ThtmlViewer Component

Properties Methods Events Tasks Copyright and Registration

Unit

HTMLView.Pas

Description

The ThtmlViewer component provides an easy way to display **H**yper**T**ext **M**arkup **L**anguage (HTML) documents in a Borland Delphi program. This version of the viewer supports most of the HTML 2.0 specs with the exception of forms. Inline bitmap, gif, and jpeg images are supported.

While HTML documents are normally associated with the Internet World Wide Web, they are also turning out to be very useful for displaying all kinds of textual material such as documentation, helpfiles, etc. Graphics are also easily incorporated in these documents.

Updates

For the most recent version of ThtmlViewer and for information on what's coming, check in at:

http://www.empire.net/~dbaldwin/

HTMLToDos Function

Syntax

```
function HTMLToDos(FileName: String): string;
```

Description

The HTMLToDos function converts an HTML style filename to one for Dos. If the filename is in Dos format to begin with, no change takes place.

Example

After the call,

```
S := HTMLToDos('file:///c|/Big/Images/Glass.bmp');
```

S will contain 'c:\Big\Images\Glass.bmp'.

Properties

- Key Properties
- Runtime Only Properties
- Base
  - Color
- CurrentFile
- DocumentTitle
- History
- HistoryIndex
  - HistoryMaxCount
- Position
- URL
  - ViewImages
- VScrollBarPosition
- VScrollBarRange

Using The ThtmlViewer Component

ThtmlViewer Reference

Tasks

- To load an HTML document from a file, use the LoadFromFile method. To load a document from a TMemo or other memory buffer, use the LoadFromBuffer method.
- Use the Position property to save a location in a document and return to it later. For positioning to a **Name** field in the document, use the PositionTo method.
- To customize response to clicking on a hotspot, respond to the OnHotSpotClick event.
- To implement a status line which shows what URL a hot spot corresponds to, respond to the OnHotSpotCovered event.
- Use the Color property to change the background color.
- To implement a history list and/or Back|Forward buttons, see Setting up a History List.

Color Property

Declaration

```
property Color: TColor;
```

Description

Color sets the viewer's background color. Either clBtnFace or clWindow are appropriate values.

Position Property

Declaration

```
property Position: LongInt;
```

Description

The Position property is a coded number (LongInt) which specifies the part of the document appearing at the top of the window. Since this number is relatively independent of document reformatting, it may be used to save a position to be returned to at a later time.

Specifying a Postion of 0 will take you to the top of the document and a very large value will take you to the end. Values in between are not easily calculated.

For an alternate method of positioning, see:

VScrollBarPosition Property

VScrollBarRange Property

Example

```
var
  SavePos: LongInt;
....
SavePos := Viewer.Position; {save the position}
....
{intervening operations which may reformat the document}
....
Viewer.Position := SavePos; {return to original spot in document}
```

ViewImages Property

Declaration

```
property ViewImages: Boolean
```

Description

The **ViewImages** property determines whether inline images are drawn or simulated with a standard marker image. 

If **ViewImages** is True and an image error occurs, an error image  is displayed.

Events

- Key Events

- OnBitmapRequest
- OnHistoryChange
- OnHotSpotCovered
- OnHotSpotClick

Base Property

Description

```
property Base: string;
```

Read only and runtime only. The Base property returns the document's base directory as given in the **<base>** tag.

OnHotSpotCovered Event

Declaration

```
property OnHotSpotCovered: THotSpotEvent;
```

Description

The OnSpotCovered event occurs when the mouse is moved over or away from an hypertext link. This event may be used to display the destination URL of a hypertext link to allow the user to decide if it should be selected.

Example

The following code may be used to provide a status line display of the hypertext link at the mouse position:

```
Viewer.OnHotSpotCovered := HotSpotChange;
...
procedure TForm1.HotSpotChange(Sender: TObject; const URL: string);
begin
Panel1.Caption := URL;
end;
```

Type TGetBitmapEvent;

Declaration

```
TGetBitmapEvent = procedure(Sender: TObject;
       const SRC: string; var Bitmap: TBitmap) of Object;
```

Description

The TGetBitmapEvent type points to a method that handles image request events. SRC is the SRC= string recovered from the <img> tag. Bitmap is the bitmap you must return from a conversion routine. If a bitmap can't be supplied because of an error condition, etc., a Nil value must be returned.

See also:

OnBitmapRequest Event

VScrollBarRange Property

Declaration

```
property VScrollBarRange: Integer;
```

The VScrollBarRange Property gives access to the viewer's vertical scrollbar range value.

See Also:

VScrollBarPosition Property

Position Property

OnBitmapRequest Event

Declaration

```
property OnBitmapRequest: TGetBitmapEvent;
```

Description

The OnBitmapRequest event occurs when ThtmlViewer encounters an inline image request and the ViewImages property is set. You can use the OnBitmapRequest event to supply your own image conversion routines for image types that aren't presently handled by ThtmlViewer.

