

# Voice and tone

## Splunk voice and tone

The way you speak and the way you write is likely very different. Traditional technical writing tends to be dry and formal, whereas conversational language is often verbose or colloquial. Splunk documentation finds a balance between the two: Splunk docs are casual and approachable, yet succinct and direct.

When you create written content, shape your docs around your reader's needs and expectations, and write to them as though they are a professional acquaintance. Aim to be confident, friendly, and comprehensive, and not insensitive, saccharine, or complicated.

Here are a few guidelines you can follow to achieve a balanced tone in your writing:

- Avoid jargon or idioms. See [Use plain language](#).
- Write in active voice and present tense whenever possible. See [Be active and present](#).
- Write in the indicative and imperative mood, and avoid using the subjunctive mood. See [Write in indicative or imperative mood](#).
- Avoid attributing human characteristics to inanimate objects. See [Avoid anthropomorphisms](#).
- Use contractions, such as isn't, don't, and can't. See [Contractions](#).
- Write precisely and concretely. See [Misplaced modifiers](#).
- Avoid qualitative language, such as calling things easy or simple. What's easy for one user might be challenging for another.

## Examples

The following table shows examples of writing that are too formal, too casual, and just right:

Too formal	Too casual	Just right
Note that Splunk Enterprise Security automatically enables SSL; therefore, confirm that the protocol in your web browser is "https" (for example: <code>https://splunkserver:8000</code> ).	ES enables SSL, so you might want to make sure that your web browser is using HTTPS, like in <code>https://splunkserver:8000</code> .	Splunk Enterprise Security enables SSL. Check that the web browser protocol uses HTTPS. For example, <code>https://splunkserver:8000</code> .
Please select the <b>Complete</b> button to complete the process and display the result.	When you select <b>Complete</b> and you see the result, you know you are done. Way to go!	Select <b>Complete</b> .
The information about the KV store status can be retrieved from the <code>kvstore/status</code> endpoint via the GET method.	Do a GET on the <code>kvstore/status</code> endpoint.	Submit a GET request to the <code>kvstore/status</code> endpoint to access KV store status information.
Forwarder management is a graphical user interface (GUI) that is built on top of the deployment server and offers a streamlined solution to configure the deployment server and monitor the status of deployment updates.	The forwarder management GUI lives on top of the deployment server. The GUI makes it easy to configure your server while monitoring any updates that might come up.	You can configure the deployment server and monitor the status of deployment updates with the forwarder management graphical interface.
Metrics can be used for the investigation, monitoring, and troubleshooting of your pipeline in real time.	You can use metrics to do some really cool things like look into, keep an eye on, and tweak your pipeline as the data is coming in.	Use metrics to investigate, monitor, and troubleshoot your pipeline in real time.
A gauge is a metric that is composed of a single numerical value that can arbitrarily increase or	A gauge is a number that can go up and down for a couple of reasons, like if it gets	A gauge represents a single numerical value that changes based on what you're

Too formal	Too casual	Just right
decrease depending upon the value of which the metric is tracking.	hotter or colder in a server room.	tracking.

## Use plain language

Writing with plain language helps the widest range of users understand your content. Splunk documentation readers are a global audience whose first language might not be English. Using plain language increases the chance that readers understand what you wrote the first time they read it.

Plain language also makes translation easier and documentation more accessible. For more information on how using plain language increases accessibility, see [Write accessible documentation](#).

Consider the following tips for using plain language in your writing:

- Avoid obscure words.
- Use simple but complete sentences. Often, you can communicate the same message in 20 words instead of 50 words.
- Identify all elements and knowledge objects with indicating nouns. Don't make readers guess what to look for in the product. See [Choose clarity over concision](#) in this topic.
- Define all acronyms and initialisms at first use, and avoid abbreviations for language. For more information on abbreviations, see [Abbreviations](#).
- Be consistent. Use the same term to mean the same thing in one topic as you do in another topic.
- Avoid ambiguous references between a pronoun and its antecedent. See [Vague pronouns](#).



## Avoid jargon and complex terminology

Jargon and complex terminology are terms that are specific to a company, profession, or field. These terms are often referred to as technical language and can confuse and frustrate readers, so avoid them whenever possible. If you have to use complex or unfamiliar Splunk-specific terminology, link to the term in the Splexicon at first use instead of redefining the term. For more information on linking to the Splexicon, see [Formatting links in Splunk documentation](#).

Consider the following tips to help you avoid jargon in your writing:

- Take into account your audience's level of knowledge.
- Consider if the user needs to know the term to understand the documentation or if you can rephrase the sentence using simpler language.
- Refer to the Usage dictionary for examples of words and phrases that are unnecessarily complex and their alternatives. See the [Usage dictionary](#).

Review the following table for examples of how to translate jargon into plain language:



 Do this	 Don't do this
Use the default settings.	Use the out-of-the-box settings.
You can deploy Splunk Enterprise on-premises.	You can deploy Splunk Enterprise on-prem.
Make sure that no deployments conflict with one another.	Perform an execution of the process steps with a core focus of ensuring that the deployments don't conflict.
Troubleshoot a hard-to-find case by searching for a specific trace.	Troubleshoot a needle-in-a-haystack case by searching for a specific trace.

 Do this	 Don't do this

## Choose clarity over concision

Write in simple, concise sentences with a straightforward tone, and regardless of sentence length, always use indicating nouns to identify special elements or knowledge objects. Naming elements and objects removes ambiguity about what the element represents in the product and the instructions. Don't remove indicating nouns for the sake of brevity or to sound casual.

