Roles:
scaling up your playbooks

Marica Antonacci - INFN BARI
marica.antonacci@ba.infn.it

Corso Ansible/Foreman/Puppet
5-8 June 2018, INFN BARI

Why roles

Roles are good for

- Managing the complexity: decompose complex jobs into smaller pieces
- Organizing multiple, related tasks and encapsulating data
- Compose reusable ansible content

Roles provide a framework for fully independent, or interdependent, collections of variables, tasks, files, templates, and modules.

Roles vs Playbooks

- Each role is typically limited to a particular theme or desired end result, with all the necessary steps to reach that result either within the role itself or in other roles listed as dependencies.
- Roles themselves are not playbooks. There is no way to directly execute a role.
- Roles have no setting for which host the role will apply to.
- Top-level playbooks are the glue that binds the hosts from your inventory to roles that should be applied to those hosts.

Where does ansible look for your roles?

Ansible will look for roles in the roles directory alongside your . playbooks. It will also look for systemwide roles in `/etc/ansible/roles`.

You can customize the system-wide location of roles by setting the **`roles_path`** setting in the defaults section of your `ansible.cfg` file:

```
[defaults]
roles_path = ~/ansible_roles
```

Role Structure

Roles add a bit of "magic" to Ansible: they assume a specific file organization.

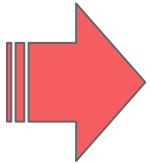
```
roles
  rolename
   - files
   - handlers
   - meta
   - templates
   - tasks
   - vars
```

The role name is the directory name within `roles/`

Within each directory, Ansible will search for and read any Yaml file called **main.yml** automatically

From playbook to role - an example: apache role

```
---
- hosts: all
  become: yes
  tasks:
  - name: Install httpd Package
    yum: name=httpd state=latest
  - name: Copy httpd configuration file
    copy: src=/data/httpd.conf dest=/etc/httpd/conf/httpd.conf
  - name: Copy index.html file
    copy: src=/data/index.html dest=/var/www/html
    notify:
    - restart apache
  - name: Start and Enable httpd service
    service: name=httpd state=restarted enabled=yes
  handlers:
  - name: restart apache
    service: name=httpd state=restarted
```



```
[root@marica01 ~]# tree apache/
apache/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

Role: Tasks

The task file is the main meat of a role.

If roles/<role_name>/tasks/main.yaml exists, all the tasks therein and any other files it includes will be embedded in the play and executed.

This allows you to split a large number of tasks into separate files, and use other features of task includes.

From playbook to role: extracting the tasks

Starting from this playbook:

```
---
- hosts: all
  become: yes
  tasks:
  - name: Install httpd Package
    yum: name=httpd state=latest
  - name: Copy httpd configuration file
    copy: src=/data/httpd.conf dest=/etc/httpd/conf/httpd.conf
  - name: Copy index.html file
    copy: src=/data/index.html dest=/var/www/html
    notify:
    - restart apache
  - name: Start and Enable httpd service
    service: name=httpd state=restarted enabled=yes
  handlers:
  - name: restart apache
    service: name=httpd state=restarted
```

```
[root@ansible apache]# cat tasks/main.yml
---
# tasks file for apache
- import_tasks: install.yml
- import_tasks: configure.yml
- import_tasks: service.yml
```

```
[root@ansible apache]# cat tasks/install.yml
---
- name: Install httpd Package
  yum: name=httpd state=latest

[root@ansible apache]# cat tasks/configure.yml
---
- name: Copy httpd configuration file
  copy: src=files/httpd.conf dest=/etc/httpd/conf/httpd.conf
- name: Copy index.html file
  copy: src=files/index.html dest=/var/www/html
  notify:
  - restart apache

[root@ansible apache]# cat tasks/service.yml
---
- name: Start and Enable httpd service
  service: name=httpd state=restarted enabled=yes
```

Handlers

Similar to tasks, handlers are automatically loaded from roles/<role_name>/handlers/main.yaml, if the file exists.

These handlers can be referenced by any task within the role, or by any tasks within any other role that lists this role as a dependency.

From playbook to role: extracting the handler

From the original playbook:

