Usage

Configuration

Configure the `doom` documentation tool

Convention

Based on the principle of "convention over documents to automatically generate the l

Internationalization

Using Internationalized Text in Reusable C

Permission Description Document

Markdown

MDX

Using MDX ena

API Docum

Referencing

Deployment

After the documentation project development is completed, we
platform

Build and Preview

Multi-version Build

Merged Directory Structure

Dynamic Mount Configuration File

Configuration

TOC

Configuration File

In most cases, we only need to use a static `yaml` configuration file, supporting `doom.config.yaml` or `doom.config.yml`. For complex scenarios, such as requiring dynamic configuration or custom `rspress` plugins, `js/ts` configuration files can be used, supporting multiple file formats including `.js/.ts/.mjs/.mts/.cjs/.cts`.

For `js/ts` configuration files, we need to export the configuration, which can be combined with the `defineConfig` function exported from `@alauda/doom/config` to provide type assistance:

```
import { defineConfig } from '@alauda/doom/config'

export default defineConfig({})
```

Basic Configuration

- `lang`: Default documentation language. To accommodate most projects, we support both Chinese and English documents by default. The default language is `en`. If the current documentation project does not require multilingual support, this can be set to `null` or `undefined`.
- `title`: Documentation title, displayed on the browser tab.
- `logo`: Logo at the top left of the documentation, supports image URLs or file paths. Absolute paths refer to files under the `public` directory, relative paths refer to files relative to the current tool directory. By default, the built-in Alauda logo from the `doom` package is used.
- `logoText`: Documentation title, displayed next to the logo at the top left.
- `icon`: Documentation favicon, defaults to the same as `logo`.
- `base`: Base path for the documentation, used when deploying to a non-root path, such as `product-docs`. Defaults to `/`.
- `outDir`: Directory for build outputs, defaults to `dist/{base}/{version}`. If specified, it changes to `dist/{outDir}/{version}`, where `version` is optional. See Multi-version Build for reference.

API Documentation Configuration

```yaml
api:
  # CRD definition file paths, relative to the directory of doom.config.*, supports glob matching, json/yaml files
  crds:
    - docs/shared/crds/*.yaml
  # OpenAPI definition file paths, relative to the directory of doom.config.*, supports glob matching, json/yaml files
  openapis:
    - docs/shared/openapis/*.json
  # When rendering OpenAPI related resource definitions, they are inlined by default. To extract related resource definitions into separate files, configure the following options.
  # Reference https://doom.alauda.cn/apis/references/CodeQuality.html#v1alpha1.CodeQualitySpec
  references:
    v1alpha1.CodeQualityBranch: /apis/references/CodeQualityBranch#v1alpha1.CodeQualityBranch
  # Optional, API documentation path prefix. If the current business uses gateway or other proxy services, this can be configured.
  pathPrefix: /apis
```

Refer to API Documentation for writing documentation.

Permission Documentation Configuration

```yaml
# The following resource file paths are relative to the directory of doom.config.*, support glob matching, json/yaml files
permission:
  functionresources:
    # `kubectl get functionresources`
    - docs/shared/functionresources/*.yaml
  roletemplates:
    # `kubectl get roletemplates -l auth.cpaas.io/roletemplate.official=true`
    - docs/shared/roletemplates/*.yaml
```

Refer to Permission Documentation for writing documentation.

Reference Documentation Configuration

```yaml
reference:
  - repo: alauda-public/product-doc-guide # Optional, repository address of the referenced documentation. If not filled, the current documentation repository address is used by default.
    branch: # [string] Optional, branch of the referenced documentation repository
    publicBase: # [string] Optional, when using a remote repository, the absolute path where static resources corresponding to /images/xx.png are located. Defaults to docs/public
    sources:
      - name: anchor # Name of the referenced documentation, used for referencing in documents, globally unique
        path: docs/index.mdx#介绍 # Path of the referenced documentation, supports anchor positioning. For remote repositories, relative to the root of the repository; for local, relative to the directory of doom.config.*
        ignoreHeading: # [boolean] Optional, whether to ignore the heading. If true, the anchor heading will not be displayed in the referenced document.
        processors: # Optional, processors for referenced document content
          - type: ejsTemplate
            data: # ejs template parameters, accessed via `<%= data.xx %>`
        frontmatterMode: merge # Optional, frontmatter handling mode for referenced documents. Defaults to ignore. Options: ignore/merge/replace/remove
```

`frontmatterMode`

- `ignore`: Ignore the frontmatter of the referenced document, keep using the current document's frontmatter.
- `merge`: Merge the frontmatter of the referenced document. If keys conflict, the referenced document's values override the current document's.

