

# Galaxy Developer Guide

You can host collections and roles on Galaxy to share with the Ansible community. Galaxy content is formatted in pre-packaged units of work such as [.ref: roles <playbooks\\_reuse\\_roles>](#), and new in Galaxy 3.2, [.ref: collections <collections>](#). You can create roles for provisioning infrastructure, deploying applications, and all of the tasks you do everyday. Taking this a step further, you can create collections which provide a comprehensive package of automation that may include multiple playbooks, roles, modules, and plugins.

**System Message: ERROR/3** (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\galaxy\ (ansible-devel) (docs) (docsite) (rst) (galaxy) dev\_guide.rst, line 7); [backlink](#)

Unknown interpreted text role "ref".

**System Message: ERROR/3** (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\galaxy\ (ansible-devel) (docs) (docsite) (rst) (galaxy) dev\_guide.rst, line 7); [backlink](#)

Unknown interpreted text role "ref".

- [Creating collections for Galaxy](#)
- [Creating roles for Galaxy](#)
  - [Force](#)
  - [Container enabled](#)
  - [Using a custom role skeleton](#)
  - [Authenticate with Galaxy](#)
  - [Import a role](#)
  - [Delete a role](#)
  - [Travis integrations](#)

## Creating collections for Galaxy

Collections are a distribution format for Ansible content. You can use collections to package and distribute playbooks, roles, modules, and plugins. You can publish and use collections through [Ansible Galaxy](#).

See [.ref: developing\\_collections](#) for details on how to create collections.

**System Message: ERROR/3** (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\galaxy\ (ansible-devel) (docs) (docsite) (rst) (galaxy) dev\_guide.rst, line 22); [backlink](#)

Unknown interpreted text role "ref".

## Creating roles for Galaxy

Use the `init` command to initialize the base structure of a new role, saving time on creating the various directories and `main.yml` files a role requires

```
$ ansible-galaxy init role_name
```

The above will create the following directory structure in the current working directory:

```
role_name/
  README.md
  .travis.yml
  defaults/
    main.yml
  files/
  handlers/
    main.yml
  meta/
    main.yml
  templates/
  tests/
    inventory
    test.yml
  vars/
    main.yml
```

If you want to create a repository for the role, the repository root should be *role\_name*.

## Force

If a directory matching the name of the role already exists in the current working directory, the `init` command will result in an error. To ignore the error use the `--force` option. Force will create the above subdirectories and files, replacing anything that matches.

## Container enabled

If you are creating a Container Enabled role, pass `--type container` to `ansible-galaxy init`. This will create the same directory structure as above, but populate it with default files appropriate for a Container Enabled role. For instance, the `README.md` has a slightly different structure, the `.travis.yml` file tests the role using [Ansible Container](#), and the meta directory includes a `container.yml` file.

## Using a custom role skeleton

A custom role skeleton directory can be supplied as follows:

```
$ ansible-galaxy init --role-skeleton=/path/to/skeleton role_name
```

When a skeleton is provided, `init` will:

- copy all files and directories from the skeleton to the new role
- any `.j2` files found outside of a `templates` folder will be rendered as templates. The only useful variable at the moment is `role_name`
- The `.git` folder and any `.git_keep` files will not be copied

Alternatively, the `role_skeleton` and ignoring of files can be configured via `ansible.cfg`

```
[galaxy]
role_skeleton = /path/to/skeleton
role_skeleton_ignore = ^.git$,^.*/.git_keep$
```

## Authenticate with Galaxy

Using the `import`, `delete` and `setup` commands to manage your roles on the Galaxy website requires authentication, and the `login` command can be used to do just that. Before you can use the `login` command, you must create an account on the Galaxy website.

The `login` command requires using your GitHub credentials. You can use your username and password, or you can create a [personal access token](#). If you choose to create a token, grant minimal access to the token, as it is used just to verify identity.

