A simple, beautiful, and embeddable JavaScript Markdown editor. Delightful editing for beginners and experts alike. Features built-in autosaving and spell checking.

simplemde.com

MIT license

9.5k stars **1.1k** forks **147** watching Activity

Public repository

master

Branches Tags

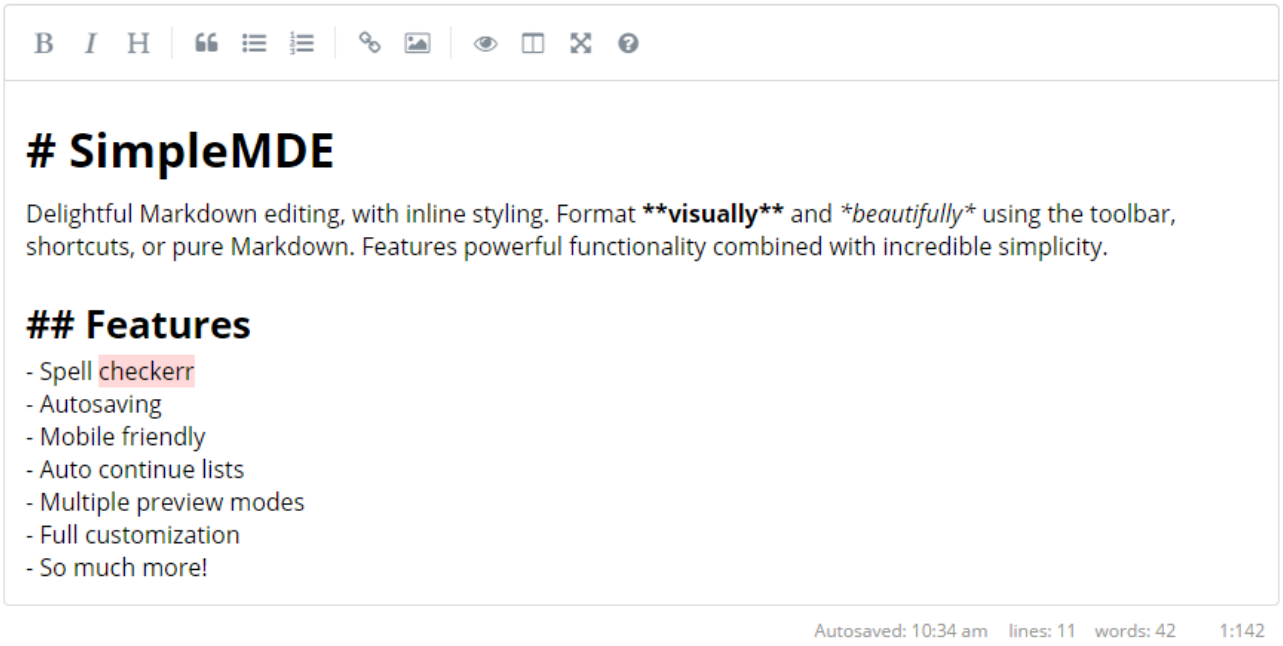
WesCossick Merge pull request #361 from NextStepWebs/deve... on Jun 14, 2016 **493**

View code

SimpleMDE - Markdown Editor

A drop-in JavaScript textarea replacement for writing beautiful and understandable Markdown. The WYSIWYG-esque editor allows users who may be less experienced with Markdown to use familiar toolbar buttons and shortcuts. In addition, the syntax is rendered while editing to clearly show the expected result. Headings are larger, emphasized words are italicized, links are underlined, etc. SimpleMDE is one of the first editors to feature both built-in autosaving and spell checking.

Demo



Why not a WYSIWYG editor or pure Markdown?

WYSIWYG editors that produce HTML are often complex and buggy. Markdown solves this problem in many ways, plus Markdown can be rendered natively on more platforms than HTML. However, Markdown is not a syntax that an average user will be familiar with, nor is it visually clear while editing. In otherwords, for an unfamiliar user, the syntax they write will make little sense until they click the preview button. SimpleMDE has been designed to bridge this gap for non-technical users who are less familiar with or just learning Markdown syntax.

