

Report Sharp-Shooter Step-By-Step Reports Creation

Last modified on: June 30, 2011



PREFACE	4
SIMPLE REPORT	5
DATA BINDING.....	19
TABLE WIZARD	39
USING SCRIPTS	60
LIST.....	78
HEADER AND FOOTER.....	98
PAGE SIZE, MULTIPAGE TEMPLATE	120
BREAKING TEXT INTO PAGES.....	138
WATERMARK.....	152
MASTER-DETAIL REPORT.....	167
CHART	192
WIDGET	209
ADVANCED TEXT	222
BARCODE	237
PICTURE.....	260
SHAPE.....	281
STYLES	297
SORTING.....	321
FILTERING.....	342
GROUPING.....	360
HYPERLINKS AND BOOKMARKS.....	383
TOTALS.....	406



NESTED GROUP.....	426
JOINT USE OF GROUPING, SORTING, FILTERING AND TOTALS	450
MANAGING SIZE	483
REPORT WITHOUT BANDS	505
PARAMETERIZED REPORT	527
MULTICOLUMN REPORT	543
HORIZONTAL LIST	562
MATRIX	575
PIVOT TABLE	592
SUBREPORTS	615
MASTER REPORT	631
SIDE BY SIDE REPORT	649
USING MS CHARTS IN REPORT SHARP-SHOOTER 5.3+	663



Preface

This user guide contains instructions on how to create templates of the common reports using Report Sharp-Shooter. Each article is a step by step description of the report template design process.

This user guide is prepared by Perpetuum Software team for Report Sharp-Shooter users.

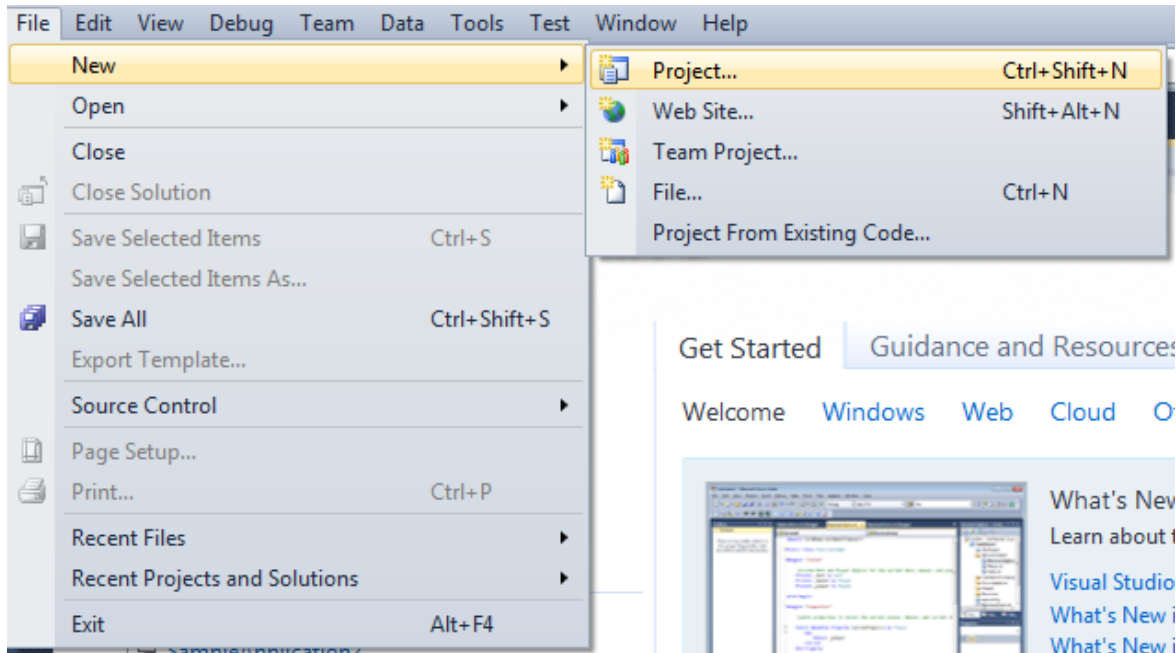


Simple Report

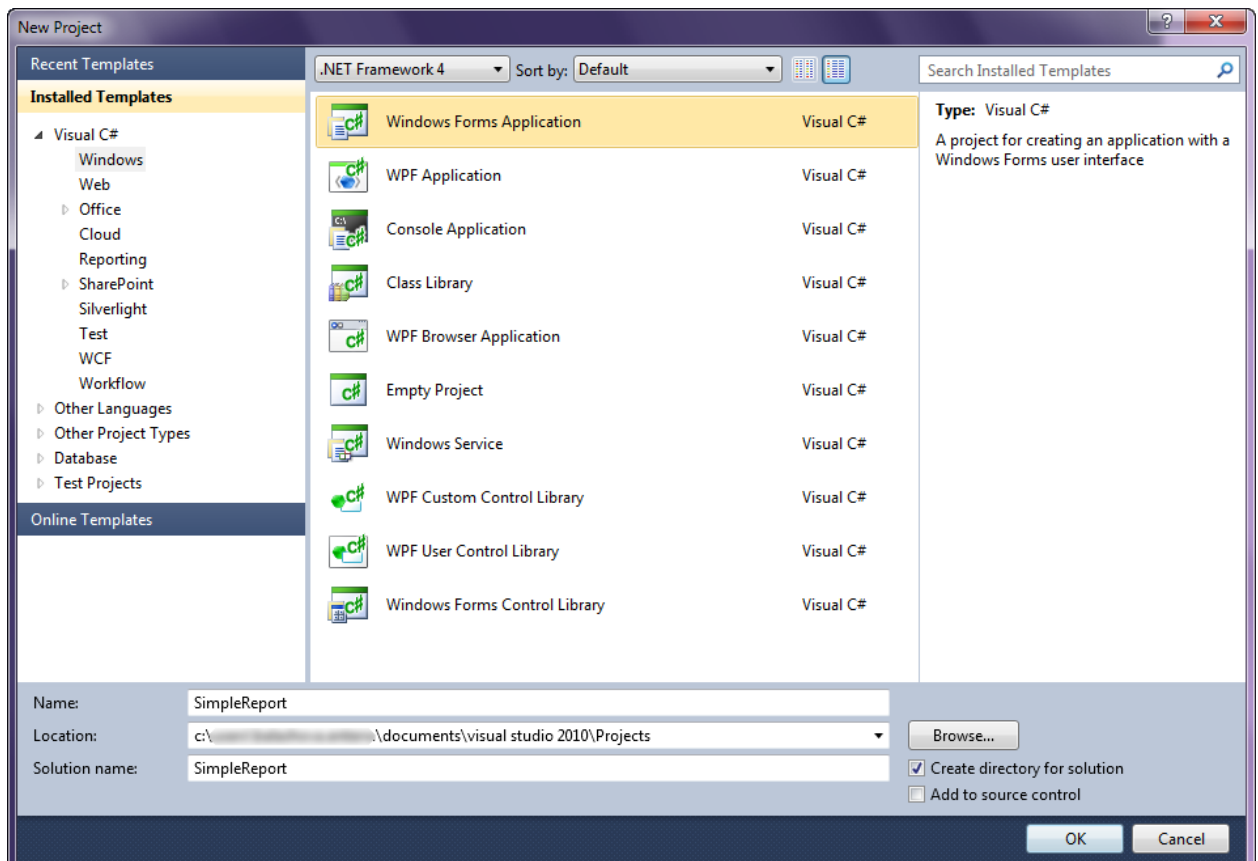
Template of the report displaying current date, logo and a note.

Step 1

Create new project in Microsoft Visual Studio environment. Select item New\Project from the main menu.

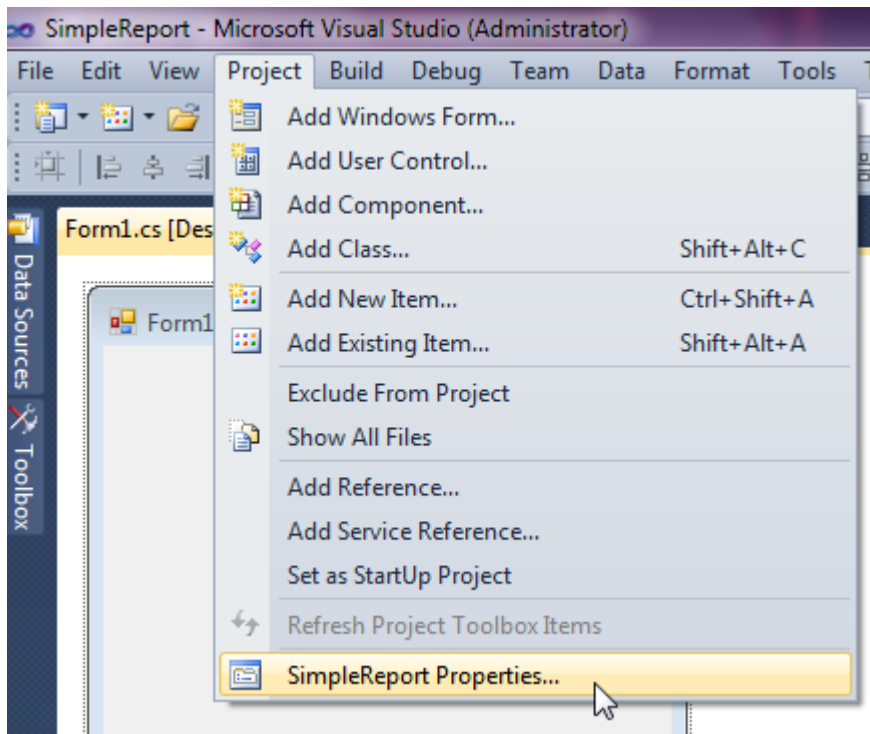


Select Windows Forms Application, set name of the project – “SimpleReport”, set directory to save the project to.

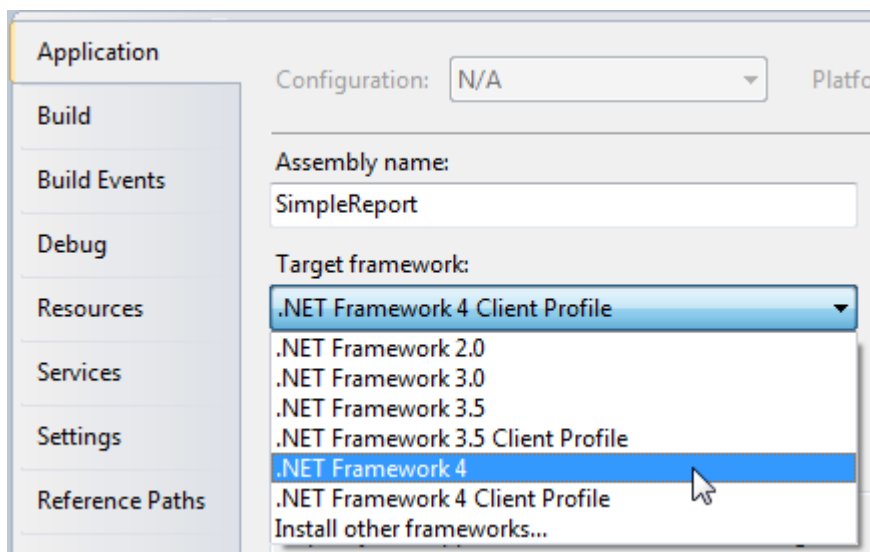


Step 2

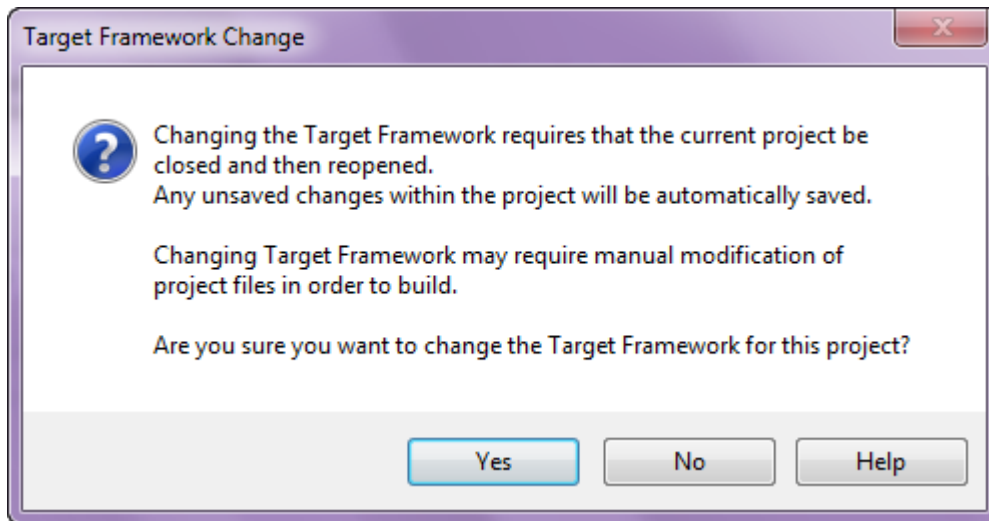
Change the project properties. Select the Project\SimpleReport Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

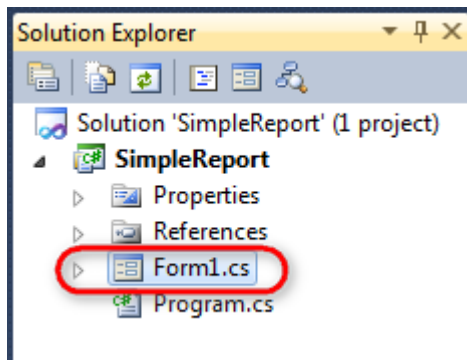


Press the "Yes" button in the opened window.

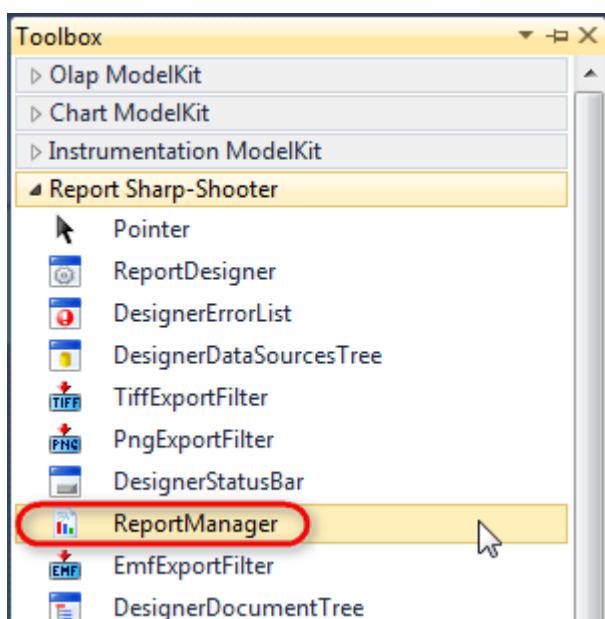


Step 3

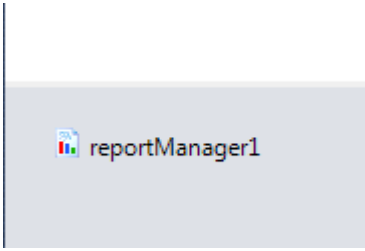
Open main form of the application in the editor by double click on the "Form1.cs" in the Solution Explorer.



Drag and drop "ReportManager" element from the Toolbox. This element stores collections of report templates and data sources.

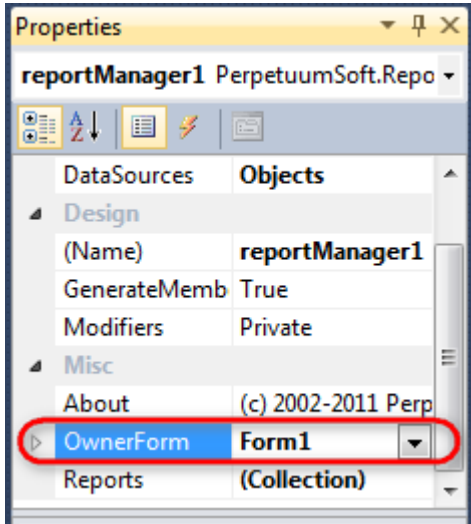


The component is displayed in the lower part of the window.



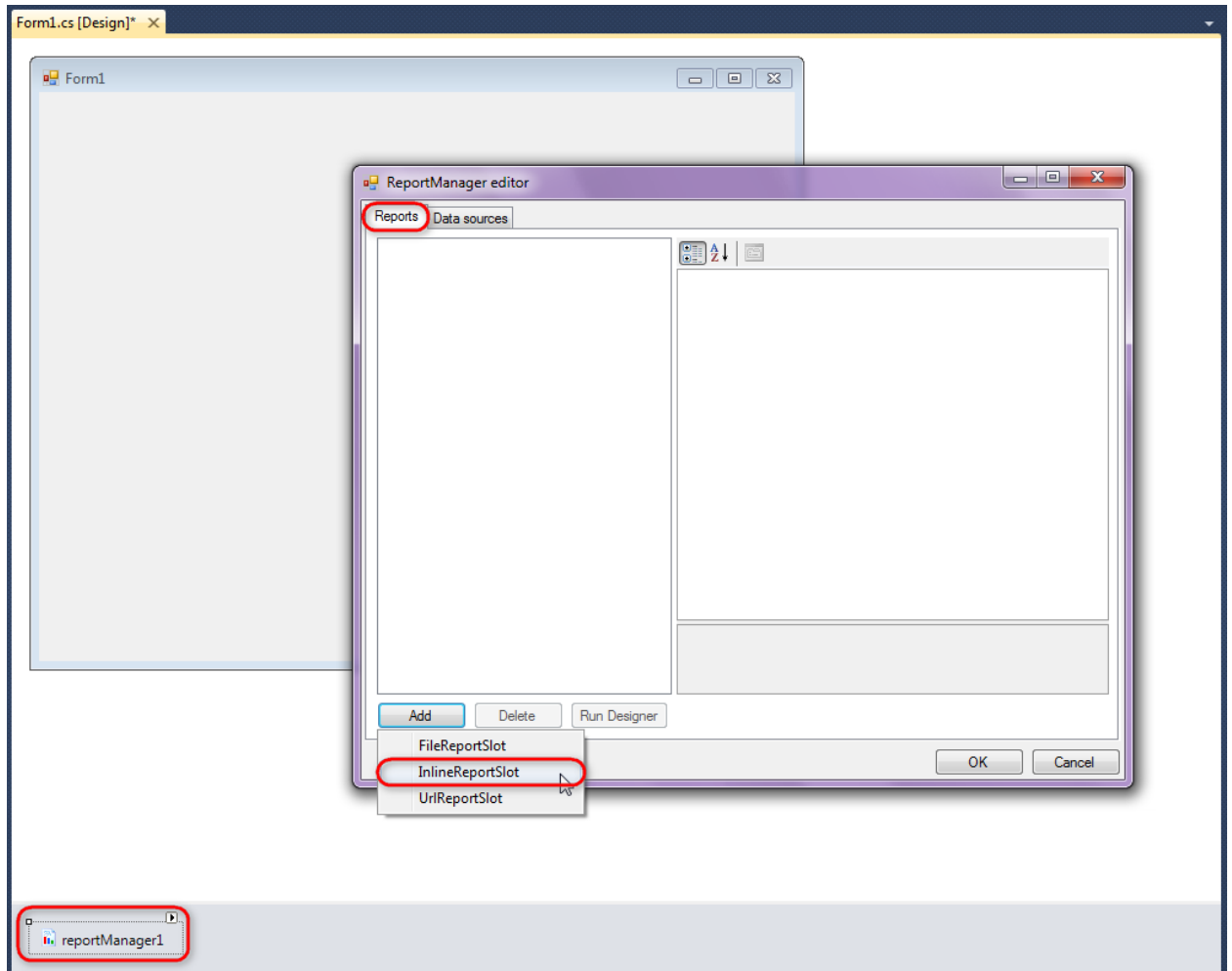
Step 4

In the property grid, initialize the property OwnerForm of the ReportManager component by selecting the form it is located on.



Step 5

Double click on the reportManager1 to open ReportManager editor.

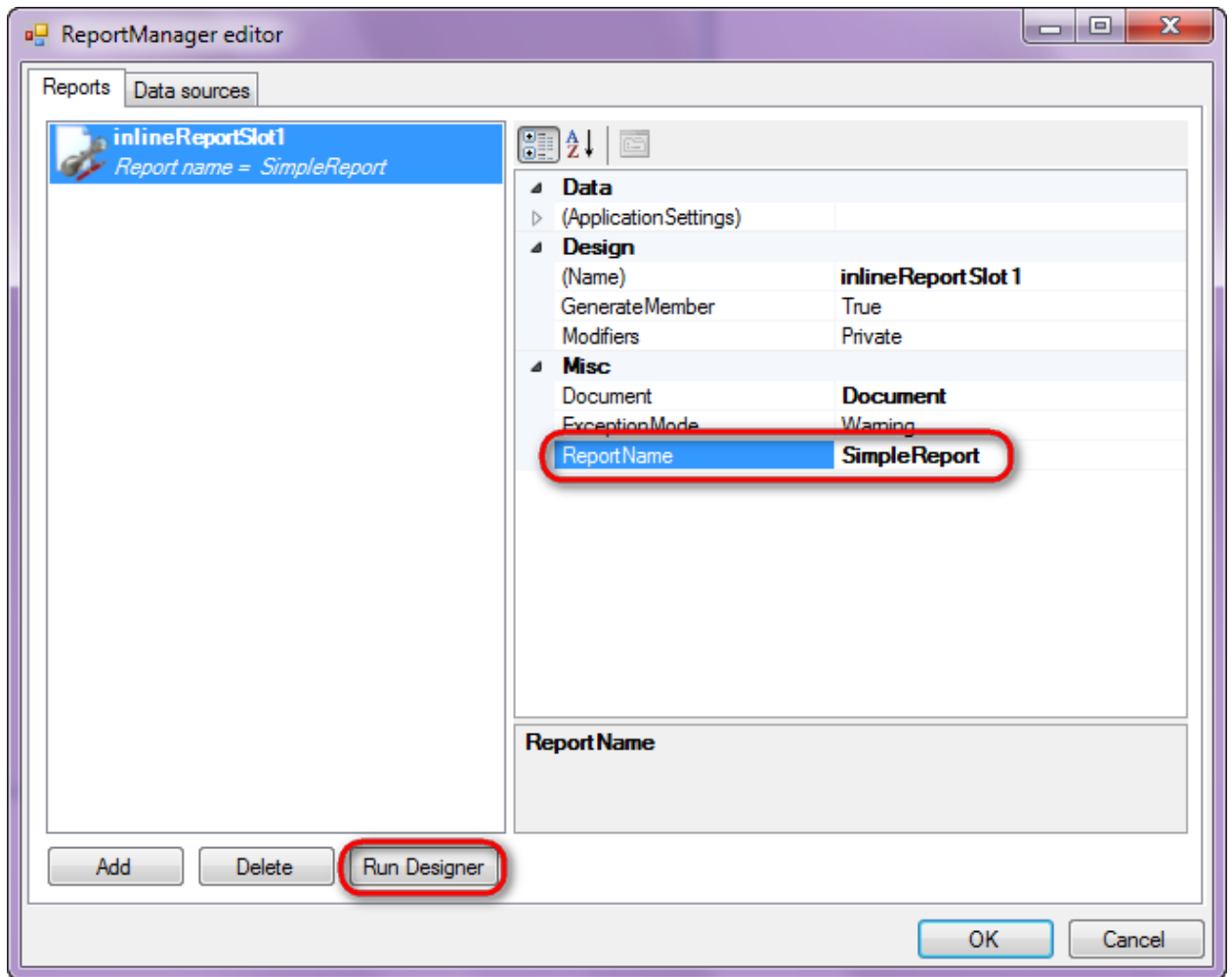


On the "Reports" tab, press button "Add" and select "InlineReportSlot".

Step 6

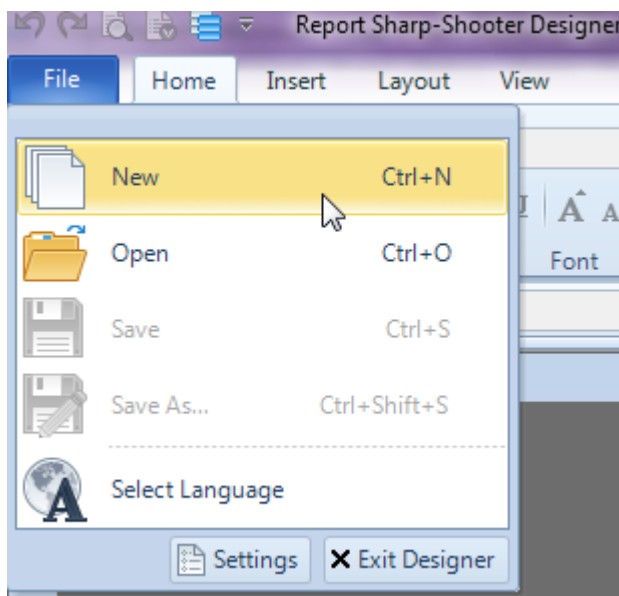
Set report name in the property ReportName – "SimpleReport".

Press button "Run Designer" in order to open Report Designer.

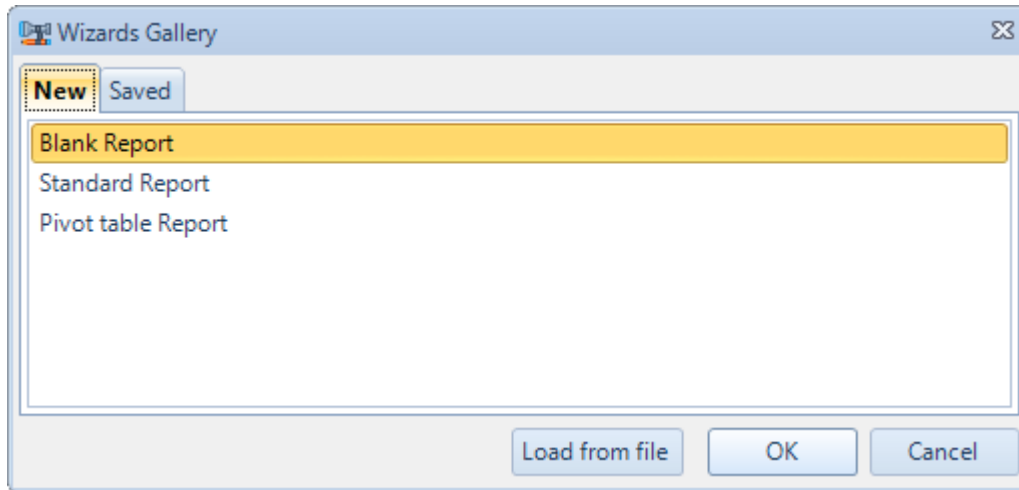


Step 7

Create new empty template – select item File\New from the main menu.

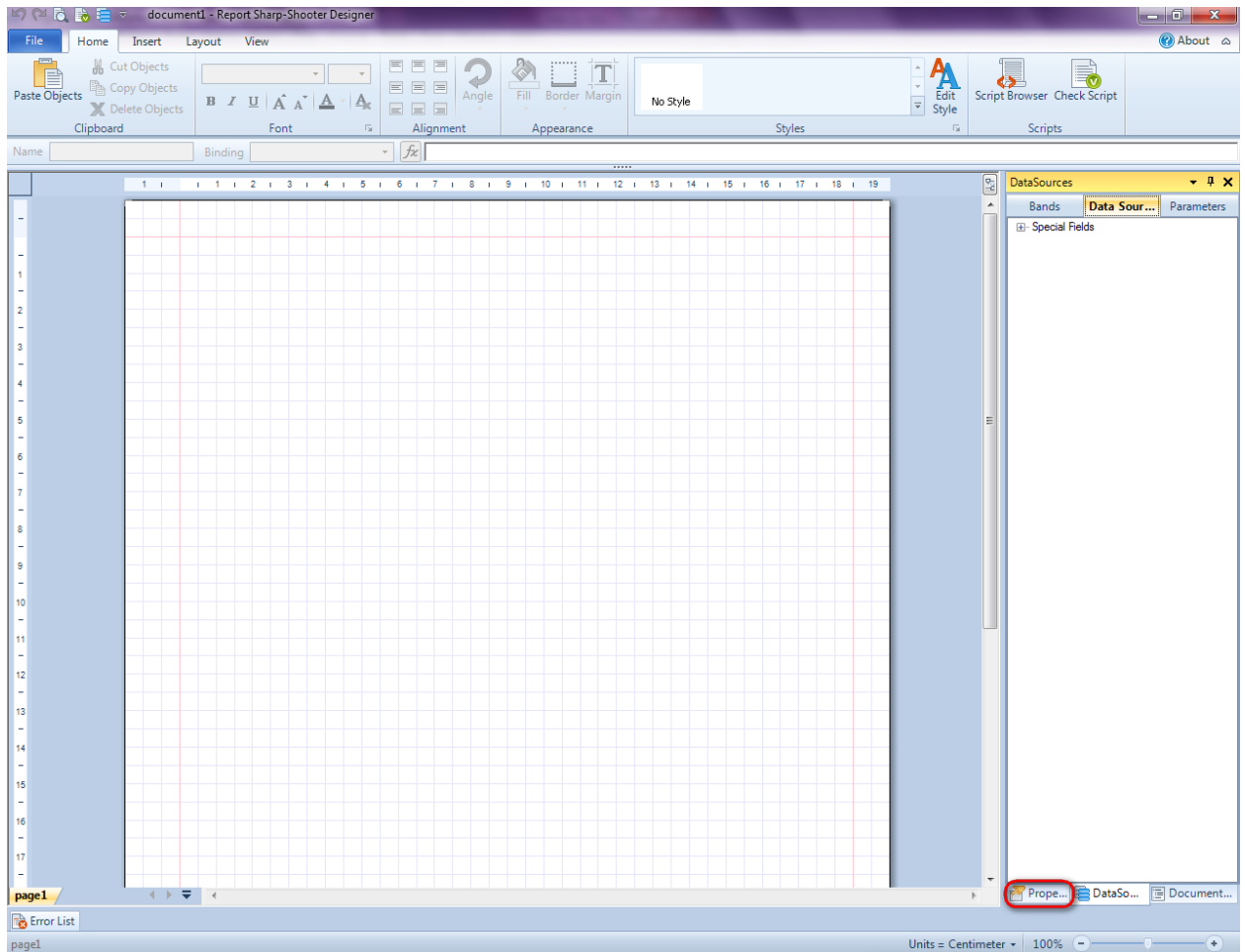


Select "Blank Report" in the Wizards Gallery and click "OK".



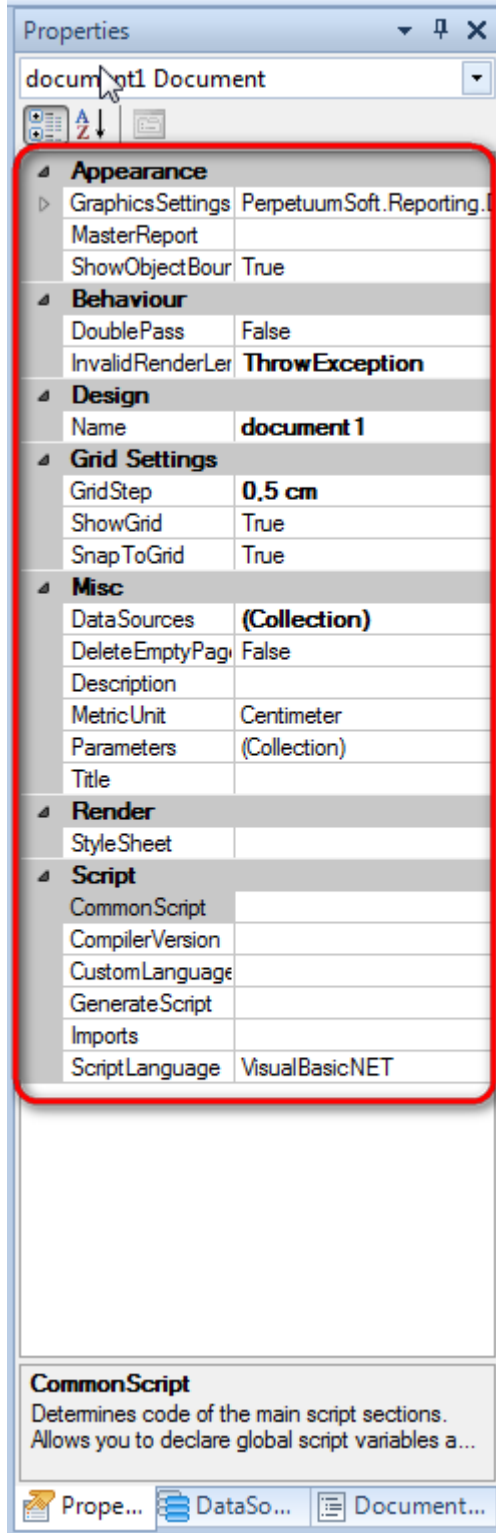
Step 8

Click the "Properties" tab of the tool window in the right part of the designer.

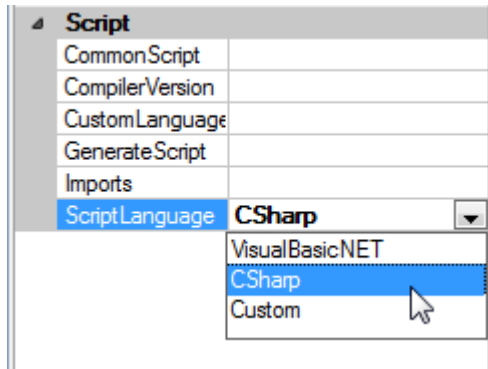




You will see properties of the edited template on the "Properties" tab

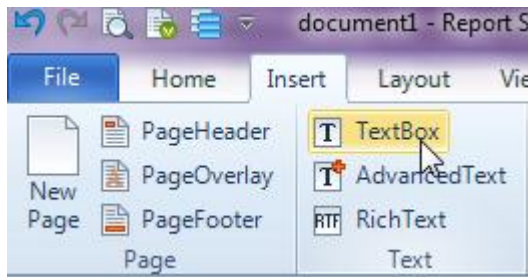


Set property ScriptLanguage = CSharp.



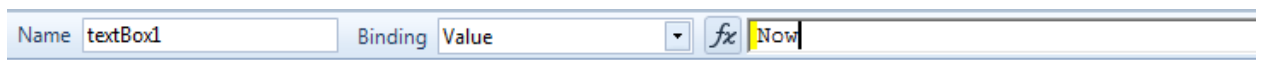
Step 9

Press button "TextBox" on the Insert tab in the group Text.

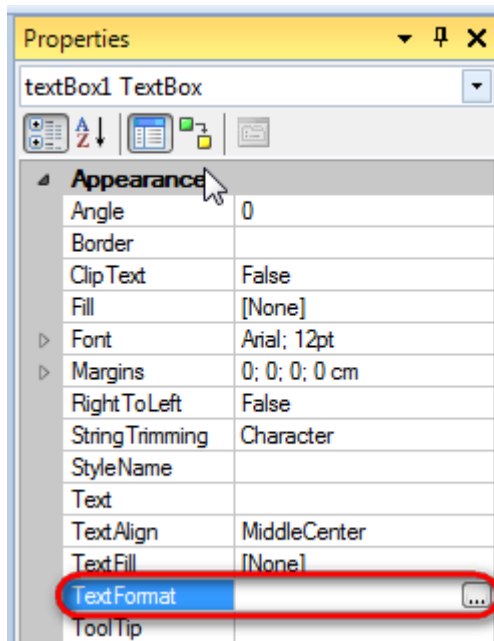


Click on the template area to add TextBox element.

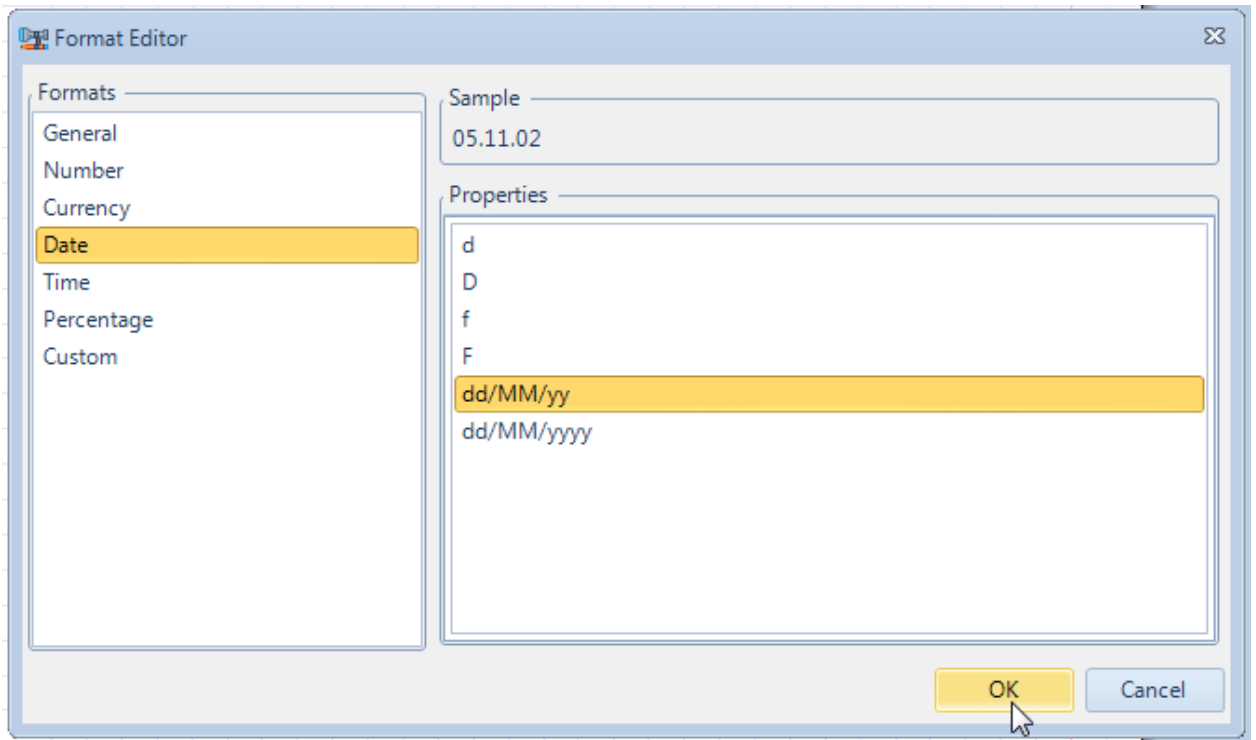
Set property Value to Now.



Select property TextFormat, press button  in order to open Format Editor.

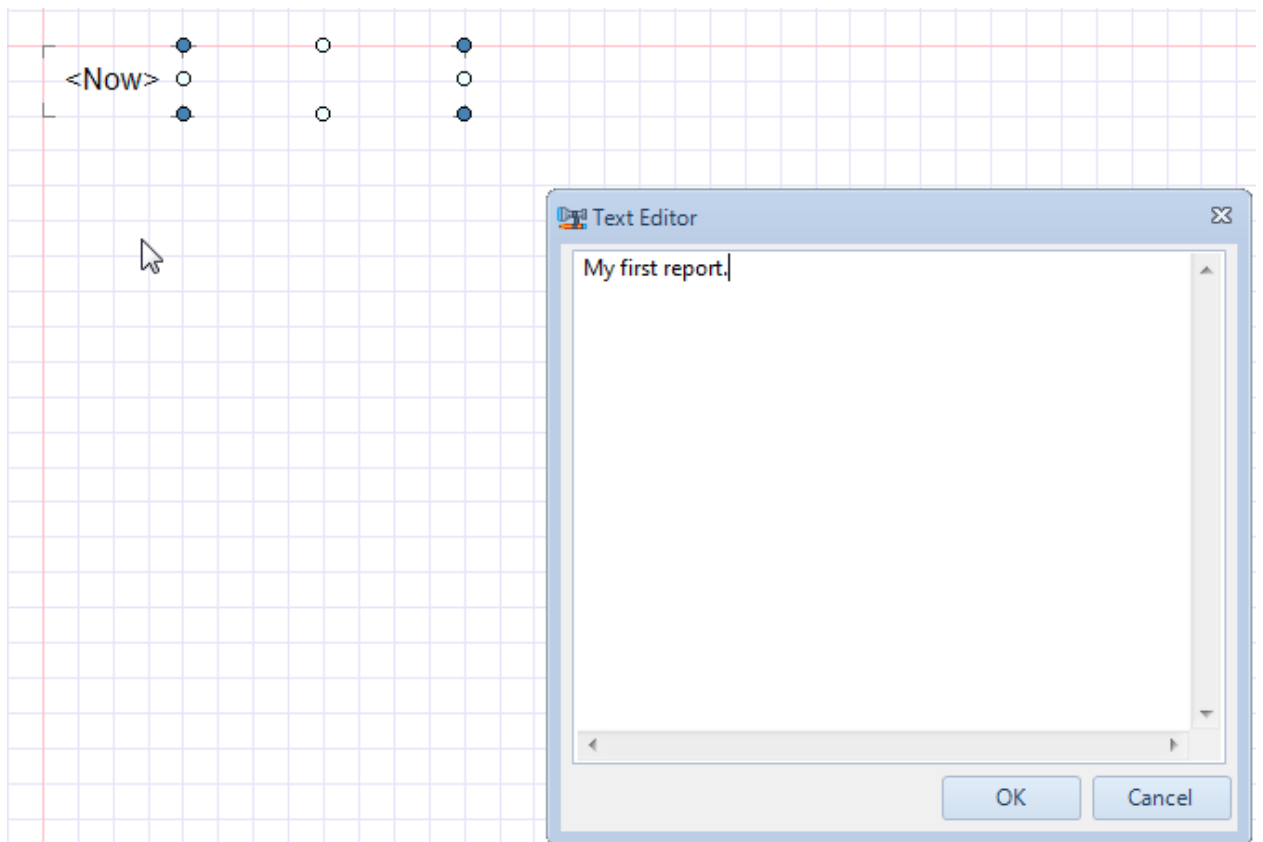


Select "Date" in the "Formats" list and "dd/MM/yy" in the "Properties" list.



Step 10

Add one more TextBox to the template. Double click on the element on the report template and open Text Editor (editor of the property Text). Write "My first report." in the editor.



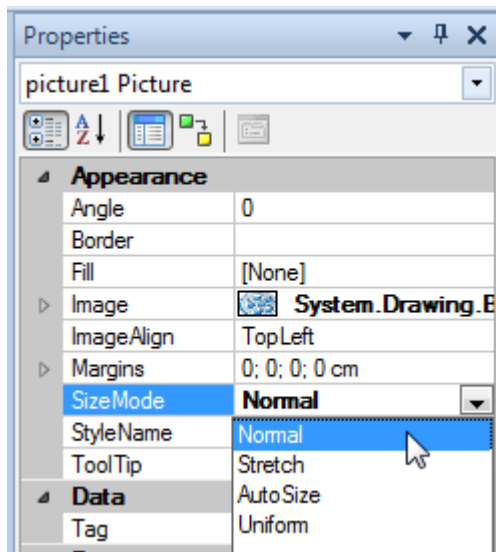
Step 11

Press "Picture" button on the Insert tab in the group Illustration.

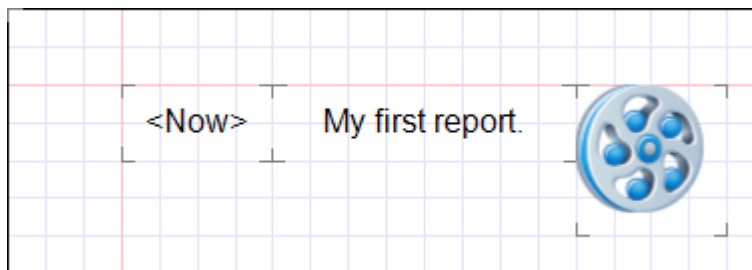


Click on the template area to add Picture element.

Double click Picture element on the template and open dialog to set path to the picture. Select an image with a logo and press "Open". Set property SizeMode = Normal.



Report template:

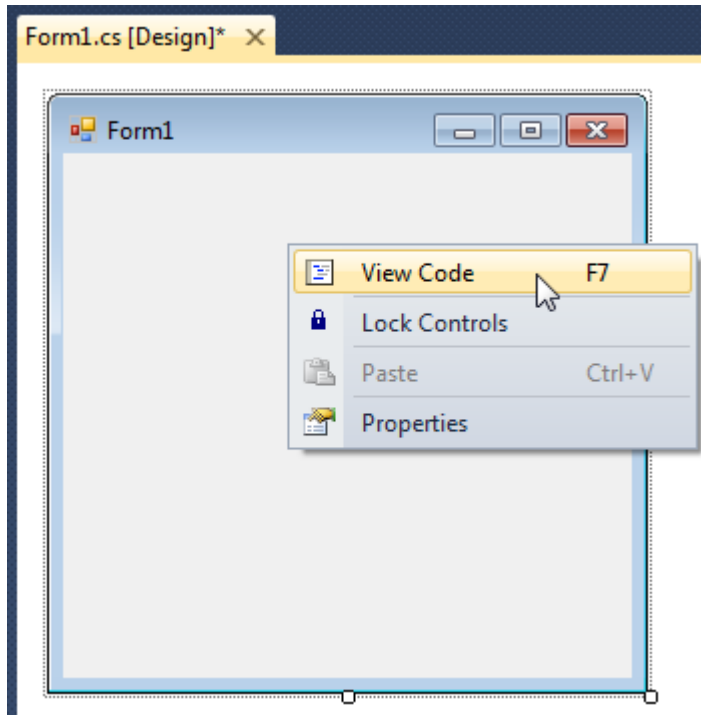


Step 11

Save report template and close Report Designer.

Step 12

Right click on the form and select "View Code" in the context menu in order to view code.

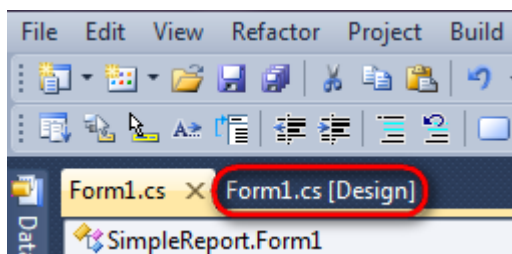


Add the following code to the class constructor in order to display report. Write the RenderCompleted event handler of the InlineReportSlot object.

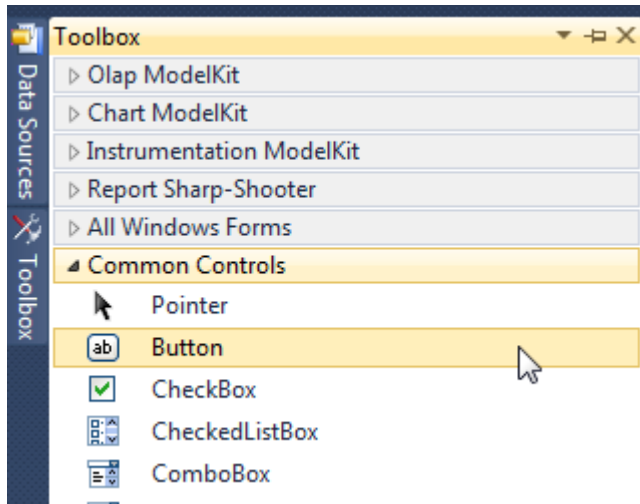
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 13

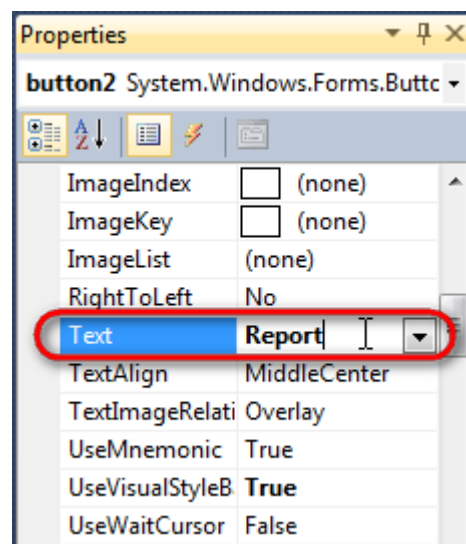
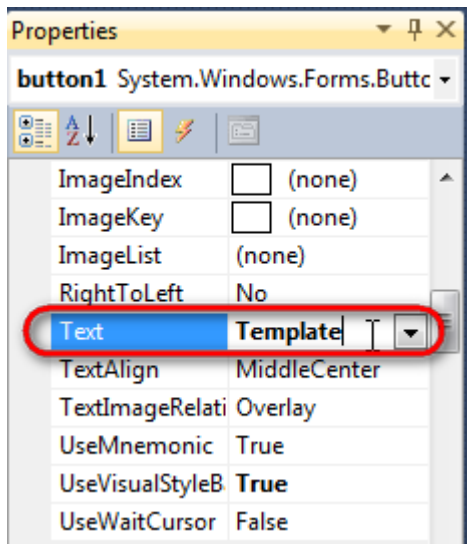
Get back to the application form by pressing the "Form1.cs[Design]" tab.



Place two buttons on the form (drag and drop "Button" element from the Toolbox).



Select Button element on the form and edit its Text property in the property grid. Set Text = Template for one button and Text = Report for the other one.



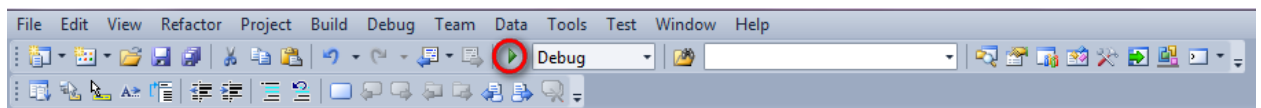
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, you can use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

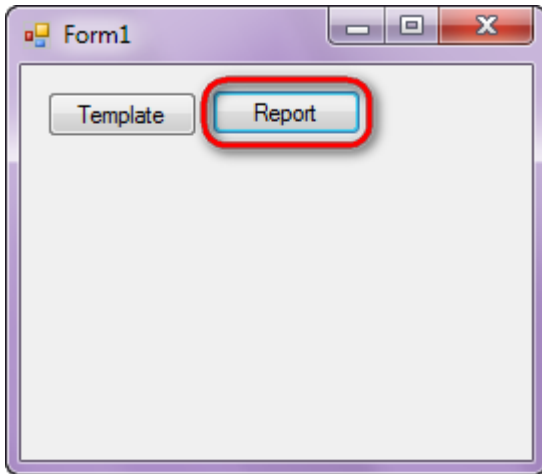
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 14

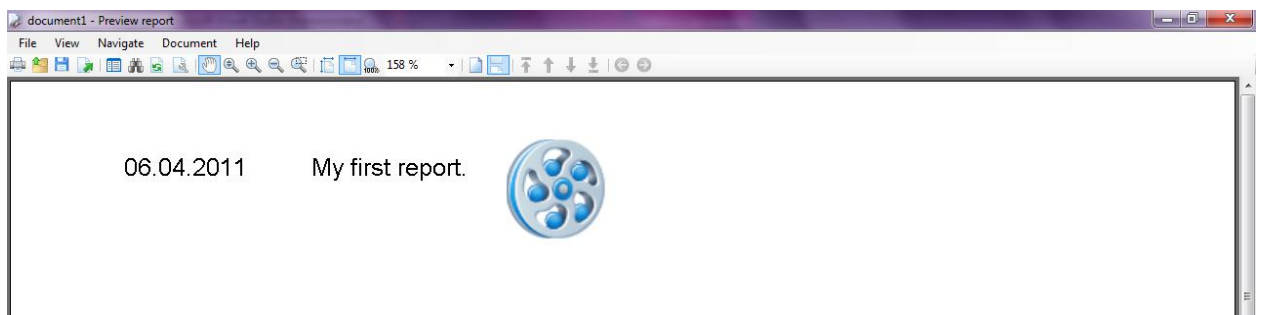
Press button "Start Debugging" on the Visual Studio toolbar in order to start application.



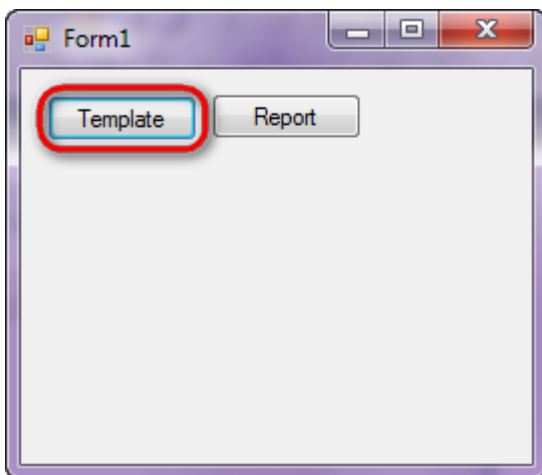
Click the "Report" button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



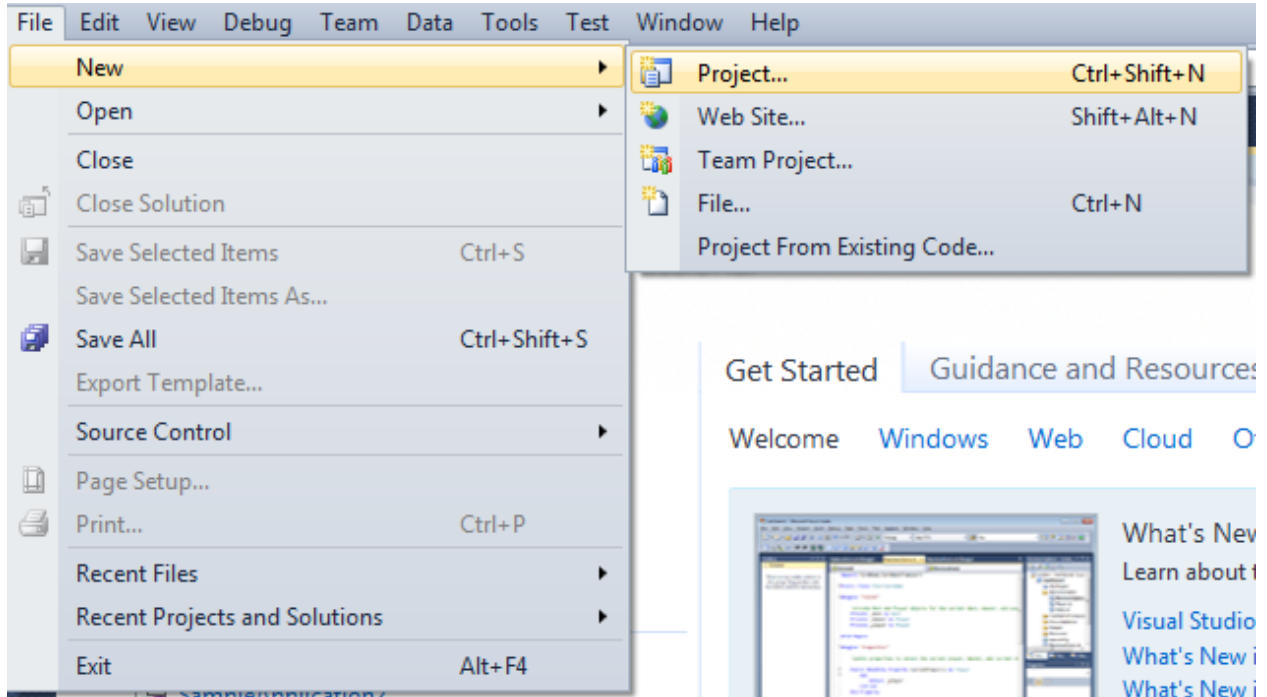


Data Binding

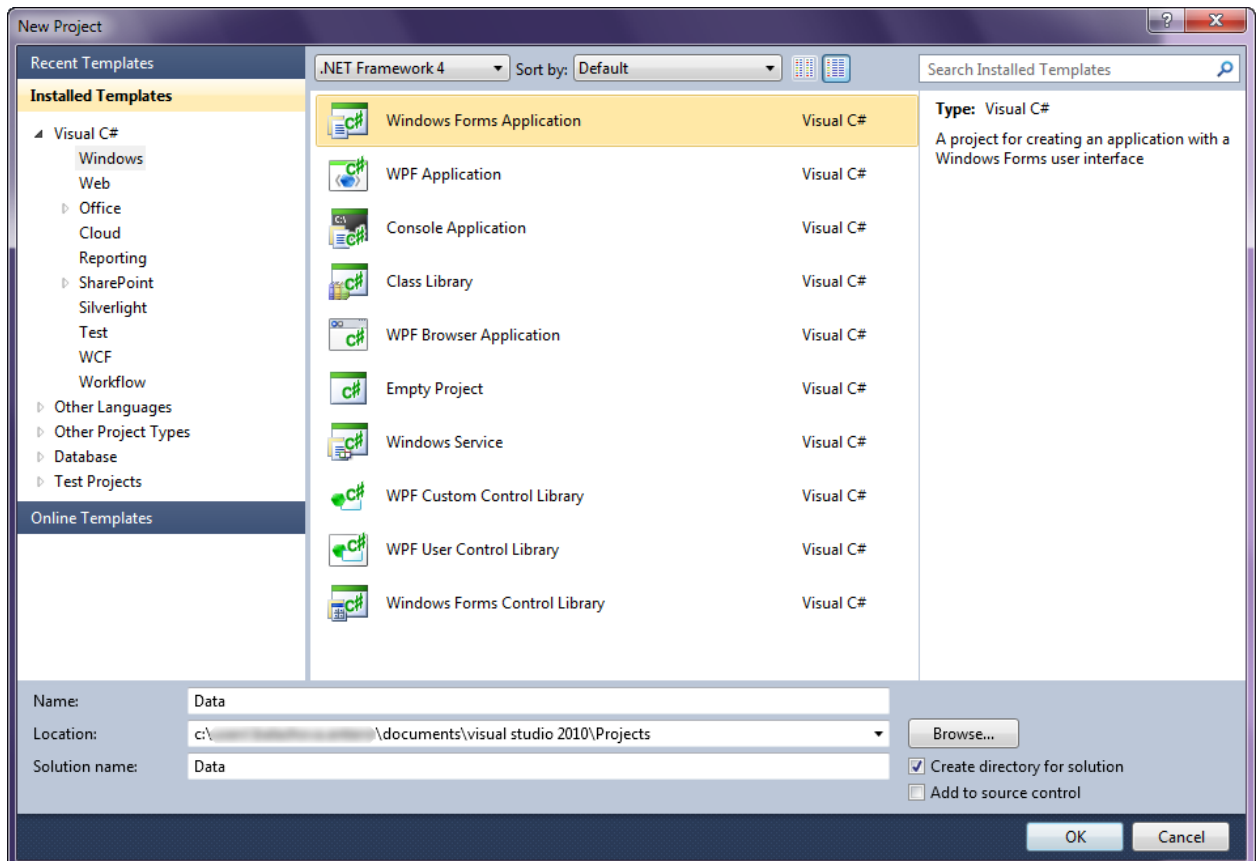
Template of the report displaying information on the employee; data is loaded from the data source.

Step 1

Create a new project in Microsoft Visual Studio. Select New\Project from the main menu.



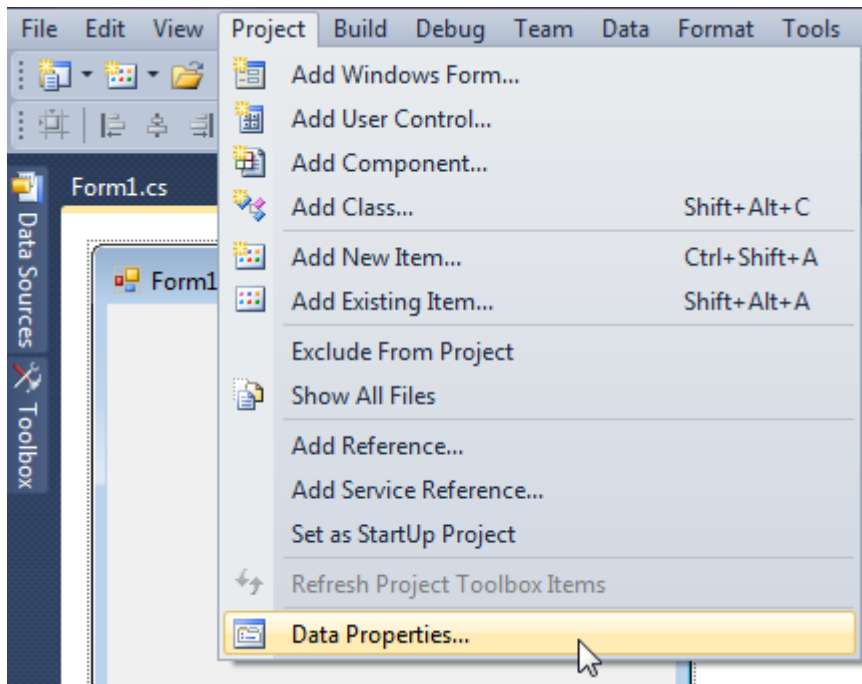
Select Windows Forms Application, set name of the project – “Data” and set directory to save the project to.



