pinEdit Manual

Browser based HTML editing

Version 3.0

www.pintexx.com
# Table of contents

1 **Introduction** ................................................................................................................... 3
   1.1 Browser Based HTML Editing ....................................................................................... 3
   1.2 Editor Functions .......................................................................................................... 3
   1.3 Key Codes .................................................................................................................. 5
   1.4 Colors ....................................................................................................................... 5
   1.5 Context Menu ........................................................................................................... 5
   1.6 Table Editing ............................................................................................................. 6
      1.6.1 Tab Key ............................................................................................................. 6
   1.7 Property dialogs ....................................................................................................... 7
   1.8 Edit Modes ............................................................................................................... 8
   1.9 Controls ................................................................................................................... 8
   1.10 RETURN Mode ....................................................................................................... 8
   1.11 Style sheets ............................................................................................................ 8
   1.12 Text modules (Java,.Net) ....................................................................................... 8
2 **Integration/Deployment** ............................................................................................... 10
3 **Using pinEdit** ............................................................................................................... 11
   3.1 Loading a document ................................................................................................. 11
   3.2 pinEdit in an IFRAME ............................................................................................. 11
   3.3 Setting parameters .................................................................................................. 11
4 **Loading and saving HTML content** ............................................................................. 13
   4.1 Reading HTML content from server ........................................................................ 13
   4.2 Saving HTML content .............................................................................................. 13
   4.3 Server extensions ..................................................................................................... 13
   4.4 Integration Mode ...................................................................................................... 14
5 **Installing Key** .............................................................................................................. 15
6 **Configuration** ............................................................................................................... 16
   6.1 Default settings ....................................................................................................... 16
      6.1.1 Language/Internationalization ......................................................................... 16
      6.1.2 Style ................................................................................................................ 16
      6.1.3 Set targets ....................................................................................................... 17
   6.2 Toolbar configuration ............................................................................................... 18
      6.2.1 Use toolbar configuration string ...................................................................... 18
      6.2.2 Use toolbar configuration script (JavaScript) .................................................... 19
      6.2.3 Menu Configuration ......................................................................................... 21
7 **Client API** ..................................................................................................................... 22
   7.1 Calling API methods ................................................................................................. 22
      7.1.1 pinEdit in frame document .............................................................................. 22
      7.1.2 pinEdit in IFRAME ......................................................................................... 22
   7.2 API .......................................................................................................................... 22
1 Introduction

To use the advantages of the web, more and more software solutions are browser based. Display of HTML content is really no problem.

But what about editing HTML content?
In diverse applications like content management systems it would be very helpful, if the user could also edit HTML pages directly in the browser.

pinEdit gives the possibility to implement HTML editing completely in a web application.

1.1 Browser Based HTML Editing

pinEdit comes with a complete, ready to use HTML editor.

1.2 Editor Functions

The following functions are available:

- Create a new document
- Open file with dialog (must be supported)
- Save file
- Save As with dialog (must be supported)
- Search and replace function
- Print file
- Spell Checker (if installed)
- Upload documents or images (must be supported)
- Cut
- Copy
- Paste
- Paste from Word (Remove Word specific formatting)
- Undo
- Redo
- Create link from selection
- Insert anchor
- Insert image
- Insert table
- Insert rule
- Insert date
- Insert date
1.3 Key Codes

The following key codes are supported:

- Control+A (select all)
- Control+C (copy)
- Control+V (paste)
- Control-Z (undo)

1.4 Colors

Use the right part of the color button ( ) to select a color with color selection dialog. The left part shows the selected color in the color bar. Format text by pressing the left part of the color button.

1.5 Context Menu

Use right mouse key to display a context menu on the following objects:

Table, Cell, Label, Page, Image, Container, Sub Page

Use the property item to edit object attributes. If an object is in absolute position mode you can use “set to foreground” to set one of two overlapping controls to foreground.
1.6 Table Editing

To edit table or cells use the table creation control and press right mouse button within a cell.