```
---
- hosts: all
  become: yes
  tasks:
  - name: Install httpd Package
    yum: name=httpd state=latest
  - name: Copy httpd configuration file
    copy: src=/data/httpd.conf dest=/etc/httpd/conf/httpd.conf
  - name: Copy index.html file
    copy: src=/data/index.html dest=/var/www/html
    notify:
    - restart apache
  - name: Start and Enable httpd service
    service: name=httpd state=restarted enabled=yes
  handlers:
  - name: restart apache
    service: name=httpd state=restarted
```

```
[root@ansible apache]# cat handlers/main.yml
---
# handlers file for apache
- name: restart apache
  service: name=httpd state=restarted
```

Variables

There are two types of variables that can be defined in a role:

- **role variables**, loaded from roles/<role_name>/vars/main.yaml
- **role defaults**, which are loaded from roles/<role_name>/defaults/main.yaml.

The difference between vars and defaults has to do with **precedence** order: role defaults are the lowest order variables. Role defaults can be thought of as place holders for actual data, a reference of what variables a developer may be interested in defining with site-specific values. Role variables, on the other hand, have a higher order of precedence. They are used for example for system-specific constants that don't change much.

Modules

A role can include custom modules.

Modules can be loaded from roles/<role_name>/library/ and can be used by any task in the role, or any later role.

Modules provided in this path will override any other copies of the same module name anywhere else on the file system, which can be a way of distributing added functionality to a core module before the functionality has been accepted upstream and released with a new version of Ansible.

Dependencies

Roles can express a dependency upon another role.

When Ansible processes a role for a set of hosts, it will first look for any dependencies listed in roles/<role_name>/meta/main.yaml. If any are defined, those roles will be processed and the tasks within will be executed (after also checking for any dependencies listed within) until all dependencies have been completed before starting on the initial role tasks.

Files

Task and handler modules can reference files relatively within `roles/<role_name>/files/`.

The filename can be provided without any prefix and will be sourced from `roles/<role_name>/files/<file_name>`.

Relative prefixes are allowed as well, in order to access files within subdirectories of `roles/<role_name>/files/`.

Modules such as **template**, **copy**, and **script** may take advantage of this.

From playbook to role: moving the config files

Starting from this playbook:

```
---
- hosts: all
  become: yes
  tasks:
  - name: Install httpd Package
    yum: name=httpd state=latest
  - name: Copy httpd configuration file
    copy: src=/data/httpd.conf dest=/etc/httpd/conf/httpd.conf
  - name: Copy index.html file
    copy: src=/data/index.html dest=/var/www/html
    notify:
    - restart apache
  - name: Start and Enable httpd service
    service: name=httpd state=restarted enabled=yes
  handlers:
  - name: restart apache
    service: name=httpd state=restarted
```

```
[root@ansible apache]# tree files/
files/
├── httpd.conf
└── index.html
```

```
[root@ansible apache]# cat tasks/configure.yml
---
- name: Copy httpd configuration file
  copy: src=files/httpd.conf dest=/etc/httpd/conf/httpd.conf
- name: Copy index.html file
  copy: src=files/index.html dest=/var/www/html
  notify:
  - restart apache
```

Templates

Templates used by the **`template`** module can be referenced relatively within roles/<role_name>/templates/.

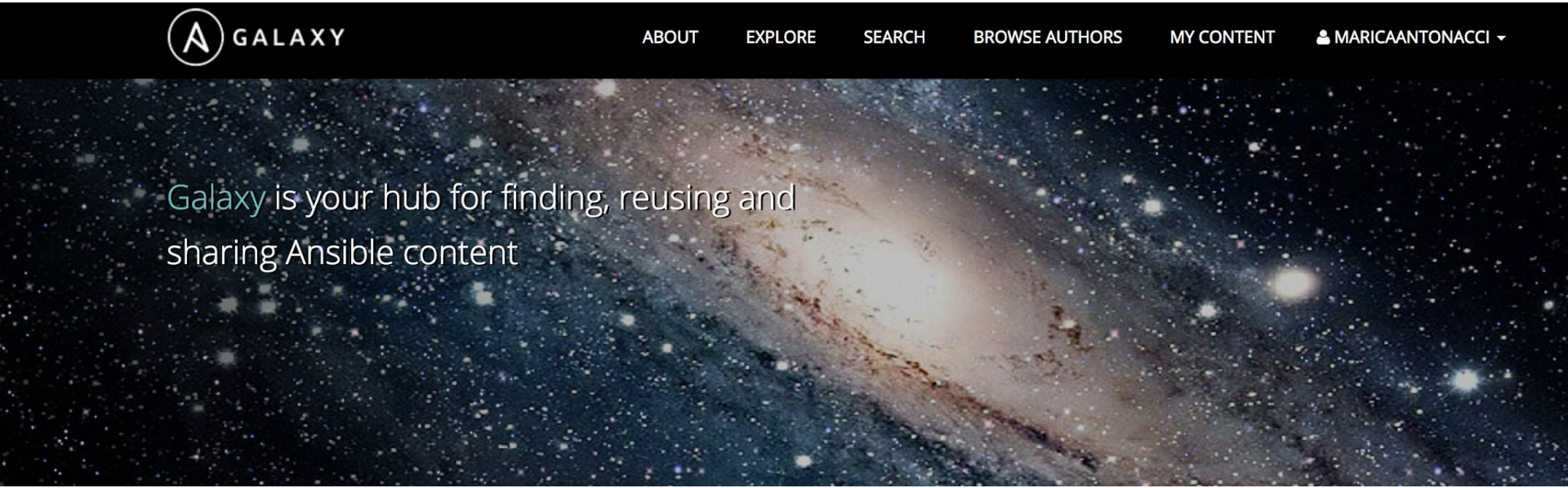
The templates directory is the location where the template module will automatically look for the **jinja2** templates included in the roles.

Create a playbook to run the role

