IT Automation with

Agenda

- Introduction to Ansible roles

Ansible roles

Introduction

- Usually as playbooks grow and make use of other resource such as variables templates files, etc it might increase complexity. Here's where roles come to the rescue!
- Roles are a way to bundle playbooks, templates, vars files and handlers together under a fixed and well known directory structure. Grouping content by roles, also make sharing ansible code easier.
- Some benefits of using Ansible roles: you can write playbook for a specific purpose such as: 'webserver role', 'firewall role' and having a well known structure makes easier code integration when working across teams.

Ansible roles

Simple Loops

Sub-directory	Description
defaults	Default values for variables that can be easily overwritten (By default it uses the main.yml file)
files	Files that might be needed by role's tasks
handlers	The main.yml file will contain handlers used by different role's tasks
meta	The main.yml file will contain meta-data information such as author name and also may have any role dependencies
tasks	The main.yml file will contain tasks that will be executed upon role invocation.
templates	Templates that might be referenced by role's tasks
vars	The main.yml file will have variables that are not expected to be overwritten elsewhere.

Ansible loops

Roles variables

- As mentioned before, roles variable will be at vars/main.yml. These variables have “high” priority and can not be overwritten by inventory variables.
- In the other hand, default variables will be at defaults/main.yml. These variables have the lowest precedence value.
- Regardless of where a certain variables has been defined, it can be accessed in role’s tasks with the usual syntax: {{ var }}
- **Important**: do **not** set variables in both places!

Ansible roles

How to use roles?

- The classical way of invoking roles in a play is through the ***roles:*** option

```
---
- hosts: webservers
  roles:
     - common
     - webservers
```

- When used in this manner, the order of execution of your playbook is:

1) Any ***pre_tasks*** defined in the play.
2) Any handlers triggered so far will be run.
3) Each role listed in ***roles*** will be executed in turn.
4) Any ***tasks*** defined in the play.
5) Any handlers triggered so far will be run.
6) Any ***post_task*** defined in the play will be run.
7) Any handlers triggered so far will be run.

Ansible roles

How to use roles?

- As of Ansible 2.4 you can also call roles within tasks section using either *import_role* or *include_role*:

```
---

- hosts: webservers
  tasks:
  - debug:
      msg: "before we run our role"
  - import_role:
      name: example
  - include_role:
      name: example
  - debug:
      msg: "after we ran our role"
```

- Roles can be invoked either with a relative or a fully qualified path:

```
---

- hosts: webservers
  roles:
    - { role: '/path/to/my/roles/common' }
```

Ansible roles

Roles parameters

- Roles can also accept parameters:

```
---

- hosts: webservers
  roles:
     - common
     - { role: foo_app_instance, dir: '/opt/a', app_port: 5000 }
     - { role: foo_app_instance, dir: '/opt/b', app_port: 5001 }
```

Or using the newer syntax:

```
---

- hosts: webservers
  tasks:
  - include_role:
       name: foo_app_instance
    vars:
      dir: '/opt/a'
      app_port: 5000
  ...
```

Ansible roles

Importing a role conditionally

- A role can be conditionally imported / included when using the newer syntax

```
---

- hosts: webservers
  tasks:
  - include_role:
      name: some_role
    when: "ansible_os_family == 'RedHat'"
```

- You can also apply tags to the roles, either using classical or newer syntax:

```
---

- hosts: webservers
  roles:
    - { role: foo, tags: ["bar", "baz"] }
```

```
---

- hosts: webservers
  tasks:
  - import_role:
      name: foo
    tags:
    - bar
    - baz
```

Ansible roles

Role dependencies

- Roles dependencies are defined in *meta/main.yml* file.
- They allow you to automatically pull in other roles when using a role.
- This file should contain a list of roles and parameters to insert before the specified role:

```
---
dependencies:
  - { role: common, some_parameter: 3 }
  - { role: apache, apache_port: 80 }
  - { role: postgres, dbname: blarg, other_parameter: 12 }
```

- Role dependencies must use classical role definition syntax.
- They are are always executed before the role that includes them, and may be recursive.