The following functions are available:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert row before</td>
<td></td>
</tr>
<tr>
<td>Insert row after</td>
<td></td>
</tr>
<tr>
<td>Delete row</td>
<td>Deletes current row</td>
</tr>
<tr>
<td>Insert column before</td>
<td></td>
</tr>
<tr>
<td>Insert column after</td>
<td></td>
</tr>
<tr>
<td>Delete column</td>
<td>Deletes current column</td>
</tr>
<tr>
<td>Delete cell</td>
<td>Deletes the current cell</td>
</tr>
<tr>
<td>Merge cell</td>
<td>Merges the current cell with the next cell</td>
</tr>
<tr>
<td>Divide cell</td>
<td>Divides a merged cell in 2 separate cells</td>
</tr>
<tr>
<td>Cell properties</td>
<td>Opens the cell properties dialog</td>
</tr>
<tr>
<td>Table properties</td>
<td>Opens the table properties dialog</td>
</tr>
</tbody>
</table>

1.6.1 Tab Key

If a table cell is selected the cursor is moved to the next table cell by pressing tab key. In the last cell a new table row is added.
1.7 Property dialogs

The following dialogs appear by pressing right mouse button:

- **Page properties**
- **Cell properties**
- **Table properties**
1.8 Edit Modes
On the bottom border you find the mode select bar.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit mode - Standard edit mode</td>
<td>Shows the HTML code. You can change the HTML code and reuse it in edit mode</td>
</tr>
<tr>
<td>HTML mode</td>
<td>Shows the HTML code. You can change the HTML code and reuse it in edit mode</td>
</tr>
<tr>
<td>Preview mode</td>
<td>Document is not editable anymore</td>
</tr>
</tbody>
</table>

1.9 Controls
If you use controls like buttons, images, etc. you can select them by clicking with the mouse. A selection frame appears. You can now drag and drop the controls.

If you want to use absolute positions, select a control and press the "absolute positions" button in the toolbar. Now you move the control by pixel.

1.10 RETURN Mode
Use the RETURN mode button in the toolbar to define, during runtime, what tag is inserted after pressing RETURN.

1.11 Style sheets
If a document contains style definitions (internal or external) the editor detects them automatically and fills the style combo box with all style rules starting with "." The styles can be assigned to each active HTML object like text, table, cell image etc.

1.12 Text modules (Java,.Net)
Text modules are small HTML pages loadable/saveable from/to server.
You can control via the adapter where the modules are located.

<table>
<thead>
<tr>
<th>Combo with all available text modules</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Inserts the text module that has been selected in the combo box at the current cursor position.</td>
<td></td>
</tr>
<tr>
<td>Creates a new text module from a selection. Select a text margin, press the button and</td>
<td></td>
</tr>
<tr>
<td>enter a name for the text module. This name is displayed in the combo box after creation.</td>
<td></td>
</tr>
<tr>
<td>Removes the selected text module.</td>
<td></td>
</tr>
</tbody>
</table>
2 Integration/Deployment

For integration in a web application the following steps has to be done.

- Copy the “editor” directory in your applications root directory
  
  o Attention: Do NOT rename the “editor” directory

Please refer to the document “pinEditStartup.pdf” for further setup instruction.
3 Using pinEdit

pinEdit is designed to work technology independent. Therefore it can work together with a lot of different technologies like Java, .Net, PHP etc.

