NetApp

Contribute to NetApp technical content in GitHub

Contributor's Guide

NetApp
February 17, 2022

Table of Contents

Contribute to NetApp technical content in GitHub

The documentation for NetApp's products and services is open source. This enables you to contribute to the content by making improvements, corrections, and suggestions. All you need is a GitHub account and a little initiative.

Overview

You can contribute to our docs by using the following options:

- Click **Request doc changes** to submit general feedback or to ask a question about the content. The NetApp content lead then reviews your request to determine what changes are needed to the docs. This is the most common option.

  View step-by-step instructions for this option.

- Click **Edit this page** to directly edit the content yourself. The NetApp content lead then reviews your edits and merges them.

  View step-by-step instructions for this option.

Request doc changes

Submitting a request for a doc change is the most common way to contribute to NetApp docs. After you submit the request, the content lead will confirm that they've received your feedback. You'll get an email notification from GitHub when they do.

If the content lead agrees that your suggestion can make the content better, they will commit the change shortly after that. You'll receive another notification that your feedback was incorporated.

> 💡 All comments that you provide are publicly visible. Anyone who navigates to the issues in the GitHub repo can see your comments.

Steps

1. If you don't already have a GitHub account, create one from github.com.
2. Log in to your GitHub account.
3. Using your web browser, open the page on docs.netapp.com that's related to your feedback.
4. On the right side of the page, click **Request doc changes**.
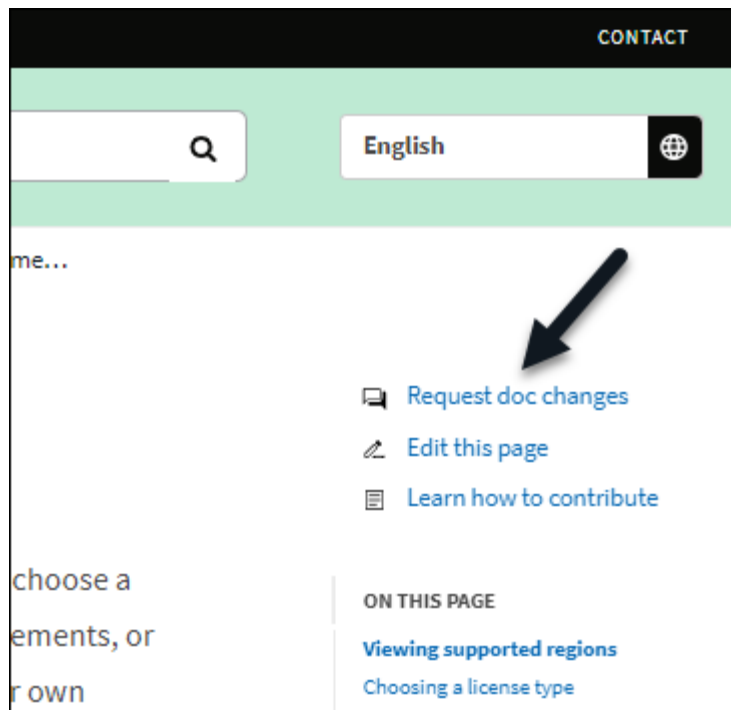
A new browser tab opens with a form that you can use to provide details to our doc team.

5. Enter a title and then provide details about your request.

   The comment field is prepopulated with the title and URL for the page. Don't delete this information because we'll need it to understand your request.

6. Click **Submit new issue** to create an issue for your request.

After you're done

Opening an issue enables collaboration through GitHub comments. You'll receive email notifications based on the preferences that you indicated in your GitHub account settings.

You can also view the status of the request by clicking **Issues** on the top banner next to the GitHub search box:
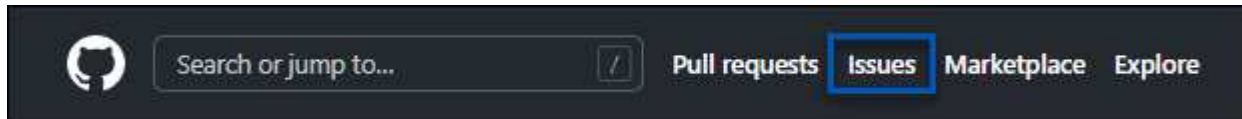
Submit edits to the docs

If you're comfortable editing the content yourself, you can submit the exact doc changes that you'd like to see by directly editing the source file.

As an outside contributor, you won't be able to publish the change directly. The content lead will review the changes, make any required edits, and then merge the changes. You'll get an email notification from GitHub when this happens.