Example

This event handler checks requested file extensions to determine which conversion routine to call. In the following, It is assumed the GifToBitmap routine will convert a Gif image to a bitmap.

```
procedure TForm1.BitmapRequest(Sender: TObject;
         const SRC: string; var Bitmap: TBitmap);
var
  S: String;
  Ext: String[5];
begin
S := HTMLToDos(SRC);  {Make sure filename is in DOS format}
Ext := ExtractFileExt(Lowercase(S));
Bitmap := TBitmap.Create;
if Ext = '.gif') then
  GifToBitmap(S, Bitmap)
else if Ext = 'bmp' then
  Bitmap.LoadFromFile(S)
else
  begin
  Bitmap.Free;
  Bitmap := Nil;     {return Nil if can't supply bitmap}
  end;
end;
```

VScrollBarPosition Property

Declaration

```
property VScrollBarPosition: Integer;
```

Description

The VScrollBarPosition Property gives access to the viewer's vertical scrollbar. This can be used for document positioning in situations where reformatting will not occur.

See Also:

VScrollBarRange Property

Position Property

DocumentTitle Property

Declaration

```
property DocumentTitle: String;
```

Read only and runtime only. The DocumentTitle property returns the document's title as found between the **<title>** tags.

ThtmlViewer Methods

LoadFromFile
LoadFromBuffer
LoadStrings
PositionTo

Documentation and Program Copyright © 1995
by L. David Baldwin
All Rights Reserved

Unregistered Version

No registration is required for hobbyists, students, occasional users, etc.

The unregistered version uses the shareware version of ImageLib's Graphics DLL to display inline GIF, JPEG and PCX images. Since this is a shareware version, there will be occasional "reminder" messages.

Note that the use of the ImageLib software is not manditory. If you dont' need inline images or can get by with only bitmaps, the IMLIB221.DLL need not be used.

Registered Version

For those using ThtmlViewer professionally, a $60 registration fee is expected. With registration comes:

- Source code (but with restrictions against any kind of publishing).
- A Password to register (and remove the shareware messages) for ImageLib's GIF images. The password is considered confidential.
- A License to use ThtmlViewer in your ***programs*** for sale or company use.

However, if you're planning to use ThtmlViewer as part of ***another*** VCL component offered for sale, please contact me for additional information.

For full registration of ImageLib's graphics package including jpeg images, contact ImageLib.

Dave Baldwin,
CompuServe ID: 76327,53
Internet: dbaldwin@empire.net
Home Page: http://www.empire.net/~dbaldwin/

22 Fox Den Rd., (Summer)
Hollis, NH 03049
(603) 465-7857

144 13th St. East, (Winter)
Tierra Verde, FL 33715
(813) 867-3030

URL Property

Declaration

```
property URL: String;
```

Description

Read only and runtime only. The URL property returns the most recently selected (left mouse click) URL.

The URL property contains the URL exactly as it is read from the document. In some cases, it may be necessary to do some conversion to the string before using it. Depending on your application and the documents you are using, you might want to:

Convert from Internet format to Dos format using the HTMLToDos function.

Add a path if it doesn't already have one. The Base property provides the base path. If the Base property is empty, the path expected is generally that of the HTML document.

OnHotSpotClick Event

Declaration

```
property OnHotSpotClick: THotSpotClickEvent;
```

Description

The OnHotSpotClick event occurs when the mouse is over an hypertext link and the left mouse button is clicked. See the THotSpotClickEvent type for information on the parameters.

Example

The following code may be used to load the hypertext link URL into an Edit box when a mouse click occurs:

```
Viewer.OnHotSpotClick := HotSpotClick;
....
procedure TForm1.HotSpotClick(Sender: TObject; const URL: String;
           var Handled: Boolean);
begin
Handled := False;
Edit1.Text := HTMLToDos(URL);
end;
```

LoadFromFile Method

Declaration

```
procedure LoadFromFile(const Filename: String);
```

Description

The LoadFromFile method loads an HTML document for viewing, where Filename is the name of the file to be loaded. If Filename has no extension, .HTM is assumed.

Filename may optionally contain a document position identifier separated from the actual filename by a **#** sign.

Example

```
Viewer.LoadFromFile('MyHTML#Contents');
```

would load MyHTML.htm and position the view to the location where the "Contents" target was defined.

See also:

LoadStrings Method

LoadFromBuffer Method

LoadStrings Method

Declaration

```
procedure LoadStrings(Strings: TStrings);
```

Description

The LoadStrings method loads an HTML document from a TStrings object.

Example

Assume an HTML document exists in a TMemo object. The following line of code could be used to then copy the document to an ThtmlViewer object:

```
Viewer.LoadStrings(Memo1.Lines);
```

See also:

LoadFromFile Method

PositionTo Method

Declaration

```
procedure PositionTo(Dest: String);
```

Description

Dest is a target name defined within the HTML document. PositionTo positions the document display to that location.