- `replace`: Replace the current document's frontmatter with that of the referenced document.
- `remove`: Remove the current document's frontmatter.

Refer to Reference Documentation for writing documentation.

Release Notes Configuration

```
releaseNotes:
  queryTemplates:
    fixed: # JQL statements that can include ejs templates
    unfixed:
```

```
release-notes.md
```

```
<!-- release-notes-for-bugs?template=fixed&project=DevOps -->
```

```
release-notes.mdx
```

```
{/* release-notes-for-bugs?template=fixed&project=DevOps */}
```

Taking the above `template=fixed&project=DevOps` as an example, `fixed` is the template name defined in `queryTemplates`. The remaining `query` parameter `project=DevOps` is passed as `ejs` ↗ template parameters to the `fixed` template, which after processing is used to form a jira `jql` ↗ request to `https://jira.alauda.cn/rest/api/2/search?jql=<jql>`. This API requires authentication, and the environment variables `JIRA_USERNAME` and `JIRA_PASSWORD` must be provided to preview the effect.

Sidebar Configuration

```yaml
sidebar:
  collapsed: false # Optional, whether the sidebar is collapsed by defaul
t. Defaults to collapsed. When documentation content is small, consider s
etting to false.
```

Internal Documentation Routes Configuration

```yaml
internalRoutes: # Optional, supports glob matching, relative to the docs
directory. When the CLI option `-i, --ignore` is enabled, matched routes/
files will be ignored.
  - '*/internal/**'
```

Only Include Documentation Routes Configuration

```yaml
onlyIncludeRoutes: # Optional, supports glob matching, relative to the do
cs directory. When the CLI option `-i, --ignore` is enabled, only routes/
files under this configuration will be enabled. Can be combined with `int
ernalRoutes` to further exclude some routes.
  - '*/internal/**'
internalRoutes:
  - '*/internal/overview.mdx'
```

Syntax Highlighting Plugin Configuration

```yaml
shiki:
  theme: # optional, https://shiki.style/themes
  langs: # optional, https://shiki.style/languages
  transformers: # optional, only available in js/ts config, https://shik
i.style/guide/transformers
```

> **WARNING**
>
> Unconfigured languages will trigger warnings in the command line and fall back to rendering as `plaintext`.

`sites.yaml` Configuration

The `sites.yaml` configuration file is used to configure subsite information associated with the current documentation site. This information is used by External Site Components and when building single-version documentation.

```yaml
- name: connectors # Globally unique name
  base: /devops-connectors # Base path for site access
  version: v1.1 # Version used for ExternalSite/ExternalSiteLink redirect
ion when building multi-version sites

  displayName: # Site display name. If not filled or language not matche
d, defaults to `name`
    en: DevOps Connectors
    zh: DevOps 连接器

  # The following properties are used to pull images when building the en
tire site. If not filled, this will be ignored during final packaging.
  # Usually required for subsite references, not required for parent site
references.
  repo: https://github.com/AlaudaDevops/connectors-operator # Site reposi
tory address. For internal gitlab repositories, related slugs like `alaud
a/product-docs` can be used directly.
  image: devops/connectors-docs # Site build image, used to pull images w
hen building the entire site.
```

