

You can override the HTML comment by using the option `omitTag` as explained above. Both, the tag and the actual comment in the Markdown file are case-insensitive.

Additional infos

- This plugin outputs a semantically correct table of contents. Sub-lists are rendered within the parent `` tag and not as a separate (empty) ``.
- Headlines can be in an arbitrary order. For example, h3, h2, h4. Please note that the jump from h2 to h4 causes a double-indentation, which is correct.

markdown-it-table-of-contents

A table of contents plugin for Markdown-it. Simple, customizable and with a default slugifier that matches that of [markdown-it-anchor](#) (>5.0.0).

Usage

CommonJS:

```
var MarkdownIt = require("markdown-it");  
var md = new MarkdownIt();
```

```
md.use(require("markdown-it-  
    anchor").default); // Optional,  
    but makes sense as you really  
    want to link to something, see  
    info about recommended plugins  
    below
```

```
md.use(require("markdown-it-table-of-  
    contents"));
```

ESM:

```
import MarkdownIt from 'markdown-it';
import MarkdownItTOC from 'markdown-it-
table-of-contents';
```

```
const md = new MarkdownIt();
md.use(MarkdownItTOC);
```

Then add `` where you want the table of contents to be added in your document.

Want to use a table of contents generator client-side in the browser? Try my ``webcomponent: [table-of-contents-element on GitHub](#). Advantage: can use more sophisticated query selector and better support for HTML customization.

Example markdown

This markdown:

Heading

Sub heading 1

Some nice text

Sub heading 2

Some even nicer text

... would render this HTML using the default options specified in “usage” above:

```
</p>
<h3 id="a-message-from-our-sponsors">A
message from our sponsors</h3>
<p>Ad</p>
<h2 id="hello">Hello world, I think you
should read this article</h2>
<p>Lorem ipsum</p>
<h2 id="what's-next?">What's next?</h2>
<p>Read this next...</p>
<h4 id="related">See related articles</h4>
<h4>
```

If you want to exclude single headlines, you can use a special HTML comment to omit the next headline from the TOC:

```
<!-- omit from toc -->
# Title
```

For this to work, the HTML comment must come right in the line before the headline you want to exclude. Furthermore, you need to allow HTML in MarkdownIT:

```
const md = new MarkdownIt({ html:
true });
```

See related articles {#related}

HTML output:

```
<h1 id="article">Article</h1>
<p>

  <ul>
    <li>
      <ul>
        <li><a href="#a-
message-from-our-sponsors">A
message from our sponsors</a></
li>
      </ul>
    </li>
    <li><a href="#hello">Hello
world, I think you should read
this article</a></li>
    <li><a href="#what's-
next%3F">What's next?</a>
      <ul>
        <li>
          <ul>
            <li><a
href="#related">See related
articles</a></li>
          </ul>
        </li>
      </ul>
    </li>
  </ul>
</p>
```

```
<h1 id="heading">Heading</h1>
```

```
<ul>
  <li><a href="#heading">Heading</a>
  <ul>
    <li><a href="#sub-
heading-1">Sub heading 1</a></
li>
    <li><a href="#sub-
heading-2">Sub heading 2</a></
li>
  </ul>
</li>
</ul>
```

```
<h2 id="sub-heading-1">Sub heading 1</
h2>
```

```
<p>Some nice text</p>
```

```
<h2 id="sub-heading-2">Sub heading 2</
h2>
```

```
<p>Some even nicer text</p>
```

Options

You may specify options when using the plugin. like so:

```
md.use(require("markdown-it-table-of-
contents"), options);
```

These options are available:

Name	Description Default
includelevel	Headings levels to use [1, 2] (2 for h2:s etc)
containerClass	The class for the container "table-of-contents"
slugify	A custom slugification function \s+/g, '-')
markerPattern	Regex pattern of the marker /^\[\[toc\]\]/im replaced with TOC HTML comment tag to exclude next <!-- omit from toc
omitTag	headline from TOC Type of list (ul for unordered, ol for ordered) A function for formatting md.renderInline(con
listType	
format	(see below)

4

Full example with unusual headline order

Of course, both plugins can be combined. markdown-it-anchor ignores headlines that already have an id attribute. Furthermore, markdown-it-table-of-contents can handle unusual heading orders. Consider the full example below:

```
var md = new MarkdownIt();
md.use(markdownItTOC, {
  "includelevel": [2,3,4]
});
md.use(require("markdown-it-attrs"));
md.use(require("markdown-it-anchor"));

# Article

### A message from our sponsors

Ad

## Hello world, I think you should read
this article {#hello}

Lorem ipsum

## What's next?

Read this next...
```

9

markdown-it-anchor

This plugin transforms all headlines in a markdown document so that the HTML code includes an id. It *slugifies* the headline:

```
## Hello world, I think you should read  
this article
```

Becomes

```
<h2 id="hello-world-i-think-you-should-  
read-this-article">Hello world,  
I think you should read this  
article</h2>
```

markdown-it-attrs

This plugin lets you attach custom attributes to your headlines. This is especially useful, if you have long headlines but want short anchors:

```
## Hello world, I think you should read  
this article {#hello}
```

Becomes

```
<h2 id="hello">Hello world, I think you  
should read this article</h2>
```

Name	Description	Default
containerHeaderHtml	Optional HTML string for container header	undefined
containerFooterHtml	Optional HTML string for container footer	undefined
transformLink	A function for transforming the TOC links	undefined
transformContainerOpen	A function for transforming the container opening tag	(see source code)
transformContainerClose	A function for transforming the container closing tag	(see source code)
getTokensText	A function for extracting text from tokens for titles	(see source code)

`format` is an optional function for changing how the headings are displayed in the TOC.

By default, TOC headings will be formatted using markdown-its internal MD formatting rules (i.e. it will be formatted using the same rules / extensions as other markdown in your document). You can override this behavior by specifying a custom format function. The function should accept two arguments:

1. content - The heading text, as a markdown string.
2. md - markdown-its internal markdown parser object. This should only be needed for advanced use cases.

```
function format(content, md) {  
  // manipulate the headings as you like  
  here.  
  return manipulatedHeadingString;  
}
```

transformLink is an optional function for transform the link as you like.

```
function transformLink(link) {  
  // transform the link as you like  
  here.  
  return transformedLink;  
}
```

transformContainerOpen and transformContainerClose can be used to replace the container element with one or several more like so:

```
md.use(markdownItTOC, {  
  transformContainerOpen: () => {
```

```
return '<button>Toggle</  
button><h3>Table of Contents</  
h3>';  
},  
transformContainerClose: () => {  
  return '';  
});
```

getTokensText is a function that can be used to change how text is extracted from tokens to support more ways how headlines are build. See source code for more information or the equivalent function in [markdown-it-anchor](#).

Recommended plugins

By default, markdown-it-table-of-contents collects all headings and renders a nested list. It uses the slugify() function to create anchor targets for the links in the list. However, the headlines in your markdown document are not touched by markdown-it-table-of-contents. You'd have a nice table of contents, but the links don't link to anything. That's why you need another plugin to generate ids (anchor link targets) for all of your headlines. There are two recommended plugins to achieve this: