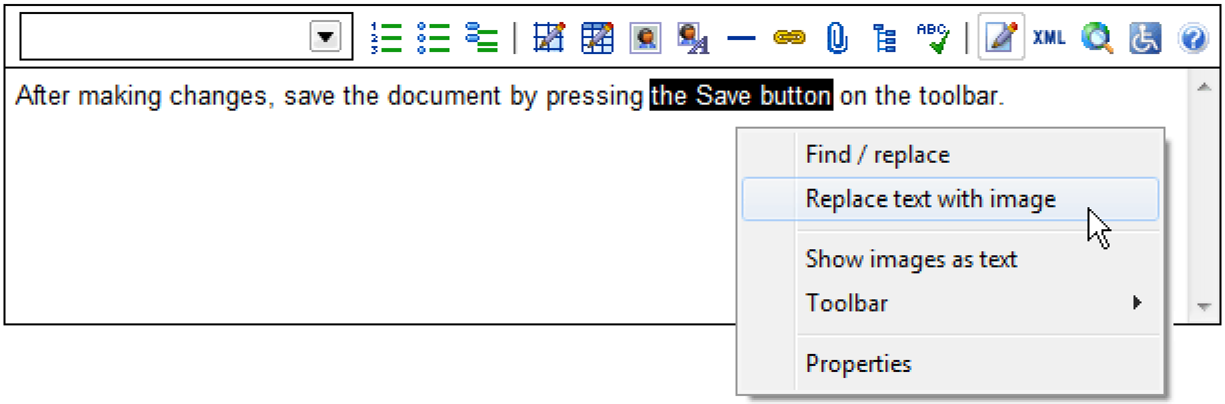
New features in XStandard version 3.0

Premier feature: "Images As Text"

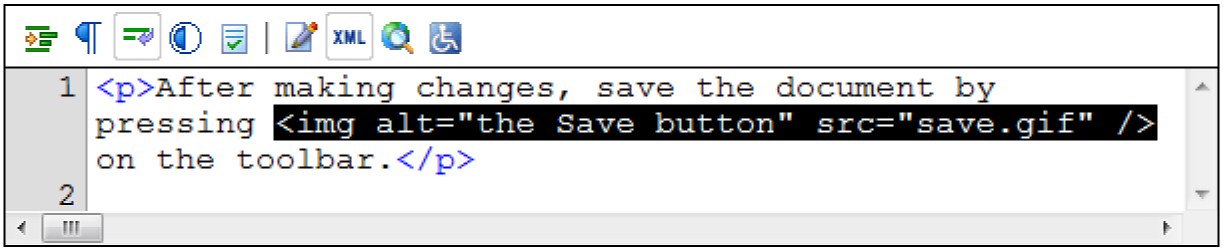
Images As Text makes it clear what the function of alternate text is, and makes authoring appropriate alternate text easy

In the following foolproof method of writing alternate text, the author highlights text in the document, then selects "Replace text with image" from the context menu or presses on the toolbar.



After making changes, save the document by pressing the Save button on the toolbar.

The image selected by the author from the image library replaces the highlighted text, which now becomes the alternate text for the image.

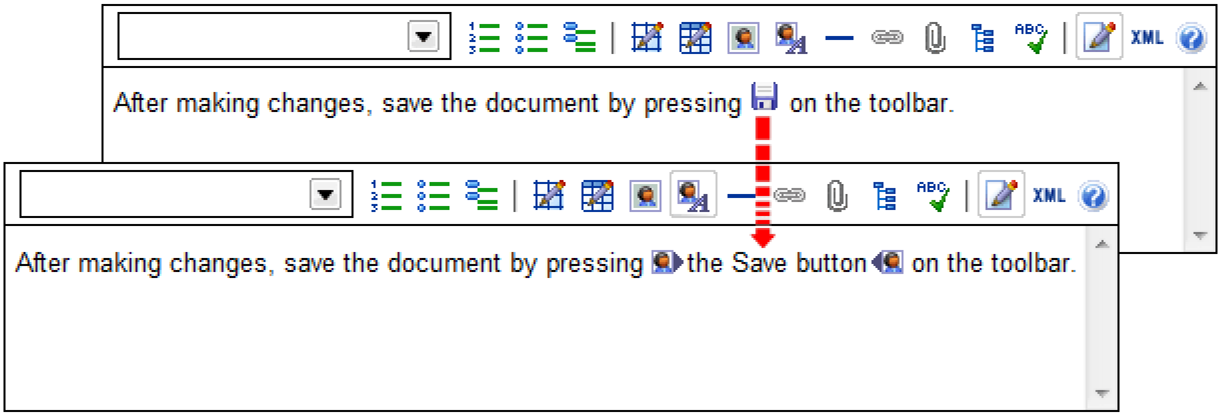


```
<p>After making changes, save the document by
pressing <img alt="the Save button" src="save.gif" />
on the toolbar.</p>
```

