JiHong88 / suneditor Public

Code Issues 320 Pull requests 8 Discussions Actions Projects 1 Security Insights

master Go to file Code

JiHong88 Merge pull request #1339 from iveretelnyk/master ... 2 weeks ago 2,849

.github	Update issue templates	2 years ago
dist	replace invisible charatcer U+00a0 with real whit...	3 weeks ago
sample	add: #764 options.__allowedScriptTag,	7 months ago
src	Merge pull request #1339 from iveretelnyk/mast...	2 weeks ago
test	fix: #1242 Range selection bug with arrow keys,	6 months ago
.gitignore	modify: ignore file	3 years ago
.jshintignore	modify: test	5 years ago
.jshintrc	update: language fr, js hint	4 years ago
.npmignore	update .npmignore	2 years ago
LICENSE.txt	0.87	6 years ago
README.md	fix: init value	6 months ago
bower.json	version up: 2.41.3	2 years ago
karma.conf.js	modify: jasmine-core version down	5 years ago

About

Pure javascript based WYSIWYG html editor, with no dependencies.

suneditor.com

#contenteditable #vanilla-js #rich-text-editor #html-editor #wysiwyg-editor #nodependence

Readme
MIT license
Activity
1.3k stars
25 watching
307 forks
Report repository

Releases 128

v2.45.1 Latest
on Jun 18

package.json	version up: 2.45.1	6 months ago
tsconfig.json	add: module-fileManager,	3 years ago
webpack.common.js	add: FR lang, fix: Backspace in list,	4 years ago
webpack.dev.js	add: FR lang, fix: Backspace in list,	4 years ago
webpack.prod.js	add: source map	3 years ago

SunEditor

Vanilla javascript based WYSIWYG web editor, with no dependencies. SunEditor supports IE11 and all modern browsers with no dependencies and polyfill.

Demo : suneditor.com

> The Suneditor is a lightweight, flexible, customizable WYSIWYG text editor for your web applications.
>
> - Pasting from Microsoft Word and Excel.
> - Custom table selection, merge and split.
> - Media embed, images upload.
> - Can use CodeMirror, KaTeX.
> - And.. many other features :)

+ 127 releases

Sponsor this project

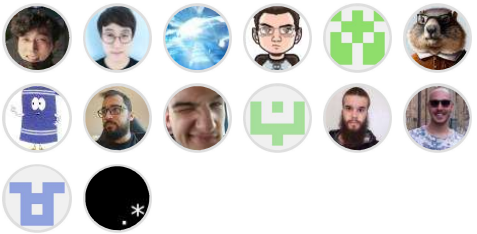
Packages

No packages published

Used by 2k

+ 2,016

Contributors 50



+ 36 contributors

Languages

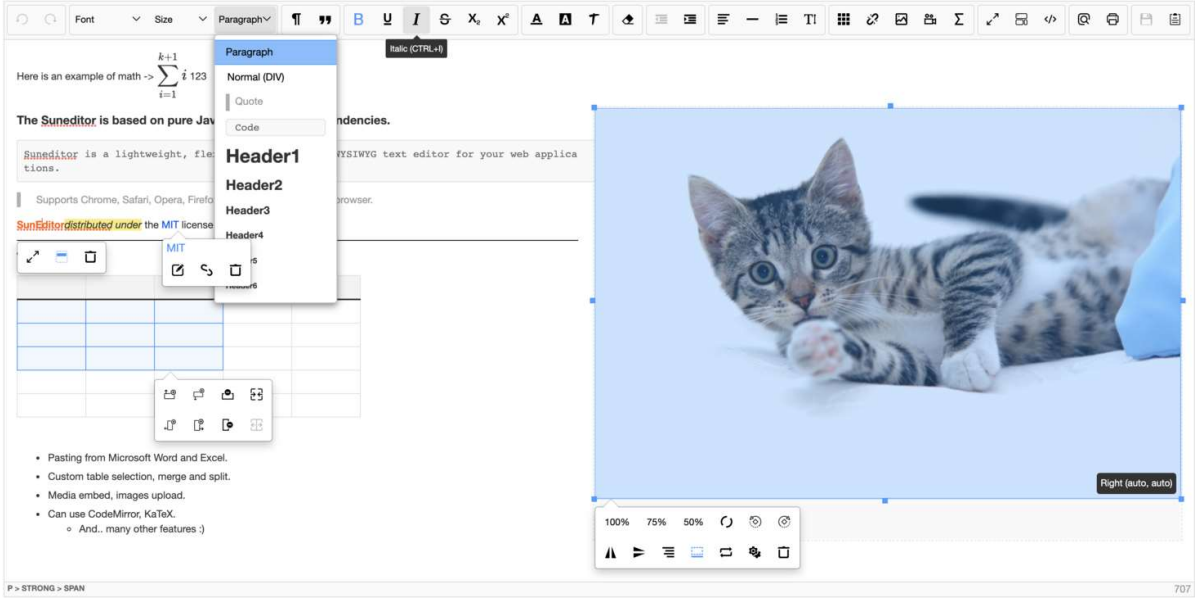


Table of contents

- Browser Support
- Install
- Getting Started
- When inserting custom tags in the editor
- Use import statement
  - Load only what you want
  - Load all plugins
  - Plugins can be used directly in the button list
- Init function
- Use CodeMirror
- Use KaTeX (math plugin)
- Options
- Functions

JavaScript 93.2% CSS 5.1%
HTML 1.7%

- Plugins list
- Examples
- Options template
- Custom plugins
- Document
- Other libraries using SunEditor
  - suneditor-react
  - angular-suneditor
  - Using SunEditor with Livewire & Alpine.JS
  - Plugin for Pluxml
  - AEM-SunEditor
- License

Browser Support

Chrome	Firefox	Opera	Safari	Edge	Internet Explorer
Yes	Yes	Yes	Yes	Yes	11+

Install

Npm

```
$ npm install suneditor --save
```

Bower

```
$ bower install suneditor --save
```

CDN

```
<link href="https://cdn.jsdelivr.net/npm/suneditor@latest/dist/css/suned .m
<!-- <link href="https://cdn.jsdelivr.net/npm/suneditor@latest/assets/css/sune
<!-- <link href="https://cdn.jsdelivr.net/npm/suneditor@latest/assets/css/sune
<script src="https://cdn.jsdelivr.net/npm/suneditor@latest/dist/suneditor.min.
<!-- languages (Basic Language: English/en) -->
<script src="https://cdn.jsdelivr.net/npm/suneditor@latest/src/lang/ko.js"></s
```

jsdelivr/suneditor

Getting Started

1. Target Element

```
<textarea id="sample">Hi</textarea>
```

2. Create

```
/**
* ID : 'suneditor_sample'
* ClassName : 'sun-eidtor'
*/
// ID or DOM object
const editor = SUNEDITOR.create((document.getElementById('sample') || 'sample'
    // All of the plugins are loaded in the "window.SUNEDITOR" object in dist/
    // Insert options
```

```
    // Language global object (default: en)
    lang: SUNEDITOR_LANG['ko']
});
```

3. Contents display

```
When you display a document created by suneditor
You need to include "src/assets/css/suneditor-contents.css" or "dist/css/sunec
Then add "sun-editor-editable" to the class name of the Tag element that displ
If you are using RTL mode, you also need to add "se-rtl".
In "suneditor-contents.css", you can define the style of all the tags created
```

When inserting custom tags in the editor

```
- Empty tags without meaning or tags that do not fit the editor's format
are modified or deleted.
    Tags with the class name "se-component" or "__se__tag" of the top-level
tag will not be deleted.
        "se-component" is the component type of the editor.
        Class name for wrapper tags such as images and videos.
```

Use import statement

1. Load only what you want

```
import 'suneditor/dist/css/suneditor.min.css'
// import 'suneditor/assets/css/suneditor.css'
// import 'suneditor/assets/css/suneditor-contents.css'
import suneditor from 'suneditor'

// How to import plugins
import image from 'suneditor/src/plugins/dialog/link'
import list from 'suneditor/src/plugins/submenu/list'
import {font, video} from 'suneditor/src/plugins'

// How to import language files (default: en)
import lang from 'suneditor/src/lang'
import {ko} from 'suneditor/src/lang'
import de from 'suneditor/src/lang/de'

suneditor.create('sample', {
    plugins: [font, video, image, list],
    buttonList: [
        ['font', 'video', 'image', 'list']
    ],
    lang: lang.ko
});
```

2. Load all plugins

```
import 'suneditor/dist/css/suneditor.min.css'
import suneditor from 'suneditor'
import plugins from 'suneditor/src/plugins'

suneditor.create('sample', {
    plugins: plugins,
    buttonList: [
        ['undo', 'redo'],
        ['font', 'fontSize', 'formatBlock'],
```

```
            ['paragraphStyle', 'blockquote'],
            ['bold', 'underline', 'italic', 'strike', 'subscript', 'superscript'],
            ['fontColor', 'hiliteColor', 'textStyle'],
            ['removeFormat'],
            '/', // Line break
            ['outdent', 'indent'],
            ['align', 'horizontalRule', 'list', 'lineHeight'],
            ['table', 'link', 'image', 'video', 'audio' /** ,'math' */], // You mu
            /** ['imageGallery'] */ // You must add the "imageGalleryUrl".
            ['fullScreen', 'showBlocks', 'codeView'],
            ['preview', 'print'],
            ['save', 'template'],
            /** ['dir', 'dir_ltr', 'dir_rtl'] */ // "dir": Toggle text direction,
        ]
    })

    // You can also load what you want
    suneditor.create('sample', {
        plugins: [plugins.font],
        // Plugins can be used directly in the button list
        buttonList: [
            ['font', plugins.image]
        ]
    })
```

3. Plugins can be used directly in the button list

```
import 'suneditor/dist/css/suneditor.min.css'
import suneditor from 'suneditor'
import {align, font, fontSize, fontColor, hiliteColor,
        horizontalRule, image, template} from 'suneditor/src/plugins'

suneditor.create('sample', {
    buttonList: [
```

```
        ['undo', 'redo', 'removeFormat'],
        [align, font, fontSize, fontColor, hiliteColor],
        [horizontalRule, image, template]
    ],
})
```

Init function

The init function can be used by predefining options and calling the cre
function on the returned object.
The value of the option argument put in the "create" function call takes
precedence

```
import 'suneditor/dist/css/suneditor.min.css'
import suneditor from 'suneditor'
import plugins from 'suneditor/src/plugins'

// all plugins
const initEditor = suneditor.init({
    plugins: plugins,
    height: 200,
    buttonList: [
        [
        'undo', 'redo',
        'font', 'fontSize', 'formatBlock',
        'paragraphStyle', 'blockquote',
        'bold', 'underline', 'italic', 'strike', 'subscript', 'superscript',
        'fontColor', 'hiliteColor', 'textStyle',
        'removeFormat',
        'outdent', 'indent',
        'align', 'horizontalRule', 'list', 'lineHeight',
        'table', 'link', 'image', 'video', 'audio', /** 'math', */ // You must
```

```
        /** 'imageGallery', */ // You must add the "imageGalleryUrl".
        'fullScreen', 'showBlocks', 'codeView',
        'preview', 'print', 'save', 'template',
        /** 'dir', 'dir_ltr', 'dir_rtl' */ // "dir": Toggle text direction, "c
        ]
    ]
});

initEditor.create('sample_1', {
    // The value of the option argument put in the "create" function call take
});

initEditor.create('sample_2', {
    // The value of the option argument put in the "create" function call take
    height: 'auto',
    buttonList: [
        ['bold', 'underline', 'italic'],
        ['removeFormat'],
        ['preview', 'print']
    ]
});
```

Use CodeMirror

```
<!-- https://github.com/codemirror/CodeMirror -->
<!-- codeMirror (^5.0.0) -->
<!-- Use version 5.x.x -->
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/codemirror@5.49.0/li
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.49.0/lib/codemirror.mir
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.49.0/mode/htmlmixed/htm
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.49.0/mode/xml/xml.js"><
<script src="https://cdn.jsdelivr.net/npm/codemirror@5.49.0/mode/css/css.js"><
```

```
import 'suneditor/dist/css/suneditor.min.css'
import suneditor from 'suneditor'
// Import codeMirror
import CodeMirror from 'codemirror'
import 'codemirror/mode/htmlmixed/htmlmixed'
import 'codemirror/lib/codemirror.css'

suneditor.create('sample', {
    codeMirror: CodeMirror // window.CodeMirror,
    // Set options
    // codeMirror: {
    //     src: CodeMirror,
    //     options: {...}
    // }
    buttonList: [
        ['codeView']
    ],
    height: 400
});
```

Use KaTeX (math plugin)

```
<!-- https://github.com/KaTeX/KaTeX -->
<!-- KaTeX (^0.11.1) -->
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/katex@0.11.1/dist/ka
<script src="https://cdn.jsdelivr.net/npm/katex@0.11.1/dist/katex.min.js"></sc
```

```
import 'suneditor/dist/css/suneditor.min.css'
import suneditor from 'suneditor'
// Import katex
import katex from 'katex'
```

```
import 'katex/dist/katex.min.css'

suneditor.create('sample', {
    katex: katex // window.katex,
    // Set options
    // katex: {
    //     src: katex,
    //     options: {...}
    // }
    buttonList: [
        ['math']
    ]
});
```

Options

```
plugins: [
    /** command */
    blockquote,
    /** Submenu */
    align,
    font,
    fontColor,
    fontSize,
    formatBlock,
    hiliteColor,
    horizontalRule,
    lineHeight,
    list,
    paragraphStyle,
    table,
    template,
    textStyle,
    /** Dialog */
    image,
```

```
        link,
        video,
        audio,
        math, // You must add the 'katex' library at options to use the 'math' plu
        /** File browser */
        // You must add the "imageGalleryUrl".
        // A button is added to the image modal.
        // You can also use image gallery by adding it directly to the button list
        imageGallery
    ]
    : Plugins array.     default: null {Array}
    // * Custom options and default options are all treated the same.
    // * When using a custom plugin and a default plugin together, register as fol
    // * {custom_plugin, ...plugins}

    // Values
    lang            : language object.   default : en {Object}
    defaultTag      : Specifies default tag name of the editor.     default: 'p' {
    textTags        : You can change the tag of the default text button.   default
                      ex) {
                          bold: 'b',
                          strike: 's'
                      }
    value           : Initial value(html string) of the edit area.
                      If not, the value of the "target textarea".   default: null
    allowedClassNames  : Specifies the allowed class name, It can be specified in
                         Appended before the "default" value. (`${option}|${defaul
    historyStackDelayTime : When recording the history stack, this is the delay ti
    frameAttrbutes  : Specifies the properties of the editing area DIV.     defaul
                      ex)  {
                        "spellcheck": false
                      }

    // Whitelist, Blacklist -------------------------------------------------------
    // (You can use regular expression syntax.)
    // _defaultTagsWhitelist : 'br|p|div|pre|blockquote|h1|h2|h3|h4|h5|h6|ol|ul|li
    addTagsWhitelist      : Add tags to the default tags whitelist of editor.   de
```

```
                        ex) 'mark|canvas|label|select|option|input|//' // "//'
                        ex) '*' // This means all tags are allowed. (Not avai]
    tagsBlacklist       : Blacklist of the editor default tags.                de
                        ex) 'h1|h2'
    // _editorTagsWhitelist  : _defaultTagsWhitelist + addTagsWhitelist - tagsBlac
    pasteTagsWhitelist  : Whitelist of tags when pasting.                      de
                        ex) 'p|h1|h2|h3'
                        ex) '*' // This means all tags are allowed. (Not avai]
    pasteTagsBlacklist  : Blacklist of tags when pasting.                      de
                        ex) 'h1|h2'
    attributesWhitelist : Add attributes whitelist of tags that should be kept u
                        // -- Fixed whitelist --
                        // Native attributes: 'contenteditable|colspan|rowspar
                        // Editor attributes: 'data-format|data-size|data-file
                        ex) {
                            'all': 'style|data-.+', // Apply to all tags
                            'input': 'checked|name' // Apply to input tag
                            '???': '*' // "*" === all attributes
                        }
    attributesBlacklist : Add attribute blacklist of tags that should be deletec
                        ex) {
                            'all': 'id', // Apply to all tags
                            'input': 'style' // Apply to input tag
                            '???': '*' // "*" === all attributes
                        }
    // Layout-------------------------------------------------------------------
    mode            : The mode of the editor ('classic', 'inline', 'balloon', 'bal
    rtl             : If true, the editor is set to RTL(Right To Left) mode.   def
    lineAttrReset   : Deletes other attributes except for the property set at the
                      If there is no value, no all attribute is deleted.    defaul
                      ex) 'class|style': Attributes other than "class" and "style"
                          '*': All attributes are deleted at line break.
    toolbarWidth    : The width of the toolbar. Applies only when the editor mode
                      'inline' or 'balloon' mode.     default: 'auto' {Number|Stri
    toolbarContainer: A custom HTML selector placing the toolbar inside.
                      The class name of the element must be 'sun-editor'.
                      Element or querySelector argument.     default: null {Elemer
```

```
                    ex) document.querySelector('#id') || '#id'
    stickyToolbar   : Top offset value of "sticky toolbar".
                      Set to 0, '0px', '50px'...
                      If set to -1 or false or null to turn off.        default: 0
    hideToolbar     : The toolbar is rendered hidden.                   default: f
    fullScreenOffset: Top offset value of "full Screen".
                      Set to 0, '0px', '50px'...     default: 0 {Number|String}
    iframe          : Content will be placed in an iframe and isolated from the re
    fullPage        : Allows the usage of HTML, HEAD, BODY tags and DOCTYPE declar
    iframeAttributes  : Attributes of the iframe.                       default: n
                        ex) {'scrolling': 'no'}
    iframeCSSFileName : Name or Array of the CSS file to apply inside the iframe.
                        You can also use regular expressions.
                        Applied by searching by filename in the link tag of docume
                        or put the URL value (".css" can be omitted).   default: '
                        ex) '.+' or ['suneditor', 'http://suneditor.com/sample/css
    previewTemplate : A template of the "preview".
                      The {{contents}} part in the HTML string is replaced with th
                      ex) "<div style='width:auto; max-width:1080px; margin:auto;'
    printTemplate   : A template of the "print".
                      The {{contents}} part in the HTML string is replaced with th
                      ex) "<div style='width:auto; max-width:1080px; margin:auto;'
    codeMirror      : If you put the CodeMirror object as an option, you can do Co
                      Use version 5.x.x // https://github.com/codemirror/CodeMirro
                      ex) codeMirror: CodeMirror // Default option
                          codeMirror: { // Custom option
                            src: CodeMirror,
                            options: {
                                /** default options **
                                * mode: 'htmlmixed',
                                * htmlMode: true,
                                * lineNumbers: true
                                * lineWrapping: true
                                */
                            }
                          }
    katex           : Required library for math plugins.              default: nu
```

```
                    Use version 0.x.x // https://github.com/KaTeX/KaTeX
                    ex) katex: katex // Default option
                        katex: { // Custom option
                          src: katex,
                          options: {
                              /** default options **
                              * throwOnError: false,
                              */
                          }
                        }
mathFontSize    : Math plugin font size list.                        default: [
                  Default value: [
                    {text: '1', value: '1em', default: true},
                    {text: '1.5', value: '1.5em'},
                    {text: '2', value: '2em'},
                    {text: '2.5', value: '2.5em'}
                  ]

// Display-------------------------------------------------------------------
position        : The position property of suneditor.                 default: r
display         : The display property of suneditor.                  default: '
popupDisplay    : Size of background area when activating dialog window ('ful

// Bottom resizing bar-------------------------------------------------------
resizingBar     : Show the bottom resizing bar.
                  If 'height' value is 'auto', it will not be resized. default
showPathLabel   : Displays the current node structure to resizingBar.  default
resizeEnable  : Enable/disable resize function of bottom resizing bar.   defa
resizingBarContainer: A custom HTML selector placing the resizing bar inside.
                      The class name of the element must be 'sun-editor'.
                      Element or querySelector argument.     default: null {E
                      ex) document.querySelector('#id') || '#id'

// Character count-----------------------------------------------------------
charCounter     : Shows the number of characters in the editor.
                  If the maxCharCount option has a value, it becomes true. de
charCounterType : Defines the calculation method of the "charCounter" option.
```

```
                  'char': Characters length.
                  'byte': Binary data size of characters.
                  'byte-html': Binary data size of the full HTML string.   def
charCounterLabel: Text to be displayed in the "charCounter" area of the bottom
                  Screen ex) 'charCounterLabel : 20/200'.          default: r
maxCharCount    : The maximum number of characters allowed to be inserted into

// Width size-----------------------------------------------------------------
width           : The width size of the editor.                  default: c
minWidth        : The min-width size of the editor.
                  Used when 'width' value is 'auto' or '~%'.     default: r
maxWidth        : The max-width size of the editor.
                  Used when 'width' value is 'auto' or '~%'.     default: r

// Height size----------------------------------------------------------------
height          : The height size of the editor.                 default: c
minHeight       : The min-height size of the editor.
                  Used when 'height' value is 'auto'.            default: r
maxHeight       : The max-height size of the editor.
                  Used when 'height' value is 'auto'.            default: r

// Editing area --------------------------------------------------------------
className       : Add a "class" to the editing area[.sun-editor-editable].
defaultStyle    : You can define the style of the editing area[.sun-editor-edi
                  It affects the entire editing area.            default: '
                  ('z-index', 'position' and 'width' properties apply to the t
                  ex) 'font-family: cursive; font-size: 10px;'

// Defining menu items--------------------------------------------------------
font            : Change default font-family array.              default: [
                  Default value: [
                    'Arial', 'Comic Sans MS', 'Courier New', 'Impact',
                    'Georgia','tahoma', 'Trebuchet MS', 'Verdana'
                  ]
fontSize        : Change default font-size array.                default: [
                  Default value: [
                    8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 36, 48,
```

```
                    ]
    fontSizeUnit    : The font size unit.                                  default: '
    alignItems      : A list of drop-down options for the 'align' plugin.   defaul
    formats         : Change default formatBlock array.                    default: [
                    Default value: [
                      'p', 'div', 'blockquote', 'pre', 'h1', 'h2', 'h3', 'h4', '
                      // "blockquote": range format, "pre": free format, "Other
                    ],
                    Custom: [{
                      tag: 'div', // Tag name
                      name: 'Custom div' || null, // default: tag name
                      command: 'replace' || 'range' || 'free', // default: "rep
                      class: '__se__format__replace_xxx' || '__se__format__range
                      // Class names must always begin with "__se__format__(rep
                    }]
    colorList       : Change default color array of color picker.          default: [
                    Default value: [
                      '#ff0000', '#ff5e00', '#ffe400', '#abf200', '#00d8ff', '#0
                      '#ffd8d8', '#fae0d4', '#faf4c0', '#e4f7ba', '#d4f4fa', '#d
                      '#ffa7a7', '#ffc19e', '#faed7d', '#cef279', '#b2ebf4', '#b
                      '#f15f5f', '#f29661', '#e5d85c', '#bce55c', '#5cd1e5', '#0
                      '#980000', '#993800', '#998a00', '#6b9900', '#008299', '#0
                      '#670000', '#662500', '#665c00', '#476600', '#005766', '#0
                    ]
                    ex) [
                      ['#ccc', '#dedede', 'OrangeRed', 'Orange', 'RoyalBlue', 'S
                      ['SlateGray', 'BurlyWood', 'DeepPink', 'FireBrick', 'Gold'
                    ]
    lineHeights     : Change default line-height array.                    default: [
                    Default value: [
                      {text: '1', value: 1},
                      {text: '1.15', value: 1.15},
                      {text: '1.5', value: 1.5},
                      {text: '2', value: 2}
                    ]
                    ex) [
                      {text: 'Single', value: 1},
```

```
                    {text: 'Double', value: 2}
                  ]
paragraphStyles : You can apply custom class to format.
                  ex) '.sun-editor-editable .__se__customClass'
                      '.sun-editor .__se__customClass' // If you want to apply
                  Default value: [
                    {
                        name: 'Spaced', // Format style name
                        class: '__se__p-spaced', // Define style for used clas
                        _class: '' // You can control the style of the tags di
                    },
                    {
                        name: 'Bordered',
                        class: '__se__p-bordered'
                    },
                    {
                        name: 'Neon',
                        class: '__se__p-neon'
                    }
                  ]
                  ex) [
                      'spaced', 'neon', // The default value is called by name
                      {
                          name: 'Custom',
                          class: '__se__customClass'
                      }
                  ]
textStyles      : You can apply custom style or class to selected text.
                  ex(using a class)) '.sun-editor-editable .__se__customClass'
                                     '.sun-editor .__se__customClass' // If yo
                  Default value: [
                    {
                        name: 'Code',
                        class: '__se__t-code',
                        tag: 'code',
                    },
                    {
```

```
            name: 'Translucent', // Text style name
            style: 'opacity: 0.5;', // Style query
            tag: 'span', // Style tag name (default: span)
            _class: '' // You can control the style of the tags di
        },
        {
            name: 'Shadow',
            class: '__se__t-shadow', // Class names (Class names m
            tag: 'span'
        }
    ]
    ex) [
        'Code', // The default value is called by name only and
        {
            name: 'Emphasis',
            style: '-webkit-text-emphasis: filled;',
```

README.md

```
// Image--------------------------------------------------------------------
imageResizing   : Can resize the image.                                  default:
imageHeightShow : Choose whether the image height input is visible.      default:
imageAlignShow  : Choose whether the image align radio buttons are visible.
imageWidth      : The default width size of the image frame.             default:
imageHeight     : The default height size of the image frame.            default:
imageSizeOnlyPercentage : If true, image size can only be scaled by percentage
imageRotation   : Choose whether to image rotation buttons display.
                  When "imageSizeOnlyPercentage" is "true" or  or "imageHeight
                  If you want the button to be visible, put it a true.     def
imageFileInput  : Choose whether to create a file input tag in the image uploa
imageUrlInput   : Choose whether to create a image url input tag in the image
                  If the value of imageFileInput is false, it will be unconditi
imageUploadHeader : Http Header when uploading images.                default: r
imageUploadUrl  : The image upload to server mapping address.         default: r
                  (When not used the "imageUploadUrl" option, image is enters
                  ex) "/editor/uploadImage"
```

```
                    request format: {
                              "file-0": File,
                              "file-1": File
                          }
                    response format: {
                              "errorMessage": "insert error message",
                              "result": [
                                  {
                                      "url": "/download/editorImg/test_image.jpg
                                      "name": "test_image.jpg",
                                      "size": "561276"
                                  }
                              ]
                          }
imageUploadSizeLimit: The size of the total uploadable images (in bytes).
                      Invokes the "onImageUploadError" method.  default: null
imageMultipleFile: If true, multiple images can be selected.    default: false
imageAccept      : Define the "accept" attribute of the input.  default: "*" {
                   ex) "*" or ".jpg, .png .."
// Image - image gallery
imageGalleryUrl     : The url of the image gallery, if you use the image galle
                      When "imageUrlInput" is true, an image gallery button is
                      You can also use it by adding "imageGallery" to the butt
                      ex) "/editor/getGallery"
                      response format: {
                            "result": [
                                {
                                    "src": "/download/editorImg/test_image.jpg
                                    "thumbnail": "/download/editorImg/test_thu
                                    "name": "Test image", // @Option - default
                                    "alt": "Alt text", // @Option - default: s
                                    "tag": "Tag name" // @Option
                                }
                            ],
                            "nullMessage": "Text string or HTML string", // It
                            "errorMessage": "Insert error message", // It is d
                        }
```

```
                    You can redefine the "plugins.imageGallery.drawItems" me
imageGalleryHeader: Http Header when get image gallery.         default: null

// Video-----------------------------------------------------------------
videoResizing   : Can resize the video (iframe, video).
videoHeightShow : Choose whether the video height input is visible.    default
videoAlignShow  : Choose whether the video align radio buttons are visible.
videoRatioShow  : Choose whether the video ratio options is visible.   default
videoWidth      : The default width size of the video frame.           default
videoHeight     : The default height size of the video frame.          default
videoSizeOnlyPercentage : If true, video size can only be scaled by percentage
videoRotation   : Choose whether to video rotation buttons display.
                  When "videoSizeOnlyPercentage" is "true" or "videoHeightShow
                  If you want the button to be visible, put it a true.     def
videoRatio      : The default aspect ratio of the video.
                  Up to four decimal places are allowed.             default:
videoRatioList  : Video ratio selection options.
                  default: [
                    {name: '16:9', value: 0.5625},
                    {name: '4:3', value: 0.75},
                    {name: '21:9', value: 0.4285}
                  ],
                  ex) [
                    {name: 'Classic Film 3:2', value: 0.6666},
                    {name: 'HD', value: 0.5625}
                  ]
youtubeQuery    : The query string of a YouTube embedded URL.        default:
                  It takes precedence over the value user entered.
                  ex) 'autoplay=1&mute=1&enablejsapi=1&controls=0&rel=0&modest
                    // https://developers.google.com/youtube/player_parameters
videoFileInput  : Choose whether to create a file input tag in the video uploa
videoUrlInput   : Choose whether to create a video url input tag in the video
                  If the value of videoFileInput is false, it will be unconditi
videoUploadHeader : Http Header when uploading videos.             default: n
videoUploadUrl  : The video upload to server mapping address.      default: n
                  ex) "/editor/uploadVideo"
                  request format: {
```

```
                        "file-0": File,
                        "file-1": File
                    }
                Use video tags. (supported video formats: '.mp4', '.webm',
                response format: {
                        "errorMessage": "insert error message",
                        "result": [
                            {
                                "url": "/download/editorVideos/test_video.
                                "name": "test_video.mp4",
                                "size": "561276"
                            }
                        ]
                    }
videoUploadSizeLimit: The size of the total uploadable videos (in bytes).
                    Invokes the "onVideoUploadError" method.  default: null
videoMultipleFile: If true, multiple videos can be selected.    default: false
videoTagAttrs    : Define "Attributes" of the video tag.
                   ex) { poster: "http://suneditor.com/docs/loading.gif", autc
videoIframeAttrs : Define "Attributes" of the iframe tag. (Youtube, Vimeo).
                   ex) { style: "border: 2px solid red;" }
videoAccept      : Define the "accept" attribute of the input.  default: "*"
                   ex) "*" or ".mp4, .avi .."

// Audio-----------------------------------------------------------------
audioWidth      : The default width size of the audio frame.        default:
audioHeight     : The default height size of the audio frame.       default:
audioFileInput  : Choose whether to create a file input tag in the audio uploa
audioUrlInput   : Choose whether to create a audio url input tag in the audio
                  If the value of audioFileInput is false, it will be unconditi
audioUploadHeader : Http Header when uploading audios.              default: n
audioUploadUrl  : The audio upload to server mapping address.       default: n
                  ex) "/editor/uploadAudio"
                  request format: {
                          "file-0": File,
                          "file-1": File
                      }
```

```
                    Use audio tags. (supported audio formats: '.mp4', '.webm', '
                    response format: {
                              "errorMessage": "insert error message",
                              "result": [
                                  {
                                      "url": "/download/editorAudios/test_audio.
                                      "name": "test_audio.mp3",
                                      "size": "561276"
                                  }
                              ]
                          }
audioUploadSizeLimit: The size of the total uploadable audios (in bytes).
                      Invokes the "onAudioUploadError" method.  default: null
audioMultipleFile: If true, multiple audios can be selected.    default: false
audioTagAttrs    : Define "Attributes" of the audio tag.        default: null
                   ex) { controlslist: "nodownload", autoplay: true }
videoAccept      : Define the "accept" attribute of the input.  default: "*" {
                   ex) "*" or ".mp3, .wav .."

// Table--------------------------------------------------------------------
tableCellControllerPosition : Define position to the table cell controller('ce

// Link---------------------------------------------------------------------
linkTargetNewWindow : Default checked value of the "Open in new window" checkb
linkProtocol    : Default protocol for the links. ('link', 'image', 'video', '
                  This applies to all plugins that enter the internet url.   c
linkRel         : Defines "rel" attribute list of anchor tag.   default: [] {A
                  // https://www.w3schools.com/tags/att_a_rel.asp
                  ex) [
                    'author',
                    'external',
                    'help',
                    'license',
                    'next',
                    'follow',
                    'nofollow',
                    'noreferrer',
```

```
                    'noopener',
                    'prev',
                    'search',
                    'tag'
                ]
    linkRelDefault  : Defines default "rel" attributes of anchor tag.   default: {
                    ex) linkRelDefault: {
                            default: 'nofollow', // Default rel
                            check_new_window: 'noreferrer noopener', // When "oper
                            check_bookmark: 'bookmark' // When "bookmark" is check
                        },
                        // If properties other than "default" start with "only:",
                        linkRelDefault: {
                            check_new_window: 'only:noreferrer noopener'
                        }
    linkNoPrefix    : If true, disables the automatic prefixing of the host URL to

    // HR------------------------------------------------------------------------
    hrItems         : Defines the hr items.
                    "class" or "style" must be specified.
                    default: [
                        {name: lang.toolbar.hr_solid, class: '__se__solid'},
                        {name: lang.toolbar.hr_dashed, class: '__se__dashed'},
                        {name: lang.toolbar.hr_dotted, class: '__se__dotted'}
                    ]
                    ex) [ {name: "Outset", style: "border-style: outset;"} ]


    // Key actions---------------------------------------------------------------
    tabDisable      : If true, disables the interaction of the editor and tab key.
    shortcutsDisable: You can disable shortcuts.    default: [] {Array}
                    ex) ['bold', 'strike', 'underline', 'italic', 'undo', 'inder
    shortcutsHint   : If false, hide the shortcuts hint.    default: true {Boolear

    // Defining save button------------------------------------------------------
    callBackSave    : Callback functions that is called when the Save button is cl
                    Arguments - (contents, isChanged).
```

```
// Templates Array------------------------------------------------------------
templates       : If you use a template plugin, add it.
                  Defines a list of templates.                        default:
                  ex) [
                    {
                        name: 'Template-1',
                        html: '<p>HTML source1</p>'
                    },
                    {
                        name: 'Template-2',
                        html: '<p>HTML source2</p>'
                    }
                  ]

// ETC------------------------------------------------------------------------
__allowedScriptTag  : Allows script tags.                               default:
placeholder     : The placeholder text.                                 default:
mediaAutoSelect : Activate the media[image, video, audio] selection status imm
icons           : You can redefine icons.                               default:
                  ex) {
                      bold: '<span class="se-icon-text">B</span>',
                      table: '<i class="xx xxx></i>',
                      insert_row_above: '<svg></svg>'
                  }

// Buttons--------------------------------------------------------------------
buttonList      : Defines button list to array {Array}
                  default: [
                    ['undo', 'redo'],
                    // ['font', 'fontSize', 'formatBlock'],
                    // ['paragraphStyle', 'blockquote'],
                    ['bold', 'underline', 'italic', 'strike', 'subscript', 'su
                    // ['fontColor', 'hiliteColor', 'textStyle'],
                    ['removeFormat'],
                    ['outdent', 'indent'],
                    // ['align', 'horizontalRule', 'list', 'lineHeight'],
```

```
                    // ['table', 'link', 'image', 'video', 'math'], // You mus
                    // ['imageGallery'], // You must add the "imageGalleryUrl'
                    ['fullScreen', 'showBlocks', 'codeView'],
                    ['preview', 'print'],
                    // ['save', 'template'],
                    // ['dir', 'dir_ltr', 'dir_rtl'],
                    // '/', Line break
                  ]

---------- ex) When do not use group: ----------
                  // If you don't want to use a group, put all the buttons in
                  [
                    ['undo', 'redo', 'bold', 'underline', 'fontColor', 'table'
                  ]

----------ex) Alignment of button group:----------
                  // Set "-[align]" to the first item in the group. (default:
                  [
                    ['-left', 'undo', 'redo']
                    ['-right', 'bold', 'underline', 'italic', 'strike'],
                  ]

----------ex) Options in the button group(#):----------
                  // Set "#fix" - Fixed the order of buttons within a group ir
                  [
                    ['bold'],
                    ['preview', 'print'],
                    ['-left', '#fix', 'rtl_l', 'rtl_r']
                  ]

---------- ex) More button: ----------
                  // The more button is defined as a string starting with a cc
                  // :Identifier - Title attribute - Button's innerHTML
                  /**
                   * "Identifier": The button's identifier. Please specify uni
                   * "Title attribute": Title attribute of the button to be di
                   * "Button's innerHTML": Define the button's "innerHTML".
```

```
                 * default.xxx -> Use the attributes of "defaultIcons".
                 * (more_text, more_paragraph, more_plus, more_horizontal, m
                 * text.xxx -> Use the text.
                 * xxx -> HTML
                 */
                [
                  ['undo', 'redo'],
                  [':t-More Text-default.more_text', 'bold', 'underline', 'i
                  [':p-More Paragraph-default.more_paragraph', 'font', 'form
                  [':r-More Rich-default.more_plus', 'table', 'link', 'image
                  [':v-View-text.View', 'fullScreen', 'codeView', 'print'],
                  ['-right', ':o-More Others-<i class="xxx"></i>', 'save', '
                ]

---------- ex) Responsive setting: ----------------------------------
                // You can specify the arrangement of buttons according to t
                // Responsive settings start with a percent sign.("%").
                // %510(Number based on "px")
                [
                  // Default
                  ['undo', 'redo'],
                  ['font', 'fontSize', 'formatBlock'],
                  ['paragraphStyle', 'blockquote'],
                  ['bold', 'underline', 'italic', 'strike', 'subscript', 'su
                  ['fontColor', 'hiliteColor', 'textStyle'],
                  ['removeFormat'],
                  ['outdent', 'indent'],
                  ['align', 'horizontalRule', 'list', 'lineHeight'],
                  ['table', 'link', 'image', 'video', 'audio', 'math'],
                  ['imageGallery'],
                  ['fullScreen', 'showBlocks', 'codeView'],
                  ['preview', 'print'],
                  ['save', 'template'],
                  ['-left', '#fix', 'dir_ltr', 'dir_rtl'],
                  // (min-width:992px)
                  ['%992', [
                      ['undo', 'redo'],
```

```
            [':p-More Paragraph-default.more_paragraph', 'font', '
            ['bold', 'underline', 'italic', 'strike'],
            [':t-More Text-default.more_text', 'subscript', 'super
            ['removeFormat'],
            ['outdent', 'indent'],
            ['align', 'horizontalRule', 'list', 'lineHeight'],
            ['-right', 'dir'],
            ['-right', ':i-More Misc-default.more_vertical', 'full
            ['-right', ':r-More Rich-default.more_plus', 'table',
        ]],
        // (min-width:768px)
        ['%768', [
            ['undo', 'redo'],
            ['p-More Paragraph-default.more_paragraph', 'font', '
            [':t-More Text-default.more_text', 'bold', 'underline'
            [':e-More Line-default.more_horizontal', 'outdent', 'i
            [':r-More Rich-default.more_plus', 'table', 'link', 'i
            ['-right', 'dir'],
            ['-right', ':i-More Misc-default.more_vertical', 'full
        ]]
    ]
```

Functions

```
import suneditor from 'suneditor'

const editor = suneditor.create('example');

editor.core; // core object (The core object contains "util" and "functions".)
editor.util; // util object

// Reset the buttons on the toolbar. (Editor is not reloaded)
// You cannot set a new plugin for the button.
```

```
editor.setToolbarButtons([
    [':moreText-More Text-default.more_horizontal', 'bold', 'underline', 'stri
    ['undo', 'redo']
]);

// Add or reset option property. (Editor is reloaded)
editor.setOptions({
    minHeight: '300px',
    buttonList: [
        ['fontColor', 'hiliteColor']
    ],
    colorList: [
        ['#ccc', '#dedede', 'OrangeRed', 'Orange', 'RoyalBlue', 'SaddleBrown']
    ]
});

// Set "options.defaultStyle" style.
// Define the style of the edit area
// It can also be defined with the "setOptions" method, but the "setDefaultSty
editor.setDefaultStyle('font-family: cursive; font-size: 10px;');

// Open a notice area
editor.noticeOpen('test notice');

// Close a notice area
editor.noticeClose();

// Copies the contents of the suneditor into a [textarea]
// * not working during enabled codeView mode
editor.save();

// Gets the suneditor's context object. Contains settings, plugins, and cached
editor.getContext();

// Gets the contents of the suneditor
// * not working during enabled codeView mode
// onlyContents {Boolean}: Return only the contents of the body without header
```

```
editor.getContents(onlyContents: Boolean);
// Gets the current contents with containing parent div(div.sun-editor-editab
//  <div class="sun-editor-editable">{contents}</div>
editor.getFullContents(onlyContents: Boolean);

// Gets only the text of the suneditor contents
// * not working during enabled codeView mode
editor.getText();

// Gets a list of images uploaded to the editor
/**
 * {
 *  element: image element
 *  src: imgage src
 *  index: data index
 *  name: file name
 *  size: file size
 *  select: select function
 *  delete: delete function
 * }
 **/
editor.getImagesInfo();

// Gets uploaded files(plugin using fileManager) information list.
// image: [img], video: [video, iframe], audio: [audio]
// When the argument value is 'image', it is the same function as "getImagesIr
/**
 * {
 *  element: image element
 *  src: imgage src
 *  index: data index
 *  name: file name
 *  size: file size
 *  select: select function
 *  delete: delete function
 * }
 * pluginName: Plugin name (image, video, audio)
```

```
 **/
editor.getFilesInfo(pluginName);

// Upload images using image plugin
// document.getElementById('example_files_input').files
editor.insertImage(FileList);

// Inserts an HTML element or HTML string or plain string at the current cursc
/**
 * @param {Boolean} notCleaningData If true, inserts the HTML string without r
 * @param {Boolean} checkCharCount If true, if "options.maxCharCount" is excee
 */
editor.insertHTML('<img src="http://suneditor.com/sample/img/sunset.jpg">', tr

// Change the contents of the suneditor
editor.setContents('set contents');

// Get the editor's number of characters or binary data size.
// You can use the "charCounterType" option format.
// If argument is no value, the currently set "charCounterType" option is usec
editor.getCharCount((null || 'char' || 'byte' || 'byte-html'));

// Add content to the suneditor
editor.appendContents('append contents');

// Switch to or off "ReadOnly" mode.
editor.readOnly(true || false)

// Disable the suneditor
editor.disable();

// Enable the suneditor
editor.enable();

// Hide the suneditor
editor.hide();
```

```
// Show the suneditor
editor.show();

// Destroy the suneditor
editor.destroy();

// Toolbar methods
// Disable the toolbar
editor.toolbar.disable();

// Enable the toolbar
editor.toolbar.enable();

// Hide the toolbar
editor.toolbar.hide();

// Show the toolbar
editor.toolbar.show();

// Event functions ------------------------------------------------------------
// It can be redefined by receiving event object as parameter.
// It is not called in exceptional cases and is called after the default event
// e: event object, core: Core object
editor.onScroll = function (e, core) { console.log('onScroll', e) }

editor.onMouseDown = function (e, core) { console.log('onMouseDown', e) }

editor.onClick = function (e, core) { console.log('onClick', e) }

editor.onInput = function (e, core) { console.log('onInput', e) }

editor.onKeyDown = function (e, core) { console.log('onKeyDown', e) }

editor.onKeyUp = function (e, core) { console.log('onKeyUp', e) }

editor.onFocus = function (e, core) { console.log('onFocus', e) }
```

```
editor.onBlur = function (e, core) { console.log('onBlur', e) }

// onchange event
// contents: core.getContents(), Core object
editor.onChange = function (contents, core) { console.log('onChange', contents

// onload event
// When reloaded with the "setOptions" method, the value of the "reload" argum
editor.onload = function (core, reload) {
    console.log('onload-core', core)
    console.log('onload-reload', reload)
}

// Clipboard event.
// Called before the editor's default event action.
// If it returns false, it stops without executing the rest of the action.
/**
 * paste event
 * e: Event object
 * cleanData: HTML string modified for editor format
 * maxCharCount: maxChartCount option (true if max character is exceeded)
 * core: Core object
 */
editor.onPaste = function (e, cleanData, maxCharCount, core) { console.log('or

// Copy event.
// Called before the editor's default event action.
// If it returns false, it stops without executing the rest of the action.
/**
 * copy event
 * e: Event object
 * clipboardData: event.clipboardData
 * core: Core object
 */
editor.onCopy = function (e, clipboardData, core) { console.log('onCopy', e) }

// Cut event.
```

```
// Called before the editor's default event action.
// If it returns false, it stops without executing the rest of the action.
/**
 * cut event
 * e: Event object
 * clipboardData: event.clipboardData
 * core: Core object
 */
editor.onCut = function (e, clipboardData, core) { console.log('onCut', e) }

// Drop event.
// Called before the editor's default event action.
// If it returns false, it stops without executing the rest of the action.
/**
 * e: Event object
 * cleanData: HTML string modified for editor format
 * maxCharCount: maxChartCount option (true if max character is exceeded)
 * core: Core object
 */
editor.onDrop = function (e, cleanData, maxCharCount, core) { console.log('onD

// Save event
// Called just after the save was executed.
/**
 * contents Editor content
 * core: Core object
 */
editor.onSave = function (contents, core) {console.log(contents) };

// Called before the image is uploaded
// If true is returned, the internal upload process runs normally.
// If false is returned, no image upload is performed.
// If new fileList are returned,  replaced the previous fileList
// If undefined is returned, it waits until "uploadHandler" is executed.
/**
 * files: Files array
 * info: {
```

```
     * - linkValue: Link url value
     * - linkNewWindow: Open in new window Check Value
     * - inputWidth: Value of width input
     * - inputHeight: Value of height input
     * - align: Align Check Value
     * - isUpdate: Update image if true, create image if false
     * - element: If isUpdate is true, the currently selected image.
     * }
     * core: Core object,
     * uploadHandler: If undefined is returned, it waits until "uploadHandler" is
     *                "uploadHandler" is an upload function with "core" and "info'
     *                [upload files] : uploadHandler(files or [new File(...),])
     *                [error]        : uploadHandler("Error message")
     *                [Just finish]  : uploadHandler()
     *                [directly register] : uploadHandler(response) // Same format
     *                                      ex) {
     *                                         // "errorMessage": "insert error messa
     *                                         "result": [ { "url": "...", "name": ".
     *                                      }
     * return {Boolean|Array|undefined}
     */
    editor.onImageUploadBefore: function (files, info, core, uploadHandler) {
        return Boolean || return (new FileList) || return undefined;
    }
    // Called before the video is uploaded
    // If true is returned, the internal upload process runs normally.
    // If false is returned, no video(iframe, video) upload is performed.
    // If new fileList are returned,  replaced the previous fileList
    // If undefined is returned, it waits until "uploadHandler" is executed.
    /**
     * files: Files array
     * info: {
     * - inputWidth: Value of width input
     * - inputHeight: Value of height input
     * - align: Align Check Value
     * - isUpdate: Update video if true, create video if false
     * - element: If isUpdate is true, the currently selected video.
```

```
     * }
     * core: Core object,
     * uploadHandler: If undefined is returned, it waits until "uploadHandler" is
     *                "uploadHandler" is an upload function with "core" and "info'
     *                [upload files] : uploadHandler(files or [new File(...),])
     *                [error]        : uploadHandler("Error message")
     *                [Just finish]  : uploadHandler()
     *                [directly register] : uploadHandler(response) // Same format
     *                                    ex) {
     *                                       // "errorMessage": "insert error messa
     *                                       "result": [ { "url": "...", "name": "
     *                                    }
     * return {Boolean|Array|undefined}
     */
    editor.onVideoUploadBefore: function (files, info, core, uploadHandler) {
        return Boolean || return (new FileList) || return undefined;
    }
    // Called before the audio is uploaded
    // If true is returned, the internal upload process runs normally.
    // If false is returned, no audio upload is performed.
    // If new fileList are returned,  replaced the previous fileList
    // If undefined is returned, it waits until "uploadHandler" is executed.
    /**
     * files: Files array
     * info: {
     * - isUpdate: Update audio if true, create audio if false
     * - currentaudio: If isUpdate is true, the currently selected audio.
     * }
     * core: Core object,
     * uploadHandler: If undefined is returned, it waits until "uploadHandler" is
     *                "uploadHandler" is an upload function with "core" and "info'
     *                [upload files] : uploadHandler(files or [new File(...),])
     *                [error]        : uploadHandler("Error message")
     *                [Just finish]  : uploadHandler()
     *                [directly register] : uploadHandler(response) // Same format
     *                                    ex) {
     *                                       // "errorMessage": "insert error messa
```

```
 *                                  "result": [ { "url": "...", "name": ".
 *                              }
 * return {Boolean|Array|undefined}
 */
editor.onAudioUploadBefore: function (files, info, core, uploadHandler) {
    return Boolean || return (new FileList) || return undefined;
}

// Called when the image is uploaded, updated, deleted.
/**
 * targetElement: Target element
 * index: Uploaded index (key value)
 * state: Upload status ('create', 'update', 'delete')
 * info: {
 * - index: data index
 * - name: file name
 * - size: file size
 * - select: select function
 * - delete: delete function
 * - element: Target element
 * - src: src attribute of tag
 * }
 * remainingFilesCount: Count of remaining files to upload (0 when added as a
 * core: Core object
*/
editor.onImageUpload = function (targetElement, index, state, info, remainingF
    console.log(`targetElement:${targetElement}, index:${index}, state('create
    console.log(`info:${info}, remainingFilesCount:${remainingFilesCount}`)
}
// Called when the video(iframe, video) is is uploaded, updated, deleted
// -- arguments is same "onImageUpload" --
editor.onVideoUpload = function (targetElement, index, state, info, remainingF
    console.log(`targetElement:${targetElement}, index:${index}, state('create
    console.log(`info:${info}, remainingFilesCount:${remainingFilesCount}`)
}
// Called when the audio is is uploaded, updated, deleted
// -- arguments is same "onImageUpload" --
```

```
editor.onAudioUpload = function (targetElement, index, state, info, remainingF
    console.log(`targetElement:${targetElement}, index:${index}, state('create
    console.log(`info:${info}, remainingFilesCount:${remainingFilesCount}`)
}

// Called when the image is upload failed.
// If you return false, the default notices are not called.
/**
 * errorMessage: Error message
 * result: Response Object
 * core: Core object
 * return {Boolean}
*/
editor.onImageUploadError = function (errorMessage, result, core) {
    alert(errorMessage)
    return Boolean
}
// Called when the video(iframe, video) upload failed
// -- arguments is same "onImageUploadError" --
editor.onVideoUploadError = function (errorMessage, result, core) {
    alert(errorMessage)
    return Boolean
}
// Called when the audio upload failed
// -- arguments is same "onImageUploadError" --
editor.onAudioUploadError = function (errorMessage, result, core) {
    alert(errorMessage)
    return Boolean
}

// Called when the editor is resized using the bottom bar
// height, prevHeight are number
editor.onResizeEditor = function (height, prevHeight, core, resizeObserverEntr
    console.log(`height: ${height}, prevHeight: ${prevHeight}`, resizeObserver
    // "resizeObserverEntry" is not provided in IE Browser.
}
```

```
// Called after the "setToolbarButtons" invocation
// Can be used to tweak buttons properties (useful for custom buttons)
/**
 * buttonList: buttonList array
 * core: Core object
 */
editor.onSetToolbarButtons = function (buttonList, core) {
    console.log(`buttonList: ${buttonList}`)
}

// It replaces the default callback function of the image upload
/**
 * xmlHttp: xmlHttpRequest object
 * info: Input information
 * - linkValue: Link url value
 * - linkNewWindow: Open in new window Check Value
 * - inputWidth: Value of width input
 * - inputHeight: Value of height input
 * - align: Align Check Value
 * - isUpdate: Update image if true, create image if false
 * - element: If isUpdate is true, the currently selected image.
 * core: Core object
 */
editor.imageUploadHandler = function (xmlHttp, info, core) {
    // Editor code
    const response = JSON.parse(xmlHttp.responseText);
    if (response.errorMessage) {
        this.plugins.image.error.call(this, response.errorMessage, response);
    } else {
        this.plugins.image.register.call(this, info, response);
    }
}
/**
 * @description It replaces the default callback function of the video upload
 * xmlHttp: xmlHttpRequest object
 * info: Input information
 * - inputWidth: Value of width input
```

```
 * - inputHeight: Value of height input
 * - align: Align Check Value
 * - isUpdate: Update video if true, create video if false
 * - element: If isUpdate is true, the currently selected video.
 * core: Core object
 */
editor.videoUploadHandler = function (xmlHttp, info, core) {
    // Editor code
    const response = JSON.parse(xmlHttp.responseText);
    if (response.errorMessage) {
        this.plugins.video.error.call(this, response.errorMessage, response);
    } else {
        this.plugins.video.register.call(this, info, response);
    }
}

/**
 * @description It replaces the default callback function of the audio upload
 * xmlHttp xmlHttpRequest object
 * info Input information
 * - isUpdate: Update audio if true, create audio if false
 * - element: If isUpdate is true, the currently selected audio.
 * core Core object
 */
editor.audioUploadHandler = function (xmlHttp, info, core) {
    // Editor code
    const response = JSON.parse(xmlHttp.responseText);
    if (response.errorMessage) {
        this.plugins.audio.error.call(this, response.errorMessage, response);
    } else {
        this.plugins.audio.register.call(this, info, response);
    }
}

// An event when toggling between code view and wysiwyg view.
/**
 * isCodeView: Whether the current code view mode
```

```
 * core: Core object
 */
editor.toggleCodeView = function (isCodeView, core) {
    console.log('isCodeView', isCodeView);
}

// An event when toggling full screen.
/**
 * isFullScreen: Whether the current full screen mode
 * core: Core object
 */
editor.toggleFullScreen = function (isFullScreen, core) {
    console.log('isFullScreen', isFullScreen);
}

// Called just before the inline toolbar is positioned and displayed on the sc
/**
 * toolbar: Toolbar Element
 * context: The editor's context object (editor.getContext()|core.context)
 * core Core object
*/
editor.showInline = function (toolbar, context, core) {
    console.log('toolbar', toolbar);
    console.log('context', context);
}

// Called just after the controller is positioned and displayed on the screen.
// controller - editing elements displayed on the screen [image resizing, tabl
/**
 * name: The name of the plugin that called the controller
 * controllers: Array of Controller elements
 * core: Core object
*/
editor.showController = function (name, controllers, core) {
    console.log('plugin name', name);
```

```
    console.log('controller elements', controllers);
}
```

Plugins list

> The plugin and the button have the same name.

Name	Type
blockquote	**command**
image	**dialog**
link	
video	
audio	
math	
align	**submenu**
font	
fontColor	
fontSize	
formatBlock	
hiliteColor	
horizontalRule	

Name	Type
lineHeight	
list	
paragraphStyle	
table	
template	
textStyle	
imageGallery	**fileBrowser**

Examples

Examples

Options template

Options template

Custom plugins

Custom plugins

Document

Document

Other libraries using SunEditor

suneditor-react (@mkhstar) - Pure React Component for SunEditor.

angular-suneditor (@BauViso) - Angular module for the SunEditor WYSIWYG Editor.

Livewire & Alpine.JS (@kaju74) - Using SunEditor with Livewire & Alpine.JS

Plugin for Pluxml (@sudwebdesign) - Plugin for Pluxml.

AEM-SunEditor (@ahmed-musallam) - Enables using SunEditor in AEM dialogs as an RTE replacement.

License