We're updating the Ansible community mission statement! Participate in our survey and let us know - What does Ansible mean to you? (https://www.surveymonkey.co.uk/r/DLG9FJN)

You are reading the **latest** (stable) community version of the Ansible documentation. If you are a Red Hat customer, refer to the Ansible Automation Platform Life Cycle (https://access.redhat.com/support/policy/updates/ansible-automation-platform) page for subscription details.

ansible.builtin.file module – Manage files and file properties

Note

This module is part of `ansible-core` and included in all Ansible installations. In most cases, you can use the short module name `file` even without specifying the `collections:` keyword. However, we recommend you use the FQCN for easy linking to the module documentation and to avoid conflicting with other collections that may have the same module name.

Synopsis

- Set attributes of files, directories, or symlinks and their targets.
- Alternatively, remove files, symlinks or directories.

- Many other modules support the same options as the `file` module - including ansible.builtin.copy (copy_module.html#ansible-collections-ansible-builtin-copy-module), ansible.builtin.template (template_module.html#ansible-collections-ansible-builtin-template-module), and ansible.builtin.assemble (assemble_module.html#ansible-collections-ansible-builtin-assemble-module).
- For Windows targets, use the ansible.windows.win_file (../windows/win_file_module.html#ansible-collections-ansible-windows-win-file-module) module instead.

Parameters

access_time
string
added in Ansible 2.7

This parameter indicates the time the file's access time should be set to.

Should be `preserve` when no modification is required, `YYYYMMDDHHMM.SS` when using default time format, or `now`.

Default is `None` meaning that `preserve` is the default for `state=[file,directory,link,hard]` and `now` is default for `state=touch`.

access_time_format
string
added in Ansible 2.7

When used with `access_time`, indicates the time format that must be used.

Based on default Python format (see time.strftime doc).

Default: `"%Y%m%d%H%M.%S"`

attributes
aliases: attr
string

The attributes the resulting filesystem object should have.

To get supported flags look at the man page for *chattr* on the target system.

This string should contain the attributes in the same order as the one displayed by *lsattr*.

The `=` operator is assumed as default, otherwise `+` or `-` operators need to be included in the string.

follow
boolean

This flag indicates that filesystem links, if they exist, should be followed.

follow=yes and *state=link* can modify *src* when combined with parameters such as *mode*.

Previous to Ansible 2.5, this was `false` by default.

Choices:

- `false`
- **`true`** ← (default)

Search this site

force
boolean

Force the creation of the symlinks in two cases: the source file does not exist (but will appear later); the destination exists and is a file (so, we need to unlink the `path` file and create symlink to the `src` file in place of it).

Choices:

- `false` ← (default)
- `true`

group
string

Name of the group that should own the filesystem object, as would be fed to *chown*.

When left unspecified, it uses the current group of the current user unless you are root, in which case it can preserve the previous ownership.

mode
any

The permissions the resulting filesystem object should have.

For those used to */usr/bin/chmod* remember that modes are actually octal numbers. You must give Ansible enough information to parse them correctly. For consistent results, quote octal numbers (for example, `'644'` or `'1777'`) so Ansible receives a string and can do its own conversion from string into number. Adding a leading zero (for example, `0755`) works sometimes, but can fail in loops and some other circumstances.

Giving Ansible a number without following either of these rules will end up with a decimal number which will have unexpected results.

As of Ansible 1.8, the mode may be specified as a symbolic mode (for example, `u+rwx` or `u=rw,g=r,o=r`).

If `mode` is not specified and the destination filesystem object **does not** exist, the default `umask` on the system will be used when setting the mode for the newly created filesystem object.

If `mode` is not specified and the destination filesystem object **does** exist, the mode of the existing filesystem object will be used.

Specifying `mode` is the best way to ensure filesystem objects are created with the correct permissions. See CVE-2020-1736 for further details.

modification_time
string
added in Ansible 2.7

This parameter indicates the time the file's modification time should be set to.

Should be `preserve` when no modification is required, `YYYYMMDDHHMM.SS` when using default time format, or `now`.

Default is None meaning that `preserve` is the default for `state=[file,directory,link,hard]` and `now` is default for `state=touch`.

modification_time_format
string
added in Ansible 2.7

When used with `modification_time`, indicates the time format that must be used.

Based on default Python format (see time.strftime doc).

Default: `"%Y%m%d%H%M.%S"`

owner
string

Name of the user that should own the filesystem object, as would be fed to *chown*.

When left unspecified, it uses the current user unless you are root, in which case it can preserve the previous ownership.

Specifying a numeric username will be assumed to be a user ID and not a username. Avoid numeric usernames to avoid this confusion.

path
aliases: dest, name
path / required

Path to the file being managed.

recurse
boolean

Recursively set the specified file attributes on directory contents.

This applies only when `state` is set to `directory`.

Choices:

- **`false`** ← (default)
- `true`

selevel
string

The level part of the SELinux filesystem object context.

This is the MLS/MCS attribute, sometimes known as the `range`.

When set to `_default`, it will use the `level` portion of the policy if available.

serole
string

The role part of the SELinux filesystem object context.

When set to `_default`, it will use the `role` portion of the policy if available.

setype
string

The type part of the SELinux filesystem object context.

When set to `_default`, it will use the `type` portion of the policy if available.

seuser
string

The user part of the SELinux filesystem object context.

By default it uses the `system` policy, where applicable.

When set to `_default`, it will use the `user` portion of the policy if available.

src
path

Path of the file to link to.

This applies only to `state=link` and `state=hard`.

For `state=link`, this will also accept a non-existing path.

