Contents of a GitHub Docs article

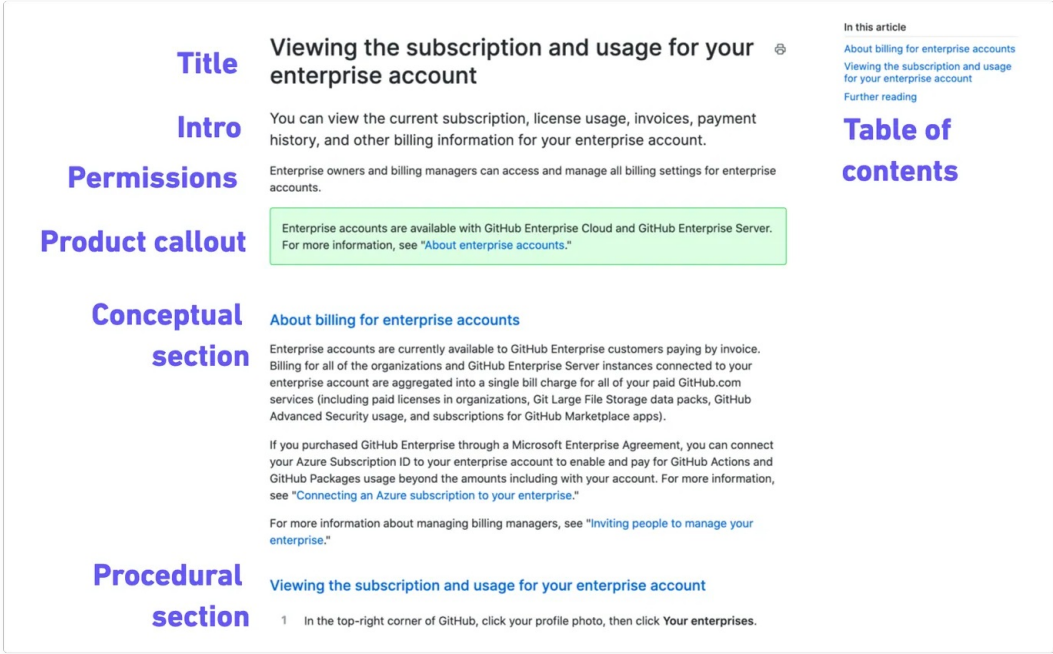
In this article

Every article includes a few standard elements, and may include conditional or optional elements. We also use a standard order for content within an article.

Articles in the "Contributing to GitHub Docs" section refer to the documentation itself and are a resource for GitHub staff and open source contributors.

About the structure of an article

Within an article, there is a standard order of content sections. Every article contains required elements. Articles will also contain conditional elements and optional elements outlined in content design or creation. See the guidelines below for more details.



Titles

Titles fully describe what a page is about, and what someone will learn by reading it.

Titles can be challenging. Use these general guidelines to help create clear, helpful, and descriptive titles. The guidelines for each content type in this article provide more specific title rules.

Titles for all content types

- Titles clearly describe what a page is about. They are descriptive and specific.
  - Use: "Browsing actions in the workflow editor"
  - Use: "Example of configuring a codespace"
  - Avoid: "Using the workflow editor sidebar"
  - Avoid: "Example"
- Titles have hard limits for length to keep them easy to understand (and easier to render on the site):
  - Category titles: 67 characters and `shortTitle` 26 characters
  - Map topic titles: 63 characters and `shortTitle` 29 characters
  - Article titles: 80 characters, 60 if possible, and `shortTitle` 30 characters, ideally 20-25 characters
- Titles use sentence case.
  - Use: "Changing a commit message"
  - Avoid: "Changing A Commit Message"
- Titles are consistent across a content type. See specific guidelines for each content type.
- Titles are general enough to scale with product changes, reflect all of the content within the article, or include content on multiple products.
  - Use: "GitHub's billing plans"
  - Avoid: "Billing plans for user and organization accounts"
- Titles use consistent terminology.

    - Develop and follow patterns within a category or on similar subjects.

- Titles use terminology from the product itself.
- Write the title and the intro at the same time.
    - Use the intro to develop the ideas presented in the title.
    - See guidance on intros for more information.
- If it's hard to come up with a title, consider the content type. Sometimes trouble choosing a title indicates that another content type would fit better.
- Think about how the title will appear in search results for multiple products.
    - What specific words do we need to include in the title or intro so that folks don't mistake it for content about a different product?
- Think about how the title will look in production.

Product callout

Use the product callout when a feature is available in specific products only and that availability cannot be conveyed by versioning alone. For example, if a feature is available for GHEC, GHES, and GHAE, you can version content about the feature for GHEC, GHES, and GHAE only. If a feature is available for Pro, Team, GHEC, GHES, and GHAE (but not Free), use a product callout to convey that availability.

All product callouts are stored as reusables in `gated-features` and added in YAML frontmatter for relevant articles.