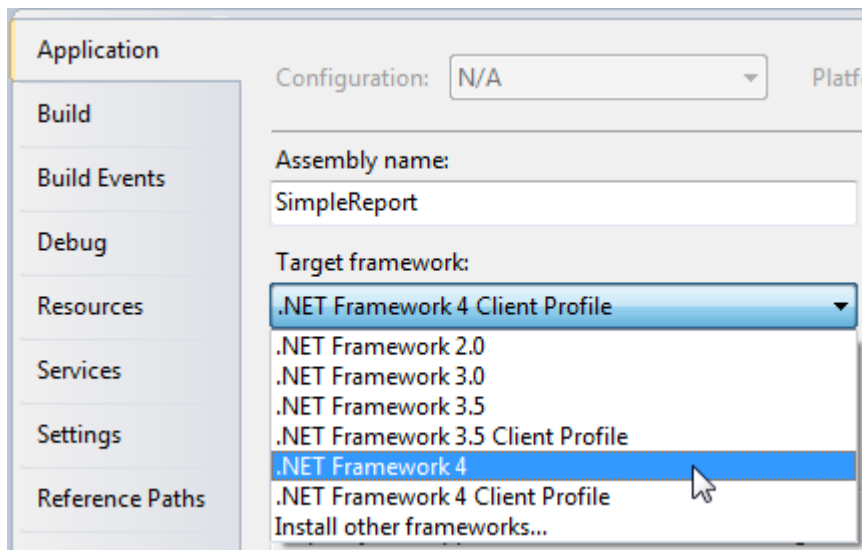


Step 2

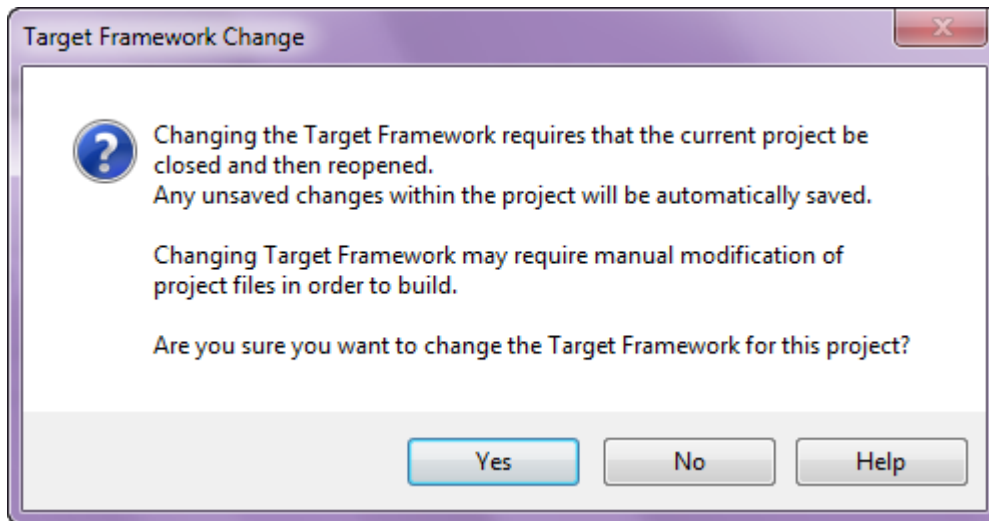
Change the project properties. Select the Project\Data Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

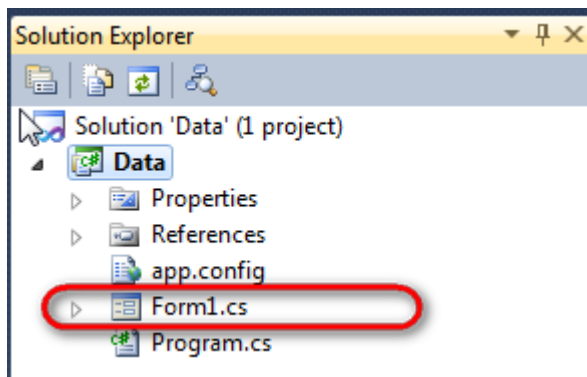


Press the "Yes" button in the opened window.

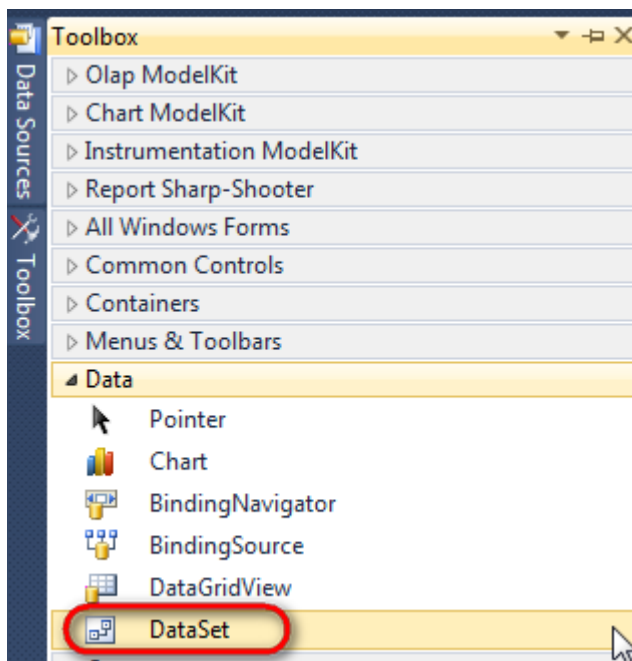


Step 3

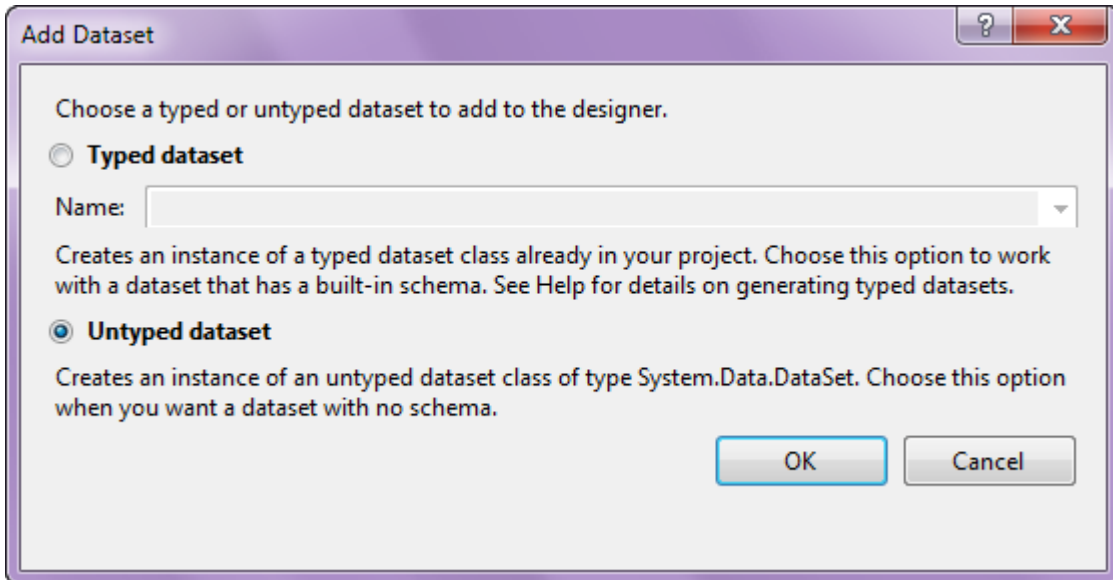
Open main form of the application in the editor by double click on "Form1.cs" in the Solution Explorer.



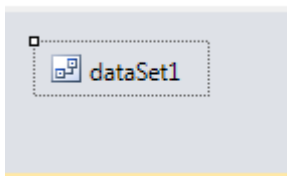
Click "DataSet" element in the Toolbox and place it onto the form.




Select "Untyped dataset", click "OK".

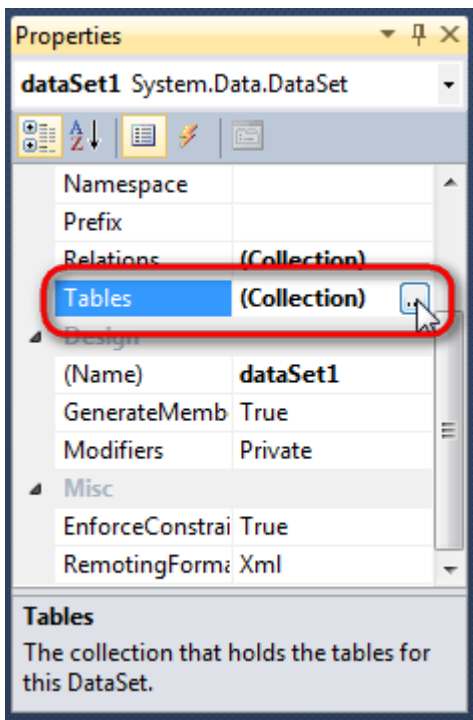


The component is displayed in the lower part of the window.

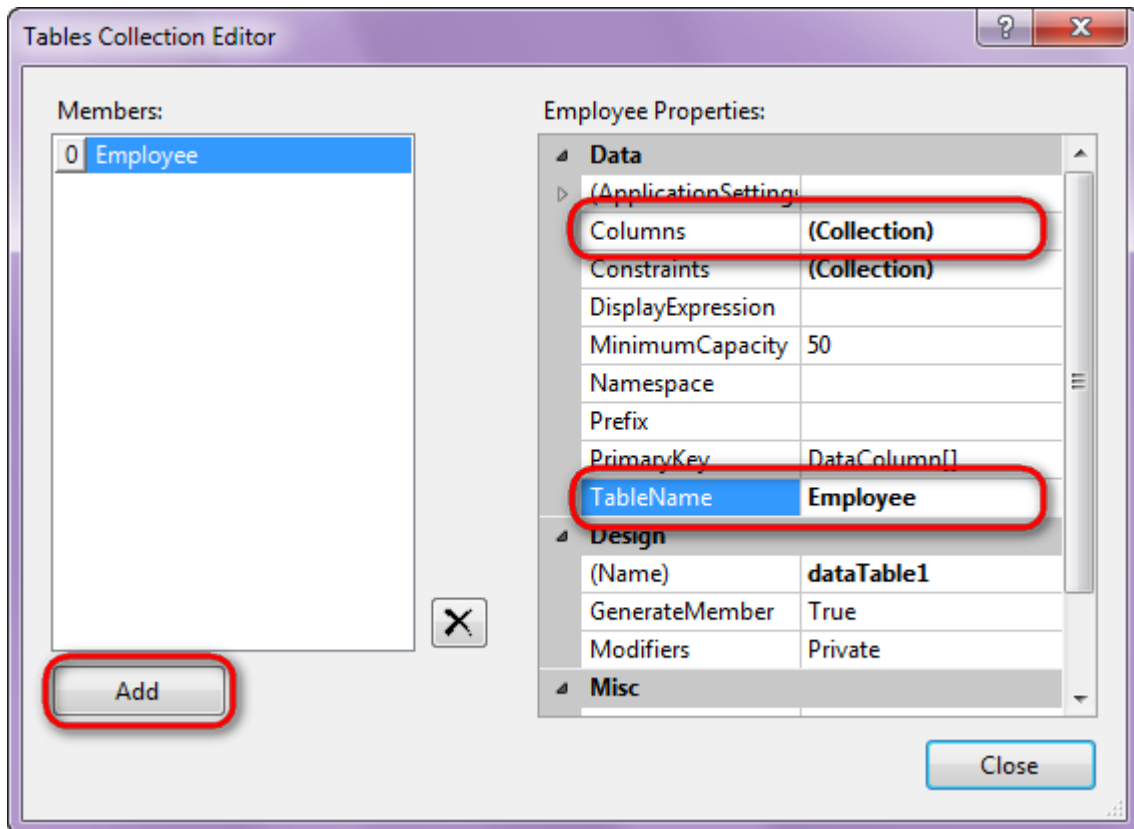


Step 4


Select dataSet1 component in the form editor. On the property grid, select "Tables" property, press button  in order to open Tables Collection Editor.

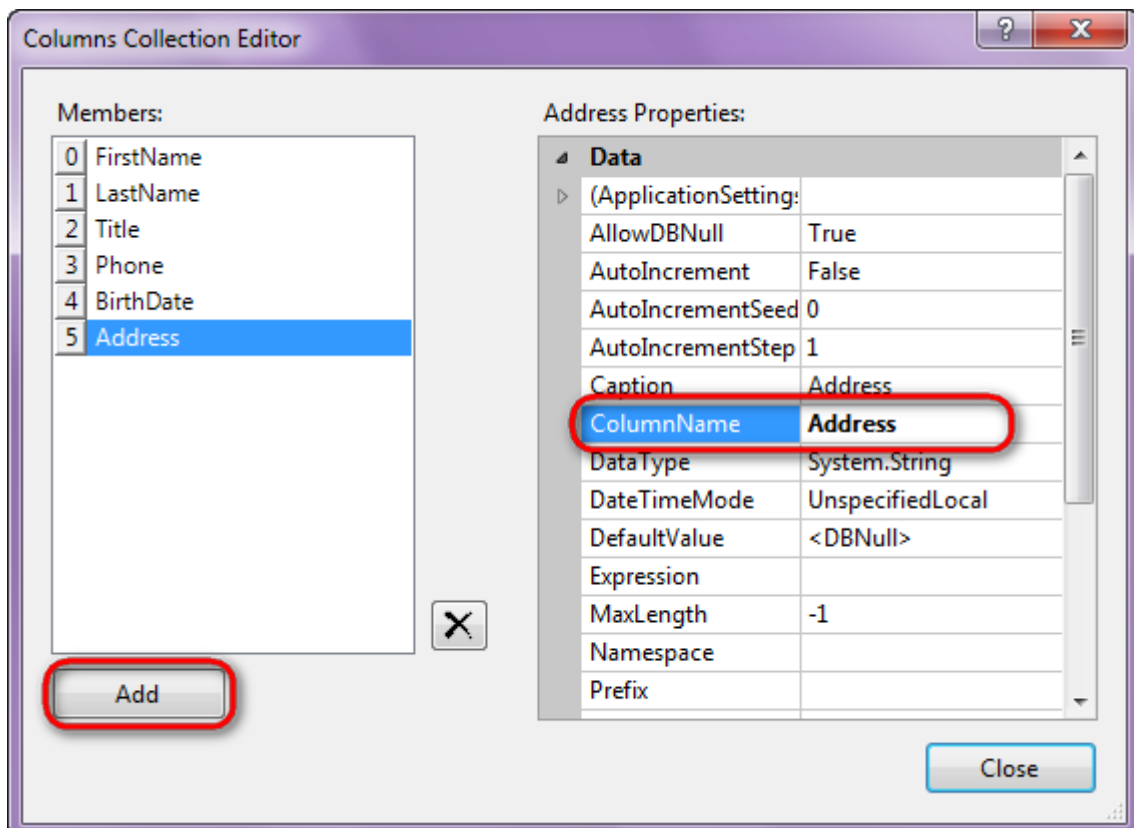


Press button "Add" in order to add a table. Set property TableName = Employee.



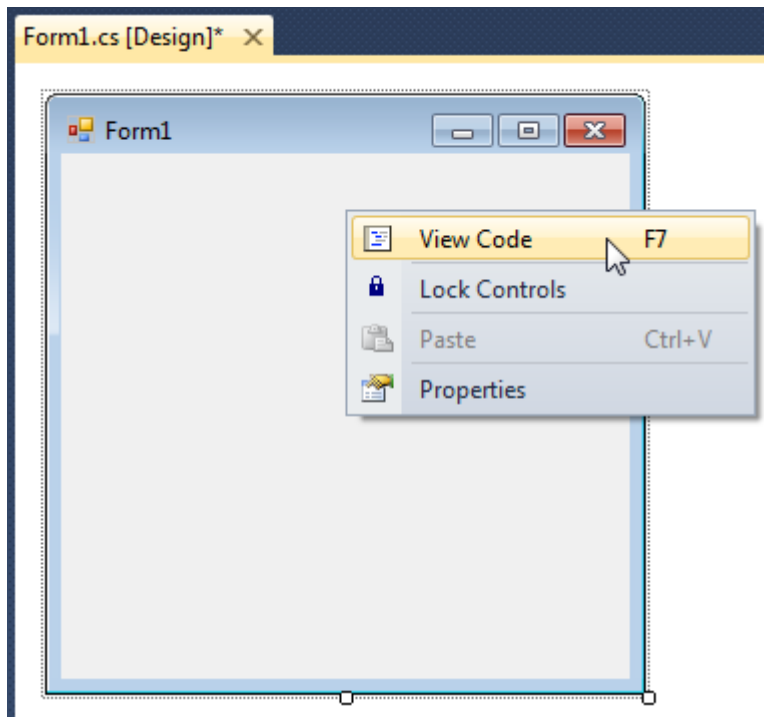
Step 5

Select "Columns" property, press button  in order to open property editor. Press "Add" to add a new column. Add four columns. Set value of the property ColumnName to "FirstName", "LastName", "Title", "Phone", "BirthDate", "Address" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu in order to view code.

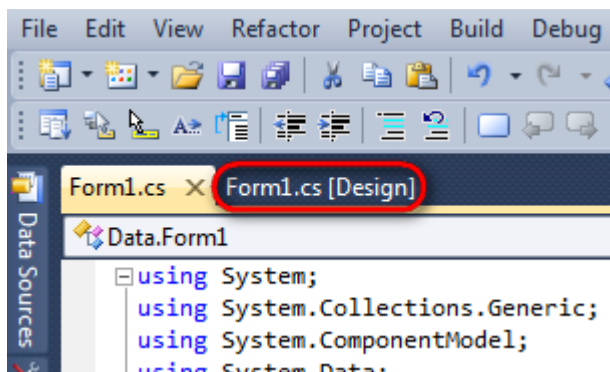


To fill a data source, add the following code to the class constructor:

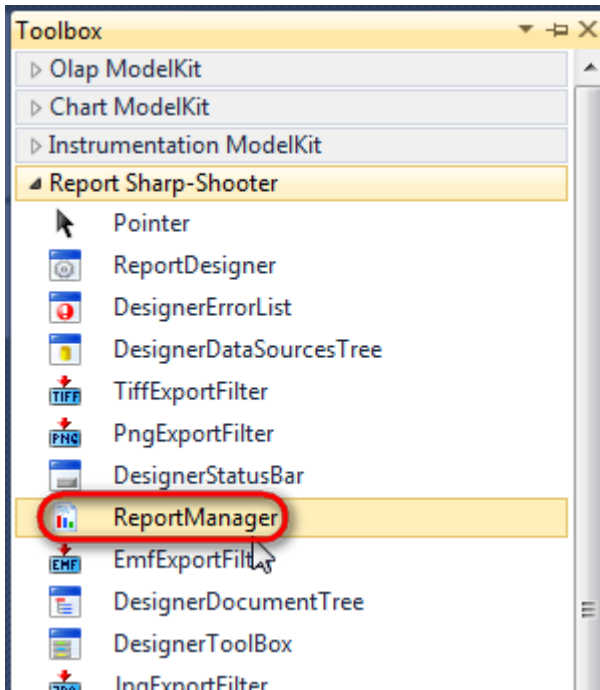
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["FirstName"] = "Robert";
    row["LastName"] = "King";
    row["Title"] = "Sales Representative";
    row["Phone"] = "(71) 555-5598";
    row["BirthDate"] = "29.05.1960";
    row["Address"] = "UK, London, Edgeham Hollow Winchester Way";
    dataTable1.Rows.Add(row);
}
```

Step 7

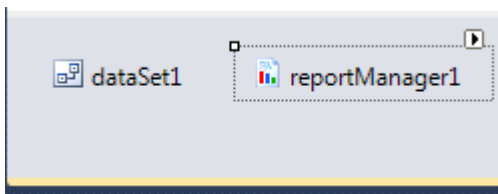
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click "ReportManager" element on the Toolbox and place it onto the form. This element is designed to store collections of report templates and data sources.

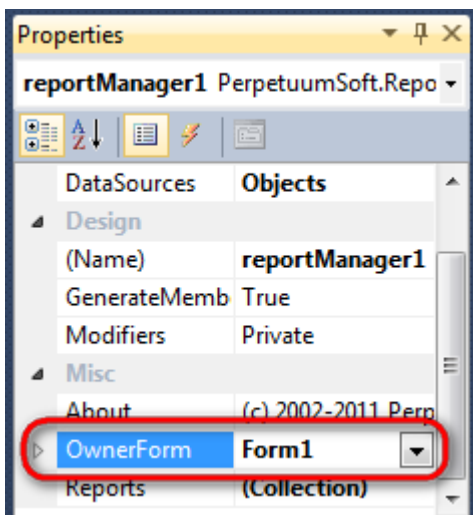


Component is displayed in the lower part of the window.



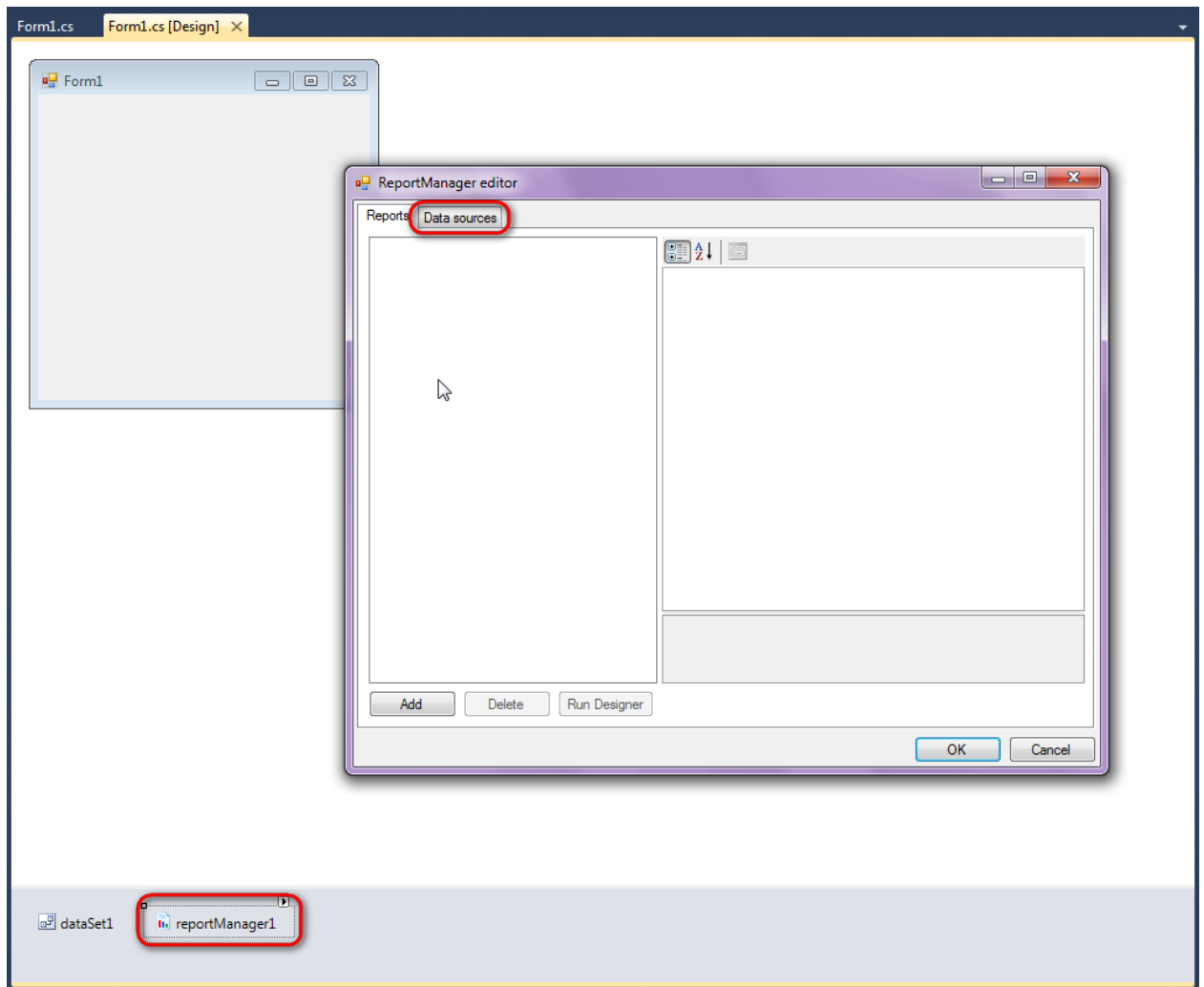
Step 8

On the properties grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

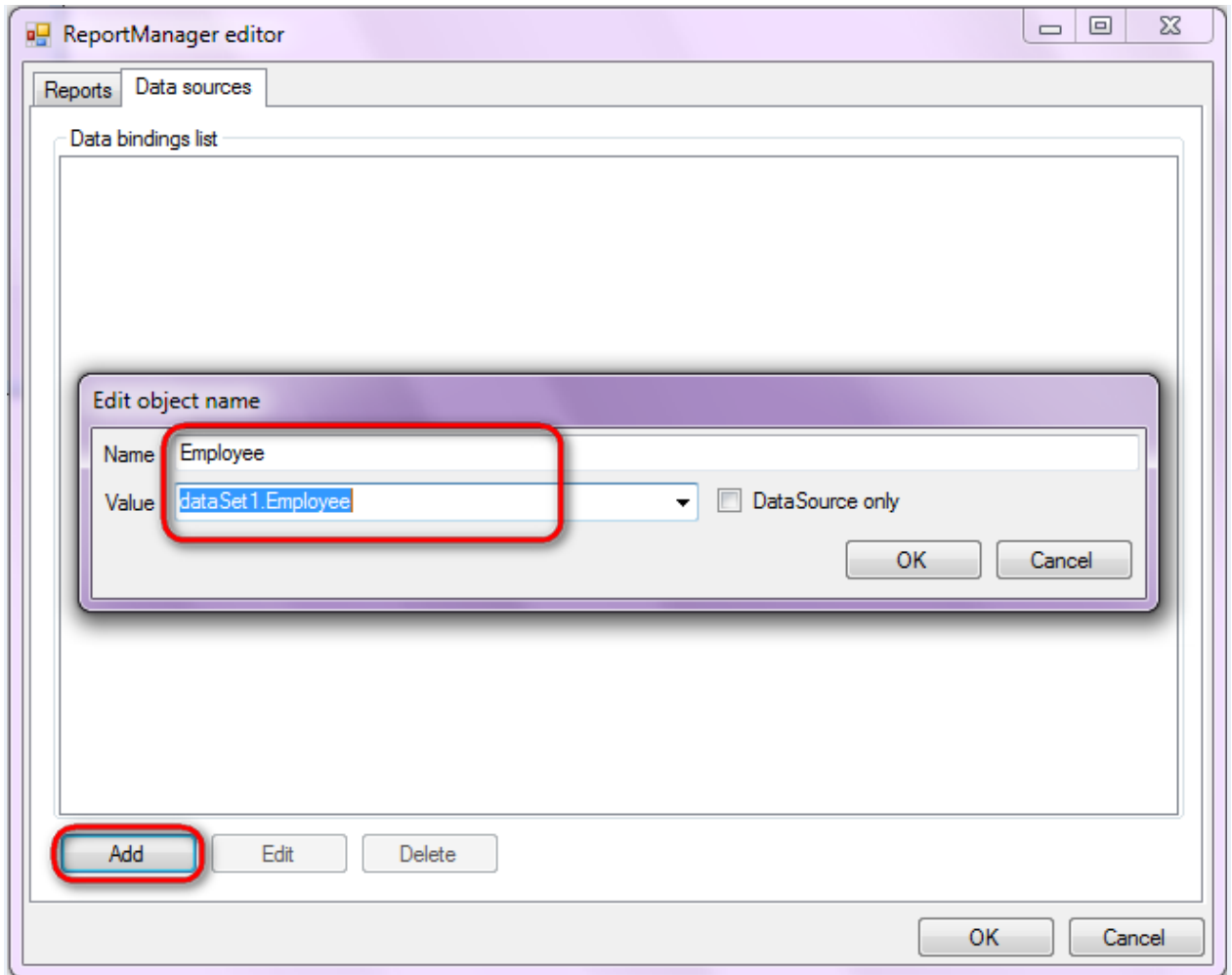


Step 9

Double click on the ReportManager component and open ReportManager editor.

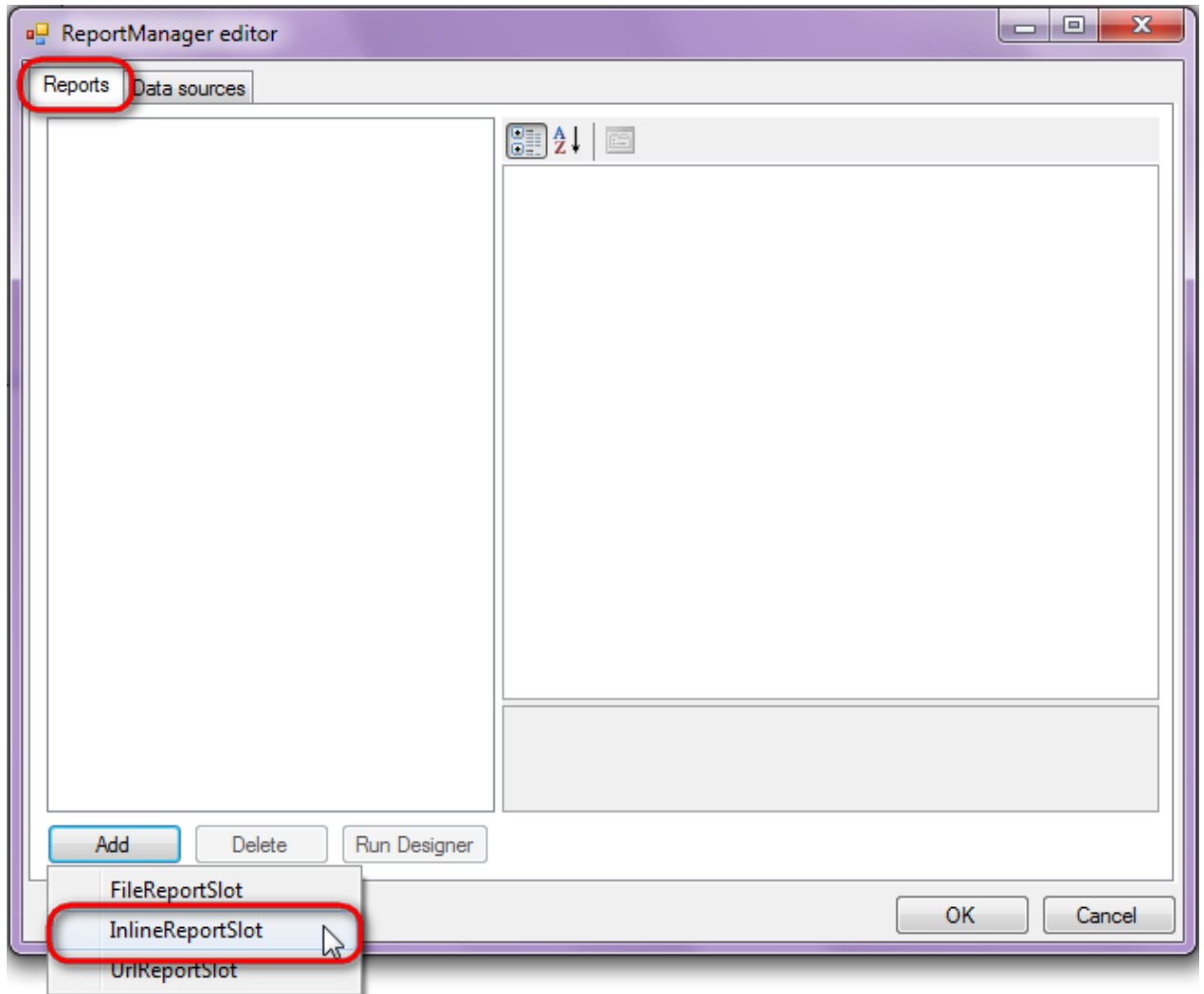


Go to "Data sources" tab, press "Add", set name of the data source - "Employee", select data source value - "dataSet1.Employee".



Step 10

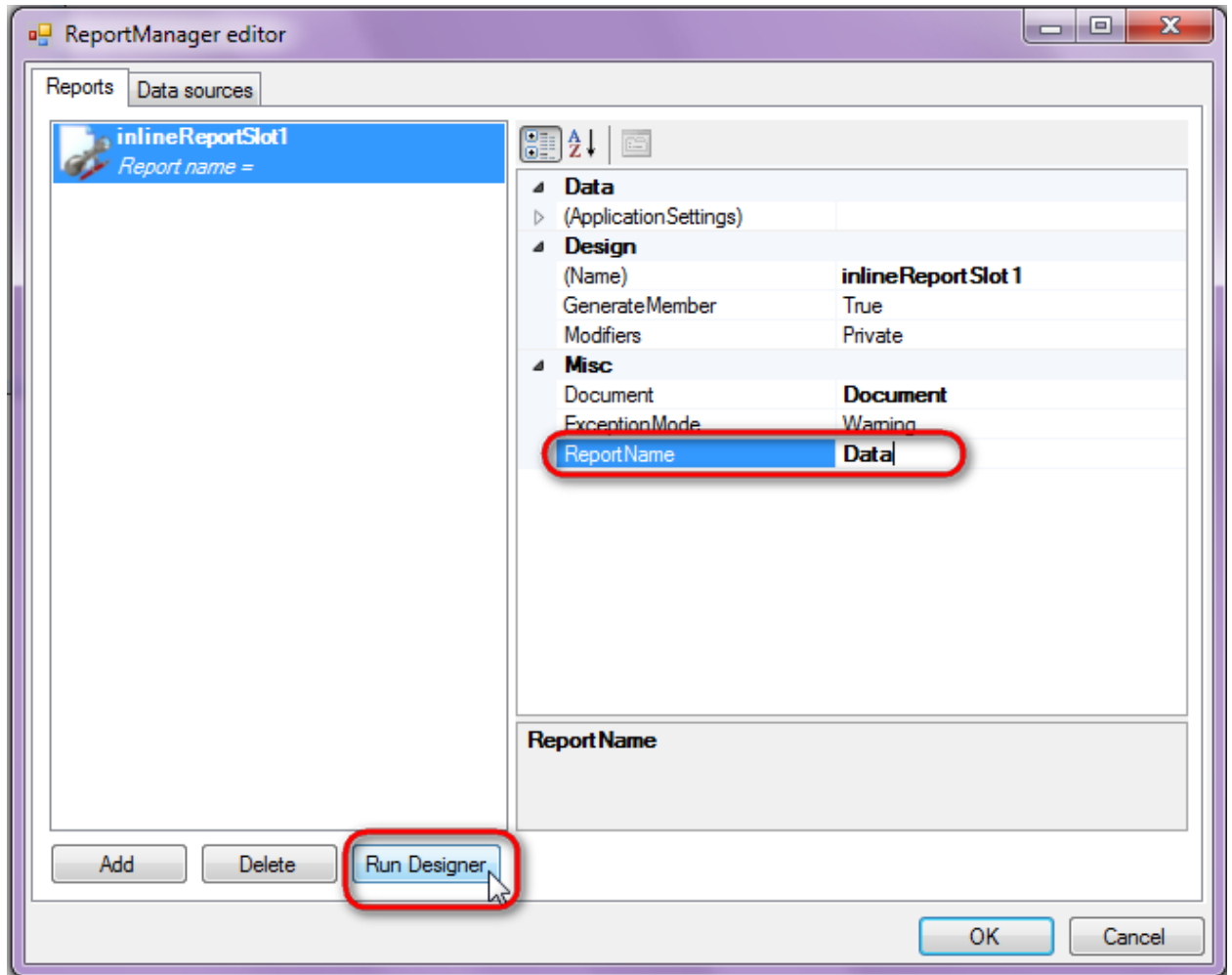
Go to "Reports" tab of the ReportManager editor, press "Add" and select "InlineReportSlot".



Step 11

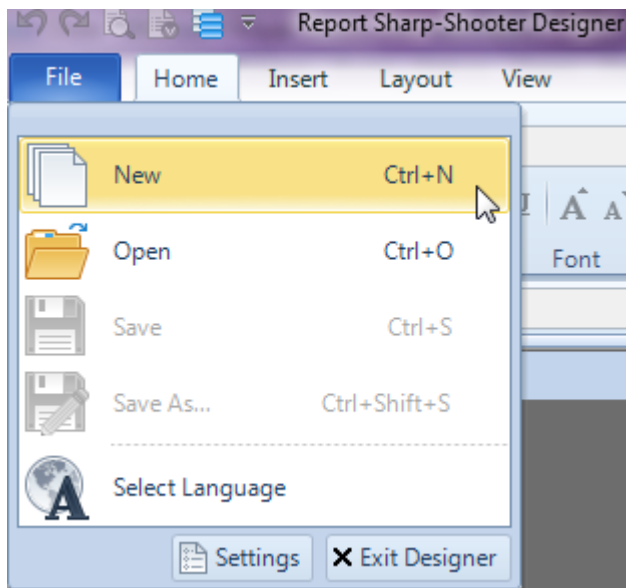
Set name of the report to "Data" in the ReportName property.

Click "Run Designer" in order to open template editor – Report Designer.

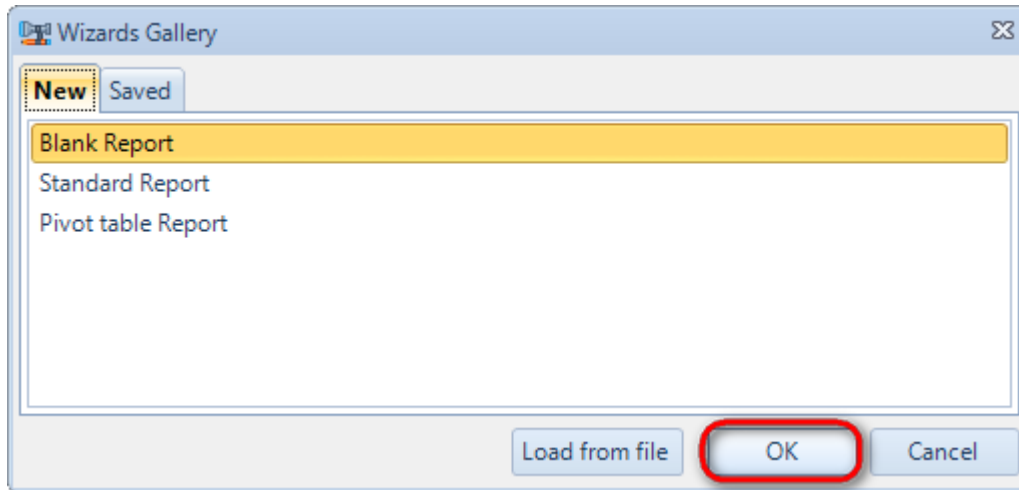


Step 12

Create new empty template – select File\New in the main menu.

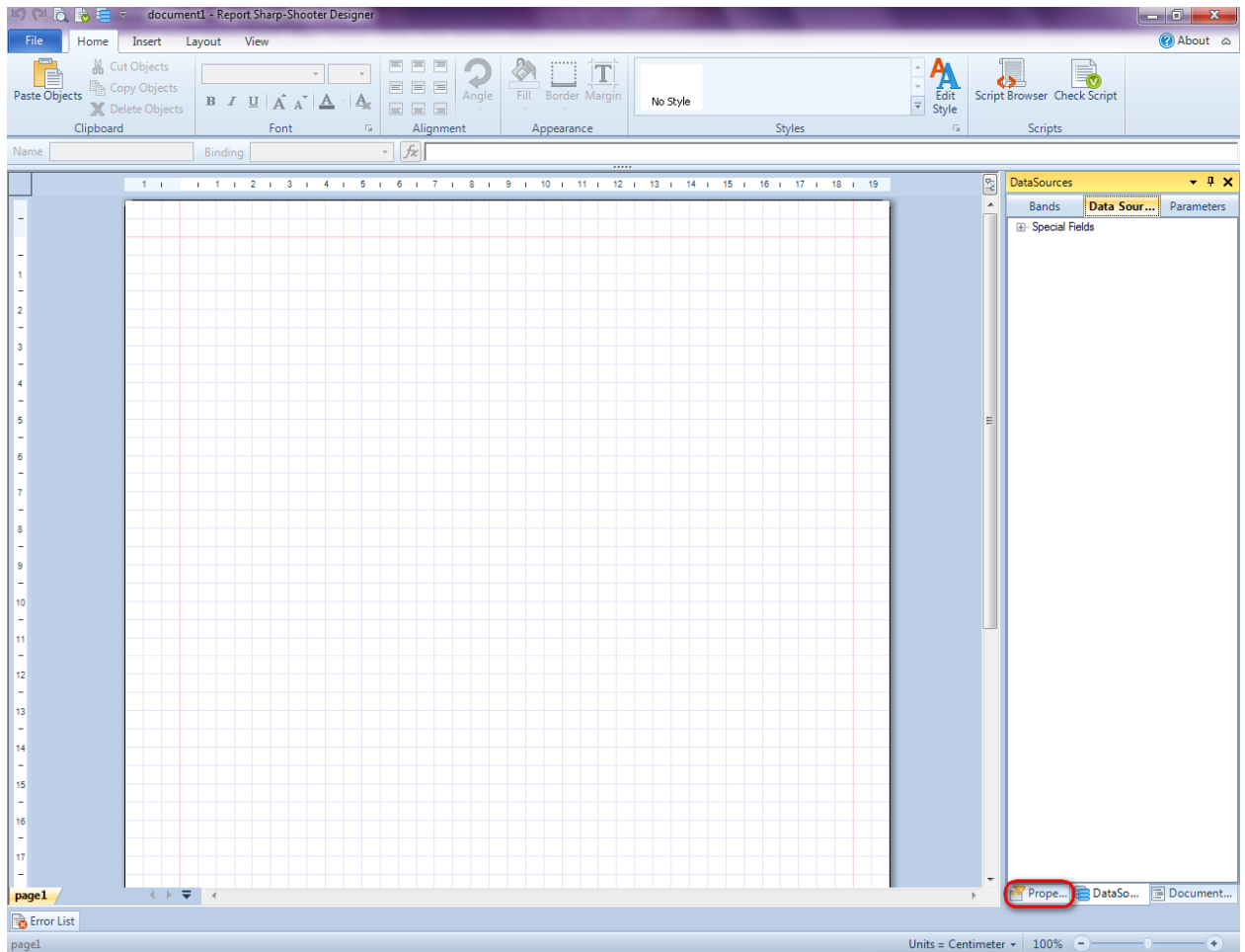


Select "Blank Report" in the Wizards Gallery and click "OK".



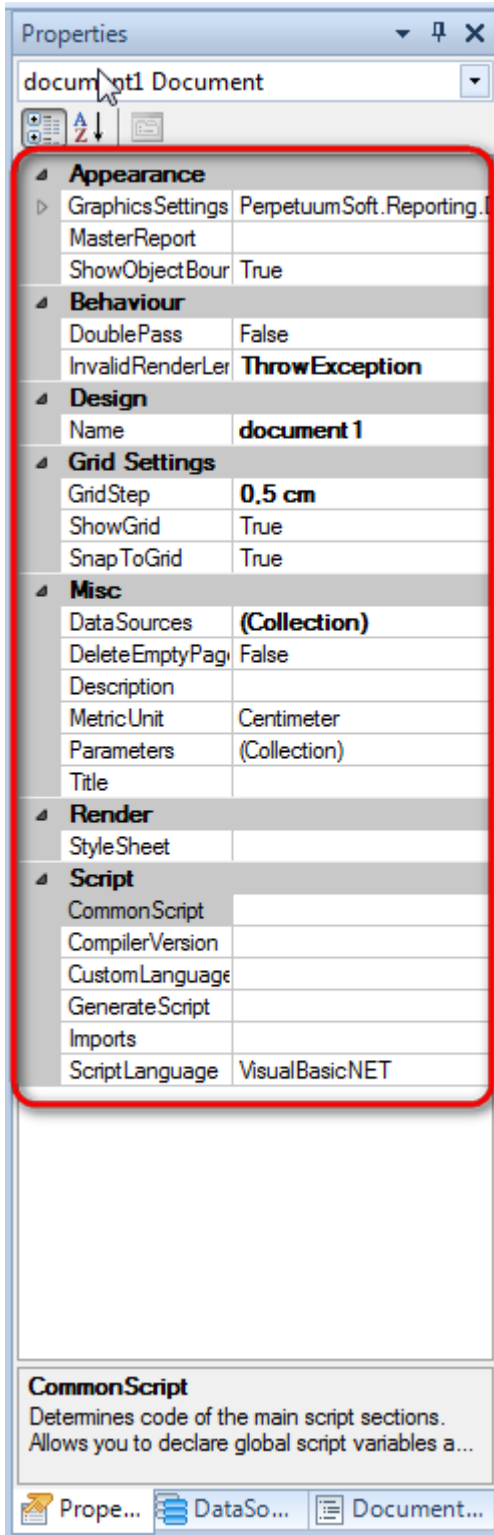
Step 13

Click the "Properties" tab of the tool window in the right part of the designer.

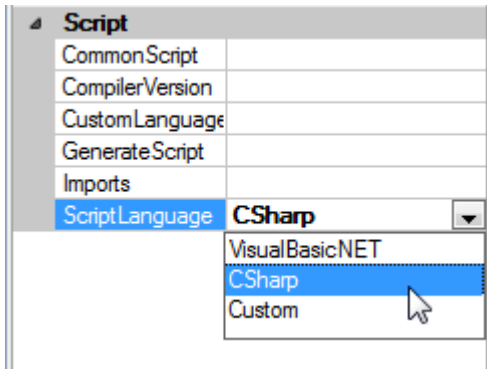




You will see properties of the edited template on the "Properties" tab

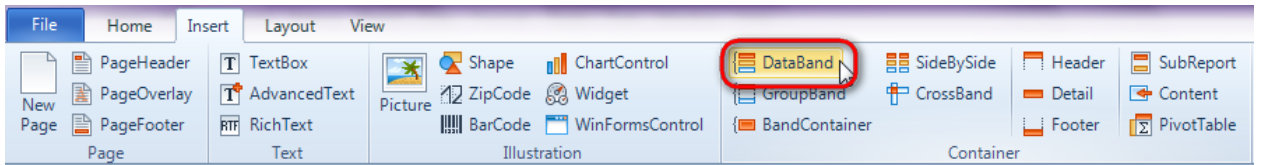


Set property ScriptLanguage = CSharp.



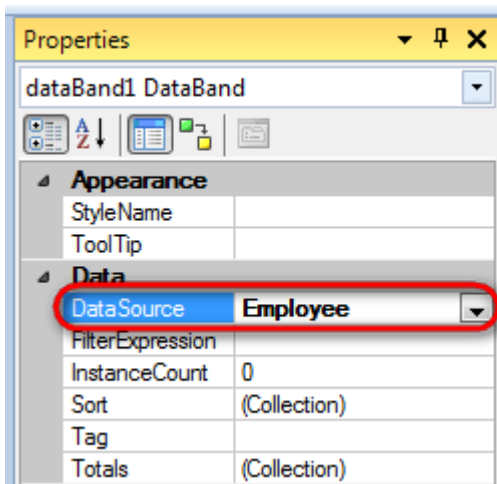
Step 14

Press "DataBand" button on the Insert tab in the group Container.



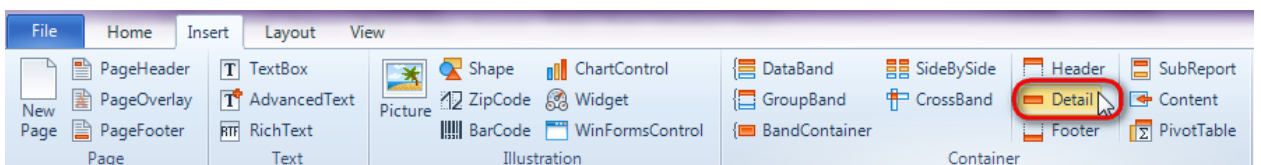
Click on the template area to add DataBand band.

Set data source in the property DataSource = Employee.

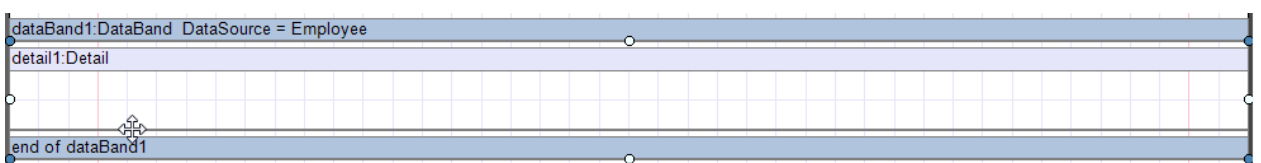


Step 15

Press "Detail" button on the Insert tab in the group Container.



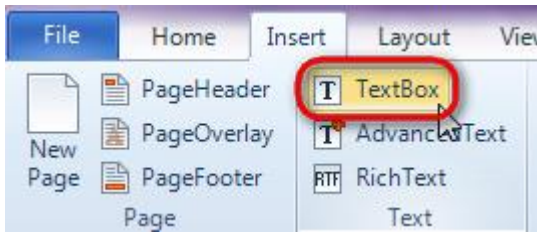
Click on the DataBand area to add the Detail band inside DataBand.





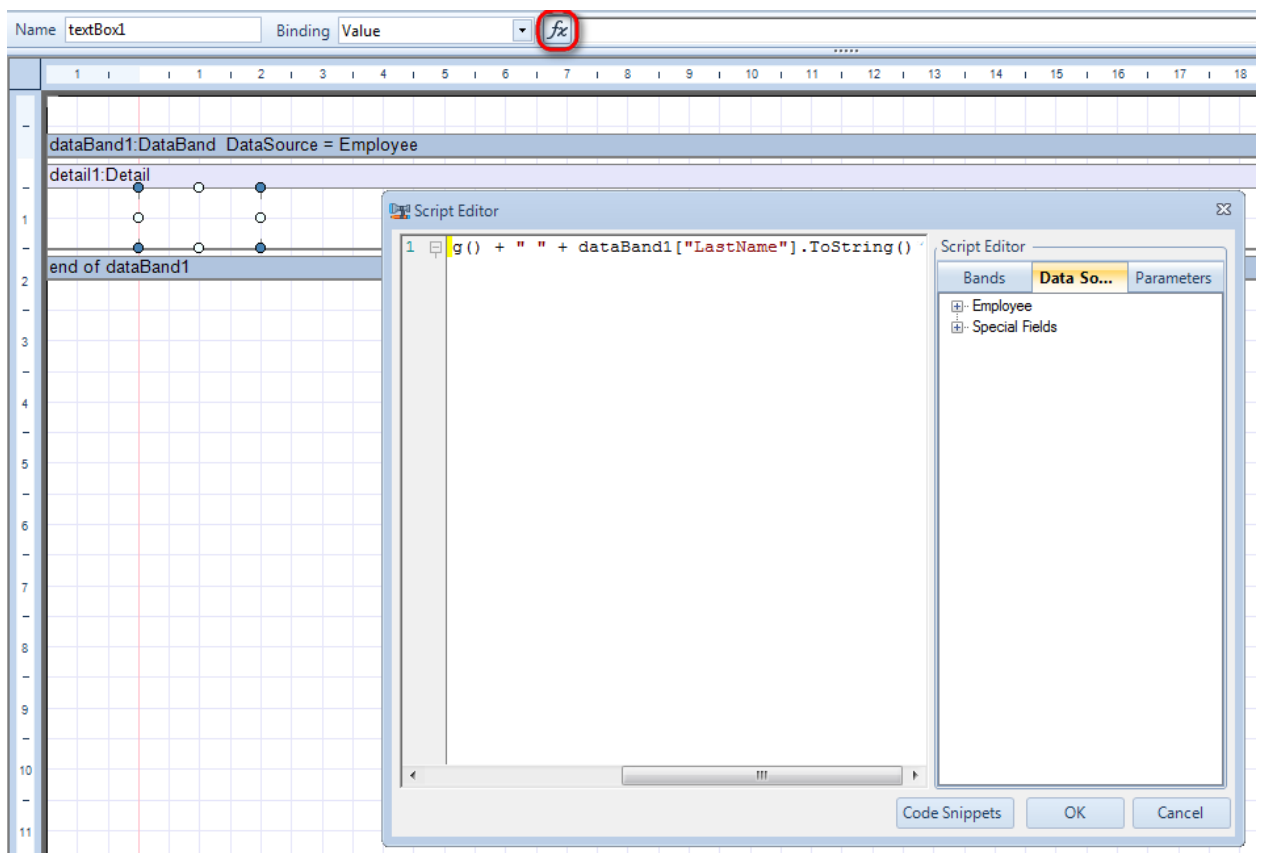
Step 16

Press "TextBox" button on the Insert tab in the group Text.



Click on the Detail area to add TextBox inside Detail band.

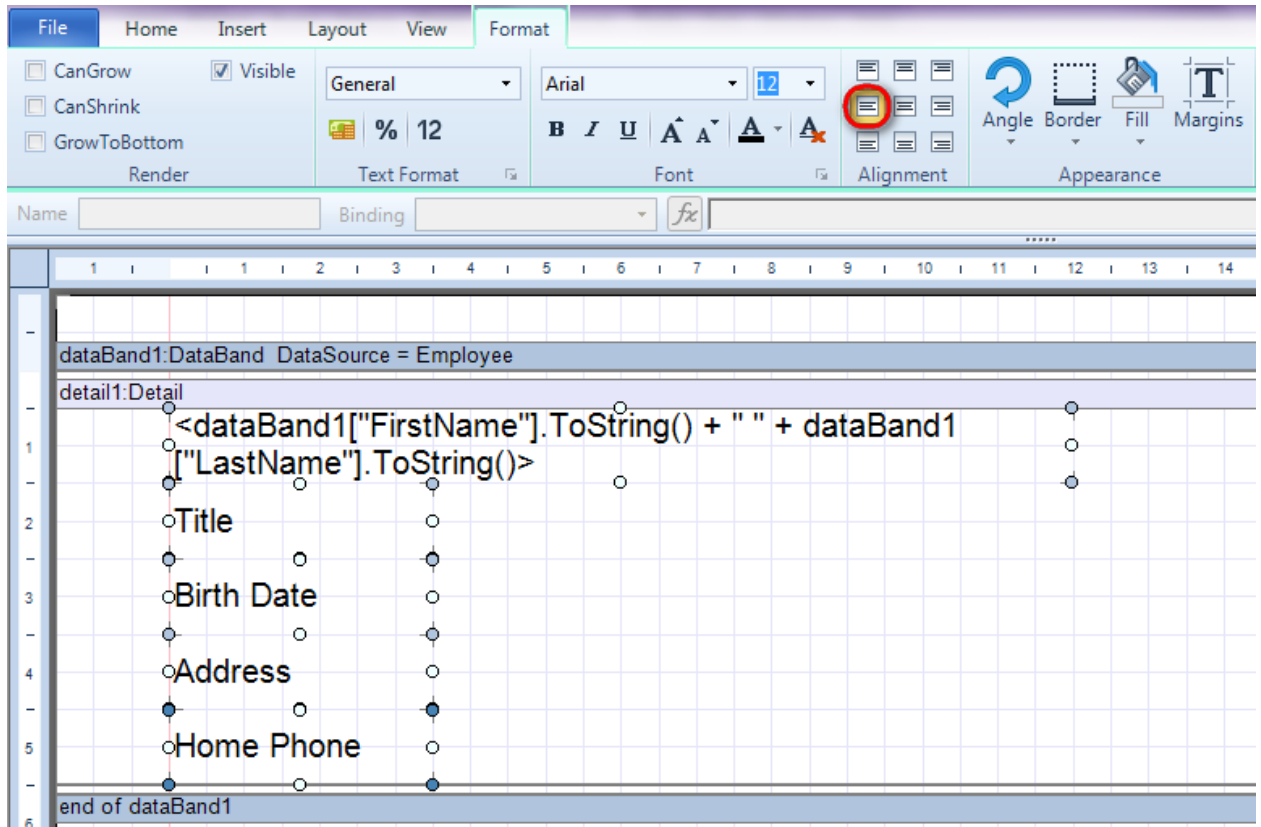
Click button "Script editor" on the toolbar to open Script editor and set Value property. Add the following code: `dataBand1["FirstName"].ToString() + " " + dataBand1["LastName"].ToString()`.



Step 17

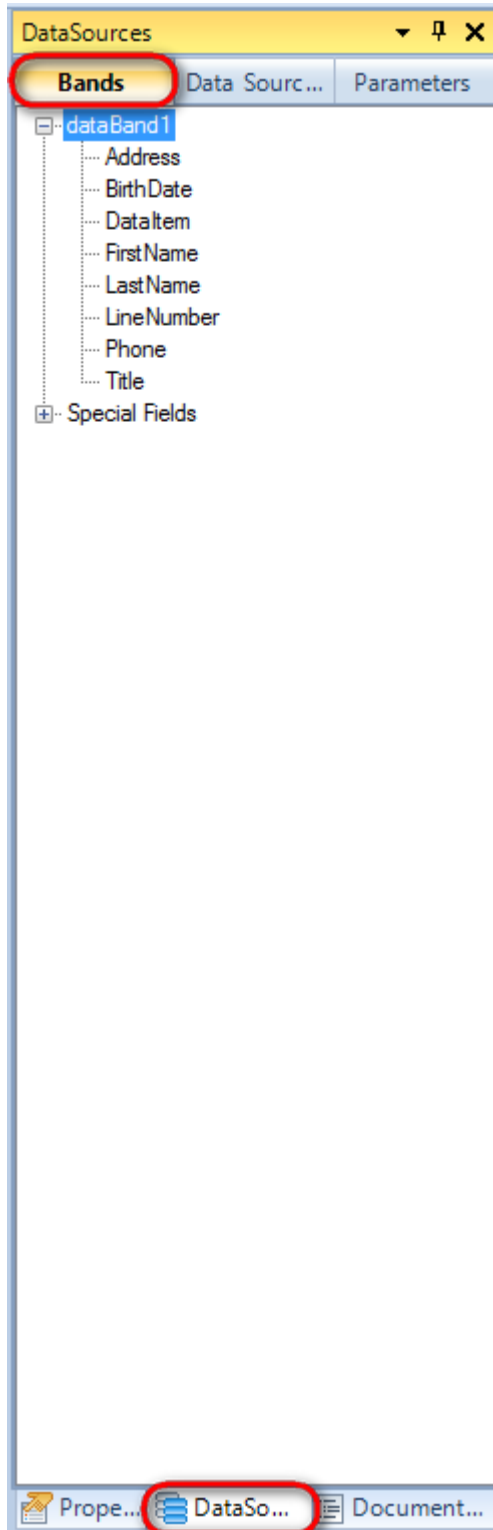
Add four TextBox elements. Set Text property value to "Title", "Birth Date", "Address", "Home Phone".

Change Detail size, size and orientation of the TextBox elements. Press Ctrl select all TextBox elements. In order to left-align text click the "Align Middle Left" button on the Format tab in the group Alignment.



Step 18

Go to "DataSources" tab.



Drag and drop fields "Title", "BirthDate", "Address", "HomePhone" from the "dataBand1" tree to the detail1 band. As a result TextBox elements are created. Value property will feature script loading data from the data source. Change size of the element and right-align text.



dataBand1:DataBand DataSource = Employee	
detail1:Detail	
<dataBand1["FirstName"].ToString() + " " + dataBand1["LastName"].ToString(>	
Title	<dataBand1["Title"]>
Birth Date	<dataBand1["BirthDate"]>
Address	<dataBand1["Address"]>
Home Phone	<dataBand1["Phone"]>
end of dataBand1	

Step 19

Save template and close Report Designer.

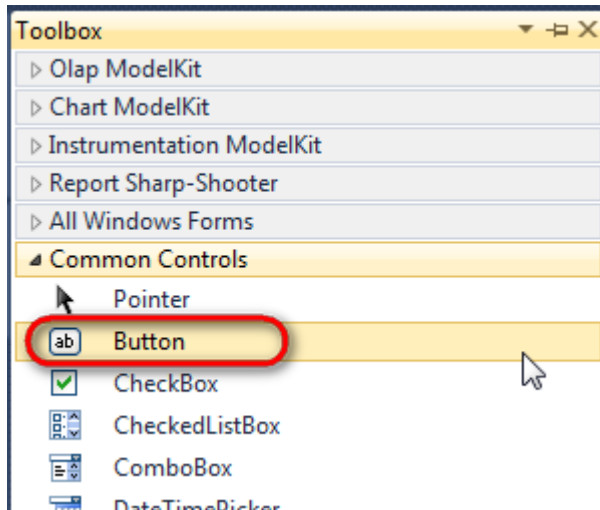
Step 20

Add code responsible for displaying report to the class constructor. Write RenderCompleted event handler of InlineReportSlot object.

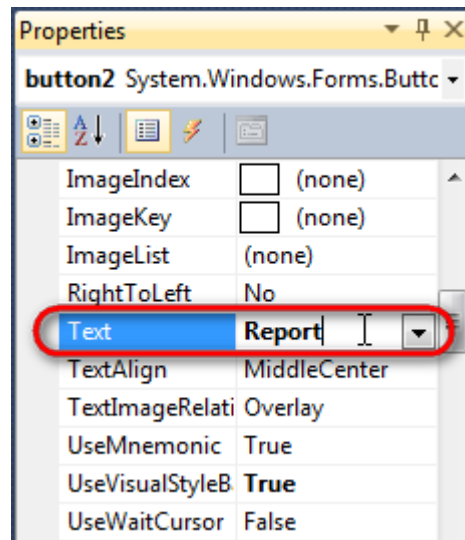
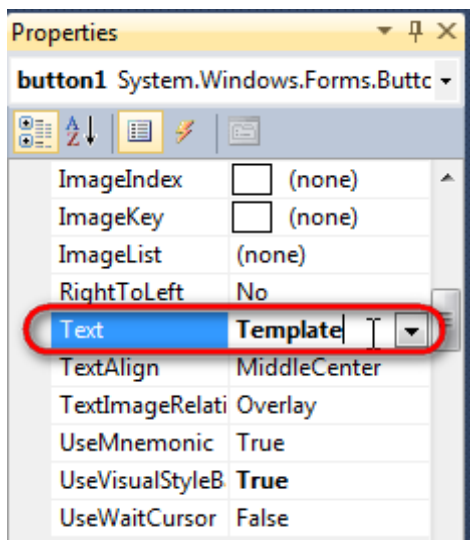
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["FirstName"] = "Robert";
    row["LastName"] = "King";
    row["Title"] = "Sales Representative";
    row["Phone"] = "(71) 555-5598";
    row["BirthDate"] = "29.05.1960";
    row["Address"] = "UK, London, Edgeham Hollow Winchester Way";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 21

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



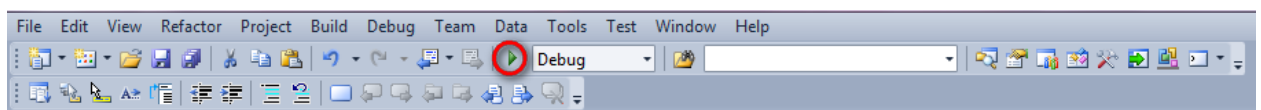
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

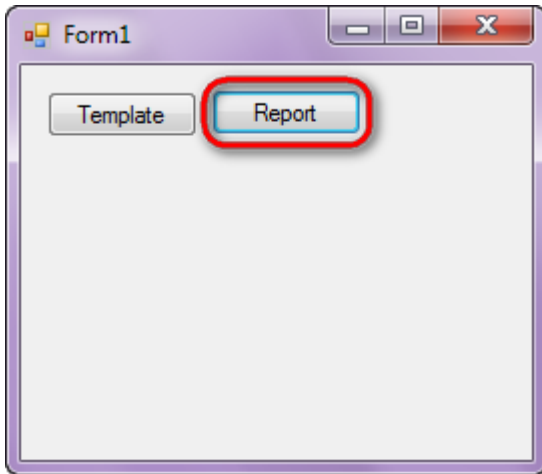
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

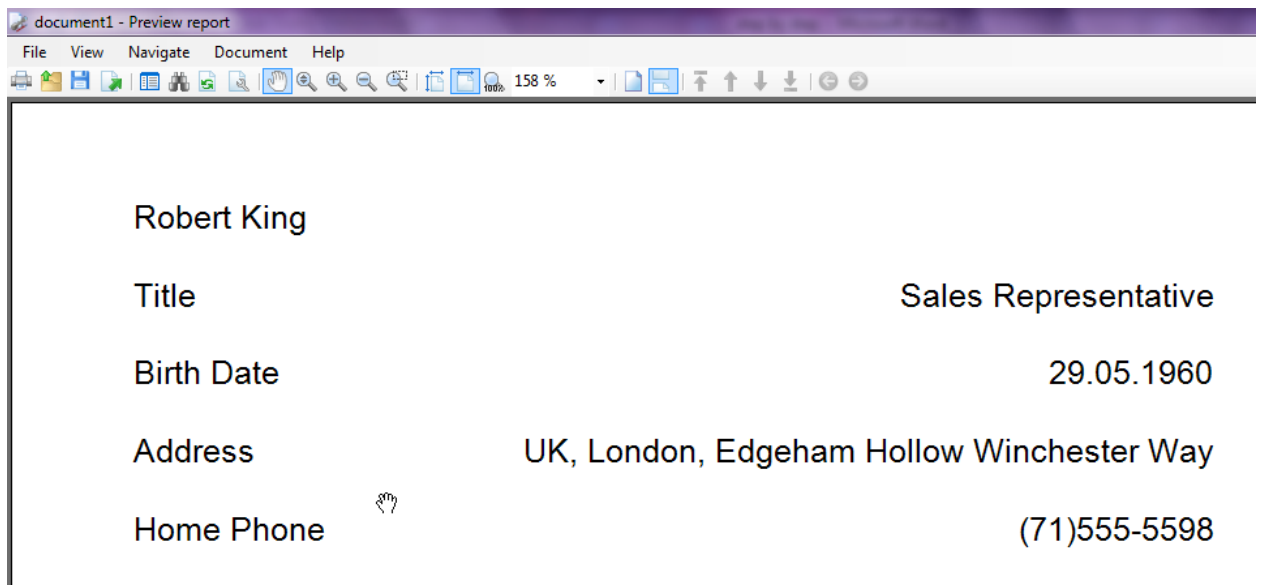
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.

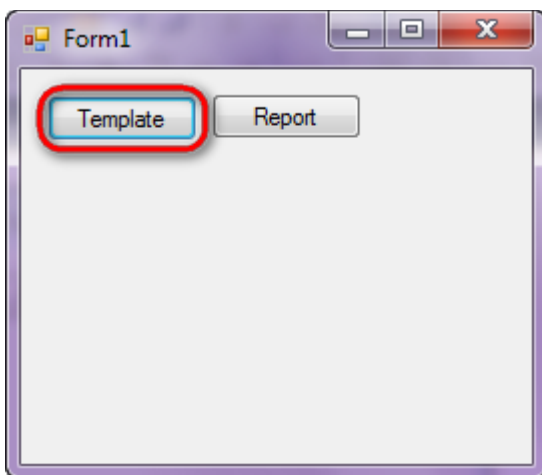




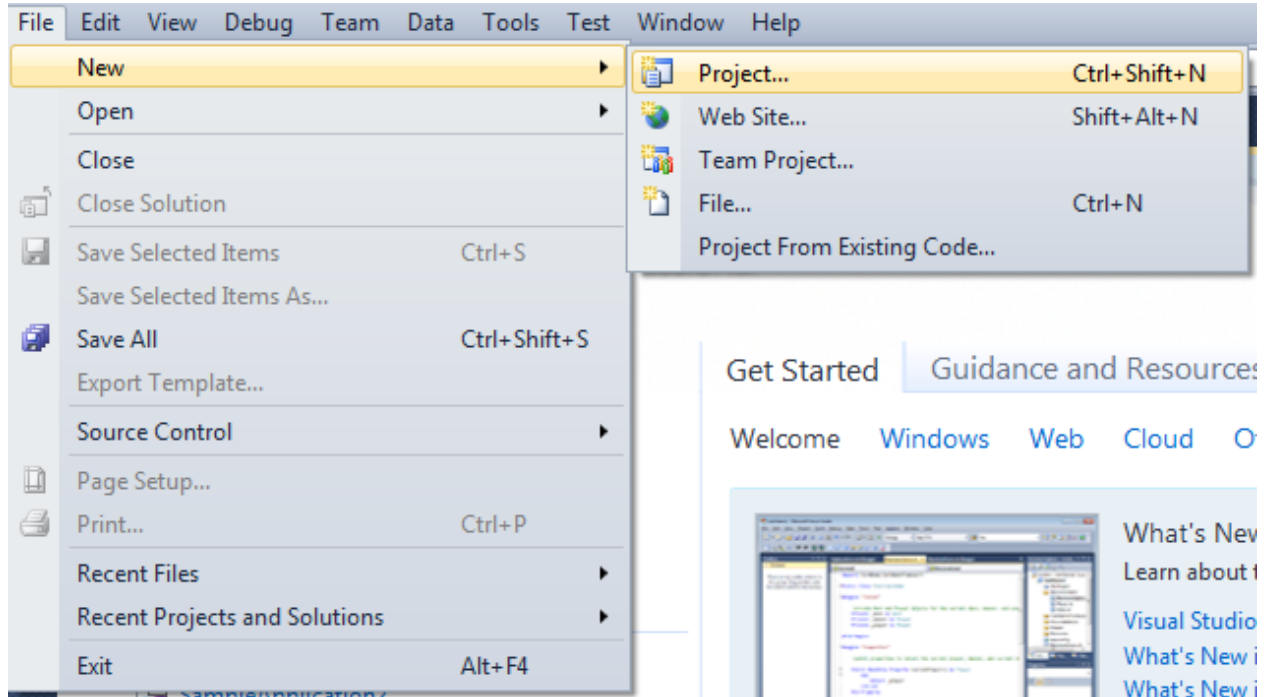
Table Wizard

Table Wizard allows a user to create different tables in a few clicks.

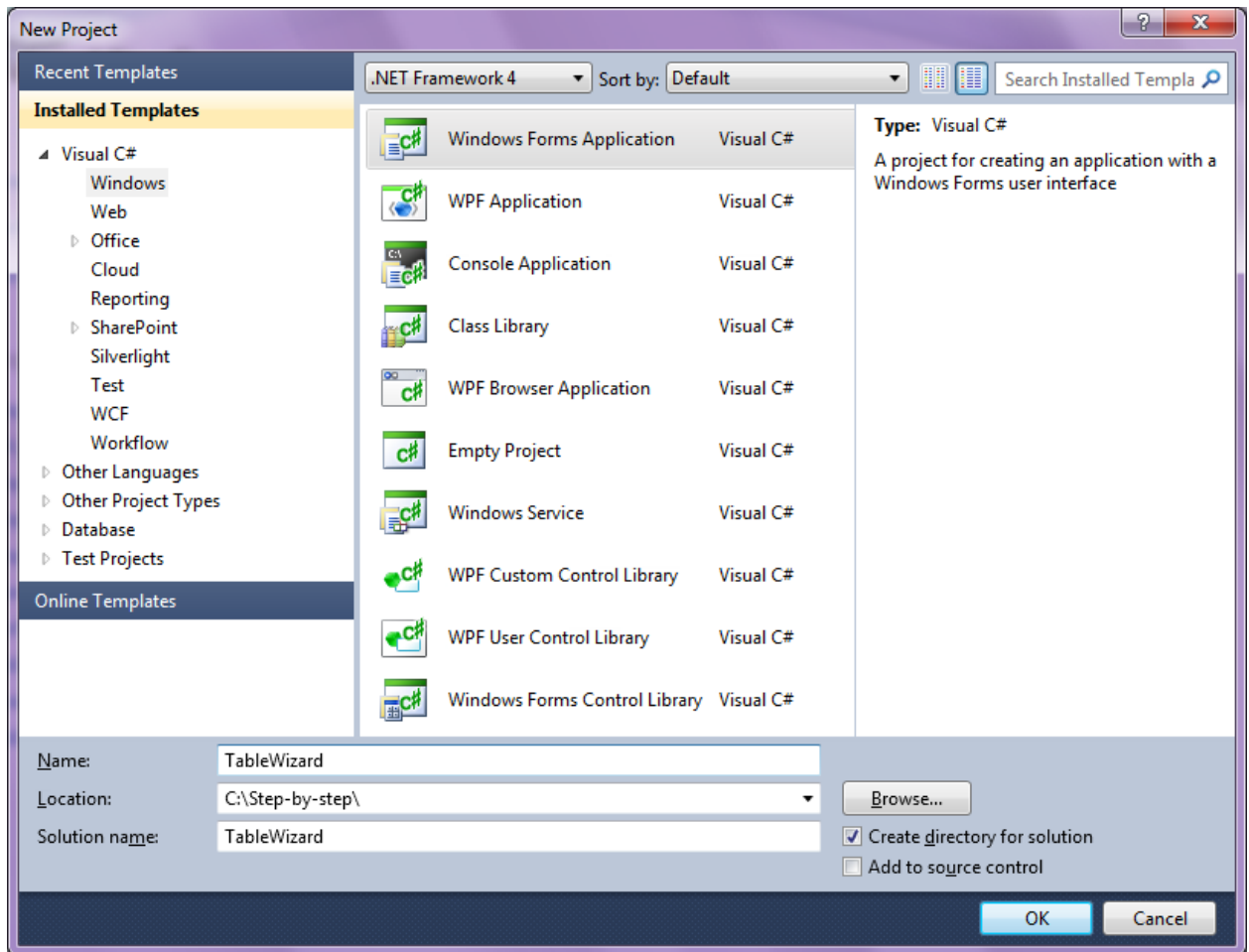
Template of the report displaying information on the employee; data is loaded from the data source.

Step 1

Create a new project in Microsoft Visual Studio. Select New\Project from the main menu.

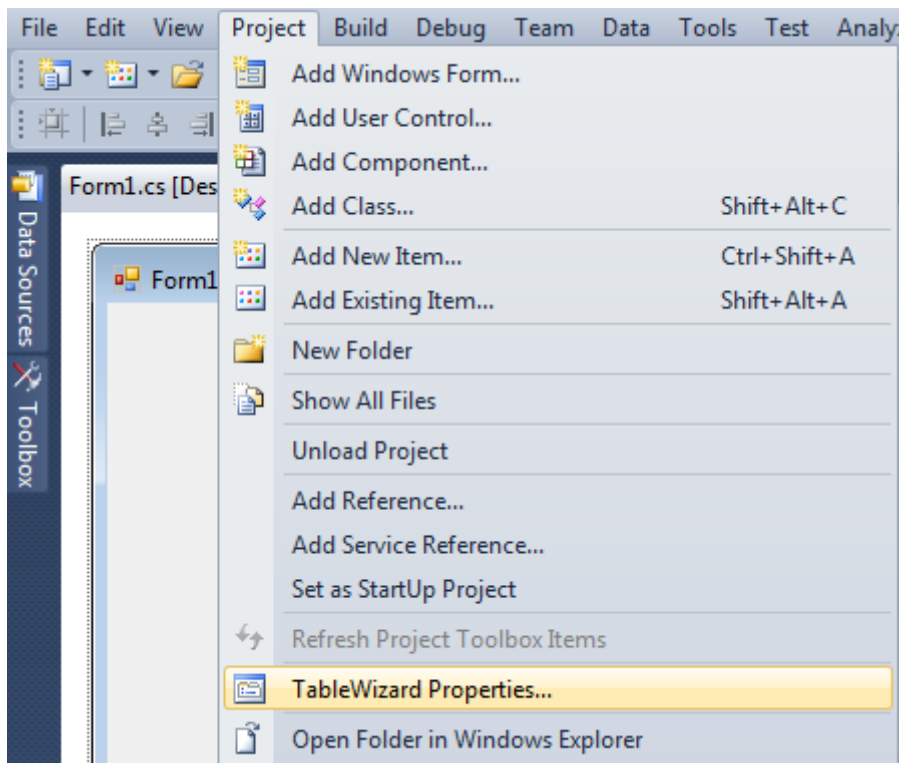


Select Windows Forms Application, set name of the project – “TableWizard” and set directory to save the project to.



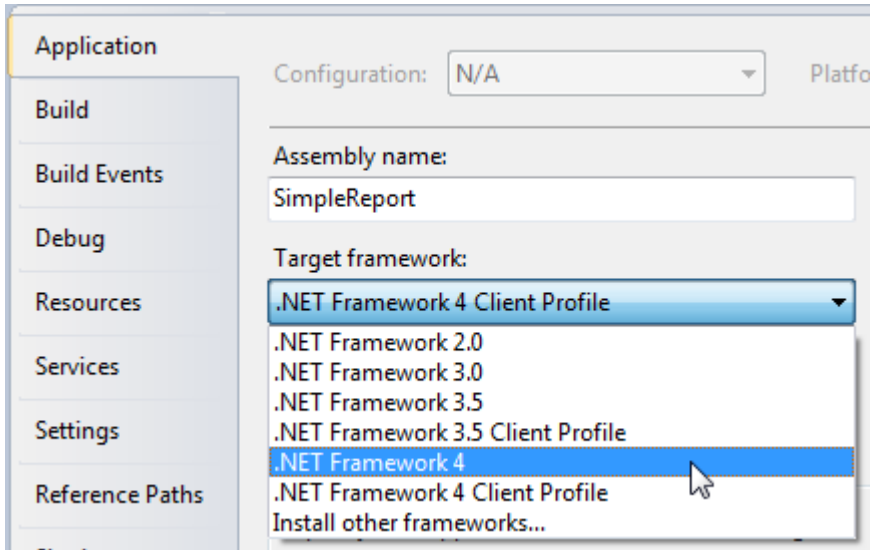
Step 2

Change the project properties. Select the Project\TableWizard Properties... item in the main menu.

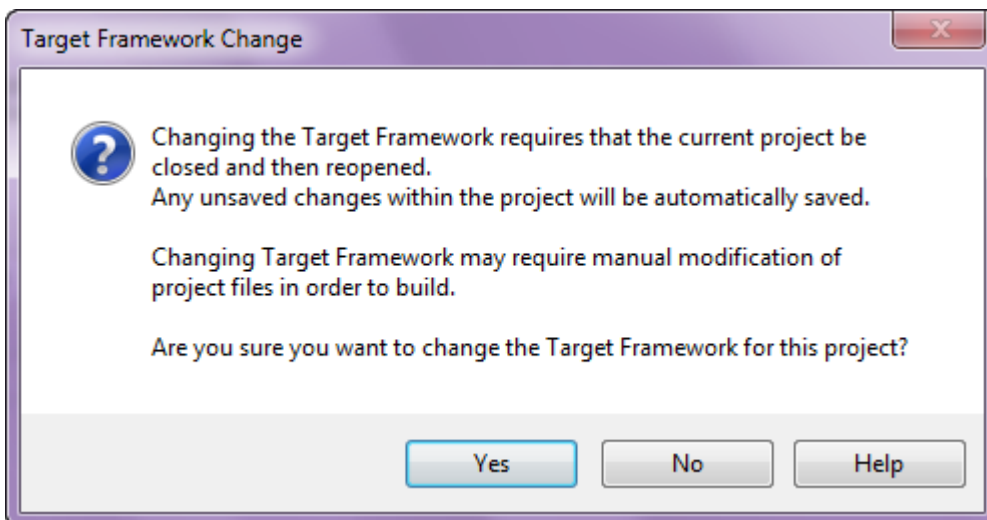




Select the Target framework\ .NET Framework4 item in the Application tab.

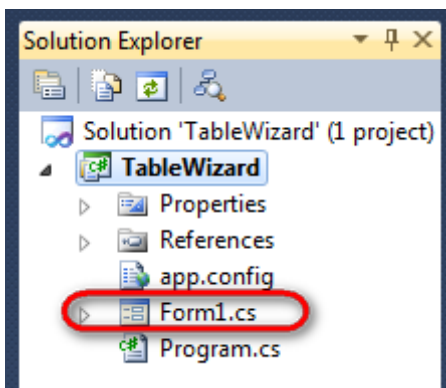


Press the "Yes" button in the opened window.

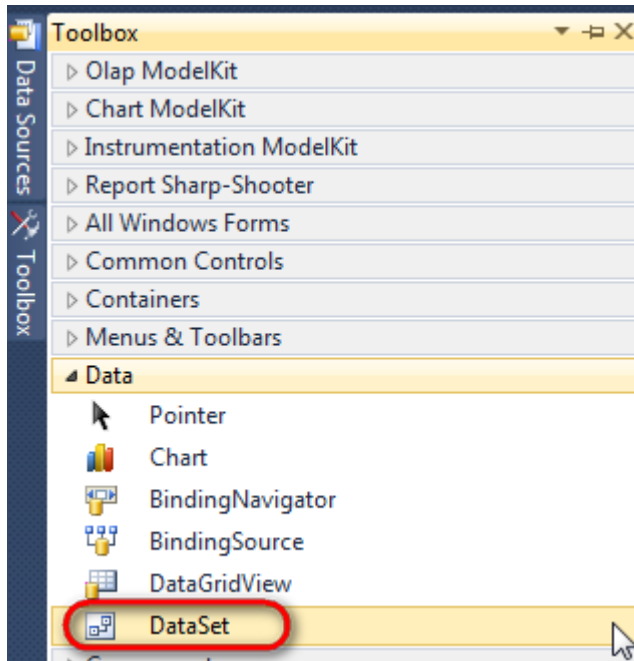


Step 3

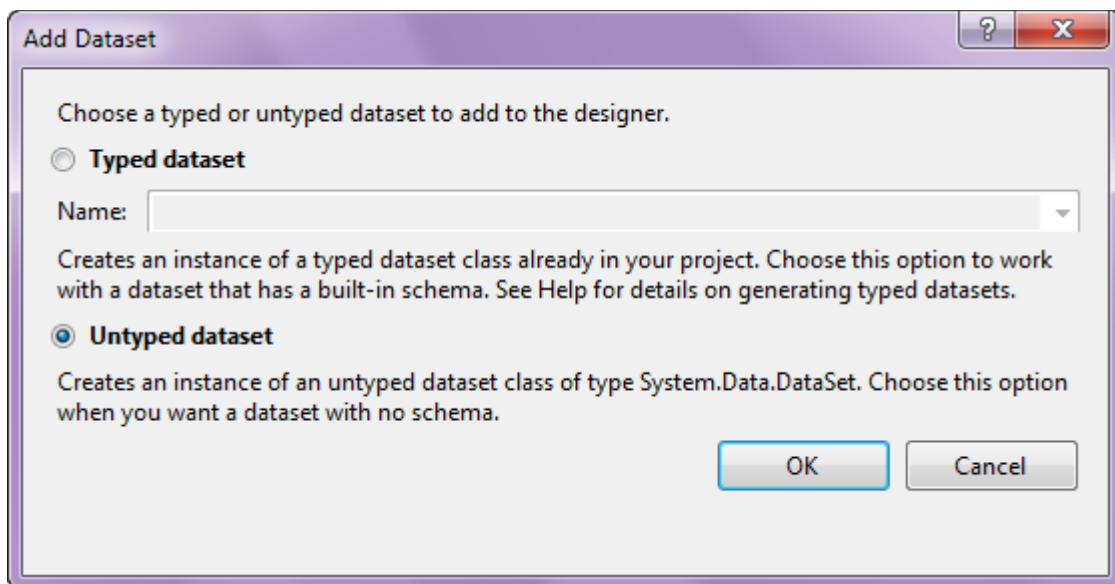
Open main form of the application in the editor by double click on "Form1.cs" in the Solution Explorer.



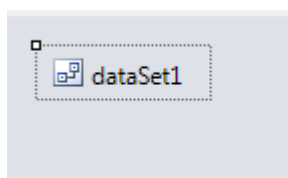
Click "DataSet" element in the Toolbox and place it onto the form.




Select "Untyped dataset", click "OK".

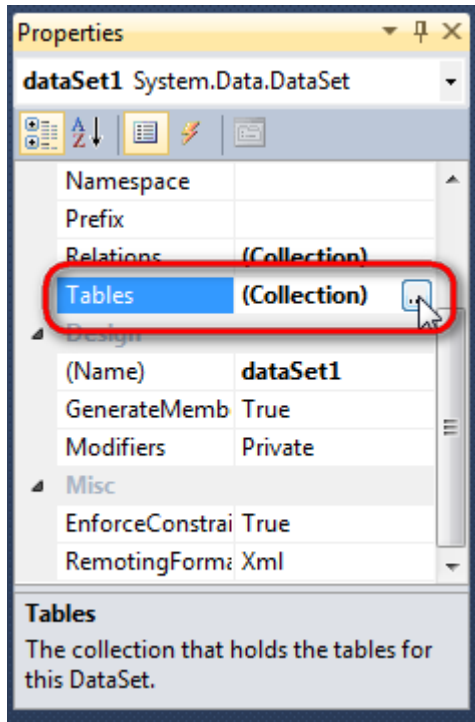


The component is displayed in the lower part of the window.

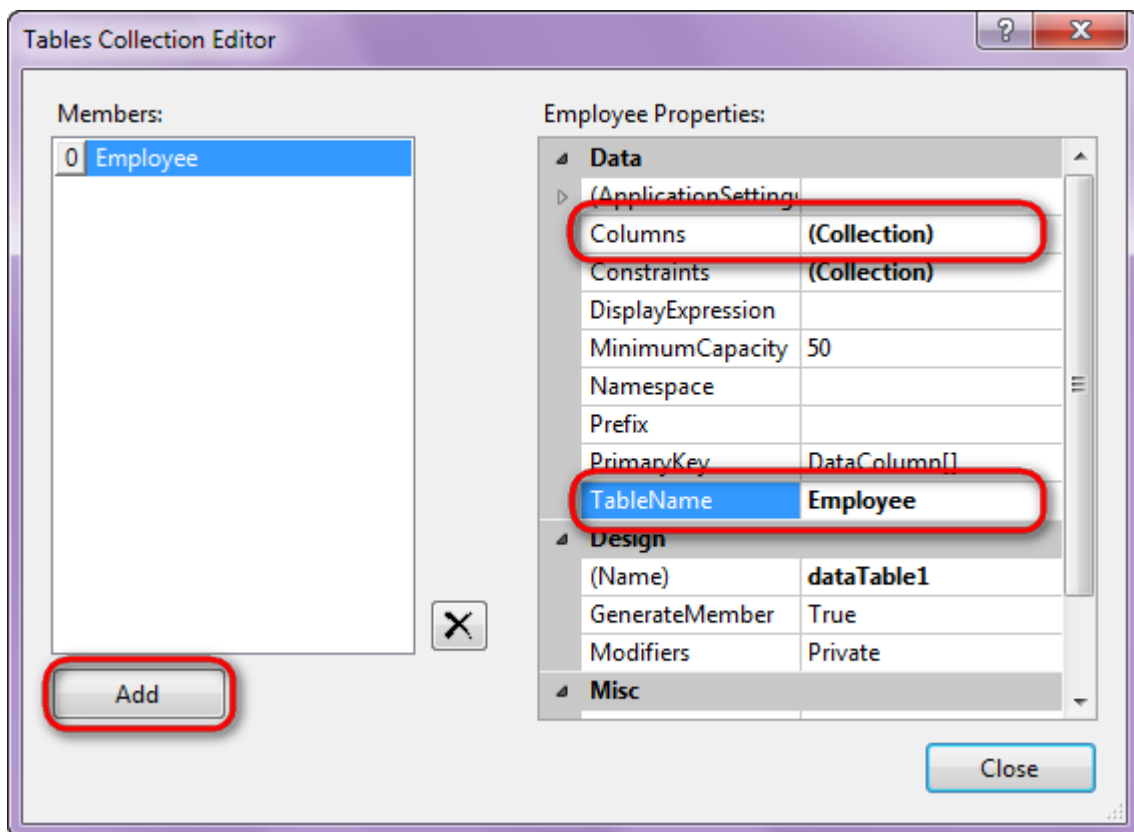


Step 4


Select dataSet1 component in the form editor. On the property grid, select "Tables" property, press button  in order to open Tables Collection Editor.

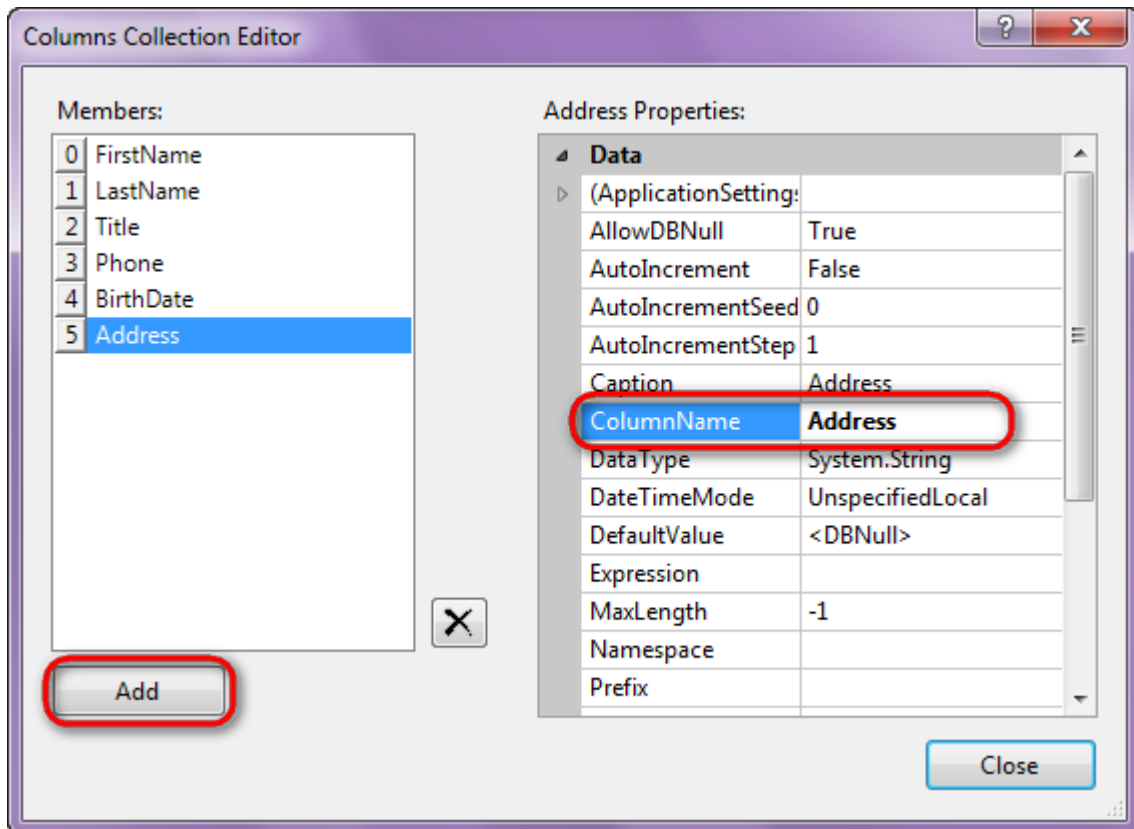


Press button "Add" in order to add a table. Set property TableName = Employee.



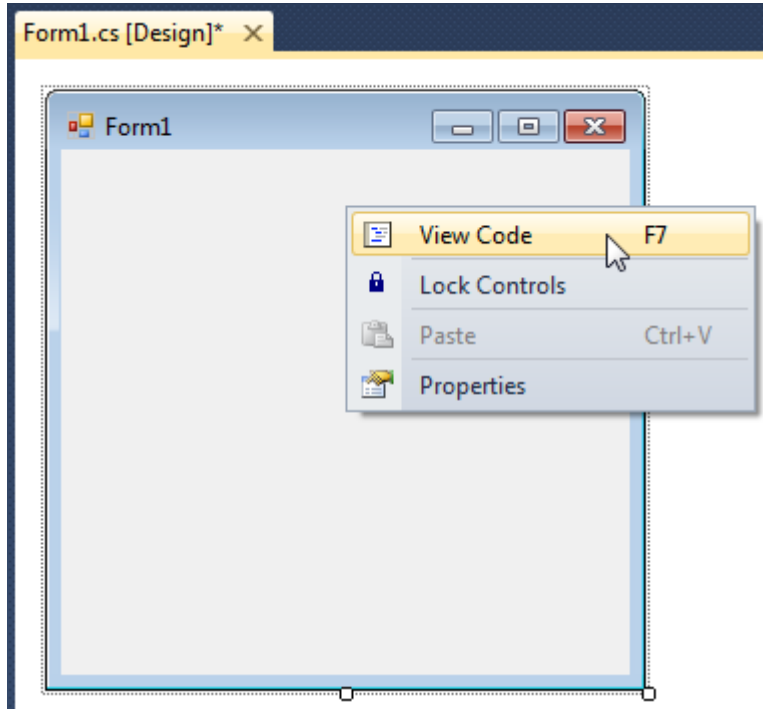
Step 5

Select "Columns" property, press button  in order to open property editor. Press "Add" to add a new column. Add four columns. Set value of the property ColumnName to "FirstName", "LastName", "Title", "Phone", "BirthDate", "Address" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu in order to view code.



To fill a data source, add the following code to the class constructor:

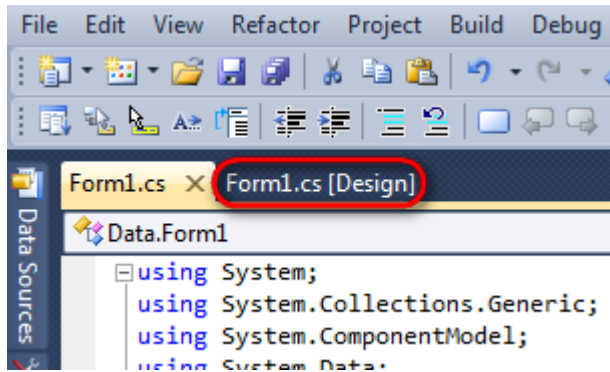
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow();
    row["FirstName"] = "Maria";
}
```



