Tags

If you have a large playbook, it may be useful to run only specific parts of it instead of running the entire playbook. You can do this with Ansible tags. Using tags to execute or skip selected tasks is a two-step process:

1. Add tags to your tasks, either individually or with tag inheritance from a block, play, role, or import.
2. Select or skip tags when you run your playbook.

Adding tags with the tags keyword

You can add tags to a single task or include. You can also add tags to multiple tasks by defining them at the level of a block, play, role, or import. The keyword `tags` addresses all these use cases. The `tags` keyword always defines tags and adds them to tasks; it does not select or skip tasks for execution. You can only select or skip tasks based on tags at the command line when you run a playbook. See :ref:`using_tags` for more details.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 18); *backlink***
>
> Unknown interpreted text role "ref".

Adding tags to individual tasks

At the simplest level, you can apply one or more tags to an individual task. You can add tags to tasks in playbooks, in task files, or within a role. Here is an example that tags two tasks with different tags:

```
tasks:
- name: Install the servers
  ansible.builtin.yum:
    name:
    - httpd
    - memcached
    state: present
  tags:
  - packages
  - webservers

- name: Configure the service
  ansible.builtin.template:
    src: templates/src.j2
    dest: /etc/foo.conf
  tags:
  - configuration
```

You can apply the same tag to more than one individual task. This example tags several tasks with the same tag, "ntp":

```
---
# file: roles/common/tasks/main.yml

- name: Install ntp
  ansible.builtin.yum:
    name: ntp
    state: present
  tags: ntp
```

```
- name: Configure ntp
  ansible.builtin.template:
    src: ntp.conf.j2
    dest: /etc/ntp.conf
  notify:
  - restart ntpd
  tags: ntp

- name: Enable and run ntpd
  ansible.builtin.service:
    name: ntpd
    state: started
    enabled: yes
  tags: ntp

- name: Install NFS utils
  ansible.builtin.yum:
    name:
    - nfs-utils
    - nfs-util-lib
    state: present
  tags: filesharing
```

If you ran these four tasks in a playbook with `--tags ntp`, Ansible would run the three tasks tagged `ntp` and skip the one task that does not have that tag.

Adding tags to includes

You can apply tags to dynamic includes in a playbook. As with tags on an individual task, tags on an `include_*` task apply only to the include itself, not to any tasks within the included file or role. If you add `mytag` to a dynamic include, then run that playbook with `--tags mytag`, Ansible runs the include itself, runs any tasks within the included file or role tagged with `mytag`, and skips any tasks within the included file or role without that tag. See :ref:`selective_reuse` for more details.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 88); *backlink***
>
> Unknown interpreted text role "ref".

You add tags to includes the same way you add tags to any other task:

```
---
# file: roles/common/tasks/main.yml

- name: Dynamic re-use of database tasks
  include_tasks: db.yml
  tags: db
```

You can add a tag only to the dynamic include of a role. In this example, the `foo` tag will *not* apply to tasks inside the `bar` role:

```
---
- hosts: webservers
  tasks:
    - name: Include the bar role
      include_role:
        name: bar
      tags:
        - foo
```

With plays, blocks, the `role` keyword, and static imports, Ansible applies tag inheritance, adding the tags you define to every task inside the play, block, role, or imported file. However, tag inheritance does *not* apply to dynamic re-use with `include_role` and `include_tasks`. With dynamic re-use (includes), the tags you define apply only to the include itself. If you need tag inheritance, use a static import. If you cannot use an import because the rest of your playbook uses includes, see :ref:`apply_keyword` for ways to work around this behavior.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 114); *backlink***
>
> Unknown interpreted text role "ref".

Tag inheritance: adding tags to multiple tasks

If you want to apply the same tag or tags to multiple tasks without adding a `tags` line to every task, you can define the tags at the

level of your play or block, or when you add a role or import a file. Ansible applies the tags down the dependency chain to all child tasks. With roles and imports, Ansible appends the tags set by the `roles` section or import to any tags set on individual tasks or blocks within the role or imported file. This is called tag inheritance. Tag inheritance is convenient, because you do not have to tag every task. However, the tags still apply to the tasks individually.

Adding tags to blocks

If you want to apply a tag to many, but not all, of the tasks in your play, use a :ref:`block <playbooks_blocks>` and define the tags at that level. For example, we could edit the NTP example shown above to use a block:

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 126); *backlink***
>
> Unknown interpreted text role "ref".

```
# myrole/tasks/main.yml
- name: ntp tasks
  tags: ntp
  block:
  - name: Install ntp
    ansible.builtin.yum:
      name: ntp
      state: present

  - name: Configure ntp
    ansible.builtin.template:
      src: ntp.conf.j2
      dest: /etc/ntp.conf
    notify:
    - restart ntpd

  - name: Enable and run ntpd
    ansible.builtin.service:
      name: ntpd
      state: started
      enabled: yes

- name: Install NFS utils
  ansible.builtin.yum:
    name:
    - nfs-utils
    - nfs-util-lib
    state: present
  tags: filesharing
```

Adding tags to plays

If all the tasks in a play should get the same tag, you can add the tag at the level of the play. For example, if you had a play with only the NTP tasks, you could tag the entire play:

```
- hosts: all
  tags: ntp
  tasks:
  - name: Install ntp
    ansible.builtin.yum:
      name: ntp
      state: present

  - name: Configure ntp
    ansible.builtin.template:
      src: ntp.conf.j2
      dest: /etc/ntp.conf
    notify:
    - restart ntpd

  - name: Enable and run ntpd
    ansible.builtin.service:
      name: ntpd
      state: started
      enabled: yes

- hosts: fileservers
  tags: filesharing
  tasks:
  ...
```

Adding tags to roles

There are three ways to add tags to roles:

1. Add the same tag or tags to all tasks in the role by setting tags under `roles`. See examples in this section.
2. Add the same tag or tags to all tasks in the role by setting tags on a static `import_role` in your playbook. See examples in :ref:`tags_on_imports`.

   > **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 199); *backlink***
   >
   > Unknown interpreted text role "ref".

3. Add a tag or tags to individual tasks or blocks within the role itself. This is the only approach that allows you to select or skip some tasks within the role. To select or skip tasks within the role, you must have tags set on individual tasks or blocks, use the dynamic `include_role` in your playbook, and add the same tag or tags to the include. When you use this approach, and then run your playbook with `--tags foo`, Ansible runs the include itself plus any tasks in the role that also have the tag `foo`. See :ref:`tags_on_includes` for details.

   > **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 200); *backlink***
   >
   > Unknown interpreted text role "ref".

When you incorporate a role in your playbook statically with the `roles` keyword, Ansible adds any tags you define to all the tasks in the role. For example:

```
roles:
  - role: webserver
    vars:
      port: 5000
    tags: [ web, foo ]
```

or:

```
---
- hosts: webservers
  roles:
    - role: foo
      tags:
        - bar
        - baz
    # using YAML shorthand, this is equivalent to:
    # - { role: foo, tags: ["bar", "baz"] }
```

Adding tags to imports

You can also apply a tag or tags to all the tasks imported by the static `import_role` and `import_tasks` statements:

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

    - name: Import tasks from foo.yml
      import_tasks: foo.yml
      tags: [ web, foo ]
```

Tag inheritance for includes: blocks and the `apply` keyword

By default, Ansible does not apply :ref:`tag inheritance <tag_inheritance>` to dynamic re-use with `include_role` and `include_tasks`. If you add tags to an include, they apply only to the include itself, not to any tasks in the included file or role. This allows you to execute selected tasks within a role or task file - see :ref:`selective_reuse` when you run your playbook.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst]**

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 254); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 254); *backlink*

Unknown interpreted text role "ref".

If you want tag inheritance, you probably want to use imports. However, using both includes and imports in a single playbook can lead to difficult-to-diagnose bugs. For this reason, if your playbook uses `include_*` to re-use roles or tasks, and you need tag inheritance on one include, Ansible offers two workarounds. You can use the `apply` keyword:

```
- name: Apply the db tag to the include and to all tasks in db.yaml
  include_tasks:
    file: db.yml
    # adds 'db' tag to tasks within db.yml
    apply:
      tags: db
  # adds 'db' tag to this 'include_tasks' itself
  tags: db
```

Or you can use a block:

```
- block:
   - name: Include tasks from db.yml
     include_tasks: db.yml
  tags: db
```

Special tags: always and never

Ansible reserves two tag names for special behavior: always and never. If you assign the `always` tag to a task or play, Ansible will always run that task or play, unless you specifically skip it (`--skip-tags always`).

For example:

```
tasks:
- name: Print a message
  ansible.builtin.debug:
    msg: "Always runs"
  tags:
  - always

- name: Print a message
  ansible.builtin.debug:
    msg: "runs when you use tag1"
  tags:
  - tag1
```

Warning

- Fact gathering is tagged with 'always' by default. It is only skipped if you apply a tag and then use a different tag in `--tags` or the same tag in `--skip-tags`.

Warning

- The role argument specification validation task is tagged with 'always' by default. This validation will be skipped if you use `--skip-tags always`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 311)

Unknown directive type "versionadded".

