# HTML Editor for Winforms 2.5.3.305

Zoople HTML Editor for Winforms is an HTML editor for .NET Winforms and WPF. DotNET framework 4.0 thru 4.8, .NET5 and .NET6

https://zoople.tech


## Features

The HTML Editor contains all the standard features found in stand-alone WYSIWYG HTML and RichText editors.

For example :

- Text modifications – Bold, Italicize, strike-thru, justification, font and color settings, indenting out denting
- HTML Tables – Can be inserted and modified, and the properties of rows and cells modified
- Formatted Source Code Editing – More advanced end users may edit the source code of the document.
- Cascading Style Sheets – HTML Elements may be assigned CSS class using a popup menu that displays all the CSS classes associated with a document.
- DOM Toolbar – The editor features a HTML DOM Toolbar that displays and allows selection and modification of any HTML node in a branch of the current document
- Property Grid - allowing for the direct editing of HTML element properties
- Access to the HTML DOM via the Document property - allows for iteration and manipulation of elements programmatically
- Fully customisable toolbars and context menus
- Inline Spelling (available as a separate license) which supports "spell as you type", error highlighting, context menu spelling suggestions, and spell checking dialog.

# Installation

**Download installation package and sample projects:** Zoople_HTMLEditor_setup.exe

**nuget:** Install-Package Zoople.HTMLEditor -Version 2.5.3.305

**Sample Projects:** Sample Projects - Github

# Getting started in Visual Studio

Either install via NuGet (as above) or right click on the VS toolbox and click "Choose Items", click browse and navigate to the folder where the control was installed, select "HTMLEditor.dll" and click OK. You can now add the editor to a Windows Form. Alternatively install from NuGet using the NuGet command above or via the NuGet GUI in VS.

# Support

Support requests and other queries can be made via email to info@zoople.tech

# API

A brief overview of the editor's API.

## Properties

### AcceptsReturn

- returns boolean

Aligns with standard TextBox behaviour for Winforms with an AcceptButton defined. Please note setting this to false disables the enter key press creating a new div, p, or br tag in the WYSIWYG window

### CurrentElement

- returns IHTMLElement (Read Only) (COM)

The current selected IHTMLElement in the editable area of the control.

## CurrentWindowsFormsElement

- returns HTMLElement (Read Only)

The current selected managed HTMLElement in the editable area of the control.

## UseRelativeURLs

- Boolean

Enables the control to use URL references that are relative to the BaseURL property.

## AllowDrop (Also UserControl Instrinc Property)

- Boolean

When true allows for the dropping of external images and Text content into the WYSIWYG area of the control. Use the CancellableUserInteraction event with the InteractionType "onexternaldrop" to manage the drop event and data.

## AllowDragInternal

- Boolean

enables disables dragging and dropping of HTML elements in the editors DOM.

## BaseURL

- String

Returns or sets the base url for the document being editted. All relative paths for images and links placed into the HTML document will use the BaseURL property to resolve the URL

## ContextMenuWYSIWYG

- ContextMenuStrip (READ ONLY)

allows for modification / addition / removal of context menu strip items in the control

## CustomColorPalette

- List Of(Color)

Allows for displaying a application defined set of colors in the main toolbar color dropdown buttons

## UseParagraphAsDefault

- Boolean

Use P tag as default seperator (Enter Key) otherwise use DIV

## ImageStorageLocation

- String

Folder to store local images. All images inserted into the editor will be copied to this folder.

## CurrentSelection

- Object

contains Selection Type, HTML Text and Plain Text string values of current selection

```
Public Structure Selection
    Public Type As String
    Public htmlText As String
    Public Text As String
    Public Items As List(Of HTMLElement)
    Public ForeColor As String
    Public BackColor As String
    Public FontName As String
End Structure
```

## CSSText

- String

Returns or sets the Cascading Style Style sheet information for the document being edited. The CSSText property will accept a well formed CSS text string. Class selectors, both global and element specific will appear in the DOM toolbar CSS styles list. Element specific selectors will only be applicable to <p> tags and global selectors to all elements (.test1 - as above). The CSSText value is not persisted in the DocumentHTML property of the control so you must include it with your email template, page snippet, final email etc before using it in you application.