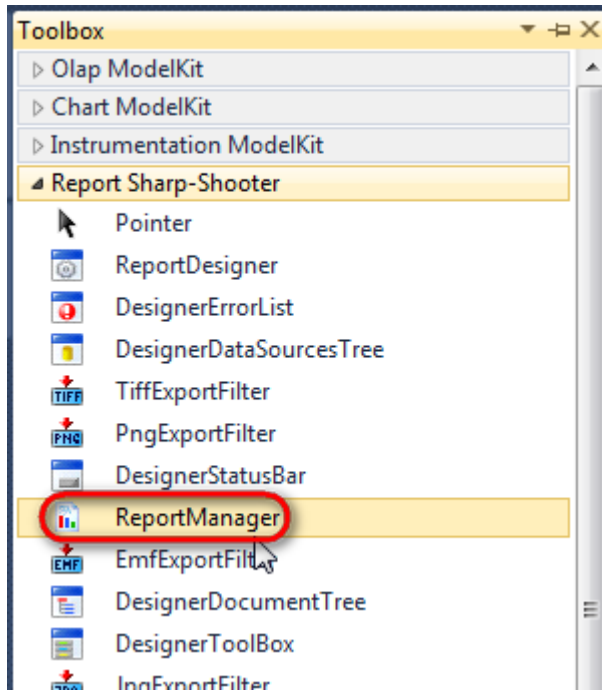
```
row["LastName"] = "Anders";  
row["Title"] = "Sales Representative";  
row["Phone"] = "(71)555-5598";  
row["BirthDate"] = "29.05.1960";  
row["Address"] = "Obere str. 57";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["FirstName"] = "Ana";  
row["LastName"] = "Trujillo";  
row["Title"] = "Owner";  
row["Phone"] = "(5)555-4729";  
row["BirthDate"] = "01.07.1971";  
row["Address"] = "Avda. de la Constitution 222";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["FirstName"] = "Antonio";  
row["LastName"] = "Moreno";  
row["Title"] = "Owner";  
row["Phone"] = "(5)555-3932";  
row["BirthDate"] = "12.03.1969";  
row["Address"] = "Mataderos 2312";  
dataTable1.Rows.Add(row);  
}
```

Step 7

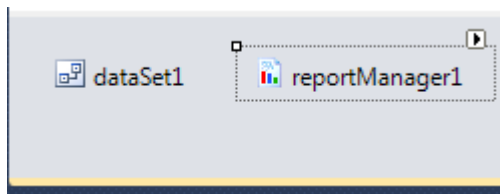
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click "ReportManager" element on the Toolbox and place it onto the form. This element is designed to store collections of report templates and data sources.

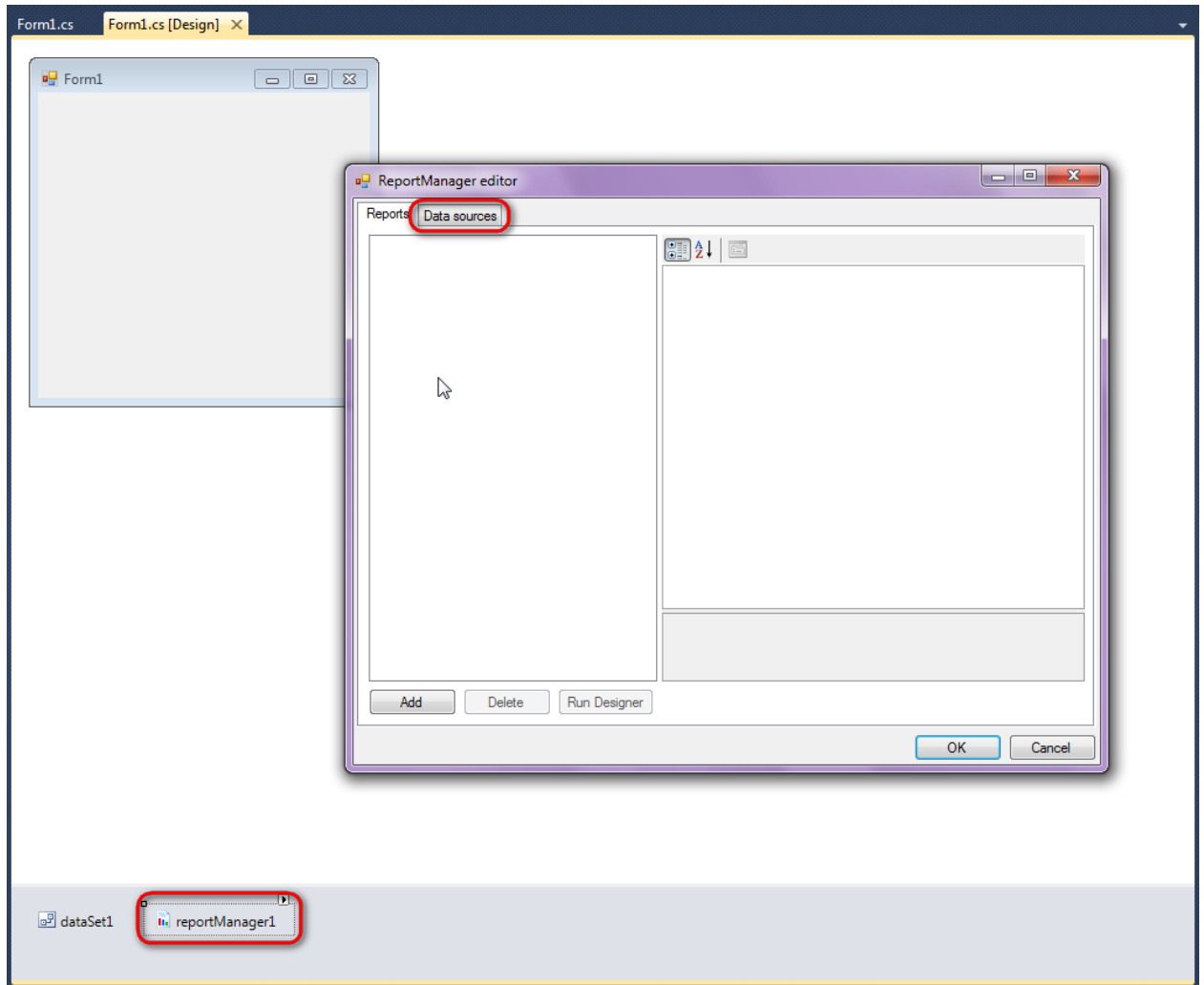


Component is displayed in the lower part of the window.

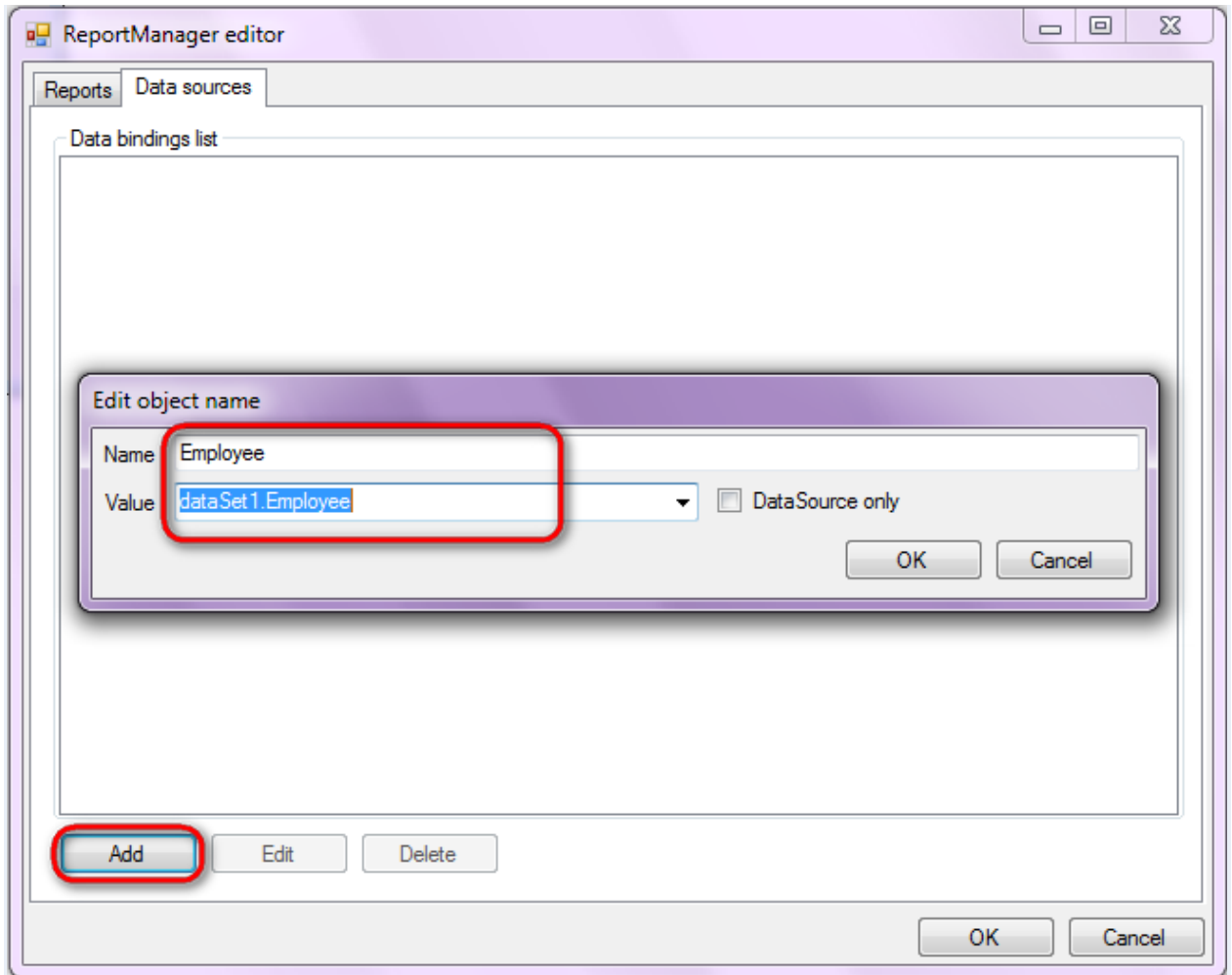


Step 8

Double click on the ReportManager component and open ReportManager editor.

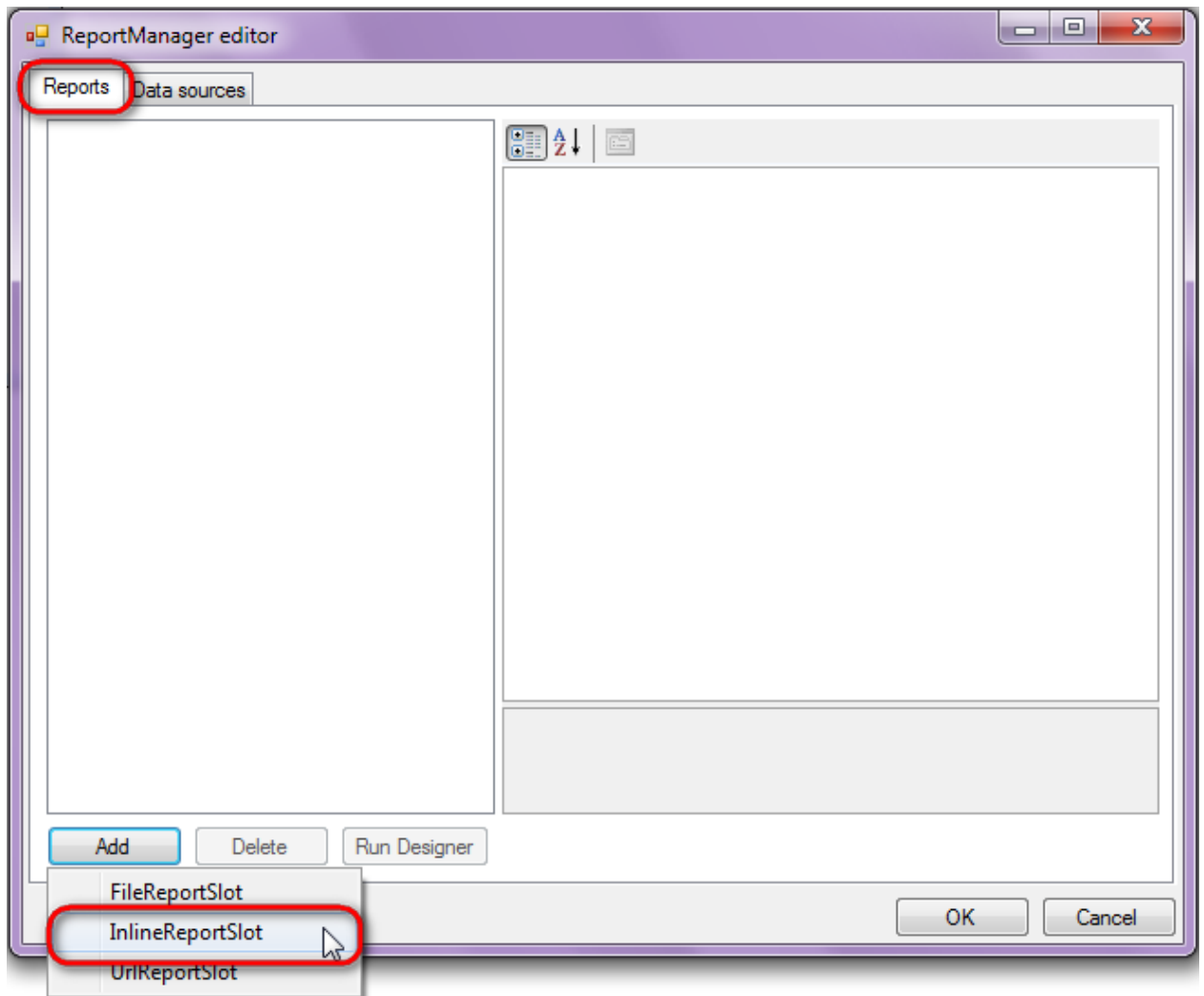


Go to "Data sources" tab, press "Add", set name of the data source - "Employee", select data source value - "dataSet1.Employee".



Step 9

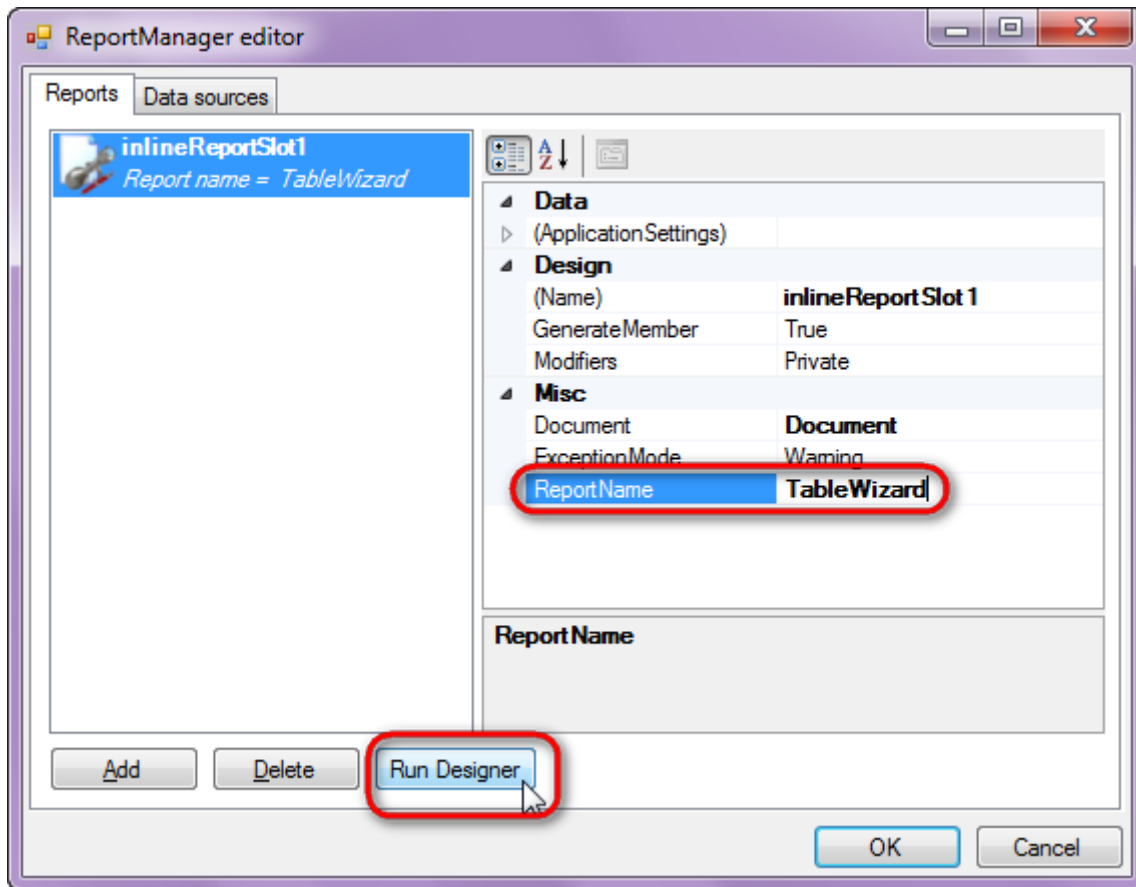
Go to "Reports" tab of the ReportManager editor, press "Add" and select "InlineReportSlot".



Step 10

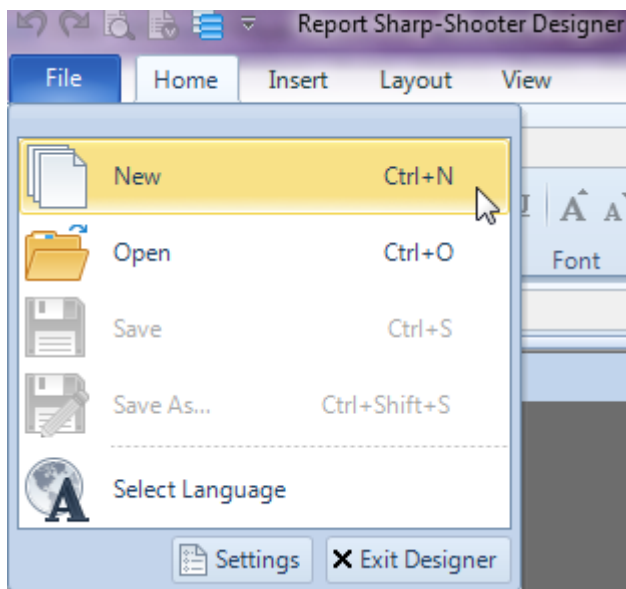
Set name of the report to "TableWizard" in the ReportName property.

Click "Run Designer" in order to open template editor – Report Designer.

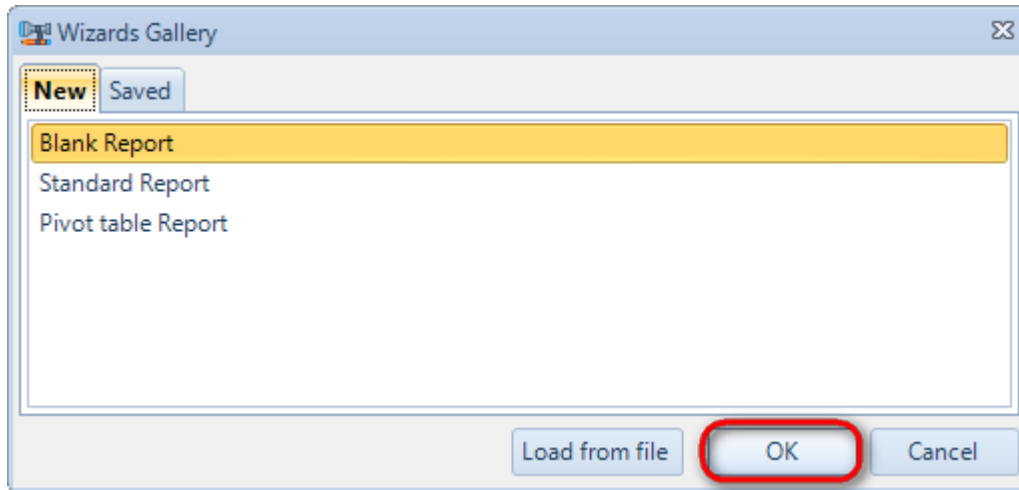


Step 11

Create new empty template – select File\New in the main menu.

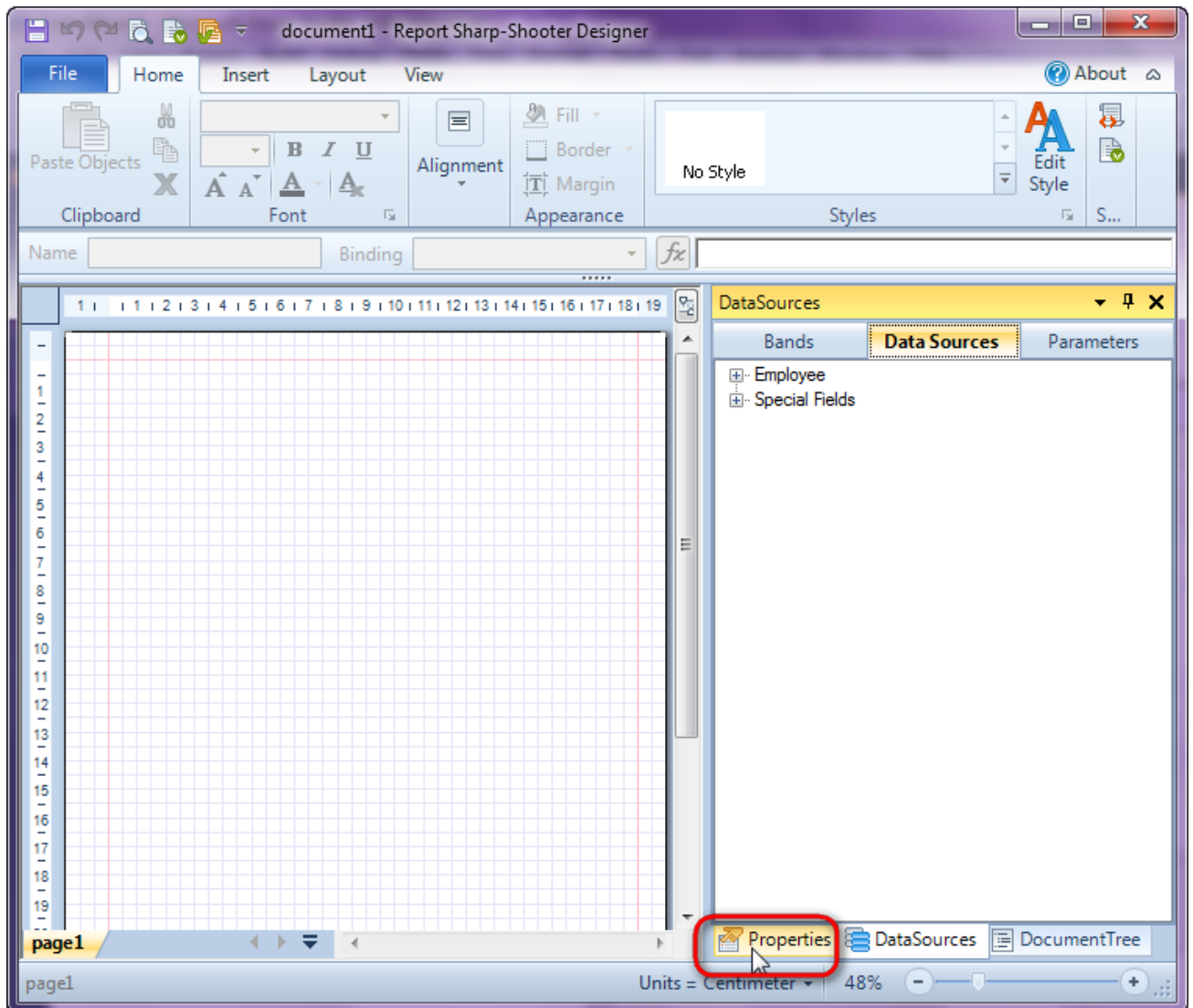


Select "Blank Report" in the Wizards Gallery and click "OK".



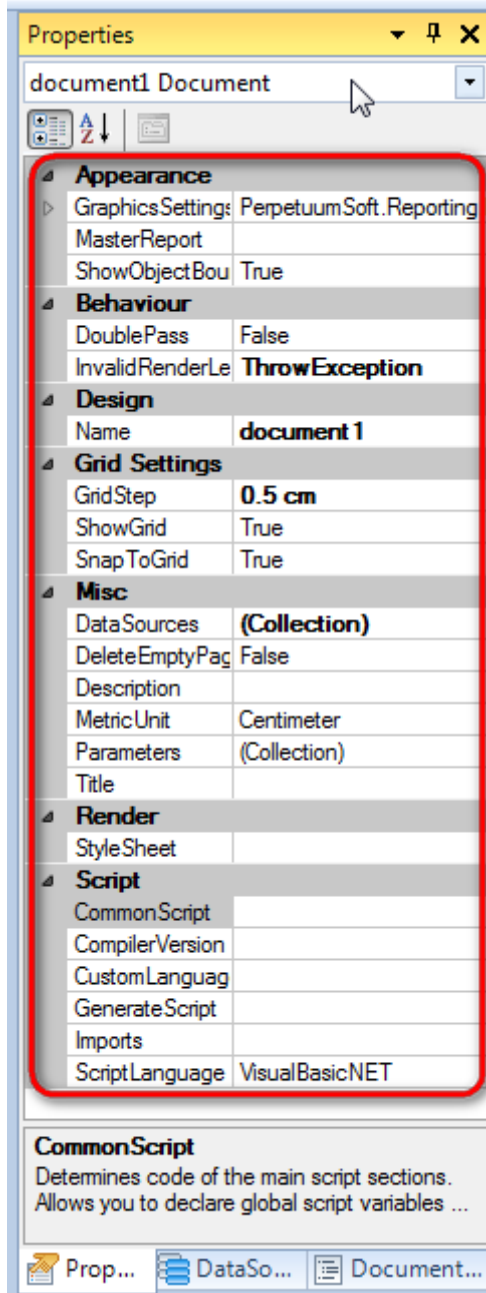
Step 12

Click the "Properties" tab of the tool window in the right part of the designer.

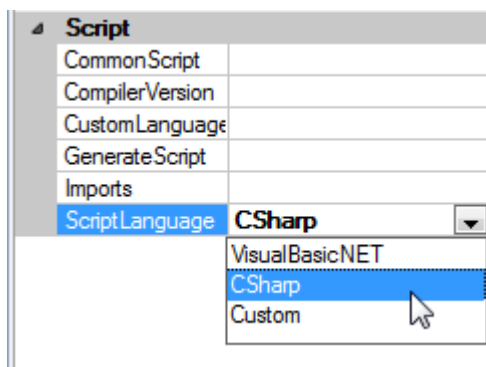




You will see properties of the edited template on the "Properties" tab

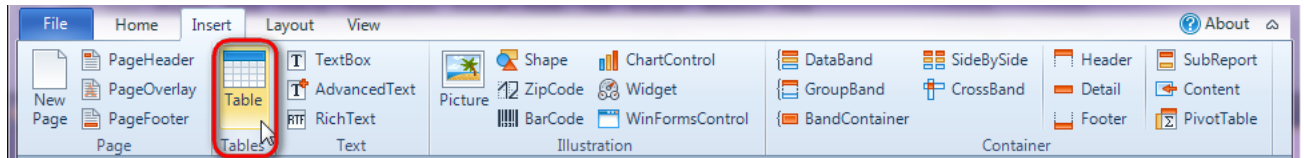


Set property ScriptLanguage = CSharp.



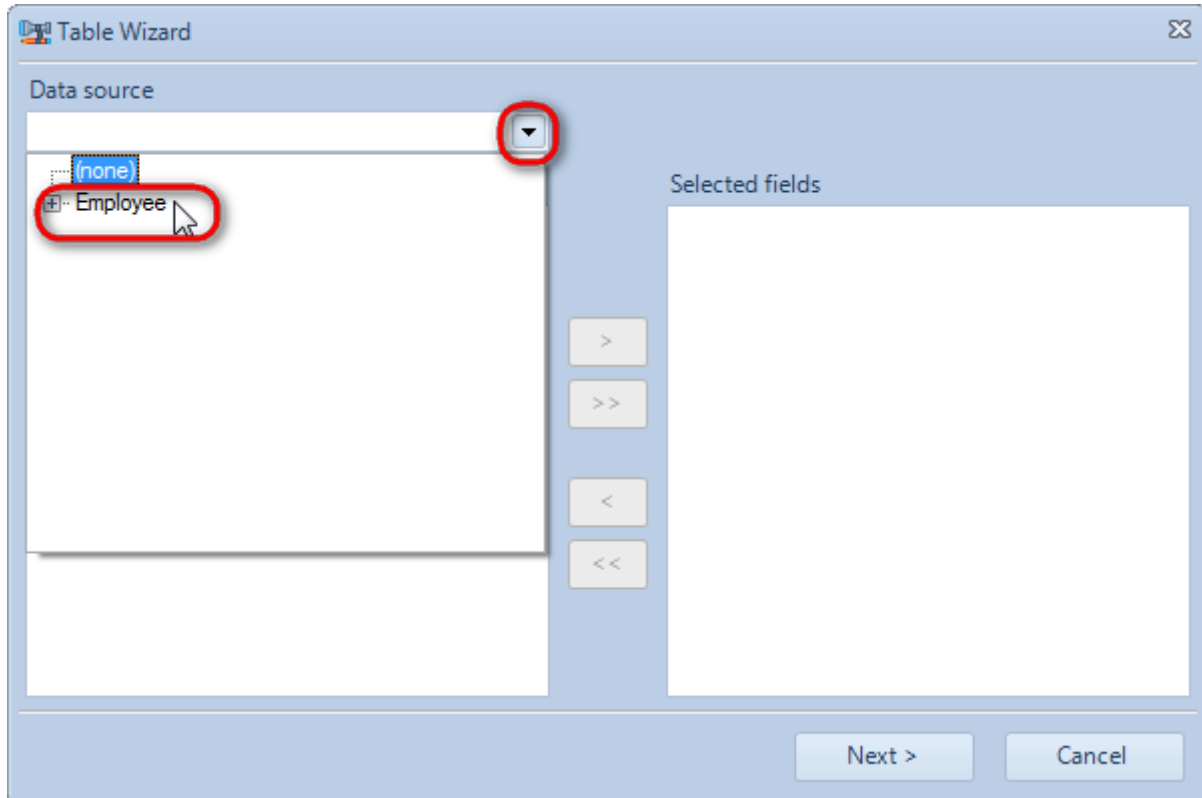
Step 13

Press "Table" button on the Insert tab in the Tables Container.



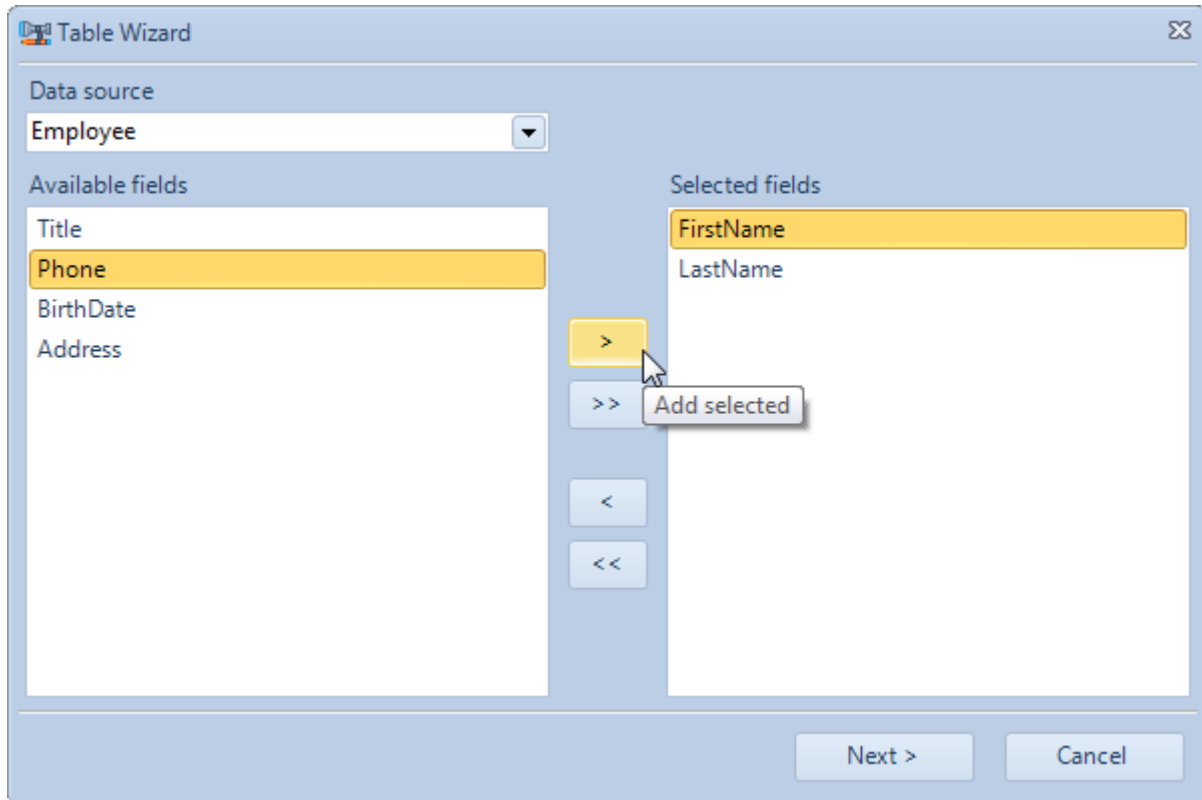
Step 14

Set data source in the property DataSource = Employee.

