



QtRptDesigner

Version 2.1.0

Programmer: Aleksey Osipov

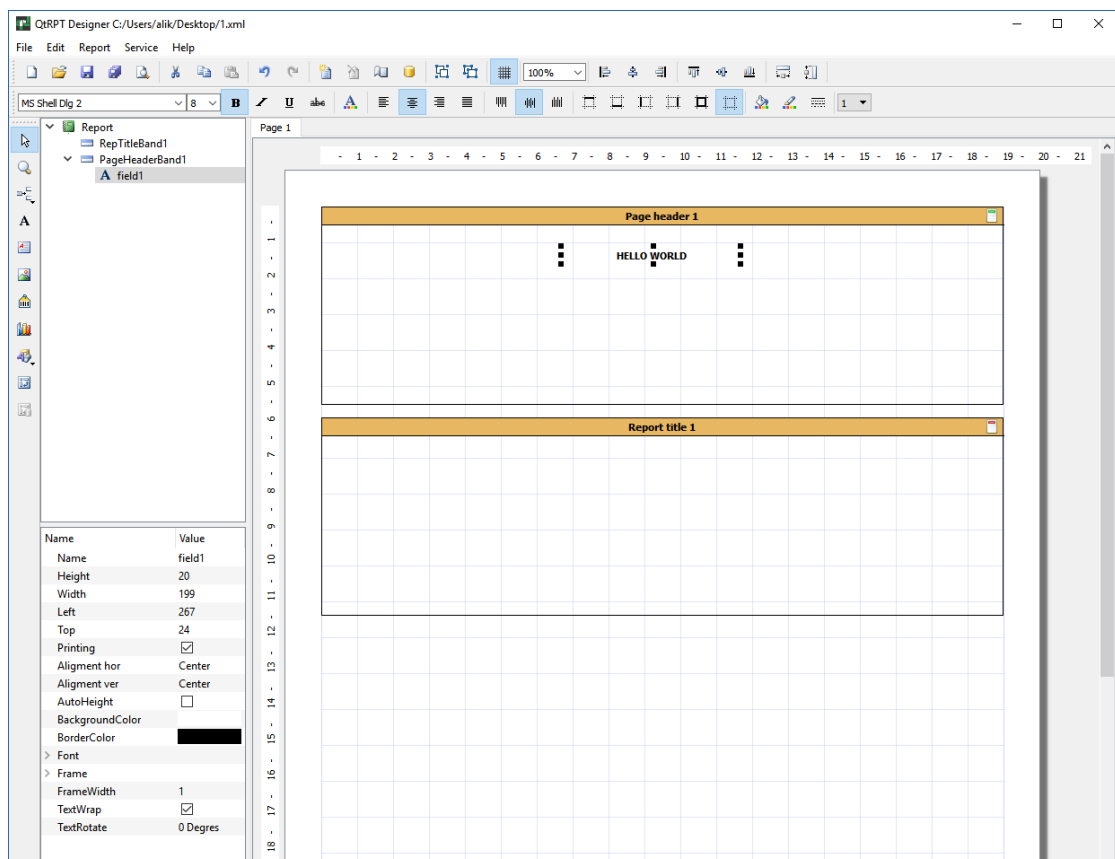
Web-site: <http://www.aliks-os.tk>

Email: aliks-os@ukr.net

Web-site: <http://www.qtrpt.tk>

Address in Facebook <https://www.facebook.com/qtrpt>

QtRptDesigner attended for preparation XML files, which will be used and processed by QtRPT engine for building reports. XML file contains information about of different items and their properties, and how this items should get data from data source, etc. QtRPT engine allows to process script embedded into the report. By script, the user can defined, how the fields must be processed depends of some condition.





























Interface of the Designer

For convenience, the buttons are grouped into a logical group and each is located on a separate panel. In addition, in the designer all elements are presented in the form of a tree, which allows you to clearly see the hierarchy of the report.