Translation Configuration

```
translate:
  # System prompt, ejs template. Parameters passed in include `sourceLang
`, `targetLang`, `userPrompt`, `additionalPrompts`, `terms`, `titleTransl
ationPrompt`.
  # Among them, `sourceLang` and `targetLang` are the strings `中文` and `
英文`,
  #     `userPrompt` is the global user configuration below, which may be
empty
  #     `additionalPrompts` is the `additionalPrompts` configuration in t
he document's `frontmatter.i18n`, which may be empty
  #     `terms` and `titleTranslationPrompt` are dynamically generated pr
ompts based on terminology and title tables contained in the document, us
ed to guide AI translation according to terminology and title mappings, w
hich may be empty
  # The default system prompt is as follows and can be modified according
to actual needs
  systemPrompt: |
You are a professional technical documentation engineer, skilled in writi
ng high-quality technical documentation in <%= targetLang %>. Please accu
rately translate the following text from <%= sourceLang %> to <%= targetL
ang %>, maintaining the style consistent with technical documentation in
<%= sourceLang %>.

## Baseline Requirements
- Sentences should be fluent and conform to the expression habits of the
<%= targetLang %> language.
- Input format is MDX; output format must also retain the original MDX fo
rmat. Do not translate the names of jsx components such as <Overview />,
and do not wrap output in unnecessary code blocks.
- **CRITICAL**: Do not translate or modify ANY link content in the docume
nt. This includes:
  - URLs in markdown links: [text](URL) - keep URL exactly as is
  - Reference-style links: [text][ref] and [ref]: URL - keep both ref and
URL unchanged
  - Inline URLs: https://example.com - keep completely unchanged
  - Image links: ![alt](src) - keep src unchanged, but alt text can be tr
anslated
  - Anchor links: [text](#anchor) - keep #anchor unchanged
  - Any href attributes in HTML tags - keep unchanged
- Do not translate professional technical terms and proper nouns, includi
ng but not limited to: Kubernetes, Docker, CLI, API, REST, GraphQL, JSON,
YAML, Git, GitHub, GitLab, AWS, Azure, GCP, Linux, Windows, macOS, Node.j
s, React, Vue, Angular, TypeScript, JavaScript, Python, Java, Go, Rust, e
```

```
tc. Keep these terms in their original form.
- The title field and description field in frontmatter should be translat
ed, other frontmatter fields should retain and do not translate.
- Content within MDX components needs to be translated, whereas MDX compo
nent names and parameter keys do not.
- Do not modify or translate any placeholders in the format of __ANCHOR_N
__ (where N is a number). These placeholders must be kept exactly as they
appear in the source text.
- Keep original escape characters like backslash, angle brackets, etc. un
changed during translation.
- Do not add any escape characters to special characters like [], (), {},
etc. unless they were explicitly present in the source text. For example:
  - If source has "Architecture [Optional]", keep it as "Architecture [Op
tional]" (not "Architecture \\[Optional]")
  - If source has "Function (param)", keep it as "Function (param)" (not
"Function \\(param)")
  - Only add escape characters if they were present in the original text
- Preserve and do not translate the following comments, nor modify their
content:
  - {/* release-notes-for-bugs */}
  - <!-- release-notes-for-bugs -->
- Remove and do not retain the following comments:
  - {/* reference-start */}
  - {/* reference-end */}
  - <!-- reference-start -->
  - <!-- reference-end -->
- Ensure the original Markdown format remains intact during translation,
such as frontmatter, code blocks, lists, tables, etc.
- Do not translate the content of the code block.
<% if (titleTranslationPrompt) { %>
<%- titleTranslationPrompt %>
<% } %>
<% if (terms) { %>
<%- terms %>
<% } %>

<% if (userPrompt || additionalPrompts) { %>
## Additional Requirements
These are additional requirements for the translation. They should be met
along with the baseline requirements, and in case of any conflict, the ba
seline requirements should take precedence.

The text for translation is provided below, within triple quotes:
"""
```

```
<% if (userPrompt) { %>
<%- userPrompt %>
<% } %>

<% if (additionalPrompts) { %>
<%- additionalPrompts %>
<% } %>
"""
<% } %>
```

Editing Documentation in Code Repository

```
editRepoBaseUrl: alauda/doom/tree/main/docs # The https://github.com/ prefix can be omitted. Only effective when the CLI flag `-R, --edit-repo` is enabled.
```

Documentation Export Configuration

```
export:
  - name: Concepts # Optional, globally unique PDF name, defaults to the documentation title
    scope: '*/concepts' # Required, string or array, document scope, supports glob matching, relative to the docs directory
```

Documentation Lint Configuration

```
lint:
  cspellOptions: # Optional, cspell configuration options, refer to https://github.com/streetsidesoftware/cspell/tree/main/packages/cspell-eslint-plugin#options
```

Algolia Search Configuration

```yaml
algolia: # Optional, Algolia search configuration, only effective when th
e CLI flag `-a, --algolia` is enabled
  appId: # Algolia Application ID
  apiKey: # Algolia API Key
  indexName: # Algolia index name
```

Please use `public/robots.txt` for Algolia crawler verification.