Most of the pinEdit functions work without direct server interaction. In the standard mode, pinEdit is just a HTML page, running in a browser window or in an IFRAME within a html page.

http:// ... /editor/pinEdit.html

3.1 Loading a document

To open a file in pinEdit just call a url like:

http:// ... /pinEdit.html?url=../test.html

3.2 pinEdit in an IFRAME

If you want to integrate pinEdit within a document then use an IFRAME:

<IFRAME style="WIDTH:100%; HEIGHT: 250px" src="editor/pinEdit.html">

3.3 Setting parameters

You can call the pinEdit.html page by passing arguments like

http://.. /editor/pinEdit.html?url=test.html

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>Loading a file from startup. The url can be absolute or relative to editor directory</td>
<td>pinEdit.html?url=test.html</td>
</tr>
<tr>
<td>lng</td>
<td>EN: English DE: German ES: Spanish PT: Potuguese NL: Dutch GR: Greek (case sensitiv)</td>
<td>pinEdit.html?lng=DE</td>
</tr>
<tr>
<td>id</td>
<td>Image directory Relative to application root. Used by the open image dialog. Don’t use any &quot;/&quot; before or</td>
<td>pinEdit.html?id=images</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>dd</td>
<td>Document directory Relative to application root. Used by the open file dialog. Don’t use any “/” before or behind.</td>
<td>pinEdit.html?dd=images</td>
</tr>
<tr>
<td>tb</td>
<td>Toolbar configuration Used in conjunction with toolbar configuration string to define the toolbar.</td>
<td>pinEdit.html?tb=T010203</td>
</tr>
<tr>
<td>is</td>
<td>Intellisense (Auto popup) key. Specify any valid character</td>
<td>pinEdit.html?is=.</td>
</tr>
<tr>
<td>hm</td>
<td>HTML mode You can define, if you see the entire html code (head inclusive) in HTML mode or only the body content. 1: Standard = Body only 2: All (For Javascript editing)</td>
<td>pinEdit.html?hm=2</td>
</tr>
<tr>
<td>rm</td>
<td>Return mode Specifies which tag is inserted after pressing RETURN key 0 = &lt;P&gt; (Standard) 1 = &lt;BR&gt;</td>
<td>pinEdit.html?rm=1</td>
</tr>
<tr>
<td>me</td>
<td>Menu Turns menu on/off 0=Off 1=On</td>
<td>pinEdit.html?me=0</td>
</tr>
<tr>
<td>rp</td>
<td>Relative Path Specifies the path to which all images and links are relative</td>
<td>pinEdit.html?rp =www.myweb.com/images</td>
</tr>
<tr>
<td>uc</td>
<td>User code The use code is string that is passed to the toolbar, menu and content menu creation functions. With this code you can specify individual toolbars, menus etc.</td>
<td>pinEdit.html?uc=ADMIN</td>
</tr>
</tbody>
</table>
4 Loading and saving HTML content

To load HTML content from a database or save content to a database you need server related functions.

4.1 Reading HTML content from server

Please refer to the evaluation/samples/readandpostcontent/... directory. There are samples for different technologies.

The HTML content is filled in a textarea. During load the editor reads the text area and writes the content in the editor.

4.2 Saving HTML content

Saving the content can be done in 2 ways:

4.2.1 Use toolbar save button

If you want to use the toolbar save button the editor content is saved in background by posting the content to file.

The html content is posted in variable “html”.
The current url (if available) is posted to “url”.

The file has to be configured in editor/config.js in variable “target”.

```javascript
// set target file, where html content is posted
// enter here your file with absolute or relative path
// in relativ mode, the location must be relativ to 'editor' directory
var target = "../samples/independant/independant.jsp";
```

Please refer to

evaluation/samples/post2page

to find some samples in different technologies.

4.2.1.2 Use a save button outside the editor

Please refer to the evaluation/samples/readandpostcontent/... directory. There are samples for different technologies.

The content of the editor is posted to a file in variable “html”.

4.3 Server extentions

To open and save files or to use text modules or uploads you need server functions. Those server functions are implemented for different technologies like Java, ASP, ASP.Net, PHP.
You are FREE to implement those functions for your technology.

Please have a look at the editor/dialogs directory how to implement.

4.4 Integration Mode

You can integrate pinEdit directly in a page. After pressing right mouse button and selecting “Edit document” the document is loaded into the editor. You can now cancel and return by pressing the close button at the upper right corner or press the save button to save the content to the server file.

