ExportLVReg unit

Components
TExportListView

Types
TExportType

Creating your own Export Formats!!!

To create your own Export Routines please read the **TCustomExportListView.htm** file included in the component's zip file.

TExportListView component

See also Properties Methods Events Tasks

Unit
ExportLVReg

Description

TExportListView exports the contents of a ListView to either the screen, a file or the clipboard (as text). It can export to many file and application formats, even the clipboard. It is a wholly professional component that shows the export progress (if that property is set) and you can even create your own custom export formats quickly and easily with it.

Note: It's suggested that you're listview be in vsReport mode. I haven't tested it with the others but they should work too.

You can register this component at http://www.igather.com/components. Also be sure to have a look at the many freeware and shareware components and classes available there.

Using TExportListView

Please have a look at the demo included in the component's zip file.

Note

It will mention HelpScribble at the bottom of this help file, which is the program I used to create this help file. If you click on it, it will say that this is an unregistered version and I am not honest... Well, it is an unregistered version but I did register HelpScribble more than a year ago (before October 11th, 1997) and did not use it for a long time and lost the executable file. Now, I could only find the most recent version and it does not use my Registration number, instead I think it uses some kind of file ... anyhow, I'm waiting for my new registration to get e-mailed to me, I've been waiting well over a month now and I can't wait any longer to release these components so that's why you see that note at the bottom. I will release a new version of my component zips as soon as I get HelpScribble to recognize that I am a registered user.

See also

TListView

Properties

- Run-time only
- Key properties
  - AllowedTypes
  - Captions
  - ExportFile
  - ExportType
  - Footer
  - Header
  - ListView
  - Options
  - Title
  - ViewOnly

Methods

Key methods

- Create
- Execute
- Print
- Choose

Events

- Key events
  - OnBeginExport
  - OnExportFailed
  - OnExportFinished
  - OnPrintFailed

TExportListView Tasks

TExportListView reference

Tasks

How do I make TExportListView speak my language?

Exporting using the Export Format specified in the component's ExportType property.
Exporting using the Export Format the user chooses.
Printing the contents of a TListView

How do I make TExportListView speak my language?

Simple really. There are 2 ways of doing it:

1. Get the **ELV_Strings.pas** file for your language and follow the installation instructions that come with it. Try downloading it from the Y-Tech Components web site.

or

2. Translate it yourself. (if there is no **ELV_Strings.pas** for your language, you must do it this way). It's simple really, just find the **ELV_Strings.pas** unit that came with **TExportListView** and translate the strings inside. There's not that many, so it's not a big job. Then recompile the package/component library that your component is installed in. Finally, if you think you made a half-decent translation, please e-mail it to **Y-Tech** at ycomp@hotpop.com

ExportFile property

See also

Applies to

TExportListView component

Declaration

```
property ExportFile: String;
```

Description

This is the name of the Exported File. It is only used if the ViewOnly property is **False**.

If this property is blank and ViewOnly is **False**, then a dialog will pop up and ask the user for the name of the export file to create.

Notes:

- The file name's extension should of course correspond to the ExportType ie. **XLS** for a Microsoft Excel Document.
- The contents of this property are lost when you change the value of the ExportType property.

See also

ViewOnly
ExportType

AllowedTypes property

Applies to
TExportListView component

Declaration
```
property AllowedTypes: TExportTypeSet;
```

Description

These are all the **Allowed Export Types**. If you don't want your users to be able to export to a particular Export Type, then just set it to **False**.

ExportType property

See also Example

Applies to

TExportListView component

Declaration

```
property ExportType: TExportType;
```

Description

Here's where you choose the type of Export File to create. When you set this property, the contents of the ExportFile property are erased.

This property also determines which Export Type will be the first to be shown in the Choose Dialog Box. This value is only used for this purpose the first time the **Choose Dialog Box** is displayed, after that the **Choose Dialog** displays the **Last-Used Export Type** by default.

See also

ExportFile

ExportType property example

```
ExportListView1.ExportType := Microsoft_Excel;
```

Captions property

Applies to
TExportListView component

Declaration
```
property Captions: TStrings;
```

Description

You can use the property to specify alternate column headings for your exported documents. That is, if you want the Column Headers to be different in the Exported Document than in the source TListView, just type new column headings here.

Footer property

See also

Applies to

TExportListView component

Declaration

```
property Footer: String;
```

Description

The Footer is displayed after the contents of the ListView in the Exported File. Text_Comma_Delimited and Text_Tab_Delimited Export Types ignore this value.