## InlineCSS

- List (Of String)

A list of inline CSS styles to the inline CSS design and applicator form

For example;

```
this.htmled.InlineCSS.Add("background-color: green; color: white;");
this.htmled.InlineCSS.Add("background-color: blue; color: #236512;");
```

## LanguageFile

- String

Get and sets the Language localisation filename being used. Alternatively, a string value containg the XML contents of a language file may be used.

## LicenceKey

- String

License Code for the HTML Editor.

## LicenceKeyActivation

- String

License activation code for the HTML Editor.

## LicenceKeyInlineSpelling

- String

License Code for the HTML Editor's optional multi-lingual spelling add-on.

## DefaultSpellingLanguage

- String

Language code for the language to be used by the inline spelling add-on. Must be in Language Code format and available on the end-user's system as an installed language.

## EnableInlineSpelling

- Boolean

Sets or Returns a boolean value indicating whether the optional multi-lingual spelling add-on is enabled.

## FontsList

- String

Semi-colon delimted list of Fonts available to the end-user in WYSIWYG editing mode. Passing null defaults to system fonts.

## FontSizesList

- String

Sets or Returns a string value containing a semi-colon seperated list if font sizes available to be selected from the FontSize toolbar combo. Valid values include px/pt/em/rem values.

For example;

```
this.htmlEditor.FontSizesList = "10pt;12pt;16pt;1em;2em;3em";
```

## InCodeView

- Boolean

Returns a boolean value indicating whether the editor in code view mode or WYSIWYG mode.

## DocumentHTML

- String

A string value for the HTML segement currently being edited in the control.

## Document

- HTMLDocument (Reads-only)

HTMLDocument Interface (Managed) for the current edited HTML segment.

## HiddenButtons

- String

A semi-colon seperated list of button names. When included in the list the named toolbar buttons are invisible.

```
tsbShowCode|ShowCode
tsbShowCode|ShowEditor
tsbPrint
tsbCut
tsbCopy
tsbPaste
tsbUndo
tsbRedo
tsbBold
tsbItalic
tsbUnderline
tsbStrikeout
tsbSuperscript
tsbSubscript
tsbRemoveFormatting
tsbAlignLeft
tsbAlignCentre
tsbAlignRight
tsbAlignJustify
tsbOrderedList
tsbUnorderedList
tsbOutdent
tsbIndent
tsbInsertLink
tsbRemoveLink
InsertImageToolStripButton
TableOptionsToolStripMenuItem
tsbForeColor
tsbBackColor
FontToolStripComboBox
FontSizeComboBox
FormatSelectionCombo
tsbElementProperties
tsbInsertSymbol
```

## UIControlsBackgroundColor

- Color

sets the background color of the UI - except the edit window which can controlled by redefining the background color of the BODY using the CSSText property

## ToolStripItems

- ToolStripItemCollection

enables programmatic access to the tool strip items in the upper toolbar of the control.

## ToolstripImageScalingSize

- Size

## ShowPropertyGrid

- Boolean

Show or hide the Property Grid for HTML elements.

## HideMainToolbar

- Boolean

Show or hide the Main (upper) Toolbar.

## HideDOMToolbar

- Boolean

Show or hide the DOM (lower) Toolbar.

## AlwaysConvertNonASCIIChars

- Boolean

Forces the control to always convert non-ASCII characters to their HTML entity equivelents.

## EditorEnabled

- Boolean

Disables/enables editing abilities in the control - combine with HideMainToolbar and HideDOMToolbar properties

### SpellingAutoCorrectionList

- HashTable

Requires EnableInlineSpelling = true

A hash table of acronyms and replacements. The acronym being the key and a “long-form” string being the associated value.

```
    Me.HtmlEditControl1.SpellingAutoCorrectionList.Add("pp", "<span style='color
    Me.HtmlEditControl1.SpellingAutoCorrectionList.Add("ff", "<span style='color
    Me.HtmlEditControl1.SpellingAutoCorrectionList.Add("cw", "&copy;")
```

### Zoom

- integer

Sets (.NET Framework only) and returns (.Net framework and .Net 5) the optical Zoom level in % of the WYSIWYG area of the control

## Methods

### cut_document()

- returns void

Cuts the active selection in code view or WYSIWYG view to the clipboard.