Relative paths are relative to the file being created (`path`) which is how the Unix command `ln -s SRC DEST` treats relative paths.

state
string

If `absent`, directories will be recursively deleted, and files or symlinks will be unlinked. In the case of a directory, if `diff` is declared, you will see the files and folders deleted listed under `path_contents`. Note that `absent` will not cause `file` to fail if the `path` does not exist as the state did not change.

If `directory`, all intermediate subdirectories will be created if they do not exist. Since Ansible 1.7 they will be created with the supplied permissions.

If `file`, with no other options, returns the current state of `path`.

If `file`, even with other options (such as `mode`), the file will be modified if it exists but will NOT be created if it does not exist. Set to `touch` or use the ansible.builtin.copy (copy_module.html#ansible-collections-ansible-builtin-copy-module) or ansible.builtin.template (template_module.html#ansible-collections-ansible-builtin-template-module) module if you want to create the file if it does not exist.

If `hard`, the hard link will be created or changed.

If `link`, the symbolic link will be created or changed.

If `touch` (new in 1.4), an empty file will be created if the file does not exist, while an existing file or directory will receive updated file access and modification times (similar to the way `touch` works from the command line).

Default is the current state of the file if it exists, `directory` if `recurse=yes`, or `file` otherwise.

Choices:

- `"absent"`
- `"directory"`
- `"file"`
- `"hard"`
- `"link"`
- `"touch"`

unsafe_writes
boolean

Influence when to use atomic operation to prevent data corruption or inconsistent reads from the target filesystem object.

By default this module uses atomic operations to prevent data corruption or inconsistent reads from the target filesystem objects, but sometimes systems are configured or just broken in ways that prevent this. One example is docker mounted filesystem objects, which cannot be updated atomically from inside the container and can only be written in an unsafe manner.

This option allows Ansible to fall back to unsafe methods of updating filesystem objects when atomic operations fail (however, it doesn't force Ansible to perform unsafe writes).

IMPORTANT! Unsafe writes are subject to race conditions and can lead to data corruption.

Choices:

- `false` ← (default)
- `true`

Attributes

check_mode

Support: full

Can run in check_mode and return changed status prediction without modifying target

diff_mode

Support: partial

permissions and ownership will be shown but file contents on absent/touch will not.

Will return details on what has changed (or possibly needs changing in check_mode), when in diff mode

platform

Platform: posix

Target OS/families that can be operated against

See Also

See also

ansible.builtin.assemble (assemble_module.html#ansible-collections-ansible-builtin-assemble-module)
Assemble configuration files from fragments.

ansible.builtin.copy (copy_module.html#ansible-collections-ansible-builtin-copy-module)
Copy files to remote locations.

ansible.builtin.stat (stat_module.html#ansible-collections-ansible-builtin-stat-module)

Search this site

Retrieve file or file system status.

ansible.builtin.template (template_module.html#ansible-collections-ansible-builtin-template-module)

Template a file out to a target host.

ansible.windows.win_file (../windows/win_file_module.html#ansible-collections-ansible-windows-win-file-module)

Creates, touches or removes files or directories.

Search this site

Examples

```
- name: Change file ownership, group and permissions
  ansible.builtin.file:
    path: /etc/foo.conf
    owner: foo
    group: foo
    mode: '0644'

- name: Give insecure permissions to an existing file
  ansible.builtin.file:
    path: /work
    owner: root
    group: root
    mode: '1777'

- name: Create a symbolic link
  ansible.builtin.file:
    src: /file/to/link/to
    dest: /path/to/symlink
    owner: foo
    group: foo
    state: link

- name: Create two hard links
  ansible.builtin.file:
    src: '/tmp/{{ item.src }}'
    dest: '{{ item.dest }}'
    state: hard
  loop:
    - { src: x, dest: y }
    - { src: z, dest: k }

- name: Touch a file, using symbolic modes to set the permissions (equivalent to 0644)
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u=rw,g=r,o=r

- name: Touch the same file, but add/remove some permissions
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u+rw,g-wx,o-rwx

- name: Touch again the same file, but do not change times this makes the task
idempotent
  ansible.builtin.file:
    path: /etc/foo.conf
    state: touch
    mode: u+rw,g-wx,o-rwx
    modification_time: preserve
    access_time: preserve

- name: Create a directory if it does not exist
  ansible.builtin.file:
    path: /etc/some_directory
    state: directory
    mode: '0755'

- name: Update modification and access time of given file
  ansible.builtin.file:
```

Search this site

```
    path: /etc/some_file
    state: file
    modification_time: now
    access_time: now

- name: Set access time based on seconds from epoch value
  ansible.builtin.file:
    path: /etc/another_file
    state: file
    access_time: '{{ "%Y%m%d%H%M.%S" | strftime(stat_var.stat.atime) }}'

- name: Recursively change ownership of a directory
  ansible.builtin.file:
    path: /etc/foo
    state: directory
    recurse: yes
    owner: foo
    group: foo

- name: Remove file (delete file)
  ansible.builtin.file:
    path: /etc/foo.txt
    state: absent

- name: Recursively remove directory
  ansible.builtin.file:
    path: /etc/foo
    state: absent
```

Return Values

Common return values are documented here (../../../reference_appendices/common_return_values.html#common-return-values), the following are the fields unique to this module:

dest
string

Destination file/path, equal to the value passed to *path*.

Returned: state=touch, state=hard, state=link

Sample: `"/path/to/file.txt"`

path
string

Destination file/path, equal to the value passed to *path*.

Returned: state=absent, state=directory, state=file

Sample: `"/path/to/file.txt"`

Authors

- Ansible Core Team

Search this site

- Michael DeHaan

Collection links

Issue Tracker (https://github.com/ansible/ansible/issues)

Repository (Sources) (https://github.com/ansible/ansible)

Communication (./#communication-for-ansible-builtin)

Search this site