Please look at the integration sample.
5 Installing Key

Since version 2.1 a key file is delivered after purchasing the product.

Please copy the "pinEdit.key" file in your "editor/key" directory.
6 Configuration

pinEdit is designed for maximum flexibility. Therefore the toolbar and the menu are created by object oriented JavaScript functions. All configuration is done in editor/config directory. The advantage is that you are free to modify the toolbar completely for your own needs.

You can
- add new (own) buttons
- add new toolbar rows
- add new toolbar objects like combo or style combo
- etc.

<table>
<thead>
<tr>
<th>Script</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>editor/config/config.js</td>
<td>Global settings</td>
</tr>
<tr>
<td>editor/config/config_toolbar.js</td>
<td>Defines the complete toolbar</td>
</tr>
<tr>
<td>editor/config/config_menu.js</td>
<td>Defines the complete menu</td>
</tr>
<tr>
<td>editor/config/config_context.js</td>
<td>Defines all context menus</td>
</tr>
<tr>
<td>editor/config/international.js</td>
<td>Defines i18n resources</td>
</tr>
</tbody>
</table>

6.1 Default settings

Please open the config.js script to configure default settings:

```
// pinEdit global settings

Set the parameters to your choice and save the file.

6.1.1 Language/ Internationalization

Select the language by using a short string like “EN.”

```

// set language
var language = "EN";   // supported
   // EN = english
   // DE = german
   // ...
```

All language strings are in the file …/config/international.js. To add new languages please define a new language string like “IT” and copy one language section at the end of the file. Then translate the strings.

```
if(language=="IT") {
    if(id=='101') return "Nuovo";
}
```

6.1.2 Style

```
// select design
```
var design = "Office"; // supported
    // "" Standard like Office 2000
    // "Office" Like Office 2003
    // "Office2003" Like Office 2003
This string corresponds with the last characters of the
    // stylesheet rules in editor/design/style/pinEdit.css

6.1.3 Set targets

Save document to server
This page is used when you have configured the save button in toolbar and press this button. Then the page is called asynchronously in background (see static save mode)

    // set target file, where html content is posted in "html" parameter
    // enter here your file with absolute or relative path
    // in relativ mode, the location must be relativ to 'editor' directory
Java:
    var target = "../servlet/Edit";

PHP(Others):
    var target = "save.php";

Text modules
    // set target file, that is called to load or save text modules
    // enter here your file with absolute or relative path
    // in relativ mode, the location must be relativ to 'editor' directory
Java (use the servlet and Java adapter):
    var targetTextModule = "../servlet/Edit";

Intellisense (Auto popup)
    // set target file, that is called to get intellisense data
    // enter here your file with absolute or relative path
    // in relativ mode, the location must be relativ to 'editor' directory
Java (use the servlet and Java adapter):
    var targetIntellisense = "../servlet/Edit";
6.2 Toolbar configuration

From 2.5 on the toolbar can be configured by using a toolbar configuration string or a script file.

Use the toolbar string as Default mode. If no < toolbar > string is specified the script is used automatically. Use the script file if you want to configure the toolbar in a very special way or if you want to create OWN toolbar buttons.

You can define, which buttons shall be available and at which position on which toolbar.

6.2.1 Use toolbar configuration string

The toolbar configuration string defines which toolbars are used and which buttons are displayed at which position. For each button a number is specified. The number must have 2 characters.

You can define, which buttons shall be available and at which position on which toolbar.

The toolbar configuration string defines which toolbars are used and which buttons are displayed at which position. For each button a number is specified. The number must have 2 characters.