```
[root@ansible ~]# cat site.yml
---
 - hosts: web
   roles:
   - apache
```

```
[root@ansible ~]# ansible-playbook -i hosts site.yml

PLAY [web] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [localhost]

TASK [apache : Install httpd Package] ******************************************
changed: [localhost]

TASK [apache : Copy httpd configuration file] **********************************
changed: [localhost]

TASK [apache : Copy index.html file] *******************************************
changed: [localhost]

TASK [apache : Start and Enable httpd service] *********************************
changed: [localhost]

RUNNING HANDLER [apache : restart apache] **************************************
changed: [localhost]

PLAY RECAP *********************************************************************
localhost                  : ok=6    changed=5    unreachable=0    failed=0
```

Ansible Galaxy



ansible-galaxy CLI tool

```
[root@ansible ~]# ansible-galaxy -h
Usage: ansible-galaxy [delete|import|info|init|install|list|login|remove|search|setup] [--help] [options] ...

Options:
  -h, --help            show this help message and exit
  -c, --ignore-certs    Ignore SSL certificate validation errors.
  -s API_SERVER, --server=API_SERVER
                        The API server destination
  -v, --verbose         verbose mode (-vvv for more, -vvvv to enable
                        connection debugging)
  --version             show program's version number and exit

 See 'ansible-galaxy <command> --help' for more information on a specific
command.
```

Search roles from Ansible Galaxy

```
[root@ansible ~]# ansible-galaxy search mesos

Found 71 roles matching your search:

 Name                                    Description
 ----                                    -----------
 andrewrothstein.mesosphere-mesos        mesosphere-mesos role
 andrewrothstein.mesosphere-mesosslave   mesosphere mesos slave role
 andrewrothstein.mesosphere-mesos-dns    mesosphere mesos dns role
 andrewrothstein.mesosphere-mesosmaster  mesosphere mesos master role
 d3v3d3.mesos                            Apache Mesos
 Cloud-PG.mesos                          Install Mesos components
 grycap.mesos                            Install Mesos components
 andrewrothstein.mesosphere-chronos      mesosphere chronos role
 indigo-dc.mesos                         Deploy Mesos master/slave
 andrewrothstein.mesosphere-bamboo       mesosphere bamboo role
 andrewrothstein.mesosphere-marathon     mesosphere marathon role
```

Filter by author or galaxy tags

```
[root@ansible ~]# ansible-galaxy search mesos --author indigo-dc

Found 3 roles matching your search:

 Name                 Description
 ----                 -----------
 indigo-dc.mesos      Deploy Mesos master/slave
 indigo-dc.spark-mesos Deploy Spark on Mesos through Marathon
 indigo-dc.dariahrepo  Deploy Dariah Repository on top of Mesos/Marathon
```

Get information about roles

