LazReport Documentation

From Free Pascal wiki

Deutsch (de) | **English (en)** | **Español (es)**

Contents

OVERVIEW

LazReport is a group of components to add reporting capabilities to applications. It uses a visual designer to create banded reports and includes a report engine with previewer and an interpreter to run user scripts. The report designer can be invoked at runtime.

LICENSE

LazReport is based on FreeReport 2.32 and thanks to Fast Reports Inc. it's available under modified LGPL, the same license as the Lazarus LCL. See files lazreport/license.txt, license-rus.txt and license-lazreport.txt for details.

AUTHORS

FreeReport was created for Fast Reports Inc.

LazReport initial port was made by Olivier Guilbaud.

Lazarus integration and fixes by Jesus Reyes A.

Many contributors, see lazreport/doc/contributors.txt file

INSTALL

To install LazReport in the Lazarus IDE:

1. Open LazReport Package. Menu: Components->Open package file (.lpk)...
2. Open file components/lazreport/source/lazreport.lpk
3. Install

The next time Lazarus is started, it should show a LazReport tab in the component palette.

DOCUMENTATION

Developer's manual and User Guides specific for LazReport are yet to be written. In the mean time, most of LazReport features are described in the FreeReport Developer's Manual (see lazreport/doc/fr_eng.sxw). Platform specific things described there like OLE objects are not implemented in LazReport, also some examples or pictures make reference to samples available only on Delphi.

Until LazReport documentation is elaborated, this Wiki page will be used as a documentation container; maybe in the future the missing documentation could be generated from here. Users are welcomed to add topics that they feel need to be documented.

Export Filters

LazReport export filters are invoked using the following code.

```
if TheReport.PrepareReport then
  TheReport.ExportTo(TfrHTMExportFilter, 'exportedfile.html');
```

Where TheReport holds an instance of TfrReport component. In this sample a TfrHTMExportFilter is used to generate a file named 'exportedfile.html'. It's not necesary to prepare the report again if it has been prepared previously. In order to use TfrHTMExportFilter the developer has to drag and drop an instance of the TfrHTMExportFilter component from the LazReport tab in the component palette to the form in Form Desginer. As an alternative the unit **lr_e_htm.pas** file can be added to the unit uses clause.

Since LazReport 0.9.6, the export filters support has been enhanced, now export filters can take parameters which users can customize either by changing values directly or by presenting the end user some UI. In order to make changes in parameters, the developer can create an event handler for TfrReport.OnExportFilterSetup event, available by selecting the TfrReport component and selecting the Events tab in Object Inspector.

The OnExportFilterSetup (of type TExportFilterSetup) event handler takes an argument sender of type TfrExportFilter, in order to use this type, the **lr_class.pas** unit must be added to the unit uses clause. All ExportFilter clases share this event and the developer has to type-cast the sender argument to the desired export filter class, for example:

```
if sender is TfrHTMExportFilter then
```

```
begin
  TfrHTMExportFilter(sender).UseCSS := false;
end;
```

Below a description of available export filters.

TfrExportFilter

is the base class of all export filters, and defines two properties that can be modified by the developer in the OnExportFilterSetup event:

- **BandTypes**: TfrBandTypes. this is a set of band types, only bands included in this set are actually exported; the other bands on the report will be ignored. Using this property a developer could for example export only the master band content, leaving titles, headers and footers out of exported output by doing:

```
sender.BandTypes := [btMasterData];
```

By default, all bands are processed but TfrCSVExportFilter changes this property to process only master header, column header and master data bands.

- **UseProgressbar**: boolean. This property enables or disables showing the progress bar while processing the export filter. This property is false by default.

TfrTextExportFilter

inheritance: TfrExportFilter <- TfrTextExportFilter
located in: **lr_e_txt.pas** file included with LazReport package.

is the base class of text based export filters. This export filter tries to make a text representation of a graphical report by fitting the original graphical coordinates into a more coarse grid where each unit is of "UsedFont" pixels, depending on the value of UsedFont value, the exported output may more or less represent the layout of objects in graphical report. Beside the properties inherited from TfrExportFilter class, TfrTextExportFilter define two more properties.