```
.. versionadded:: 2.5
```

If you assign the `never` tag to a task or play, Ansible will skip that task or play unless you specifically request it (`--tags never`).

For example:

```
tasks:
  - name: Run the rarely-used debug task
    ansible.builtin.debug:
     msg: '{{ showmevar }}'
    tags: [ never, debug ]
```

The rarely-used debug task in the example above only runs when you specifically request the `debug` or `never` tags.

Selecting or skipping tags when you run a playbook

Once you have added tags to your tasks, includes, blocks, plays, roles, and imports, you can selectively execute or skip tasks based on their tags when you run :ref:`ansible-playbook`. Ansible runs or skips all tasks with tags that match the tags you pass at the command line. If you have added a tag at the block or play level, with `roles`, or with an import, that tag applies to every task within the block, play, role, or imported role or file. If you have a role with lots of tags and you want to call subsets of the role at different times, either :ref:`use it with dynamic includes <selective_reuse>`, or split the role into multiple roles.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 332); *backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 332); *backlink***
>
> Unknown interpreted text role "ref".

:ref:`ansible-playbook` offers five tag-related command-line options:

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 334); *backlink***
>
> Unknown interpreted text role "ref".

- `--tags all` - run all tasks, ignore tags (default behavior)
- `--tags [tag1, tag2]` - run only tasks with either the tag `tag1` or the tag `tag2`
- `--skip-tags [tag3, tag4]` - run all tasks except those with either the tag `tag3` or the tag `tag4`
- `--tags tagged` - run only tasks with at least one tag
- `--tags untagged` - run only tasks with no tags

For example, to run only tasks and blocks tagged `configuration` and `packages` in a very long playbook:

```
ansible-playbook example.yml --tags "configuration,packages"
```

To run all tasks except those tagged `packages`:

```
ansible-playbook example.yml --skip-tags "packages"
```

Previewing the results of using tags

When you run a role or playbook, you might not know or remember which tasks have which tags, or which tags exist at all. Ansible offers two command-line flags for :ref:`ansible-playbook` that help you manage tagged playbooks:

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 357); *backlink***
>
> Unknown interpreted text role "ref".

- `--list-tags` - generate a list of available tags
- `--list-tasks` - when used with `--tags tagname` or `--skip-tags tagname`, generate a preview of tagged tasks

For example, if you do not know whether the tag for configuration tasks is `config` or `conf` in a playbook, role, or tasks file, you can display all available tags without running any tasks:

```
ansible-playbook example.yml --list-tags
```

If you do not know which tasks have the tags `configuration` and `packages`, you can pass those tags and add `--list-tasks`.

Ansible lists the tasks but does not execute any of them.

```
ansible-playbook example.yml --tags "configuration,packages" --list-tasks
```

These command-line flags have one limitation: they cannot show tags or tasks within dynamically included files or roles. See :ref:`dynamic_vs_static` for more information on differences between static imports and dynamic includes.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 374); *backlink***
>
> Unknown interpreted text role "ref".

Selectively running tagged tasks in re-usable files

If you have a role or a tasks file with tags defined at the task or block level, you can selectively run or skip those tagged tasks in a playbook if you use a dynamic include instead of a static import. You must use the same tag on the included tasks and on the include statement itself. For example you might create a file with some tagged and some untagged tasks:

```
# mixed.yml
tasks:
- name: Run the task with no tags
  ansible.builtin.debug:
    msg: this task has no tags

- name: Run the tagged task
  ansible.builtin.debug:
    msg: this task is tagged with mytag
  tags: mytag

- block:
  - name: Run the first block task with mytag
    ...
  - name: Run the second block task with mytag
    ...
  tags:
  - mytag
```

And you might include the tasks file above in a playbook:

```
# myplaybook.yml
- hosts: all
  tasks:
  - name: Run tasks from mixed.yml
    include_tasks:
      name: mixed.yml
    tags: mytag
```

When you run the playbook with `ansible-playbook -i hosts myplaybook.yml --tags "mytag"`, Ansible skips the task with no tags, runs the tagged individual task, and runs the two tasks in the block.

Configuring tags globally

If you run or skip certain tags by default, you can use the :ref:`TAGS_RUN` and :ref:`TAGS_SKIP` options in Ansible configuration to set those defaults.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 421); *backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 421); *backlink***
>
> Unknown interpreted text role "ref".

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\[ansible-devel][docs][docsite][rst][user_guide]playbooks_tags.rst, line 423)**
>
> Unknown directive type "seealso".

```
.. seealso::

   :ref:`playbooks_intro`
       An introduction to playbooks
   :ref:`playbooks_reuse_roles`
       Playbook organization by roles
   `User Mailing List <https://groups.google.com/group/ansible-devel>`_
       Have a question?  Stop by the google group!
   :ref:`communication_irc`
       How to join Ansible chat channels
```