Ansible documentation style guide

Welcome to the Ansible style guide! To create clear, concise, consistent, useful materials on docs.ansible.com, follow these guidelines:

Linguistic guidelines

We want the Ansible documentation to be:

- clear
- direct
- conversational
- easy to translate

We want reading the docs to feel like having an experienced, friendly colleague explain how Ansible works.

Stylistic cheat-sheet

This cheat-sheet illustrates a few rules that help achieve the "Ansible tone":

Rule	Good example	Bad example
Use active voice	You can run a task by	A task can be run by
Use the present tense	This command creates a	This command will create a
Address the reader	As you expand your inventory	When the number of managed nodes grows
Use standard English	Return to this page	Hop back to this page
Use American English	The color of the output	The colour of the output

Header case

Headers should be written in sentence case. For example, this section's title is `Header case`, not `Header Case` or `HEADER CASE`.

Avoid using Latin phrases

Latin words and phrases like `e.g.` or `etc.` are easily understood by English speakers. They may be harder to understand for others and are also tricky for automated translation.

Use the following English terms in place of Latin terms or abbreviations:

Latin	English
i.e	in other words
e.g.	for example
etc	and so on
via	by/ through
vs./versus	rather than/against

reStructuredText guidelines

The Ansible documentation is written in reStructuredText and processed by Sphinx. We follow these technical or mechanical guidelines on all rST pages:

Header notation

Section headers in reStructuredText can use a variety of notations. Sphinx will 'learn on the fly' when creating a hierarchy of headers. To make our documents easy to read and to edit, we follow a standard set of header notations. We use:

- ### with overline, for parts:

```
##############
Developer guide
##############
```

- *** with overline, for chapters:

```
*******************
Ansible style guide
*******************
```

- === for sections:

```
Mechanical guidelines
=====================
```

- --- for subsections:

```
Internal navigation
-------------------
```

- ^^^ for sub-subsections:

```
Adding anchors
^^^^^^^^^^^^^^
```

- """ for paragraphs:

```
Paragraph that needs a title
""""""""""""""""""""""""""""""
```

Syntax highlighting - Pygments

The Ansible documentation supports a range of Pygments lexers for syntax highlighting to make our code examples look good. Each code-block must be correctly indented and surrounded by blank lines.

The Ansible documentation allows the following values:

- none (no highlighting)
- ansible-output (a custom lexer for Ansible output)
- bash
- console
- csharp
- ini
- json
- powershell
- python
- rst
- sh
- shell
- shell-session
- text
- yaml
- yaml+jinja

For example, you can highlight Python code using following syntax:

```
.. code-block:: python

   def my_beautiful_python_code():
       pass
```

Internal navigation

Anchors (also called labels) and links work together to help users find related content. Local tables of contents also help users navigate quickly to the information they need. All internal links should use the :ref: syntax. Every page should have at least one anchor to support internal :ref: links. Long pages, or pages with multiple levels of headers, can also include a local TOC.

> **Note**
>
> Avoid raw URLs. RST and sphinx allow :https://my.example.com, but this is unhelpful for those using screen readers. :ref: links automatically pick up the header from the anchor, but for external links, always use the :link title <link-url>`_ format.

Adding anchors

- Include at least one anchor on every page

- Place the main anchor above the main header
- If the file has a unique title, use that for the main page anchor:

```
.. _unique_page::
```

- You may also add anchors elsewhere on the page

Adding internal links

- All internal links must use `:ref:` syntax. These links both point to the anchor defined above:

```
:ref:`unique page`
:ref:`this page <unique_page>`
```

The second example adds custom text for the link.

Adding links to modules and plugins

Ansible 2.10 and later require the extended Fully Qualified Collection Name (FQCN) as part of the links:

```
ansible_collections. + FQCN + _module
```

For example:

```
:ref:`ansible.builtin.first_found lookup plugin <ansible_collections.ansible.builtin.first_found_looku
```

displays as :ref:`ansible.builtin.first_found lookup plugin <ansible_collections.ansible.builtin.first_found_lookup>`.

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\style_guide\[ansible-devel][docs][docsite][rst][dev_guide][style_guide]index.rst`, **line 228);** *backlink*
>
> Unknown interpreted text role "ref".

Modules require different suffixes from other plugins:

- Module links use this extended FQCN module name with `_module` for the anchor.
- Plugin links use this extended FQCN plugin name with the plugin type (`_connection` for example).

```
:ref:`arista.eos.eos config <ansible collections.arista.eos.eos config module>`
:ref:`kubernetes.core.kubectl connection plugin <ansible_collections.kubernetes.core.kubectl_connection>`
```

> **Note**
>
> `ansible.builtin` is the FQCN for modules included in `ansible.base`. Documentation links are the only place you prepend `ansible_collections` to the FQCN. This is used by the documentation build scripts to correctly fetch documentation from collections on Ansible Galaxy.

Adding local TOCs

The page you're reading includes a local TOC. If you include a local TOC:

- place it below, not above, the main heading and (optionally) introductory text
- use the `:local:` directive so the page's main header is not included
- do not include a title

The syntax is:

```
.. contents::
   :local:
```

Accessibility guidelines

Ansible documentation has a goal to be more accessible. Use the following guidelines to help us reach this goal.

Images and alternative text

> Ensure all icons, images, diagrams, and non text elements have a meaningful alternative-text description. Do not include screen captures of CLI output. Use `code-block` instead.
>
> ```
> .. image:: path/networkdiag.png
>    :width: 400
>    :alt: SpiffyCorp network diagram
> ```

Links and hypertext

> URLs and cross-reference links have descriptive text that conveys information about the content of the linked target. See :ref:`style_links` for how to format links.

Tables

Tables have a simple, logical reading order from left to right, and top to bottom. Tables include a header row and avoid empty or blank table cells. Label tables with a descriptive title.

```
.. table:: File descriptions

   +----------+---------------------------+
   |File      |Purpose                    |
   +==========+===========================+
   |foo.txt   |foo configuration settings |
   +----------+---------------------------+
   |bar.txt   |bar configuration settings |
   +----------+---------------------------+
```

Colors and other visual information

- Avoid instructions that rely solely on sensory characteristics. For example, do not use `Click the square, blue button to continue.`
- Convey information by methods and not by color alone.
- Ensure there is sufficient contrast between foreground and background text or graphical elements in images and diagrams.
- Instructions for navigating through an interface make sense without directional indicators such as left, right, above, and below.

Accessibility resources

Use the following resources to help test your documentation changes:

- axe DevTools browser extension - Highlights accessibility issues on a website page.
- WAVE browser extension from WebAIM - another accessibility tester.
- Orca screen reader - Common tool used by people with vision impairments.
- color filter - For color-blind testing.

More resources

These pages offer more help with grammatical, stylistic, and technical rules for documentation.