```
ansible-galaxy info indigo-dc.mesos
```

```
Role: indigo-dc.mesos
        description: Deploy Mesos master/slave
        active: True
        commit: 07df63c8c9e41d4af30c96a80fe9f1fb8ff12a29
        commit_message: comment download of infn ca cert
        commit_url: https://github.com/indigo-dc/ansible-role-mesos/commit/07df63c8c9e41d4af30c96a80fe9f1fb8ff12a29
        company: INDIGO-DataCloud
        created: 2017-06-11T21:37:28.634Z
        dependencies: [{'role': 'indigo-dc.docker'}]
        download_count: 618
        forks_count: 3
        galaxy_info:
                author: marica.antonacci@gmail.com
                company: INDIGO-DataCloud
                galaxy_tags: ['mesos']
                license: Apache
                min_ansible_version: 2.0
                platforms: [{'name': 'Ubuntu', 'versions': ['trusty', 'xenial']}, {'name': 'EL', 'versions': [7]}]
        github_branch:
        github_repo: ansible-role-mesos
        github_user: indigo-dc
        id: 18387
        install_date: Thu May 31 14:35:23 2018
        intalled_version: indigo_2
        is_valid: True
        issue_tracker_url: https://github.com/indigo-dc/ansible-role-mesos/issues
        license: Apache
        min_ansible_version: 2.0
        modified: 2018-05-31T14:35:20.110Z
        namespace: indigo-dc
        open_issues_count: 0
        path: [u'/etc/ansible/roles', u'/usr/share/ansible/roles']
        readme: Mesos Role
```

Some of the data being displayed by the info command lives within the role itself, in the meta/main.yaml file

Download and install roles from Ansible Galaxy

```
ansible-galaxy install role_name(s)[,version]
```

```
[root@ansible ~]# ansible-galaxy install indigo-dc.mesos
- downloading role 'mesos', owned by indigo-dc
- downloading role from https://github.com/indigo-dc/ansible-role-mesos/archive/indigo_2.tar.gz
- extracting indigo-dc.mesos to /etc/ansible/roles/indigo-dc.mesos
- indigo-dc.mesos (indigo_2) was installed successfully
- adding dependency: indigo-dc.docker
- downloading role 'docker', owned by indigo-dc
- downloading role from https://github.com/indigo-dc/ansible-role-docker/archive/indigo_2.tar.gz
- extracting indigo-dc.docker to /etc/ansible/roles/indigo-dc.docker
- indigo-dc.docker (indigo_2) was installed successfully
```

The role and its dependencies are automatically installed.

Download and install roles from Git repo

```
ansible-galaxy install scm+role_repo_url[,version]
```

```
[root@ansible ~]# ansible-galaxy -v install git+https://github.com/indigo-dc/ansible-role-mesos.git,devel
Using /etc/ansible/ansible.cfg as config file
- extracting ansible-role-mesos to /etc/ansible/roles/ansible-role-mesos
- ansible-role-mesos (devel) was installed successfully
- adding dependency: indigo-dc.docker
- downloading role 'docker', owned by indigo-dc
- downloading role from https://github.com/indigo-dc/ansible-role-docker/archive/indigo_2.tar.gz
- extracting indigo-dc.docker to /etc/ansible/roles/indigo-dc.docker
- indigo-dc.docker (indigo_2) was installed successfully
```

Requires git installed on the ansible control machine

Creating roles with `ansible-galaxy`

`ansible-galaxy` tool can also be used to generate *scaffolding*, an initial set of files and directories involved in a role:

```
$ ansible-galaxy init apache
#This command will create the skeleton role in the current working dir
```

```
$ ansible-galaxy init --init-path=INIT_PATH apache
#This command will create the skeleton role in the INIT_PATH dir
```

```
[root@ansible ~]# ansible-galaxy init apache
- apache was created successfully
[root@ansible ~]# tree apache/
apache/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