When the document is read as text only (by search engines, assistive technologies, browsers with image rendering turned off, etc.), the alternate text will read perfectly within the surrounding content.

Images As Text makes editing alternate text equally simple

Images As Text allows alternate text to be edited right in the document. In the following screen shot, the author selects "Show images as text" from the context menu or presses on the toolbar, in order to reveal the current alternate text. The current alternate text appears in the document between image markers and can be edited directly in the document, just like surrounding content.

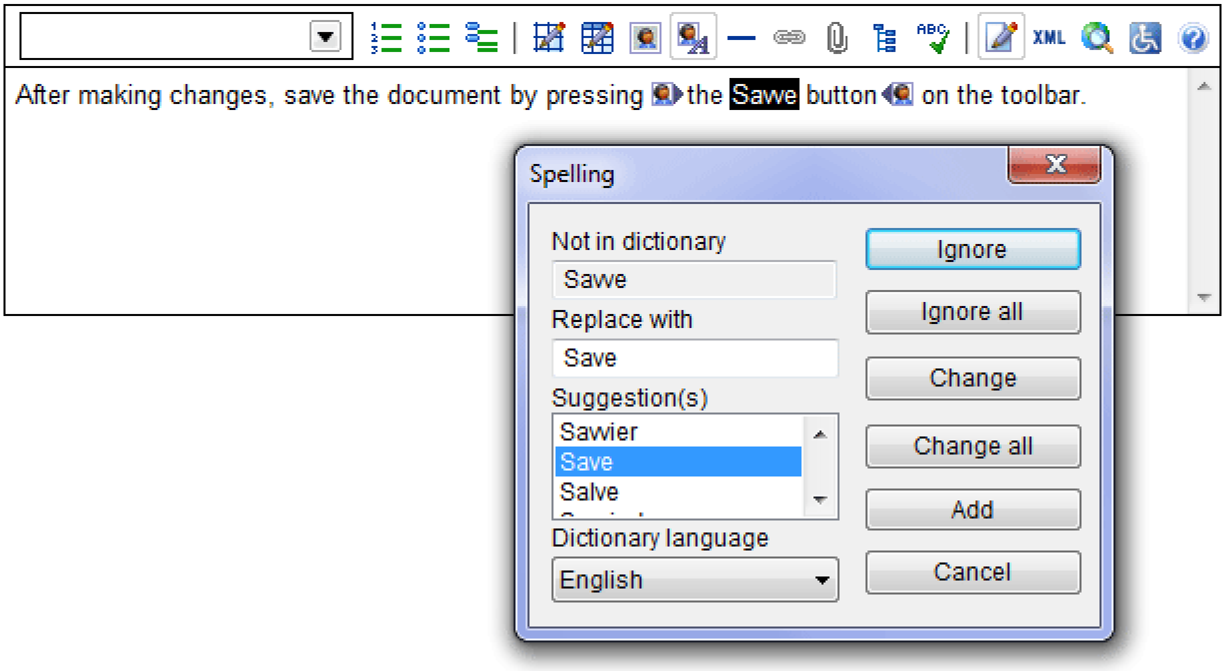


After making changes, save the document by pressing on the toolbar.

After making changes, save the document by pressing the Save button on the toolbar.

Images As Text exposes alternate text to find/replace and spell checking

When performing editing operations such as find/replace or spell checking, the editor automatically switches to Images As Text mode. This includes alternate text in find/replace and spell checking operations.



After making changes, save the document by pressing the Sawe button on the toolbar.