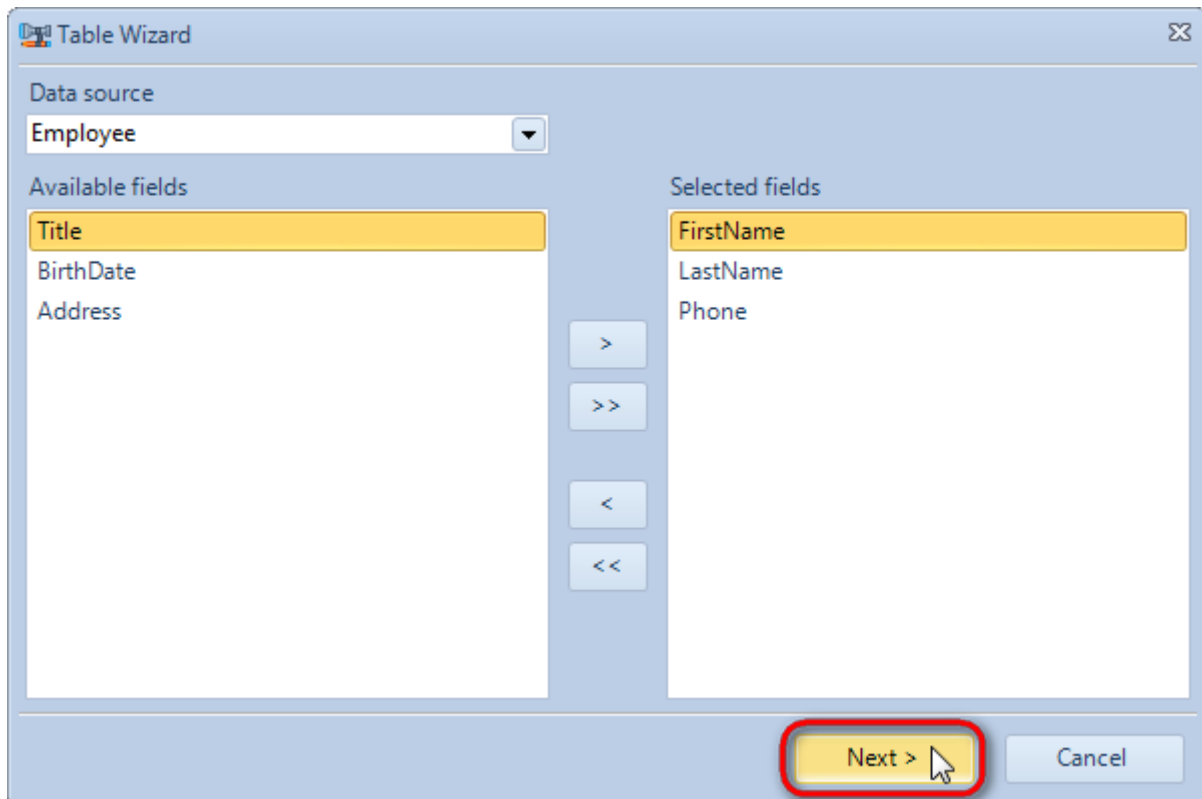


Step 15

Select the fields for displaying in the table by double-click on the "Available fields" list or by click on the ">" button.



Click "Next >".



Step 16

Change the properties of the selected fields in the appeared wizard window by double-click on the edit field.

Set Field title for the "FirstName" field to "First Name".



Set Field title for the "LastName" field to "Last Name".

Set Width property for the "FirstName" and the "LastName" fields to "4.00"

Leave the "Column count" and the "Layout type" properties as they are. Click "Next".

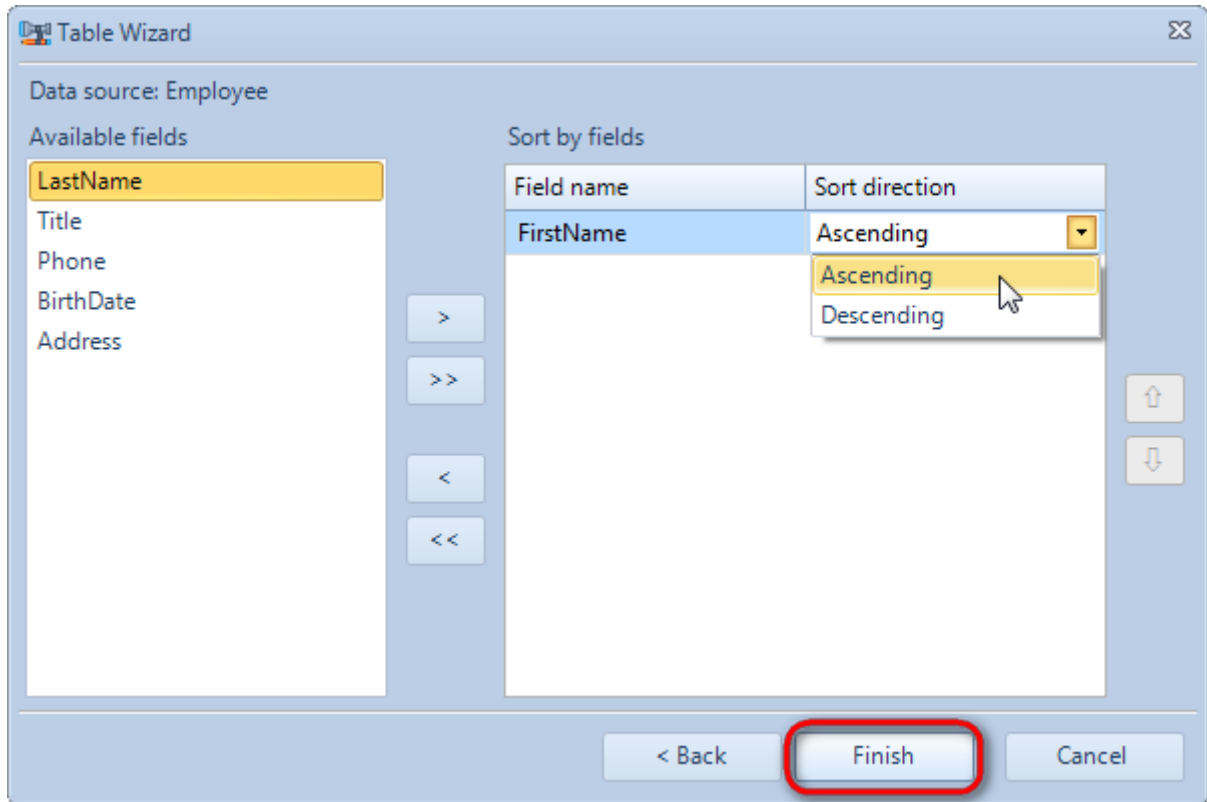
Field name	Field title	Width (cm)	Aggregate function
FirstName	First Name	4.00	
LastName	Last Name	4.00	
Phone	Phone	6.17	

Column count: 1 Layout type: List

< Back Next > Cancel

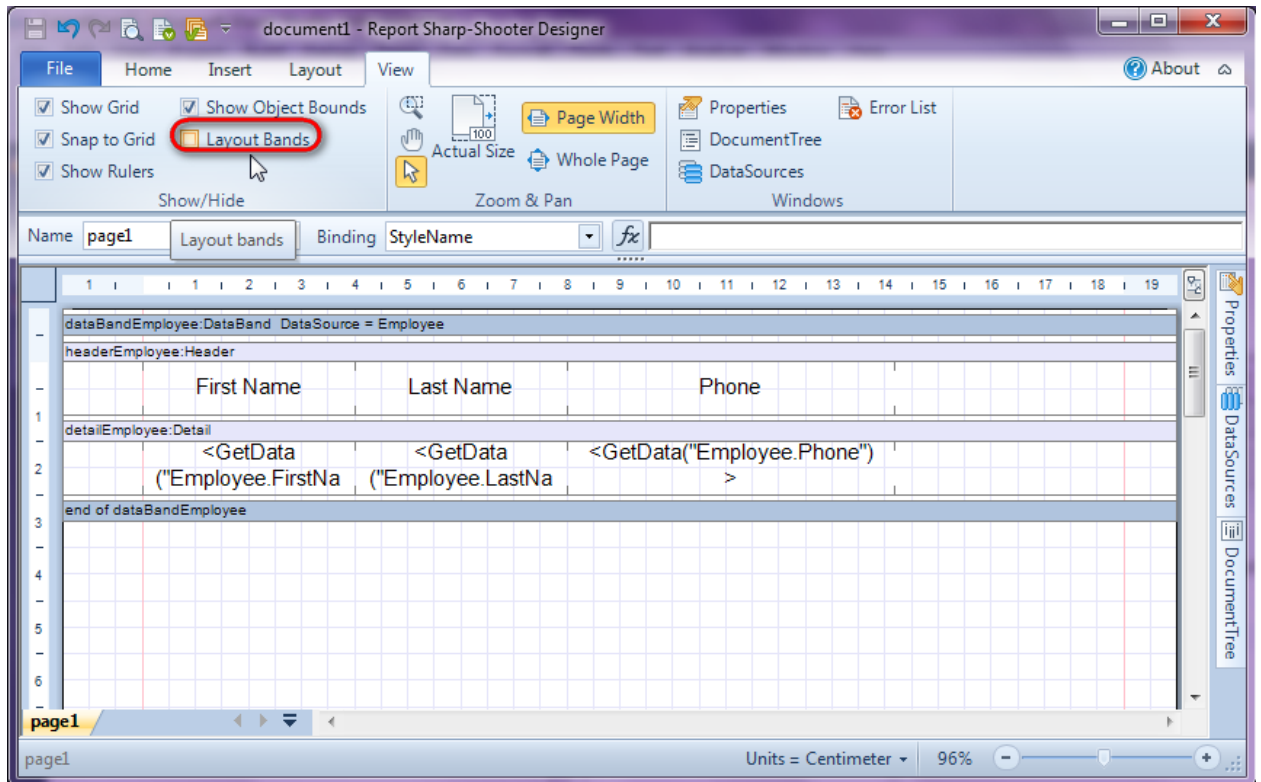
Step 17

Select the fields from the "Available fields" list to sort the table rows if the table requires sorting. Click "Finish".



Step 18

Check the "Layout Bands" on the "View" tab of the Ribbon panel to locate elements in a more representative form.



Step 19

Save the template and close the Report Designer.



Step 20

Add code responsible for displaying report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent ();

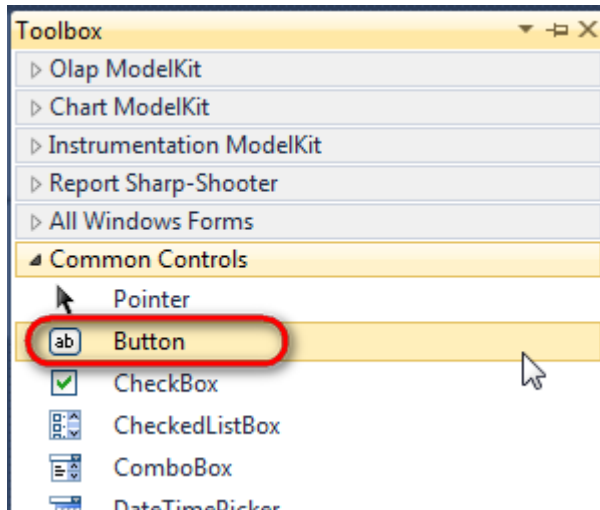
    DataRow row = dataTable1.NewRow ();
    row["FirstName"] = "Maria";
    row["LastName"] = "Anders";
    row["Title"] = "Sales Representative";
    row["Phone"] = "(71) 555-5598";
    row["BirthDate"] = "29.05.1960";
    row["Address"] = "Obere str. 57";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["FirstName"] = "Ana";
    row["LastName"] = "Trujillo";
    row["Title"] = "Owner";
    row["Phone"] = "(5) 555-4729";
    row["BirthDate"] = "01.07.1971";
    row["Address"] = "Avda. de la Constitution 222";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["FirstName"] = "Antonio";
    row["LastName"] = "Moreno";
    row["Title"] = "Owner";
    row["Phone"] = "(5) 555-3932";
    row["BirthDate"] = "12.03.1969";
    row["Address"] = "Mataderos 2312";
    dataTable1.Rows.Add (row);

    inlineReportSlot1.RenderCompleted += new
    EventHandler (reportSlot_RenderCompleted);
}

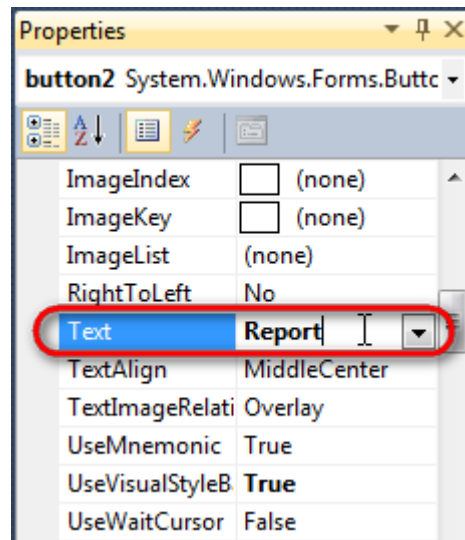
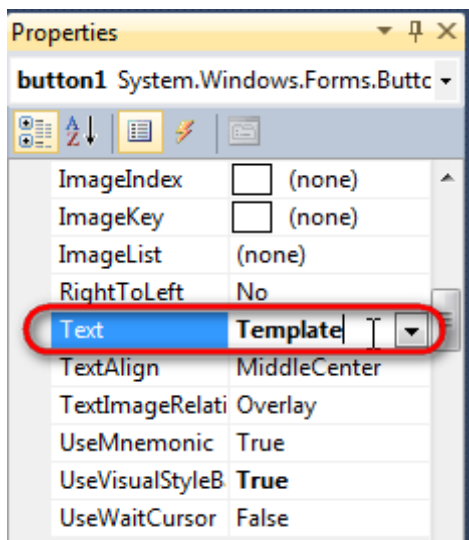
private void reportSlot_RenderCompleted (object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
    PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 21

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



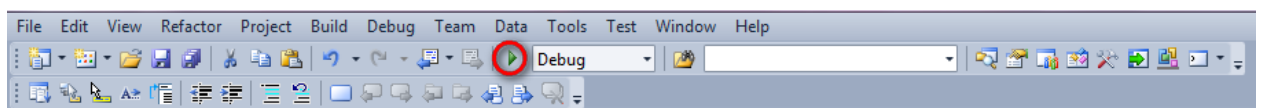
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

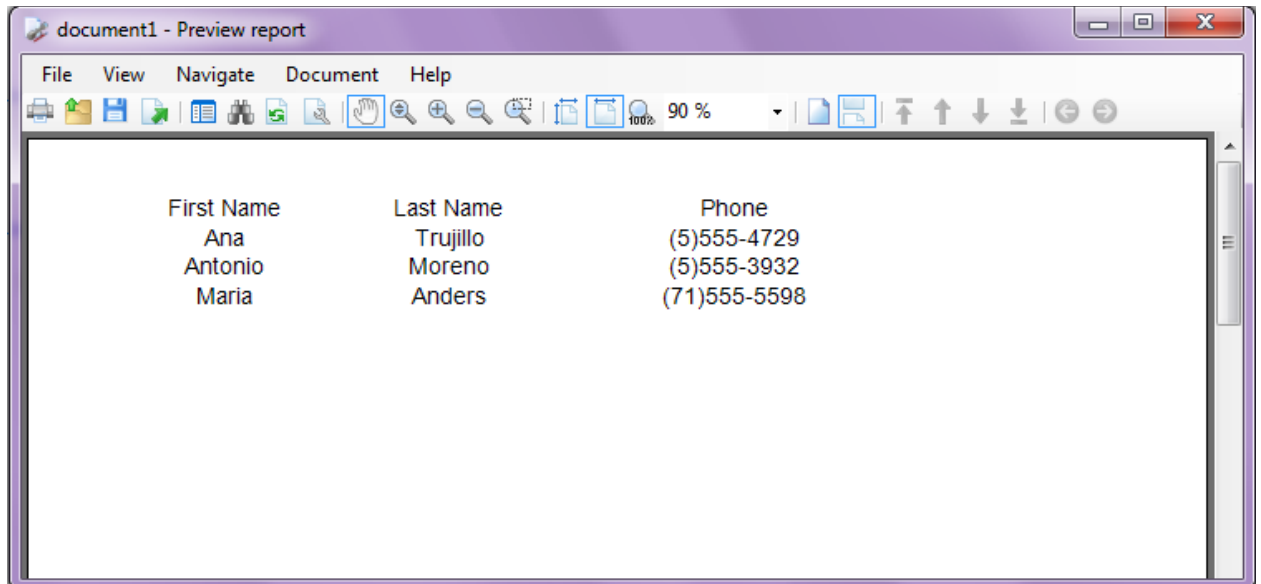
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



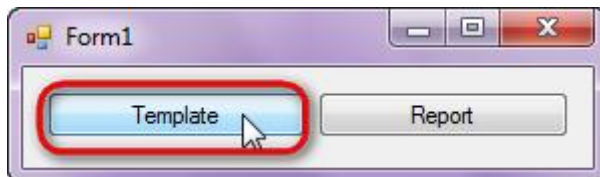
Click the “Report” button in the opened application window.



The generated report will open in a Report Viewer.



In order to edit a template, close Report Viewer and press "Template" on the application form.



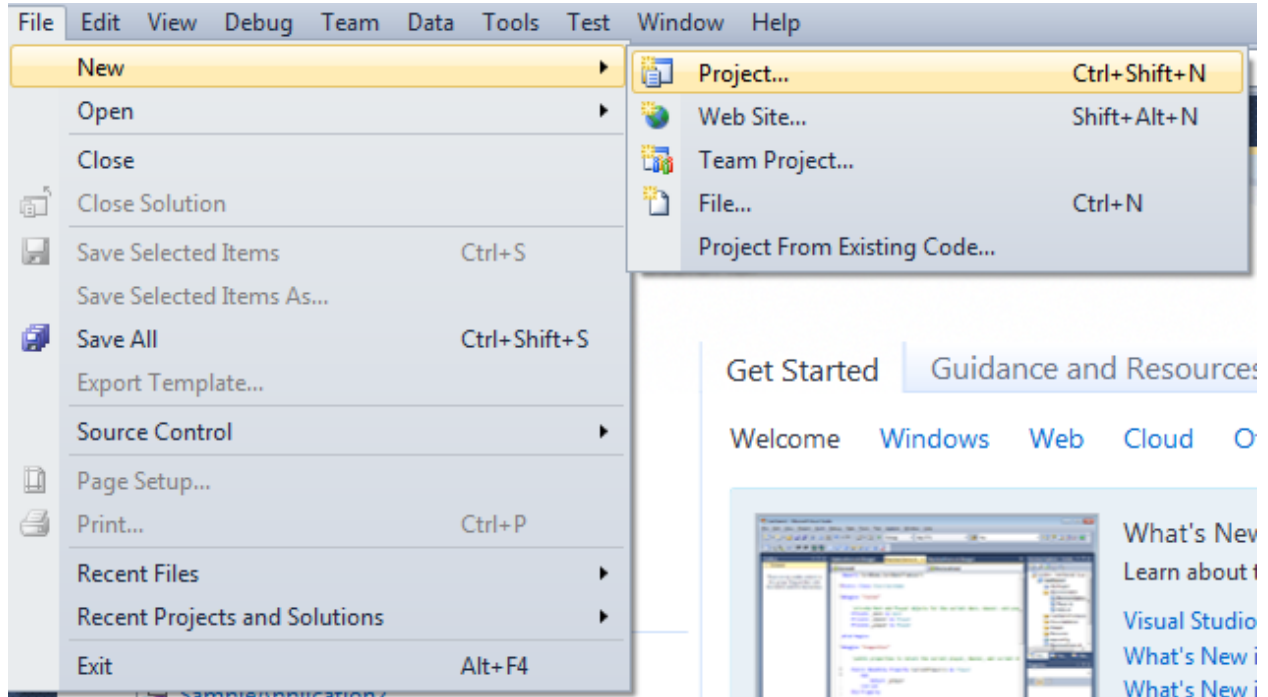


Using Scripts

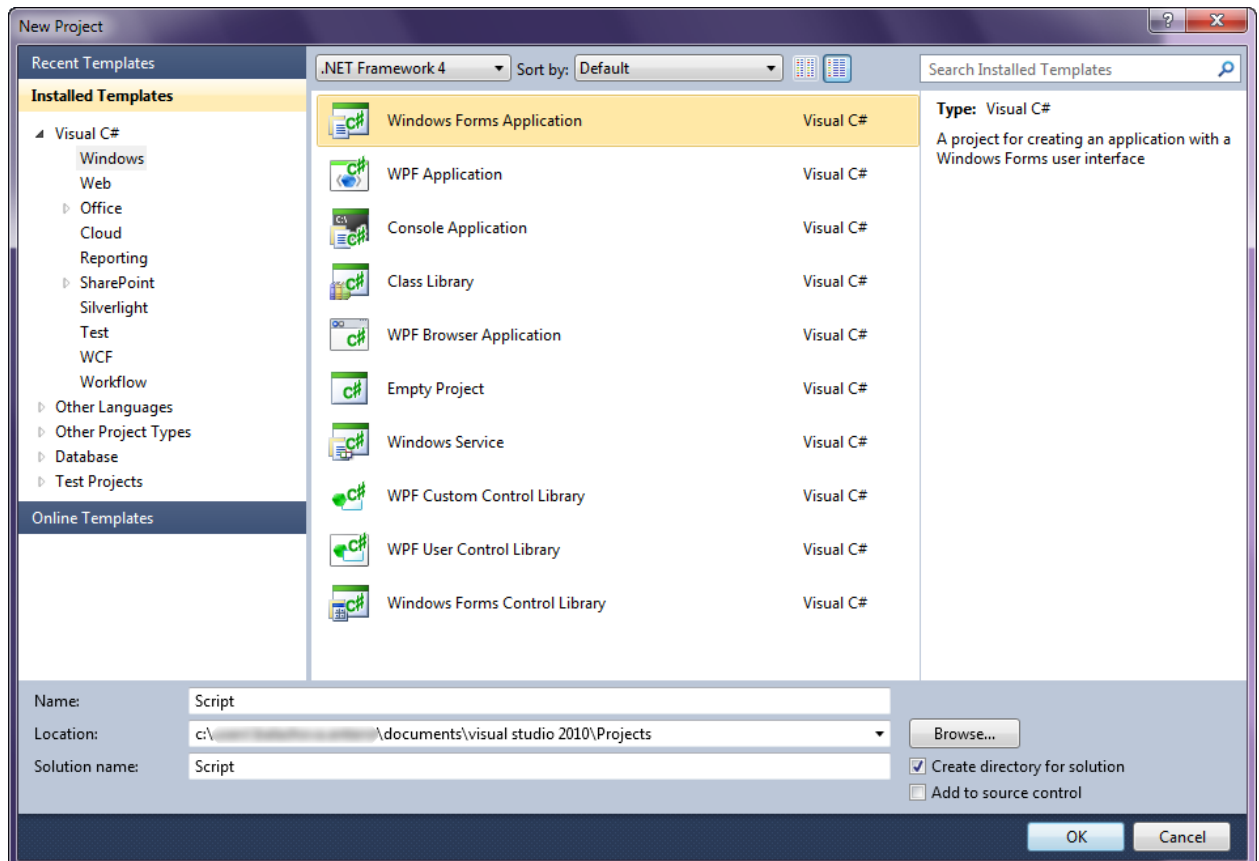
The template of a report contains numbers from 1 to 12 in the form of dial. The orientation of TextBox element can be changed with the help of scripts depending on line number.

Step 1

Create a new project in Microsoft Visual Studio. Select New\Project from the main menu.

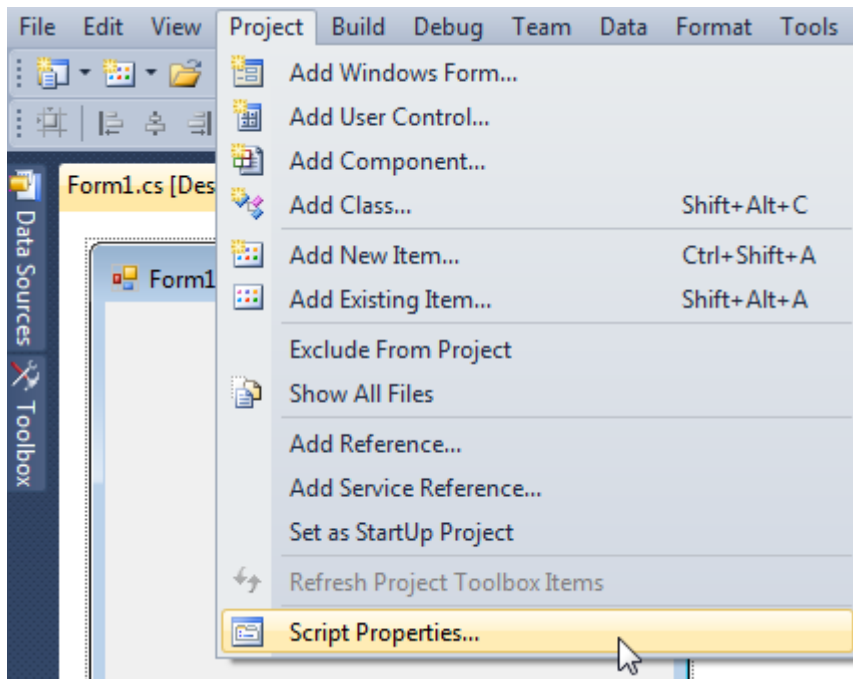


Select Windows Forms Application, set the project name – "Script", and directory to save the project to.

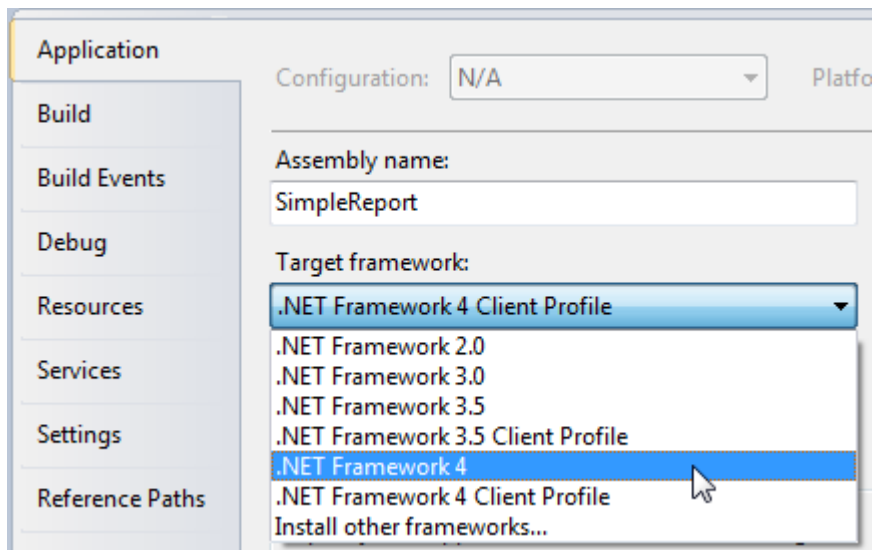


Step 2

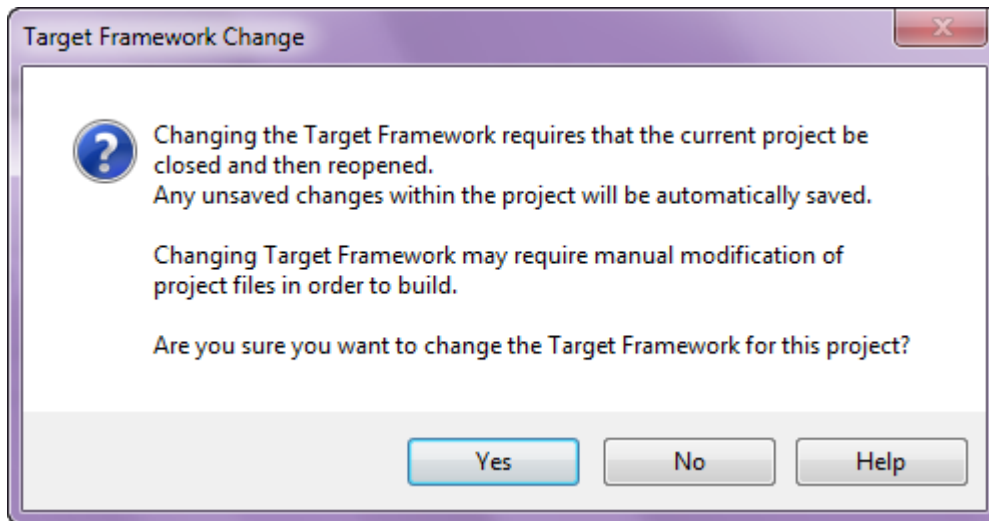
Change the project properties. Select the Project\Script Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

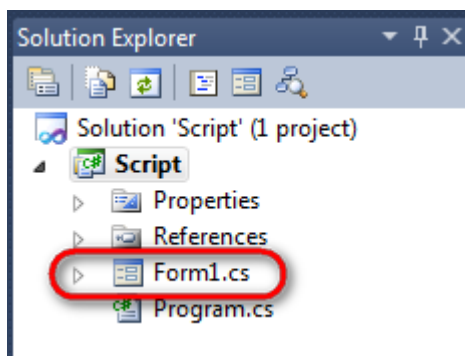


Press the "Yes" button in the opened window.

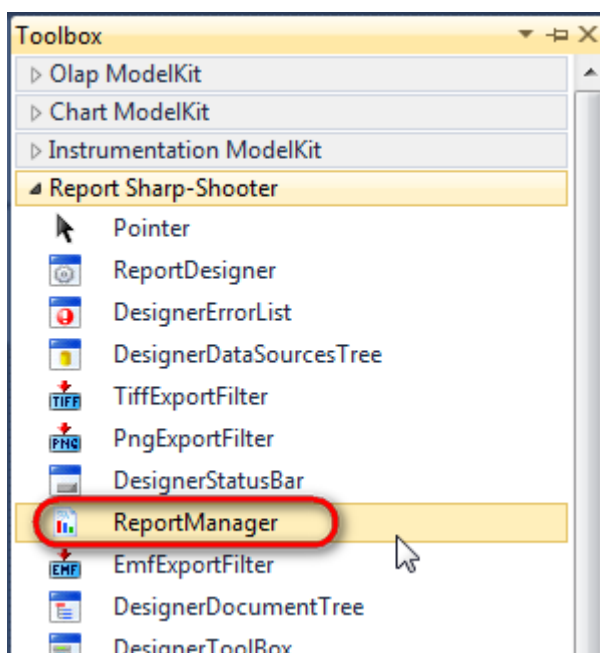


Step 3

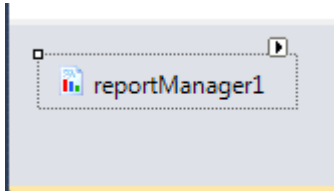
Open the main form of the application in the editor by double click on "Form1.cs" in the Solution Explorer.



Drag and drop the "ReportManager" component from the Toolbox onto the form. This component is used to store collections of report templates and data sources.

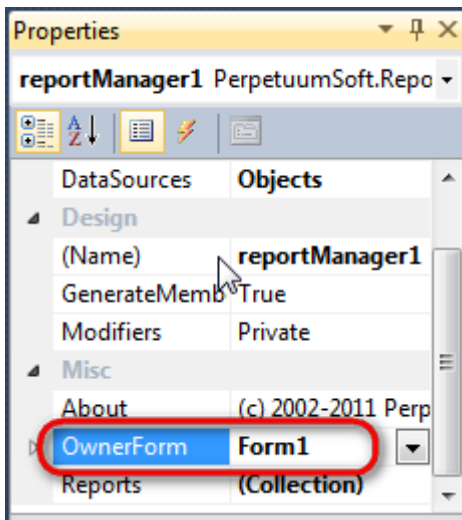


The component is displayed in the lower part of the window.



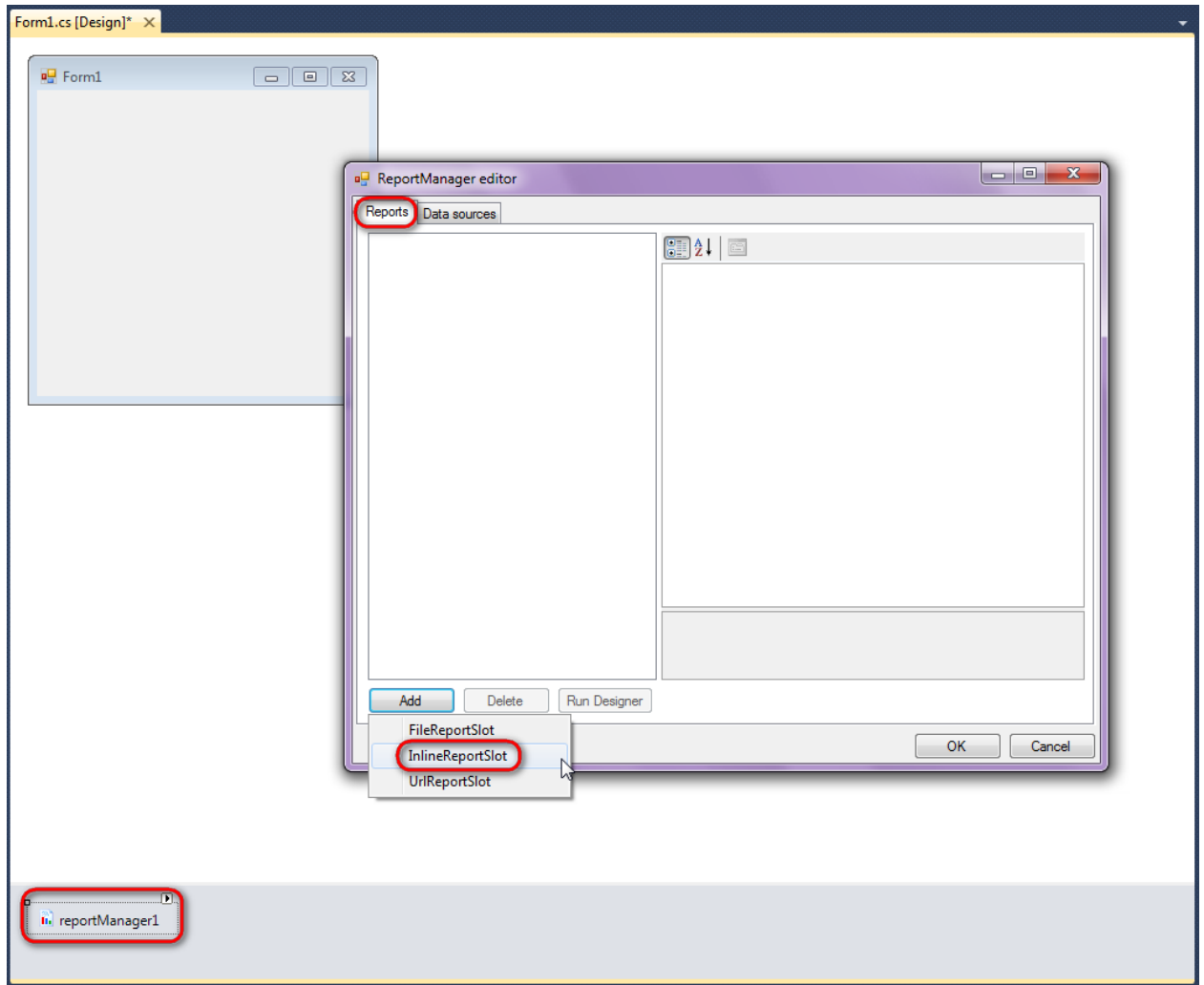
Step 4

On the property grid, initialize the OwnerForm property of the ReportManager component by selecting a form on which it is located.



Step 5

Double click on reportManager1 and open the ReportManager editor.

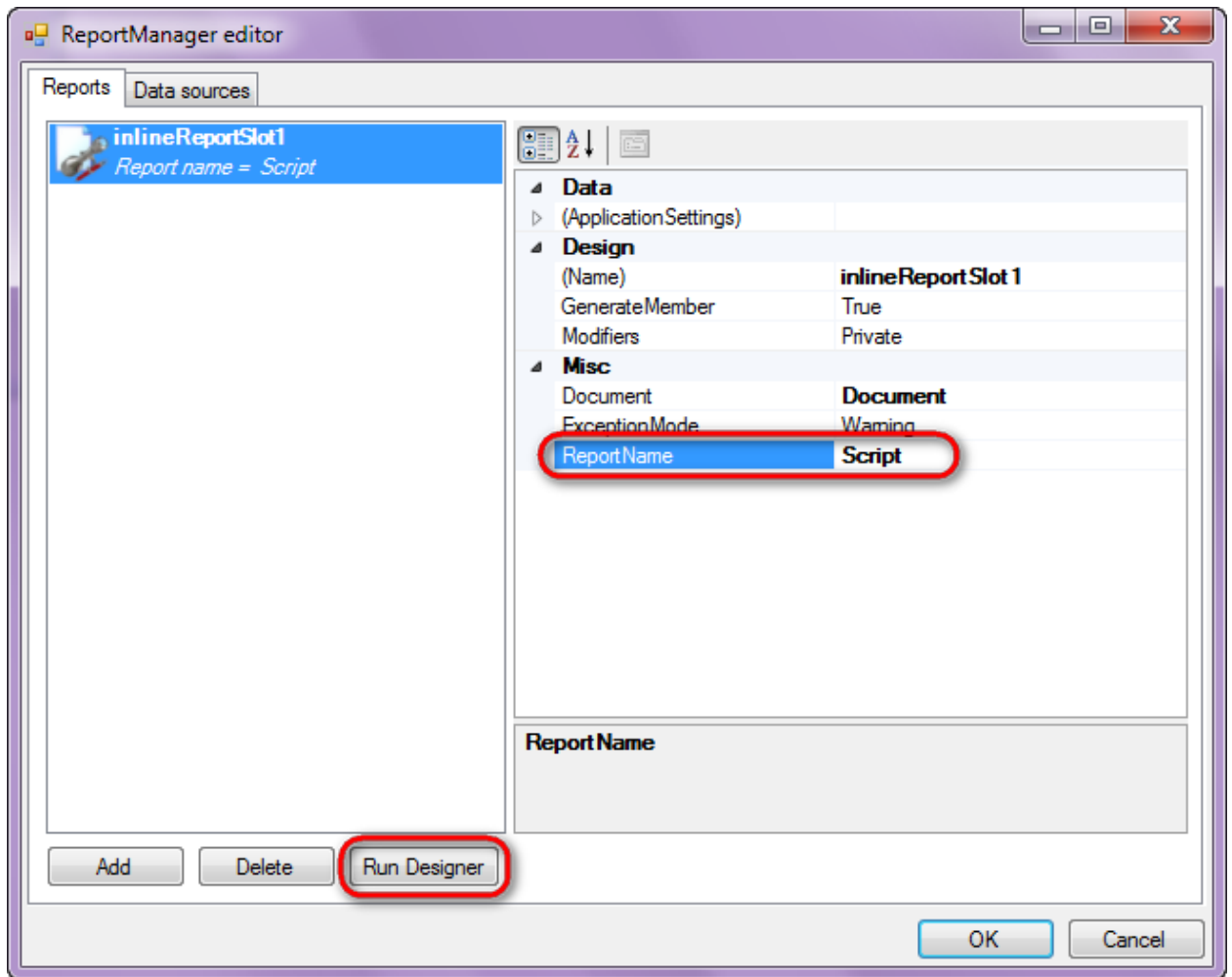


Click the "Add" button on the "Reports" tab and select "InlineReportSlot".

Step 6

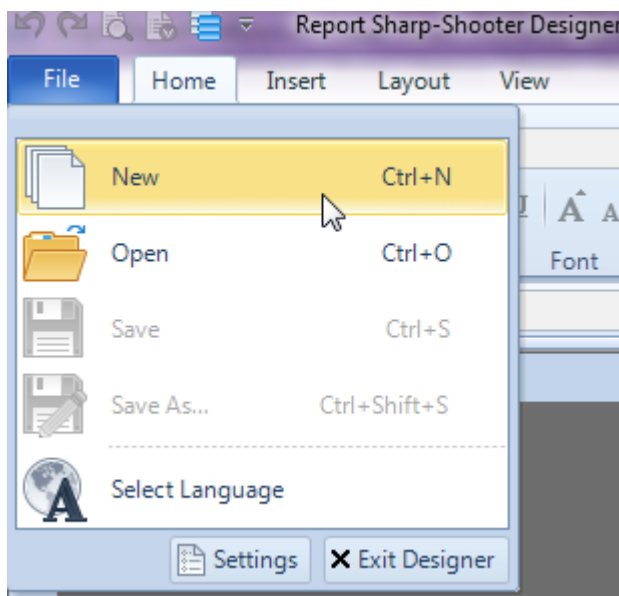
Set the report name in the ReportName - "Script" property.

Click "Run Designer" to open the Report Designer template editor.

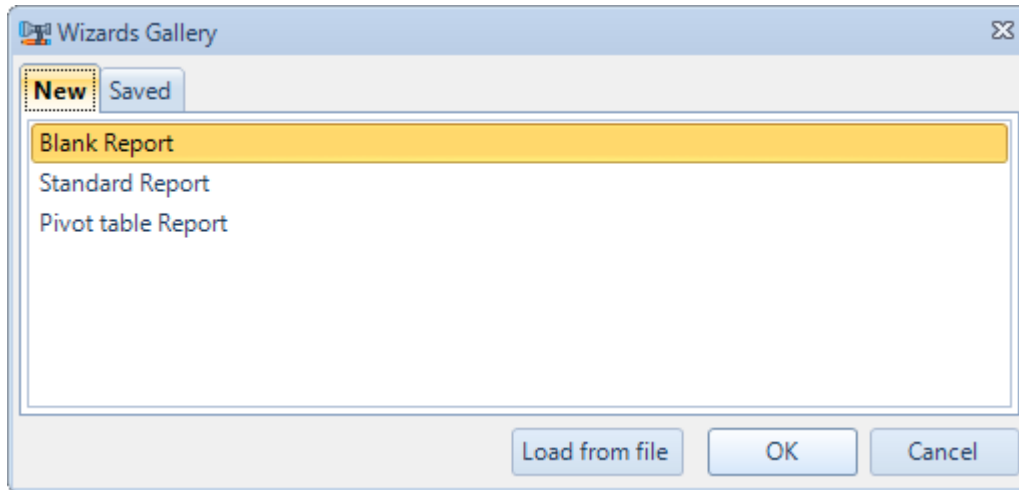


Step 7

Create a new empty template – select the File\New item in the main menu.

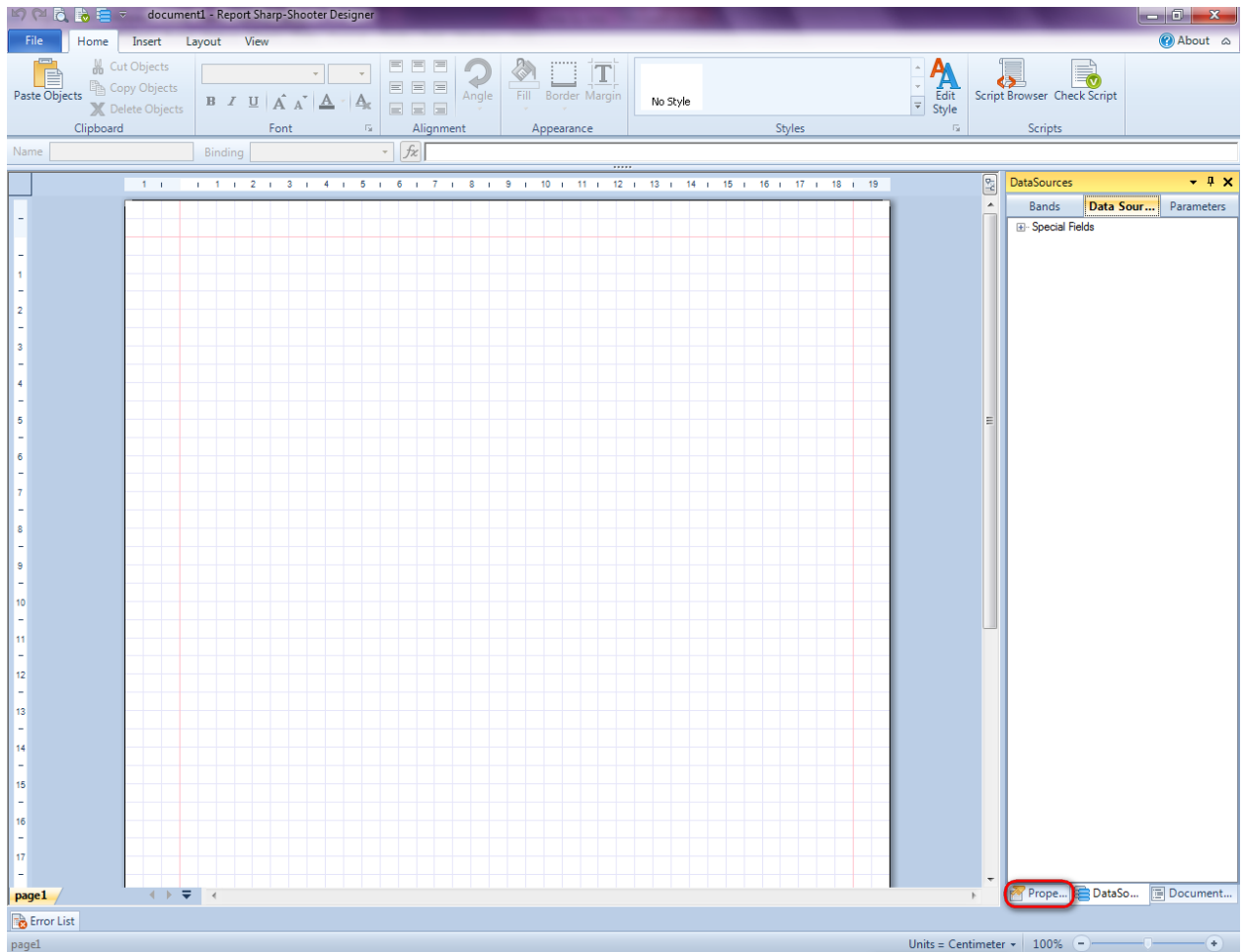


Select "Blank Report" in the Wizards Gallery and click "OK".



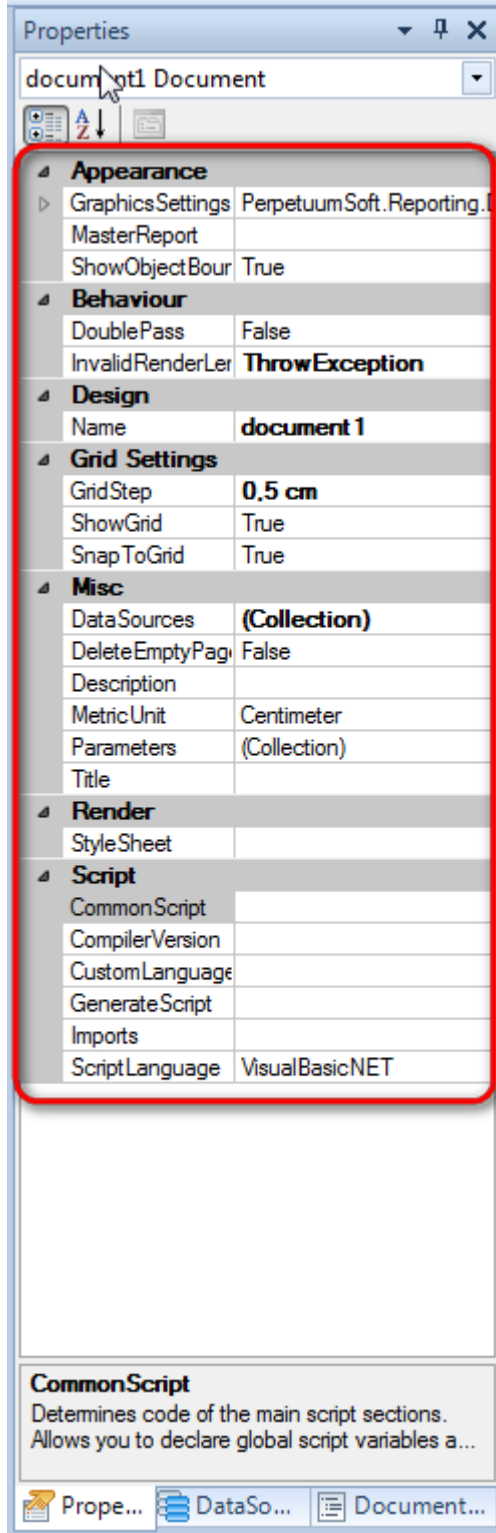
Step 8

Click the "Properties" tab of the tool window in the right part of the designer.

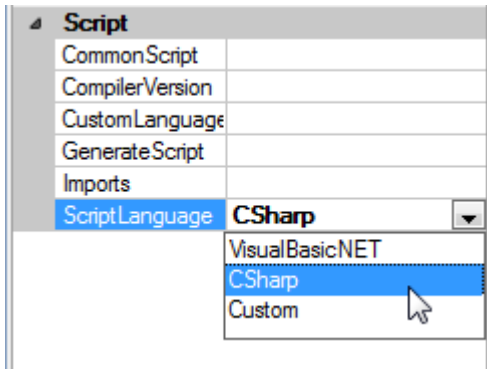




You will see properties of the edited template on the "Properties" tab

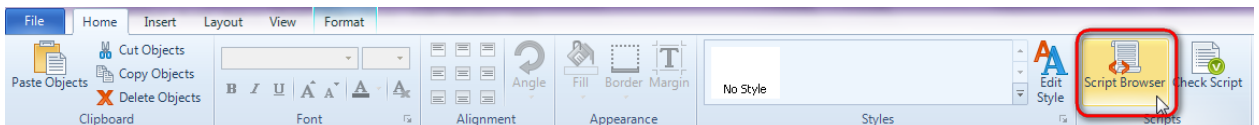


Set property ScriptLanguage = CSharp.

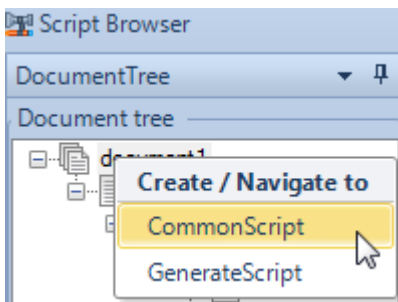


Step 9

To open Script Browser press the "Script Browser" button in the Home tab in the Scripts group.

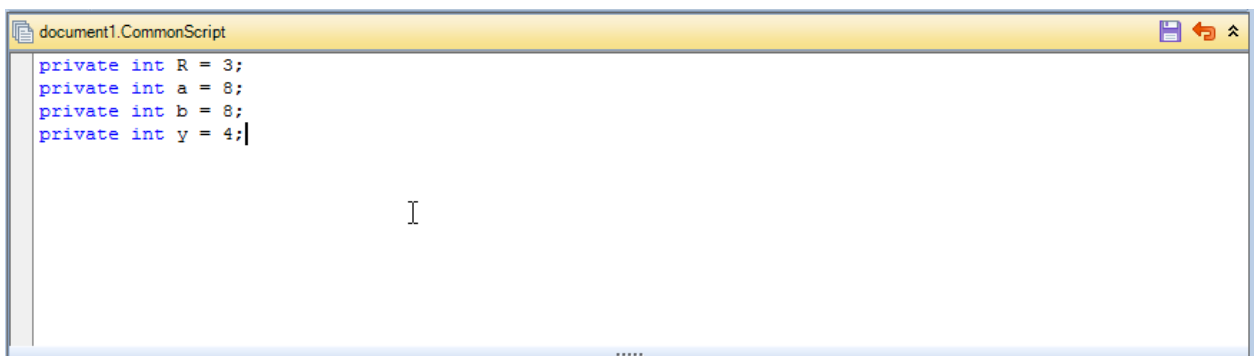


Right-click on Document1 in the DocumentTree. Select CommonScript in appeared context menu.



Write the following code in the opened window:

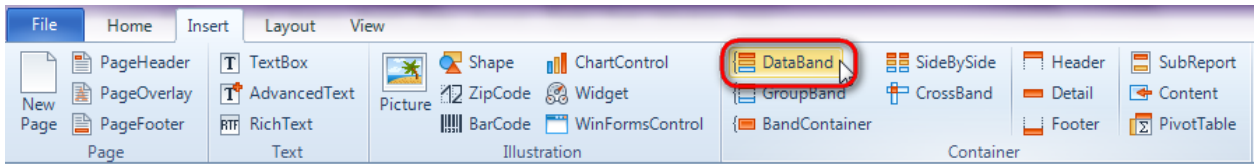
```
"private int R = 3;  
private int a = 8;  
private int b = 8;  
private int y = 4;"
```



Click the "Save all" button and close the Script Browser.

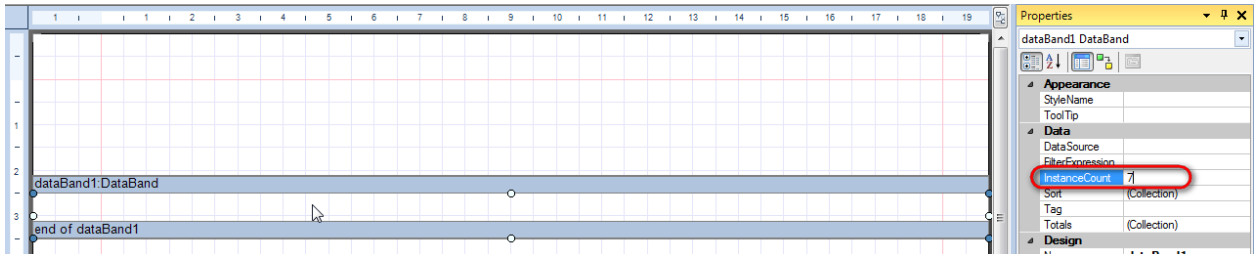
Step 10

Press the "DataBand" button in the Insert tab in the group Container.



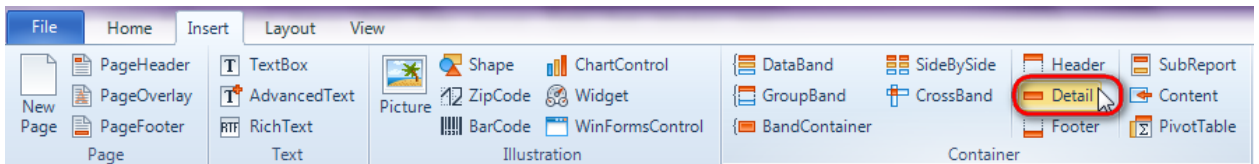
Click on the template area to add DataBand band.

Set property InstanceCount = 7.

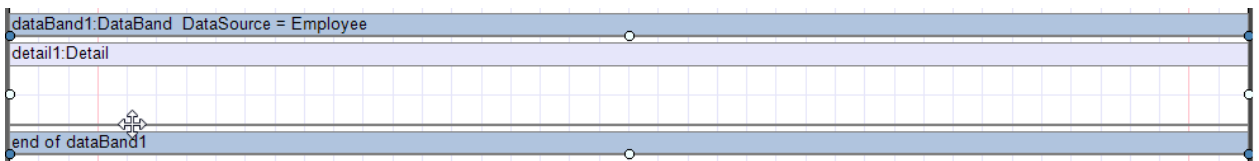


Step 11

Press the "Detail" button in the Insert tab in the group Container.



Click on the DataBand area to add the Detail band inside DataBand.



Step 12

Press the "TextBox" button in the Insert tab in the group Text.

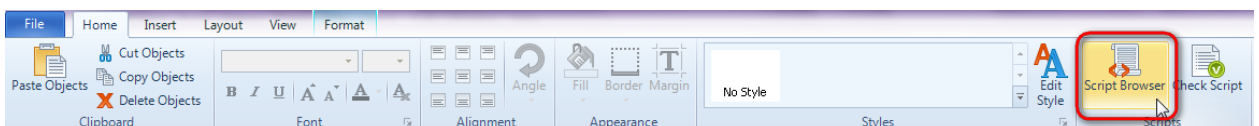


Click on the Detail area to add TextBox inside Detail band.

Add one more TextBox in the same way.

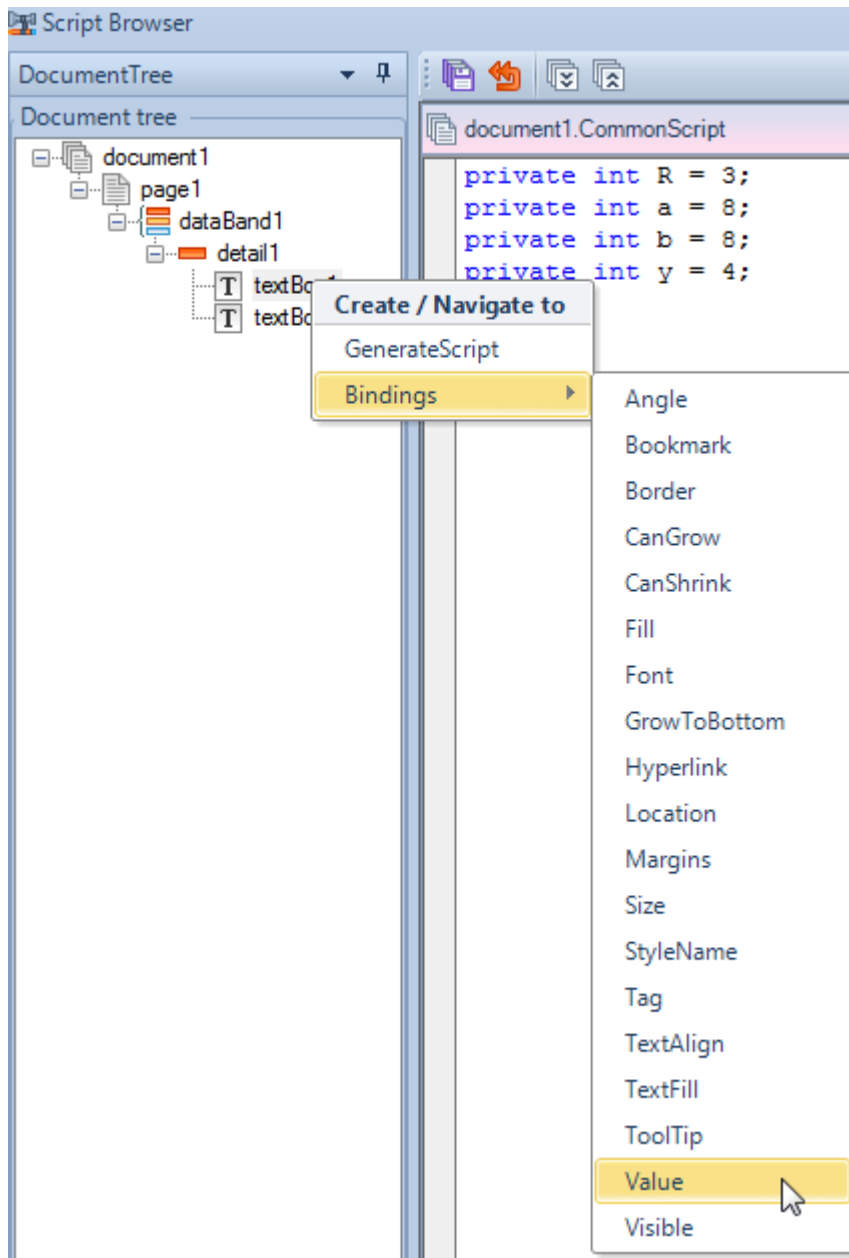
Step 13

To open Script Browser press the "Script Browser" button in the Home tab in the Scripts group.

