

Markdown Extensions

VitePress comes with built in Markdown Extensions.

Header Anchors

Headers automatically get anchor links applied. Rendering of anchors can be configured using the `markdown.anchor` option.

Links

Both internal and external links gets special treatments.

Internal Links

Internal links are converted to router link for SPA navigation. Also, every `index.md` contained in each sub-directory will automatically be converted to `index.html`, with corresponding URL `/`.

For example, given the following directory structure:

```
.
├─ index.md
├─ foo
│   ├─ index.md
│   ├─ one.md
│   └─ two.md
└─ bar
    ├─ index.md
    ├─ three.md
    └─ four.md
```

And providing you are in `foo/one.md` :

```
[Home](/) <!-- sends the user to the root index.md -->
[foo](/foo/) <!-- sends the user to index.html of directory foo -->
[foo heading](./#heading) <!-- anchors user to a heading in the foo index file -->
[bar - three](./bar/three) <!-- you can omit extension -->
[bar - three](./bar/three.md) <!-- you can append .md -->
[bar - four](./bar/four.html) <!-- or you can append .html -->
```

Page Suffix

Pages and internal links get generated with the `.html` suffix by default.

External Links

Outbound links automatically get `target="_blank" rel="noopener noreferrer" :`

- vuejs.org
- [VitePress on GitHub](#)

Frontmatter

YAML frontmatter is supported out of the box:

```
---
title: Blogging Like a Hacker
lang: en-US
---
```

This data will be available to the rest of the page, along with all custom and theming components.

For more details, see [Frontmatter](#).

GitHub-Style Tables

Input

```
| Tables      | Are          | Cool |
| ----- | :-----: | -----: |
| col 3 is   | right-aligned | $1600 |
| col 2 is   | centered    | $12 |
| zebra stripes | are neat    | $1 |
```

Output

Tables	Are	Cool
col 3 is	right-aligned	\$1600
col 2 is	centered	\$12
zebra stripes	are neat	\$1

Emoji 🎉

Input

```
:tada: :100:
```

Output



A [list of all emojis](#) is available.

Table of Contents

Input

```
[[toc]]
```

Output

- [Header Anchors](#)
- [Links](#)
 - [Internal Links](#)
 - [Page Suffix](#)
 - [External Links](#)
- [Frontmatter](#)
- [GitHub-Style Tables](#)
- [Emoji 🌈](#)
- [Table of Contents](#)
- [Custom Containers](#)
 - [Default Title](#)
 - [Custom Title](#)
- [Syntax Highlighting in Code Blocks](#)
- [Line Highlighting in Code Blocks](#)
- [Line Numbers](#)
- [Import Code Snippets](#)
- [Advanced Configuration](#)

Rendering of the TOC can be configured using the `markdown.toc` option.

Custom Containers

Custom containers can be defined by their types, titles, and contents.

Default Title

Input

```
 ::: info
This is an info box.
 :::

 ::: tip
```

md

```
This is a tip.  
:::  
  
::: warning  
This is a warning.  
:::  
  
::: danger  
This is a dangerous warning.  
:::  
  
::: details  
This is a details block.  
:::
```

Output

INFO

This is an info box.

TIP

This is a tip.

WARNING

This is a dangerous warning.

DANGER

This is a dangerous warning.

► Details

Custom Title

You may set custom title by appending the text right after the "type" of the container.

Input

```
::: danger STOP
Danger zone, do not proceed
:::

::: details Click me to view the code
```js
console.log('Hello, VitePress!')
```
:::
```

Output

STOP

Danger zone, do not proceed

▶ Click me to view the code

Syntax Highlighting in Code Blocks

VitePress uses **Shiki** to highlight language syntax in Markdown code blocks, using coloured text. Shiki supports a wide variety of programming languages. All you need to do is append a valid language alias to the beginning backticks for the code block:

Input

```
```js
export default {
 name: 'MyComponent',
 // ...
}
```

```
```html
<ul>
  <li v-for="todo in todos" :key="todo.id">
    {{ todo.text }}
  </li>
</ul>
```

```
</li>
</ul>
...
```

Output

```
export default {
  name: 'MyComponent'
  // ...
}
```

js

```
<ul>
  <li v-for="todo in todos" :key="todo.id">
    {{ todo.text }}
  </li>
</ul>
```

html

A [list of valid languages](#) is available on Shiki's repository.

You may also customize syntax highlight theme in app config. Please see [markdown options](#) for more details.

Line Highlighting in Code Blocks

Input

```
```js{4}
export default {
 data () {
 return {
 msg: 'Highlighted!'
 }
 }
}
```
```

Output

```
export default {
  data () {
    return {
      msg: 'Highlighted!'
    }
  }
}
```

js

In addition to a single line, you can also specify multiple single lines, ranges, or both:

- Line ranges: for example `{5-8}`, `{3-10}`, `{10-17}`
- Multiple single lines: for example `{4,7,9}`
- Line ranges and single lines: for example `{4,7-13,16,23-27,40}`

Input

```
```${js}{1,4,6-7}
export default { // Highlighted
 data () {
 return {
 msg: `Highlighted!
 This line isn't highlighted,
 but this and the next 2 are.`
 motd: 'VitePress is awesome',
 lorem: 'ipsum'
 }
 }
}
```
```

Output

```
export default { // Highlighted
  data () {
    return {
      msg: `Highlighted!
      This line isn't highlighted,
      but this and the next 2 are.`
      motd: 'VitePress is awesome',
      lorem: 'ipsum',
    }
  }
}
```

js


```
}  
}
```

Line Numbers

You can enable line numbers for each code blocks via config:

```
export default {  
  markdown: {  
    lineNumbers: true  
  }  
}
```

js

Please see [markdown options](#) for more details.

Import Code Snippets

You can import code snippets from existing files via following syntax:

```
<<< @/filepath
```

md

It also supports [line highlighting](#):

```
<<< @/filepath{highlightLines}
```

md

Input

```
<<< @/snippets/snippet.js{2}
```

md

Code file

```
export default function () {  
  // ..  
}
```

js

Output

```
export default function () {  
  // ..  
}
```

js

TIP

The value of `@` corresponds to the source root. By default it's the VitePress project root, unless `srcDir` is configured.

You can also use a **VS Code region** to only include the corresponding part of the code file. You can provide a custom region name after a `#` following the filepath:

Input

```
<<< @/snippets/snippet-with-region.js#snippet{1}
```

md

Code file

```
// #region snippet  
function foo() {  
  // ..  
}  
// #endregion snippet  
  
export default foo
```

js

Output

```
function foo() {  
  // ..  
}
```

js

Advanced Configuration

VitePress uses [markdown-it](#) as the Markdown renderer. A lot of the extensions above are implemented via custom plugins. You can further customize the `markdown-it` instance using the `markdown` option in `.vitepress/config.js` :

```
const anchor = require('markdown-it-anchor')  
  
module.exports = {  
  markdown: {  
    // options for markdown-it-anchor  
    // https://github.com/valeriangalliat/markdown-it-anchor#permalinks  
    anchor: {  
      permalink: anchor.permalink.headerLink()  
    },  
  
    // options for markdown-it-toc-done-right  
    toc: { level: [1, 2] },  
  
    config: (md) => {  
      // use more markdown-it plugins!  
      md.use(require('markdown-it-xxx'))  
    }  
  }  
}
```

See full list of configurable properties in [Configs: App Configs](#).

Last updated: 2022/5/31 09:39:48