Spelling

Not in dictionary: Sawe

Replace with: Save

Suggestion(s): Sawier, Save, Salve

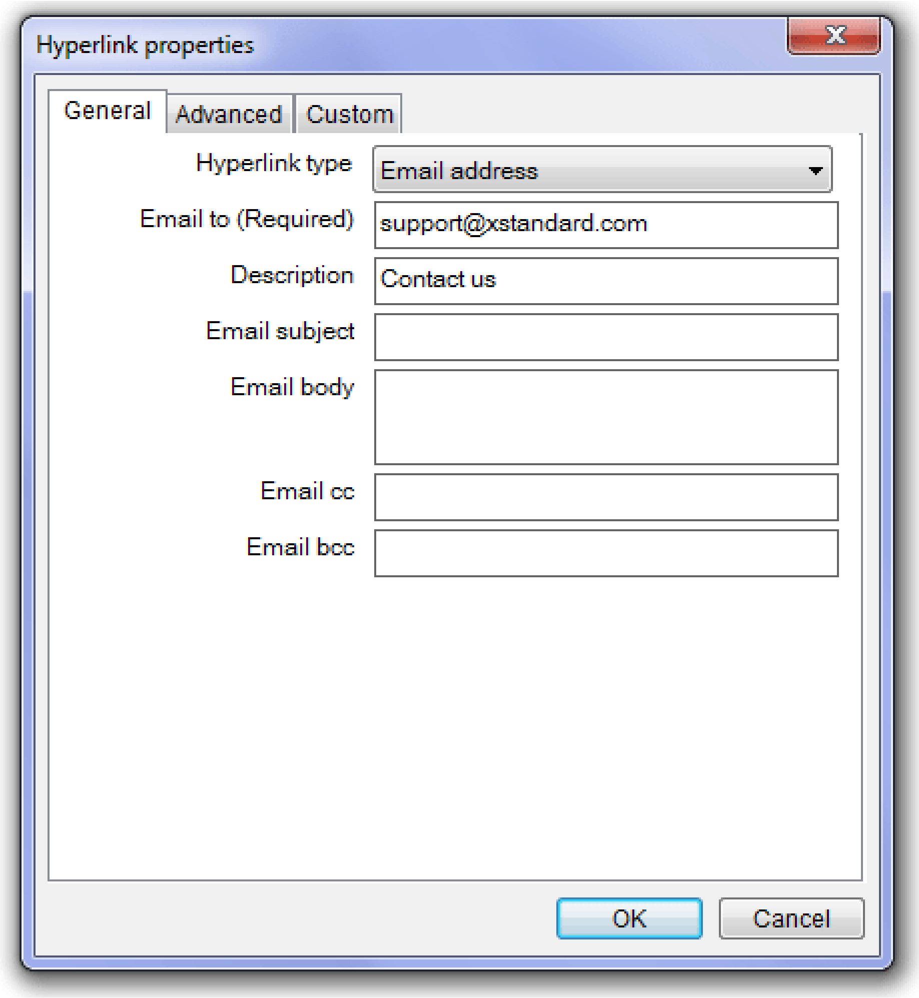
Dictionary language: English

Benefits of Images As Text

- The Images As Text feature significantly reduces the skill required to author appropriate alternate text. Composing and editing alternate text directly in the document, rather than in a pop-up dialog box, is easy and produces better results. It makes it clear what alternate text will work and what won't.
- Some authors struggle to understand what alternate text is. This method of authoring alternate text "in-context" makes it immediately evident to authors what the function of alternate text is: to replace images with text that reads coherently within the content surrounding images.
- Content authors now experience and appreciate alternate text as a living part of the document, not as remote, abstracted background information that is only vaguely related to an image.
- Alternate text now fits perfectly into the flow of the document, ensuring the document makes sense when read as text only, or with images.
- Alternate text is for the first time exposed to processing by popular editing features such as find/replace and spell checking, which improves still further the quality of alternate text.

An interface for creating email and bookmark hyperlinks

The following new interface allows the creation of email hyperlinks.