Ansible roles

Role duplication

- Ansible will only allow a role to execute once, even if defined multiple times if the parameters defined for the role are not different on each definition:

```
---
- hosts: webservers
  roles:
  - foo
  - foo
```

- However if roles are invoked with different parameters in each role definition then they could be run more than once:

```
---
- hosts: webservers
  roles:
  - { role: foo, message: "first" }
  - { role: foo, message: "second" }
```

- An alternative way to achieve the same would be to use ***allow_duplicates: true*** in meta/main.yml file of the role.

Ansible roles

Order of execution

- Usually tasks from roles are executed before playbook tasks (by default). However this behavior can be altered and have playbook tasks be executed either before of after roles tasks.
- A *pre_tasks* block contains a list of tasks that will run before roles tasks.
- A *post_tasks* block contains a list of tasks that will run after roles tasks.

```
- hosts: all
  - pre_tasks:
      - debug: msg="esto se ejecuta primero"
  - roles:
      - common
      - webserver
  - tasks:
      - debug: msg="esto después de los roles"
  - post_tasks:
      - debug: msg="esto después de todo".
```

Demo

Ansible Galaxy

Ansible Galaxy

Introduction

- Ansible Galaxy (https://galaxy.ansible.com) is a public repository for finding and sharing community developed roles. It has plenty of ready-to-use roles that can be used to quickly start your automation tasks.
- You can easily share your own Ansible developed roles by just authenticating using your GitHub account.
- Roles can be search based on several criteria: keywords, platforms, tags, etc either in the aforementioned website or directly through the ansible-galaxy cli
- The ansible-galaxy cli works in a similar way to popular package management tools such as yum, apt, etc. It allows you to search modules based on platform, author and galaxy-tags.

```
→ ansible-galaxy search 'nginx' --platform Ubuntu

Found 798 roles matching your search:

Name                                   Description
----                                   -----------
1davidmichael.ansible-role-nginx       Nginx installation for Linux, FreeBSD and OpenBSD.
1it.sudo                               Ansible role for managing sudoers
```

Ansible Galaxy

Introduction

- Since Galaxy Project (https://github.com/ansible/galaxy) is open source you may want to run your own internal Galaxy repo. For ansible-galaxy cli to use the custom API endpoint, just pass the --server option pointing to your custom repo.
- You can easily share your own Ansible developed roles by just authenticating using your GitHub account.
- The ***ansible-galaxy info*** sub-command is used to show more information about a specific role:

```
→ ansible-galaxy info geerlingguy.nginx

Role: geerlingguy.nginx
        description: Nginx installation for Linux, FreeBSD and OpenBSD.
        active: True
        commit: ba806a5c1a2f946bc3be4611b15e28ac1c555279
        commit_message: Update tests for optimum efficiency.
        commit_url: https://api.github.com/repos/geerlingguy/ansible-role
        company: Midwestern Mac, LLC
```

Ansible Galaxy

How to install galaxy roles?

- Example:

```
ansible-galaxy install -r requirements.yml
```

```
# from galaxy
- src: yatesr.timezone

# from GitHub
- src: https://github.com/bennojoy/nginx

# from GitHub, overriding the name and specifying a specific tag
- src: https://github.com/bennojoy/nginx
  version: master
  name: nginx_role

# from a webserver, where the role is packaged in a tar.gz
- src: https://some.webserver.example.com/files/master.tar.gz
  name: http-role

# from Bitbucket
- src: git+http://bitbucket.org/willthames/git-ansible-galaxy
  version: v1.4

# from Bitbucket, alternative syntax and caveats
- src: http://bitbucket.org/willthames/hg-ansible-galaxy
  scm: hg

# from GitLab or other git-based scm
- src: git@gitlab.company.com:mygroup/ansible-base.git
  scm: git
  version: "0.1"  # quoted, so YAML doesn't parse this as a floating-point value
```

Ansible Galaxy

How to manage galaxy roles?

- Use the *ansible-galaxy list* command to list current installed galaxy roles.
- To remove an installed galaxy role use the *ansible-galaxy remove <role>* sub-command
- It's also possible to create standard role skeleton from *ansible-galaxy init <role-name>*

```
defaults/
    main.yml
files/
handlers/
    main.yml
meta/
    main.yml
templates/
tests/
    inventory
    test.yml
vars/
    main.yml
```

Questions?