- **UsedFont**:integer. this property define the pixel dimensions on the output grid, objects on report are fitted into this grid by reclaculating each x and y position. The default value is 10, if user changes this value to 0 o less, LazReport will show automatically a dialog asking for the UsedFont value, if user enter a invalid value, a 10 value will be used. The 10 value is used because is the value that better fits the usual reports which are made with fonts 10-13 points.
- **UseBOM**:boolean. This property enable the inclusion of UTF-8 Byte Order Mark character at the start of text output (see BOM (http://en.wikipedia.org/wiki/Byte_order_mark)) needed by some editors/viewers, by default no BOM is used in exported output.

TfrCSVExportFilter

inheritance: TfrExportFilter <- TfrTextExportFilter <- TfrCSVExportFilter
located in: lr_e_csv.pas file included in LazReport package.

This special text export filter produces Comma Separated Value output (actually any character can be used as separator), it differs from its ancestor in that it doesn't try to create text layout representation of graphical

report, instead, for each record output it tries to guess the fields order from the source report, it then produce a list of fields using a separator defined by the user. Beside the properties inherited from its ancestor classes it defines some properties to customize the generated output.

- **QuoteType**:TfrQuoteType. This property controls whether the generated field value should be wrapped using specified quote char or not. Possible values are *qtNone*, *qtQuoteChar* and *qtAutoQuote*. With *qtNone* the field value is never wrapped, *qtQuoteChar* will use the character specified by property **QuoteChar**, any instance of **QuoteChar** already present in field value is duplicated. *qtAutoQuote* first try to find if any instance of **separator** or **QuoteChar** is already present in field value, if affirmative it behaves as if *qtQuoteChar* has been specified, on the contrary case, the field value is not wrapped just as if *qtNone* has been specified. **QuoteType** property is set to *qtQuoteChar* by default.
- **QuoteChar**:TUTF8Char. This holds the character to be used to wrap the field value in case of **QuoteType** of value *qtQuoteChar* has been specified or deduced if *qtAutoQuote* is set. Any instance of this character in the field value will be duplicated. by default **QuoteChar** is set to the " character.
- **Separator**:TUTF8Char. This is the character used to separate the fields in each record, by default is set to the ',' (COMMA) character but any UTF-8 valid character could be used. Some CSV files are actually TAB separated files, to get this, set Separator:=#9;

The CSV exporter do not use the inherited UsedFont property value so any value set will be ignored. With default property values, TfrCSVExportFilter will produce Excel compatible files, the only caveat is that Excel will not recognize the file as UTF-8 encoded. To force Excel to recognize the encoding automatically, set the property **UseBOM** to true.

TfrHTMExportFilter

inheritance: TfrExportFilter <- TfrTextExportFilter <- TfrHTMExportFilter.
Located in: lr_e_htm.pas in LazReport package.

This special text export filter produces valid "HTML 4.01 Transitional" output. Currently it defines only one additional property.

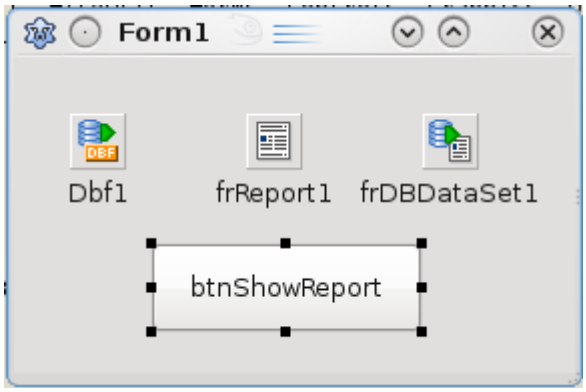
- **UseCSS**:boolean. This property controls whether or not the produced output include CSS information, this property is set to true by default.