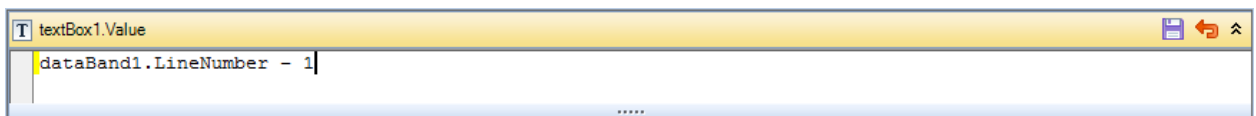




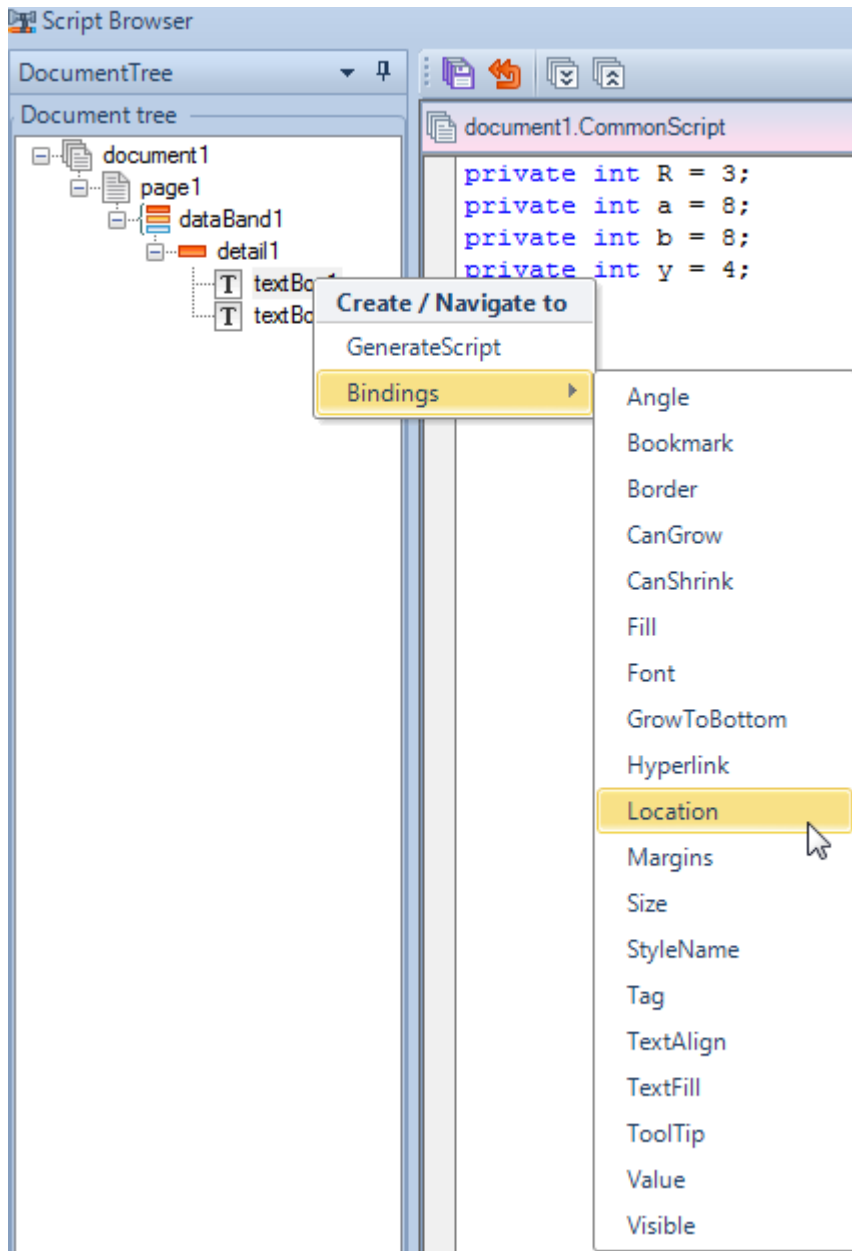
Right-click on textBox1 in the DocumentTree. Select Bindings\Value in appeared context menu.



Write the following code in the opened window: "dataBand1.LineNumber - 1".

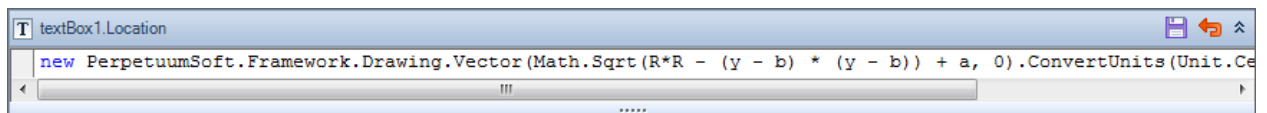


Right-click on textBox1 in the DocumentTree. Select Bindings\Location in appeared context menu.



Write the following code in the opened window:

```
"new PerpetuumSoft.Framework.Drawing.Vector(Math.Sqrt(R*R - (y - b) * (y - b)) + a, 0).ConvertUnits(Unit.Centimeter,Unit.InternalUnit)"
```



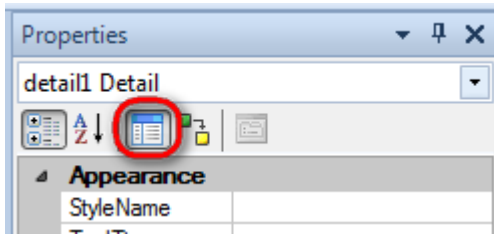
Step 14

In the same way, set the Value property to "13 - dataBand1.LineNumber", Location property - "new PerpetuumSoft.Framework.Drawing.Vector(-Math.Sqrt(R*R - (y - b) * (y - b)) + a, 0).ConvertUnits(Unit.Centimeter,Unit.InternalUnit)" to TextBox2.

Click the "Save all" button and close the Script Browser.

Step 15

Select the Detail band. Click the "Properties" button in the Properties tab.



Edit band size: Size: Y = 0 cm.


Layout	
Location	0; 0,5 cm
Size	21; 0 cm
X	21 cm
Y	0 cm

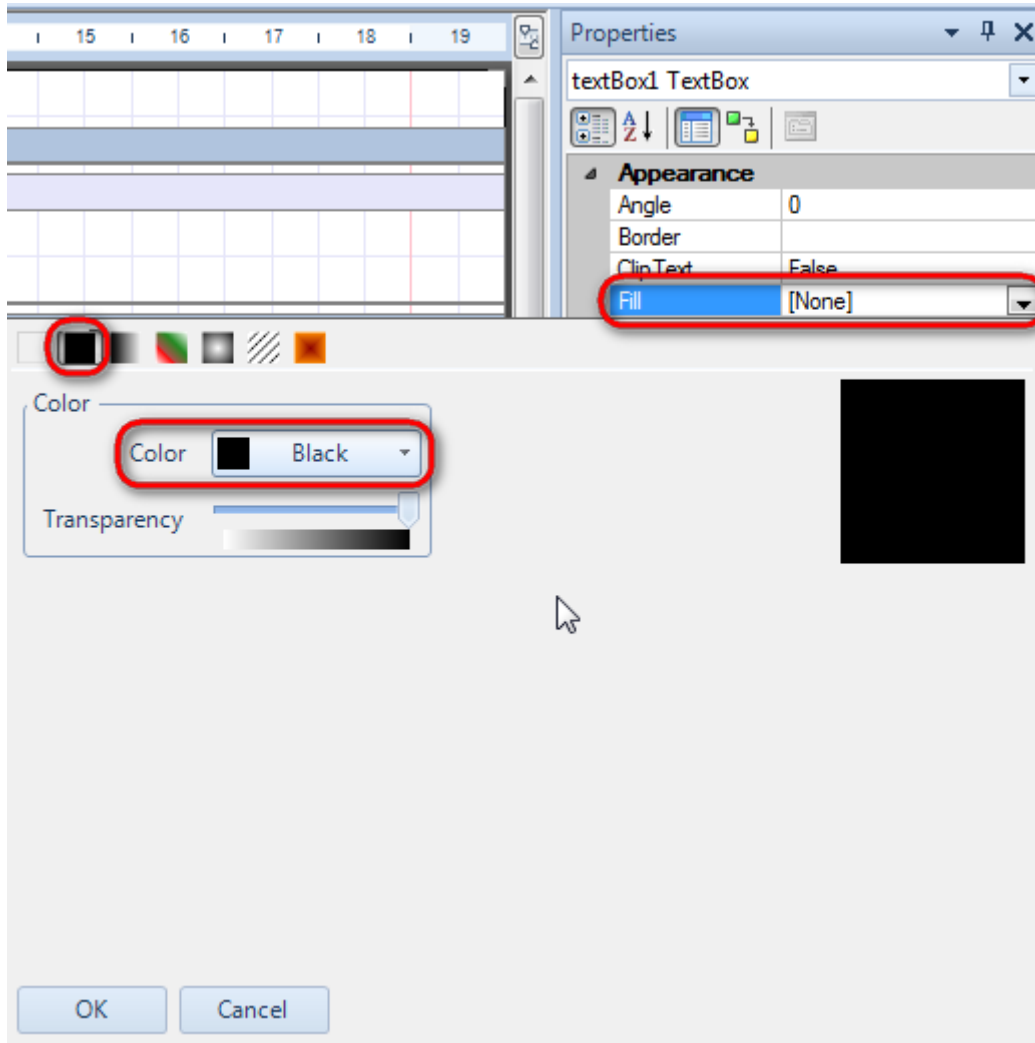
Step 16


Now it is necessary to modify TextBoxes appearance.

Select both TextBoxes. Set the Size property: X = 1 cm, Y = 0,5 cm.

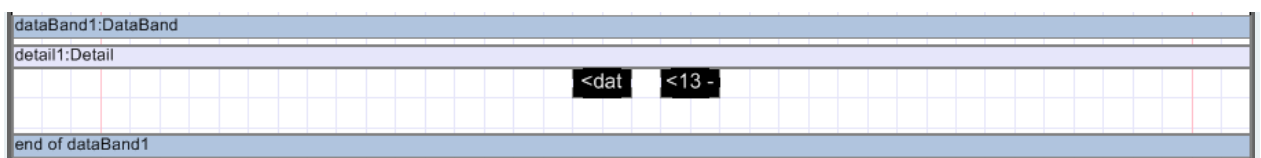
Layout	
Location	
Size	1; 0,5 cm
X	1 cm
Y	0,5 cm

Select the Fill property in the "Properties" tab, click the  button. Select SolidFill, Color = Black property.



Select the TextFill property in the "Properties" tab, click the  button. Select "SolidFill", Color = White property.

The report template looks as follows:

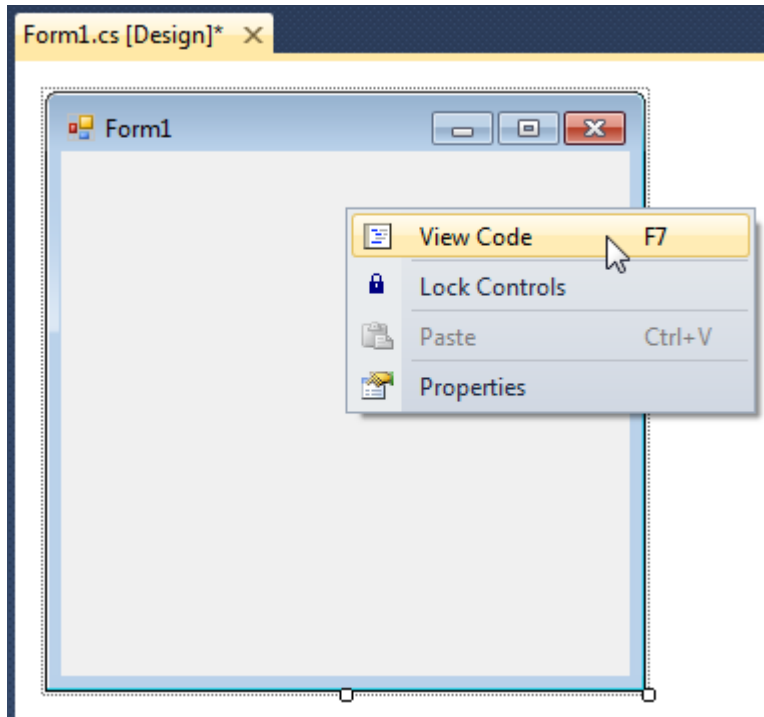


Step 17

Save the template and close Report Designer.

Step 18

Right click on the form and select "View Code" in the context menu to view code.

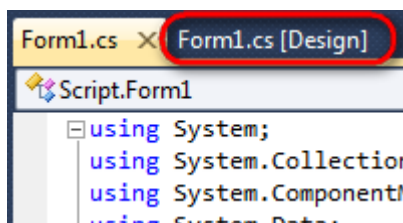


Add code for report display to the class constructor. Write the RenderCompleted event handler of the InlineReportSlot object.

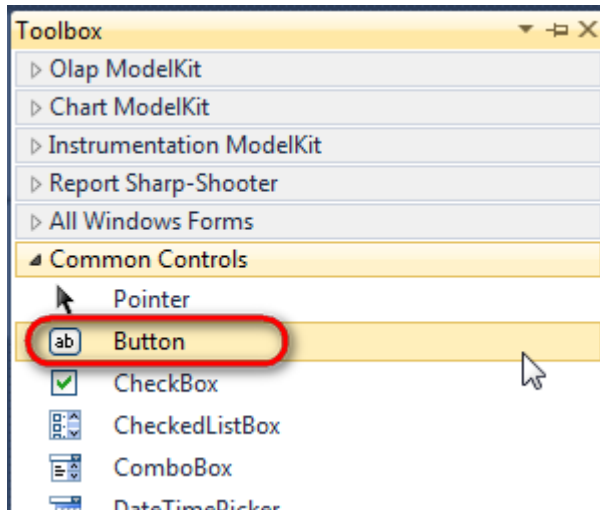
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 19

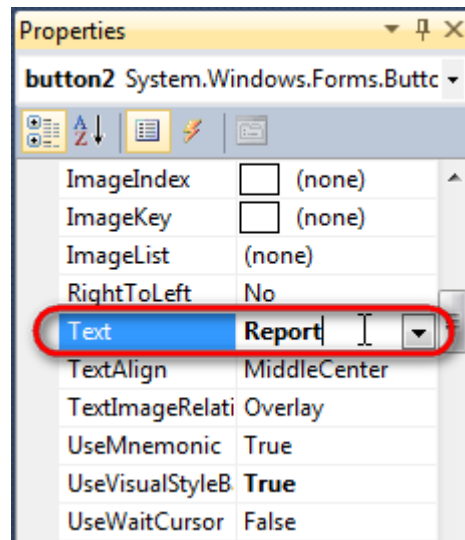
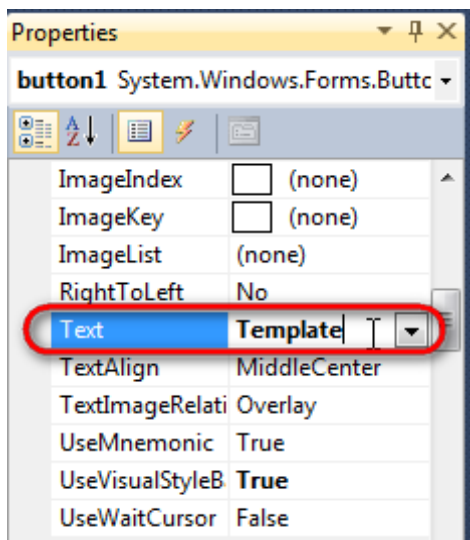
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop the "Button" element from the Toolbox onto the form).



Select the Button element in the form; edit the Text property in the property grid. Set Text = Template for the first button and Text = Report for the second one.



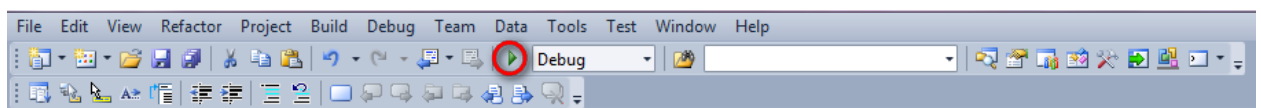
Create the Click event handlers for the buttons – double click on the Button element in the form. Add code, which launches the report generation, to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

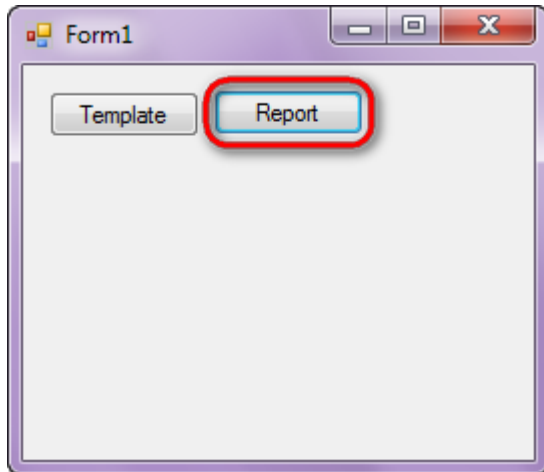
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 20

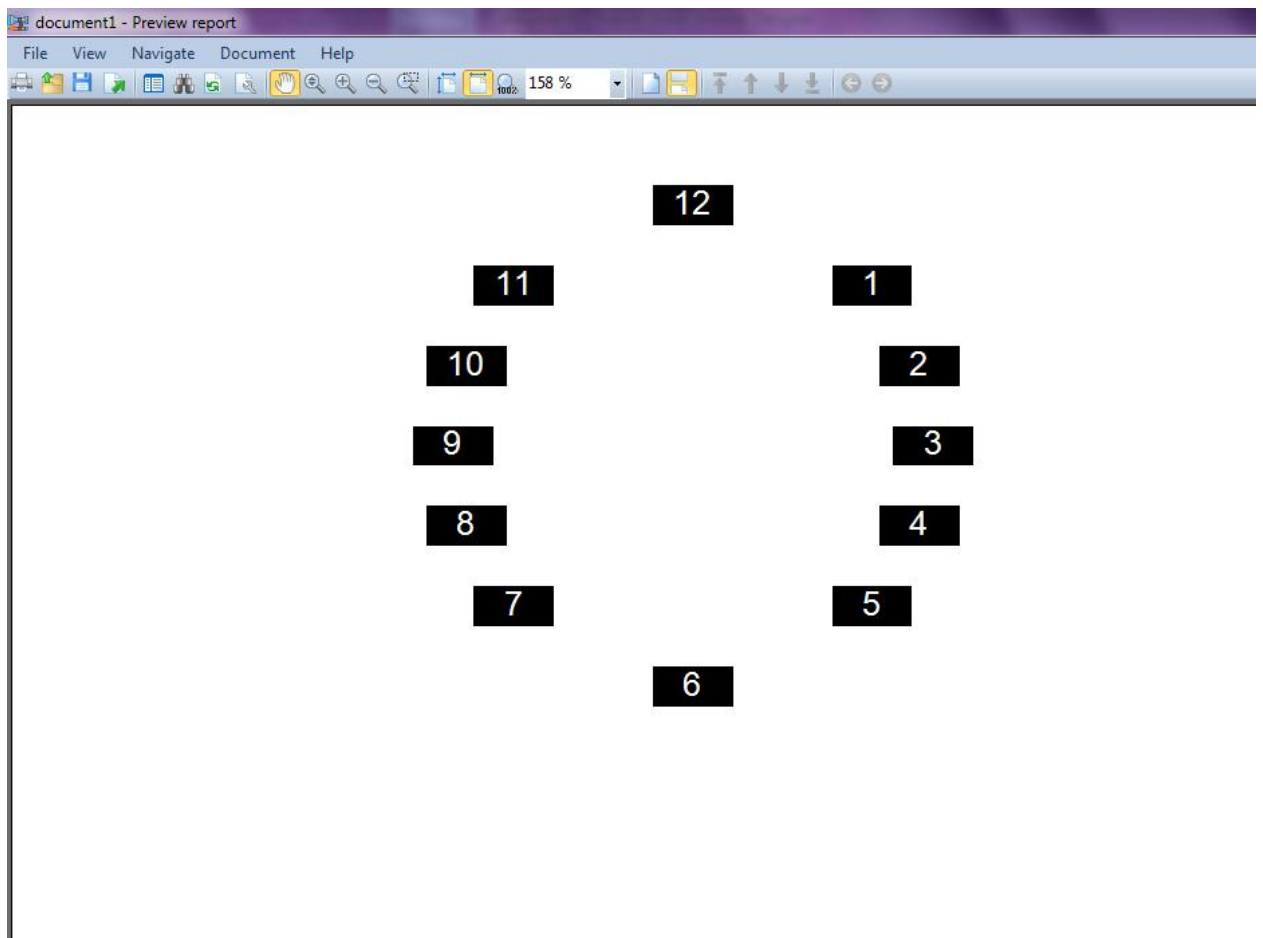
Click “Start Debugging” in the Visual Studio toolbar in order to run the application.



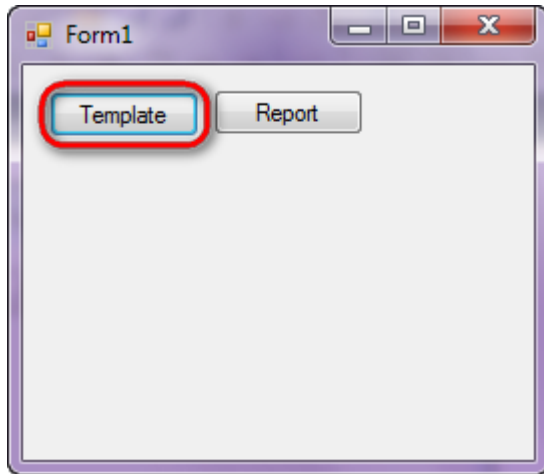
Click the “Report” button in the opened application window.



The generated report can be opened with Report Viewer.



In order to edit the template, close Report Viewer and press "Template" in the application form.



Similar sample can be found in the Samples Center – Scripting & Binding\Location changing.

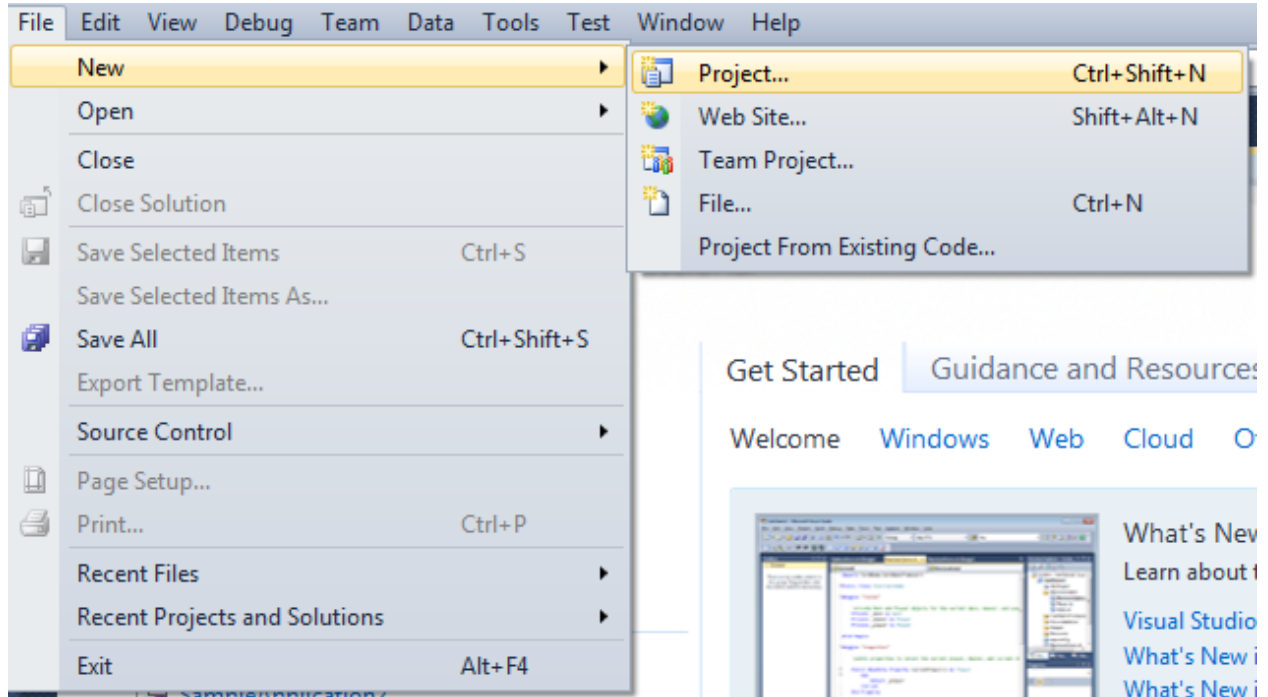


List

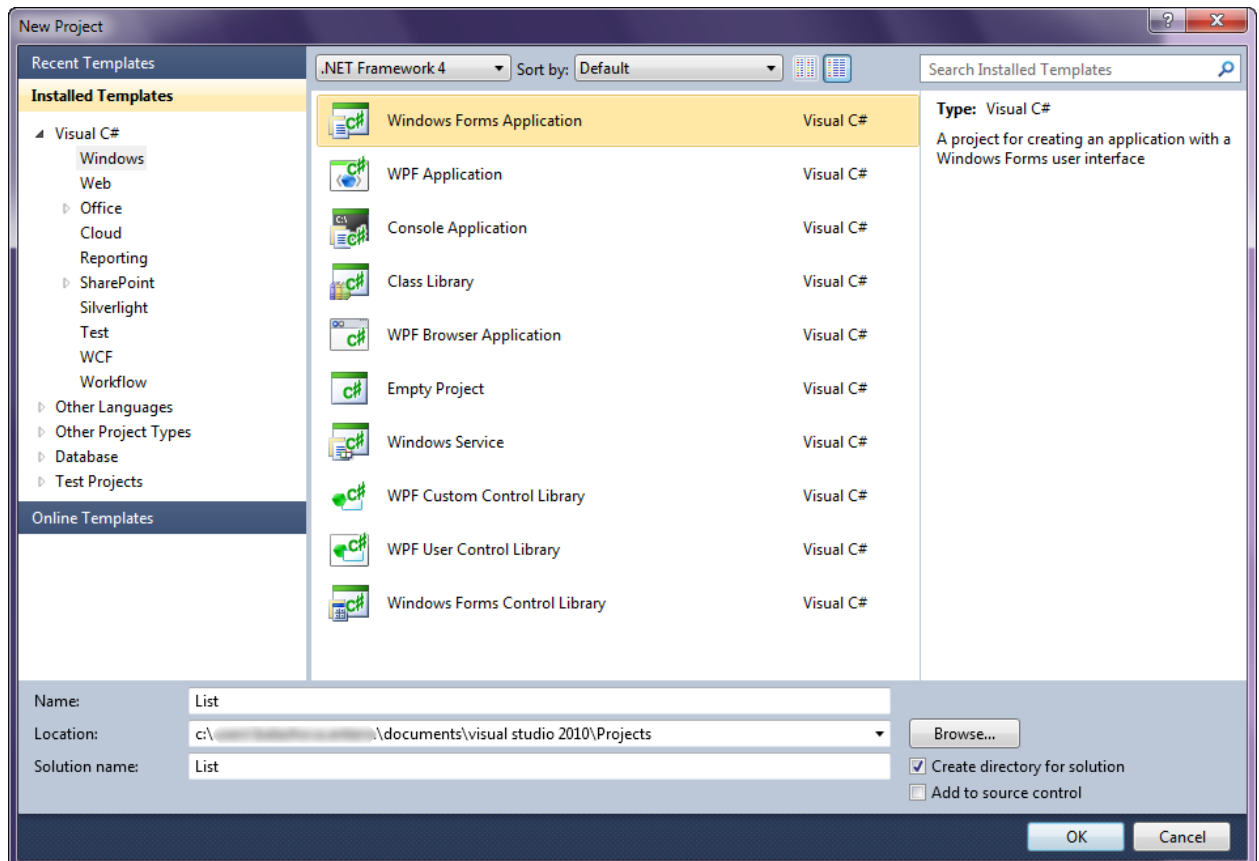
Template of a report containing a list of customers including company address, contact person and phone.

Step 1

Create a new project in Microsoft Visual Studio. Select New\Project from the main menu.

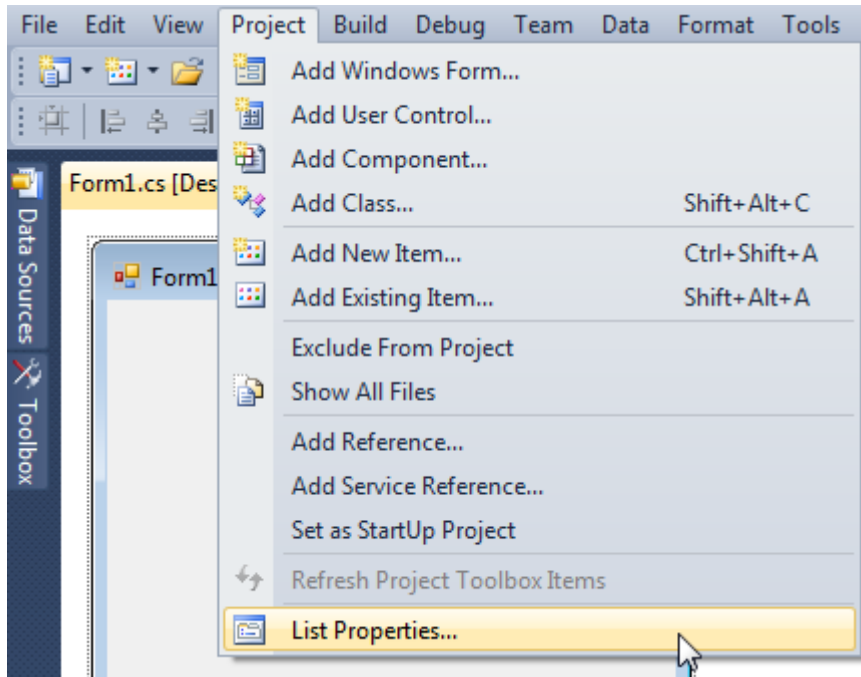


Select Windows Forms Application, set project name – “List”, set directory to save the project to.

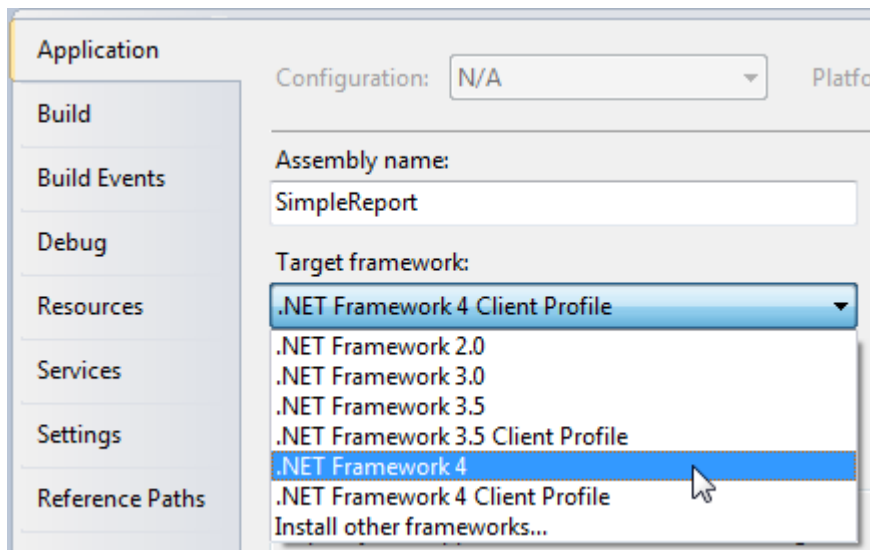


Step 2

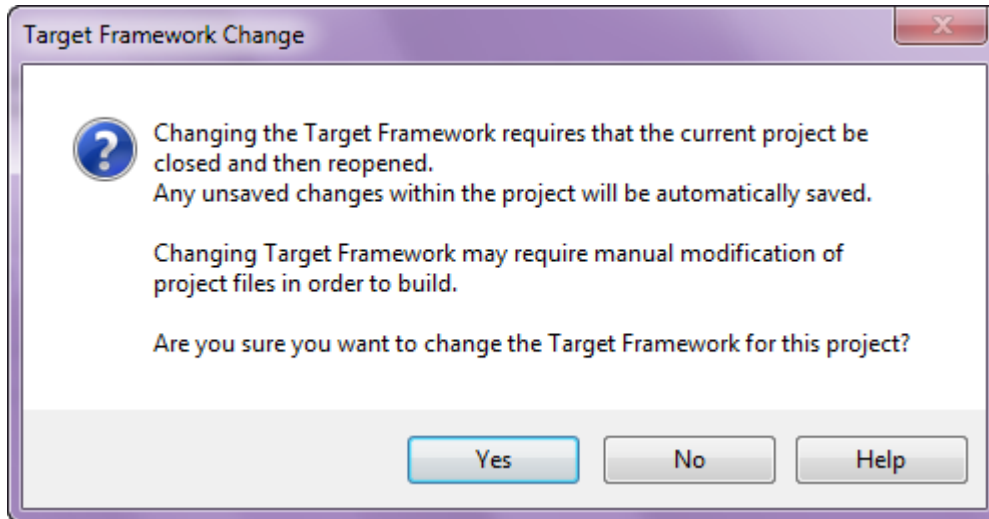
Change the project properties. Select the Project\List Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

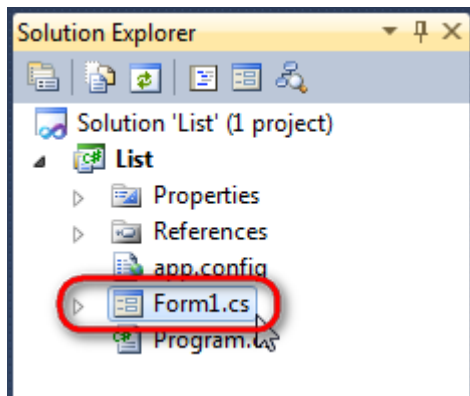


Press the "Yes" button in the opened window.

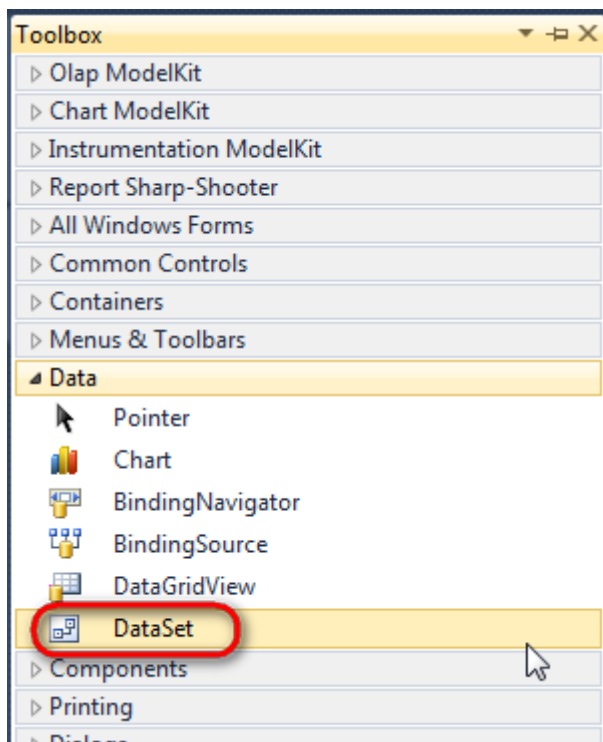


Step 3

Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.

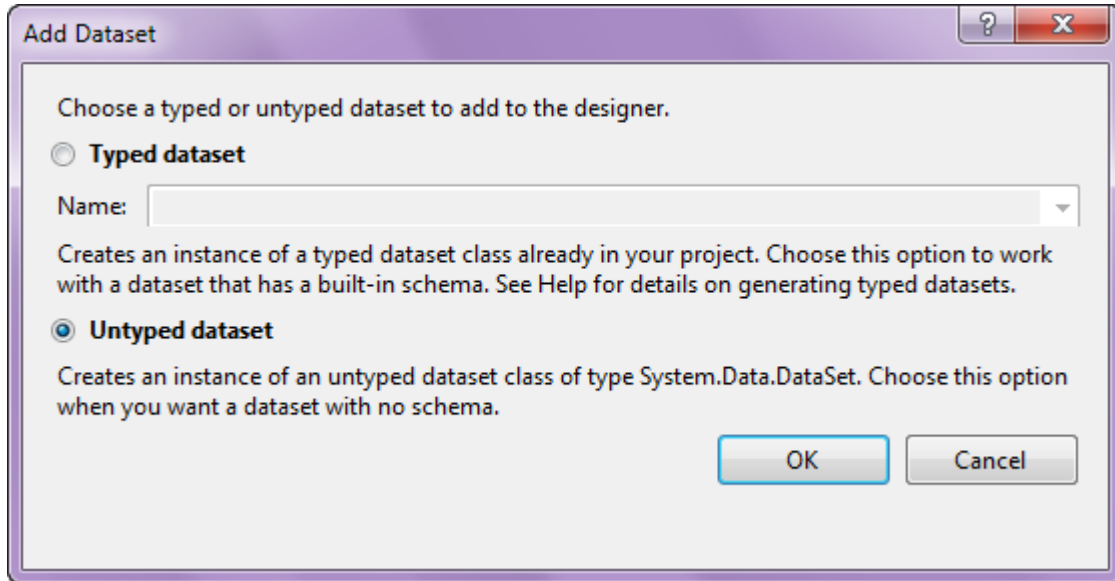


Click "DataSet" element on the Toolbox and place DataSet onto the form.

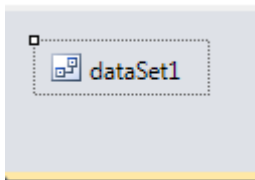





Select "Untyped dataset", click "OK"

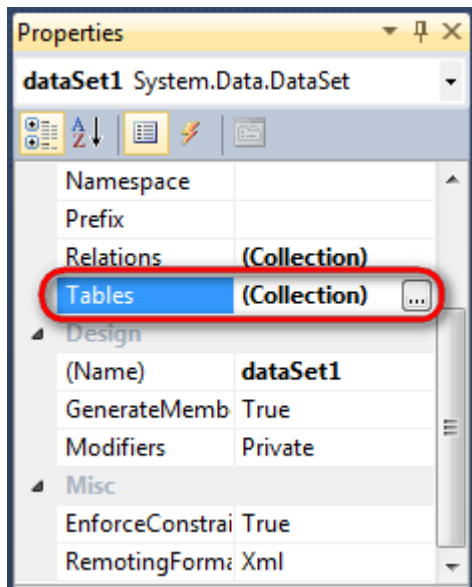


The component is available in the lower part of the window.

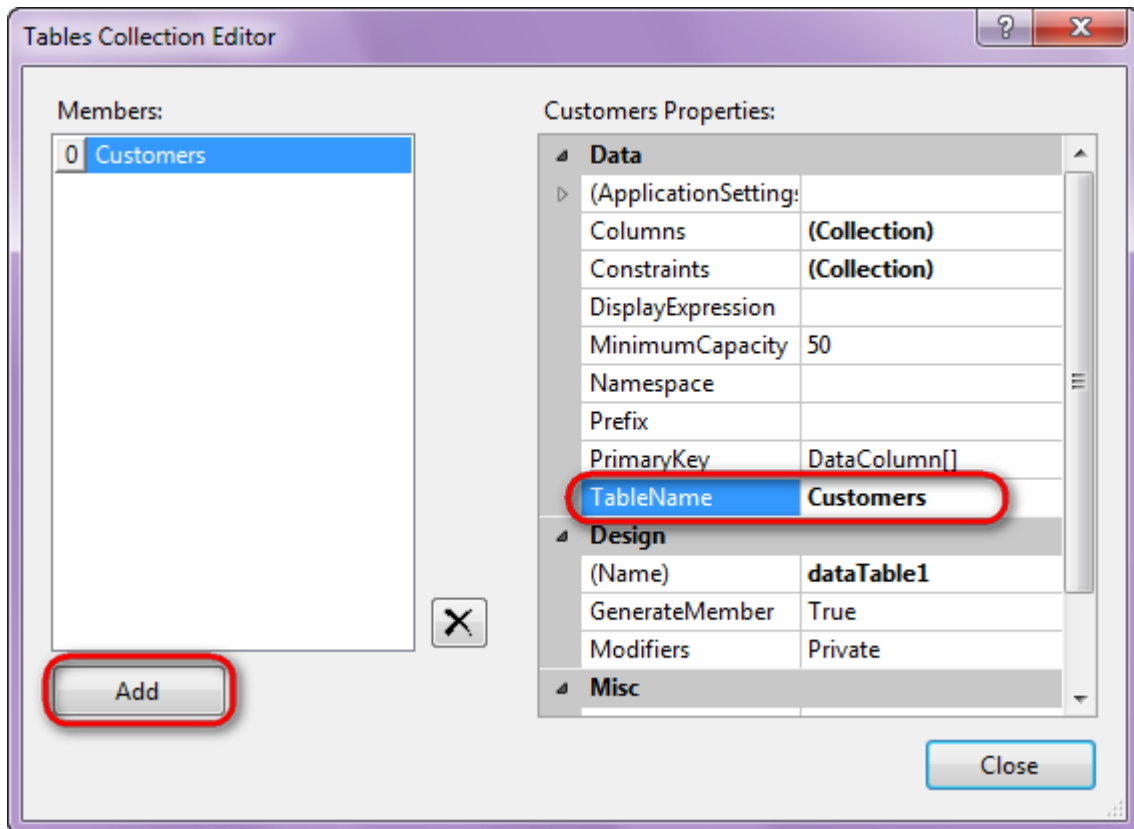


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

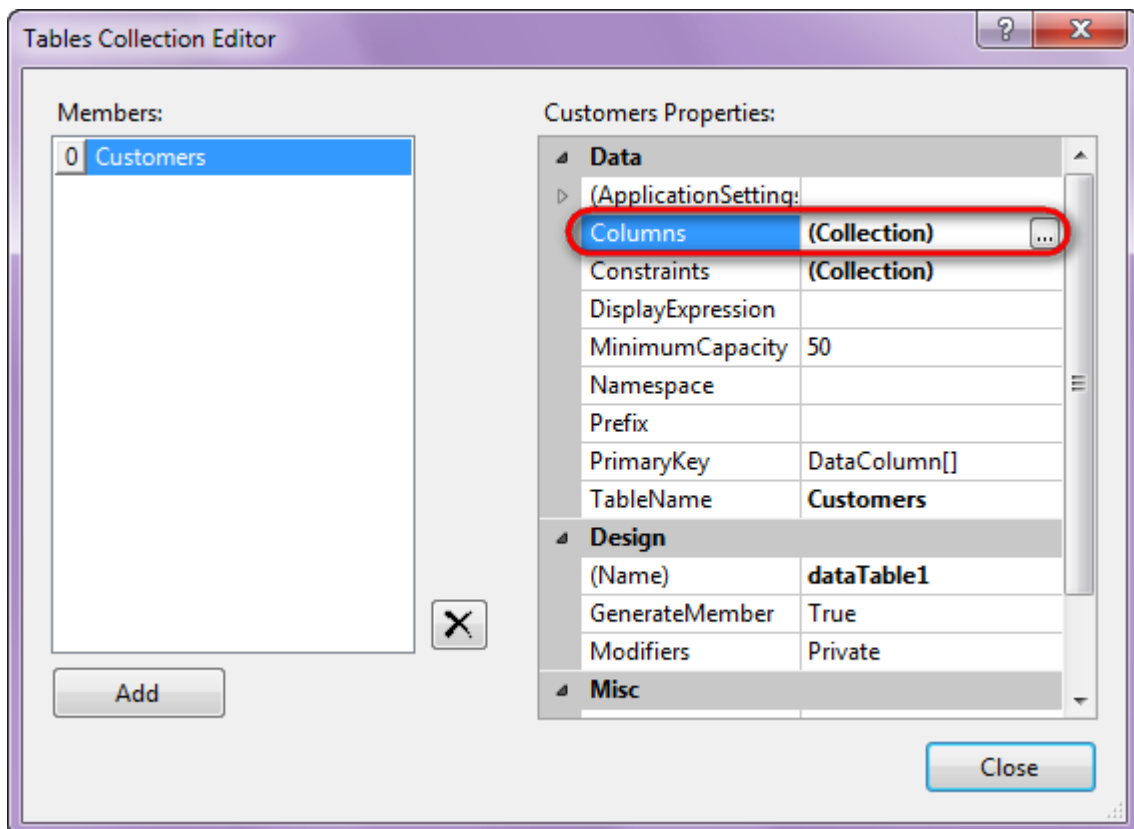


Click "Add" in order to add table. Set property TableName = Customers.

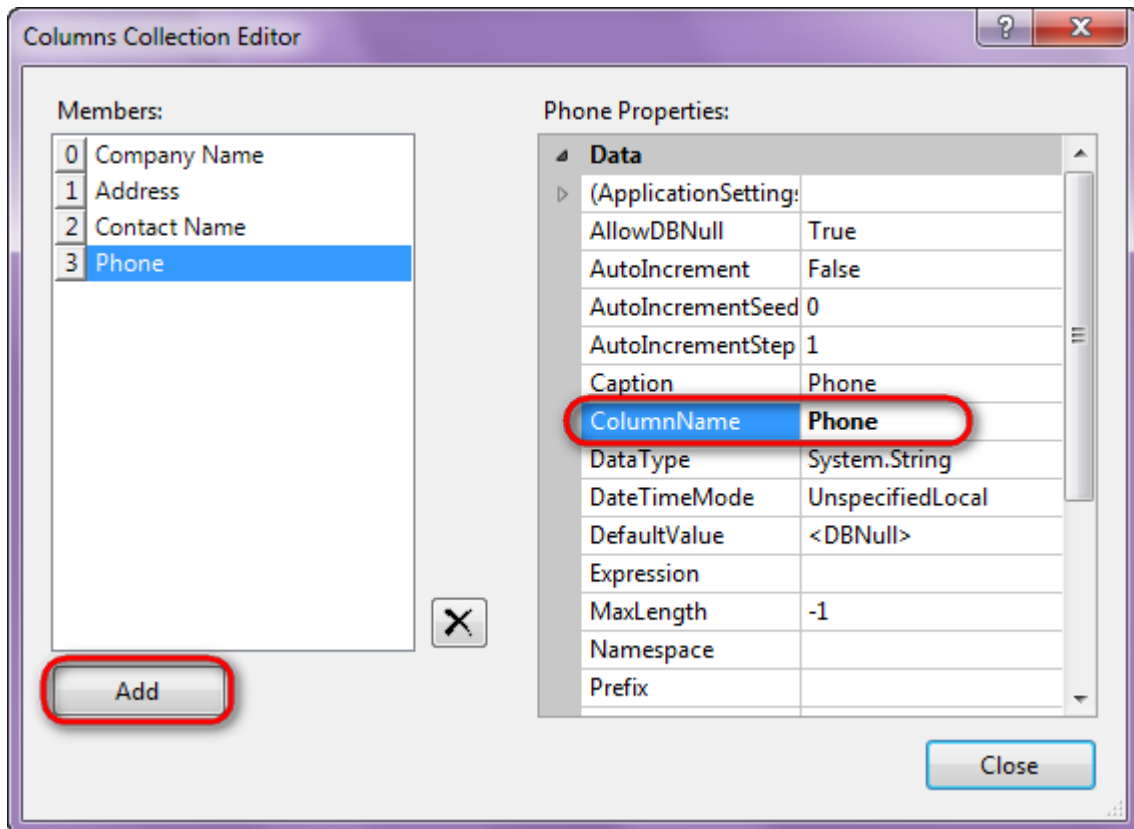


Step 5

Select Columns property, click button  in order to open property editor.



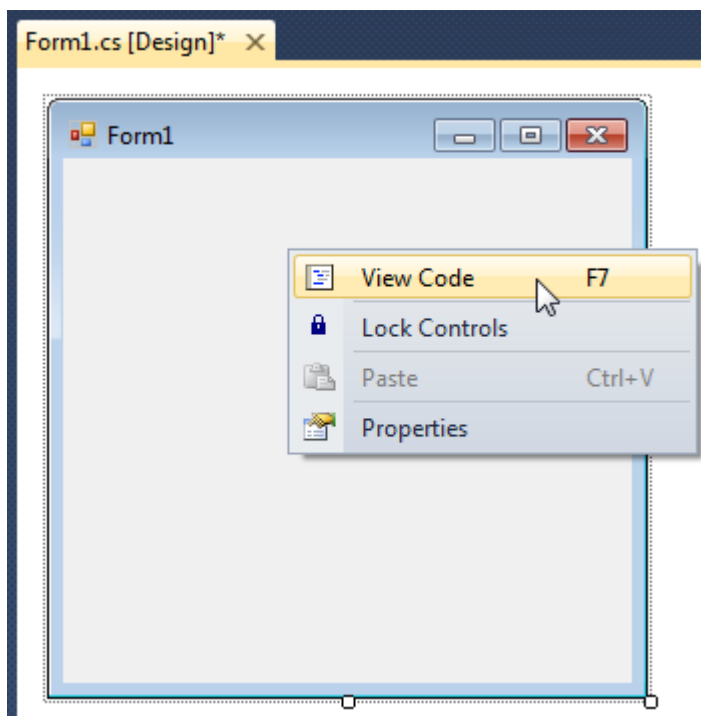
Click "Add" to add a new column. Add four columns. Set ColumnName property to "CompanyName", "Address", "ContactName", "Phone" correspondingly.



Close the editors.

Step 6

Right click on the form and select "View Code" in the context menu to view code.



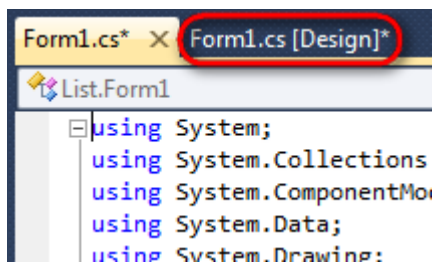
Add the following code to the class constructor in order to fill data source.

```
public Form1 ()  
{  
    InitializeComponent ();
```

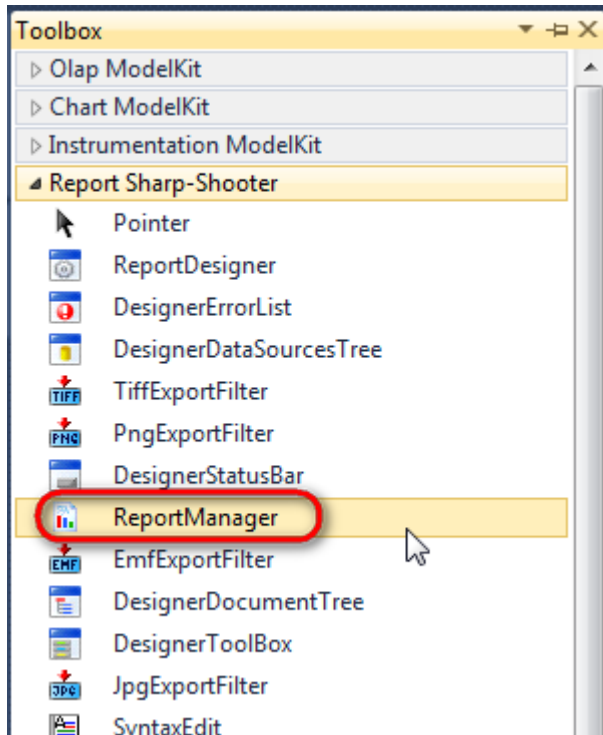
```
DataRow row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["Address"] = "Obere Str. 57";
row["ContactName"] = "Maria Anders";
row["Phone"] = "030-0074321";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Ana Trujillo Emparedados y helados";
row["Address"] = "Avda. de la Constitución 2222";
row["ContactName"] = "Ana Trujillo";
row["Phone"] = "(5) 555-4729";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Ernst Handel";
row["Address"] = "Kirchgasse 6";
row["ContactName"] = "Roland Mendel";
row["Phone"] = "7675-3425";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Toms Spezialitäten";
row["Address"] = "Luisenstr. 48";
row["ContactName"] = "Karin Josephs";
row["Phone"] = "0251-031259";
dataTable1.Rows.Add(row);
}
```

Step 7

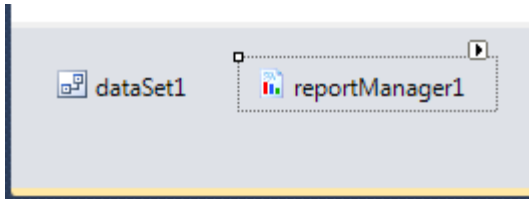
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

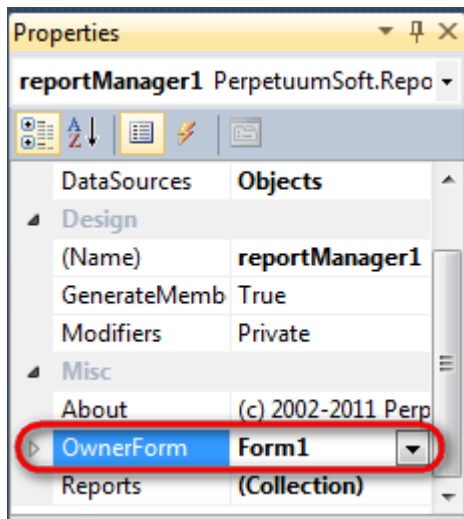


The component is available in the lower part of the window.



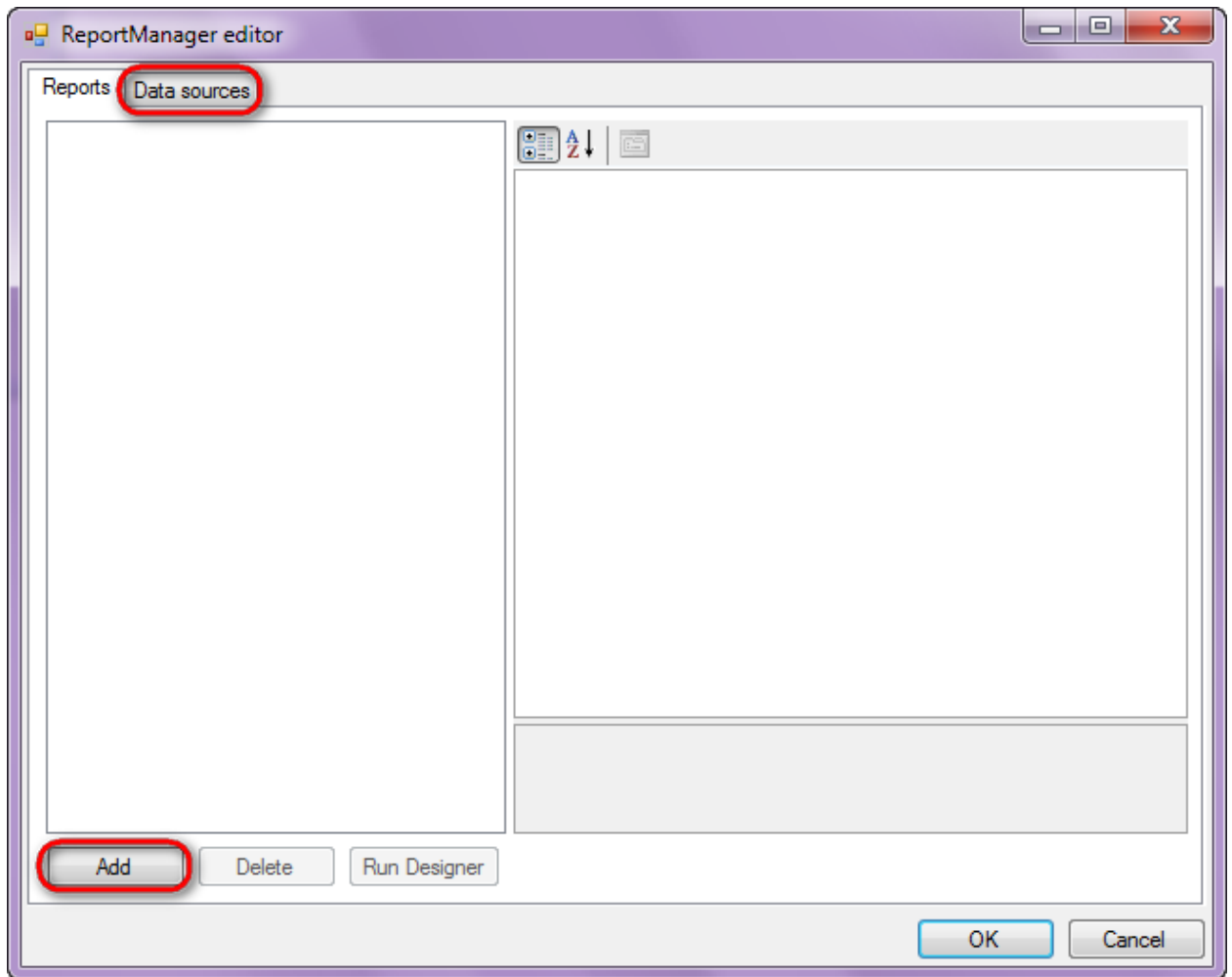
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

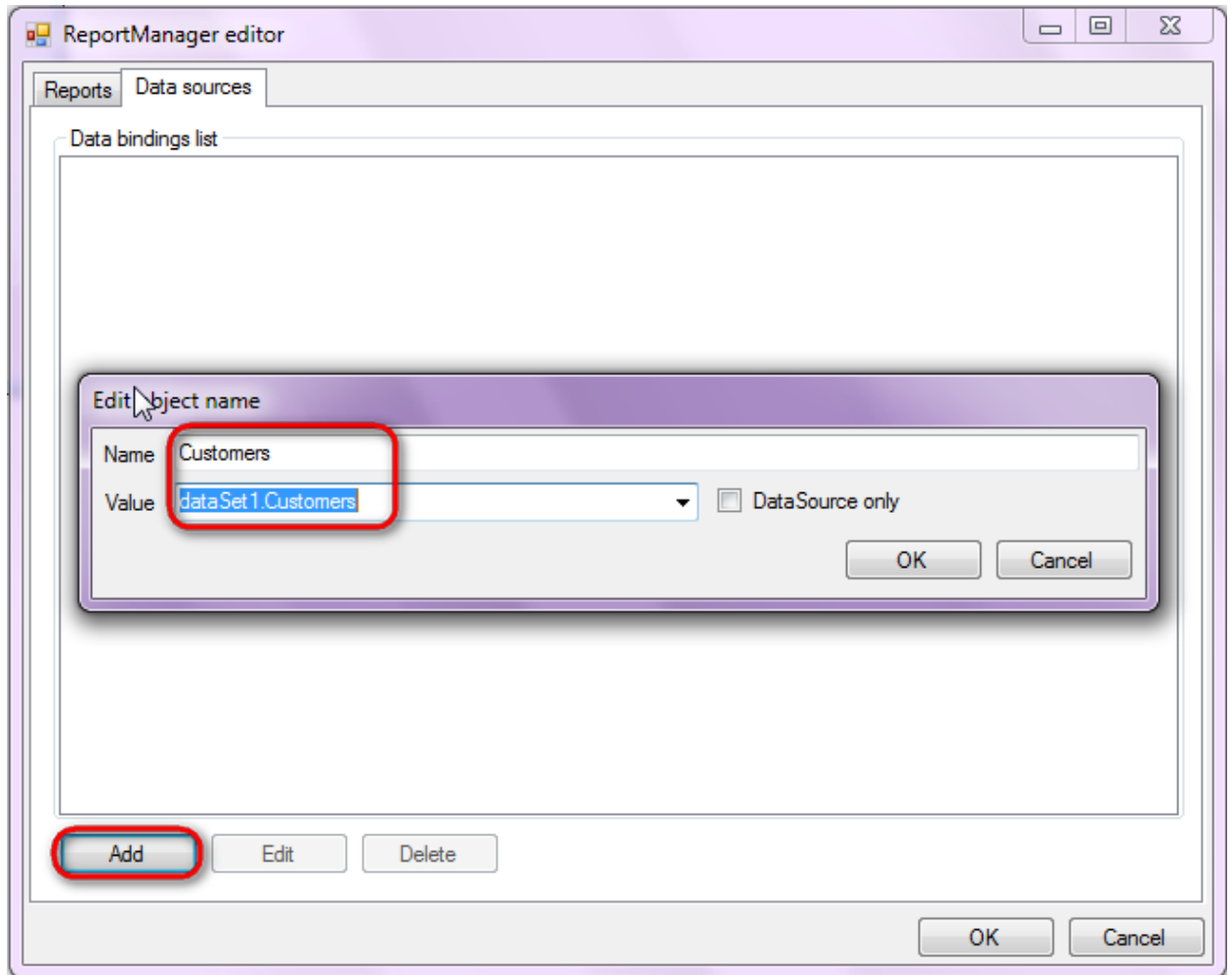


Step 9

Double click on ReportManager to open ReportManager editor.

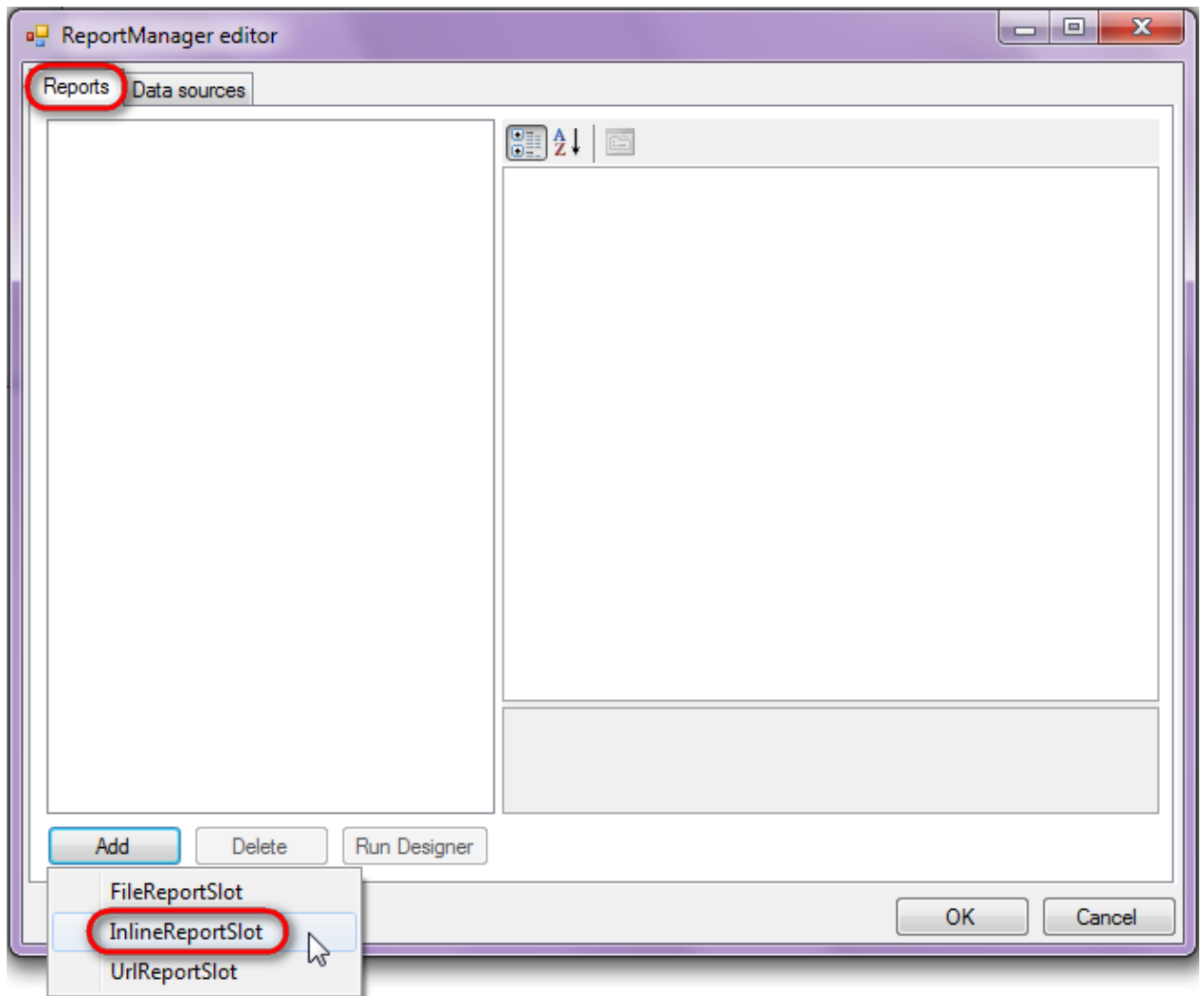


Go to "Data sources" tab, click "Add", set data source name - "Customers", select data source value - "dataSet1.Customers".