See also

Header

Header property

See also

Applies to

TExportListView component

Declaration

```
property Header: String;
```

Description

The Header is displayed before the ListView Contents in the Exported File. Text_Comma_Delimited and Text_Tab_Delimited Export Types ignore this value.

See also

Footer

ListView property

See also

Applies to

TExportListView component

Declaration

```
property ListView: TListView;
```

Description

This is the `TListView` to export. It **must be assigned** before calling the Execute or Choose methods.

See also

Execute

Options property

Applies to
TExportListView component

Declaration
```
property Options: TExportOptions;
```

Description

Please look at the TExportOptions Type.

Title property

Applies to
TExportListView component

Declaration
```
property Title: String;
```

Description
Title for the Exported File. This is not the filename but the title for the document (ie. the first thing the user will usually read). Text_Comma_Delimited and Text_Tab_Delimited Export Types ignore this value.

ViewOnly property

See also

Applies to

TExportListView component

Declaration

```
property ViewOnly: Boolean;
```

Description

When the Execute or Choose methods are called the following happens:

If **True**, it displays the Exported File on the screen but does not write it to a file on the computer (unless it uses a temporary file). If **True**, the value of the ExportFile property is ignored when the file.

If **False**, it *does not* display the exported file to the screen but creates an exported file on the user's computer.

See also

ExportFile

Execute method

See also Example

Applies to

TExportListView component

Declaration

```
function Execute: Boolean;
```

Description

Calling this method will perform the export. If the export fails it will return false.

Notes:

- The ListView property must be assigned before calling this method.
- If ViewOnly is **False** and the ExportFile property is blank then a dialog will pop up asking the user what he wants to name the export file.
- If ViewOnly is **True** and the ExportFile property is not blank, a file of the type specified in ExportType will be created containing the contents of the ListView.
- This method assumes you know what type of ExportType to create. If you want to let the user choose, call the Choose method instead.
- Any Hidden TListColumn (ie. not visible, meaning the Width property is 0) will not be exported. So if you don't want to export any column, just set it's width to 0.")

See also

ViewOnly
ExportFile

Print
Choose

Execute method example

```
with ExportListView1 do
    if Execute then // Attempt Export...
        if not ViewOnly then // If Exporting to Screen,
            ShowMessage(Format('Succesfully Exported to "%s"', [ExportFile])); // otherwise, show
             file name also.
```

Note: "**if** Execute **and not** ViewOnly " would look nicer, it just depends if you have short-circuit boolean evaluation set or not **{$B}**

OnBeginExport event

See also

Applies to

TExportListView component

Declaration

```
property OnBeginExport: TNotifyEvent;
```

Description

This event is triggered when an export starts.

See also

OnExportFinished
OnExportFailed

OnExportFailed event

See also

Applies to

TExportListView component

Declaration

```
property OnExportFailed: TNotifyEvent;
```

Description

This event is only triggered if an export fails. You can put messages to the user here, however TExportListView has it's own built in error messages so it's not necessary.

See also

OnBeginExport
OnExportFinished

OnPrintFailed event

Applies to
TExportListView component

Declaration
```
property OnPrintFailed: TNotifyEvent;
```

Description
This event is only triggered if printing fails. Do what you want with this event handler.

OnExportFinished event

See also

Applies to

TExportListView component

Declaration

```
property OnExportFinished: TNotifyEvent;
```

Description

This event is triggered when an export succeeds. You can put a "beep;" in here if you want some kind of audible notification for your user.

See also

OnBeginExport
OnExportFailed

Choose method

See also Example

Applies to

TExportListView component

Declaration

```
function Choose: Boolean;
```

Description

Calling this method will first ask the user what type of export he wants, then it will prompt him for a file name and finally it will actually export it. This is absolutely the easiest way to export something, you don't need to worry about setting any properties at all except the ViewOnly property.

Notes:

- If you already know what type of ExportType to use, use the Execute method instead.
- Any Hidden TListColumn (ie. not visible, meaning the Width property is 0) will not be exported. So if you don't want to export any column, just set it's width to 0.")
- The first time the Choose Dialog Box is displayed, it defaults to the Export Type specified by the ExportType property. Subsequent calls to the Choose method will display the **Last-Used Export Type** instead.

See also

Execute

Choose method example

ExportListView1.Choose;

Note: If you want something a bit fancier, have a look at the Execute method example code.