QUICK START

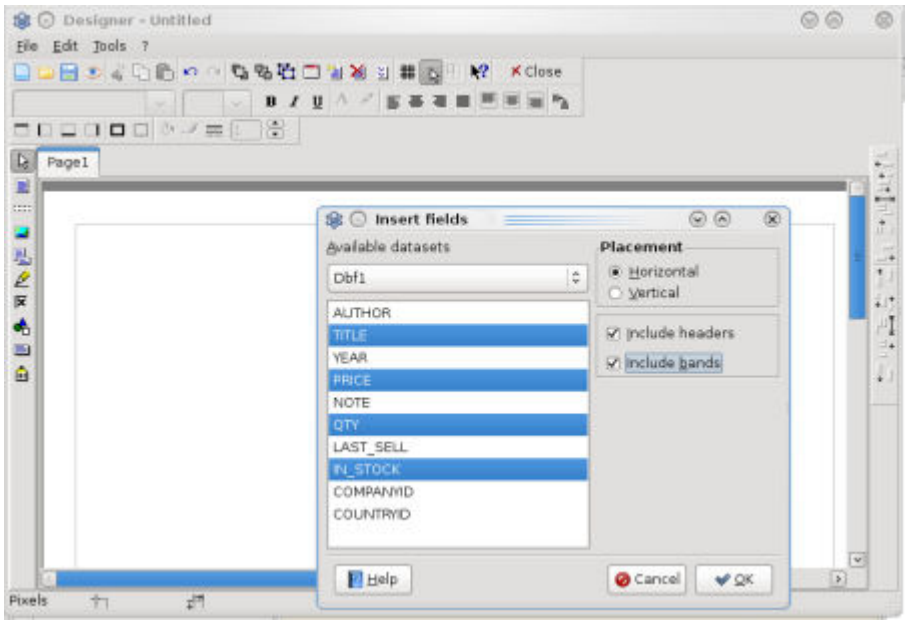
Report Records from a Dataset

In this sample we will design a report to print records from a dataset derived component (TDbf, TSQLQuery, TZTable, TZQuery, Etc.). First it's assumed that LazReport is already installed and the dataset component, called "Dbf1" here, is already configured and active.

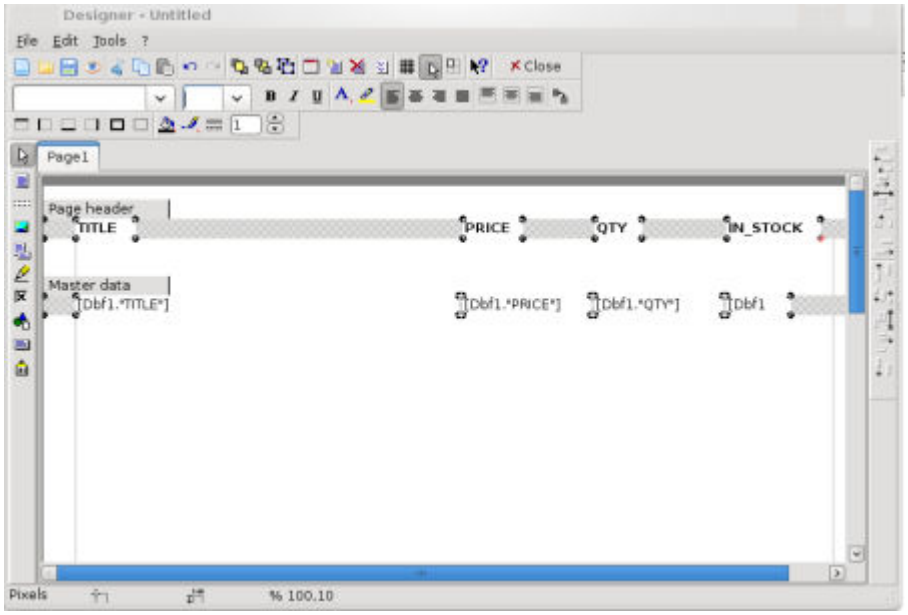
1. From LazReport tab, select TfrReport component and drop it in Form Designer, it will be named "frReport1".
2. Drop a TfrDbDataset component, it will be named "frDbDataset1"
3. Drop a TButton, it will be named Button1, change its name to "btnShowReport"
4. With frDbDataset1 selected in Object Inspector select "Dbf1" for the "Dataset" property
5. Now select frReport1 and using the same procedure link its "Dataset" property to "frDbDataset1":