Review the following table for examples of how to include indicating nouns:

 Do this	 Don't do this
Submit a GET request.	Do a GET.
Use the <code>map_get()</code> function to extract nested values from the <code>attributes</code> field.	<code>map_get()</code> extracts nested values from <code>attributes</code> .
The <code>auth</code> header is included by default, unless the <code>noAuth</code> flag is set.	<code>Auth</code> header is included by default, unless <code>noAuth</code> is set.
If you use the <code>start.ini</code> file to define JVM arguments, add the <code>javaagent</code> argument after the <code>--exec</code> option.	If you use <code>start.ini</code> to define JVM arguments, add <code>javaagent</code> after <code>--exec</code> .
From the command line, run the following command to start your app in develop mode:  <code>yarn run start</code>	Run the following to start in dev mode:  <code>yarn run start</code>

## Be active and present



Strive for active voice and present tense in your documentation. Writing in active voice allows readers to place themselves as the subject in the documentation, and present tense tells readers what they can expect as they work through the topics.

While using passive voice and future tense is sometimes appropriate, keep your audience in mind by writing in plain language and centering your writing around the user as much as possible. For more information, see [Use plain language](#).

### Active voice

Active voice makes content clear by indicating who or what performs the action, even when the subject is implied. Minimize the use of passive voice in your writing.

Review the following table for examples rewritten to use active voice:

 Do this	 Don't do this
Select <b>Print</b> to print the document.	When the <b>Print</b> button is selected, the document is printed.
The forwarder sends data to the receiver.	Data is sent to the receiver.

### When is it okay to use passive voice?



Write in passive voice carefully and sparingly. The following examples describe situations where you might choose to use passive voice instead of active voice:

- When you want to avoid placing blame on the user, such as in an error message
- When defining the actor in the sentence is irrelevant or awkward

## Present tense

Present tense allows readers to experience the documentation while they read it. Minimize the use of past tense or future tense in your writing.

Review the following table for examples rewritten to use present tense:

 Do this	 Don't do this
The summary statistics continue to calculate allowed items, but the items don't appear in the dashboard.	The tool processes all apps before they are published to Splunkbase.
All apps are processed through the tool before they are published to Splunkbase.	The tool will now process all apps before being published to Splunkbase.
After the search is completed, the table updates automatically.	After the search was completed, the table updated automatically.

### ***When is it okay to use future tense?***

Write in future tense carefully and sparingly. Do not use future tense if you can rewrite the sentence to use present tense instead. In general, strive to keep documentation timeless, and avoid writing about time-sensitive information that might become irrelevant in the near future. For more information, see [Timeless](#).

The following examples describe situations where you might choose to use future tense:

- In tutorial content, when describing what the reader will achieve during the tutorial part
- In deprecation or end-of-life statements for a product

## Write in indicative or imperative mood

The form that a verb takes tells a reader its mood or how to regard the sentence. The following major moods appear in American English writing: indicative, imperative, and subjunctive. In Splunk documentation, write with indicative or imperative mood, and avoid using the subjunctive mood.

### **Use indicative or imperative mood**

The indicative mood states facts. The imperative mood expresses commands or requests. Use these moods when writing Splunk docs.

See the following examples:

- You can send data to the indexer by using a forwarder.
- Send data to the indexer by using a forwarder.
- The forwarder sends data to the indexer.



### **Avoid subjunctive mood**

The subjunctive mood expresses doubt and causes confusion over whether you're making a recommendation or stating a requirement. In general, don't write specific recommendations. See [Recommendations](#).

Write clear instructions for Splunk customers and avoid the following subjunctive mood verbs:

- Should
- Would
- Could



Review the following table for examples of sentences correctly using the indicative and imperative mood and their subjunctive mood counterparts:

 <b>Do this</b>	 <b>Don't do this</b>
For security reasons, give only administrators access to this instance.	For security reasons, only administrators should have access to this instance.
The example shows a type of script you can create for your deployment.	The example shows a type of script you would create for your deployment.
Start by setting up a new stanza in the transforms.conf file.	You could start by setting up a new stanza in transforms.conf.

## Avoid anthropomorphisms

Anthropomorphisms attribute human characteristics to inanimate objects. Computers don't think, want, worry, or do other things that are uniquely human. In addition, anthropomorphisms can be imprecise, unclear, and do not translate well. Avoid anthropomorphisms in your writing to promote clear documentation written in plain language. For more information, see [Use plain language](#).

Review the following table for examples of anthropomorphisms and how to rewrite them:

 <b>Do this</b>	 <b>Don't do this</b>
The forwarder sends data to the location that you set in the configuration file.	The configuration file tells the forwarder where to send the data.
Processing components process the data.	Processing components handle the data.
The server sends a request to other parts of the network.	The server wants to talk with other parts of the network.
The software creates a bucket for each unique IP address it detects.	The software creates a bucket for each unique IP address it sees.
A receiver prepares to accept data that a forwarder sends to it.	A receiver listens for data that a forwarder sends to it.
The metric shows how many CPU cores the process is using.	The metric tells you how many CPU cores the process is using.

## Writing with commonly used anthropomorphisms

Some anthropomorphic terms, like ingest and consume, are common in technical lexicon. You can include these terms in your writing if they are well understood by your audience. In general, strive to write clearly using plain language as much as possible.

See the following examples of acceptable anthropomorphisms:

- Now the service can ingest data from a third-party source.
- The service consumes data from the repository.
- The process handles search requests.
- The algorithm learns by estimating the coefficient values in the test set using the training set data.
- A single search head manages searches and interacts with users.