Contents

HelpFiles

Unit HelpFiles (in HELPFILE help file)

Unit to define abstract help file object. Included in SCANH3xx.ZIP as sample of comment formatting and TeX output.

Example This comment will be highlighted as example text in most output formats. The sample below won't be word-wrapped.

```
program HelloWorld;
begin
  writeln('Hello, world!');
end.
```

Types: *PTopic TTopic PIndexItem TIndexItem PIndex TIndex PHelpFile THelpFile*

PHelpFile HelpFiles

HelpFiles.PHelpFile = ^THelpFile

PIndex HelpFiles

HelpFiles.PIndex = ^TIndex

PIndexItem HelpFiles

HelpFiles.PIndexItem = ^TIndexItem

PTopic HelpFiles

HelpFiles.PTopic = ^TTopic

THelpFile HelpFiles

HelpFiles.THelpFile = object(TObject)

This is the main abstract object representing a help file. It serves as a container for THelpTopics.

Fields

Index *THelpFile.Index: PIndex*;

This is a *TIndex* maintained by the help file.

Methods

AddTopic *procedure THelpFile.AddTopic(ATopic: PTopic); virtual;*

Writes the topic at the end of the base file, and records it with the appropriate topic number. If a topic with that number existed previously, it'll effectively be deleted. Atopic is disposed after adding it.

DisplayTopic *procedure THelpFile.DisplayTopic(var Where: Text; TopicNum: Word); virtual;*

Displays the given topic number.

Done *destructor THelpFile.Done; virtual;*

Destroy the object and dispose of the *Index*.

GetSubTitle *function THelpFile.GetSubTitle(TopicNum: Word): String; virtual;*

Constructs a topic subtitle.

GetTitle *function THelpFile.GetTitle(TopicNum: Word): String; virtual;*

Constructs a topic title for the given topic number.

GetTopic *function THelpFile.GetTopic(Context: Word): PTopic; virtual;*

Extracts the given topic from the help file.

Init	*constructor THelpFile.Init;* Construct an empty help file, and initialize *Index* to nil.
NewTopic	*function THelpFile.NewTopic(Context: Word; Someinfo: Pointer): PTopic; virtual;* Constructs a new topic of the appropriate type. Someinfo might be used by a descendant type.
NumTopics	*function THelpFile.NumTopics: Word; virtual;* Return the number of topics in this file.
Rewrite	*procedure THelpFile.Rewrite(s: PStream); virtual;* Rewrites the help file to the given stream.
SetMainTopic	*procedure THelpFile.SetMainTopic(TopicNum: Word); virtual;* Defines which Topic is the main contents topic.

TIndex HelpFiles

HelpFiles.TIndex = object(TSortableCollection)

This is an index for a help file, meant to hold *TIndexItem* records.

Fields

Sortby	*TIndex.Sortby: (ByToken, BySubTitle, ByContext)*; Marks which sort order should be used.

Methods

AddItem	*procedure TIndex.AddItem(const ATitle, ASubtitle: String; Atopicnum: Word);* Add a new index entry by specifying the strings to use.
AddTokens	*procedure TIndex.AddTokens(ATitle, ASubtitle: TToken; Atopicnum: Word);* Add a new index entry by specifying the token numbers.
Compare	*function TIndex.Compare(Item1, Item2: Pointer): Integer; virtual;* Compares two index items according to the *Sortby* field.

FreeItem *procedure TIndex.FreeItem(Item: Pointer); virtual;*

Disposes of a TIndexItem.

Insert *procedure TIndex.Insert(Item: Pointer); virtual;*

This inserts duplicates after existing values.

TIndexItem HelpFiles

HelpFiles.TIndexItem = record

This is an item stored in the index for a help file.

Fields

Context *TIndexItem.Context: Word*;

The context or topic number.

Subtitle *TIndexItem.Subtitle,*

The token number of the subtitle string.

Token: TToken;

The token number of the name of index entry.

Token *See Subtitle*

TTopic HelpFiles

HelpFiles.TTopic = object(TObject)

An object holding a single topic as part of a *THelpFile*.

Fields

FixedLines *TTopic.FixedLines: Boolean*;

Whether lines should be fixed or wrapped.

Highlighting *TTopic.Highlighting: byte*;

Counts the current highlight level.

Marked *TTopic.Marked: Boolean*;

Whether text is currently being marked.

StartofLine *TTopic.StartofLine: Boolean*;

Whether the text is currently at the start of a line.

Text *TTopic.Text: PStream*;

A stream to which the text of the topic is written.

TopicNum *TTopic.TopicNum: Word*;

The topic number in the help file.

Methods

BlankLine *procedure TTopic.BlankLine; virtual;*

Writes a blank line to the help topic, starting a new paragraph afterwards.

Done *destructor TTopic.Done; virtual;*

Dispose of *Text* and destroy object.

EndXrefList *procedure TTopic.EndXrefList; virtual;*

Ends a list of cross-referenced topics started by *StartXrefList*.

GetLine *function TTopic.GetLine(var Buffer; MaxLen: Word): Word; virtual;*

Gets the next line of text, return the length.

HighLight *procedure TTopic.HighLight(On: Boolean); virtual;*

Turns highlighting of the text on or off. If turned on twice, it will need to be turned off twice to return to standard.

Init *constructor TTopic.Init(Atopicnum: Word);*

Initialize an empty topic with the given value for *TopicNum*.

MoreLines *function TTopic.MoreLines: Boolean; virtual;*

True if there are more lines of text.

ResetHighLight *procedure TTopic.ResetHighLight; virtual;*

Turns highlighting off regardless of the initial state.

StartXrefList *procedure TTopic.StartXrefList(const s: String); virtual;*

Starts a list of cross-referenced topics. End the list with *EndXrefList*.

ToggleFixed *procedure TTopic.ToggleFixed; virtual;*

Toggles word-wrap mode. Generally, help files start out word-wrapping; this should turn it off. The TTopic method just toggles *FixedLines*.

ToggleMarked *procedure TTopic.ToggleMarked; virtual;*

Toggles word marking mode. Typically marked text would be used for code samples, as in the Borland help files. This one just toggles *Marked*.

Write *procedure TTopic.Write(s: String); virtual;*

Writes the string to the help text.

WriteKeyWord *procedure TTopic.WriteKeyWord(const s: String; Crossref: Word); virtual;*

Writes the string with a marker that it's a cross-reference.

WriteLn *procedure TTopic.WriteLn(const s: String); virtual;*

Writes, then adds a newline.

WriteXref *procedure TTopic.WriteXref(const s: String; Len, Crossref: Word); virtual;*

Like *WriteKeyWord*, but writes an entry to a cross-ref list. Len is the length in characters of the longest Xref to come; this may be used to format nicely. Assumes that *StartXrefList* has been called.

Index