Edit Panel



Contains the following buttons











	New report
	Open
	Save
	Save as
	Preview
	Cut
	Copy
	Paste
	Undo
	Redo
	New report page
	Delete page of the report
	Page setting
	Script editor
	Data source <i>(Not needed for Codiak users)</i>
	Group selected items
	Ungroup
	Show grid
	Align by left edge all selected items
	Align by center all selected items
	Align by right edge all selected items
	Align by top edge all selected items
	Align by center (vertical) edge all selected items
	Align by bottom edge all selected items
	Set same width on selected items
	Set same height on selected items

Formatting Panel



	Font selection
	Bold
	Italic
	Underline
	Strikeout
	Font color
	Align left
	Align center
	Align right
	Justify
	Align top
	Align V center
	Align bottom
	Top border
	Bottom border
	Left border
	Right border
	Set all borders
	Remove all borders
	Background color
	Border color
	Border style
	Border thickness

Items Panel элементов

	Select tool
	Magnifying glass
	Add band
	Add field
	Add rich text
	Add image
	Add barcode
	Add chart
	Add draw
	Add cross table (<i>Not needed for Codiacy users</i>)

Bands

Band is area where the user's fields can be placed. The band can be a following type:

Band type	Purpose of the band
Report Title	This band is printed only once at the start of the report and only on the first page
Page Header	This band is printed at the top on each page
Data Grouping Header	This band is used when you need to group data that are placed on the Master Data band and show Header for each group
Master Header	This band is printed before Master Data band
Master Data	This band is intended for printing of some data set. The band printing is repeated depends of records count
Master Footer	This band is printed after Master Data band
Data Grouping Footer	This band is used when you need to group data that are placed on the Master Data band and show Footer for each group
Page Footer	This band is printed at the bottom on each page
Report Summary	This band is printed only once at the end of the report and only on the last page

To make report more flexible, you can put more than Master Data Band and corresponded bands (Master Header, Master Footer and Grouping Bands). Each Master Data Band has own data source.

Note for Codiacy Users: You should enter the alias for data source of the band

Report structure and field's property

The screenshot shows a report structure tree on the left and a properties panel on the right. The tree is expanded to show 'field1' under 'PageHeaderBand1'. The properties panel lists various attributes for 'field1'.

Name	Value
Name	field1
Height	20
Width	199
Left	267
Top	24
Printing	<input checked="" type="checkbox"/>
Alignment hor	Center
Alignment ver	Center
AutoHeight	<input type="checkbox"/>
BackgroundColor	
BorderColor	
Font	
Frame	
FrameWidth	1
TextWrap	<input checked="" type="checkbox"/>
TextRotate	0 Degrees

At left from report you can see the report structure and set of property. When some field or band is selected, the appropriate properties are shown. From the list of properties, you can change any property, as well as by pressing appropriate button on the panel

Page settings

The screenshot shows the 'Page settings' dialog box with various tabs and settings. The 'Size' tab is active, showing dimensions for A4 paper. Other tabs like Orientation, Margins, Page border, and Watermark are also visible.

Size

Width: 21.00 Cm
Height: 29.70 Cm

Orientation

☒ Portrait
☐ Landscape

Margins

Left: 1.00 Cm Right: 1.00 Cm
Top: 1.00 Cm Bottom: 1.00 Cm

Page border

☐ Draw border Border color: ...
Border width: 1 Border style: ...

Watermark

☒ Use watermark
Image: [Image]
Opacity: 50

You can set parameters for each page of the report. Set a size and orientation, margins and background images (watermark).

Working with the Fields

The purpose of the field (element) is display information. The field should be placed on the band only. There are several types of the fields. Before placing field on the report, it should, contains at least one band. To place field on the band, click on selected field on the Panel of Elements, then click on band. The field will be placed. On selecting, the field's properties will be shown in the list at the right. You can change the property of the field in the list or by clicking appropriate button on the panel.

Fields moving and resizing

The field can be moved or resized by mouse or by keyboard. When you use the keyboard, press **Ctrl + Arrow** key to move the field or **Shift + Arrow** to resize the field. You can select the several fields, to do it, please click on the first field, then press and hold on Ctrl key. While the Ctrl is pressed, click on the other fields that you want to select. Then you can use appropriate combination of keys (Ctrl + Arrow) or (Shift + Arrow) to move or resize the selected fields simultaneously.

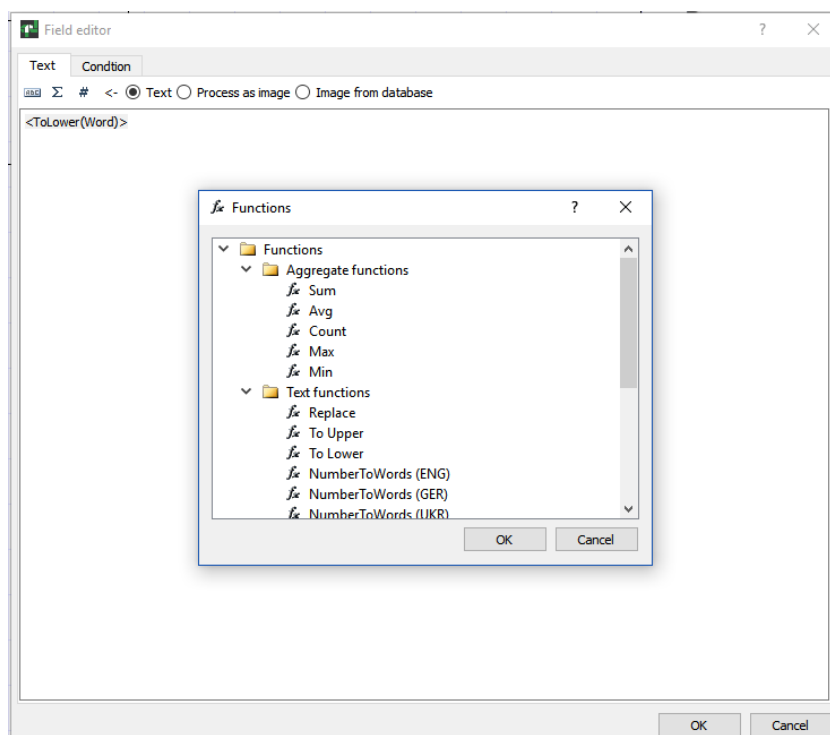
Field's Alignment

Several selected fields can be aligning to the one of edge. To do it, select the several fields and click on the appropriate button. All fields will be aligned to edge of the first selected field.

Data of the field

If the text in the should be static – just enter the text. If the text should be taken from any data source of third party application, you should enter the alias in the following form

[AliasOfTheField] – the square brackets indicate that it is alias. In the field you can some build-in functions. To add function or take a look at the full list of function, double click on the field, then click on “Add function” button. The angle brackets indicate that it is a “function”.



You can mix using aliases and build-in functions. For example:

<ToLower([companyName])> - This means that will be taken string from alias “companyName” and all cases will be set to lower.

The price is <NumberToWords(‘ENG’, [price])> dollars - This means that will be take value from alias “price” and digital value will be written as a string. For example, if the price is 100, then the result of processing will be as “The price is one hundred dollars”.