If you need help with our writing style or source syntax, you can use these resources:

- Style guide for NetApp docs
- AsciiDoc syntax

Steps

1. If you don't already have a GitHub account, create one from github.com.
2. Log in to your GitHub account.
3. Using your web browser, open the page on docs.netapp.com that you want to edit.
4. On the right side of the page, click **Edit this page**.
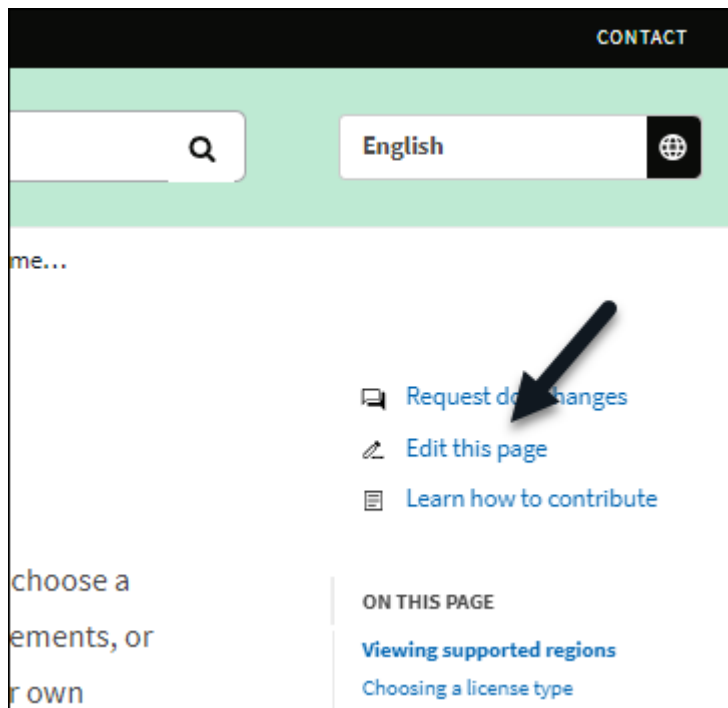
5. Click the pencil icon.
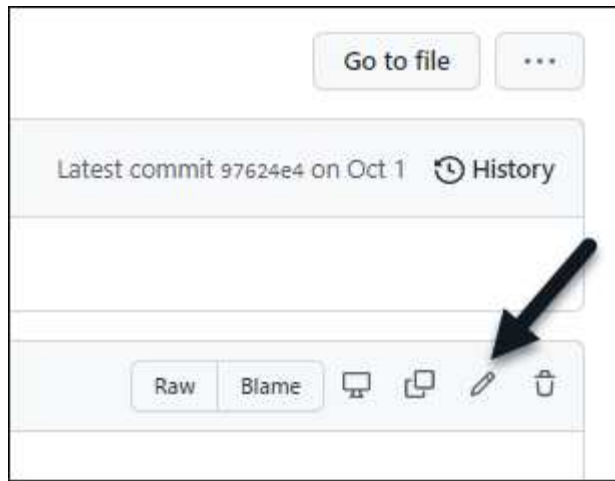
6. Edit the content.

   The content is written in AsciiDoc, a lightweight markup language. If you need help, click here to learn about AsciiDoc syntax.

7. To commit your changes, scroll down the page and fill out the form:

   a. Enter a title and optional description.

   b. Select **Create a new branch for this commit and start a pull request**.

   c. Click **Propose changes**.

      GitHub automatically fills in a branch name (for example, *username-patch-n*) for the change.

8. Provide a comment about the edit that you made, then click **Create pull request**.

After you're done

After you propose the changes, we'll review them, make any necessary edits, and then merge the changes into the GitHub repository.

You can view the status of the pull request by clicking **Pull requests** on the top banner next to the GitHub search box:

Style guide for NetApp docs

Our style is conversational and empathetic, but we stay professional and get to the point. Follow these guidelines when writing content for NetApp docs.

Write conversationally

Write like you speak when you're explaining something to a professional colleague. Writing in a conversational tone helps you connect with your audience.

> 💡 Read your writing out loud. As Elmore Leonard, an American novelist and screenwriter, said, "If it sounds like writing, I rewrite it."

Too formal	Too casual	Our style
If you experience issues with Cloud Tiering, you can view the health status on the Cluster Dashboard to determine why the error occurred. The health reflects the status of the ONTAP system and the Service Connector.	It's no fun when failures happen. Don't worry though, just check out the Cluster Dashboard to see what happened and what's affected.	When there's a failure, Cloud Tiering displays a "Failed" health status on the Cluster Dashboard. View the status to get information about the failure.
You can pay for Cloud tiering in multiple ways. You can pay for Cloud Tiering through a pay-as-you-go subscription, you can pay through an ONTAP tiering license, or you can pay through a combination of both. You cannot get licensing from within Cloud Manager, you must go directly to the Cloud Tiering service to set it up.	We have like three ways you can pay for Cloud tiering. A pay-as-you-go subscription of course, but also an ONTAP tiering license, or even a combination of both. But you can't get licensing from Cloud Manager, so just go directly to the Cloud Tiering service to set it up.	Pay for Cloud Tiering through a pay-as-you-go subscription, an ONTAP tiering license, or a combination of both. Licensing isn't available from within Cloud Manager, but you can go directly to the Cloud Tiering service to set it up.
Before you can provision storage, you must discover your ONTAP cluster from Cloud Manager. After discovery is complete, you must open the working environment to provision storage.	All you do is first discover your ONTAP cluster from Cloud Manager, then open the working environment to start provisioning storage. Easy!	After you discover your ONTAP cluster from Cloud Manager, open the working environment to provision storage.
Analyze the wellness attributes of your storage system in one of two ways, depending on what you want to analyze: Either select the wellness attribute widget on the dashboard or select View All to view the entire list of wellness attributes.	You've got two ways to analyze your system's wellness attributes: Just select the wellness attribute widget on the dashboard. Or, if you'd rather see the whole list of wellness attributes in one view, select View All.	Select the wellness attribute widget on the dashboard or select View All to view the list of all actions and risks.