Info

Due to the current architecture limitations of `rspress`, using Algolia search requires implementing via a custom theme↗. To unify usage of related theme features, we provide the `@alauda/doom/theme` theme entry. Please add the following theme configuration file to enable:

theme/index.ts

```ts
export * from '@alauda/doom/theme'
```

Sitemap Configuration

```yaml
siteUrl: https://docs.alauda.cn # Optional, site URL used to generate sit
emap, only effective when the CLI flag `-S, --site-url` is enabled
```

Convention

TOC

Directory Structure

The left sidebar is automatically generated based on the file directory structure, where the `index` file in the first-level directory acts as the document's homepage and will display as the first item in the left navigation. Subfolders can use `index.md` or `index.mdx` and define the first-level title to set the grouping title for the left sidebar. Other sub-documents will be automatically merged into the current group, and nested subfolders will follow the same rules.

```
├── index.md
├── start.mdx
└── usage
    ├── index.mdx
    └── convention.md
```

We also agree that:

1. The `public` directory is used to store static resources such as images, videos, etc.

2. The `public/_remotes` directory is used to store static resources associated with remote reference documents. Please do not directly rely on resources from this directory; you may add `*/public/_remotes` to `.gitignore` to prevent these from being committed to the code repository.
3. The `shared` directory is for storing common components, reusable documents, etc., and will not automatically generate document data.

Metadata

At the beginning of the document, you can define the document's metadata such as title, description, author, category, etc., through the `frontmatter`.

```
---
title: Title
description: Description
author: Author
category: Category
---
```

In the body of the document, when using `.mdx` files, you can access these metadata through `frontmatter` as described in MDX.

Sorting

Other documents, except for `index.md` or `index.mdx`, will be sorted by default according to their file names. You can customize the `weight` value in the `frontmatter` to adjust the order of documents in the left sidebar (the smaller the `weight` value, the higher the priority in sorting).

```
---
weight: 1
---
```

> **Warning**
>
> Note: Currently, changes to the left navigation configuration require a service restart to take effect, and it is usually not necessary to pay too much attention during development.

Preview

Sometimes, we do not need to display special content on the group homepage. In this case, you can use `index.mdx` file and the `Overview` component to display the list of documents in the current group. This will showcase the titles, descriptions, and secondary title information of the grouped list file.

```
# Usage

<Overview />
```

You can refer to Usage for the effect.

Markdown

In addition to the standard gfm syntax, Doom includes some additional Markdown extension features.

TOC

Callouts

Source code annotation component

> **Note**
>
> 1. Please use inline code comments according to the actual language, such as `;`, `%`, `#`, `//`, `/** */`, `--`, and `<!-- -->`, etc.
> 2. If you want to treat them as code comments, use `[\!code callout]` for escaping.
> 3. Sometimes, `:::callouts` may display abnormally due to nested indentation; you can use `<div class="doom-callouts">` or the `<Callouts>` component instead.

```sh
Memory overhead per virtual machine ≈ (1.002 × requested memory) \
              + 218 MiB \  # [!code callout]
              + 8 MiB × (number of vCPUs) \  # [!code callout]
              + 16 MiB × (number of graphics devices) \  # [!code callou
t]
              + (additional memory overhead) # [!code callout]
```

:::callouts

1. Required for the processes that run in the `virt-launcher` pod.
2. Number of virtual CPUs requested by the virtual machine.
3. Number of virtual graphics cards requested by the virtual machine.
4. Additional memory overhead:
   - If your environment includes a Single Root I/O Virtualization (SR-IO
V) network device or a Graphics Processing Unit (GPU), allocate 1 GiB add
itional memory overhead for each device.
   - If Secure Encrypted Virtualization (SEV) is enabled, add 256 MiB.
   - If Trusted Platform Module (TPM) is enabled, add 53 MiB.