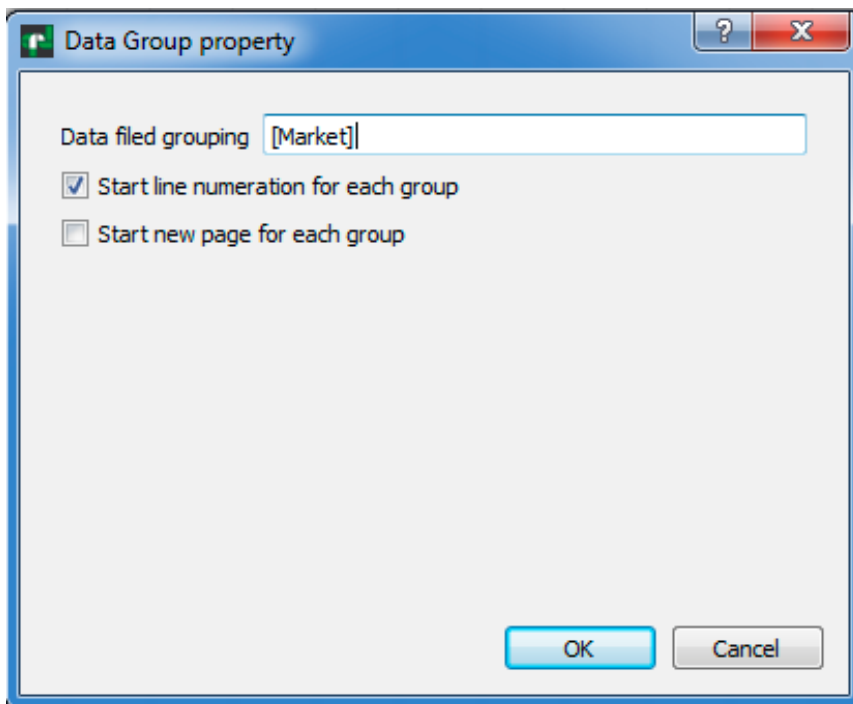
You can also use some system variables:

- Date
- Time
- Page
- TotalPages
- LineNo
- LineCount

During the report generations, the appropriate value will be inserted into the field. For example,

The page <Page> of <TotalPages> - Will generate: “The page 1 of 3”.

Data group property



Note for Codiac Users: In the “Data field grouping”, you should enter the alias of the field.

Barcode property

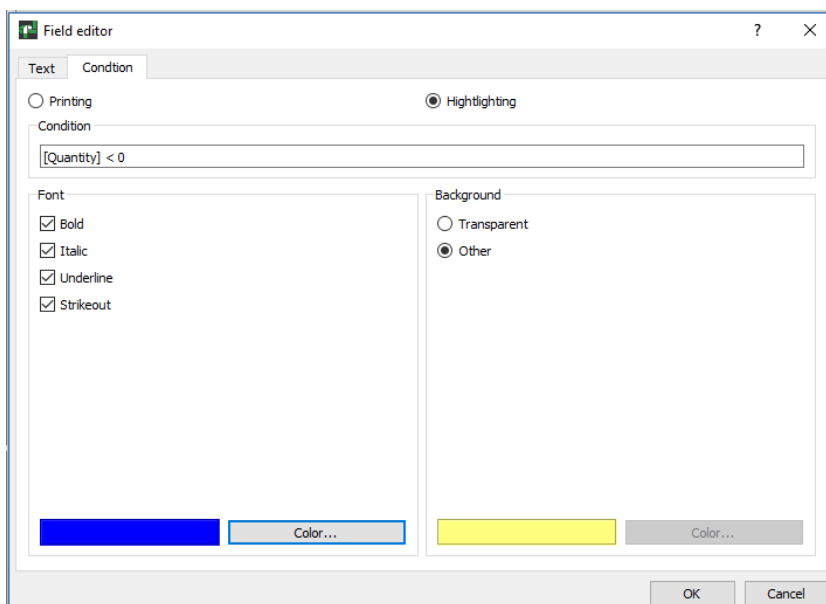
Please note: that since version 1.4.5, to use barcode feature, you need to have built QtZint library. The source files are in folder Zint-2.4.4. After building, place (QtZint.dll, QtZint.so file) library into the folder where your application able to find it.