Importing roles with `ansible-galaxy`

1st step: create the role github repo

2nd step: ansible login

3rd step: import the role in Ansible Galaxy

ansible-login

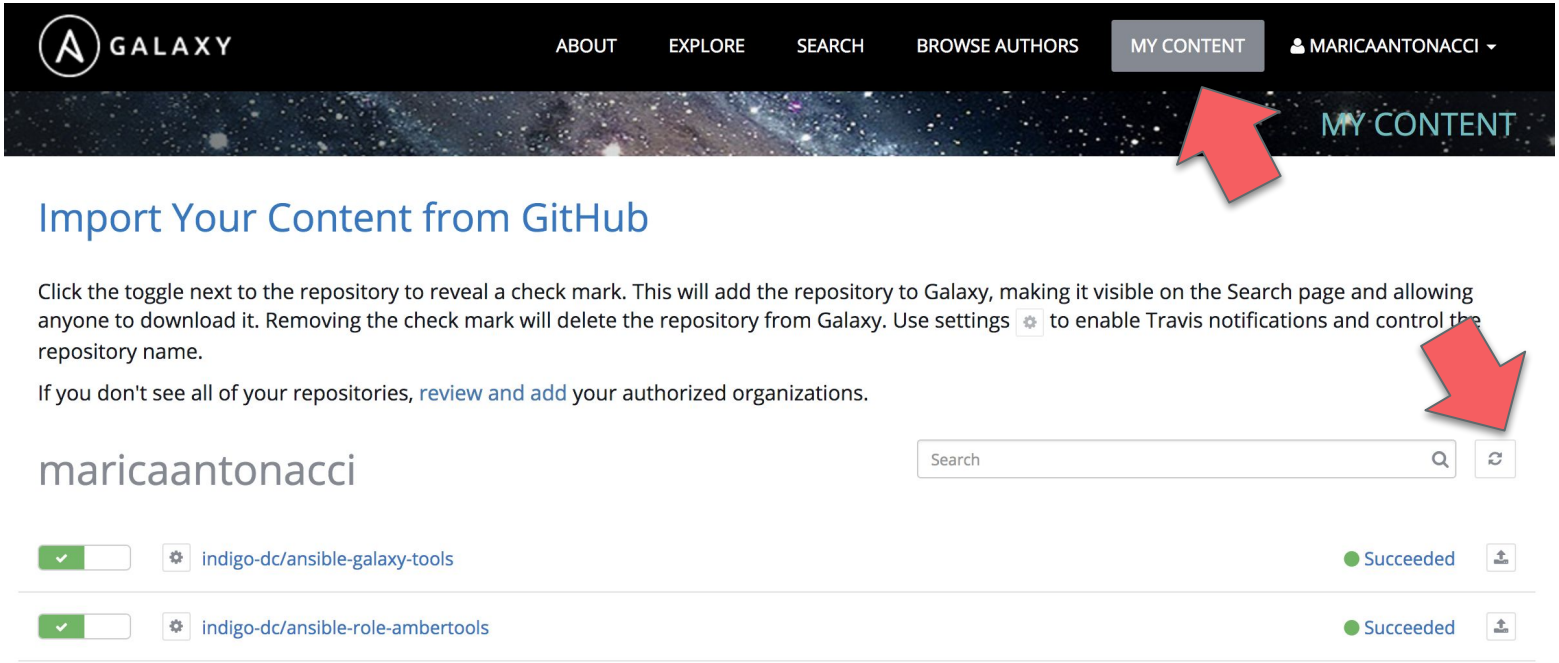
```
[root@ansible ~]# ansible-galaxy login -h
Usage: ansible-galaxy login [options]

Options:
  --github-token=TOKEN  Identify with github token rather than username and
                        password.
  -h, --help            show this help message and exit
  -c, --ignore-certs    Ignore SSL certificate validation errors.
  -s API_SERVER, --server=API_SERVER
                        The API server destination
  -v, --verbose         verbose mode (-vvv for more, -vvvv to enable
                        connection debugging)
  --version             show program's version number and exit
```