```plaintext
// NEW       1
// OPEN      2
// SAVE      3
// SEARCH    4
// PRINT     5
// SPELL     64
// CUT       6
// COPY      7
// PASTE     8
// PASTEWORD 61
// UNDO      9
// REDO      10
// LINK      11
// ANCHOR    60
// IMAGE     12
// TABLE     13
// RULE      14
// CHAR      15
// DATE      16
// TIME      17
// MARQUEE   62
// PAGEBREAK 57
// ZOOM      18
// REMOVE    37
// SELALL    58
// STYLE     19
// FORMAT    56
// FONT      20
// FONTSIZE  21
// BOLD      22
// ITALIC    23
// UNDERLINE 24
// SUPER     25
// SUB       26
// LEFT      27
// CENTER    28
```
To define the string use T for top toolbar and B for bottom toolbar. Separate each toolbar with a semicolon (;).

Sample:

This is the complete toolbar string with all buttons:

```
T0102036504056466SE06070861SE0910SE111213141516176276SE99;T195620212223242526SE27282930SE313233343536373863SE4647SE48SE49505152SE3768695818B535455
```

6.2.2 Use toolbar configuration script (JavaScript)

Use the toolbar configuration script in editor/config/config_toolbar.js. The top and bottom toolbar creation, here is an example:

```
// add a standard button
objToolbar1.add(new Button("","","design/image/link.gif","editLink()",getLanguageString(language,109),design,""));
```
// add the image selector, uses several lines (section)
var objImage = new MenuButton("menu_image.html","design/image/image.gif","onMenuImageClicked",getLanguageString(language,110),design,"" );
objImage.width = "150";
objImage.height = "72";
objToolbar1.add(objImage);

// add the table selector
objToolbar1.add(new TableButton("design/image/table.gif",".onCreateTable",getLanguageString(language,111),design,"" ));

If you don't want to use any of those buttons, uncomment the code, where the "add" is done.
In JavaScript you uncomment with:

// Uncomment single line
or
/*
   Uncomment block
*/

6.2.2.1 Position Toolbar Buttons
If you want to change the position of a button within the toolbar then move the line or section to the
position you like.
If you want to move the button to another toolbar then move the line or section to the position you like
and change the toolbar object.

// button in toolbar 1
objToolbar1.add(new Button("","","design/image/link.gif","editLink() …

// button now in toolbar 2
objToolbar2.add(new Button("","","design/image/link.gif","editLink() …

6.2.2.2 Configure combo content
pinEdit uses combos for font, font size and zoom values.
You can change these settings by modifying these section in config_toolbar.js

   // now add the fonts
cmbFont.add("Arial","Arial");
cmbFont.add("Courier","Courier");
cmbFont.add("Tahoma","Tahoma");
cmbFont.add("Times New Roman","Times New Roman");
cmbFont.add("Verdana","Verdana");
… add your font here

   // now add the font size
cmbFontSize.add("8","1");
cmbFontSize.add("10","2");
cmbFontSize.add("12","3");
6.2.3 Menu Configuration

To define the menu, please modify the script in Editor/config/config_menu.js.

Example:

```javascript
objMenuBar = new MenuBar();

objMenuBarItem = new MenuBarItem(getLanguageString(language,3018));

objMenu = new Menu("onMenuItemClick");
objMenu.add(new MenuItem(getLanguageString(language,101) ,"design/image/" + design + "/neu.gif","","NEW"));

objMenuBarItem.setMenu(objMenu);
objMenuBar.add(objMenuBarItem);

(Currently sub menus are not supported.)
```
7 Client API

pinEdit provides a client API for use in JavaScript for all modes. Only the dialog functions are available in extended mode.

7.1 Calling API methods

7.1.1 pinEdit in frame document

If you have a frame document named “fraEditor” then call the API functions with:

```javascript
top.fraEditor.<API function>
```

7.1.2 pinEdit in IFRAME

If you have in IFRAME on your page called “editorFrame” with pinEdit loaded you can access the API functions with:

```javascript
document.getElementById("editorFrame").contentWindow.<API method>
```

Sample (read HTML content)