:::

```sh
Memory overhead per virtual machine ≈ (1.002 × requested memory) \
              + 218 MiB \   1
              + 8 MiB × (number of vCPUs) \   2
              + 16 MiB × (number of graphics devices) \   3
              + (additional memory overhead)   4
```

1. Required for the processes that run in the `virt-launcher` pod.
2. Number of virtual CPUs requested by the virtual machine.
3. Number of virtual graphics cards requested by the virtual machine.
4. Additional memory overhead:

   - If your environment includes a Single Root I/O Virtualization (SR-IOV) network device or a Graphics Processing Unit (GPU), allocate 1 GiB additional memory overhead for each device.
   - If Secure Encrypted Virtualization (SEV) is enabled, add 256 MiB.

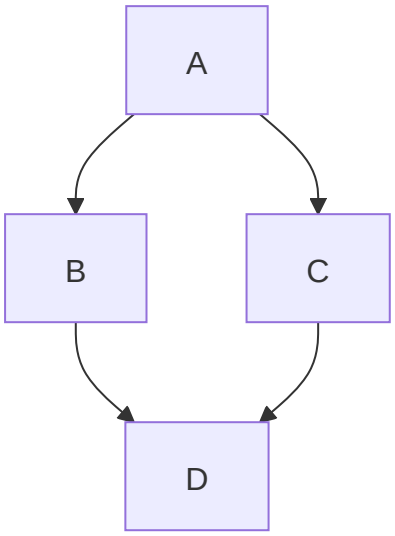
- If Trusted Platform Module (TPM) is enabled, add 53 MiB.

For more source code transformation features, please refer to Shiki Transformers↗.

Mermaid↗

Chart drawing tool

```
```mermaid
graph TD;
    A-->B;
    A-->C;
    B-->D;
    C-->D;
```
```



With Markdown Preview Mermaid↗, you can preview in real-time in VSCode.

MDX

MDX↗ is an extension syntax of Markdown that allows the use of JSX syntax within Markdown. For usage, refer to rspress MDX↗.

TOC

rspress Components

Most of the built-in components ↗ provided by the `rspress` theme have been adjusted to global components, which can be used directly in `.mdx` files without import, including:

- `Badge`
- `Card`
- `LinkCard`
- `PackageManagerTabs`
- `Steps`
- `Tab/Tabs`
- `Toc`

Other less commonly used components can be imported from `rspress/theme`, for example:

preview.mdx

```
import { SourceCode } from '@rspress/core/theme'

<SourceCode href="/" />
```

doom Components

`doom` provides some global components to assist in documentation writing, which can be used directly without import. Currently, these include:

`Overview`

Document overview component used to display the document directory

`Directive`

Sometimes, due to nested indentation, the custom container ↗ syntax may fail. You can use the `Directive` component as a replacement.

```
- The directory structure of multilingual documents (`doc/en`) needs to b
e exactly the same as the documents under the `doc/zh` directory to ensur
e that the links in multilingual documents are identical except for the l
anguage identifier.

  <Directive type="danger" title="Note">
    If you are using automated translation tools, you do not need to worr
y about
    this issue. The automated translation tools will automatically genera
te the
    target language document directory structure based on `doc/zh`.
  </Directive>
```

- The directory structure of multilingual documents (`doc/en`) needs to be exactly the same as the documents under the `doc/zh` directory to ensure that the links in multilingual documents are identical except for the language identifier.

  > **Note**
  >
  > If you are using automated translation tools, you do not need to worry about this issue. The automated translation tools will automatically generate the target language document directory structure based on `doc/zh`.

`ExternalSite`

Component for referencing external sites

```
<ExternalSite name="connectors" />
```

> **Note**
>
> Because DevOps Connectors releases on a different cadence from Alauda Container Platform, the DevOps Connectors documentation is now available as a separate documentation set at DevOps Connectors↗.