Step 10

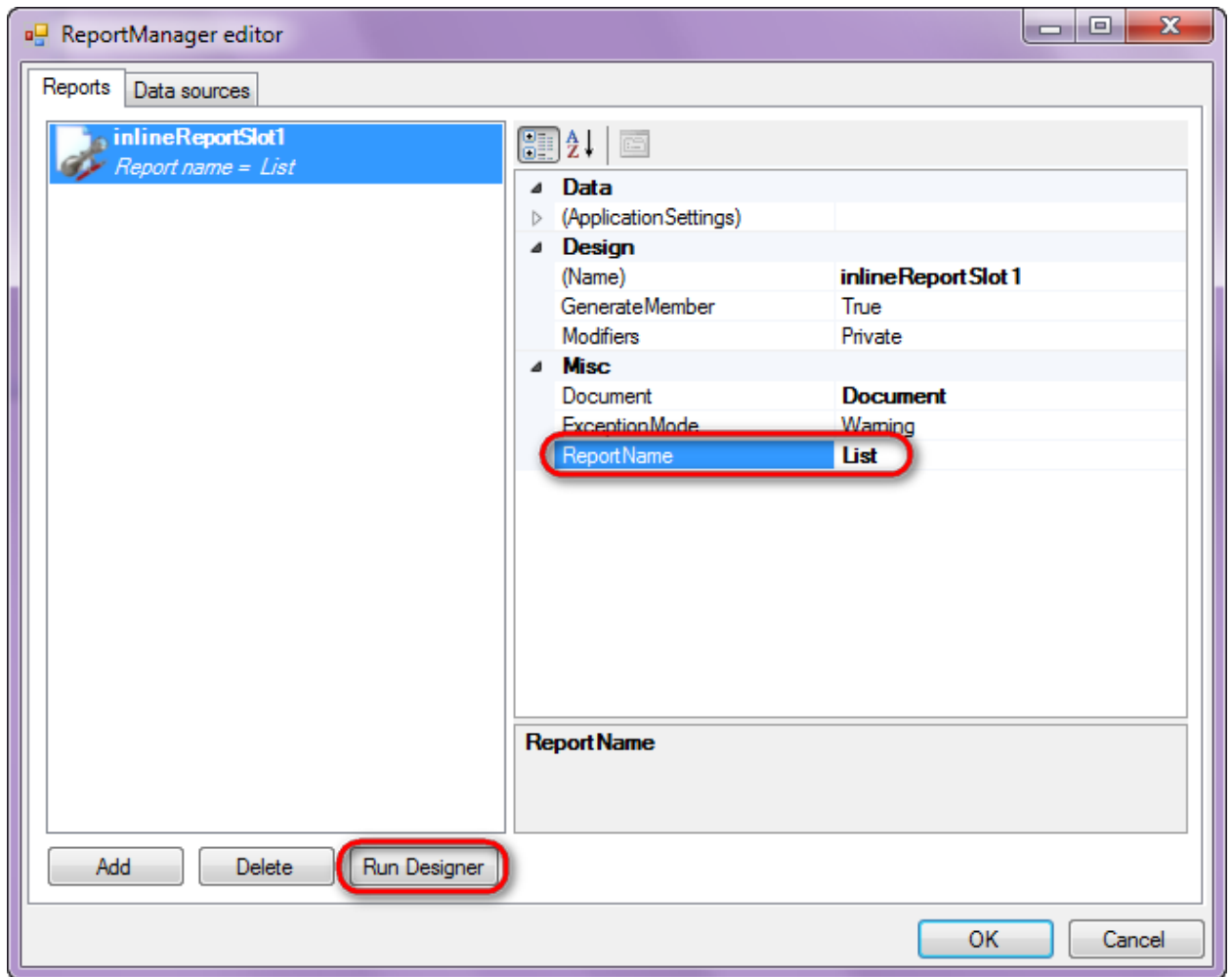
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

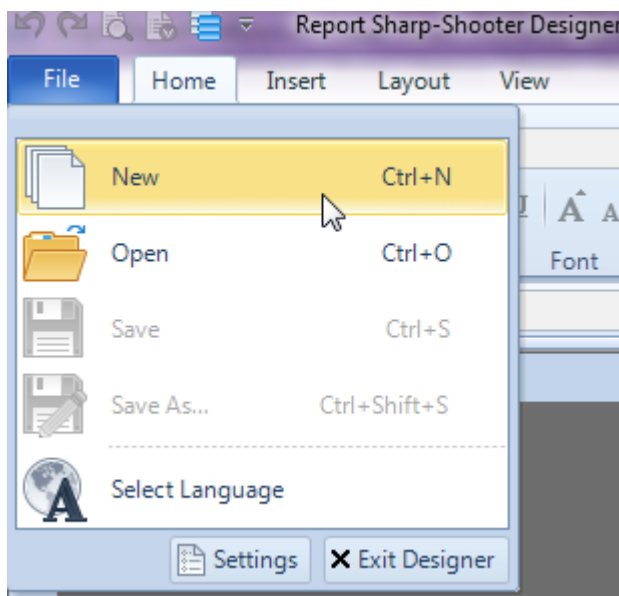
Set name of the report in the property ReportName – “List”.

Click “Run Designer” in order to open template editor – Report Designer.

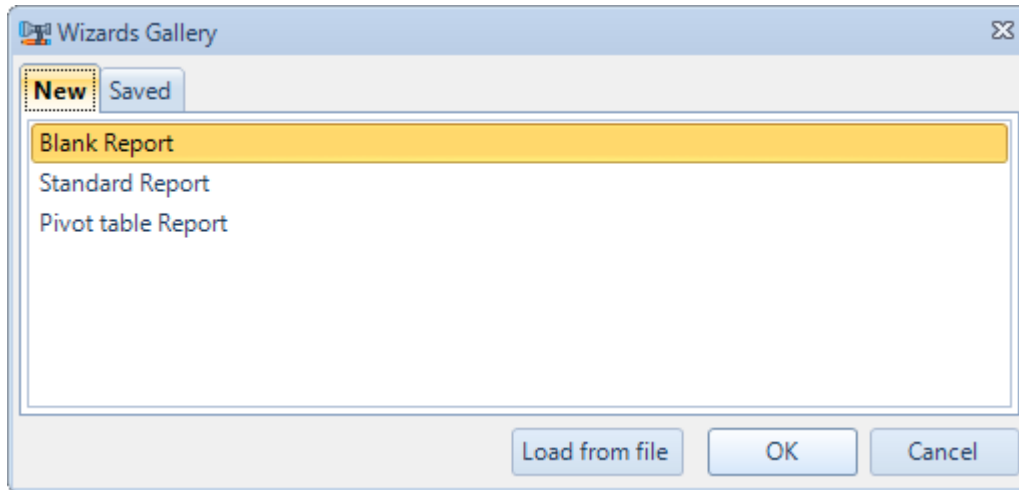


Step 12

Create new empty template – select File\New from the main menu.

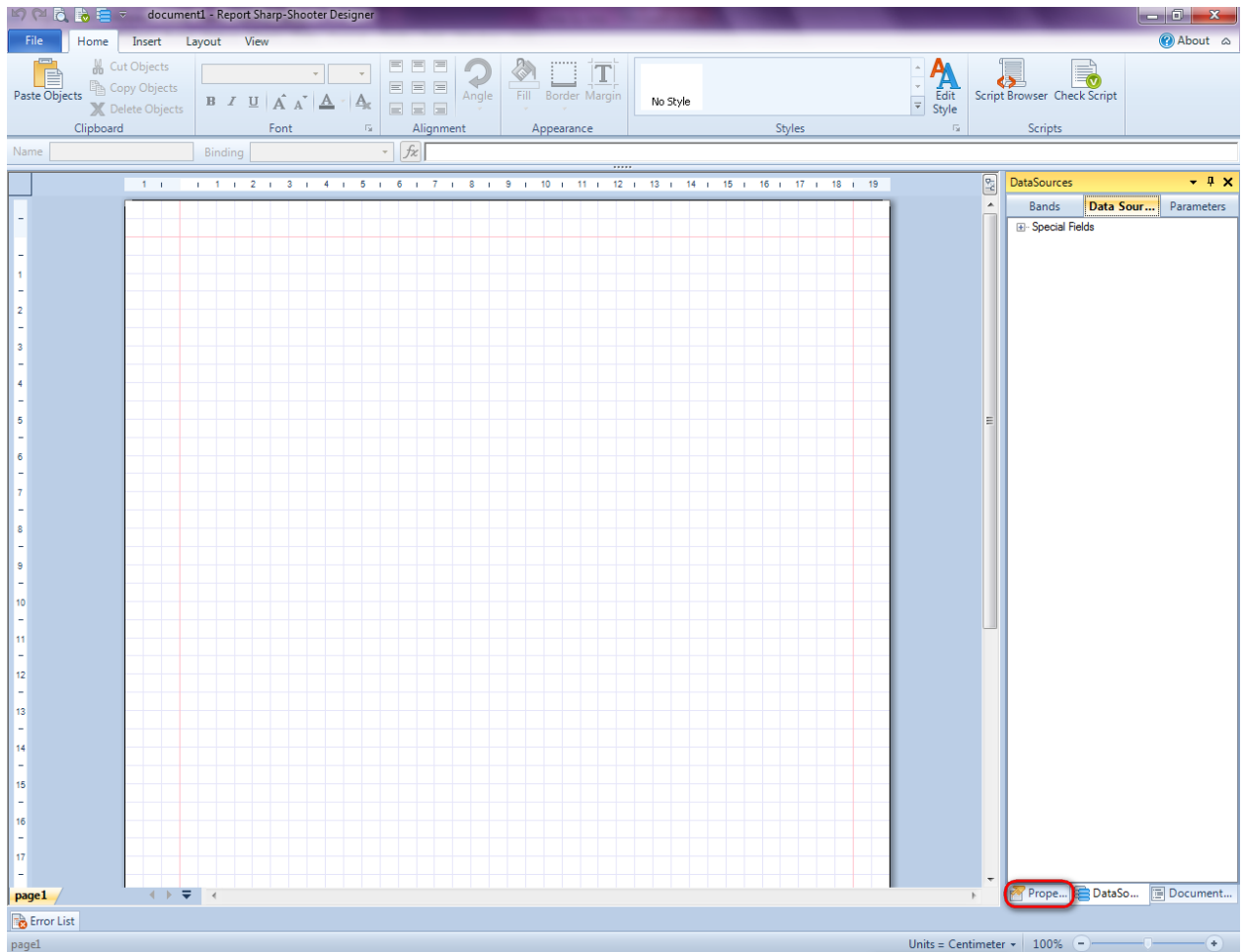


Select "Blank Report" in the Wizards Gallery and click "OK".

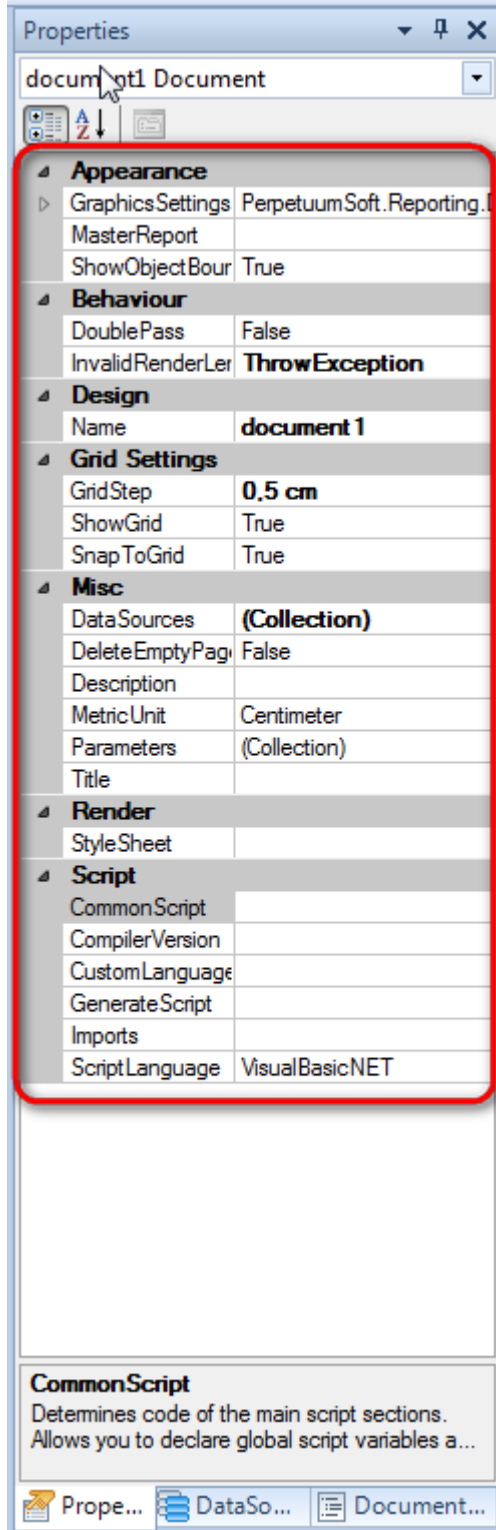


Step 13

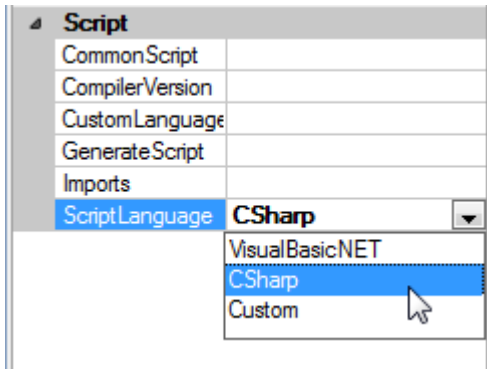
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

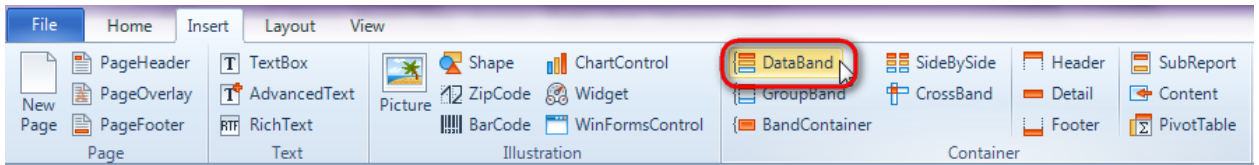


Set property ScriptLanguage = CSharp.



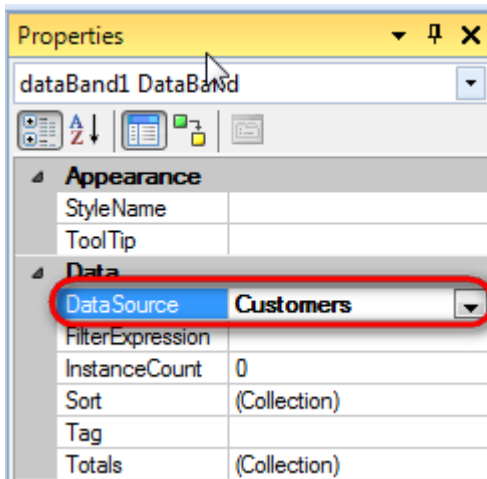
Step 14

Press "DataBand" button on the Insert tab in the group Container.



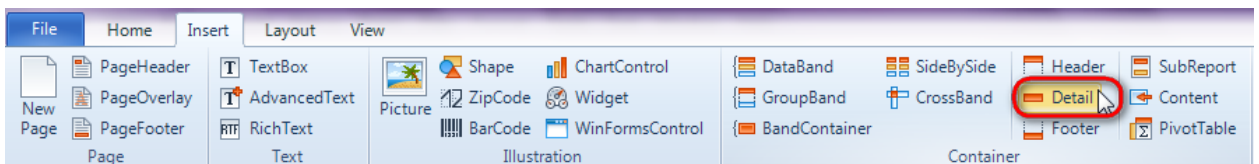
Click on the template area to add DataBand band.

Set data source in the property DataSource = Customers.

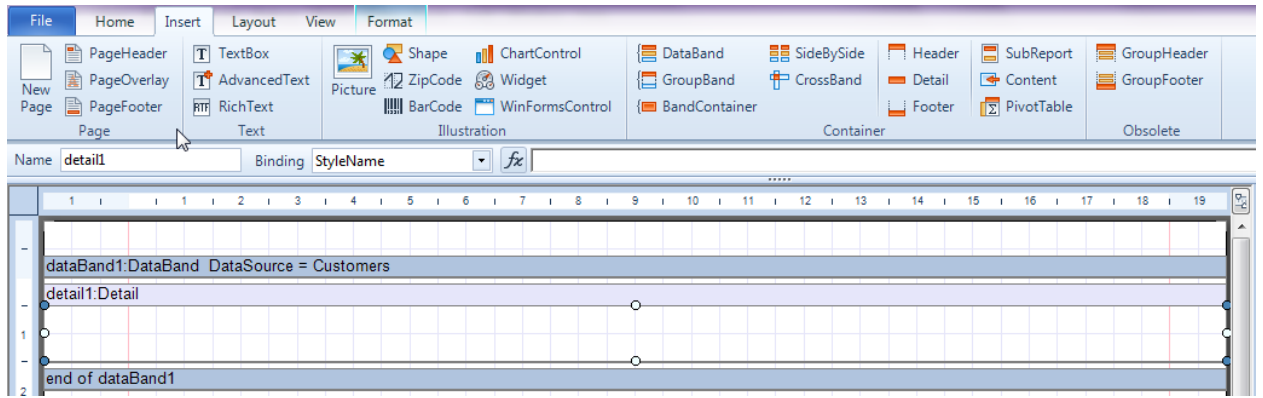


Step 15

Press "Detail" button on the Insert tab in the group Container.

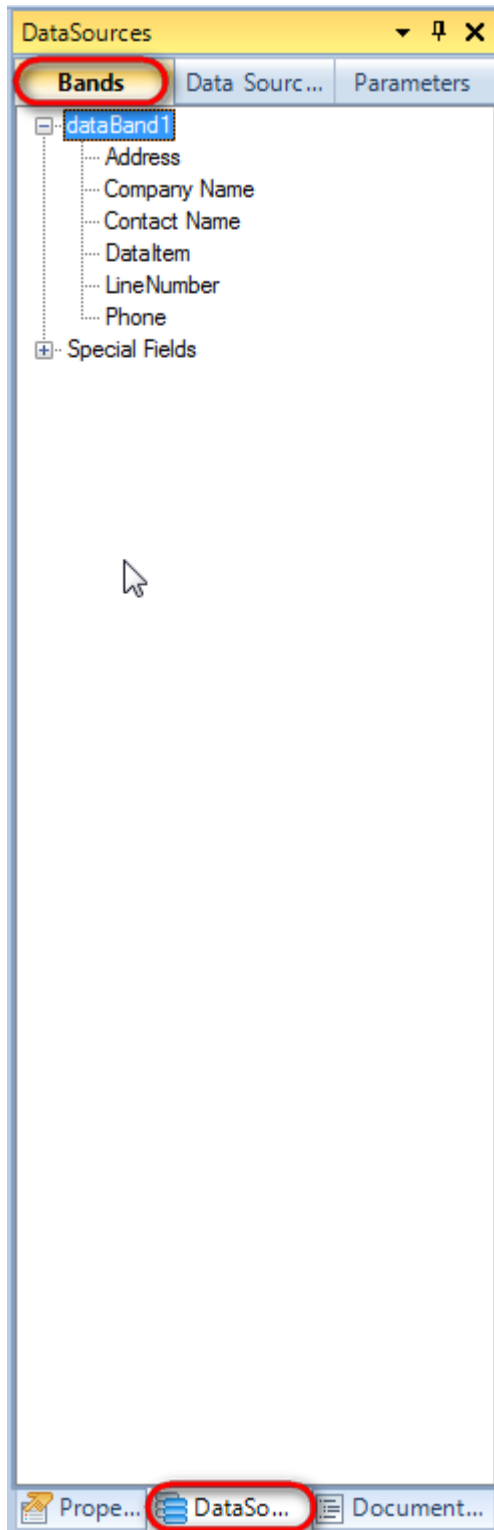


Click on the DataBand area to add the Detail band inside DataBand.



Step 16

Go to "Data Source" tab.



Drag and drop fields: "Company Name", "Address", "Contact Name", and "Phone" from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.

Change size of the elements and locate them in the way shown in the picture below.



dataBand1:DataBand DataSource = Customers				
detail1:Detail				
<dataBand1 ["CompanyName"]>	<dataBand1 ["ContactName"]>	<dataBand1 ["Address"]>	<dataBand1["Phone"]>	
end of dataBand1				

Step 17

Save template, close Report Designer.

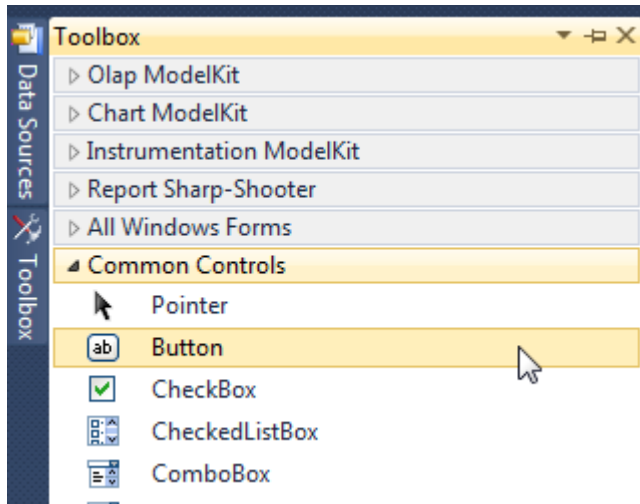
Step 18

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

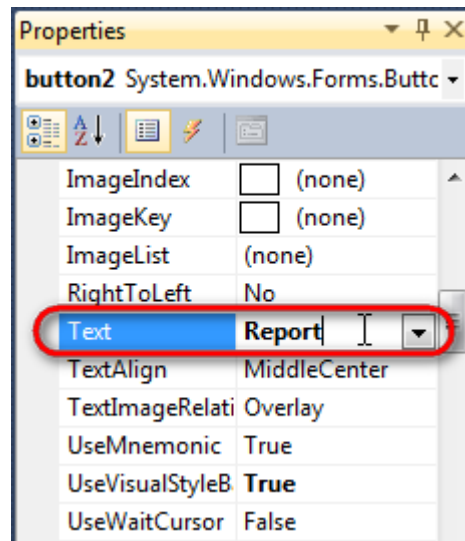
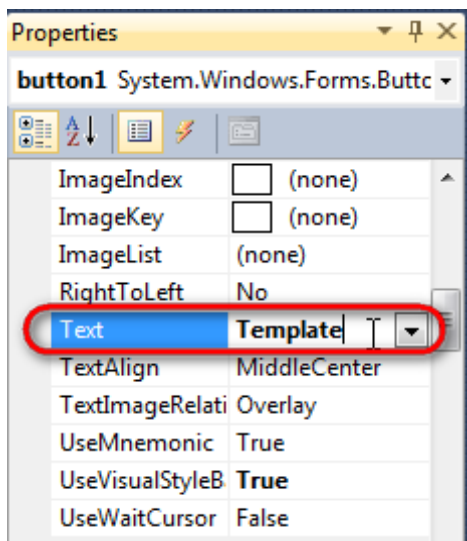
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["Address"] = "Obere Str. 57";
    row["ContactName"] = "Maria Anders";
    row["Phone"] = "030-0074321";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["Address"] = "Avda. de la Constitución 2222";
    row["ContactName"] = "Ana Trujillo";
    row["Phone"] = "(5) 555-4729";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["Address"] = "Kirchgasse 6";
    row["ContactName"] = "Roland Mendel";
    row["Phone"] = "7675-3425";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["Address"] = "Luisenstr. 48";
    row["ContactName"] = "Karin Josephs";
    row["Phone"] = "0251-031259";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
    EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
    PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 19

Place two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



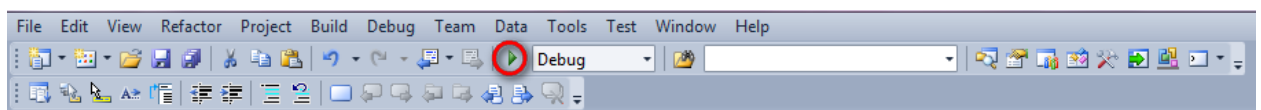
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

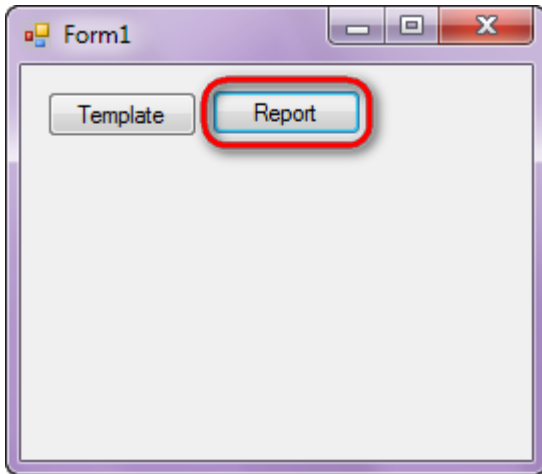
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 20

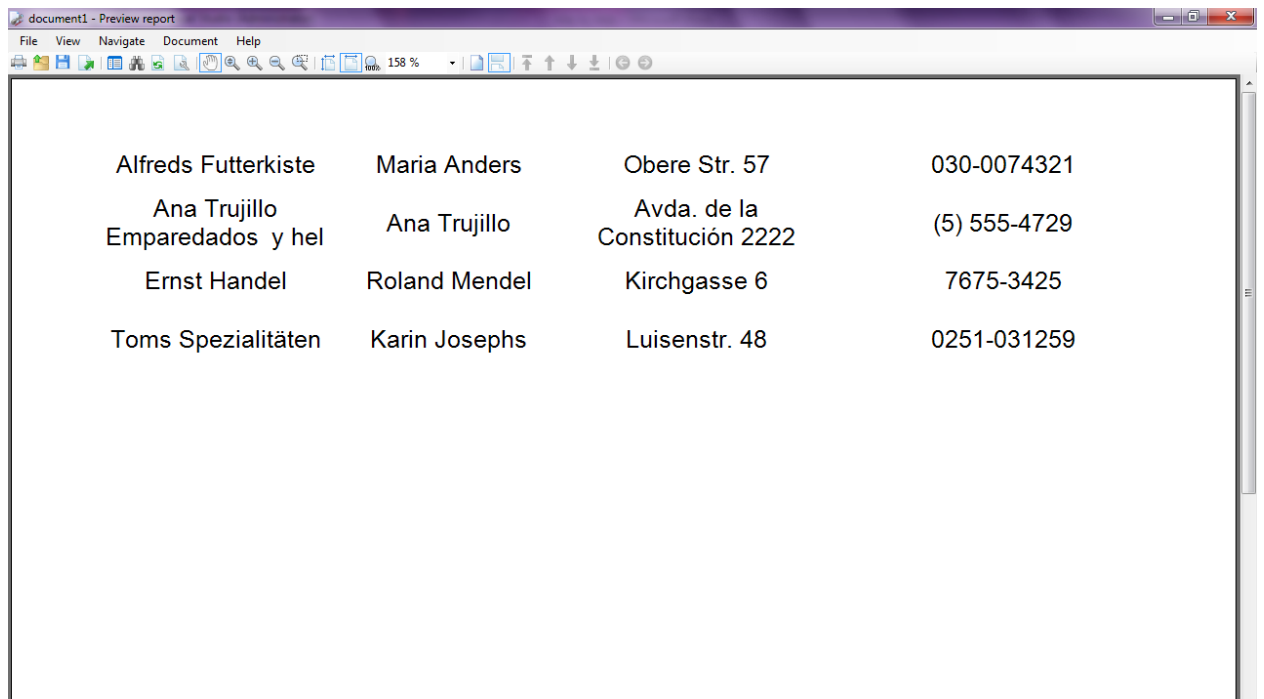
Click “Start Debugging” on the Visual Studio toolbar in order to run application.



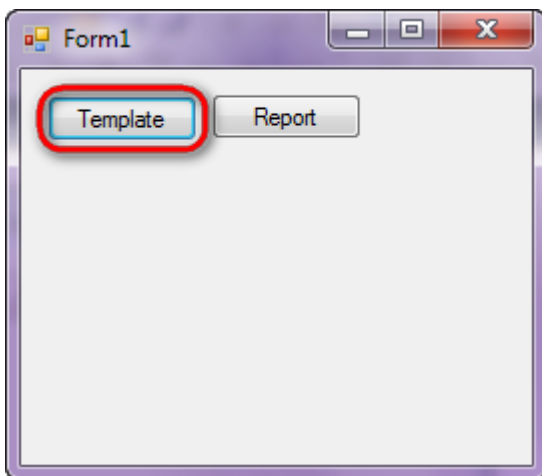
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar application sample is located in the following folder "Samples\Report Sharp-Shooter\CSharp\GettingStartedExample".

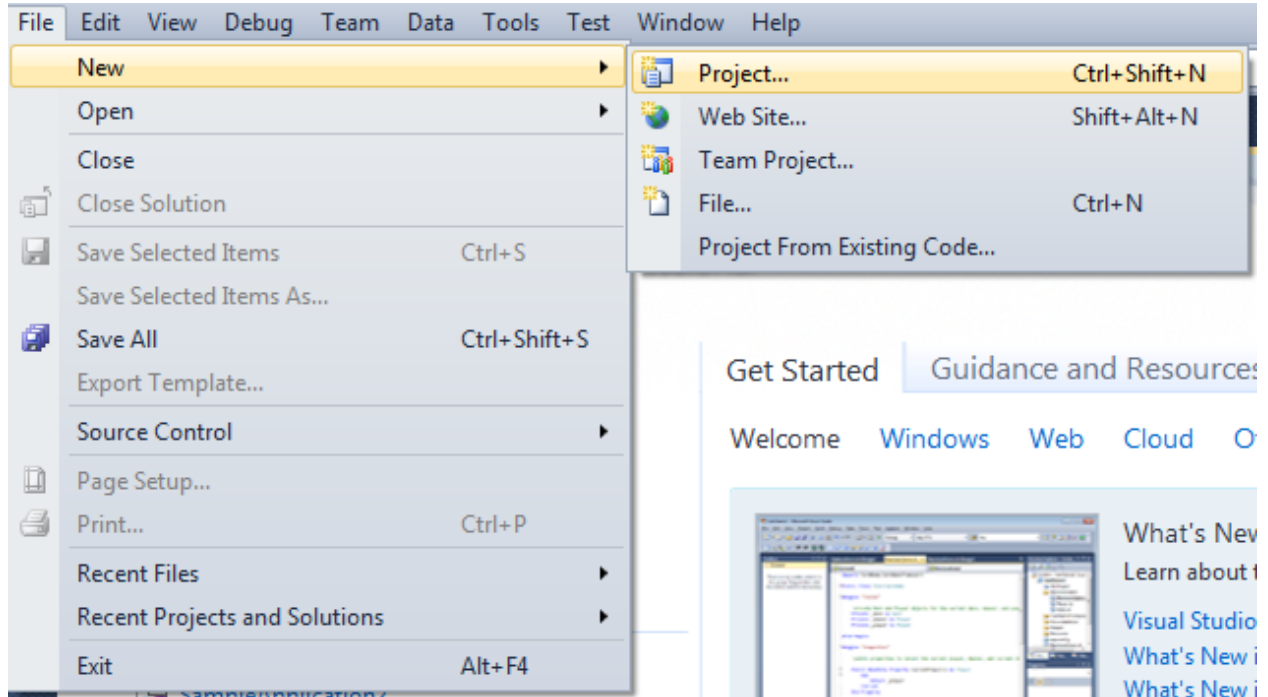


Header and Footer

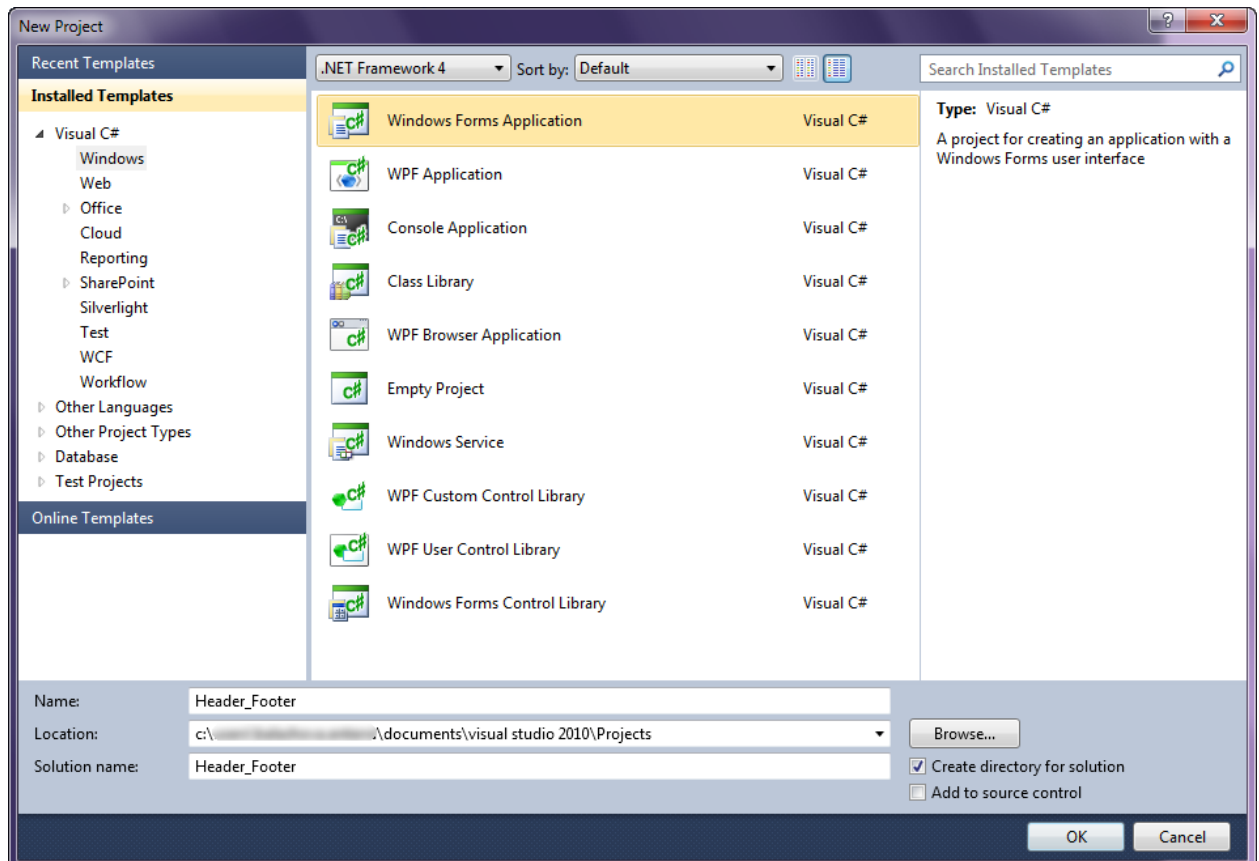
Template of a report containing list of companies with additional information, each column has header and footer.

Step 1

Create a new project in Microsoft Visual Studio. Select New\Project from the main menu.

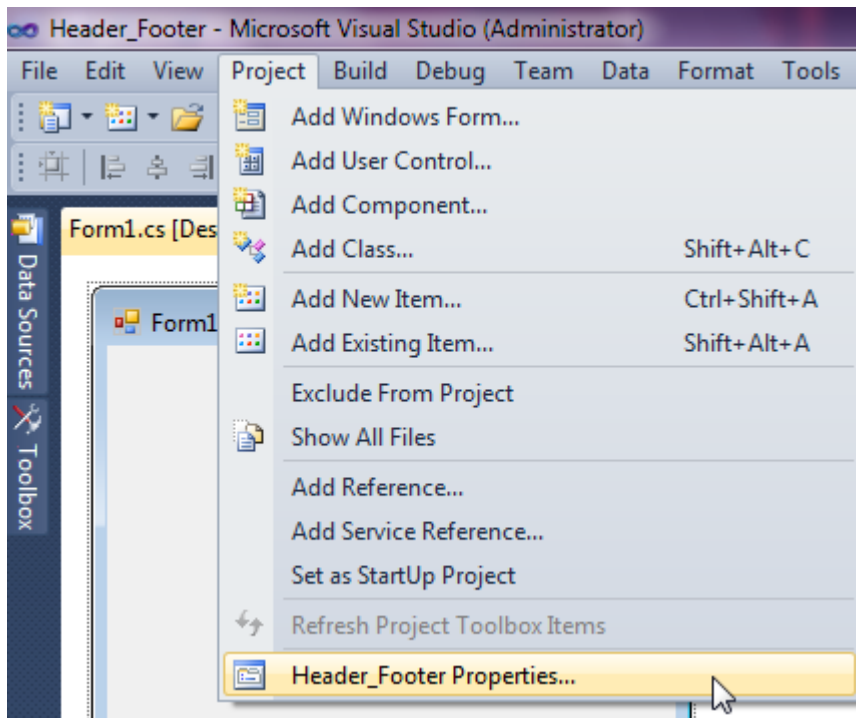


Select Windows Forms Application, set project name – “Header_Footer”, set directory to save project to.

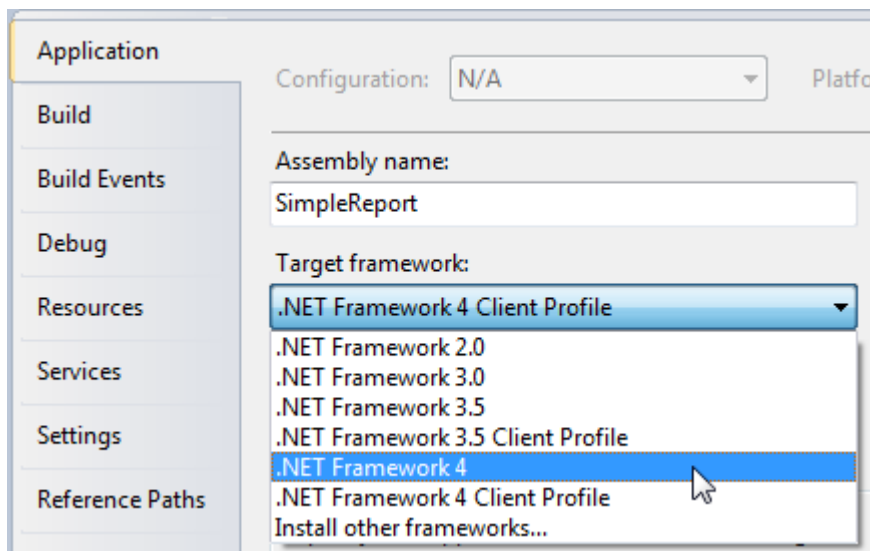


Step 2

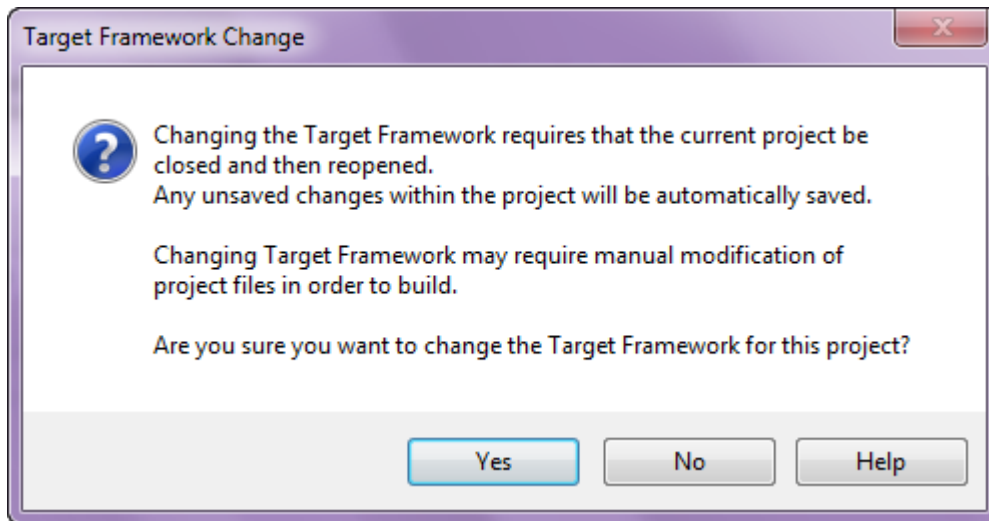
Change the project properties. Select the Project\Header_Footer Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

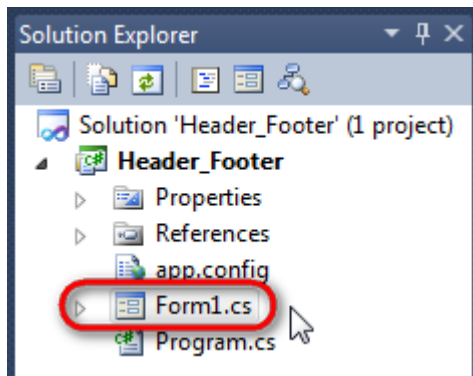


Press the "Yes" button in the opened window.

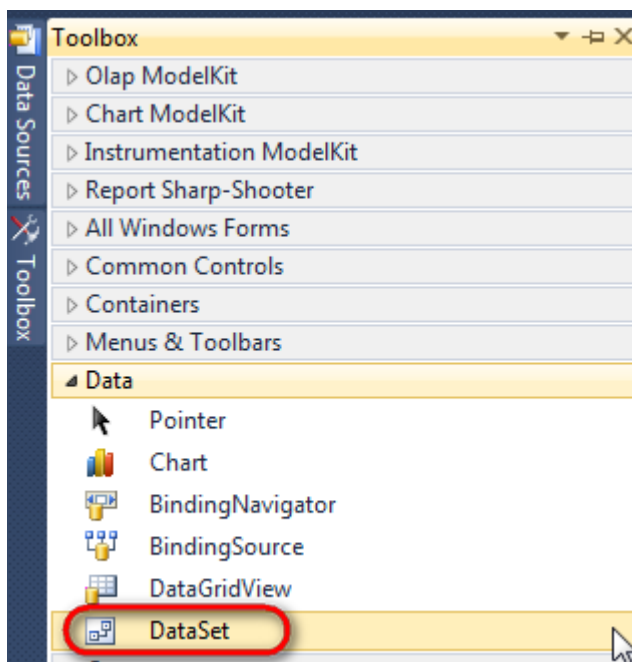


Step 3

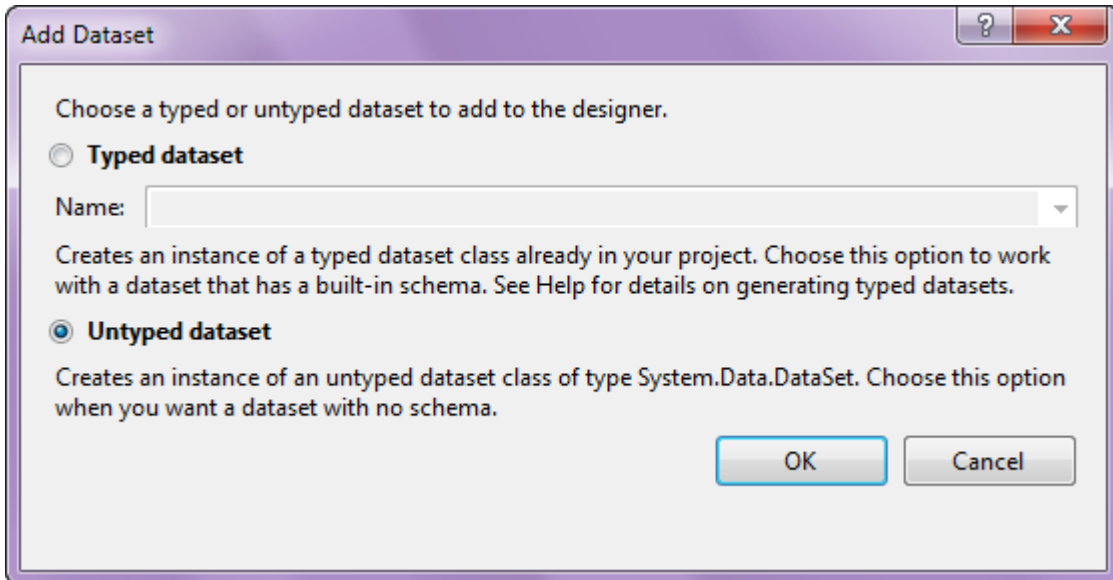
Open main form of the application in the editor by double click on the "Form1.cs" in the Solution Explorer.



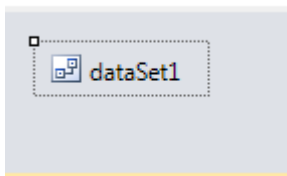
Click "DataSet" element on the Toolbox and place it onto the form.




Select "Untyped dataset", click "OK".

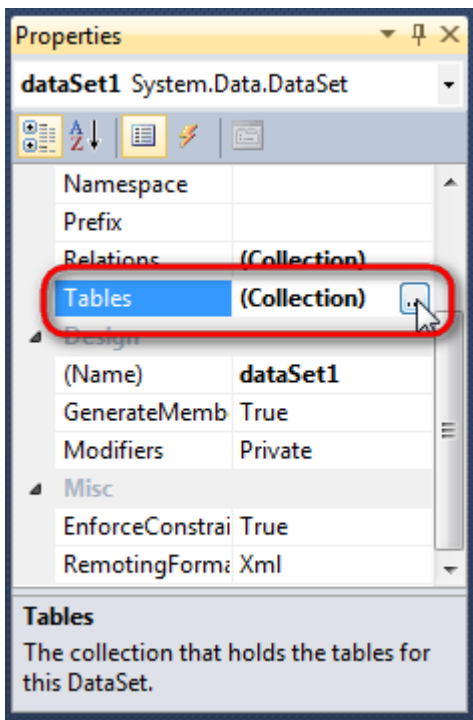


The component is available in the lower part of the window.

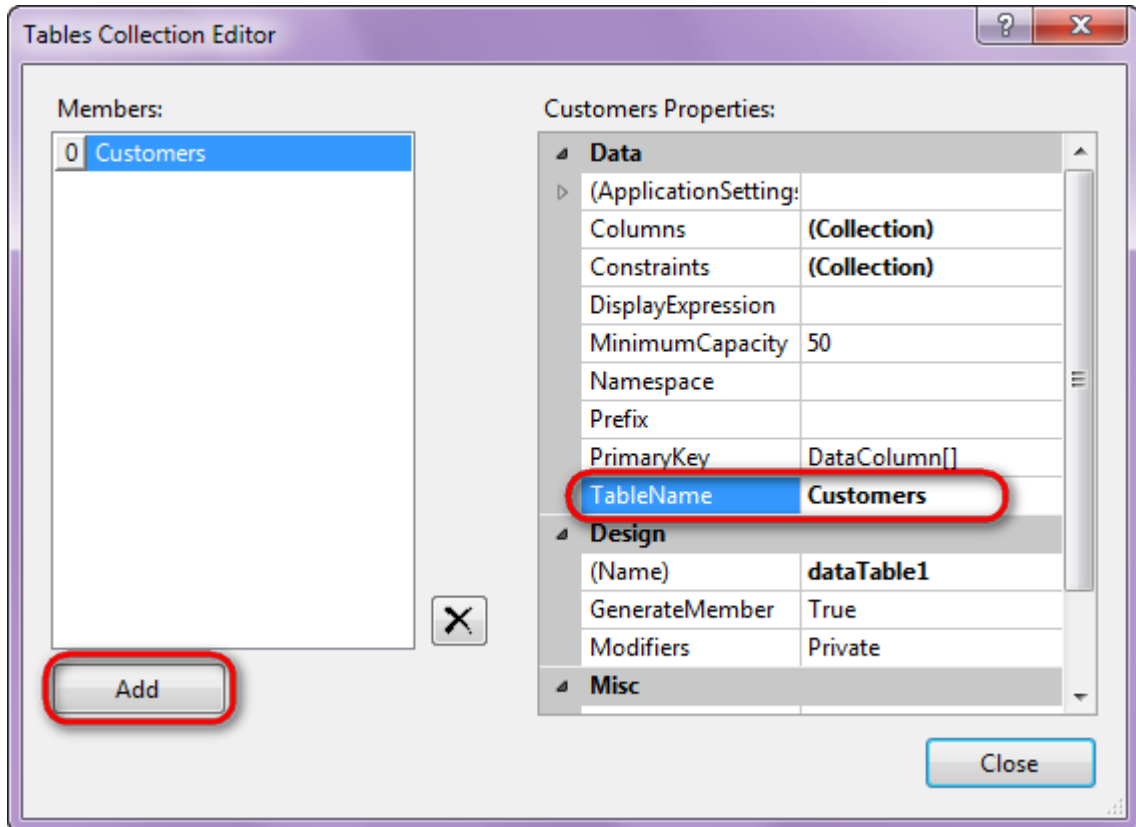


Step 4

Select dataSet1 component in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

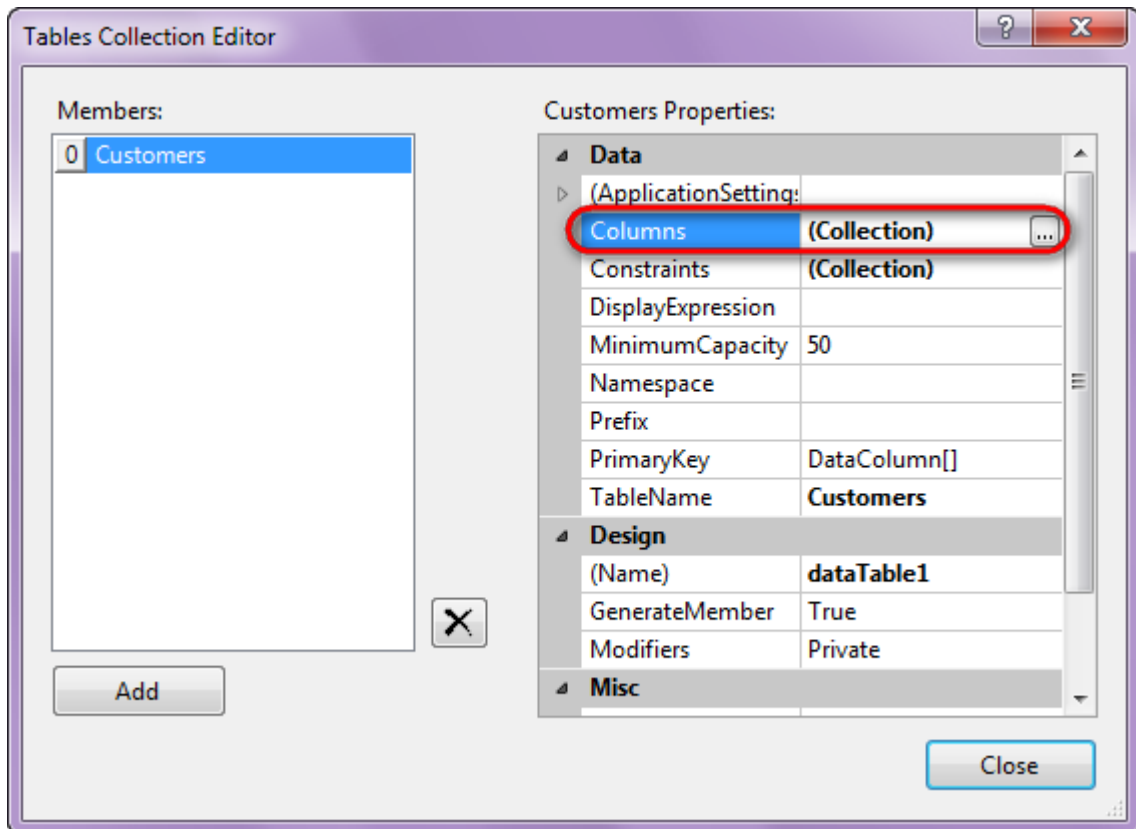


Click "Add" to add a table. Set property TableName = Customers.

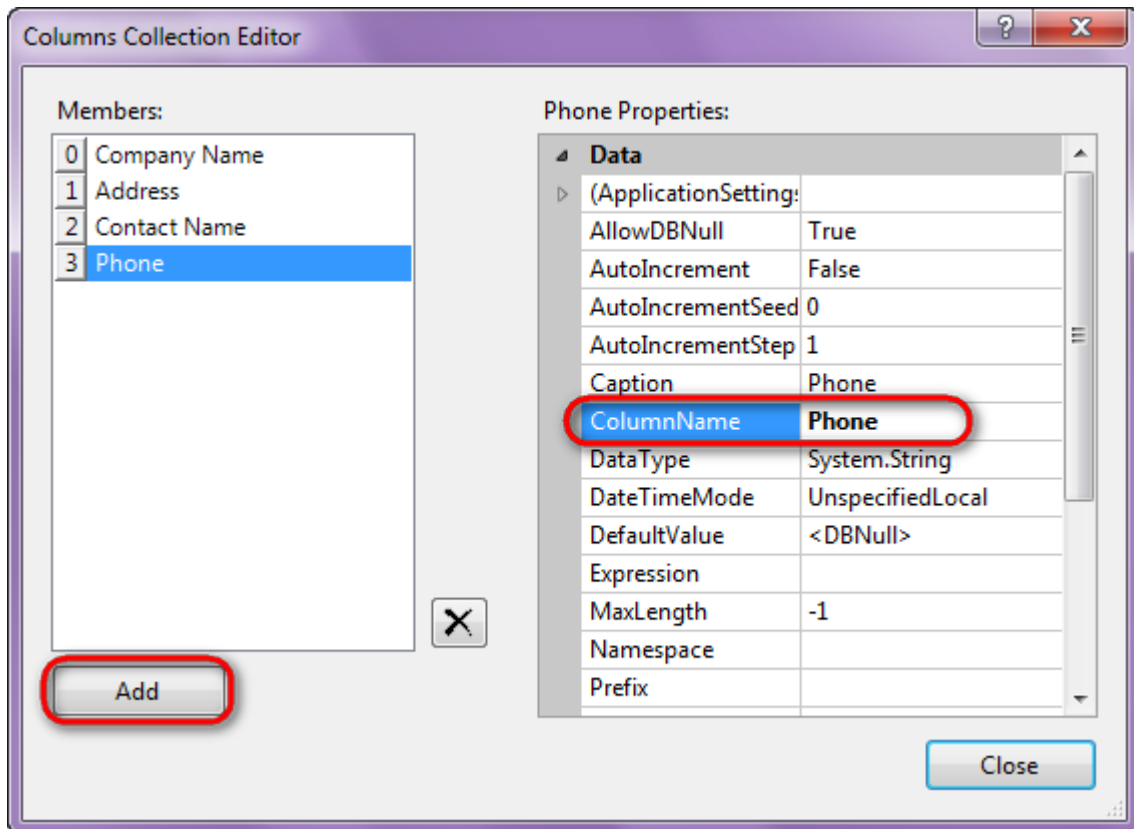


Step 5

Select Columns property, click button  in order to open property editor.



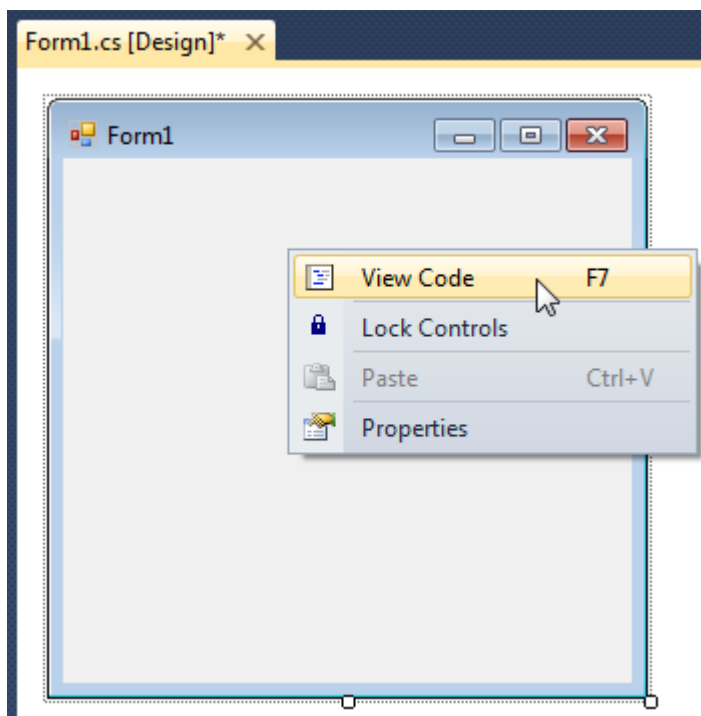
Click "Add" to add new column. Add four columns. Set ColumnName property to "CompanyName", "Address", "ContactName", "Phone".



Close the editors.

Step 6

Right click on the form and select "View Code" in the context menu to view code.



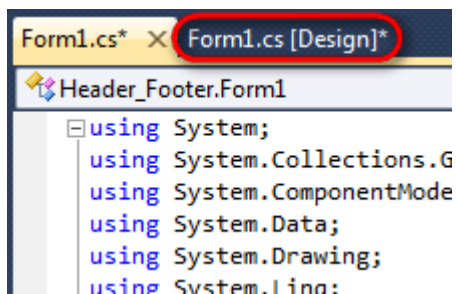
Add the following code to the class constructor to fill the data source.

```
public Form1 ()  
{  
    InitializeComponent ();
```

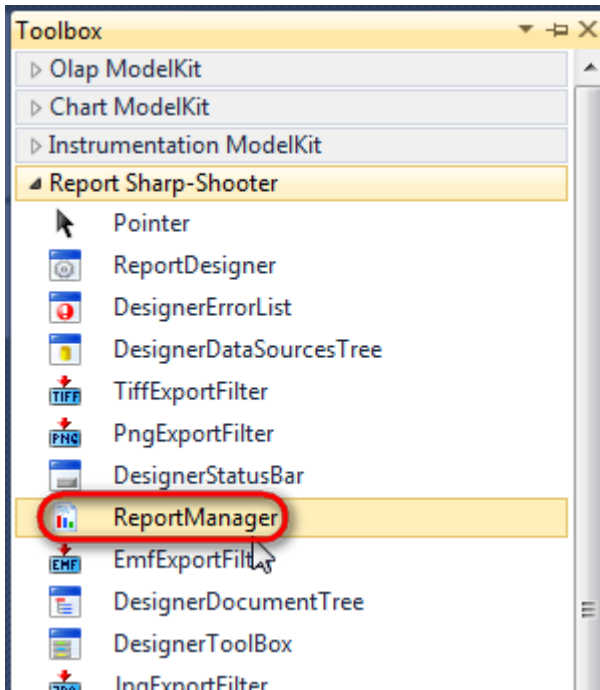
```
DataRow row = dataTable1.NewRow();  
row["CompanyName"] = "Alfreds Futterkiste";  
row["Address"] = "Obere Str. 57";  
row["ContactName"] = "Maria Anders";  
row["Phone"] = "030-0074321";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["Address"] = "Avda. de la Constitución 2222";  
row["ContactName"] = "Ana Trujillo";  
row["Phone"] = "(5) 555-4729";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["Address"] = "Kirchgasse 6";  
row["ContactName"] = "Roland Mendel";  
row["Phone"] = "7675-3425";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["Address"] = "Luisenstr. 48";  
row["ContactName"] = "Karin Josephs";  
row["Phone"] = "0251-031259";  
dataTable1.Rows.Add(row);  
}
```

Step 7

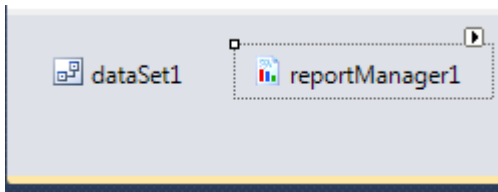
Get back to the application form by clicking "Form1.cs[Design]" tab.



Click "ReportManager" element on the Toolbox and place the component onto the form. This component is designed to store collections of data sources and report templates.

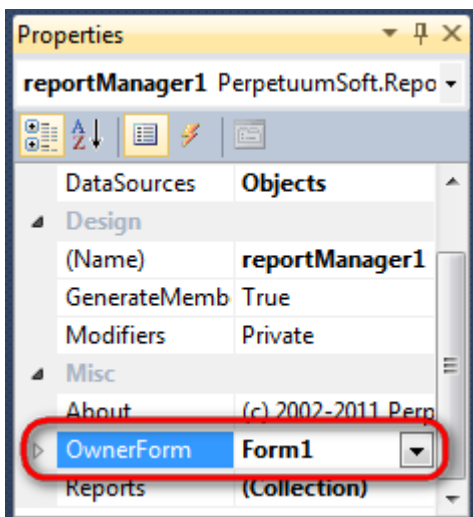


The component is available in the lower part of the window.



