RichText Field

The RichText field type provides a rich text editor that supports Markdown content. It allows content editors to format text, add links, images, and other media, making it a versatile choice for creating rich content.

> **Note for Netlify/Decap CMS users**
>
> For backward compatibility with Netlify/Decap CMS, the Markdown field type remains available as an alias of the RichText field type. You can use either `richtext` or `markdown` as the `widget` value in your field configuration.

User Interface

Editor

A Lexical-based rich text editor, including headings, lists, links, images, code blocks, and more. It provides a user-friendly interface for writing and formatting content.

The built-in toolbar includes buttons for common formatting options, which can be customized using the `buttons` option. The editor also supports different modes, including a raw Markdown editing mode, which can be configured using the `modes` option. Additional editor components can be added to enhance the editing experience using the `editor_components` option.

Local/remote images can be pasted or dropped into the editor to insert them. Note: pasting multiple images is not supported in Firefox.

> **Breaking change from Netlify/Decap CMS**
>
> Remark plugins are not supported because Sveltia CMS uses the Lexical framework instead of Slate. The `CMS.registerRemarkPlugin` API method is a noop in Sveltia CMS.

Preview

A read-only view of the rich text content, rendered as HTML.

Data Type

A Markdown string. See the [Data Output] documentation for details on the Markdown syntax used by Lexical.

If the `required` option is set to `false` and the field is left empty, the value will be an empty string.

You need to parse the Markdown string using a Markdown parser in your framework to convert it to HTML for rendering on your website. Some frameworks have built-in support for Markdown, while others may require additional libraries. Please refer to your framework's documentation on how to handle Markdown content. See also the [how-to] for advice on handling line breaks in Markdown.

> **Future Plans**
>
> We plan to add support for HTML output in future releases. It will provide additional features specific to HTML content, including text alignment, link targets, and more.

Data Validation

- If the `required` option is set to `true`, the rich text content must not be an empty string.
- If the `pattern` option is provided, the rich text content must match the specified regular expression pattern.

Options

In addition to the [common field options], the Markdown field supports the following options:

Required Options

`widget`

- **Type**: `string`
- **Default**: `string`

Must be set to `richtext` .

Optional Options

> **Breaking changes from Netlify/Decap CMS**
>
> Sveltia CMS has changed the default value of the `sanitize_preview` option to `true` for improved security. In Netlify/Decap CMS, the default is `false` , which may expose users to XSS vulnerabilities.
>
> Also, Sveltia CMS does not support the deprecated camelCase `editorComponents` option. Use `editor_components` instead.

`default`

- **Type**: `string`
- **Default**: `""`

The default content for the field. The format should match the selected `format` option.

`minimal`

- **Type**: `boolean`
- **Default**: `false`

Whether to limit the editor height. When set to `true` , the editor height is reduced and a scrollbar appears when the content exceeds the height.

`modes`

- **Type**: `array`
- **Default**: `[rich_text, raw]`

The modes available in the editor. Possible values are `rich_text` and `raw` . The `raw` mode allows users to edit the raw Markdown text.

The following configurations are possible:

- Default modes: `[rich_text, raw]`
- Turn on raw mode by default: `[raw, rich_text]`
- Rich text only: `[rich_text]`
- Raw mode only: `[raw]`

If multiple modes are enabled, users can switch between them using a mode selector in the editor toolbar.

`buttons`

- **Type**: `array`
- **Default**: all available buttons (see below)

The button names to display in the editor toolbar.

The following `buttons` are available in the rich text editor toolbar:

- Inline formatting: `bold` , `italic` , `strikethrough` , `code` , `link`
- Block types: `heading-one` , `heading-two` , `heading-three` , `heading-four` , `heading-five` , `heading-six` , `bulleted-list` , `numbered-list` , `quote`

By default, all buttons are enabled. You can customize the toolbar by specifying the desired buttons in the `buttons` option.

> **Note for Netlify/Decap CMS users**
>
> Unlike Netlify/Decap CMS, all the block type buttons are available under the block type selector in Sveltia CMS. Users can select the block type from a dropdown menu rather than having separate buttons for each block type.

> **Future Plans**
>
> These buttons are disabled when `raw` mode is active. This behavior may be changed in future releases to allow certain buttons to function in `raw` mode as well.

`editor_components`

- **Type**: `array`
- **Default**: `[code-block, image]`

The editor component names to include in the rich text editor.

Editor components are custom blocks that can be inserted into the content. Sveltia CMS includes built-in components and also allows for custom components.

Sveltia CMS includes the following built-in editor components for the RichText field:

- `code-block` : Allows users to insert and format code blocks with syntax highlighting.
- `image` : Enables users to add images to their content, with support for uploading and selecting images from the media storage. The image can be linked or unlinked based on the `linked_images` option.

Both are enabled by default. You can disable them by omitting them from the `editor_components` option.

> **Note for Netlify/Decap CMS users**
>
> Unlike Netlify/Decap CMS, the `code-block` component in Sveltia CMS is implemented as a block type. Users can insert it using the block type selector rather than the insert button. Also, the `image` component is displayed as a separate button in the toolbar for easier access.

> **Future Plans**
>
> More built-in editor components may be added in future releases, such as `table` .

Developers can create [custom editor components] to extend the functionality of the rich text editor. Custom components can be registered globally in Sveltia CMS.

`linked_images`

- **Type**: `boolean`
- **Default**: `true`

Whether to allow linking images in the editor. When set to `true` , users can add links to images. When set to `false` , images will be inserted without links.

`sanitize_preview`

- **Type**: `boolean`
- **Default**: `true`

Whether to sanitize the preview content to prevent cross-site scripting (XSS) attacks. The sanitization process uses DOMPurify to remove potentially harmful HTML tags and attributes from the content before rendering the preview.

> **Security Risk**
>
> Setting the `sanitize_preview` option to `false` can expose your CMS to XSS vulnerabilities if untrusted users have access to the CMS, especially when using Open Authoring. Malicious users could inject harmful scripts into the content, which would then be executed in the browsers of anyone viewing the preview.
>
> We recommend keeping this option enabled unless disabling it fixes a broken preview and you fully trust all users of your CMS or you're the sole user.

Examples

Standard Markdown Field

This example shows a basic Markdown editor with default settings.

YAML TOML JSON JavaScript

yaml

```yaml
- name: body
  label: Body
  widget: richtext
```

Basic Markdown Field with Limited Buttons

This example shows a minimal Markdown editor with only bold, italic, and link buttons.

YAML TOML JSON JavaScript

yaml

```yaml
- name: content
  label: Content
  widget: richtext
  default: To get started, write your **Markdown** content here.
```

```
    minimal: true
    buttons: [bold, italic, link]
```

Disabling Code Block Component

This example shows how to disable the `code-block` editor component.

YAML TOML JSON JavaScript

yaml

```yaml
- name: content
  label: Content
  widget: richtext
  editor_components: [image]
```

Previous page
Relation

Next page
Select