Github token can be generated here: https://github.com/settings/tokens

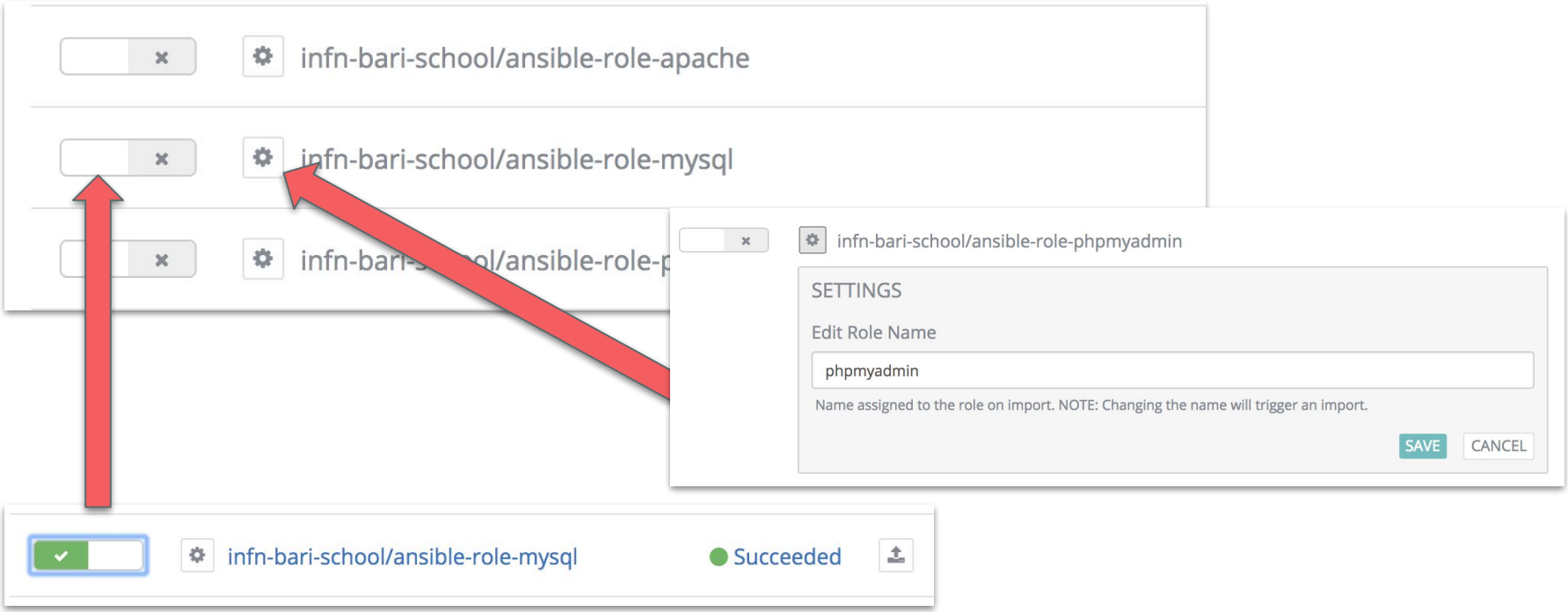
ansible-import

```
[root@ansible ~]# ansible-galaxy import -h
Usage: ansible-galaxy import [options] github_user github_repo

Options:
  --branch=REFERENCE    The name of a branch to import. Defaults to the
                        repository's default branch (usually master)
  -h, --help            show this help message and exit
  -c, --ignore-certs    Ignore SSL certificate validation errors.
  --no-wait             Don't wait for import results.
  --role-name=ROLE_NAME
                        The name the role should have, if different than the
                        repo name
  -s API_SERVER, --server=API_SERVER
                        The API server destination
  --status              Check the status of the most recent import request for
                        given github_user/github_repo.
  -v, --verbose         verbose mode (-vvv for more, -vvvv to enable
                        connection debugging)
  --version             show program's version number and exit
```

Import your role from Galaxy web portal



Import your role from Galaxy web portal



Check the import status

IMPORTS

+ ADD

ansible-role-mysql

infn-bari-school/ansible-role-mysql
Finished: 6/5/18 5:20 PM

infn-bari-school/ansible-role-mysql

master Update meta/main.yml

Finished about 22 seconds ago

Commit cd11e3f
Stars 0
Forks 0
Watchers 1

```
Starting import 247259: role_name=mysql repo=infn-bari-school/ansible-role-mysql
Accessing branch: master
Parsing and validating meta data.
Setting role name to mysql
Parsing galaxy_tags
Parsing platforms
Parsing cloud platforms
No cloud platforms found in meta data
Parsing and validating README
Adding repo tags as role versions
Removing old tags
Import completed
Status SUCCESS : warnings=1 errors=0
```