Field highlighting

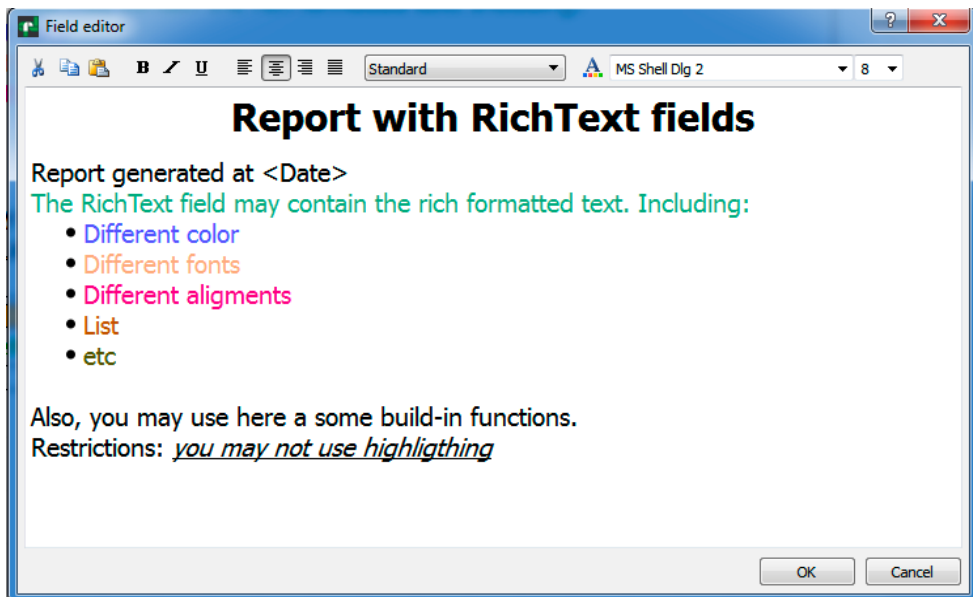
You can highlight the field depends of value. You should switch to “Condition” tab, click on “Highlight” and enter some condition, for example

[Condition] < 0, so when alias Condition will have value below zero, the font color will be blue and background color will be yellow in our example. You can set font Bold, Italic, underline or Strikeout.



