there is an efficient way to manage these dynamic values it would be beneficial. This is where Ansible template modules come into play.

A template in Ansible is a file which contains all your configuration parameters, but the dynamic values are given as variables. During the playbook execution, depending on the conditions like which cluster you are using, the variables will be replaced with the relevant values.

You can do much more than replacing the variables though, with the help of Jinj2 templating engine. You can have conditional statements, loops, write macros, filters for transforming the data, do arithmetic calculations, etc.

The template files will usually have the .j2 extension, which denotes the Jinja2 templating engine used.

The variables in a template file will be denoted by the double curly braces, ‘{{ variables }}’.

A Basic Example of Ansible Template Module

At the bare minimum, you need to have two parameters when using the Ansible module.

src: the source of the template file. This can be relative or absolute path.

dest: the destination path on the remote server

In the following task. I am using the template module on the hello\_world.j2 file. Double braces cover the variables.

- hosts: all

vars:

variable\_to\_be\_replaced: 'Hello world'

inline\_variable: 'hello again'

tasks:

- name: Ansible Template Example

template:

src: hello\_world.j2

dest: /Users/mdtutorials2/Documents/Ansible/hello\_world.txt

hello\_world.j2

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

{{ variable\_to\_be\_replaced }}

This line won't be changed

Variable given as inline - {{ inline\_variable }} - :)

output - hello\_world.txt

------

Hello world

This line won't be changed

Variable given as inline - hello again - :)

mdtutorials2$ ls -lrt hello\_world.txt

-rw-r--r--  1 root  wheel  81 Oct 16 07:23 hello\_world.txt

As you can see, both variables in the hello\_world are replaced by their values.

Note: If no file name is given in the destination parameter, and only the directory path, then the name of the file will be template file name. In the above example, it would have been hello\_world.j2.

Additional attributes of the template module

There are other modules of template module which we can use to change some default behaviours

* force – If the destination file already exists, then this parameter decides whether it should be replaced or not. By default, the value is ‘yes’. So if there is any difference between the rendered source file and the destination file, destination file would be replaced. If you do not want this behaviour, set the value to ‘no’.
* Mode – If you want to set the permissions for the destination file explicitly, then you can use this parameter.
* backup – If you want a backup file to be created in the destination directory, you should set the value of the backup parameter to ‘yes’. By default, the value is ‘no’. The backup file will be created every time there is a change in the destination directory. A timestamp would be appended to the filename. So if I changed the ‘template.j2’ file in the last example two time and also changed the backup parameter to yes, then I get the following two files after two runs.
* hello\_world.txt.30956.2017-10-16@07:34:53~
* hello\_world.txt.31132.2017-10-16@07:35:35~

Using ‘for’ loop structure inside Ansible template

One of the main program expression we usually use is the ‘for’ loop. It can be used to iteratively go through the values of a list, dictionary etc.

It is possible to use this in ansible templates also using the jinja2 format.

In the following example, I am looping through the value 0 to 2 using the python range function. On each iteration, a line with the variable is printed.

Jinja\_loop.j2

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

Ansible template for loop example

{% for i in range(3)%}

This is the {{ i }}th variable

{% endfor %}

output

------

mdtutorials2$$ cat hello\_world.txt

Ansible template for loop example

This is the 0th variable

This is the 1th variable

This is the 2th variable

But, in the above example, each iteration is printed on new lines. It is because it is retaining the whitespaces. But in some scenarios, we may want to remove the white spaces. So, how can we do that?

For such scenarios, we can use the minus sign(-) to manually strip the whitespaces including newlines. For example, in the following task, I am adding the ‘-‘ sign to the end of the ‘for’ expression. This will remove the white spaces at the end of the block.  
The resulting output shows all the variables on the same line.

Removing the whitespaces in ansible template

{% for i in range(3) %}

variable {{ i }}

{%- endfor %}

output

------

mdtutorials2 $ cat hello\_world.txt

Removing the whitespaces in ansible template

variable 0 variable 1 variable 2

For more information on how to control the whitespaces checkout the [jinja2 documentation](http://jinja.pocoo.org/docs/2.9/templates/#whitespace-control).

Now, let us see how to use ansible variables like list or dictionary in the templates files.

Using list variables in Ansible templates

In the below task, I am looping over the *list1* variable in the template, using the for loop structure. Note that, after each iteration, a *new line* is also added. So the three list items will be in three lines.

- hosts: loc

vars:

list1: ['Template iterate','Template loop','Template item']

tasks:

- name: Ansible template loop example.

- template:

src: templates\_example2.j2

dest: /home/mdtutorials2/output.txt

mode: 0777

templates\_example2.j2

This is an example of template module loop with a list.

{% for item in list1 %}

{{ item }}

{% endfor %}

output.txt

This is an example of using with\_items in template module loop with a list.

Template iterate

Template loop

Template item

Ansible template with\_items for multiple files

You can use the with\_items parameter on a dictionary to render multiple files. We are using the dictionary since the source and destination will be different for each template.

In the following example, I am rendering three templates, each with different source and destination.

- hosts: loc

tasks:

- name: Ansible template with\_items example.

template:

src: "{{ item.src }}"

dest: "{{ item.dest }}"

mode: 0777

with\_items:

- {src: 'ex.j2',dest: '/home/dnpjose/ex\_rew1.txt'}

- {src: 'ex2.j2',dest: '/home/dnpjose/ex\_rew2.txt'}

- {src: 'ex3.j2',dest: '/home/dnpjose/ex\_rew3.txt'}

4. Ansible template comment example

You might give a lot of comments in your code for clarity. You can also give the comments in the template file. But sometimes you won’t want it to appear in the rendered file. You can do this by giving Jinja2 style comments by enclosing the comments within **{# … #}**.