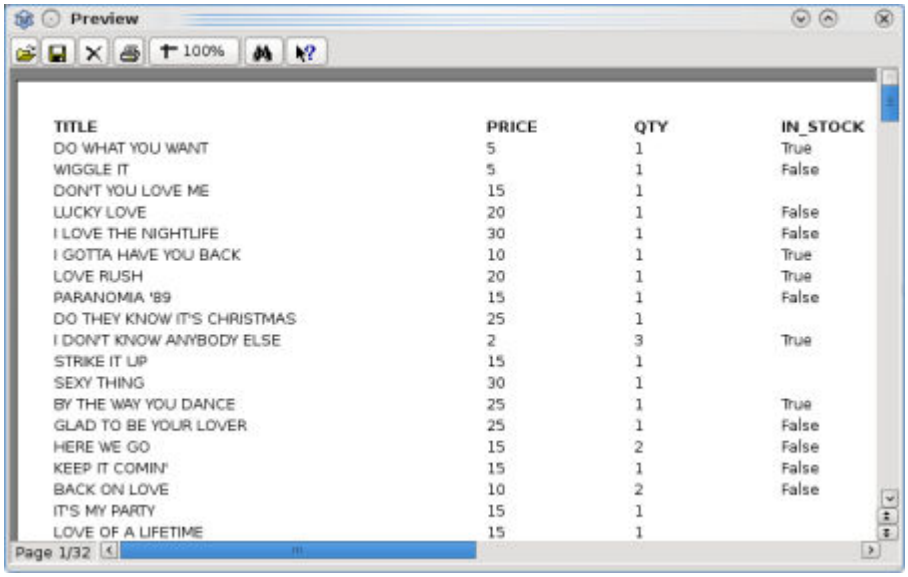
6. Right Click frReport1 and select "Design Report" from the menu, LazReport report designer will appear.
7. From Designer window, select menu Tools->Tools->Insert DB Fields, Insert fields Dialog will appear.
8. In field list select the fields you want in the report
9. Check "Include Headers" and "Include Bands" options, the result should be something like this:



10. Press the "Ok" button, LazReport will arrange bands and fields like this:



11. Press the preview button , LazReport will then arrange things and show:



TITLE	PRICE	QTY	IN_STOCK
DO WHAT YOU WANT	5	1	True
WIGGLE IT	5	1	False
DON'T YOU LOVE ME	15	1	
LUCKY LOVE	20	1	False
I LOVE THE NIGHTLIFE	30	1	False
I GOTTA HAVE YOU BACK	10	1	True
LOVE RUSH	20	1	True
PARANOMIA '89	15	1	False
DO THEY KNOW IT'S CHRISTMAS	25	1	
I DON'T KNOW ANYBODY ELSE	2	3	True
STRIKE IT UP	15	1	
SEXY THING	30	1	
BY THE WAY YOU DANCE	25	1	True
GLAD TO BE YOUR LOVER	25	1	False
HERE WE GO	15	2	False
KEEP IT COMIN'	15	1	False
BACK ON LOVE	10	2	False
IT'S MY PARTY	15	1	
LOVE OF A LIFETIME	15	1	

12. Save the report with Menu File->Save, select the same directory as the current project and name it as listing1.lrf
13. Close the report designer.
14. Double click btnShowReport and type next code:

```
frReport1.LoadFromFile('listing1.lrf');
```

frReport1.ShowReport;

BUG REPORTS

Please report problems using the lazarus/freepascal bugtracker (http://www.freepascal.org/mantis/main_page.php) , project: "Lazarus Packages", Category "LazReport", for patches please submit a bug report and attach the patch to it.

Retrieved from "http://wiki.freepascal.org/index.php?title=LazReport_Documentation&oldid=56989"
Category: Components
