Lab: Playbooks introductions

Overview

Both YAML and Jinja2 are very common formats used across many modern devops tools. In this lab, we will explore the YAML and Jinja2 and use them to define simple ansible playbooks

Lab: Playbooks introductions

Instructions

Read this lab like a book, all text is there for a reason!

"→" denotes an action you must take

Use your favorite editor to edit files within the console. I suggest VI, nano, or emacs.

```
White boxes with black text denote commands and file contents
```

```
Black boxes with green text denote example output
```

Task 1: Install Tools

Step 1: Install a YAML linter

→ Run the following command in the terminal on your WorkSpace.

```
pip install --user yaml-lint
```

Step 2: Install jinja2 CLI

→ Run the following command in the terminal on your WorkSpace.

```
pip install --user jinja2-cli
```

Task 2: Extend a YAML file

Ansible has great docs describing YAML syntax here. We are going to write some YAML files that describe the Teenage Mutant Ninja Turtles.

Step 1: Copy an existing YAML file

→ Make a new directory to work in

```
mkdir ~/yaml-lab
cd ~/yaml-lab
```

→ Copy our YAML into leo.yaml

```
---
name: Leonardo
 nickname: Leo
weapons:
  - "sword"
  - "throwing stars"
color: blue
```

→ Lint our YAML

```
yamllint leo.yaml
```

You should see an error, as the YAML is not valid.

Step 2: Fix the issue

→ fix what's wrong

→ Check your fix, you should see no output

```
yamllint leo.yaml
```

Task 3: Turn your YAML into a Jinja2 template

Lets make our yaml file generic so we can pass in details of any ninja turtle.

Step 1: Add variables

→ Copy leo.yml into a new file

```
cp leo.yml tmnt.yml.j2
```

→ Edit the new template so if we execute the following command we get the correct output.

The command:

```
jinja2 tmnt.yml.j2 -D name=donatello -D nickname=donnie -D color=purple -D
primary_weapon=staff
```

should produce:

```
---
name: donatello
nickname: donnie
weapons:
  - "staff"
  - "throwing stars"
color: purple
```

Once the correct output is produced, move on.

```
jinja2 tmnt.yml.j2 -D name=donatello -D nickname=donnie -D color=purple -D
primary_weapon=staff && \
jinja2 tmnt.yml.j2 -D name=leonardo -D nickname=leo -D color=blue -D primary_weapon=sword &&
\
jinja2 tmnt.yml.j2 -D name=raphael -D nickname=raph -D color=red -D primary_weapon=sai && \
jinja2 tmnt.yml.j2 -D name=michaelangelo -D nickname=mikey -D color=orange -D
primary_weapon=nunchucks
```

Task 4: Handle formatting

Step 1: Update your template to capitalize "name" and "nickname"

We have decided that we should force proper formatting, such as capitalized names. One way to accomplish this is through the use of filters (docs). Use the capitalize filter.

→ Make it so if we execute the following command we get the correct output.

The command:

```
jinja2 tmnt.yml.j2 -D name=donatello -D nickname=donnie -D color=purple -D
primary_weapon=staff
```

should produce:

```
---
name: Donatello
nickname: Donnie
weapons:
  - "staff"
  - "throwing stars"
color: purple
```

Task 5: Update a playbook

Now that you have extensive hands-on experience with YAML and jinja, :P, let's start editing Ansible playbooks.

Step 1: Download files

→

```
cd ~
git clone https://github.com/ameade/ansible-labs.git
```

Step 2: Print variables with the debug module

Debug module - https://docs.ansible.com/ansible/latest/modules/debug_module.html

→ Change directories

```
cd ~/ansible-labs/playbooks_intro/
```

→ Update playbook1.yml to output "The name is Bond; James Bond." Using the first_name and last_name variables that are defined.

→ Check the syntax of your changes

```
ansible-playbook playbook1.yml --syntax-check
```

→ Run the playbook

```
ansible-playbook playbook1.yml
```

```
...
TASK [debug]
*************************************************************************************
ok: [localhost] => {
    "msg": "The name is Bond; James Bond."
}
```

Step 3: Find the lowest number

→ Update playbook2.yml to output the lowest number

Step 4: Create a file containing a random number

→ Update playbook3.yml to create a file containing a random number

→ Check that the file was created and contains a number

Step 5: Combine our playbooks

→ Make a single playbook named playbook4.yml that performs all the tasks of the other three playbooks