Too formal	Too casual	Our style
When the Setup wizard starts, follow the instructions in the wizard to set up the node and join it to the cluster. The following steps walk you through the steps in the wizard.	The Setup wizard appears (almost like magic), to guide you through the simple process of setting up the node and joining it to the cluster.	Follow the instructions in the Setup wizard to set up the node and join it to the cluster.
You can choose from among multiple content format types to install and set up your new system. Each format type provides complete instructions. Choose the format type that most closely matches the way you like to learn.	How do you want to set up and install your system? We provide instructions in multiple content format types, but only you know how you like to learn.	Choose a content format type to guide you through installing and setting up your system.

Write simply

Avoid big and confusing words. Keep it simple. You're explaining something to a professional colleague, not showing off your vocabulary.

Rather than this: "Dissociate the user from your NetApp Cloud Central account."

Do this: "Remove the user from your NetApp Cloud Central account."

Write minimally

Short, simple sentences make content easier to read or scan. It's okay to use a longer sentence every now and then but follow it with a shorter one. Like this.

Rather than this: "To replicate data between a Cloud Volumes ONTAP system in AWS and ONTAP systems in other networks, you must have a VPN connection between the Amazon VPC and the other network—for example, an Azure VNet or your corporate network."

Do this: "Data replication between networks requires connection through a VPN. For example, between your Amazon VPC and your corporate network or between AWS and Azure."

See also [minimalism].

Write actively

Avoiding passive voice is a standard rule for tech writing, but it's especially important to use active voice when you want to sound conversational.

Rather than this: "The required permissions must be provided before you deploy your first cluster."

Do this: "Provide the required permissions before you deploy your first cluster."

See also active voice (versus "passive voice").

Use inclusive language

NetApp believes that its product documentation should not contain discriminatory, exclusive language. The words that we use can make a difference between forging a positive relationship with our customers or alienating them. Especially with written words, impact is more important than intent.

As you create content for NetApp products, avoid language that can be interpreted as degrading, racist, sexist, or otherwise oppressive. Instead, use language that is accessible and welcoming to everyone who needs to use the documentation. For example, instead of "master/slave" use "primary/secondary."

We know that we have work to do in order to remove all non-inclusive language from our documentation and our products. We're actively working on our standards and best practices and expect to update this section with additional guidance in the future.

Get to the point

Start with what's important to the user. Find out what the user is trying to do and focus on helping them achieve that goal.

Rather than this: "Cloud Sync can sync data from one NFS server to another NFS server using data-in-flight encryption. Encrypting the data can help if you have strict security policies for transferring data over networks."

Do this: "If your business has strict security policies, use data-in-flight encryption to sync data between NFS servers in different networks."

Use lots of visuals

Most people are visual learners. Use videos, diagrams, and screenshots to improve learning. Visuals also help to break up blocks of text.

Examples

- Example #1
- Example #2

See also [graphics].

Create scannable content

Use headings, lists, and tables to help users scan for what they want.

Examples

- Example #1
- Example #2

Focus on a user goal or a specific aspect of that goal

If you're describing how to complete a series of tasks, put it all on one page in a series of sections, including conceptual and reference-based information. Don't break up your page into several mini-pages—that requires too much clicking. At the same time, don't create long, intimidating pages. Use your best judgment to decide when a page is too long.

Examples

- [Example #1](#)
- [Example #2](#)

Organize content around the user's goal

Help users find the info they need when they need it. Get them in and out of the docs as quickly as possible, by organizing the content as follows:

The first entry in the left-hand navigation (high level)

Organize content around the goals the user is trying to achieve. For example, getting started or protecting data.

The second entries in the navigation (medium level)

Organize content around the broad tasks that compose the goals. For example, setting up disaster recovery or setting up data protection.

Individual pages (detailed level)

Organize content around the individual tasks that compose the broad tasks, with each one focusing on a single learning or doing aspect of that broad task. For example, the tasks required to set up disaster recovery.

Write for a global audience

We write for our customers and partners around the world, and much of our content is translated using Neural Machine Translation tools or human translation. Follow these guidelines for clearer writing and easier translation:

- Write short, simple sentences.
- Use standard grammar and punctuation.
- Use one word for one meaning and one meaning for one word.
- Use common contractions.
- Use graphics to clarify or replace text.
- Avoid embedding text in graphics.
- Avoid having three or more nouns in a string.
- Avoid unclear antecedents.
- Avoid jargon, colloquialisms, and metaphors.
- Avoid nontechnical examples.
- Avoid using hard returns and spacing.
- Don't use humor or irony.
- Don't use discriminatory content.
- Don't use gender-biased language unless you're writing for a specific persona.

A to Z guidelines

active voice (versus "passive voice")

In active voice, the subject of the sentence is the doer of the action:

- If you shut down the system improperly, the interface displays a warning message.
- NetApp received the contract.