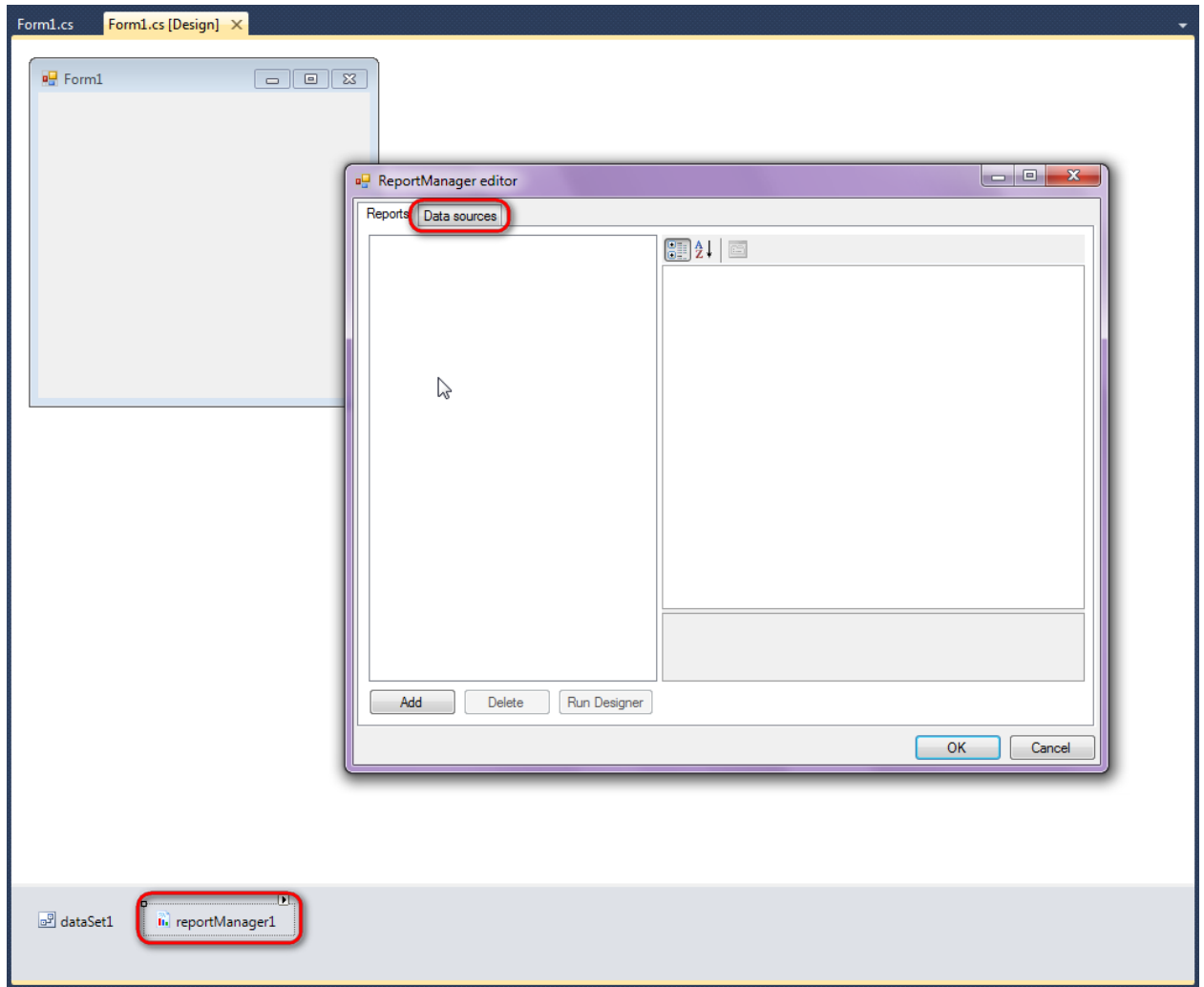
Step 8

On the property grid, initialize the OwnerForm property of the ReportManager element by selecting the form it is located on.

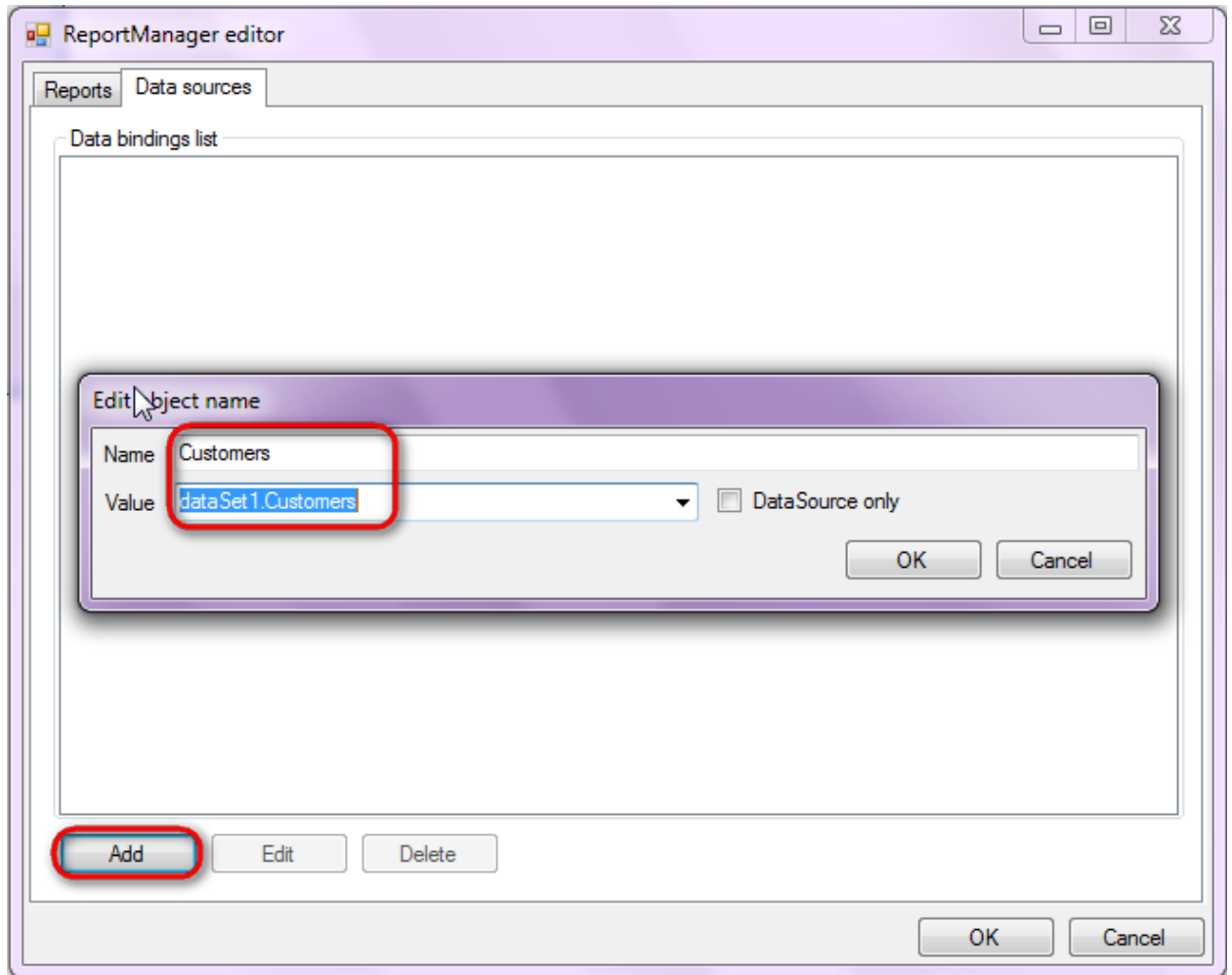


Step 9

Double click on the ReportManager and open ReportManager editor.

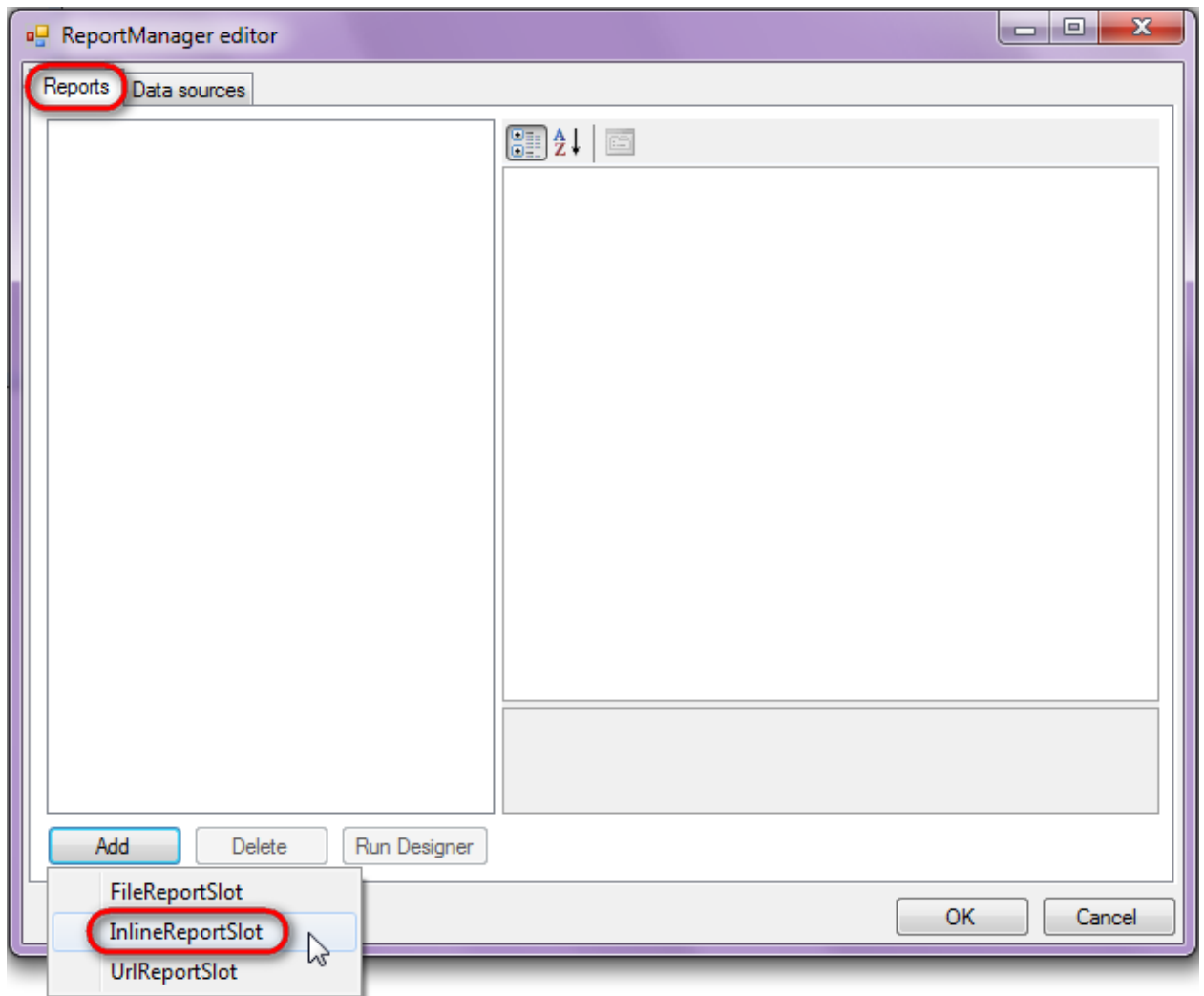


Go to the "Data sources" tab, click "Add", set name of the data source to "Customers", select data source value to "dataSet1.Customers".



Step 10

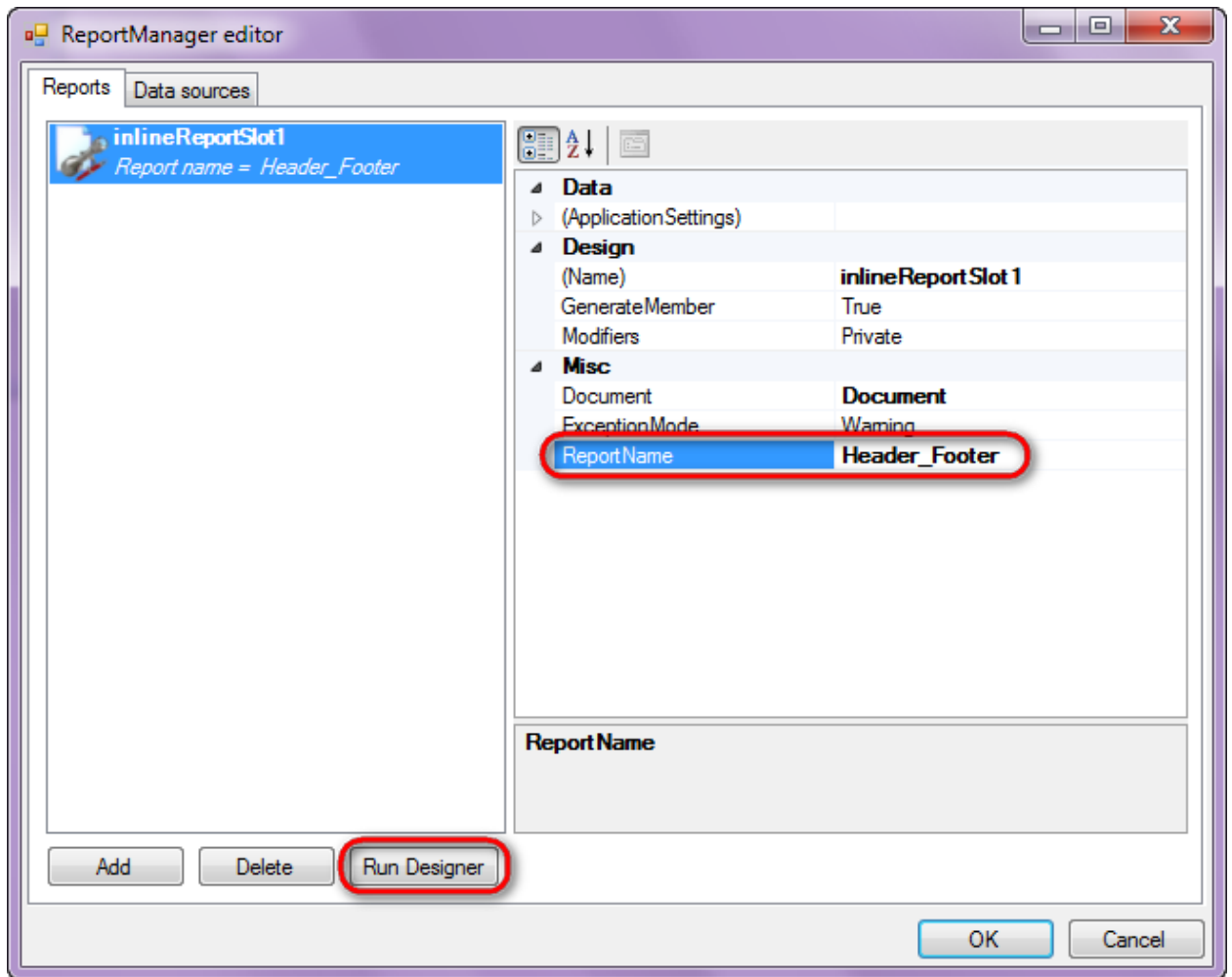
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

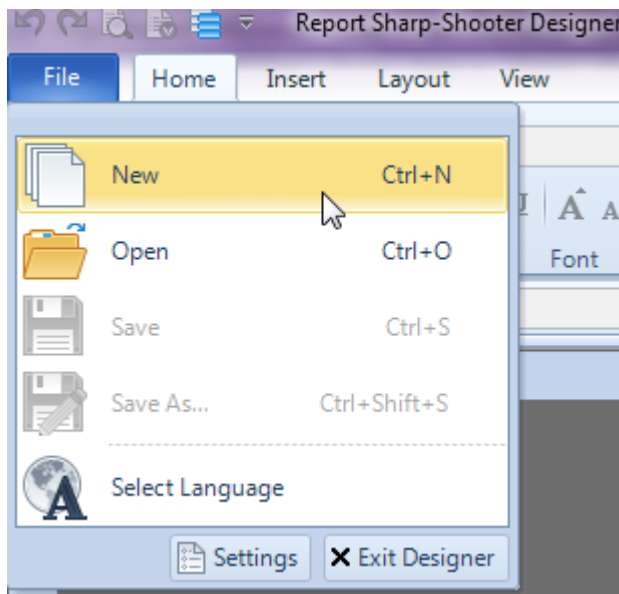
Set name of the report in the property ReportName – “Header_Footer”.

Click “Run Designer” to open template editor Report Designer.

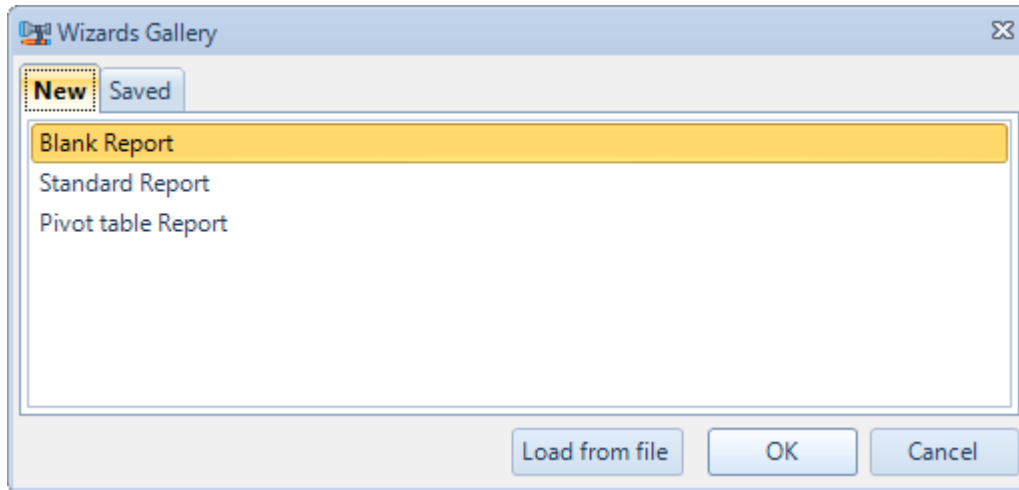


Step 12

Create new empty template – select File\New from the main menu.

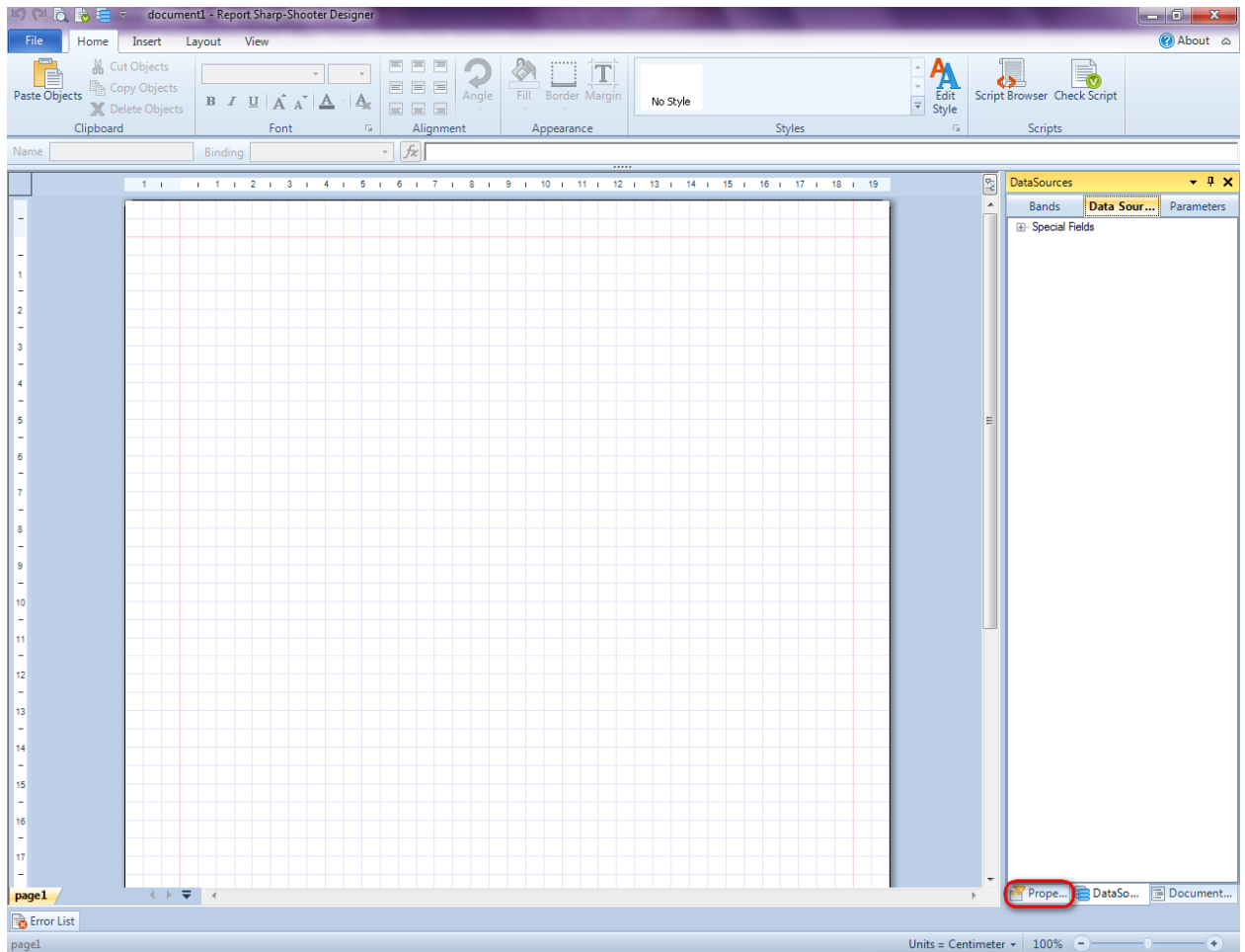


Select "Blank Report" in the Wizards Gallery and click "OK".



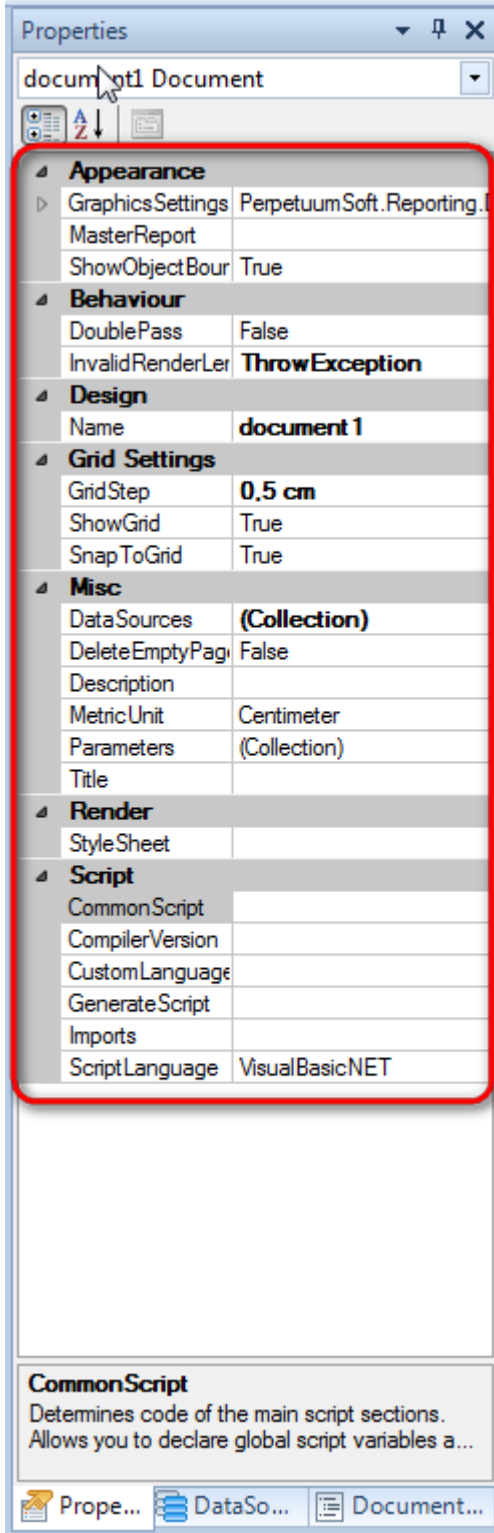
Step 13

Click the "Properties" tab of the tool window in the right part of the designer.

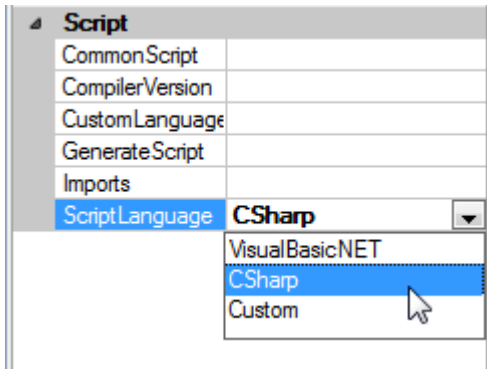




You will see properties of the edited template on the "Properties" tab

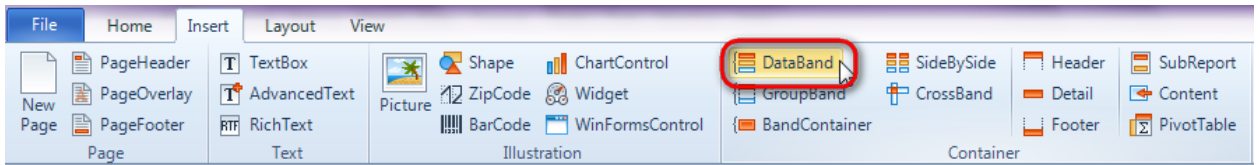


Set property ScriptLanguage = CSharp.



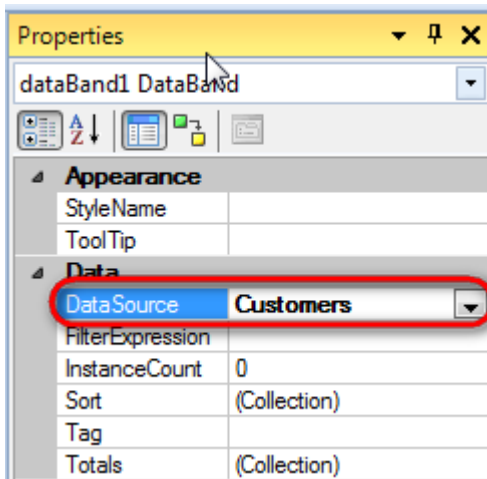
Step 14

Press "DataBand" button on the Insert tab in the group Container.



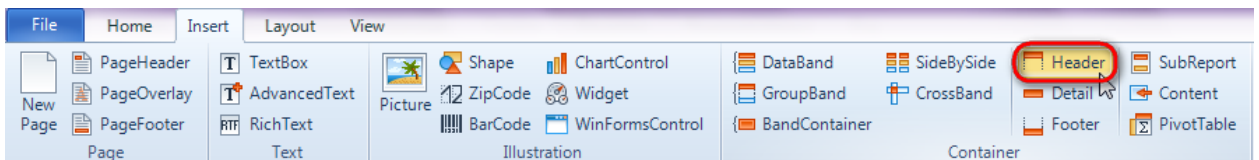
Click on the template area to add DataBand band.

Set data source in the property DataSource = Customers.



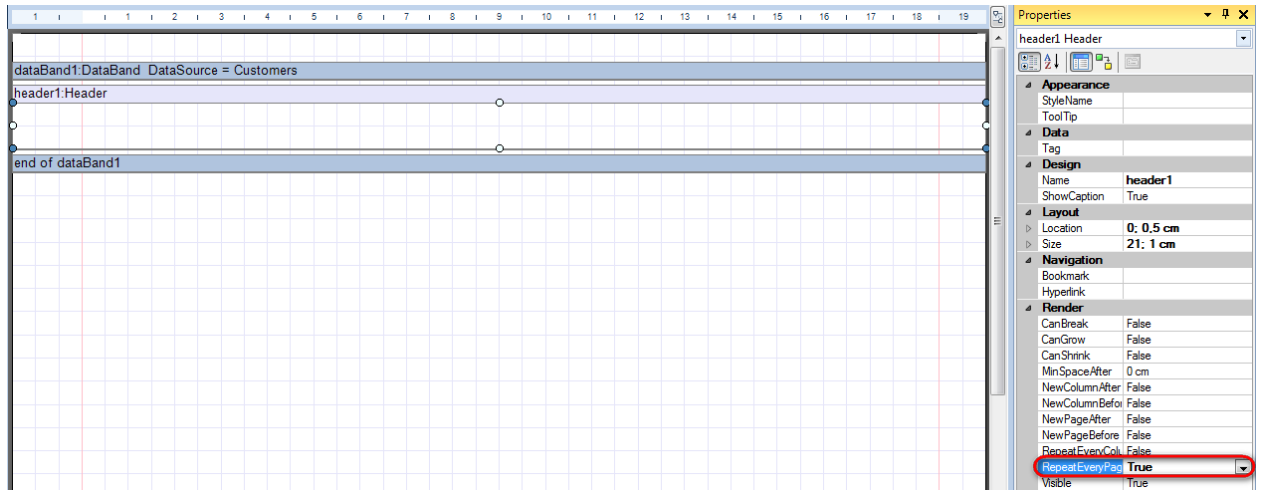
Step 15

Press "Header" button on the Insert tab in the group Container.



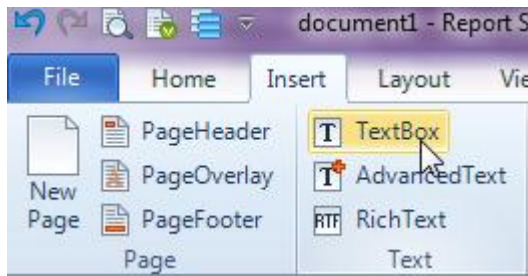
Click on the DataBand area to add Header band inside the DataBand.

Set property RepeatEveryPage to "True".

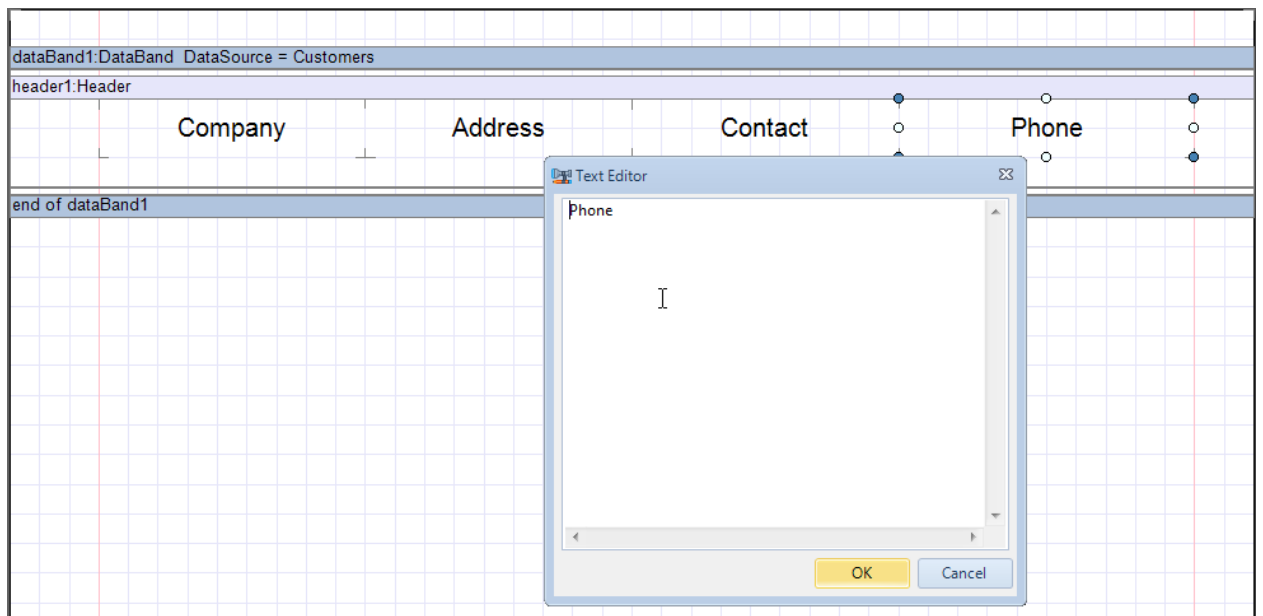


Step 16

Press button "TextBox" on the Insert tab in the group Text.

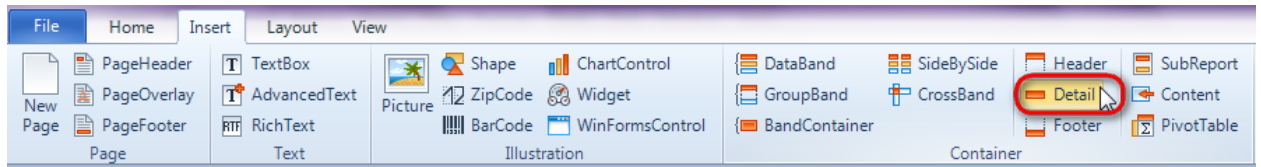


Click on the Header band area to TextBox inside Header. In the same way add another three TextBox elements. Double click on the TextBox to open Text editor – editor of the Text property. Set Text property to: "Company", "Address", "Contact", and "Phone".



Step 17

Press "Detail" button on the Insert tab in the group Container.

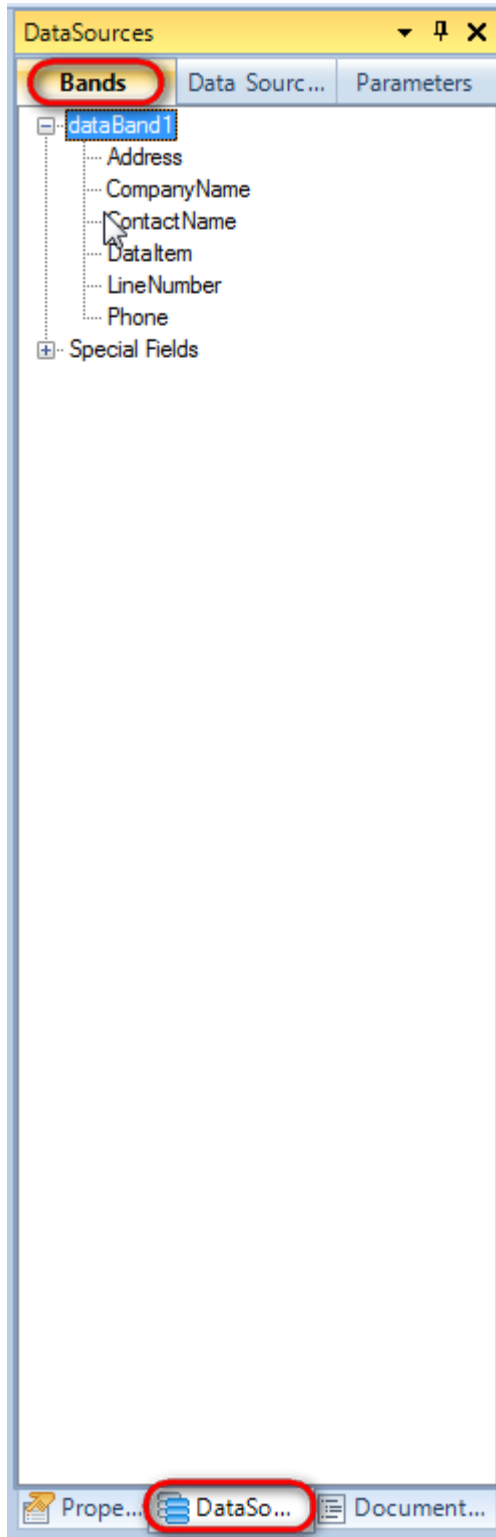


Click on the DataBand area to add the Detail band inside DataBand.

dataBand1:DataBand DataSource = Customers			
header1:Header			
Company	Address	Contact	Phone
detail1:Detail			
end of dataBand1			

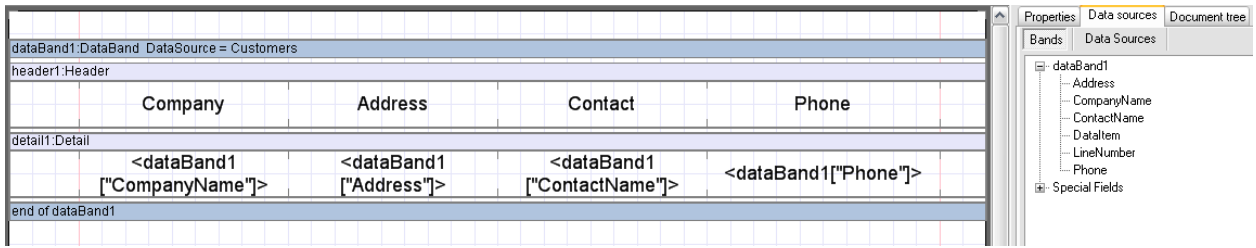
Step 18

Go to "DataSources" tab.



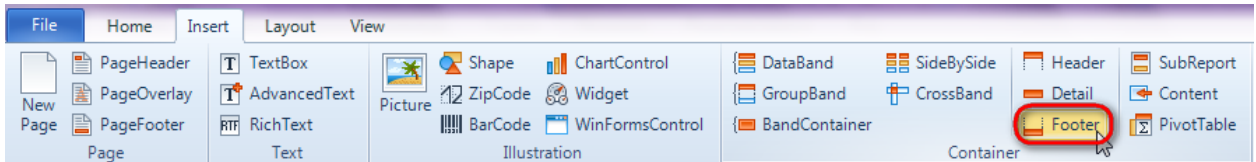
Drag and drop fields "CompanyName", "Address", "ContactName", "Phone" from the dataBand1 to the detail1 band. As a result TextBox elements will be created. Script loading data from the data source will be added to the Value property.

Change size of the elements and locate them in the following way:



Step 19

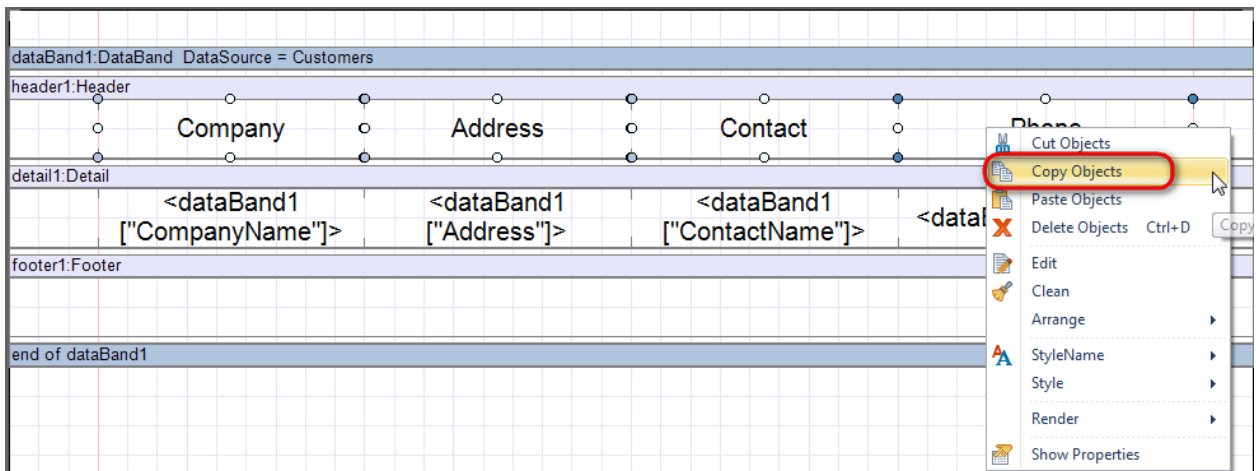
Press "Footer" button on the Insert tab in the group Container.



Click on the DataBand area to add Footer inside the DataBand.

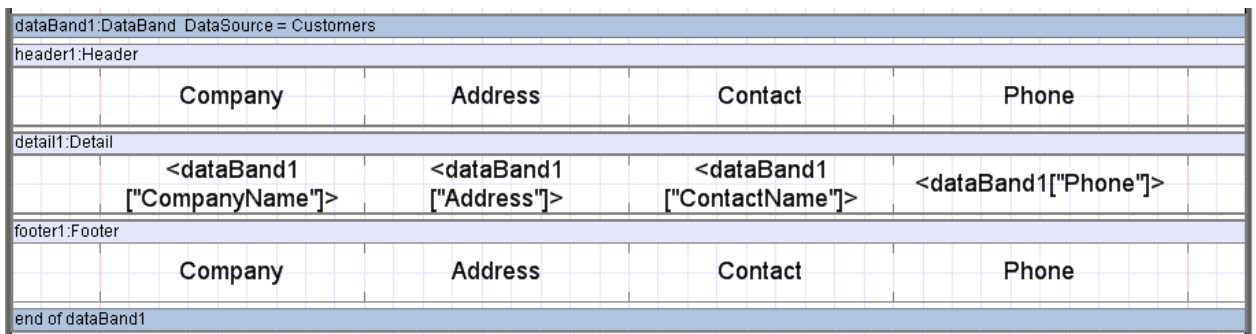
Step 20

Press Shift key and select all TextBox elements from the Header. Open context menu by right click on any of the selected elements and select "Copy Objects".



Select Footer band, open context menu by right click on the Footer band, select "Paste Objects".

Report template should look in the following way:



Step 21

Save template, close Report Designer.