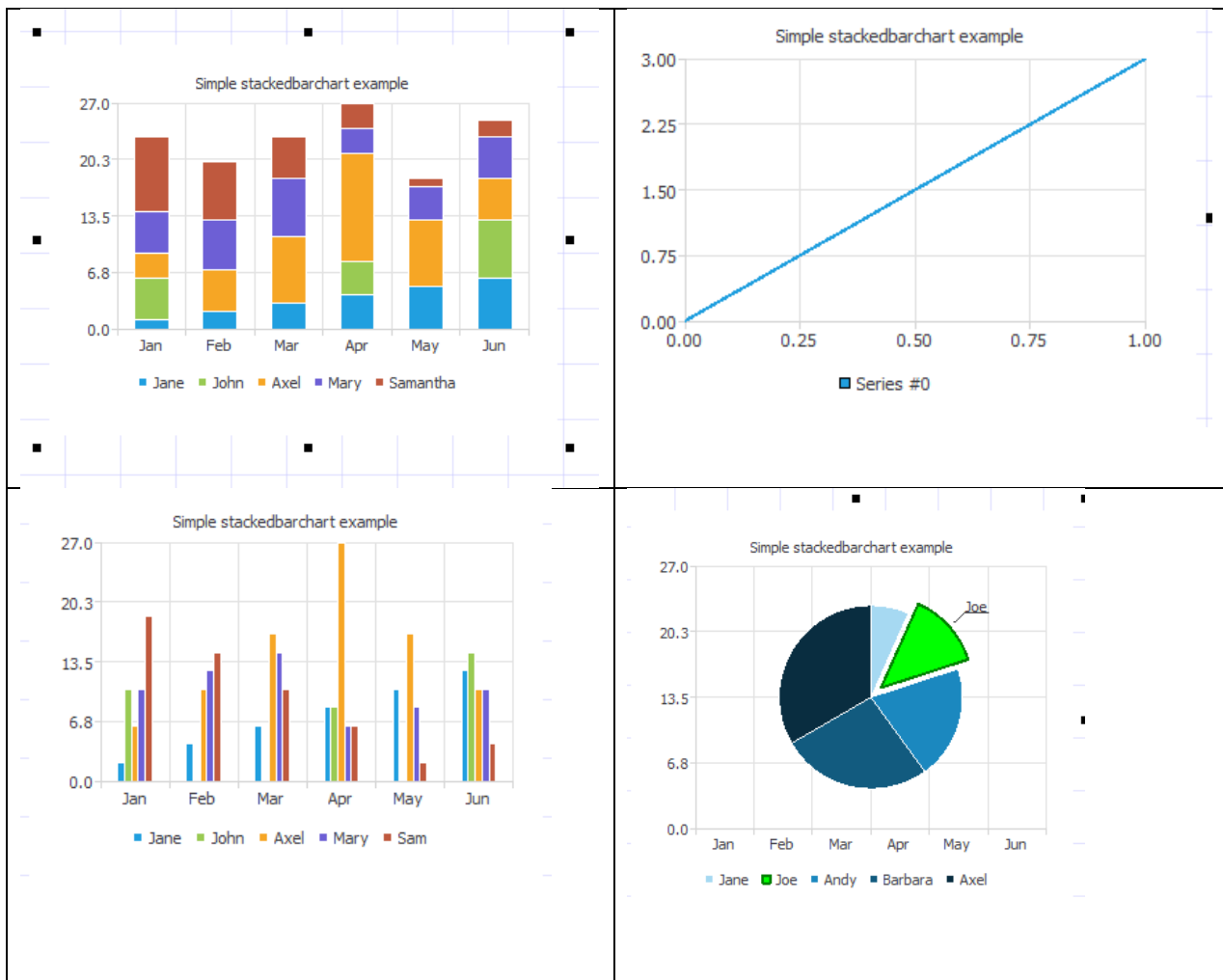
Rich Text field editor

To edit the field “Rich text”, double click on the chosen container. The dialog box in which you can bring the text will open, change its property. You may change font’s property such as Bold, Italic, Underline, change color of the text. This field also supports system variables and work with external data. However, this field doesn’t works with conditions.



Charts

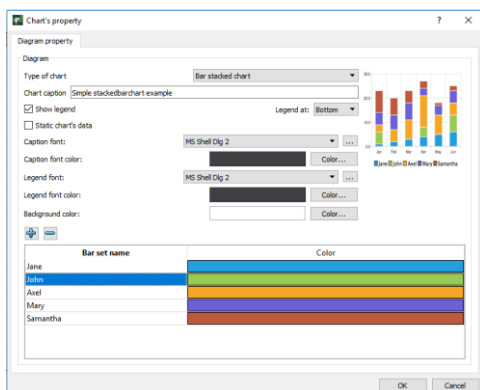
The QtRptDesigner allows to work with a different type of charts, the examples below.



To edit the properties of the chart, you should double click on the chart. Will be open the dialog with various fields to setup the properties of the chart. The chart can be static, i.e. value stay un-changed or dynamic, i.e. the values for diagram will get from data source.

Note for Codiac Users: For dynamic chart, you should enter the alias for data source of the chart (series).

For Stacked bar chart you should setup data source for each series. You can setup various font, font color, background color, captions, etc.



Script editor

Script editor allows to enter scripts, which will be executed during report building. Script allows to carry out the some control of process of report building. The following objects are available in the script engine.

QtRPT - name of QtRPT instance in the script engine

has following properties, members and functions

- `pageList` - list of the `RptPageObject`

RptPageObject - represents report page, and it has the following properties, and functions

- `setVisible(bool value)` - makes report report page visible/invisible

RptFieldObject

- `value` - value of the field, including variables

- `width` - width of the field

- `height` - height of the field

- `top` - top of the field

- `left` - left of the field

- `visible` - visibility of the field

- `rotate` - rotation of the field

- `fontColor` - color of the font

- `backgroundColor` - background color

The examples of some command:

To get the list of report pages

```
debug(QtRPT.pageList);
```

To get count of report pages

```
debug(QtRPT.pageList.length);
```

To get the report page name

```
debug(QtRPT.pageList[0].objectName);
```

Common functions

for debugging, you can use `print` or `debug` function

`print` in QtCreator word `TEST` + value of variable `[var1]`

```
debug('TEST:' + [var1]);
```

```
print('TEST:' + [var1]);
```

Common section

- not depends from order of the fields and evaluate as as report starts

checking the value of variable `var1` from user application

and makes report page visible/invisible depends of value.