Active voice keeps writing crisp and clear. Use active voice and address users directly as "you" unless you have a specific reason to use passive voice.

In passive voice, the doer of the action is unclear:

- A warning message is displayed if the system is shut down improperly.
- NetApp was awarded the contract.

Use passive voice when:

- You don't know who or what performed the action.
- You want to avoid blaming users for the results of an action.
- You can't write around it, such as for some prerequisite information.

For additional verb conventions, see:

- Microsoft Writing Style Guide
- Chicago Manual of Style
- Merriam-Webster Dictionary Online

admonitions

Use the following labels to identify content separately from the main content flow:

- NOTE

  Use NOTE for important information that must be distinct from the rest of the text. Avoid using NOTE for "nice to know" information that isn't required for users to learn about the task or complete the task.

- TIP

  Use TIP sparingly, if at all, because our policy is to always document best-practice information by default. If necessary, use TIP to contain best-practice information that helps users use a product or complete a step or task easily and efficiently.

- CAUTION

  Use CAUTION to warn users about conditions or procedures that can cause personal injury that is not lethal or extremely hazardous.

after (versus "once")

- Use "after" to indicate a chronology: "Turn on your computer after you plug it in."
- Use "once" only to mean "one time."

also

- Use "also" to mean "additionally."
- Don't use "also" to mean "alternatively."

and/or

Choose the more precise term if there is one. If neither term is more precise than the other, use "and/or."

as

Don't use "as" to mean "because."

by using (versus "using" or "with")

- Use "by using" when the entity that is doing the using is the subject: "You can add new components to the repository by using the Components menu."
- You can begin a sentence with either "using" or "with," which are sometimes acceptable with product names: "Using SnapDrive, you can manage virtual disks and Snapshot copies in a Windows environment."

can (versus "might," "may," "should," or "must")

- Use "can" to indicate capability: "You can commit your changes at any time during this procedure."
- Use "might" to indicate possibility: "Downloading multiple programs might affect processing time."
- Don't use "may," which is ambiguous because it could mean either capability or permission.
- Use "should" to indicate a recommended but optional action. Consider using an alternative phrase instead, such as "we recommend."
- Avoid using "must" because it's passive. Consider restating the thought as an instruction using imperative voice. If you do use "must," use it to indicate a required action or condition.

capitalization

Use sentence-style capitalization (lowercase) for almost everything. Only capitalize:

- The first word of sentences and headings, including table headings
- The first word of list items, including sentence fragments
- Proper nouns
- Doc titles and subtitles (capitalize all major words and prepositions of five or more letters)
- UI elements, but only if they are capitalized in the interface. Otherwise, use lowercase.

CAUTION notices

Use CAUTION to warn users about conditions or procedures that can cause personal injury that is not lethal or extremely hazardous.

See [admonitions] for other labels that identify content separately from the main content flow.

consistency

"Write like you speak when you're explaining something to a professional colleague" means something different to everyone. Our professionally conversational style helps connect us to users—and increases the frequency of minor inconsistencies among multiple contributing authors:

- Focus on making the content clear and easy to use. If all content is clear and easy to use, minor inconsistencies don't matter.
- Be consistent within the page you're writing.
- Always follow the guidelines in Write for a global audience.

contractions

Contractions reinforce a conversational tone, and many contractions are easy to understand and translate.

- Do use contractions like these, which are easy to understand and translate:

aren't	you're
isn't	we're
wasn't	it's
weren't	let's
didn't	we'll (if future tense is required)
doesn't	won't (if future tense is required)
don't	you'll (if future tense is required)

- Don't use contractions like these, which are hard to understand and translate:

would've	should've
wouldn't've	shouldn't've
could've	couldn't've

ensure (versus "confirm" or "verify")

- Use "ensure" to mean "to make certain." Include "that," as appropriate: "Ensure that there is sufficient white space around illustrations."
- Never use "ensure" to imply a promise or guarantee: "Use Cloud Manager to ensure that you can provision NFS and CIFS volumes on ONTAP clusters."
- Use "confirm" or "verify" when you mean that the user should double-check something that already exists or has happened already: "Verify that NFS is set up on the cluster."

graphics

Continually evaluate content for opportunities to include helpful illustrations, diagrams, flow charts, screen captures, or other visual references. Graphics often convey complex concepts and steps more clearly than text.

- Include a description of what the illustration is intended to communicate: "The following illustration shows the AC power supply LEDs on the back panel.
- Refer to the location of the illustration as "following" or "preceding," not "above" or "below."

grammar

Except where noted otherwise, follow the grammar, punctuation, and spelling conventions detailed in:

- Microsoft Writing Style Guide
- Chicago Manual of Style
- Merriam-Webster Dictionary Online

if not

Don't use "if not" by itself to refer to the previous sentence:

Rather than this: "The computer should be off. If not, turn it off."

