Jinja templates in Ansible can be very powerful. They can also be a leading contributor to hair loss. Why? In some ways it comes down to documentation, a mixing of languages (YAML, Python, Jinja2), and variables.

During a recent consulting project with a customer, focused on network automation, we embarked on a journey to re-evaluate how routers were provisioned. A significant part of this initiative was to dynamically create configuration templates for routers, based on variable input. In developing the j2 (the Jinja2 templating language) logic to do things like calculate bandwidth figures, we ran into some limitations. Mainly the ability to have a variable's value accessible outside of the loop that is currently being run.

> *Please keep in mind that it is not possible to set variables inside a block and have them show up outside of it. This also applies to loops.*
> *— http://jinja.pocoo.org/docs/2.9/templates/#assignments*

So, let's dive right into it. First, I'll describe the issue in detail (feel free to follow along on your own Ansible install).

Given the following playbook structure:

```
├── output.txt
├── varloop.j2
├── varloop.yml
└── vars.yml
```

Hosts is our inventory file and is simply our localhost that we are running Ansible on.

Our vars.yml file looks like this:

```
---
  people:
    - name: Mike
      fav_colour: Blue
    - name: Kyle
      fav_colour: Yellow
    - name: Shea
      fav_colour: Blue
    - name: Aly
      fav_colour: Yellow
    - name: Daniyal
      fav_colour: Yellow
    - name: Tim
      fav_colour: Orange

  colours:
    - name: Blue
      things:
        - Sky
        - Sea
        - Jeans
    - name: Yellow
      things:
        - Egg yolk
        - Taxi
        - Banana
```

```
      - Lemon
      - Sun
  - name: Orange
    things:
      - Pumpkin
      - Basketball
      - Carrots
      - Oranges
```

As you can see we have 2 dictionary variables defined. One containing a list of people with their favourite colours, and a second one containing a list of colours with things that happen to be of those colours. Now we write a simple playbook that will call on a Jinja template we will write. The playbook is called `varloop.yml`:

```
---
  - name: Demonstrating variables in Jinja2 Loops
    hosts: localhost
    connection: local
    vars_files:
      - vars.yml
    gather_facts: no
    tasks:
      - name: Create the Jinja2 based template
        template: src=./varloop.j2 dest=./output.txt
```

The playbook simply uses the variable file we specified and calls the template module in a task to build a file called output.txt from a j2 template. Now for the j2 template itself `varloop.j2`:

```
{% for colour in colours %}
  Colour number {{ loop.index }} is {{ colour.name }}.
{% set colour_count = 0 %}
{% for person in people if person.fav_colour == colour.name %}
{% set colour_count = colour_count + 1 %}
{% endfor %}
  Currently {{ colour_count }} people call {{ colour.name }} their favourite.
  And the following are examples of things that are {{ colour.name }}:
{% for item in colour.things %}
    - {{ item }}
{% endfor %}

{% endfor %}
```

In this j2 template we are attempting the following:

- looping over the list of colours and attempting to run a nested loop inside, that counts the number of people who's favourite is the colour of the current loop iteration
- listing all of the things that are of the colour of the current loop iteration

We get an output that is not what we expect. We are NOT able to extract the value of the *colour_count* variable outside of the inner loop where we are performing the count. See the output here:

```
Colour number 1 is Blue.
  Currently 0 people call Blue their favourite.
```

```
And the following are examples of things that are Blue:
  - Sky
  - Sea
  - Jeans

Colour number 2 is Yellow.
Currently 0 people call Yellow their favourite.
And the following are examples of things that are Yellow:
  - Egg yolk
  - Taxi
  - Banana
  - Lemon
  - Sun

Colour number 3 is Orange.
Currently 0 people call Orange their favourite.
And the following are examples of things that are Orange:
  - Pumpkin
  - Basketball
  - Carrots
  - Oranges
```

In our example we see that because we can't call the variable outside of the inner loop, the counting didn't work. A quick modification to your `/etc/ansible.cfg` file and a small change to your template, and we can get this working. First add the following line to your `ansible.cfg`:

```
[defaults]
jinja2_extensions = jinja2.ext.do,jinja2.ext.i18n
```

Then modify your `varloop.j2` file like this:

```
{% for colour in colours %}
  Colour number {{ loop.index }} is {{ colour.name }}.
{% set colour_count = 0 %}
{% for person in people if person.fav_colour == colour.name %}
{% set colour_count = colour_count + 1 %}
{% do colour.update({'people_count':colour_count}) %}
{% endfor %}
  Currently {{ colour.people_count }} people call {{ colour.name }} their favourite.
  And the following are examples of things that are {{ colour.name }}:
{% for item in colour.things %}
    - {{ item }}
{% endfor %}

{% endfor %}
```

Notice the **"do"** block in the updated template. This block allows us to use the **update** function to update an element in a dictionary variable. (we unlocked this with that extensions line in our `ansible.cfg` file) In our case we are updating the colour of the current loop iteration to include a new key/value pair called **people_count**. So in each loop iteration of the **colours** variable, we are able to now add a new element that contains the number of people who deem this their favourite colour.

```
  Colour number 1 is Blue.
    Currently 2 people call Blue their favourite.
    And the following are examples of things that are Blue:
      - Sky
      - Sea
      - Jeans

    Colour number 2 is Yellow.
    Currently 3 people call Yellow their favourite.
    And the following are examples of things that are Yellow:
      - Egg yolk
      - Taxi
      - Banana
      - Lemon
      - Sun

    Colour number 3 is Orange.
    Currently 1 people call Orange their favourite.
    And the following are examples of things that are Orange:
      - Pumpkin
      - Basketball
      - Carrots
      - Oranges
```

Notice in our output after running Ansible with our updated template file, the numbers are properly counted.

Hope this helps you out in your template writing. Jinja2 can be a pain, but ultimately a very powerful tool. If you get stuck … reach out! There is more to come!

//take the first step

Tagged:

red-hat **ansible** **jinja2** **templates** **variables**

//share this

twitter google reddit linkedin email

//comments