Install

Via npm.

```
npm install simplemde --save
```

Via bower.

```
bower install simplemde --save
```

Via jsDelivr. *Please note, jsDelivr may take a few days to update to the latest release.*

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/simplemde/latest/simple
```

```
<script src="https://cdn.jsdelivr.net/simplemde/latest/simplemde.min.js"></sc
```

Quick start

After installing, load SimpleMDE on the first textarea on a page

```
<script>
var simplemde = new SimpleMDE();
</script>
```

Using a specific textarea

Pure JavaScript method

```
<script>
var simplemde = new SimpleMDE({ element: document.getElementById("MyID") });
</script>
```

jQuery method

```
<script>
var simplemde = new SimpleMDE({ element: $("#MyID")[0] });
</script>
```

Get/set the content

```
simplemde.value();
```

```
simplemde.value("This text will appear in the editor");
```

Configuration

- **autoDownloadFontAwesome**: If set to `true`, force downloads Font Awesome (used for icons). If set to `false`, prevents downloading. Defaults to `undefined`,

which will intelligently check whether Font Awesome has already been included, then download accordingly.

- **autofocus**: If set to `true`, autofocuses the editor. Defaults to `false`.
- **autosave**: *Saves the text that's being written and will load it back in the future. It will forget the text when the form it's contained in is submitted.*
  - **enabled**: If set to `true`, autosave the text. Defaults to `false`.
  - **delay**: Delay between saves, in milliseconds. Defaults to `10000` (10s).
  - **uniqueId**: You must set a unique string identifier so that SimpleMDE can autosave. Something that separates this from other instances of SimpleMDE elsewhere on your website.
- **blockStyles**: Customize how certain buttons that style blocks of text behave.
  - **bold** Can be set to `**` or `__`. Defaults to `**`.
  - **code** Can be set to ```` ``` ```` or `~~~`. Defaults to ```` ``` ````.
  - **italic** Can be set to `*` or `_`. Defaults to `*`.
- **element**: The DOM element for the textarea to use. Defaults to the first textarea on the page.
- **forceSync**: If set to `true`, force text changes made in SimpleMDE to be immediately stored in original textarea. Defaults to `false`.
- **hideIcons**: An array of icon names to hide. Can be used to hide specific icons shown by default without completely customizing the toolbar.
- **indentWithTabs**: If set to `false`, indent using spaces instead of tabs. Defaults to `true`.
- **initialValue**: If set, will customize the initial value of the editor.
- **insertTexts**: Customize how certain buttons that insert text behave. Takes an array with two elements. The first element will be the text inserted before the cursor or highlight, and the second element will be inserted after. For example, this is the default link value: `["[", "](http://)"]`.
  - horizontalRule
  - image
  - link
  - table
- **lineWrapping**: If set to `false`, disable line wrapping. Defaults to `true`.
- **parsingConfig**: Adjust settings for parsing the Markdown during editing (not previewing).
  - **allowAtxHeaderWithoutSpace**: If set to `true`, will render headers without a space after the `#`. Defaults to `false`.

    - **strikethrough**: If set to `false`, will not process GFM strikethrough syntax. Defaults to `true`.
    - **underscoresBreakWords**: If set to `true`, let underscores be a delimiter for separating words. Defaults to `false`.
- **placeholder**: Custom placeholder that should be displayed
- **previewRender**: Custom function for parsing the plaintext Markdown and returning HTML. Used when user previews.
- **promptURLs**: If set to `true`, a JS alert window appears asking for the link or image URL. Defaults to `false`.
- **renderingConfig**: Adjust settings for parsing the Markdown during previewing (not editing).
    - **singleLineBreaks**: If set to `false`, disable parsing GFM single line breaks. Defaults to `true`.
    - **codeSyntaxHighlighting**: If set to `true`, will highlight using highlight.js. Defaults to `false`. To use this feature you must include highlight.js on your page. For example, include the script and the CSS files like:

      ```
      <script
      src="https://cdn.jsdelivr.net/highlight.js/latest/highlight.min.js">
      </script>
      <link rel="stylesheet"
      href="https://cdn.jsdelivr.net/highlight.js/latest/styles/github.min.c
      ss">
      ```
- **shortcuts**: Keyboard shortcuts associated with this instance. Defaults to the array of shortcuts.
- **showIcons**: An array of icon names to show. Can be used to show specific icons hidden by default without completely customizing the toolbar.
- **spellChecker**: If set to `false`, disable the spell checker. Defaults to `true`.
- **status**: If set to `false`, hide the status bar. Defaults to the array of built-in status bar items.
    - Optionally, you can set an array of status bar items to include, and in what order. You can even define your own custom status bar items.
- **styleSelectedText**: If set to `false`, remove the `CodeMirror-selectedtext` class from selected lines. Defaults to `true`.
- **tabSize**: If set, customize the tab size. Defaults to `2`.
- **toolbar**: If set to `false`, hide the toolbar. Defaults to the array of icons.
- **toolbarTips**: If set to `false`, disable toolbar button tips. Defaults to `true`.

```
// Most options demonstrate the non-default behavior
var simplemde = new SimpleMDE({
	autofocus: true,
	autosave: {
		enabled: true,
		uniqueId: "MyUniqueID",
		delay: 1000,
	},
	blockStyles: {
		bold: "__",
		italic: "_"
	},
	element: document.getElementById("MyID"),
	forceSync: true,
	hideIcons: ["guide", "heading"],
	indentWithTabs: false,
	initialValue: "Hello world!",
	insertTexts: {
		horizontalRule: ["", "\n\n-----\n\n"],
		image: ["![](http://", ")"],
		link: ["[", "](http://)"],
		table: ["", "\n\n| Column 1 | Column 2 | Column 3 |\n| ------
	},
	lineWrapping: false,
	parsingConfig: {
		allowAtxHeaderWithoutSpace: true,
		strikethrough: false,
		underscoresBreakWords: true,
	},
	placeholder: "Type here...",
	previewRender: function(plainText) {
		return customMarkdownParser(plainText); // Returns HTML from
	},
	previewRender: function(plainText, preview) { // Async method
		setTimeout(function(){
			preview.innerHTML = customMarkdownParser(plainText);
		}, 250);

		return "Loading...";
	},
	promptURLs: true,
	renderingConfig: {
		singleLineBreaks: false,
		codeSyntaxHighlighting: true,
	},
	shortcuts: {
		drawTable: "Cmd-Alt-T"
	},
```

```
        showIcons: ["code", "table"],
        spellChecker: false,
        status: false,
        status: ["autosave", "lines", "words", "cursor"], // Optional usage
        status: ["autosave", "lines", "words", "cursor", {
                classNam: "keystrokes",
                defaultValue: function(el) {
                        this.keystrokes = 0;
                        el.innerHTML = "0 Keystrokes";
                },
                onUpdate: function(el) {
                        el.innerHTML = ++this.keystrokes + " Keystrokes";
                }
        }], // Another optional usage, with a custom status bar item that cou
        styleSelectedText: false,
        tabSize: 4,
        toolbar: false,
        toolbarTips: false,
});
```

Toolbar icons

Below are the built-in toolbar icons (only some of which are enabled by default), which can be reorganized however you like. "Name" is the name of the icon, referenced in the JS. "Action" is either a function or a URL to open. "Class" is the class given to the icon. "Tooltip" is the small tooltip that appears via the `title=""` attribute. Note that shortcut hints are added automatically and reflect the specified action if it has a keybind assigned to it (i.e. with the value of `action` set to `bold` and that of `tooltip` set to `Bold`, the final text the user will see would be "Bold (Ctrl-B)").

Additionally, you can add a separator between any icons by adding `"|"` to the toolbar array.

Name	Action	Tooltip Class
bold	toggleBold	Bold fa fa-bold
italic	toggleItalic	Italic fa fa-italic
strikethrough	toggleStrikethrough	Strikethrough fa fa-strikethrough

Name	Action	Tooltip Class
heading	toggleHeadingSmaller	Heading fa fa-header
heading-smaller	toggleHeadingSmaller	Smaller Heading fa fa-header
heading-bigger	toggleHeadingBigger	Bigger Heading fa fa-lg fa-header
heading-1	toggleHeading1	Big Heading fa fa-header fa-header-x fa-header-1
heading-2	toggleHeading2	Medium Heading fa fa-header fa-header-x fa-header-2
heading-3	toggleHeading3	Small Heading fa fa-header fa-header-x fa-header-3
code	toggleCodeBlock	Code fa fa-code
quote	toggleBlockquote	Quote fa fa-quote-left
unordered-list	toggleUnorderedList	Generic List fa fa-list-ul
ordered-list	toggleOrderedList	Numbered List fa fa-list-ol
clean-block	cleanBlock	Clean block fa fa-eraser fa-clean-block
link	drawLink	Create Link fa fa-link
image	drawImage	Insert Image fa fa-picture-o

Name	Action	Tooltip Class
table	drawTable	Insert Table fa fa-table
horizontal-rule	drawHorizontalRule	Insert Horizontal Line fa fa-minus
preview	togglePreview	Toggle Preview fa fa-eye no-disable
side-by-side	toggleSideBySide	Toggle Side by Side fa fa-columns no-disable no-mobile
fullscreen	toggleFullScreen	Toggle Fullscreen fa fa-arrows-alt no-disable no-mobile
guide	This link	Markdown Guide fa fa-question-circle

Customize the toolbar using the `toolbar` option like:

```
// Customize only the order of existing buttons
var simplemde = new SimpleMDE({
        toolbar: ["bold", "italic", "heading", "|", "quote"],
});

// Customize all information and/or add your own icons
var simplemde = new SimpleMDE({
        toolbar: [{
                        name: "bold",
                        action: SimpleMDE.toggleBold,
                        className: "fa fa-bold",
                        title: "Bold",
                },
                {
                        name: "custom",
                        action: function customFunction(editor){
                                // Add your own code
                        },
                        className: "fa fa-star",
                        title: "Custom Button",
                },
                "|", // Separator
                ...
```

```
        ],
});
```

Keyboard shortcuts

SimpleMDE comes with an array of predefined keyboard shortcuts, but they can be altered with a configuration option. The list of default ones is as follows:

Shortcut	Action
Cmd-'	"toggleBlockquote"
Cmd-B	"toggleBold"
Cmd-E	"cleanBlock"
Cmd-H	"toggleHeadingSmaller"
Cmd-I	"toggleItalic"
Cmd-K	"drawLink"
Cmd-L	"toggleUnorderedList"
Cmd-P	"togglePreview"
Cmd-Alt-C	"toggleCodeBlock"
Cmd-Alt-I	"drawImage"
Cmd-Alt-L	"toggleOrderedList"
Shift-Cmd-H	"toggleHeadingBigger"
F9	"toggleSideBySide"
F11	"toggleFullScreen"

Here is how you can change a few, while leaving others untouched:

```
var simplemde = new SimpleMDE({
        shortcuts: {
                "toggleOrderedList": "Ctrl-Alt-K", // alter the shortcut for
                "toggleCodeBlock": null, // unbind Ctrl-Alt-C
                "drawTable": "Cmd-Alt-T" // bind Cmd-Alt-T to drawTable actio
        }
```

README.md

Shortcuts are automatically converted between platforms. If you define a shortcut as "Cmd-B", on PC that shortcut will be changed to "Ctrl-B". Conversely, a shortcut defined as "Ctrl-B" will become "Cmd-B" for Mac users.

The list of actions that can be bound is the same as the list of built-in actions available for toolbar buttons.

Height

To change the minimum height (before it starts auto-growing):

```
.CodeMirror, .CodeMirror-scroll {
        min-height: 200px;
}
```

Or, you can keep the height static:

```
.CodeMirror {
        height: 300px;
}
```

Event handling

You can catch the following list of events:
https://codemirror.net/doc/manual.html#events

```
var simplemde = new SimpleMDE();
simplemde.codemirror.on("change", function(){
        console.log(simplemde.value());
});
```

Removing SimpleMDE from textarea

You can revert to the initial textarea by calling the `toTextArea` method. Note that this clears up the autosave (if enabled) associated with it. The textarea will retain any text from the destroyed SimpleMDE instance.

```
var simplemde = new SimpleMDE();
...
```

```
simplemde.toTextArea();
simplemde = null;
```

Useful methods

The following self-explanatory methods may be of use while developing with SimpleMDE.

```
var simplemde = new SimpleMDE();
simplemde.isPreviewActive(); // returns boolean
simplemde.isSideBySideActive(); // returns boolean
simplemde.isFullscreenActive(); // returns boolean
simplemde.clearAutosavedValue(); // no returned value
```

How it works

SimpleMDE began as an improvement of lepture's Editor project, but has now taken on an identity of its own. It is bundled with CodeMirror and depends on Font Awesome.

CodeMirror is the backbone of the project and parses much of the Markdown syntax as it's being written. This allows us to add styles to the Markdown that's being written. Additionally, a toolbar and status bar have been added to the top and bottom, respectively. Previews are rendered by Marked using GFM.

Releases 30

1.11.2 Latest
on Jun 14, 2016

+ 29 releases

Packages

No packages published

Contributors 29