`ExternalSiteLink`

Component for referencing external site links

```
<ExternalSiteLink name="connectors" href="link.mdx#hash" children="Conten
t" />
```

Content ↗

TIP

In mdx, `<ExternalSiteLink name="connectors" href="link" children="Content" />` has a different meaning from the following content:

```
<ExternalSiteLink name="connectors" href="link">
  Content {/* will be rendered inside a `p` element */}
</ExternalSiteLink>
```

If you do not want the text to be rendered inside a `p` element, you can pass it using the `children` attribute as shown in the example above.

`AcpApisOverview` and `ExternalApisOverview`

Components for referencing external site API overviews

```
<AcpApisOverview />
{/* same as following */}
<ExternalApisOverview name="acp" />

<ExternalApisOverview name="connectors" />
```

Note

For the introduction to the usage methods of ACP APIs, please refer to ACP APIs Guide ↗.

> **Note**
>
> For the introduction to the usage methods of DevOps Connectors APIs, please refer to DevOps Connectors APIs Guide ↗.

Term

Term component, plain text, dynamically mounted and injected

```
<Term name="company" textCase="capitalize" />
<Term name="product" textCase="lower" />
<Term name="productShort" textCase="upper" />
<Term name="alaudaCloudLink" />
```

Alauda alauda container platform ACP Alauda Cloud ↗

`props`

- `name`: Built-in term name, refer to dynamic mount configuration file
- `textCase`: Text case transformation, optional values are `lower`, `upper`, `capitalize`

`TermsTable`

Built-in term list display component

```
<TermsTable />
```

Name	Chinese	Chinese Bad Cases	English	English Bad Cases	Description
company	灵雀云	-	Alauda	-	公司品牌

Name	Chinese	Chinese Bad Cases	English	English Bad Cases	Description
product	灵雀云容器平台	-	Alauda Container Platform	-	产品品牌
productShort	ACP	-	ACP	-	产品品牌简称
alaudaCloudLink	Alauda Cloud ↗	-	Alauda Cloud ↗	-	-

`props`

- `terms`: `NormalizedTermItem[]`, optional, custom term list for convenient reuse when rendering custom terms in internal documentation

`JsonViewer`

```
<JsonViewer value={{ key: 'value' }} />
```

yaml json

```yaml
key: value
```

Custom Component Reuse

According to the convention, we can extract reusable content into the `shared` directory, then import it where needed, for example:

```
import CommonContent from './shared/CommonContent.mdx'

<CommonContent />
```

If you need to use more runtime↗ related APIs, you can implement components with `.jsx/.tsx` and then import and use them in `.mdx` files.

```
// shared/CommonContent.tsx
export const CommonContent = () => {
  const { page } = usePageData()
  return <div>{page.title}</div>
}

// showcase/content.mdx
import { CommonContent } from './shared/CommonContent'
;<CommonContent />
```

WARNING

Note: Currently, components exported from `.mdx` do not support passing `props`. Refer to this issue↗. Therefore, for scenarios requiring `props` passing, please develop using `.jsx/.tsx` components.

Internationalization

Most of the internal documentation of `alauda` is bilingual in Chinese and English. Therefore, we by default support using `en` / `zh` subfolders to store documents in different languages. It is recommended to also store static resources under `en` / `zh` subfolders in the `public` directory, which facilitates the management of both documentation content and static resources.

TOC

`i18n.json`

For reusable components, if you need to support both Chinese and English within the same component, you first need to create an `i18n.json` file under the `docs` directory. Then, in the component, use `useI18n` to get the text for the current language, for example:

docs/i18n.json

```
{
  "title": {
    "zh": "标题",
    "en": "Title"
  },
  "description": {
    "zh": "描述",
    "en": "description"
  }
}
```

.ts/.tsx

```
import { useI18n } from '@rspress/core/runtime'

export const CommonContent = () => {
  const t = useI18n()
  return <h1>{t('title')}</h1>
}
```

.mdx

```
import { useI18n } from '@rspress/core/runtime'

# {useI18n()('title')}

{useI18n()('description')}
```