Hyperlink properties

Hyperlink type: Email address

Email to (Required): support@xstandard.com

Description: Contact us

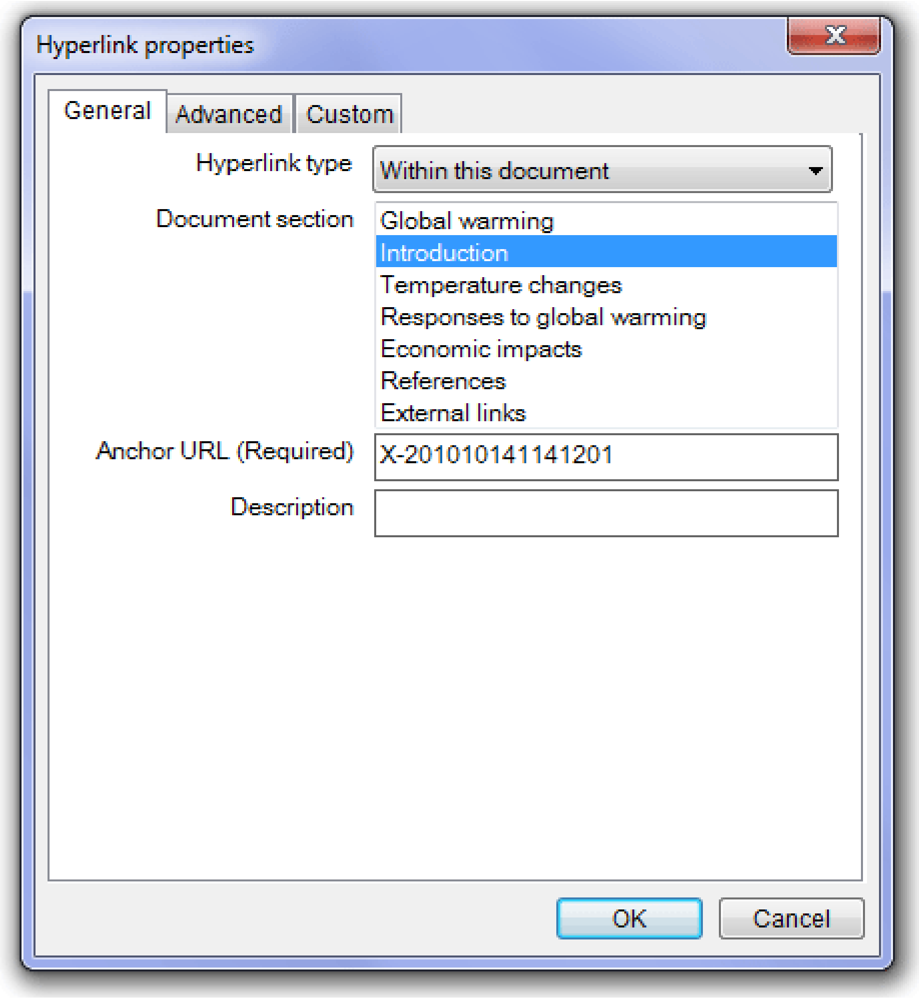
Email subject

Email body

Email cc

Email bcc

The following new interface allows the creation of "bookmarks" (links within the current document). The editor automatically turns headings (`h1` to `h6`) into anchors.



Hyperlink type: Within this document

Document section:
- Global warming
- Introduction
- Temperature changes
- Responses to global warming
- Economic impacts
- References
- External links

Anchor URL (Required): X-201010141141201

Description:

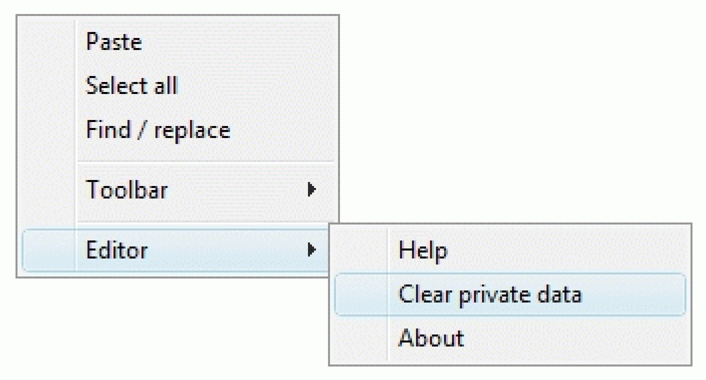
More programmatic API functions

Sub *CallProperties(sQPath As String)*

(Available in XStandard Pro)
Programmatically bring up the Properties dialog box for a given element. For example: `.CallProperties("/body[1]/h2[1]")`

Sub *ClearCache()*

Programmatically removes cached configuration files. This method is used in conjunction with `EnableCache` property. It is equivalent to manually selecting `Editor > Clear private data` from the context menu.