Do this: "Verify that the computer is off."

if (versus "whether" or "when")

- Use "if" to indicate a condition, such as in "if this, then that" constructions.
- Use "whether" when there is a stated or implied "or not" condition. To ease translation, it is often best to replace "whether or not" with "whether" alone.
- Use "when" to indicate a passage of time.

imperative voice

- Use imperative voice for steps, directives, requests, and headings for lists of user actions:
  - "On the Working Environments page, click Discover and select ONTAP Cluster."
  - "Rotate the cam handle so that it is flush against the power supply."
- Consider using imperative voice to replace passive voice:

  Rather than this: "The required permissions must be provided before you deploy your first cluster."

  Do this: "Provide the required permissions before you deploy your first cluster."

- Avoid using imperative voice to embed steps in conceptual and reference information.

IP and IPv6 addresses

For IP addresses (including IPv6) in examples, it's safe to include any address that starts with "10.x".

future functionality or releases

Don't refer to the timing or content of upcoming product releases or features, other than to say that a feature or function is "not currently supported."

KB articles: referring to

Refer to KB (NetApp Knowledgebase) articles in content when appropriate. For resources pages and GitHub content, put the link in running text.

lists

Lists of info are usually easier to scan and absorb than blocks of text. Consider ways to simplify complex info by presenting it in list form. Here are some general guidelines, but use your judgment:

- Make sure that the reason for the list is clear. Introduce the list with a complete sentence, a sentence fragment with a colon, or a heading.
- Lists should have between two and seven entries. In general, the shorter the info in each entry, the more entries you can add while keeping the list scannable.
- List entries should be as scannable as possible. Avoid blocks of text that get in the way of keeping list entries scannable.
- List entries should start with a capital letter, and list entries should be grammatically parallel. For example, start each entry with a noun or a verb:
  - If all list entries are complete sentences, end them with periods.
  - If all list entries are sentence fragments, don't end them with periods.
- List entries should be ordered in a logical way, such as alphabetically or chronologically.

localization

See Write for a global audience.

minimalism

- Do users need this content at this place, at this time?
- Can I present the content in fewer words without sounding too formal or too casual?
- Can I shorten or simplify a long sentence or break it into two or more sentences?
- Can I use a list to make the content more scannable?
- Can I use a graphic to augment or replace a block of text?

NOTE information

Use NOTE for important information that must be distinct from the rest of the text. Avoid using NOTE for "nice to know" information that isn't required for users to learn about the task or complete the task.

See [admonitions] for other labels that identify content separately from the main content flow.

numbers

- Use Arabic numerals for 10 and all numbers greater than 10, with these exceptions:
    - If you begin a sentence with a number, use a word, not an Arabic numeral.
    - Use words (not numerals) for approximate numbers.
- Use words for numbers that are less than 10.
- If a sentence contains a mixture of numbers less than 10 and greater than 10, use Arabic numerals for all numbers.
- For additional number conventions, see:
    - Microsoft Writing Style Guide
    - Chicago Manual of Style

plagiarism

We document NetApp products and the interaction of NetApp products with third-party products. We do not document third-party products. We should never need to copy and paste third-party content into our docs and we should never do it.

prerequisites

Prerequisites identify the conditions that must exist or the actions that users must have completed before they start the current task.

- Identify the nature of the content with a heading, such as "Prerequisites," "Before you begin," or "Before you get started."
- Use passive voice for prerequisite wording if it makes sense to do so:
    - "NFS or CIFS must be set up on the cluster."
    - "You must have the cluster management IP address and the password for the admin user account to add the cluster to Cloud Manager."
- Clarify the prerequisite as needed: "NFS or CIFS must be set up on the cluster. You can set up NFS and CIFS using System Manager or the CLI."
- Consider other ways to present the information, for example whether it would be appropriate to reword the content as the first step in the current task:
    - Prerequisite: "You must have the required permissions before you deploy your first cluster."
    - Step: "Provide the required permissions to deploy your first cluster."

prior (versus "before," "previous," or "preceding")

- If possible, replace "prior" with "before."
- If you can't use "before," use "prior" as an adjective to refer to something that occurred earlier in time or with a higher order of importance.
- Use "previous" to indicate something that occurred at an unspecified time earlier.
- Use "preceding" to indicate something that occurred immediately beforehand.

punctuation

Keep it simple. In general, the more punctuation included in a sentence, the more brain cells it takes to understand.

- Use a serial comma (Oxford comma) before the conjunction ("and" or "or") in a narrative list of three or more items.
- Limit use of semicolons and colons.
- Except where noted otherwise, follow the grammar, punctuation, and spelling conventions detailed in:
    - Microsoft Writing Style Guide
    - Chicago Manual of Style
    - Merriam-Webster Dictionary Online

since

Use "since" to indicate a passage of time. Don't use "since" to mean "because."

spelling

Except where noted otherwise, follow the grammar, punctuation, and spelling conventions detailed in:

- Microsoft Writing Style Guide
- Chicago Manual of Style
- Merriam-Webster Dictionary Online

that (versus "which" or "who")

- Use "that" (without a trailing comma) to introduce clauses that are required for the sentence to make sense.
- Use "that" even if the sentence is clear in English without it: "Verify that the computer is off."
- Use "which" (with a trailing comma) to introduce clauses that add supporting information but are not required for the sentence to make sense.
- Use "who" to introduce clauses referring to people.