CI: Travis integration

```
$ ansible-galaxy setup travis github_user github_repo
xxx-travis-token-xxx
```

requires your Travis token from https://travis-ci.org/

Setting up your role for testing

```
[root@marica01 ansible-role-mysql]# tree tests/
tests/
├── inventory
└── test.yml
```

```
[root@marica01 ansible-role-mysql]# cat tests/inventory
localhost

[root@marica01 ansible-role-mysql]# cat tests/test.yml
---
- hosts: localhost
  remote_user: root
  roles:
    - ansible-role-mysql
```

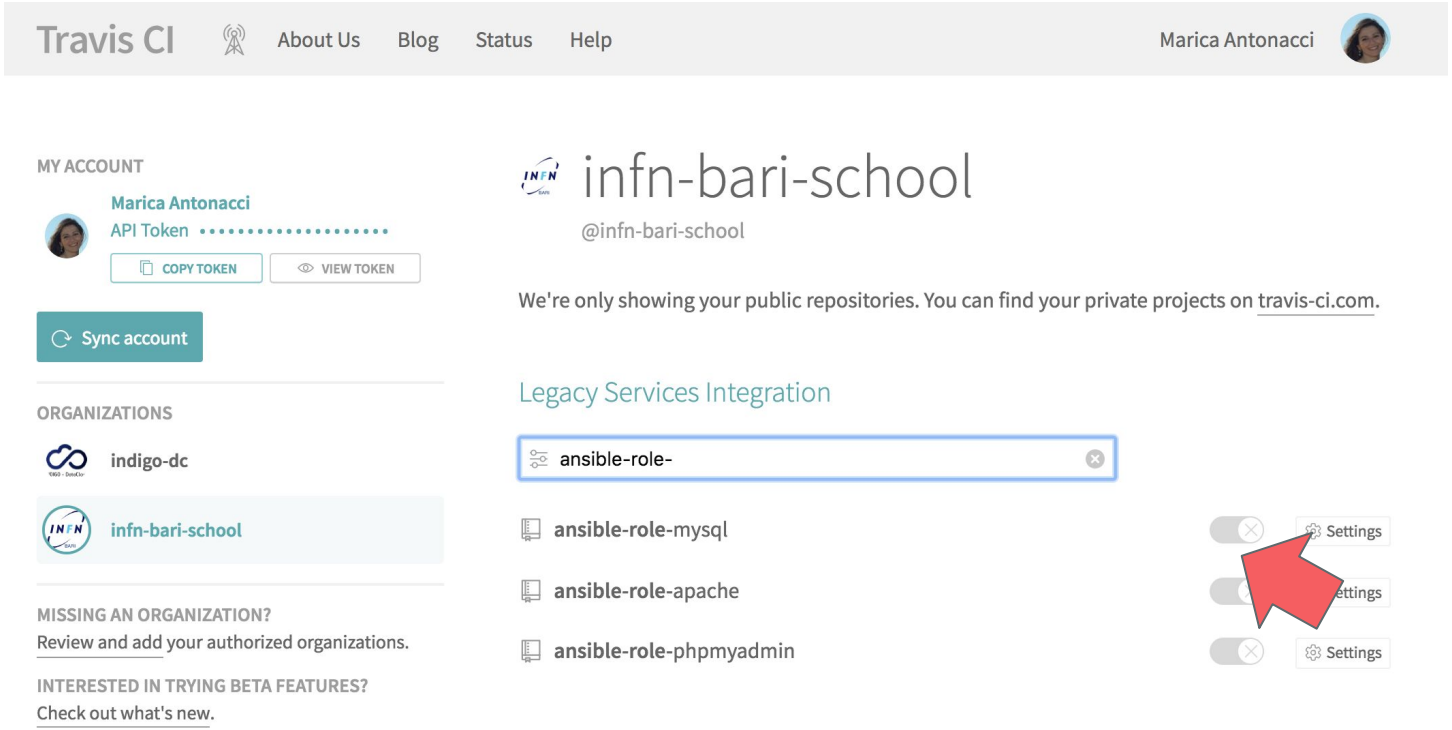
Setting up your role for testing (cont.)