API Documentation

According to actual business needs, we generally divide APIs into three types: standard K8S API, advanced API, and CRD (Custom Resource Definition). Therefore, the directory structure is usually organized as follows:

```
├── apis
│   ├── advanced_apis # Advanced APIs
│   ├── crds # CRDs
│   ├── kubernetes_apis # K8S APIs
│   └── references # Common references
```

TOC

K8S API

kubernetes_apis/workload/daemonset.mdx

```
# DaemonSet [apps/v1]

<K8sAPI
  name="io.k8s.api.apps.v1.DaemonSet"
  pathPrefix="/kubernetes/{cluster}"
/>
```

Refer to DaemonSet.

crds/ArtifactCleanupRun.mdx

```
# ArtifactCleanupRun

<K8sAPI name="artifactcleanupruns.artifacts.katanomi.dev" />
```

Refer to ArtifactCleanupRun.

`props`

- `name`: Reference name under OpenAPI schema `definitions` (v2) or `components/schemas` (v3), or CRD `metadata.name`
- `namespaced`: Indicates whether the resource is namespace-scoped; defaults to `true`, meaning the API Endpoints include the namespace path parameter `namespaces/{namespace}`
- `pathPrefix`: Can be used to override the global configuration `api.pathPrefix`
- `filepath`: Similar to specifying openapi path, used to specify a particular openapi or CRD file
- `apiGroup`: Optional, specifies the API group; openapi will try to read the referenced `x-kubernetes-group-version-kind`, same below
- `apiVersion`: Optional, specifies the API version; CRD will default to the first version in `spec.versions`

- `apiKind`: Optional, specifies the API resource kind

Advanced API

```
advanced_apis/codeQualityTaskSummary.mdx
```

```
# CodeQualityTaskSummary

<OpenAPIPath path="/plugins/v1alpha1/template/codeQuality/task/{task-id}/
summary" />
```

Refer to CodeQualityTaskSummary.

`props`

- `path`: Path under OpenAPI schema `paths`
- `pathPrefix`: Can be used to override the global configuration `api.pathPrefix`
- `openapiPath`: See specifying openapi path

CRD (deprecated)

> **WARNING**
>
> Please use the `K8sAPI` component instead of the `K8sCrd` component. The `K8sCrd` component will be removed in future versions.

```
crds/ArtifactCleanupRun-K8sCrd.mdx
```

```
# ArtifactCleanupRun - K8sCrd

<K8sCrd name="artifactcleanupruns.artifacts.katanomi.dev" />
```

Refer to ArtifactCleanupRun-K8sCrd.

`props`

- `name`: CRD `metadata.name`
- `crdPath`: Similar to specifying openapi path, used to specify a particular CRD file

Common References

```
references/CodeQuality.mdx
```

```
# CodeQuality

<OpenAPIRef schema="v1alpha1.CodeQuality" />
```

Refer to CodeQuality.

`props`

- `schema`: Reference name under OpenAPI schema `definitions` (v2) or `components/schemas` (v3)
- `openapiPath`: See specifying openapi path

Specifying openapi Path

For the `OpenAPIPath` and `OpenAPIRef` components, by default, all openapi definition files are searched until a match is found. If you need to specify a particular openapi file, you can use the `openapiPath` attribute:

```
<OpenAPIPath
  path="/plugins/v1alpha1/template/codeQuality/task/{task-id}/summary"
  openapiPath="shared/openapis/katanomi.json"
/>
```

Permission Description Document

```
<K8sPermissionTable functions={['devops-testplans', 'devops-testmodules']} />
```

TOC

`props`

- `functions`: `string[]` - Required. An array of `FunctionResource` resource names to be displayed.

Example

Function	Action	Platform Administrator	Platform auditors	Project Manager	Namespace Administrator
testplans `devops-testplans`	View	✓	✓	✓	✓
	Create	✓	×	✓	✓

Function	Action	Platform Administrator	Platform auditors	Project Manager	Namespace Administrator
	Update	✓	×	✓	✓
testmodules `devops-testmodules`	Delete	✓	×	✓	✓
	View	✓	✓	✓	✓
	Create	✓	×	✓	✓
	Update	✓	×	✓	✓
	Delete	✓	×	✓	✓

Referencing Documents

In Markdown files:

```
<!-- reference-start#name -->

<!-- reference-end -->
```

In MDX files:

```
{/* reference-start#name */}

{/* reference-end */}
```

The `name` above refers to the name of the referenced document. For more information, please refer to Document Reference Configuration. If the referenced document content uses static resources from a remote repository, the related static resources will be automatically stored locally in the `<root>/public/_remotes/<name>` directory.