TIP information

Use TIP sparingly, if at all, because our policy is to always document best-practice information by default. If necessary, use TIP to contain best-practice information that helps users use a product or complete a step or task easily and efficiently.

See [admonitions] for other labels that identify content separately from the main content flow.

trademarks

We don't include trademark symbols in most of our technical content because the legal statements in our templates are sufficient. However, we do follow all usage rules when using NetApp trademarked terms:

- Use trademarked terms (with or without the symbol) only as adjectives, never as nouns, verbs, or verbals.
- Don't abbreviate, hyphenate, or italicize trademarked terms.

- Don't pluralize trademarked terms. If a plural form is required, use the trademarked name as an adjective that modifies a plural noun.
- Don't use a possessive form of a trademarked term. You can use the possessive form of company names, such as NetApp, when the names are being used in a general sense, rather than as trademarked terms.

user interface

Rely on the interface as much as possible to guide the user.

General guidelines

Our style for documenting UIs is simple and minimal:

- Assume that the user is using the interface while reading the content.
- Rely on the interface to guide the user:
  - Don't walk the user through a wizard or screen step by step. Only call out important things that are not apparent from the interface.
  - Don't include "click OK" or "click Save" or "the volume is created" or anything else that's obvious to someone doing the task.
  - Assume success. Unless you expect an operation to fail most of the time, do not document the failure path. Assume that the interface provides proper guidance.
- Don't use "click" at all. Always use "select" because that word covers mouse, touch, keyboard, and any other way of making a choice.
- Focus content on a workflow that addresses a customer use case and on getting the user to the right place in the interface to start the workflow.
- Always document the one best way to achieve the user goal.
- If the workflow requires a significant decision, make sure to document a decision rule.
- Use the minimum number of steps necessary for most users most of the time.

Naming UI elements

Avoid documenting to the level of granularity that requires naming UI elements. Rely on the interface to guide the user through the specifics of the interaction. If you must get that specific, name the label on the element. For example, "Select the desired volume" or "Select 'Use existing volume.'" There is no need to name menus or radio buttons or checkboxes, just use the label.

For icons that users must select, use an image of the icon. Don't try to name it. This rule applies to icons like the arrow, pencil, gear, kabob, hamburger, and so on.

Representing displayed labels

Follow the spelling and capitalization used by the user interface when identifying labels. If a label is followed by ellipses, do not include the ellipses when naming the object. Encourage developers to use title-style capitalization for user interface labels, to make writing about them easier.

Using screen captures

An occasional screen capture ("screenshot") helps users be confident that they are in the right place in an interface when starting or changing interfaces during a workflow. Don't use screen captures to show what data to enter or what value to select.

while (versus "although")

- Use "while" to indicate something occurring in time.
- Use "although" to represent an activity that occurs at nearly the same time or shortly after another activity.

workflow

Users read our content to accomplish a specific goal. Users want to find the content they need, accomplish their goals, and go home to their families. Our job is not to document products or features, our job is to document user goals. Workflows are the most direct way to help users accomplish their goals.

A workflow is a series of steps or subtasks that describes how to achieve a user goal. The scope of a workflow is a complete goal.

For example, the steps to create a volume would not be a workflow, because creating a volume in itself is not a complete goal. The steps to make storage available to an ESX server could be a workflow. The steps would include not only creating a volume, but exporting the volume, setting any necessary permissions, creating a network interface, and so on.
Workflows are derived from customer use cases. A workflow shows only the one best way to achieve the goal.

AsciiDoc reference

AsciiDoc is a lightweight markup language, similar to Markdown. We chose AsciiDoc over standard Markdown because it provides more out-of-box capabilities. While it's more powerful, it's still simple to use. Refer to the sections below to get started writing in AsciiDoc.

See the AsciiDoctor User Manual for additional help.

The basics

You need to know a few things to contribute simple doc updates.

Headings

```
= Page title
== Level 1 section
=== Level 2 section
==== Level 3 section
===== Level 4 section
```

You can have only one page title, but you can have multiple section titles. For example, you might have three level 1 sections that include level 2 and 3 sections:

```
= Page title
== Level 1 section
=== Level 2 section
== Level 1 section
== Level 1 section
=== Level 2 section
==== Level 3 section
```

Bold text

```
*Text*
```

Italic text

```
_Text_
```

Bulleted lists

```
* Item 1
+
Continuation text for the previous list item.

* Item 2
** Item 2a

* Item 3
```

💡 The + is a list continuation. It keeps the text inline with the list item. Omitting the + affects the formatting of that line.

Labeled lists

```
Item 1::
Description 1

Item 2::
Description 2
```

or

```
[horizontal]
Item 1::
Description 1

Item 2::
Description 2
```

When you add [horizontal] above item 1, the label and description appear on the same line. That works well when you have very short descriptions.

Example without [horizontal]

Item 1
  Description 1

Item 2
  Description 2

Example with [horizontal]

Item 1   Description 1

Item 2   Description 2

Steps

```
.Steps

. Step 1

. Step 2
+
Info for step 2

. Step 3
.. Step 3a
.. Step 3b

. Step 4
```

> 💡 The + is a list continuation. It keeps the text inline with the list item. Omitting the + affects the formatting of that line.