Create the `.travis.yml`

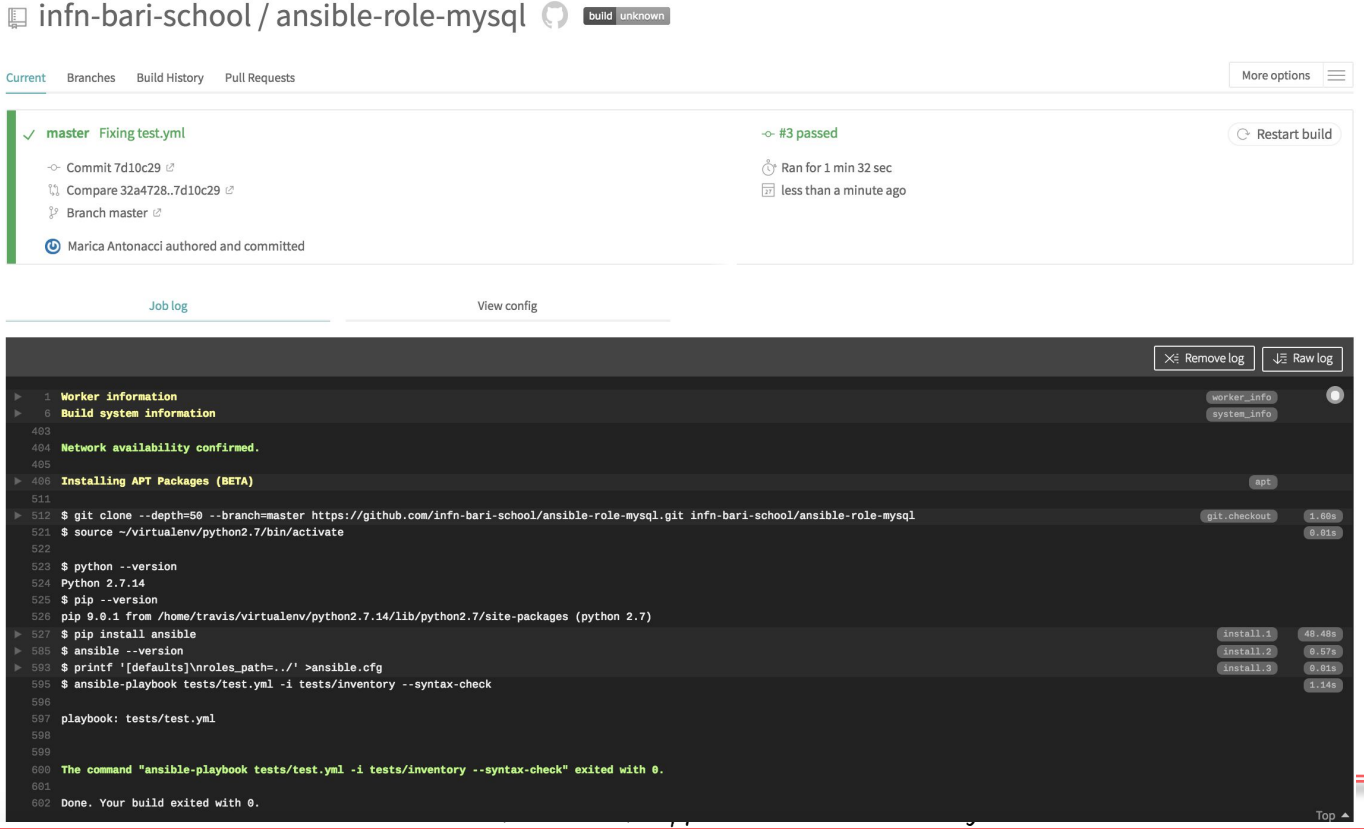
```
---
language: python
python: "2.7"

# Use the new container infrastructure
sudo: false

# Install ansible
addons:
  apt:
    packages:
    - python-pip

install:
  # Install ansible
  - pip install ansible

  # Check ansible version
  - ansible --version

  # Create ansible.cfg with correct roles_path
  - printf '[defaults]\nroles_path=../' >ansible.cfg

script:
  # Basic role syntax check
  - ansible-playbook tests/test.yml -i tests/inventory --syntax-check

notifications:
  webhooks: https://galaxy.ansible.com/api/v1/notifications/
```

Insert your test here.
The simplest test is the syntax check

Enable travis checks



A new build is automatically triggered



The build status is reported in Galaxy