```javascript
var editor = document.getElementById("editorFrame").contentWindow;
var html   = editor.editGetHtml();
```

7.2 API

The following methods are available:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>editWrite(html)</td>
<td>Creates a new document and writes HTML</td>
</tr>
<tr>
<td>editLoad(URL)</td>
<td>Load file described with “URL”</td>
</tr>
<tr>
<td>editSave()</td>
<td>Posts the HTML content asynchronously to the page defined in config.js</td>
</tr>
<tr>
<td>editSaveDialog()</td>
<td>Save As</td>
</tr>
<tr>
<td>editPrint()</td>
<td>Prints the page</td>
</tr>
<tr>
<td>editGetHtml()</td>
<td>Reads the HTML content</td>
</tr>
<tr>
<td>editGetBodyHtml()</td>
<td>Reads the HTML body content</td>
</tr>
<tr>
<td>editNew()</td>
<td>Creates a new empty document</td>
</tr>
<tr>
<td>editCut()</td>
<td>Cuts the selected html code</td>
</tr>
<tr>
<td>editCopy()</td>
<td>Copies the selected html code</td>
</tr>
<tr>
<td>editPaste()</td>
<td>Pastes the selected html code</td>
</tr>
<tr>
<td>editUndo()</td>
<td>Undo function</td>
</tr>
<tr>
<td>editRedo()</td>
<td>Redo function</td>
</tr>
<tr>
<td>editLink()</td>
<td>Inserts a link</td>
</tr>
<tr>
<td>editSuperscript()</td>
<td>Superscript</td>
</tr>
<tr>
<td>editSubscript()</td>
<td>Subscript</td>
</tr>
<tr>
<td>editBold()</td>
<td>Format bold</td>
</tr>
<tr>
<td>editItalic()</td>
<td>Format italic</td>
</tr>
<tr>
<td>editUnderline()</td>
<td>Format underline</td>
</tr>
<tr>
<td>editIndent()</td>
<td>Indent</td>
</tr>
<tr>
<td>editOutdent()</td>
<td>Outdent</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td><code>editJustifyLeft()</code></td>
<td>Justify left</td>
</tr>
<tr>
<td><code>editJustifyCenter()</code></td>
<td>Justify center</td>
</tr>
<tr>
<td><code>editJustifyRight()</code></td>
<td>Justify right</td>
</tr>
<tr>
<td><code>editJustifyFull()</code></td>
<td>Justify full</td>
</tr>
<tr>
<td><code>editOrderedList()</code></td>
<td>Ordered list</td>
</tr>
<tr>
<td><code>editUnorderedList()</code></td>
<td>Unordered list</td>
</tr>
<tr>
<td><code>editRemoveFormat()</code></td>
<td>Remove format</td>
</tr>
<tr>
<td><code>editBackColor(color)</code></td>
<td>Set back color</td>
</tr>
<tr>
<td><code>editColor(color)</code></td>
<td>Set fore color</td>
</tr>
<tr>
<td><code>editGetText()</code></td>
<td>Get text only</td>
</tr>
<tr>
<td><code>editSetText(text)</code></td>
<td>Set text</td>
</tr>
<tr>
<td><code>editInsertHtml(html)</code></td>
<td>Insert HTML</td>
</tr>
<tr>
<td><code>editInsertText(text)</code></td>
<td>Inserts text</td>
</tr>
<tr>
<td><code>editGetSelectedHtml()</code></td>
<td>Get selected HTML</td>
</tr>
<tr>
<td><code>editGetSelectedText()</code></td>
<td>Get selected text</td>
</tr>
<tr>
<td><code>editAbsolute()</code></td>
<td>Set to absolute mode</td>
</tr>
<tr>
<td><code>editZoom(value)</code></td>