### copy_document()

- returns void

Copies the active selection in code view or WYSIWYG view to the clipboard.

### paste_document()

- returns void

Pastes the clipboard contents to active selection in code view or WYSIWYG view.

## InsertAtCursor(string HTMLText, ed_InsertType insertWhere)

- returns void

Inserts HTML at the current cursor or selection position.

```
- ed_InsertAfterSelection – insert after the current selection
- ed_InsertBeforeSelection - insert before the current selection
- ed_InsertReplaceSelection – replace the current selection.
```

## InsertImage(string Filename)

- returns HTMLElement

Inserts an image into the document, and copies it to the ImageStoreLocation if specified. Method returns the final storage location of file

## SetDirty(boolean ForceUpdate)

- returns void

sets the content dirty flag to true (ForceUpdate = true - forces an immediate refresh on databinding sources)

## IterateTextForSpellChecking()

- returns void

Causes the control to iterate through the text content of the edited doument word by word. Use the SpellCheckWord event to perform spelling checks on indivual words using a any 3rd party spell checking library (nHunSpell, NetSpell, Phonix etc).

## bold_document()

- returns void

Adds or removes bolding from the current text selection in WYSIWYG view.

## Italic_document()

- returns void

Adds or removes italics from the current text selection in WYSIWYG view.

## Underline_document()

- returns void

Adds or removes underlines from the current text selection in WYSIWYG view.

## setForeColorOnSelection(newColor as Color)

- returns void

Sets the foreground color on the currently selected text

## setBackgroundColorOnSelection(newColor as Color)

- returns void

Sets the background color on the currently selected text

## Strikeout_document()

- returns void

Adds or removes strikeout from the current text selection in WYSIWYG view.

## Superscript_document()

- returns void

Adds or removes superscript from the current text selection in WYSIWYG view.

## Subscript_document()

- returns void

Adds or removes subscript from the current text selection in WYSIWYG view.

## Indent_document()

- returns void

indents the current selection in WYSIWYG view.

## Outdent_document()

- returns void

Outdents the current selection in WYSIWYG view.

## Justify_Left()

- returns void

justifies left the current selection in WYSIWYG view.

## Justify_Right()

- returns void

justifies right the current selection in WYSIWYG view.

## Justify_Center()

- returns void

justifies center the current selection in WYSIWYG view.

## Justify_Full()

- returns void

justifies full width the current selection in WYSIWYG view.

## Remove_Formatting()

- returns void

removes markup specific formatting from the current selection in WYSIWYG view.

## Insert_Link()

- returns void

Opens the insert link dialog in WYSIWYG view.

## Remove_Link()

- returns void

removes hyperlink markup from the current selection in WYSIWYG view.

## undo_document()

- returns void

undoes the most recent change.

## redo_document()

- returns void

redoes the most recent change.

## numbered_list()

- returns void

Creates a ordered list ( <OL> ) containing the current selection if there is one.

## unnumbered_list()

- returns void

Creates a un-ordered list ( <UL> ) containing the current selection if there is one.

## print_Document(boolean DontShowDialog)

- returns void

Opens the standard windows print dialog allowing the currently edited document To be printed.

## setFontNameOnSelection(string FontName)

- returns void

sets the font face to be used on the current selection in WYSIWYG view.

## setFontSizeOnSelection(string FontName)

- returns void

sets the font size (1..7) to be used on the current selection in WYSIWYG view.

## InsertHTMLELement(string TagName)

- returns HtmlElement

Inserts an HTML element at the caret location or over the current selection of the type specified in tagName (e.g p, img, div etc)

## GetItemsByClassName(string ClassName)

- returns List Of HtmlElement

returns a list of HTMLElements where the Class attribute matches ClassName

## GetItemsByAttributeValue(string AttributeName, string AttributeValue)

- returns List Of HtmlElement

returns a list of HTMLElements where the Attribute defined by AttributeName matches AttributeValue

## GetItemsByTagName(string TagName)

- returns List Of HtmlElement

returns a list of HTMLElements where the tag name matches TagName

## GetElementsByClassName(string Classname)

- returns a List of HTMLElement

returns a list of HTMLElements where the class attribute name matches Classname

## ChangeElementStyle(HtmlElement Element, string Attribute, string Value)

- returns void

Add or updates a property to an elements CSS Style attribute. Passing null removes the attribute if it is already present. Attribute name is the CSS attribute (e.g. background color), 'Value' is attribute value (e.g. #FF0000)

## SpellCheckDocument(bool ShowDialog)

- returns void

Performs a spell check of the text content in the HTML editor, highlighting any spelling errors found.