Images

```
image:file.png["alt text"]
```

alt text means alternative text. It describes the image that appears on the page. The primary use is for visually-impaired users who use screen readers.

Two notes:

1. It's best to enclose alt text in quotes because punctuation like commas can affect the ability to transform the content from AsciiDoc to HTML.

2. The AsciiDoctor docs state that *block images* should be on their own line with *two* colons:
   `image::file.png`

   But we prefer to use one colon, as shown above. Using one colon has the same result and it works better with our internal tools.

Videos

Hosted on YouTube:

```
video::id[youtube]
```

Hosted locally in GitHub:

```
video::https://docs.netapp.com/us-en/contribute//media/file.mp4
```

Links

The syntax that you should use depends on what you're linking to:

- Link to an external site
- Link to a section on the same page
- Link to another page in the docs

Link to an external site

```
url[link text^]
```

The ^ opens the link in a new browser tab.

Link to a section on the same page

```
<<section_title>>
```

For example:

```
For more details, see <<Headings>>.
```

The link text can be something other than the section title:

```
<<section_title,Different link text>>
```

For example:

```
<<Headings,Learn the syntax for headings>>.
```

Link to another page in the docs

The file needs to be in the same GitHub repository:

```
xref:{relative_path}<file_name>.html[Link text]
```

To link directly to a section in the file, add a hash (#) and the section's title:

```
xref:{relative_path}<file_name>.html#<section-name-using-dashes-and-all-
lower-case>[Link text]
```

For example:

```
xref:{relative_path}style.html#use-simple-words[Use simple words]
```

Notes, tips, and cautions

You might want to draw attention to certain statements by using notes, tips, or caution statements. Format them as follows:

```
NOTE: text

TIP: text

CAUTION: text
```

Use each of these sparingly. You don't want to create pages that are full of notes and tips. They become less meaningful if you do.

Here's what each of these looks like when the AsciiDoc content is turned into HTML:

This is a note. It includes extra info that a reader might need to know.

A tip provides useful information that can help a user do something or understand something.

A caution advises the reader to act carefully. Use this in rare circumstances.

Advanced stuff

If you're authoring new content, you'll want to review this section for some nitty-gritty details.

Document headers

Each AsciiDoc file includes two types of headers. The first is for GitHub and the second is for AsciiDoctor, which is the publishing tool that turns the AsciiDoc content into HTML.

The GitHub header is the very first set of content in the .adoc file. It needs to include the following:

```
---
sidebar: sidebar
permalink: <file_name>.html
keywords: keyword1, keyword2, keyword3, keyword4, keyword5
summary: "A summary."
---
```

The keywords and summary directly affect search results. In fact, the summary itself displays in the search results. You should make sure that it's user friendly. The best practice is to have the summary mirror your lead paragraph.

> 💡 It's best to enclose the summary in quotes because punctuation like colons can affect the ability to transform the content from AsciiDoc to HTML.

The next header goes directly underneath the document title (see Headings). This header should include the following:

```
:hardbreaks:
:icons: font
:linkattrs:
:relative_path: ./
:imagesdir: {root_path}{relative_path}./media/
```

You won't need to touch any of the parameters in this heading. Just paste it in and forget it.

Lead paragraph

The first paragraph that appears under the document title should include the following syntax directly above it:

```
[.lead]
This is my lead paragraph for this content.
```

[.lead] applies CSS formatting to the lead paragraph, which has a different format than the text that follows it.

Tables

Here's syntax for a basic table:

```
[cols=2*,options="header",cols="25,75"]
|===
| heading column 1
| heading column 2
| row 1 column 1 | row 1 column 2
| row 2 column 1 | row 2 column 2
|===
```

There are *many* ways to format a table. Refer to the AsciiDoctor User Manual for additional help.

> 💡 If a cell contains formatted content like bulleted lists, it's best to add an "a" in the column header to enable formatting. For example:
> [cols="2,2,4a" options="header"]

See the AsciiDoc Syntax Quick Reference for more table examples.

Task headings

If you're explaining how to perform a task, you might include introductory information before you get to the steps. And you might need to say what to do after completing the steps. If you do, it's best to organize that information using headers, which enables scanning.

Use the following headings as needed:

What you'll need

The information the user needs to complete the task.

About this task

Some extra contextual info the user might need to know about this task.

Steps

The individual steps to complete the task.

What's next?

What the user should do next.

Each of these should include a . right before the text, like so:

```
.What you'll need
.About this task
.Steps
.What's next?
```

This syntax applies bold text in a larger font.