<td>Set zoom value in %</td>
</tr>
<tr>
<td><code>editInsertObject(obj)</code></td>
<td>Insert an object :</td>
</tr>
<tr>
<td></td>
<td>'FORM'</td>
</tr>
<tr>
<td></td>
<td>'LABEL'</td>
</tr>
<tr>
<td></td>
<td>'BUTTON'</td>
</tr>
<tr>
<td></td>
<td>'INPUT'</td>
</tr>
<tr>
<td></td>
<td>'LISTBOX'</td>
</tr>
<tr>
<td></td>
<td>'COMBO'</td>
</tr>
<tr>
<td></td>
<td>'CHECK'</td>
</tr>
<tr>
<td></td>
<td>'OPTION'</td>
</tr>
<tr>
<td></td>
<td>'AREA'</td>
</tr>
<tr>
<td></td>
<td>'TABLE'</td>
</tr>
<tr>
<td></td>
<td>'IMAGE'</td>
</tr>
<tr>
<td></td>
<td>'DIV'</td>
</tr>
<tr>
<td></td>
<td>'HIDDEN'</td>
</tr>
<tr>
<td></td>
<td>'MARQUEE'</td>
</tr>
<tr>
<td></td>
<td>'IFRAME'</td>
</tr>
<tr>
<td></td>
<td>'RULE'</td>
</tr>
<tr>
<td><code>setMode(mode)</code></td>
<td>Mode :</td>
</tr>
<tr>
<td></td>
<td>&quot;EDIT&quot;: Standard edit mode</td>
</tr>
<tr>
<td></td>
<td>&quot;HTML&quot;: HTML source mode</td>
</tr>
<tr>
<td></td>
<td>&quot;PREVIEW&quot;: Preview mode</td>
</tr>
<tr>
<td><strong>New in 3.0:</strong></td>
<td></td>
</tr>
<tr>
<td><code>editGetDocument()</code></td>
<td>Gets the editor document</td>
</tr>
<tr>
<td><code>editMenuVisible(value)</code></td>
<td>Enables/Disables menu</td>
</tr>
<tr>
<td><code>editTableHighlight()</code></td>
<td>Toggles Table highlight</td>
</tr>
<tr>
<td><code>editSetDirection(mode)</code></td>
<td>Set the write direction of current object</td>
</tr>
<tr>
<td></td>
<td>Mode:</td>
</tr>
<tr>
<td></td>
<td>&quot;RTL&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;LTR&quot;</td>
</tr>
<tr>
<td><code>editPreview()</code></td>
<td>Opens a new browser window with current document</td>
</tr>
<tr>
<td><code>editSetSaveMode(mode)</code></td>
<td>Sets the “Save” mode, that means which HTML content is transferred to target URL. The mode value is specified in config.js</td>
</tr>
<tr>
<td><code>editSetBr(value)</code></td>
<td>Sets the RETURN mode</td>
</tr>
<tr>
<td></td>
<td>False = P</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>editGetHtmlRel()</td>
<td>Reads HTML content with relative images and links</td>
</tr>
<tr>
<td>editGetBodyHtmlRel()</td>
<td>Reads HTML body content with relative images and links</td>
</tr>
<tr>
<td>editSelectAll()</td>
<td>Selects all</td>
</tr>
<tr>
<td>editUpload(dir)</td>
<td>Displays upload dialog.</td>
</tr>
<tr>
<td></td>
<td>Dir = path relative to web root</td>
</tr>
<tr>
<td></td>
<td>I.e. “images/upload”</td>
</tr>
<tr>
<td>editSetStyle(style)</td>
<td>Assigns the style to current object</td>
</tr>
<tr>
<td></td>
<td>Style = style name i.e. “font1”</td>
</tr>
<tr>
<td>editGetActiveObject()</td>
<td>Returns the active object:</td>
</tr>
<tr>
<td></td>
<td>Use</td>
</tr>
<tr>
<td></td>
<td>editGetActiveObject().tagName</td>
</tr>
<tr>
<td></td>
<td>to get the current tag name.</td>
</tr>
</tbody>
</table>