Here is an example using `<!-- reference-start#ref -->`:

TOC

Document Reference Configuration

```yaml
reference:
  - repo: alauda-public/product-doc-guide # Optional, repository address for the referenced document. If not provided, the current document repository address will be used by default.
    branch: # [string] Optional, branch of the referenced document repository.
    publicBase: # [string] Optional, the directory where static resources for remote repository located, corresponding to absolute paths like /images/xx.png. Default is docs/public.
    sources:
      - name: anchor # Name of the referenced document, used to reference within the document and must be globally unique.
        path: docs/index.mdx#introduction # Path to the referenced document, supports anchor targeting; for remote repositories, relative to the repository root directory, and for local, relative to the directory of doom.config.*.
        ignoreHeading: # [boolean] Optional, whether to ignore headings. If true, the anchor's title will not be displayed in the referenced document.
        processors: # Optional, processors for handling the content of the referenced document.
          - type: ejsTemplate
            data: # EJS template parameters, accessed via `<%= data.xx %>`.
        frontmatterMode: merge # Optional, mode for handling the frontmatter of the referenced document. Default is ignore. Possible values are ignore/merge/replace/remove.
```

`frontmatterMode`

- `ignore`: Ignores the frontmatter of the referenced document and retains the frontmatter of the current document.
- `merge`: Merges the frontmatter of the referenced document. If there are the same keys, the values from the referenced document will overwrite those in the current document.
- `replace`: Replaces the frontmatter of the current document with that of the referenced document.

- `remove`: Removes the frontmatter of the current document.

For writing documentation, refer to Document Reference.

Deployment

TOC

Build and Preview

Before deployment, we need to build the project for production and preview it locally to ensure the project runs correctly:

```
doom build # Build static assets
doom serve # Preview the build assets in production mode
```

Multi-version Build

By default, `doom build` outputs the build artifacts to the `dist` directory. If you need to build multiple versions of the documentation, you can specify the version number with the `-v` parameter, for example:

```
# Usually determined by branch name, e.g., release-4.0 corresponds to ver
sion 4.0
doom build -v 4.0 # Build version 4.0, output to dist/4.0, documentation
access path is {base}/4.0
doom build -v master # Build master version, output to dist/master, docum
entation access path is {base}/master
doom build -v {other} # Build other versions, output to dist/{other}, doc
umentation access path is {base}/{other}

# unversioned and unversioned-x.y are special version numbers used to bui
ld documentation without version prefix
doom build -v unversioned # Build documentation without version prefix, o
utput to dist/unversioned, documentation access path is {base}
doom build -v unversioned-4.0 # Build documentation without version prefi
x but showing version 4.0 in the navbar, output to dist/unversioned, docu
mentation access path is {base}
```

Merged Directory Structure

```
├── console-platform
│   ├── 4.0
│   ├── 4.1
│   ├── index.html
│   ├── overrides.yaml
│   └── versions.yaml
├── console-devops-docs
│   ├── 4.0
│   ├── 4.1
│   ├── index.html
│   ├── overrides.yaml
│   └── versions.yaml
├── console-tekton-docs
│   ├── 1.0
│   ├── 1.1
│   ├── index.html
│   ├── overrides.yaml
│   └── versions.yaml
```

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Redirecting...</title>
    <meta http-equiv="refresh" content="0; url=/console-docs/4.1" />
  </head>
  <body>
    <p>Redirecting to <a href="/console-docs/4.1">/console-docs/4.1</a></
p>
  </body>
</html>
```

Dynamic Mount Configuration File

overrides.yaml

```
# Documentation information, each document can mount to override default
configuration
title:
  en: Doom - Alauda
  zh: Doom - 灵雀云
logoText:
  en: Doom - Alauda
  zh: Doom - 灵雀云
```

versions.yaml

```
- '4.1'
- '4.0'
```