The following shows authenticating with the Galaxy website using a GitHub username and password:

```
$ ansible-galaxy login

We need your GitHub login to identify you.
This information will not be sent to Galaxy, only to api.github.com.
The password will not be displayed.

Use --github-token if you do not want to enter your password.

GitHub Username: dsmith
Password for dsmith:
Successfully logged into Galaxy as dsmith
```

When you choose to use your username and password, your password is not sent to Galaxy. It is used to authenticate with GitHub and create a personal access token. It then sends the token to Galaxy, which in turn verifies that your identity and returns a Galaxy access token. After authentication completes the GitHub token is destroyed.

If you do not want to use your GitHub password, or if you have two-factor authentication enabled with GitHub, use the `--github-token` option to pass a personal access token that you create.

## Import a role

The `import` command requires that you first authenticate using the `login` command. Once authenticated you can import any GitHub repository that you own or have been granted access.

Use the following to import to role:

```
$ ansible-galaxy import github_user github_repo
```

By default the command will wait for Galaxy to complete the import process, displaying the results as the import progresses:

```
Successfully submitted import request 41
Starting import 41: role_name=myrole repo=githubuser/ansible-role-repo ref=
```

```
Retrieving GitHub repo githubuser/ansible-role-repo
Accessing branch: devel
Parsing and validating meta/main.yml
Parsing galaxy_tags
Parsing platforms
Adding dependencies
Parsing and validating README.md
Adding repo tags as role versions
Import completed
Status SUCCESS : warnings=0 errors=0
```

## Branch

Use the `--branch` option to import a specific branch. If not specified, the default branch for the repo will be used.

## Role name

By default the name given to the role will be derived from the GitHub repository name. However, you can use the `--role-name` option to override this and set the name.

## No wait

If the `--no-wait` option is present, the command will not wait for results. Results of the most recent import for any of your roles is available on the Galaxy web site by visiting *My Imports*.

## Delete a role

The `delete` command requires that you first authenticate using the `login` command. Once authenticated you can remove a role from the Galaxy web site. You are only allowed to remove roles where you have access to the repository in GitHub.

Use the following to delete a role:

```
$ ansible-galaxy delete github_user github_repo
```

This only removes the role from Galaxy. It does not remove or alter the actual GitHub repository.

## Travis integrations

You can create an integration or connection between a role in Galaxy and [Travis](#). Once the connection is established, a build in Travis will automatically trigger an import in Galaxy, updating the search index with the latest information about the role.

You create the integration using the `setup` command, but before an integration can be created, you must first authenticate using the `login` command; you will also need an account in Travis, and your Travis token. Once you're ready, use the following command to create the integration:

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

The `setup` command requires your Travis token, however the token is not stored in Galaxy. It is used along with the GitHub username and repo to create a hash as described in [the Travis documentation](#). The hash is stored in Galaxy and used to verify notifications received from Travis.

The `setup` command enables Galaxy to respond to notifications. To configure Travis to run a build on your repository and send a notification, follow the [Travis getting started guide](#).

To instruct Travis to notify Galaxy when a build completes, add the following to your `.travis.yml` file:

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

## List Travis integrations

Use the `--list` option to display your Travis integrations:

```
$ ansible-galaxy setup --list
```

ID	Source	Repo
2	travis	github_user/github_repo
1	travis	github_user/github_repo

## Remove Travis integrations

Use the `--remove` option to disable and remove a Travis integration:

```
$ ansible-galaxy setup --remove ID
```

Provide the ID of the integration to be disabled. You can find the ID by using the `--list` option.

**System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\galaxy\ansible-devel) (docs) (docsite) (rst) (galaxy) dev\_guide.rst, line 238)**

Unknown directive type "seealso".

```
.. seealso::
   :ref:`collections`
      Shareable collections of modules, playbooks and roles
   :ref:`playbooks_reuse_roles`
      All about ansible roles
   `Mailing List <https://groups.google.com/group/ansible-project>`_
      Questions? Help? Ideas? Stop by the list on Google Groups
   :ref:`communication_irc`
      How to join Ansible chat channels
```