Example

```
Viewer.PositionTo('#Contents');
```

The **#** sign may be omitted if desired.

URL - Uniform Resource Locator

A sequence of characters for specifying Internet resources. In HTML documents these specify hypertext link targets, image files, etc. As loosely interpreted for DOS HTML viewers, an URL can be a DOS path and filename.

Reformatting occurs mainly when the window width is resized causing line wrap to change. Reformatting can also occur when **ViewImages** is toggled.

Type THotSpotEvent

Declaration

```
THotSpotEvent = procedure(Sender: TObject;
       const URL: string) of Object;
```

Description

The THotSpotEvent type is the type for the hot spot covered event. This event occurs when the mouse is positioned over a hypertext link. URL is the string referenced by the hot spot.

See also:

OnHotSpotCovered Event

Type THotSpotClickEvent

Declaration

```
THotSpotClickEvent = procedure(Sender: TObject;
       const URL: string; var Handled: Boolean) of Object;
```

Description

The THotSpotClickEvent type is the type for hot spot click events. These events occur when the mouse is clicked on a hypertext link. URL is the string referenced by the hot spot.

The method which processes the THotSpotClickEvent should returned Handled = True if it wants no further handling of the URL. If Handled is set to false, ThtmlViewer will perform default handling.

See also:

OnHotSpotClick Event

OnHotSpotCovered Event

CurrentFile Property

Declaration

```
property CurrentFile: String;
```

Read only and runtime only. The CurrentFile property returns the full path and filename of the file currently loaded. If no file is loaded or the load was accomplished using the LoadFromBuffer method, CurrentFile will be an empty string.

LoadFromBuffer Method

Declaration

```
procedure LoadFromBuffer(Buffer: PChar; BufSize: LongInt);
```

Description

The LoadFromBuffer method loads an HTML document from a memory buffer. BufSize is the number of characters to be loaded. A terminating null character is not required.

Example

Assume an HTML document exists in a TMemo object. The following code could be used to then copy the document to an ThtmlViewer object:

```
var
  Buffer: PChar;
  Size: integer;
begin
Size := Memo1.GetTextLen;
Inc(Size);
GetMem(Buffer, Size);
Memo1.GetTextBuf(Buffer,Size);
Viewer.LoadFromBuffer(Buffer, Size-1);
FreeMem(Buffer, Size);
end;
```

See also:

LoadFromFile Method

History Property

Declaration

```
property History: TStrings;
```

Read only and runtime only. The History property contains a list of the most recently loaded document filenames with the current filename at index 0.

For additional information on setting up a history list, see Setting up a History List.

Setting up a History List

A history list is useful for backtracking to earlier loaded documents or places previously visited in the same document. From the users point of view, histrories are normally accessed as menu items and/or Back/Forward buttons.

The key properties and events relating to history lists are the History, HistoryIndex, and HistoryMaxCount properties and the OnHistoryChange event.

1. Set HistoryMaxCount to the number of list items wanted. HistoryMaxCount should be set to 0 if no history list is desired.
2. History (TStrings) will then be automatically updated as files are loaded and positions changed. The History strings may be used as menuitems or as listbox contents. Ordering is such that the current file is at index 0.
3. An OnHistoryChange event occurs each time the history list changes. A method responding to this event may be used to determine which buttons and/or menuitems are visible, grayed, or checked.
4. The HistoryIndex property may be read to determine which history list item is currently active. Setting of the HistoryIndex property to a new value loads and positions the appropriate file. HistoryIndex can not be set outside the range of the current history list.

HistoryMaxCount Property

Declaration

```
property HistoryMaxCount: Integer;
```

The HistoryMaxCount property determines the number of History items which will be maintained. If HistoryMaxCount is 0, then no history list will be maintained.

For further information on setting up a History list, see Setting up a History List.

HistoryIndex Property

Declaration

```
property HistoryIndex: Integer;
```

Runtime only. The HistoryIndex property shows which history item represents the currently loaded file and file position. Setting HistoryIndex to a new value automatically causes the loading and positioning of the applicable file.

For further information on setting up a History list, see Setting up a History List.

OnHistoryChange Event

Declaration

```
property OnHistoryChange: TNotifyEvent;
```

Description

The OnHistoryChange event occurs whenever the history list changes. For further information on history lists, see Setting up a History List.

The use of GIF Images in a commercial product may necessitate royalty payments to Unisys Corp. For further information, contact:

Mark T. Starr, Esq
Internet: StarrMT@po4.bb.unisys.com
Unisys Corp - MS/C2SW1
PO Box 500
Blue Bell, PA 19424

The full ImageLib shareware package may be downloaded from CompuServe's Delphi Forum, Library 5. The filename is IMAGELIB.ZIP.

For further technical information, contact:

Jan Dekkers
Skyline Tools,
Compuserve: 72130,353
Internet: 72130.353@compuserve.com