TExportType type

See also

Unit

ExportLVReg

Declaration

```
type TExportType = (HTML,
                    Microsoft_Excel,
                    Microsoft_Word,
                    Text,
                    Text_Comma_Delimited,
                    Text_Tab_Delimited,
                    RichText,
                    Clipboard);
```

Description

These are all the export types currently available to the TExportListView component. You can set them using the ExportType property.

See also

ExportType property

TExportTypeSet type

See also

Unit

ExportLVReg

Declaration

```
type TExportTypeSet = set of TExportType;
```

Description

This typed is used by the AllowedTypes property.

See also

AllowedTypes

TExportOptions type

See also

Unit

ExportLVReg

Declaration

```
type TExportType = set of (ExportInvisibleCols,
                           NumberRows,
                           SelectedRowsOnly,
                           ShowClipboardMsg,
                           ShowProgress,
                           TimeStamp);
```

Description

If any of the following are included in the **set**, then this is what they will do:

ExportInvisibleCols:

Exports all columns, regardless of whether they are visible or not.

NumberRows:

Numbers each row exported.

SelectedRowsOnly:

Exports only the Selected Rows.

ShowClipboardMsg:

Shows a dialog informing the user that the export to the Clipboard has finished.

ShowProgress:

Shows the Export Progress if it's **True**. (recommended, because users can get impatient sometimes and it's remarkable how much a little progress bar will do for them :)

TimeStamp:

Marks the exported document with the **Time** & **Date** of Export.

Note: Progress can be aborted by pressing the **ESC** key or closing the Progress Form.

See also

~~Options property~~

Print method

See also

Applies to

TExportListView component

Declaration

```
function Print: Boolean;
```

Description

Calling this method will print the contents of the TListView specified by the ListView property. If the user cancels or printing fails it will return **False**.

Notes:

- The ListView property must be assigned before calling this method.
- The contents of the ViewOnly, ExportFile, ExportType properties are completely ignored because those are only used when actually exporting. All other properties are used though, so make sure they're set correctly.
- If you want to implement Print Setup in your application also, it's easy... Just drop a TPrintSetupDialog component on the form and execute it from your **File|Print Setup** menu item.

See also

Execute
Choose

HelpScribble

HelpScribble is a help authoring tool written by Jan Goyvaerts. This help file was created with the unregistered version of HelpScribble, which is why you can read this ad. Once the author of this help file is so honest to register the shareware he uses, you will not see this ad again in his help files.
Recompiling the help project with the registered version is all it takes to get rid of this ad.

HelpScribble is a stand-alone help authoring tool. It does *not* require an expensive word processor. (Only a help compiler as Microsoft likes keeping the .hlp format secret. Not my fault.)

Here are some of HelpScribble's features:

- The Setup program will *properly* install and uninstall HelpScribble and all of its components, including registry keys.
- Create, edit and navigate through topics right in the main window. No need to mess with heaps of dialog boxes.
- All topics are listed in a grid in the main window so you won't lose track in big help projects. You can even set bookmarks.
- Use the built-in Browse Sequence Editor to easily create browse sequences.
- Use the built-in Window Editor to change the look of your help window and create secondary windows.
- Use the built-in Contents Editor to create Windows 95-style contents files. Works *a lot* better than Microsoft's HCW.
- No need to mess with Microsoft's SHED: use the built-in SHG Editor to create hotspot bitmaps. Draw your hotspots on the bitmap and pick the topic to link to from the list.
- With the built-in Macro Editor you can easily compose WinHelp macros whenever needed. It will tell you what the correct parameters are and provide information on them.
- If you have a problem, just consult the online help. The help file was completely created with HelpScribble, of course.
- HelpScribble is shareware. However, the unregistered version is *not* crippled in any way. It will only add a small note to your help topics to encourage you to be honest and to register the shareware you use.

These options are very interesting for Delphi and C++Builder developers:

- If you are a component writer, use the Delphi Parser to build an outline help file for your component. Just fill in the spaces and you are done. HelpScribble can also extract the comments from your source file and use them as the default descriptions.
- If you are an application writer, HelpScribble provides you with a property editor for the HelpContext property. You can select the topic you need from a list of topic titles or simply instruct to create a new topic. No need to remember obscure numbers.
- The property editor also provides a tree view of all the components on your form and their HelpContext properties. This works very intuitively. (Much nicer than those help tools that simply mess with your .dfm files.)
- HelpScribble can perform syntax highlighting on any Delphi source code in your help file.

HelpScribble is shareware, so feel free to grab your copy today from my web site at http://www.ping.be/jg/