Please note, that here only first value of variable will be taken

If you want to process each value of variable from data set, you must place such processing in the function of appropriate field

```
if ([var1] > 1000)
```

```
    QtRPT.pageList[0].setVisible(false);
```

else

```
QtRPT.pageList[0].setVisible(true);
```

To process field, you must define function with name field1 before getting data from user application

VERY IMPORTANT!

The name functions must be as nameOfField+BeforeData

or nameOfField+AfterData

BeforeData functions will be executed before start of processing appropriate field

AfterData function will be executed after getting data from user application

Below some examples of using

```
function field1BeforeData()
{
    // making the field visible/invisible depends of value
    if ([var1] > 1000) {
        field1.visible = false;
        field1.value = 'HELLO, I am invisible'; // assigning the value to the field
    } else {
        field1.visible = true;
    }
}
```

*/*Processing field with name field1 after getting data from user application*

Please note, that use After function preferably only for changing text value of the field

**/*

```
function field1AfterData(value)
{
    /*This function mainly intended for modifying of result string.
    DONT use the following settings in AFTER section
    Field1.visible = false;

    If you DONT want to change value string, just comment 'return'
    Here we modified the value that comes from user application and add
    the string to it*/
    field1.fontColor = QColor(255,0,0);
    return value + " It is corrected string";
}
```

*/*Here we rotate the field*/*

```
function field2BeforeData()
{
    var w = field2.width;
    var h = field2.height;
    field2.rotate = 3;
    field2.height = w;
    field2.width = h;
}
```

*/*Here we change the width of field3, set the same as field4 has*/*

```
function field3BeforeData()
{
    field3.value = 'We change width of the field';
}
```

```
    field3.width = field4.width;
}

/*Here we set the font and background color*/
function field4BeforeData()
{
    field4.value = 'We change the color of font and background';
    field4.fontColor = QColor(0,255,0);
    field4.backgroundColor = QColor(255,0,0);
}

/*Making field invisible*/
function field6BeforeData()
{
    field6.visible = false;
}
```

Plugins

The QtRptDesigner functionality can be enhanced via plugin technology. Some plugins can be automatically launched on QtRptDesigner start, some can be run via Main Menu -> Service -> Plugins -> Name of Plugin. Possible to define should plugin execute some function before report preview or not.

How to create custom plugin

To create custom plugin, you must inherit the CustomInterface like in the code below

```
#pragma once

#include <QObject>
#include <QtPlugin>
#include "CustomInterface.h"

class SessionPlugin : public QObject, CustomInterface
{
    Q_OBJECT
    Q_PLUGIN_METADATA(IID "qtrpt.project.CustomInterface")
    Q_INTERFACES(CustomInterface)
    Q_CLASSINFO("PluginName", "Plugin Name")
    Q_CLASSINFO("AddToMenu", "false")
    Q_CLASSINFO("RunOnLoading", "true")
    Q_CLASSINFO("ShowReport", "true")

public:
    explicit SessionPlugin(QObject *parent = 0);
    bool execute(QSharedPointer<QDomDocument> xmlDoc) override;
    void saveData(QSharedPointer<QDomDocument> xmlDoc) override;
    void showReport(QSharedPointer<QDomDocument> xmlDoc) override;
    void clear(QSharedPointer<QDomDocument> xmlDoc) override;
};
```

The following CLASSINFO parameters affect on behavior of plugin

- PluginName – Just name of plugin
- AddToMenu – Define, will or not this plugin added to Menu of plugins
- RunOnLoad – Define, will or not this plugin running on start of designer
- ShowReport – Define, will or not the plugins function “***showReport***” executes before preview of report in the designer

Description of public functions:

- execute – This function is calling when user select the name of plugin in Menu
- saveData – This function is calling before report saving
- showReport – This function is calling before report showing
- clear – This function is calling when the new report is created