Command syntax

When providing command input, enclose the command within ` to apply monospace font:

```
`volume show -is-encrypted true`
```

Here's what that looks like:

```
volume show -is-encrypted true
```

For command output or command examples, use the following syntax:

```
----
cluster2::> volume show -is-encrypted true

Vserver  Volume  Aggregate  State  Type  Size  Available  Used
-------  ------  ---------  -----  ----  -----  ---------  ----
vs1      vol1    aggr2      online  RW   200GB    160.0GB  20%
----
```

The four dashes enable you to enter separate lines of text that appear together.
Here's the result:

```
cluster2::> volume show -is-encrypted true

Vserver  Volume  Aggregate  State  Type  Size  Available  Used
-------  ------  ---------  -----  ----  -----  ---------  ----
vs1      vol1    aggr2      online  RW   200GB    160.0GB  20%
```

Variable text

In commands and command output, enclose variable text within underscores to apply italics.

```
`vserver nfs modify -vserver _name_ -showmount enabled`
```

Here's what that command and the variable text looks like:

```
vserver nfs modify -vserver name -showmount enabled
```

ⓘ   Underscores aren't supported with code syntax highlighting at this time.

Code syntax highlighting

Code syntax highlighting provides a developer-focused solution for documenting the most popular languages.

Output example 1

```
POST https://netapp-cloud-account.auth0.com/oauth/token
Header: Content-Type: application/json
Body:
{
            "username": "<user_email>",
            "scope": "profile",
            "audience": "https://api.cloud.netapp.com",
            "client_id": "UaVhOIXMWQs5i1WdDxauXe5Mqkb34NJQ",
            "grant_type": "password",
            "password": "<user_password>"
}
```

Output example 2

```json
[
    {
        "header": {
            "requestId": "init",
            "clientId": "init",
            "agentId": "init"
        },
        "payload": {
            "init": {}
        },
        "id": "5801"
    }
]
```

Supported languages

- bash
- curl
- https
- json
- powershell
- puppet
- python
- yaml

Implementation

Copy and paste the following syntax and then add a supported language and the code:

```
[source,<language>]
<code>
```

For example:

```
[source,curl]
curl -s https:///v1/ \
-H accept:application/json \
-H "Content-type: application/json" \
-H api-key: \
-H secret-key: \
-X [GET,POST,PUT,DELETE]
```

Content reuse

If you have a chunk of content that's repeated across different pages, you can easily write it once and reuse it across those pages. Reuse is possible from within the same repository and across repositories. Here's how it works.

1. Create a folder in your repository named _include

   For example, take a look at the Cloud Tiering repository.

2. Add a .adoc file in that folder that includes the content that you'd like to reuse.

   It can be a sentence, a list, a table, one or more sections, and so on. Don't include anything else in the file—no headers or anything.

3. Now go to the files where you'd like to reuse that content.

4. If you're reusing the content from within the *same* GitHub repository, use the following syntax on a line by itself:

   ```
   include::_include/<filename>.adoc[]
   ```

   For example:

   ```
   include::_include/s3regions.adoc[]
   ```

5. If you're reusing the content in a *different* repository, use the following syntax on a line by itself:

   ```
   include::https://raw.githubusercontent.com/NetAppDocs/<reponame>/main/_i
   nclude/<filename>.adoc[]
   ```

   For example:

```
include::https://raw.githubusercontent.com/NetAppDocs/cloud-
tiering/main/_include/s3regions.adoc[]
```

That's it!

If you want to learn more about the include directive, check out the AsciiDoctor User Manual.

Images with links

You can add a link to an image so it acts like a "button."

Output example

In the following screenshot, "View in Documentation Center" is an image that contains a link.
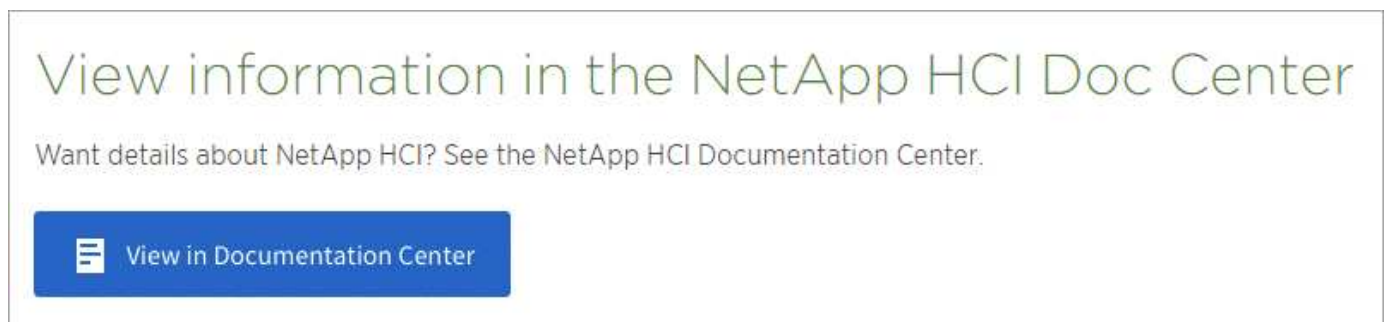
Syntax

Use the following syntax when adding your image:

```
image:<file_name>.<ext>[alt=<text>,link=<url>,window=_blank]
```

"window=_blank" opens the link in a new browser tab (or window).

For example:

```
image:btn-view-in-doc-center.png[alt=A button titled View in Documentation
Center that links to the NetApp HCI Doc
Center,link=http://docs.netapp.com/hci/index.jsp,window=_blank]
```