Testing collections

Testing your collection ensures that your code works well and integrates well with the rest of the Ansible ecosystem. Your collection should pass the general compile and sanity tests for Ansible code. You should also add unit tests to cover the code in your collection and integration tests to cover the interactions between your collection and ansible-core.

Testing tools

The main tool for testing collections is `ansible-test`, Ansible's testing tool described in :ref:`developing_testing` and provided by both the `ansible` and `ansible-core` packages.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 16);** ***backlink***
>
> Unknown interpreted text role "ref".

You can run several compile and sanity checks, as well as run unit and integration tests for plugins using `ansible-test`. When you test collections, test against the ansible-core version(s) you are targeting.

You must always execute `ansible-test` from the root directory of a collection. You can run `ansible-test` in Docker containers without installing any special requirements. The Ansible team uses this approach in Azure Pipelines both in the ansible/ansible GitHub repository and in the large community collections such as community.general and community.network. The examples below demonstrate running tests in Docker containers.

Compile and sanity tests

To run all compile and sanity tests:

```
ansible-test sanity --docker default -v
```

See :ref:`testing_compile` and :ref:`testing_sanity` for more information. See the :ref:`full list of sanity tests <all_sanity_tests>` for details on the sanity tests and how to fix identified issues.

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 31);** ***backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 31);** ***backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 31);** ***backlink***
>
> Unknown interpreted text role "ref".

Adding unit tests

You must place unit tests in the appropriate `tests/unit/plugins/` directory. For example, you would place tests for `plugins/module_utils/foo/bar.py` in `tests/unit/plugins/module_utils/foo/test_bar.py` or `tests/unit/plugins/module_utils/foo/bar/test_bar.py`. For examples, see the unit tests in community.general.

To run all unit tests for all supported Python versions:

```
ansible-test units --docker default -v
```

To run all unit tests only for a specific Python version:

```
ansible-test units --docker default -v --python 3.6
```

To run only a specific unit test:

```
ansible-test units --docker default -v --python 3.6 tests/unit/plugins/module_utils/foo/test_bar.py
```

You can specify Python requirements in the `tests/unit/requirements.txt` file. See :ref:`testing_units` for more information, especially on fixture files.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 56); *backlink*

Unknown interpreted text role "ref".

Adding integration tests

You must place integration tests in the appropriate `tests/integration/targets/` directory. For module integration tests, you can use the module name alone. For example, you would place integration tests for `plugins/modules/foo.py` in a directory called `tests/integration/targets/foo/`. For non-module plugin integration tests, you must add the plugin type to the directory name. For example, you would place integration tests for `plugins/connections/bar.py` in a directory called `tests/integration/targets/connection_bar/`. For lookup plugins, the directory must be called `lookup_foo`, for inventory plugins, `inventory_foo`, and so on.

You can write two different kinds of integration tests:

- Ansible role tests run with `ansible-playbook` and validate various aspects of the module. They can depend on other integration tests (usually named `prepare_bar` or `setup_bar`, which prepare a service or install a requirement named `bar` in order to test module `foo`) to set-up required resources, such as installing required libraries or setting up server services.
- `runme.sh` tests run directly as scripts. They can set up inventory files, and execute `ansible-playbook` or `ansible-inventory` with various settings.

For examples, see the integration tests in community.general. See also :ref:`testing_integration` for more details.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 70); *backlink*

Unknown interpreted text role "ref".

Since integration tests can install requirements, and set-up, start and stop services, we recommended running them in docker containers or otherwise restricted environments whenever possible. By default, `ansible-test` supports Docker images for several operating systems. See the list of supported docker images for all options. Use the `default` image mainly for platform-independent integration tests, such as those for cloud modules. The following examples use the `fedora35` image.

To execute all integration tests for a collection:

```
ansible-test integration --docker fedora35 -v
```

If you want more detailed output, run the command with `-vvv` instead of `-v`. Alternatively, specify `--retry-on-error` to automatically re-run failed tests with higher verbosity levels.

To execute only the integration tests in a specific directory:

```
ansible-test integration --docker fedora35 -v connection_bar
```

You can specify multiple target names. Each target name is the name of a directory in `tests/integration/targets/`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\[ansible-devel][docs][docsite][rst][dev_guide]developing_collections_testing.rst`, line 90)

Unknown directive type "seealso".

```
.. seealso::

   :ref:`developing_testing`
       More resources on testing Ansible
   :ref:`contributing_maintained_collections`
       Guidelines for contributing to selected collections
   `Mailing List <https://groups.google.com/group/ansible-devel>`_
```

```
    The development mailing list
:ref:`communication_irc`
    How to join Ansible chat channels
```