Function *GetAttributes(sQPath As String) As String*

(Available in XStandard Pro)
Returns an `XML` document containing the attributes for a given element. For example, `.GetAttributes("/body[1]/p[1]/a[1]")` may return:

```
<attributes>
<attr>
<name>class</name>
<value>intro</value>
</attr>
<attr>
<name>href</name>
<value>/news/</value>
</attr>
</attributes>
```

Sub *RemoveAttribute(sQPath As String, sName As String)*

(Available in XStandard Pro)
Removes an attribute from a given element.

Sub *SetAttribute(sQPath As String, sName As String, sValue As String)*

(Available in XStandard Pro)
Adds/updates an attribute for a given element.

Event *DialogPropertiesActivated(sQPath As String, sElement As String, sAttributes As String, sMetadata As String)*

(Available in XStandard Pro)
Occurs when the Properties dialog box is requested. This event can be used to replace the editor's Properties dialog box with your own Properties dialog box. Below is a description of each argument:
`sQPath` is a qualified XPath expression of the current element being edited. This value can be blank if the element has not been inserted into markup yet.
`sElement` is the name of the element being inserted/edited.
`sAttributes` is an XML document describing the attributes for this element. For example:

```
<attributes>
<attr>
<name>href</name>
<value>/files/report.doc</value>
</attr>
<attr>
<name>title</name>
<value>Year 2009 Annual Report</value>
</attr>
</attributes>
```

`sMetadata` is reserved for future use.
To pass to the editor attributes from your own dialog box and to cancel the editor's dialog box, call `SetDialogProperties()` during this event.
To capture this event in Web-based applications, use the following code:

```
<script type="text/javascript">
//<![CDATA[
function xsDialogPropertiesActivated(id, qpath, element, attributes, metadata) {
alert('Editor: ' + id + '; function: xsDialogPropertiesActivated()');
}
//]]>
```

```
</script>
```

Note, for Web-based applications, do not use floating divs as dialog boxes because they will render behind the editor in the browser's z-order.

Sub *SetDialogProperties(sAttributes As String, bCancelDialog As Boolean, bCancelOperation As Boolean)*

(Available in XStandard Pro)
Passes attribute values to the edtior or the Properties dialog box during `DialogPropertiesActivated()`event. Below is a description of each argument:
`sAttributes` is an XML document describing the attributes for this element. For example:

```
<attributes>
<attr>
<name>href</name>
<value>/files/report.doc</value>
</attr>
<attr>
<name>title</name>
<value>Year 2009 Annual Report</value>
</attr>
</attributes>
```

`bCancelDialog` is a way for your code to disable the editor's Properties dialog box. If the value is `True`, the editor will not display the Properties dialog box.
`bCancelOperation` is a way for your code to cancel modifications to the markup. For example, if the user presses "Cancel" button in your dialog box, you should set this value to `True`.

Event *Paste(bMarkup, sData)*

(Available in XStandard Pro)
This event fires when content is pasted into the editor. `bMarkup` is a boolean value indicating if the content to be pasted is plain text or markup. `sData` contains the content to be pasted. To modify the pasted content before it is inserted into the editor, call `SetPaste()` method.
To capture this event in Web-based applications, use the following code:

```
<script type="text/javascript">
//<![CDATA[
function xsPaste(id, markup, data) {
alert('Editor: ' + id + '; function: xsPaste()');
}
//]]>
</script>
```

Sub *SetPaste(bMarkup As Boolean, sData As String)*

(Available in XStandard Pro)
This method modifies the pasted content during the `Paste()` event.

Other new features in version 3.0

- Better support for Microsoft-only intranet environments. Some intranets use proxies and NTLMauthentication. Setting `ProxySetting` param tag to value `platform` should automatically pass NTLMauthentication over the network without prompting for a username/password.
- Support for data URLs.
- Support for Microsoft Active Accessibility (MSAA) API. Please note, this is an experimental feature because to create a **complete** accessible solution for users of assistive technologies such as screen readers, applications, Web browsers and screen readers must do their part. XStandard has done its part but some browsers and screen readers are lagging behind.

- Enhancements to CSS rendering.

See the "Features" chart for a complete listing of XStandard's features and functionality.