How to write a product callout

- Product callouts follow a strict format, clearly identifying the feature and which products it's available in.
- Product callouts also include a link to "GitHub's products" and occasionally to another relevant article.
- Examples:
    - [Feature name] is available in [product(s)]. For more information, see "GitHub's products."
    - [Feature name] is available in public repositories with [free product(s)], and in public and private repositories with [paid products]. For more information, see "GitHub's products."

Examples of articles with product callouts

Check the source files and `gated-features` to see how source content is written.

- Managing a branch protection rule

Intro

The top of every page has an intro that provides context and sets expectations, allowing readers to quickly decide if the page is relevant to them. Intros also are displayed in search results to provide contextual information to help readers choose a result.

How to write an intro

- Article intros are one to two sentences long.

- Map topic and category intros are one sentence long.
- API reference intros are one sentence long.
    - The intro for an API page should define the feature so that someone knows whether the feature meets their needs without reading the entire article.
- Intros contain a high-level summary of the page’s content, developing the idea presented in a title with more detail.
    - Use approachable synonyms of words in the page’s title to help readers understand the article’s purpose differently. Avoid repeating words from the title when possible.
- Intros are relatively evergreen and high-level, so they can scale with future changes to the content on the page without needing to be frequently updated.
- For searchability, include keywords on the page's subject in the intro.
- When a term in the intro has an acronym we’ll use elsewhere in the article, indicate the acronym.
- Intros generally don't contain permissions for any tasks contained within the article.

Permissions statements

Every procedure includes a permissions statement explaining the role required to take the action described in the procedure, which helps people understand whether they'll be able to complete the task.

Occasionally, it's relevant to mention required permissions in conceptual content, especially in standalone conceptual articles. Make sure to also include a permissions statement in related procedures (or write a longer article combining all of the content).

How to write a permissions statement

- When a single set of permissions applies to all procedures in an article, use the permissions frontmatter.
- When an article contains multiple procedures and different permissions apply, include a separate permissions statement under each relevant header, before each procedure.
- Don't include permissions in an article’s intro.
- Roles exist at different levels. Refer only to the role at the same level as the action. For example, you need admin access to a repository (repository-level role) to configure protected branches. You can get admin access to a repository by being an organization owner (organization-level role), but the repository-level role is what actually governs your ability to take the action, so that is the only role that should be mentioned in the permissions statement.
- Language to use in a permissions statement:
    - [ACCOUNT ROLE] can [ACTION].
    - People with [FEATURE ROLE] access for a [FEATURE] can [ACTION].
    - AVOID: [ACCOUNT ROLE] and people with [FEATURE ROLE] access for a [FEATURE] can [ACTION].

Examples of permissions statements

- Article with single permissions statement for multiple procedures: Enforcing repository management policies in your enterprise

Tool switcher

Some articles have content that varies depending on what tool someone uses to complete a task, such as the GitHub CLI or GitHub Desktop. For most content, the same conceptual or procedural information will be accurate for multiple tools. However, if the only way to make information clear and accurate is by distinguishing content by tool, use the tool switcher. Do not use the tool switcher just to show examples in different languages. Only use the tool switcher if the tasks or concepts change based on what tool someone uses. For more information, see "Creating tool switchers in articles".

Table of contents

Tables of contents are automatically generated. For more information see "Autogenerated mini-TOCs."

Conceptual content

Conceptual content helps people understand or learn about a topic. For more information, see "Conceptual content type" in the content model.

Referential content

Referential content provides structured information related to actively using a product or feature. For more information, see "Referential content type" in the content model.

Prerequisites

Prerequisites are information that people need to know before proceeding with a procedure, so that they can prepare everything they need before starting the task.

How to write prerequisites

- Write prerequisites immediately before a procedure's numbered steps.
- You can use a list, a sentence, or a paragraph to explain prerequisites.
- You can also use a separate prerequisites section when:
  - The prerequisite information is very important and should not be missed.
  - There's more than one prerequisite.
- To repeat or highlight important information about data loss or destructive actions, you may also use a warning or danger callout to share a prerequisite.

Title guidelines for prerequisites

- When using a separate section, use a header called `Prerequisites`

Examples of articles with prerequisites sections

- Installing GitHub Enterprise Server on AWS
- Enabling subdomain isolation

Procedural content

Procedural content helps people complete tasks. For more information, see "Procedural

content type" in the content model.

Troubleshooting content

Troubleshooting content helps people avoid or work through errors. For more information, see "Troubleshooting content type" in the content model.

Further reading

Further reading sections highlight additional targeted articles that aren't already included within the article's content or sidebar. Use further reading sections sparingly when they provide real value.

How to write a further reading section

- Use a bulleted list.
- Use further reading sections sparingly and when they provide high value - see style guide for guidelines on linking.

Title and format for further reading sections

```
### Further reading
- "[Article title](article-URL)"
- [External resource title](external-resource-URL) in External Resource Name
```