### ShowFindTextDialog()

- returns void

Shows a "Find Text" dialog in both WYSIWYG and Code View modes

## Events

### HTMLChanged

- HTMLChanged(Object sender, eventargs e)

Fires when the content in the control is modified by the user

### AfterChangeEditingViews

- AfterChangeEditingViews(sender As Object, e As EventArgs)

Fires after changing to and from codeview to WYSIWYG editing modes

e.InCodeEditView (boolean) - indicates whether the control is in code editing view

### BeforeGetDocumentHTML

- BeforeGetDocumentHTML(object sender, BeforeGetDocumentHTMLEventsArgs e)

Raised before the retriveal of the DocumentHTML property. Modify BeforeGetDocumentHTMLEventsArgs htmlSource to change the value of DocumentHTML being retrieved

### CommandsToolbarButtonClicked

- CommandsToolbarButtonClicked(object sender, CommandsToolbarButtonClickedEventsArgs e)

Used to override the deault actions of the built in toolbar buttons.

e.ButtonIdentifier = Unique ID of the button

e.Cancelled = set to true to implement your own code

### CancellableUserInteraction

- CancellableUserInteraction(sender Object, CancellableUserInteractionEventArgs e)

Used to override the deault actions of the built in user interactions like keydown, mousedown, drag drop etc.

InteractionType - string - type of user interaction ("ondrop", "onkeydown", "onkeypress", "onmousedown", "onmouseup", "ondblclick", "onbeforecopy", "onbeforecut", "onbeforepaste", "ondrop", "ondrag", "onexternaldrop", "onmousewheel")

```
or use *EditorUIEvents* enumerations
```

e.Cancel - boolean - set to true to implement your own code or prevent the default action

e.EditorKeys - structure as describe below - argument to describe keyboard events as follows

```
Public Structure EditorKeys
    Public Keycode As Integer
    Public Control As Boolean
    Public Alt As Boolean
    Public Shift As Boolean
End Structure
```

## ConfigurationError

- ConfigurationError(object sender, ConfigurationErrorEventArgs e)

## DocumentLoadComplete

- DocumentLoadComplete(object sender, EventArgs e)

Fires when the editor's HTML DOM creation is complete and accessible

## OperationError

- OperationError(object sender, OperationErrorEventArgs e)

## SpellCheckWord

- SpellCheckWord(object sender, SpellCheckWordEventArgs e)

Used to implement spell checking on individual words. If the value of e.NewWord is changed the existing word in the document (e.WordToCheck) is replaced. Set to an emtpy string to keep the old word.

## SpellCheckComplete

- SpellCheckComplete(object sender, EventArgs e)

Fires when the text iteration completes

## CurrentSelectionInfoUpdated

- CurrentSelectionInfoUpdated(object sender, SelectionInfoUpdatedEventArgs e)

summarizes informatrion regarding current selection formatting information for custom toolbars and menus

## UserInteraction

- UserInteraction(sender Object, CancellableUserInteractionEventArgs e)

InteractionType - string - type of user interaction ("ondrop", "onkeydown", "onkeypress", "onmousedown", "onmouseup", "ondblclick", "onbeforecopy", "onbeforecut", "onbeforepaste")

```
or use *EditorUIEvents* enumerations
```

e.EditorKeys - structure as describe below - argument to describe keyboard events as follows

```
Public Structure EditorKeys
    Public Keycode As Integer
    Public Control As Boolean
    Public Alt As Boolean
    Public Shift As Boolean
End Structure
```

## ZoomLevelChanged

- ZoomLevelChanged(sender Object, EventArgs e)

Fires when the Zoom level changes