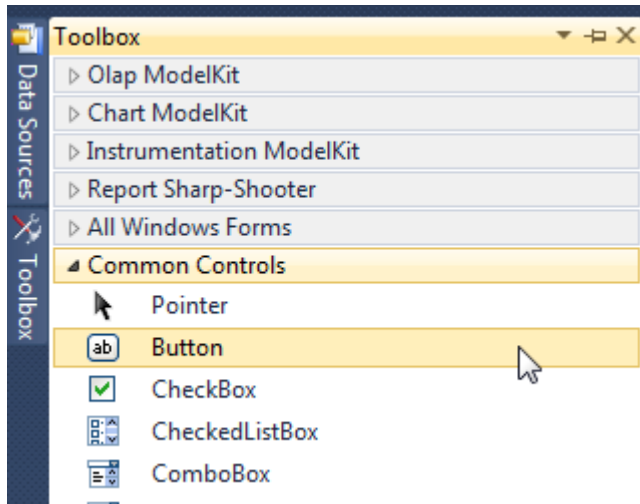
Step 22

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

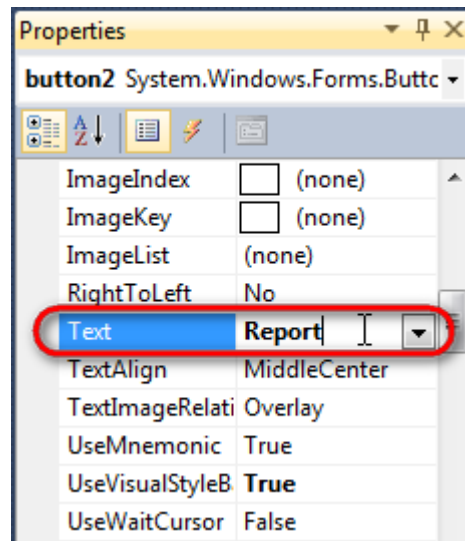
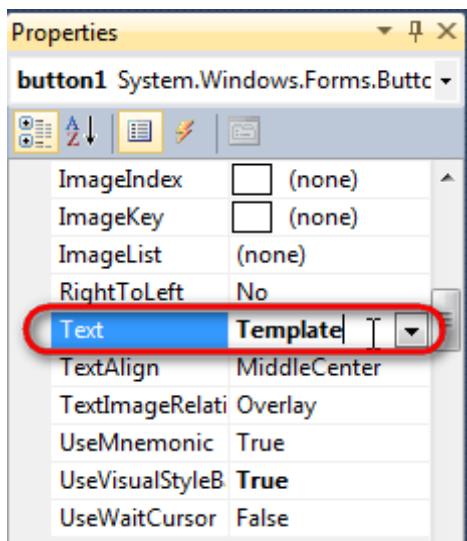
```
public Form1()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["Address"] = "Obere Str. 57";
    row["ContactName"] = "Maria Anders";
    row["Phone"] = "030-0074321";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["Address"] = "Avda. de la Constitución 2222";
    row["ContactName"] = "Ana Trujillo";
    row["Phone"] = "(5) 555-4729";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["Address"] = "Kirchgasse 6";
    row["ContactName"] = "Roland Mendel";
    row["Phone"] = "7675-3425";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["Address"] = "Luisenstr. 48";
    row["ContactName"] = "Karin Josephs";
    row["Phone"] = "0251-031259";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
    EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
    PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 23

Place two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the second one.



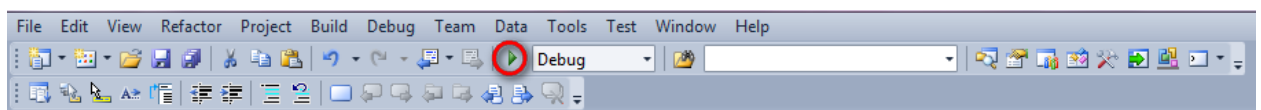
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

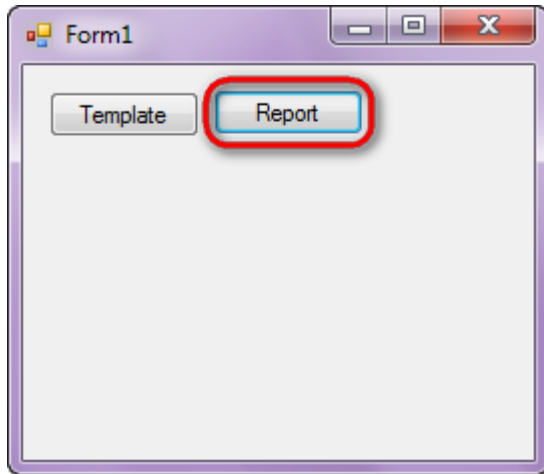
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 24

Click "Start Debugging" on the Visual Studio toolbar in order to run application.



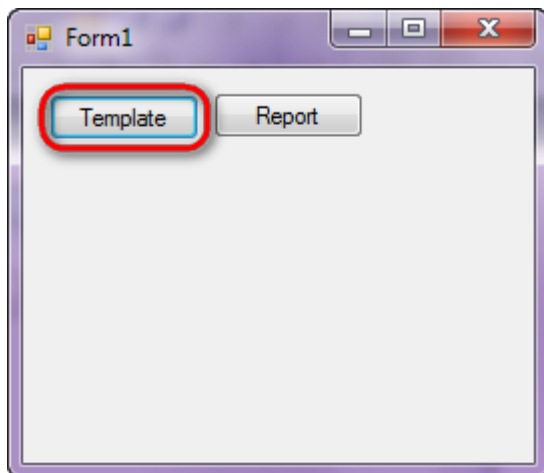
Click the "Report" button in the opened application window.



Generated report will open with Report Viewer.

Company	Address	Contact	Phone
Alfreds Futterkiste	Obere Str. 57	Maria Anders	030-0074321
Ana Trujillo Emparedados y helad	Avda. de la Constitución 2222	Ana Trujillo	(5) 555-4729
Ernst Handel	Kirchgasse 6	Roland Mendel	7675-3425
Toms Spezialitäten	Luisenstr. 48	Karin Josephs	0251-031259
Company	Address	Contact	Phone

In order to edit template, close Report Viewer and press "Template" on the application form.



Similar application sample is located in the following folder "Samples\Report Sharp-Shooter\CSharp\Simple List".

Similar sample in the Samples Center is Reports\Simple Reports\List.

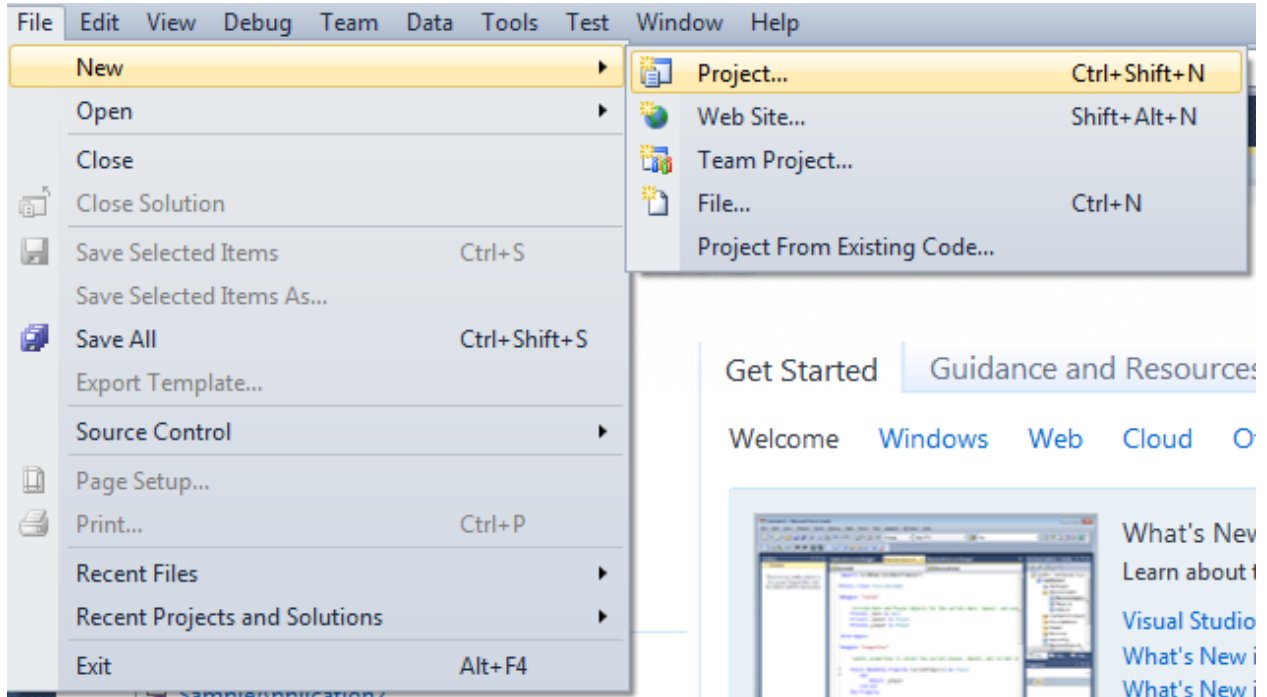


Page Size, Multipage Template

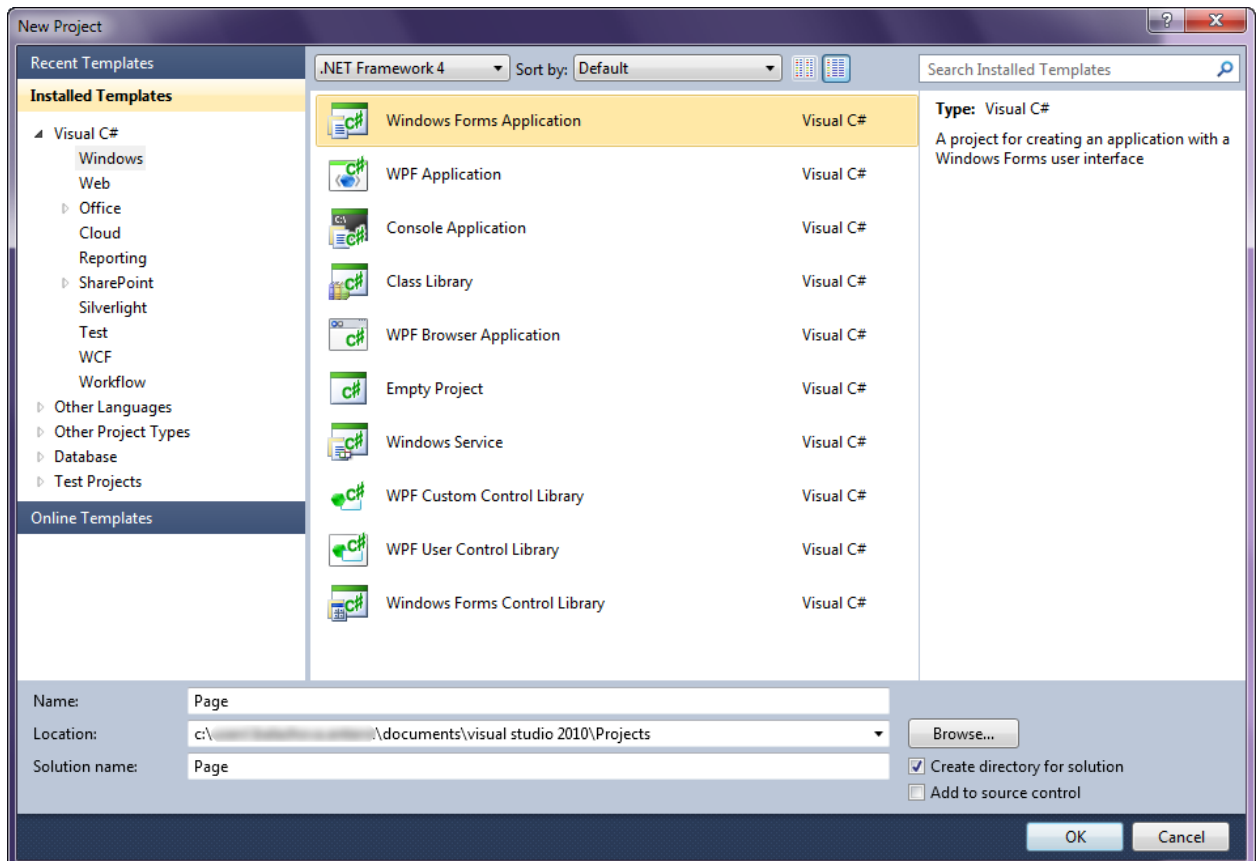
Template of a report containing two pages of different size. Each page contains information on page number and size.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.



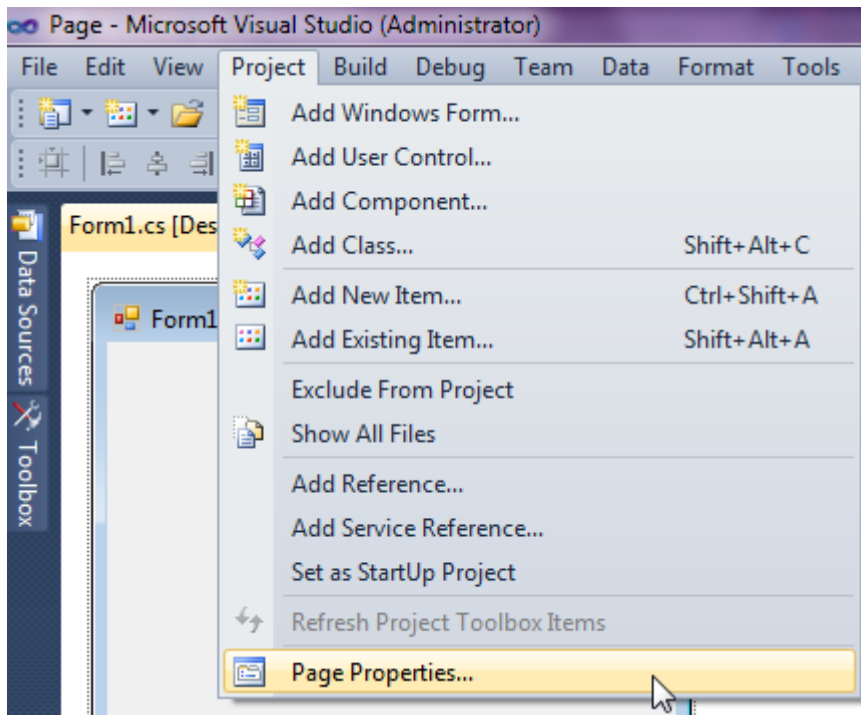
Select Windows Forms Application, set project name – “Page”, set directory to save the project to.



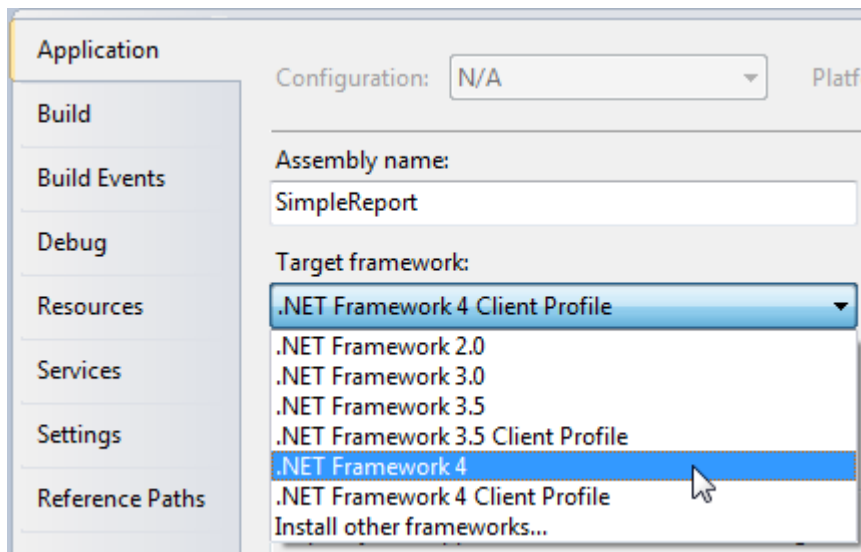


Step 2

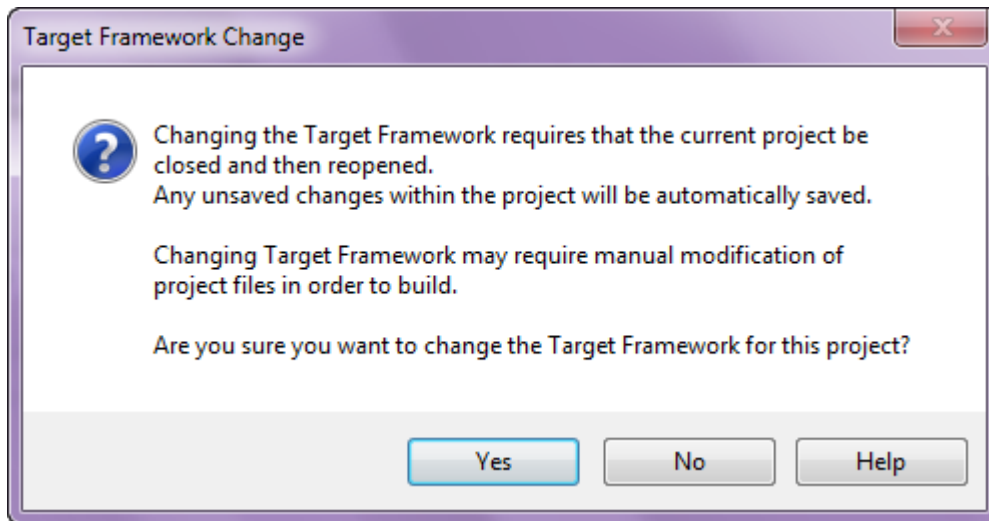
Change the project properties. Select the Project\Page Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

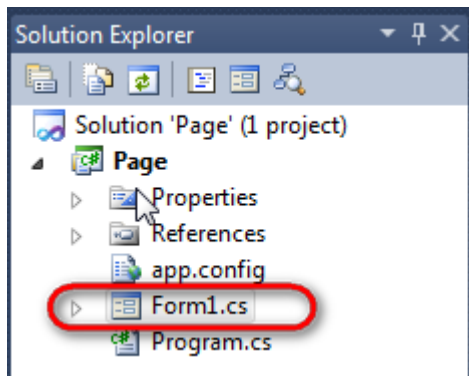


Press the "Yes" button in the opened window.

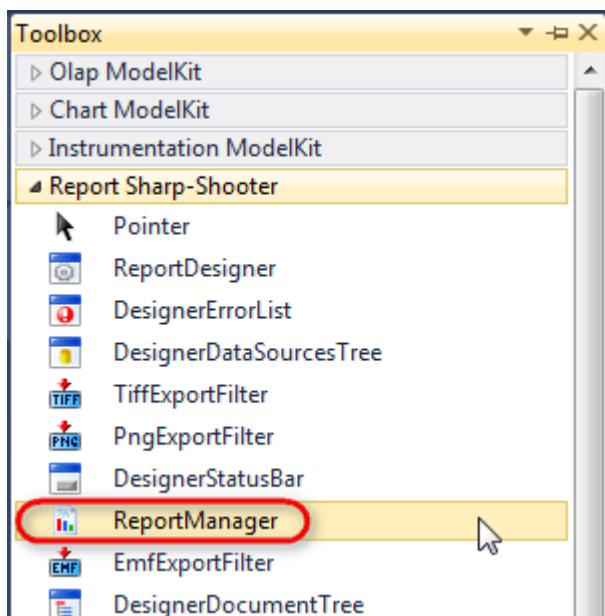


Step 3

Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

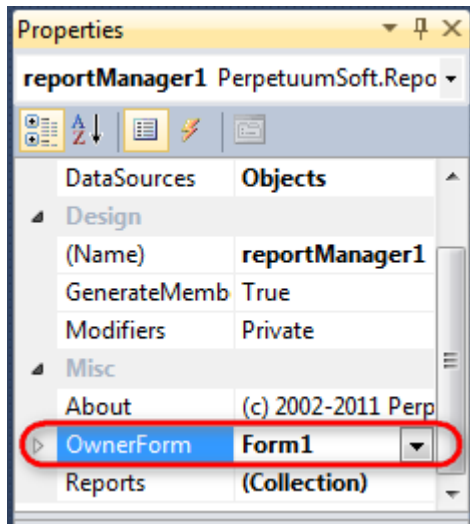


The component is available in the lower part of the window.

 reportManager1

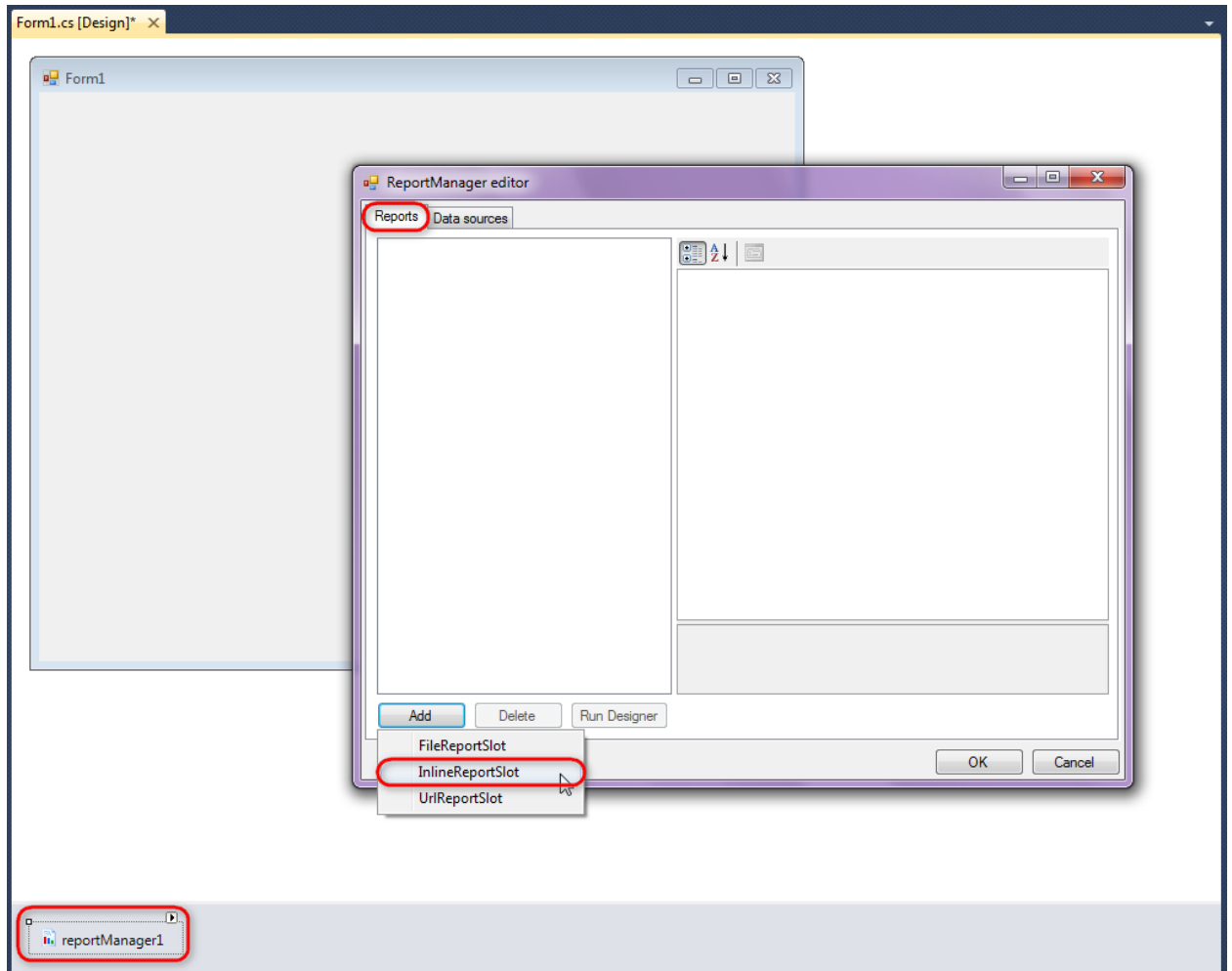
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

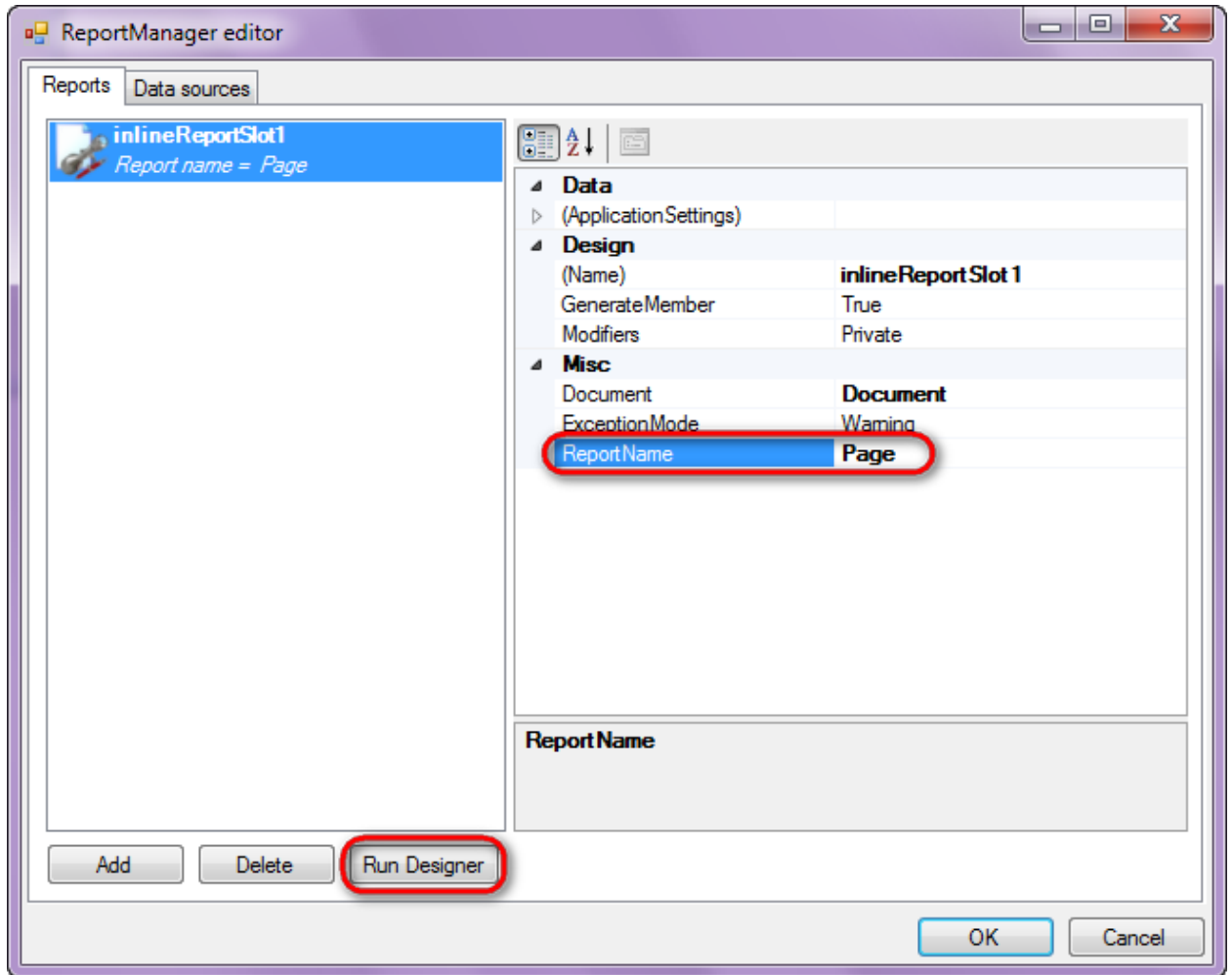


On the "Reports" tab click "Add" and select "InlineReportSlot".

Step 6

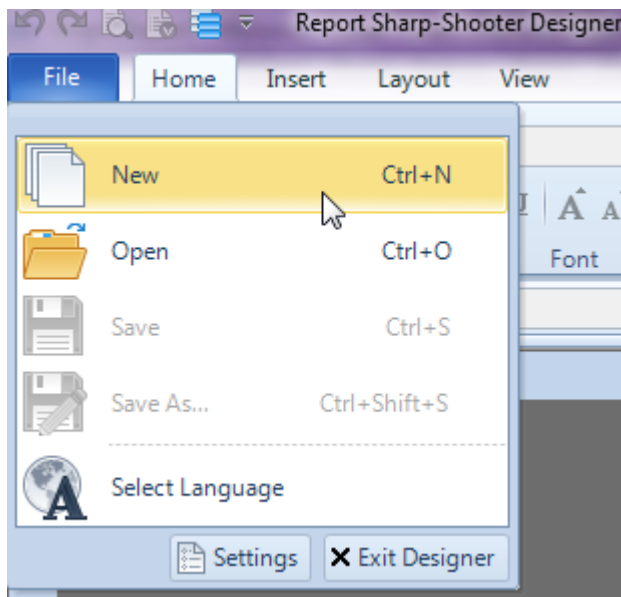
Set name of the report in the property ReportName – "Page".

Click "Run Designer" to open template editor Report Designer.

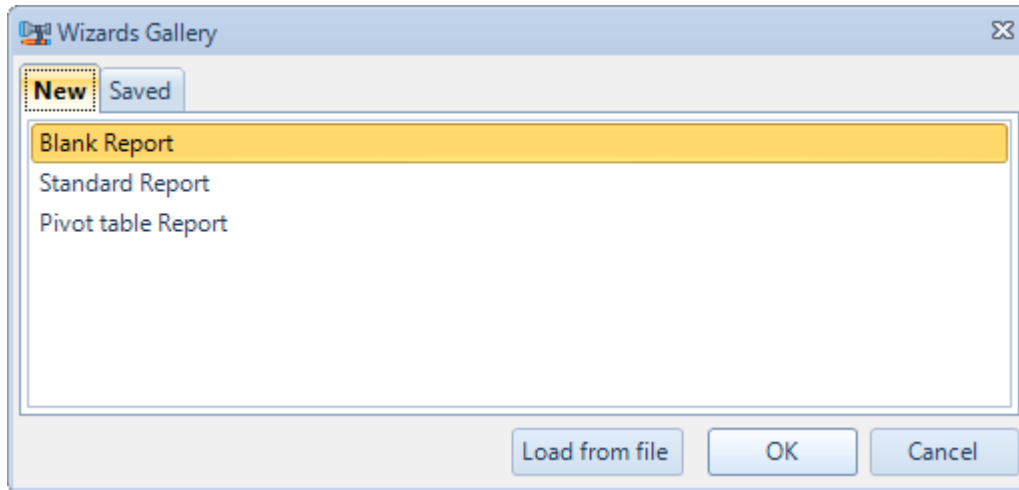


Step 7

Create new empty report – select File\New from the main menu.

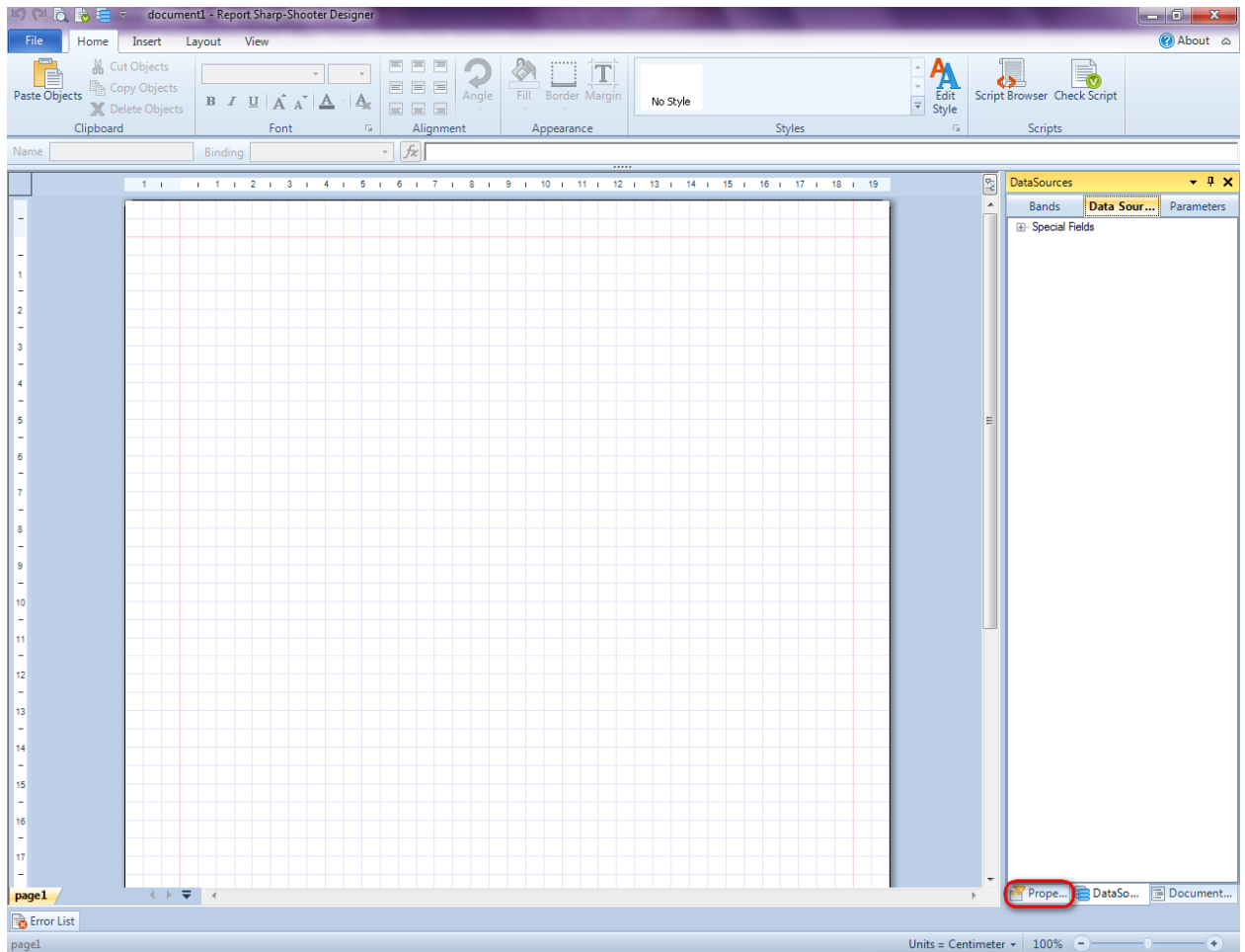


Select "Blank Report" in the Wizards Gallery and click "OK".

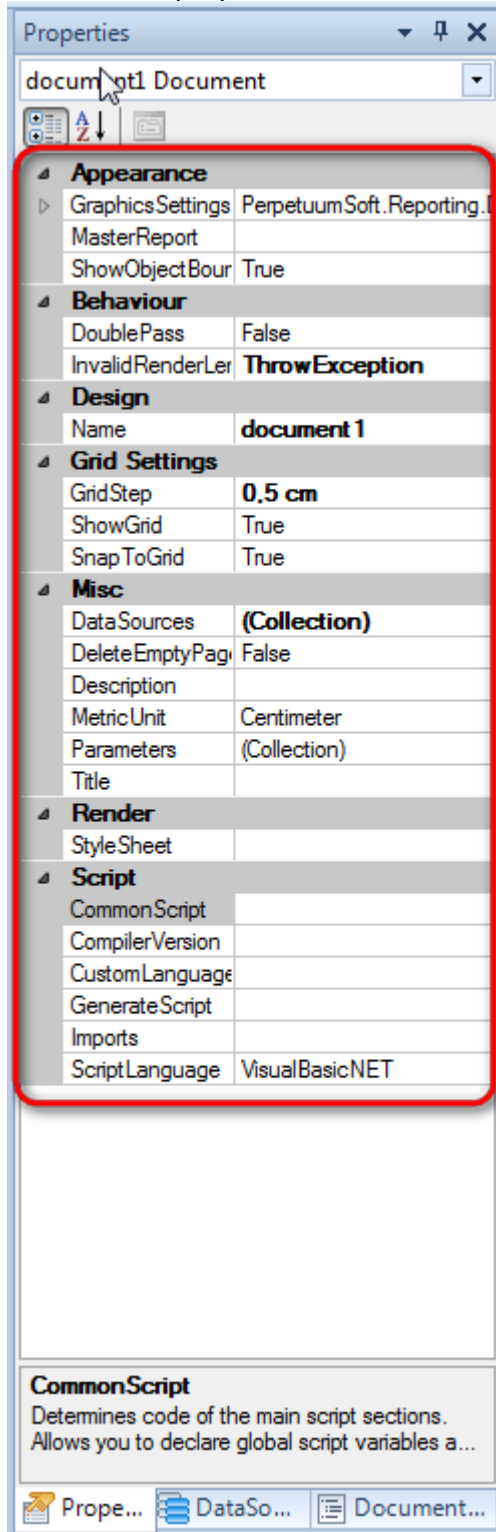


Step 8

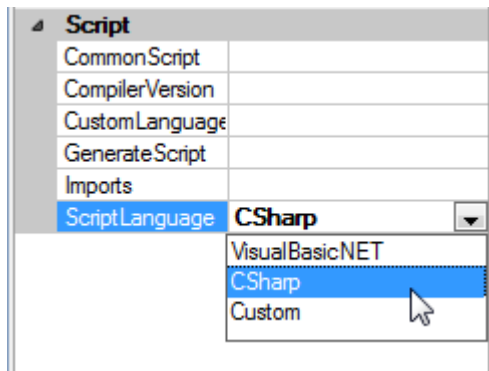
Click the "Properties" tab of the tool window in the right part of the designer.




You will see properties of the edited template on the “Properties” tab

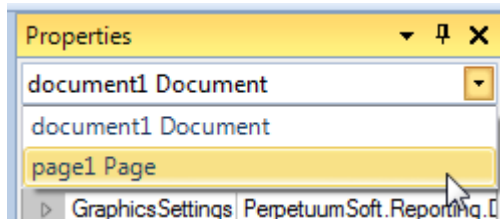


Set property ScriptLanguage = CSharp.

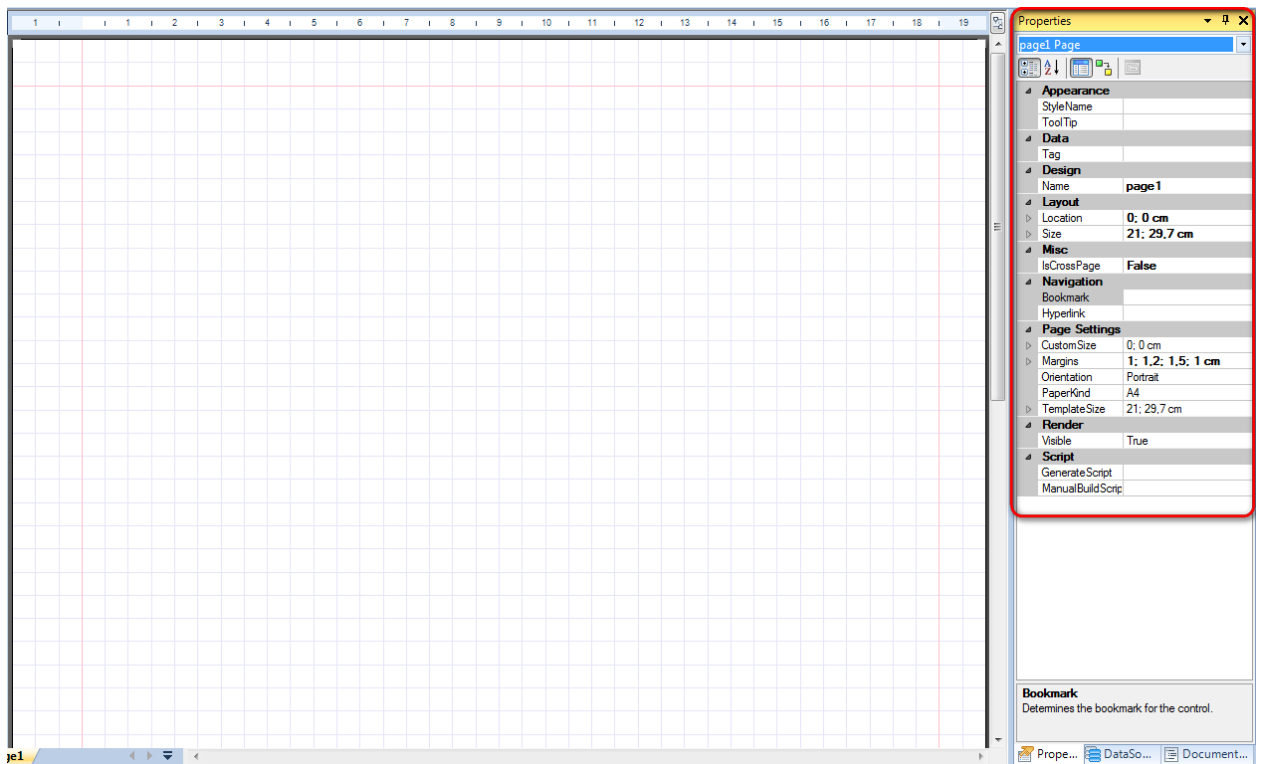



Step 9

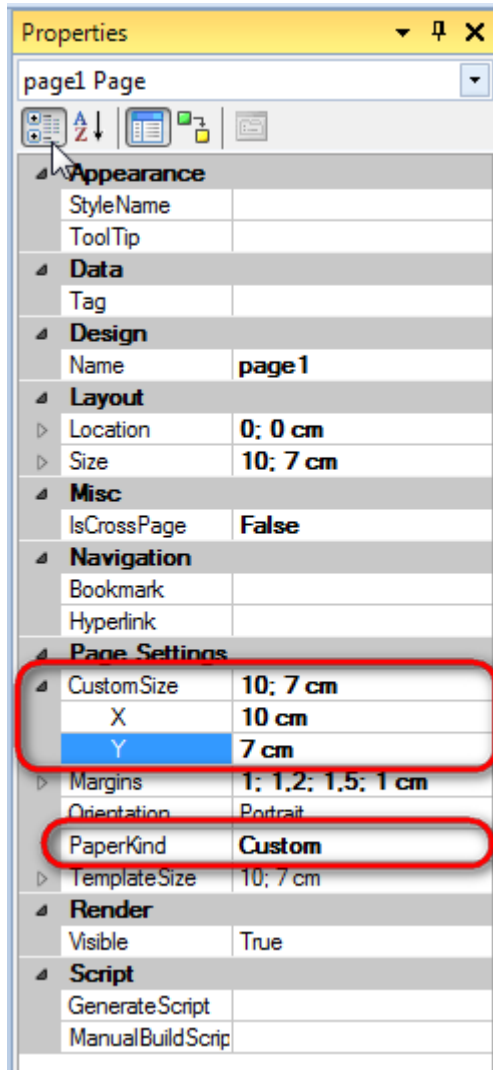
Press the  button in Properties. Select the "page1 Page" item in the combo box.



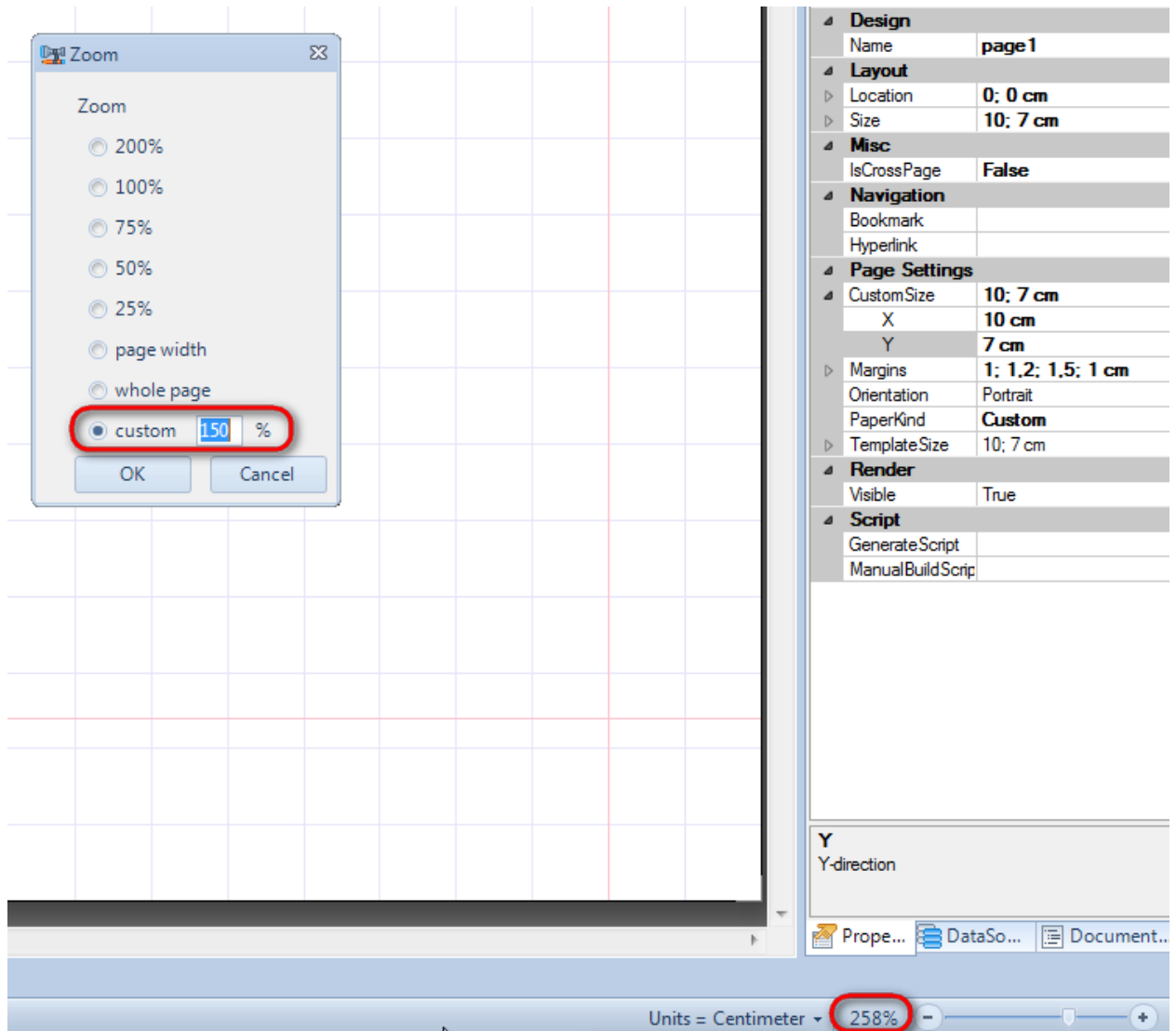
You will see properties of the edited page in the "Properties" tab of Tool Window in the right part of the designer.




Set the PaperKind = Custom property. Click the  button which is located to the left from CustomSize and set X = 10 cm; Y = 7 cm.

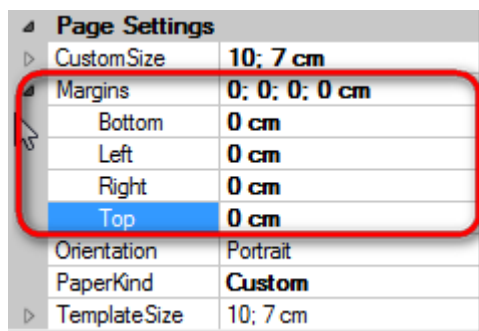


To make the report template preview more convenient, change zoom in the toolbar. Set value to "150%".



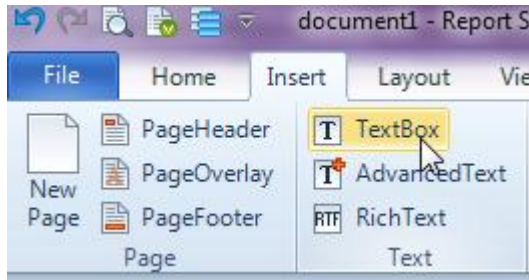
Step 10

Click the  button. That is located to the left from the Margins property, set all values to "0".



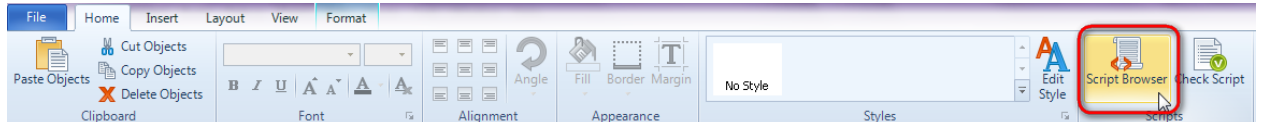
Step 11

Press the "TextBox" button in the Insert tab in the group Text.

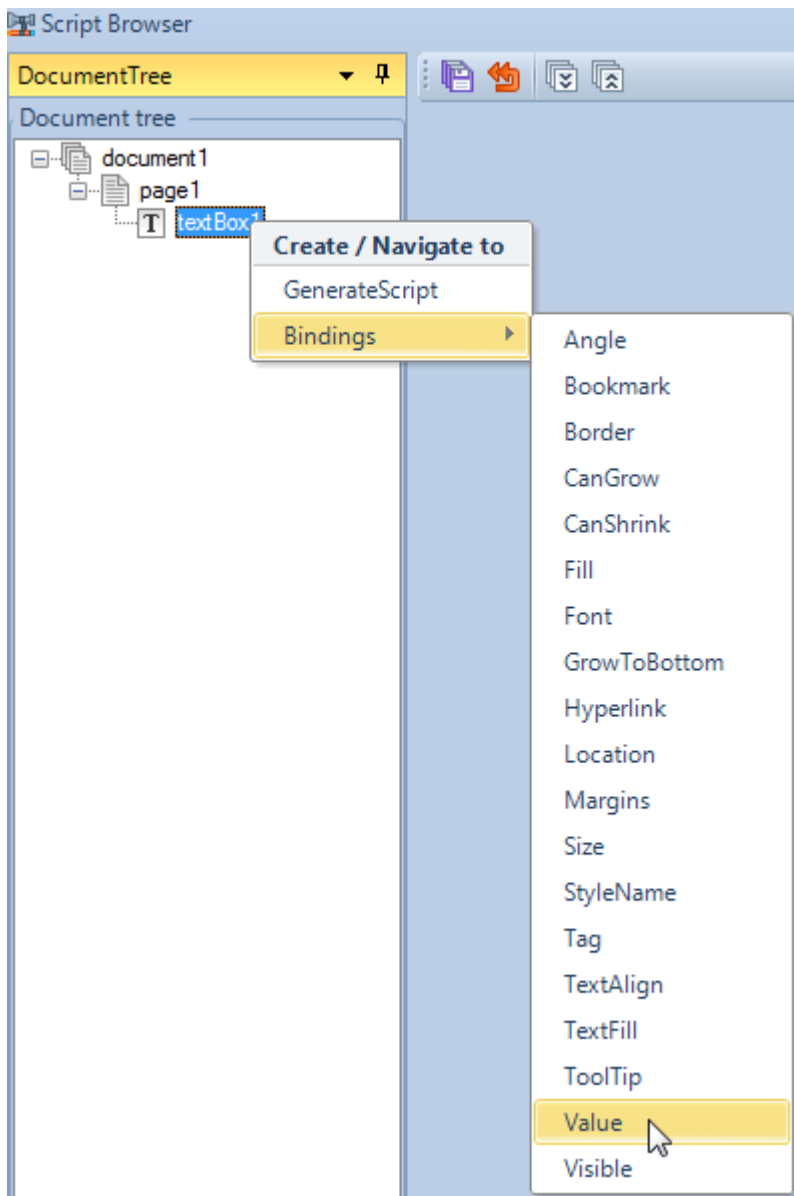


Click on the template area to add the TextBox element.

Click the "Script Browser" button in the Home tab in the group Scripts.



Right-click on TextBox1 in the DocumentTree. Select Bindings\Value in appeared context menu.



In the opened window, write the following code:

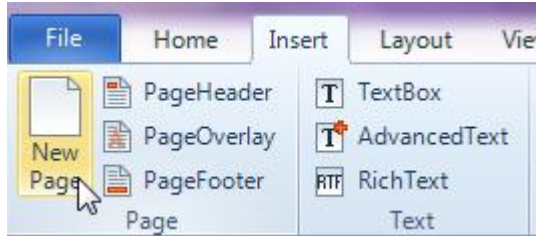


"Page N°" + PageNumber.ToString() + " of " + PageCount.ToString() + ". 10x7 cm".

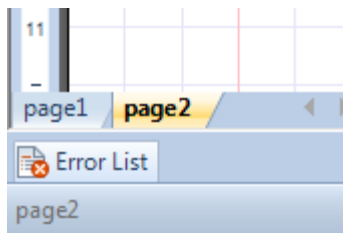
Click the "Save all" button and close the Script Browser.

Step 12

Press the "NewPage" button in the Insert tab in the group Page to add a new template.

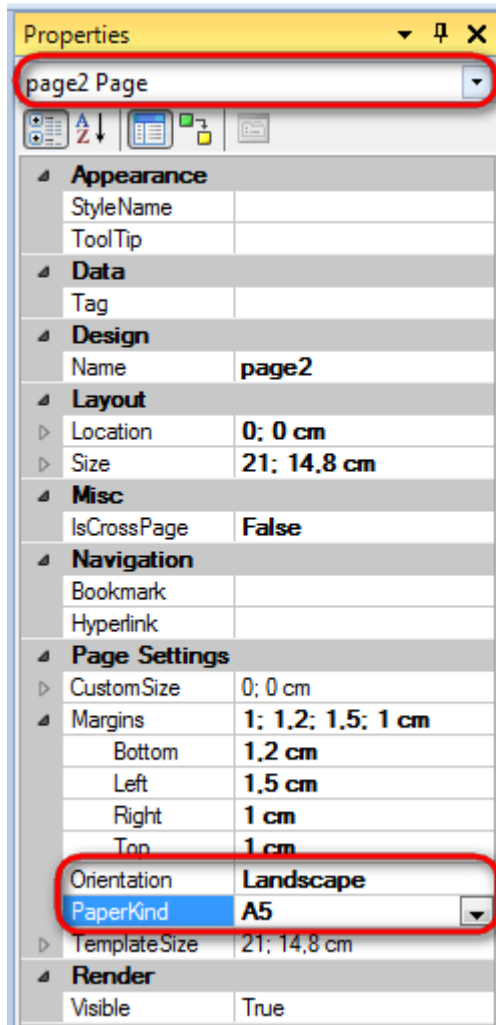


To switch between pages use tabs in the left lower corner.



Step 13

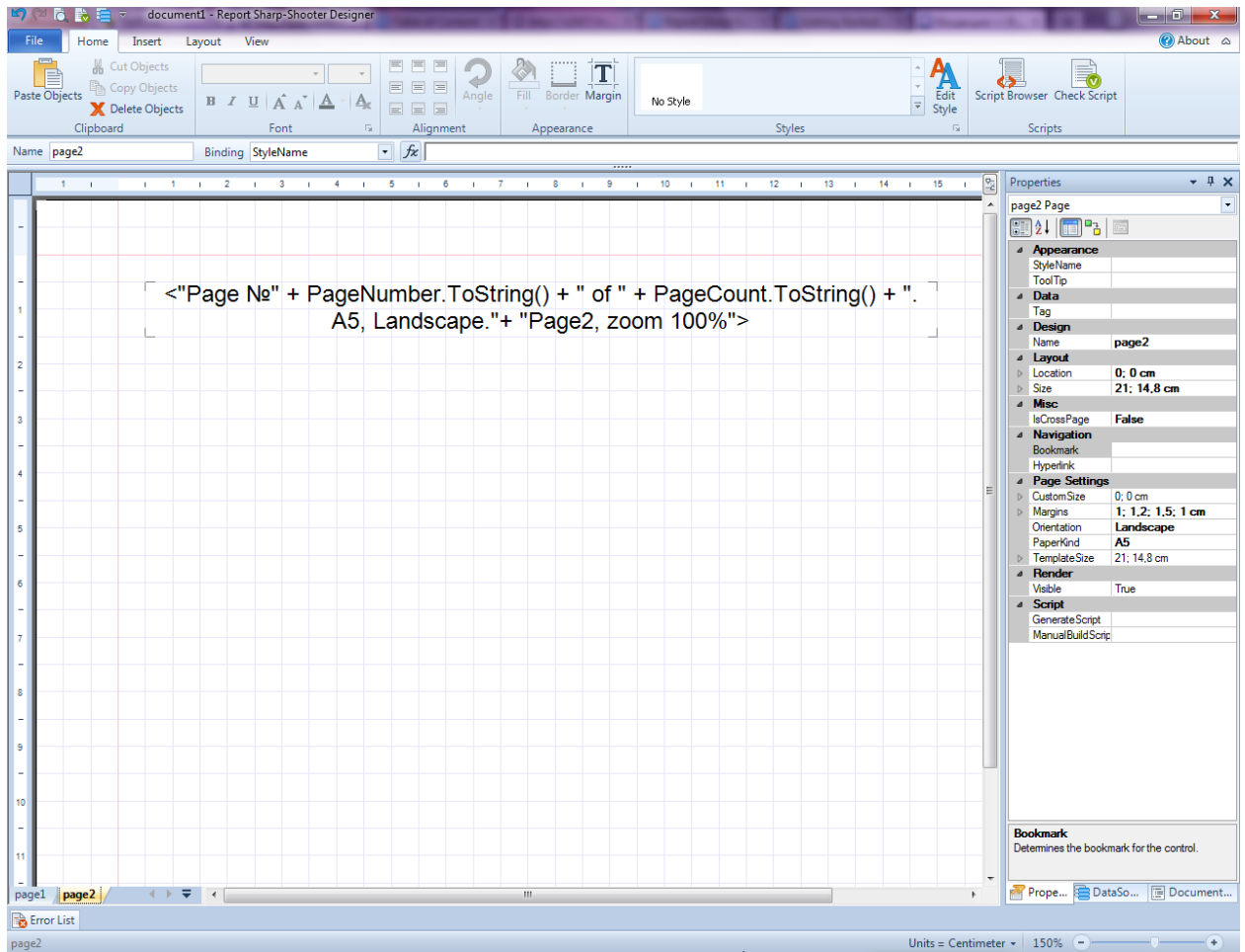
Go to properties of the page2. Set properties PaperKind = A5, Orientation = Landscape.



Step 14

Add the TextBox element to page2. Open Script Browser. Right click on the TextBox2 item in the DocumentTree. Select Bindings\Value in the appeared context menu. Add the following script in the opened window: "Page N°" + PageNumber.ToString() + " of " + PageCount.ToString() + ". A5, Landscape." + "Page2, zoom 100%".

Press the "Save all" button and close Script Browser.

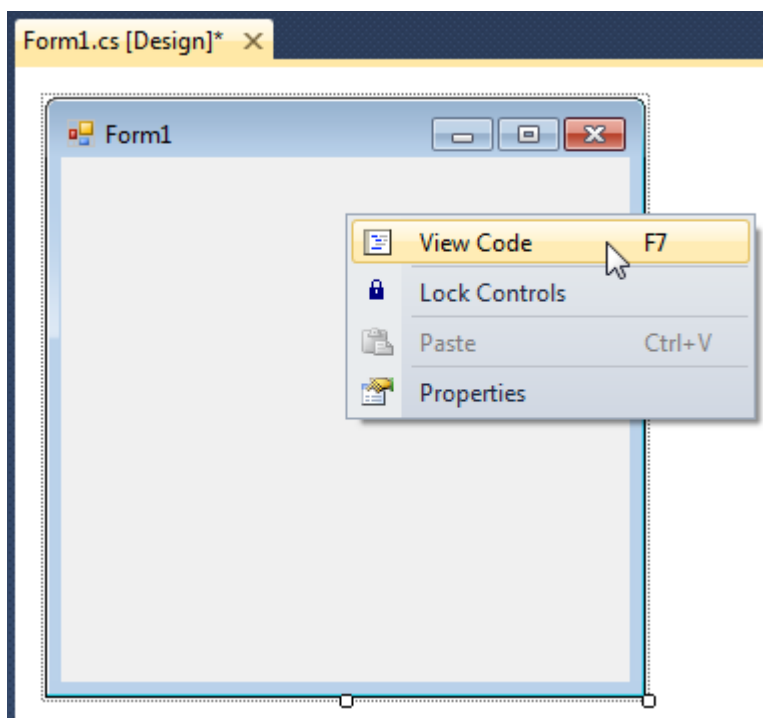


Step 15

Save the template, close Report Designer.

Step 16

Right click on the form and select "View Code" in the context menu to view code.



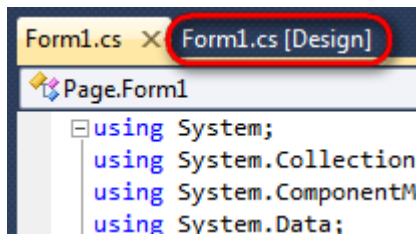


To display the report, add the code to the class constructor. Write the RenderCompleted event handler of the InlineReportSlot object.

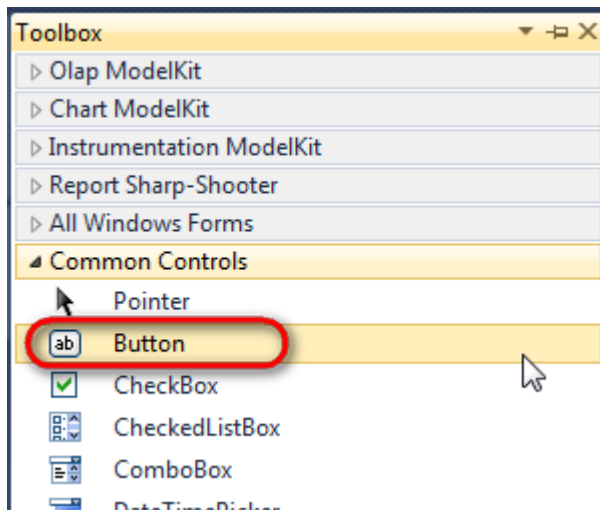
```
public Form1 ()
{
    InitializeComponent();
    inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 17

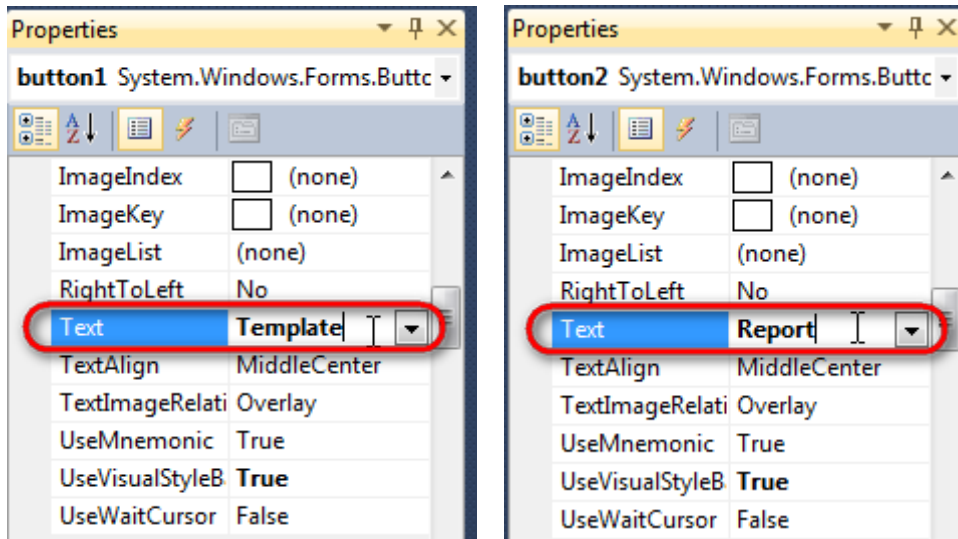
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop the "Button" element from the Toolbox onto the form).



Select the Button element in the form, edit the Text property in the property grid. Set Text = Template for the first button and Text = Report for the second one.



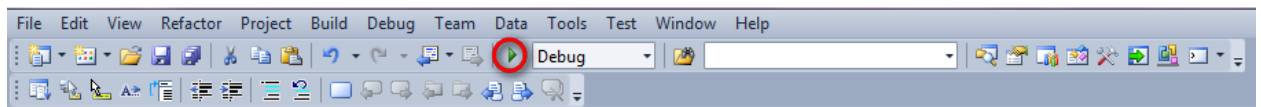
Create the Click event handlers for the buttons – double click on the Button element in the form. Add code for report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

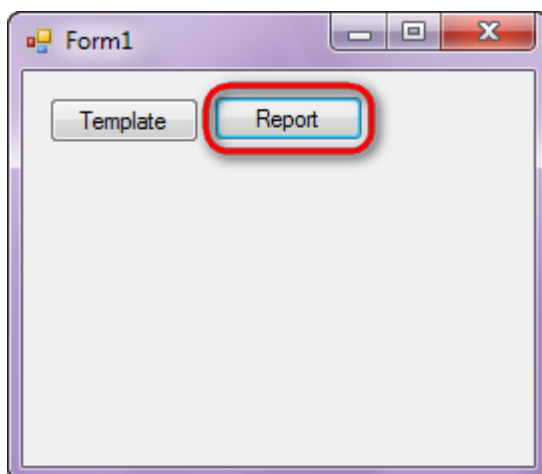
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 18

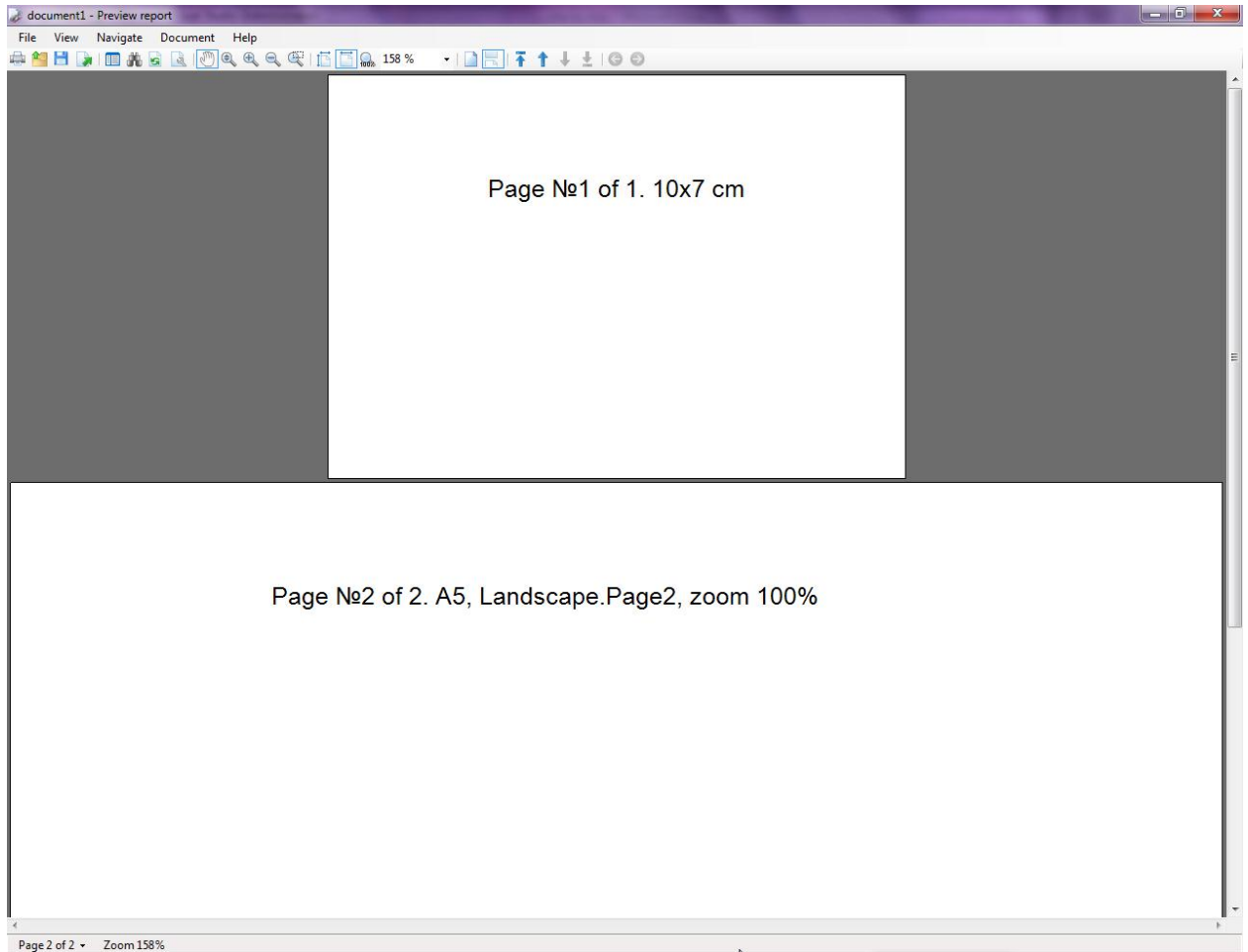
Click “Start Debugging” in the Visual Studio toolbar in order to run application.



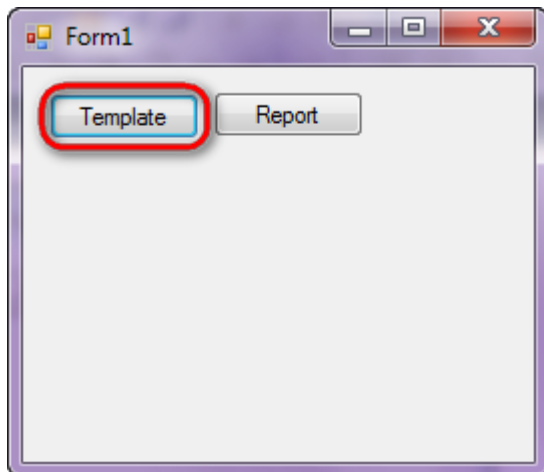
Click the “Report” button in the opened application window.



The generated report can be opened with Report Viewer.



In order to edit the template, close Report Viewer and press "Template" in the application form.



Similar sample can be found in the Samples Center: Reports\Special features\The use of two pages template.

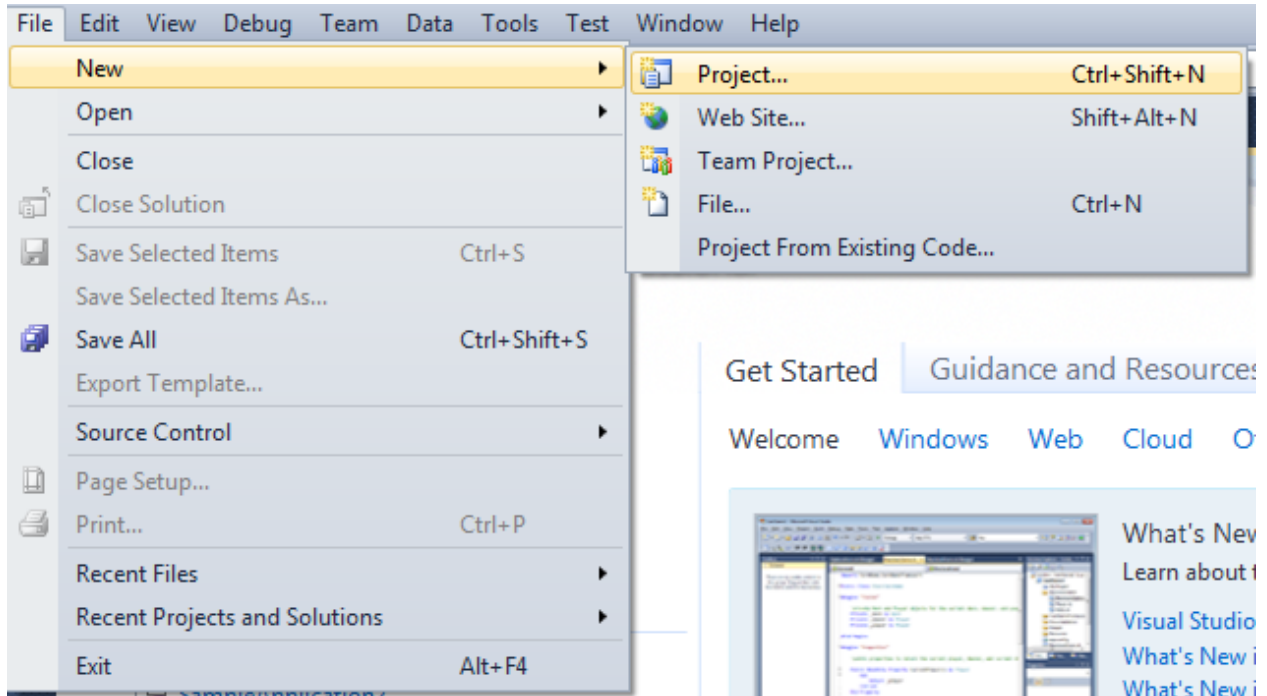


Breaking Text into Pages

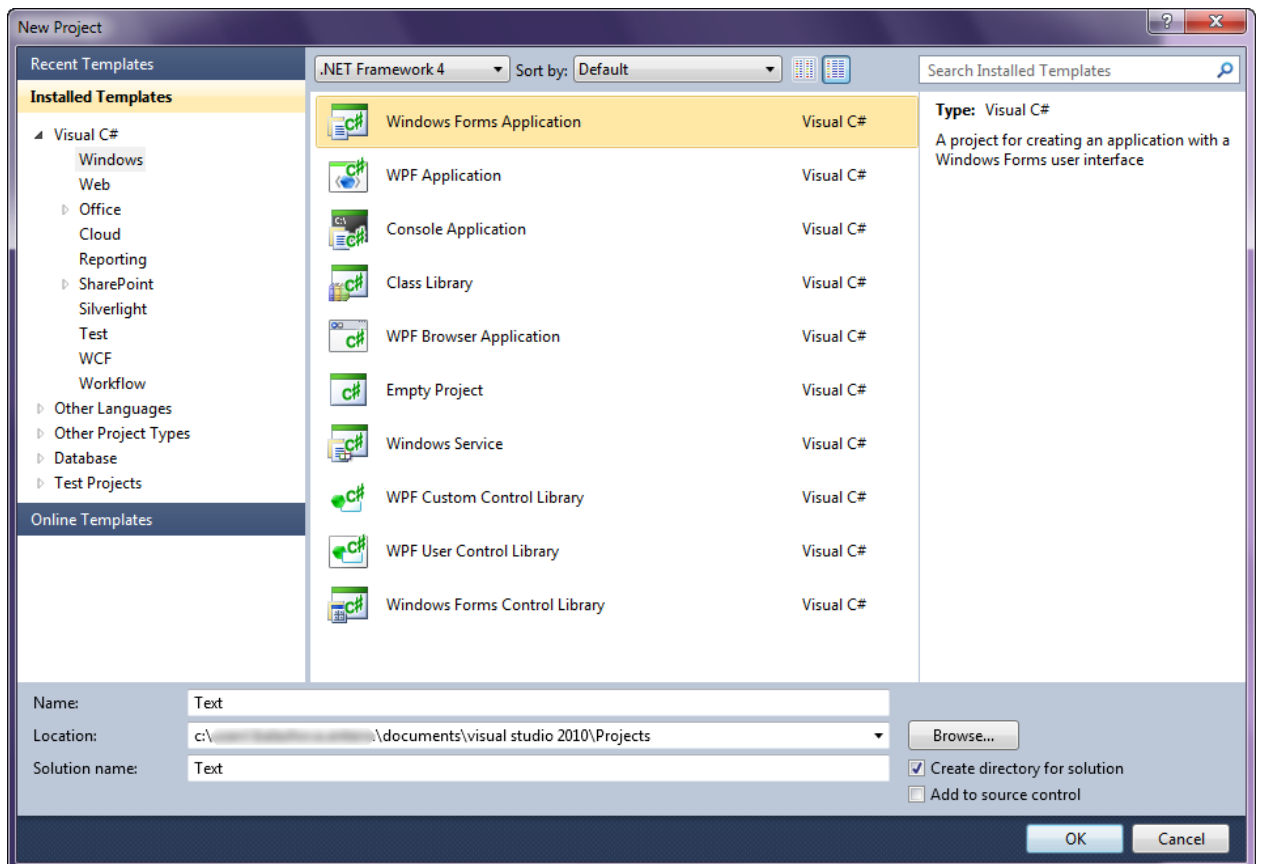
Template of a report containing breaking long text into pages.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

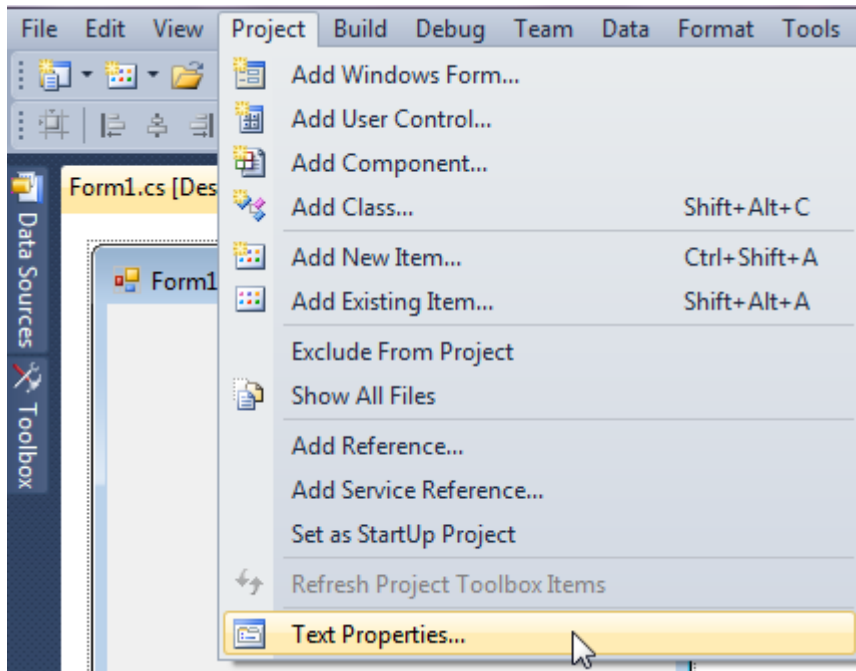


Select Windows Forms Application, set project name – “Text”, set directory to save the project to.

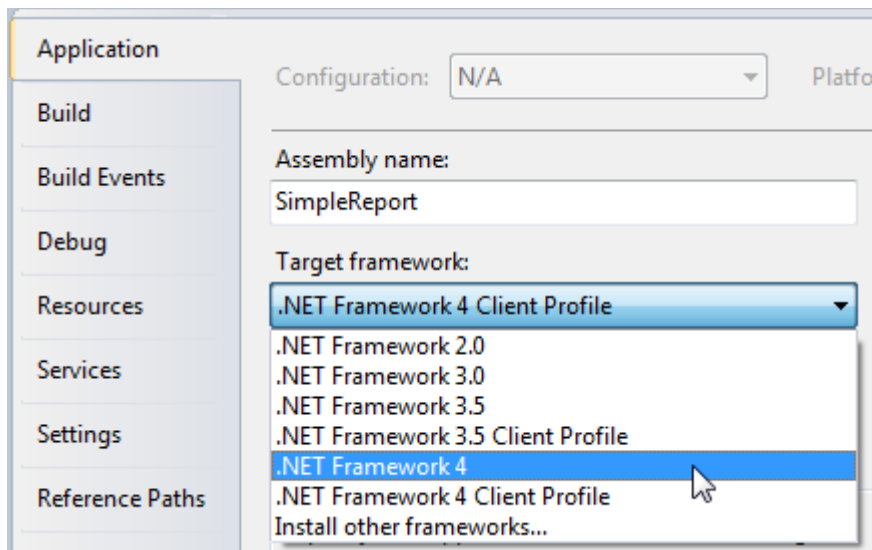


Step 2

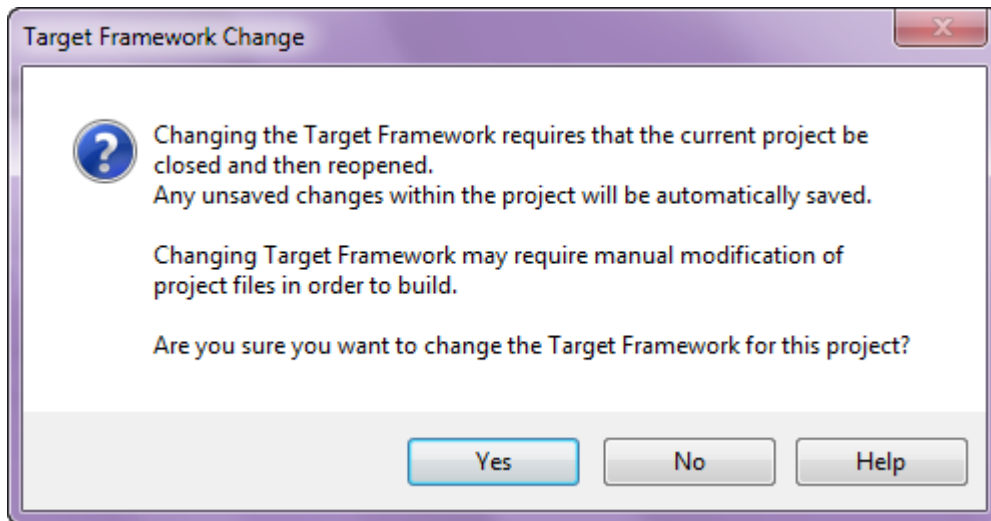
Change the project properties. Select the Project\Text Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

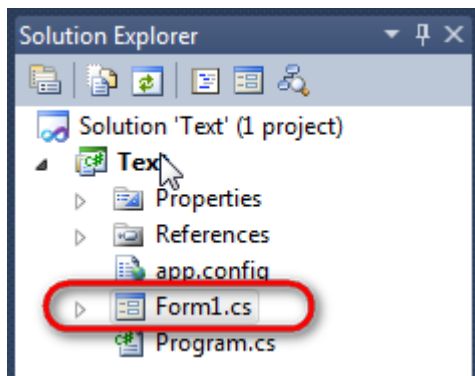


Press the "Yes" button in the opened window.

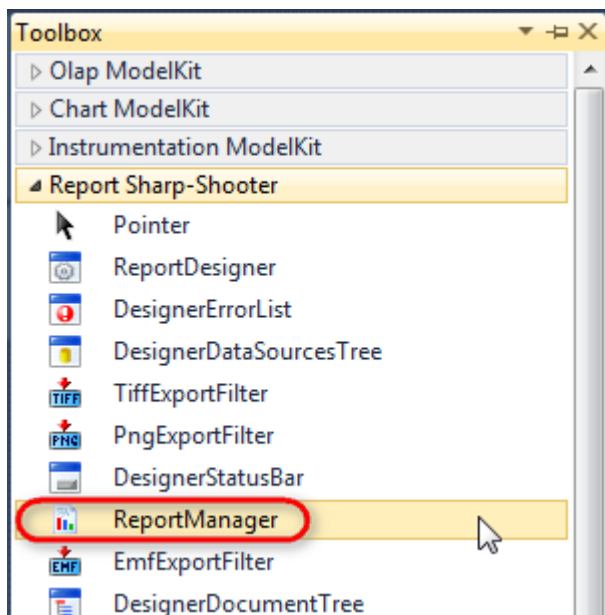


Step 3

Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

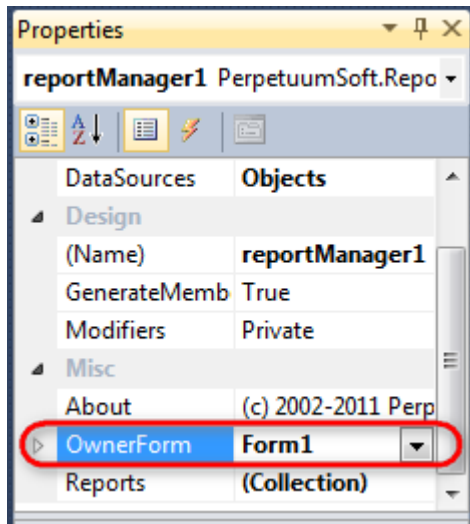


The component is available in the lower part of the window.



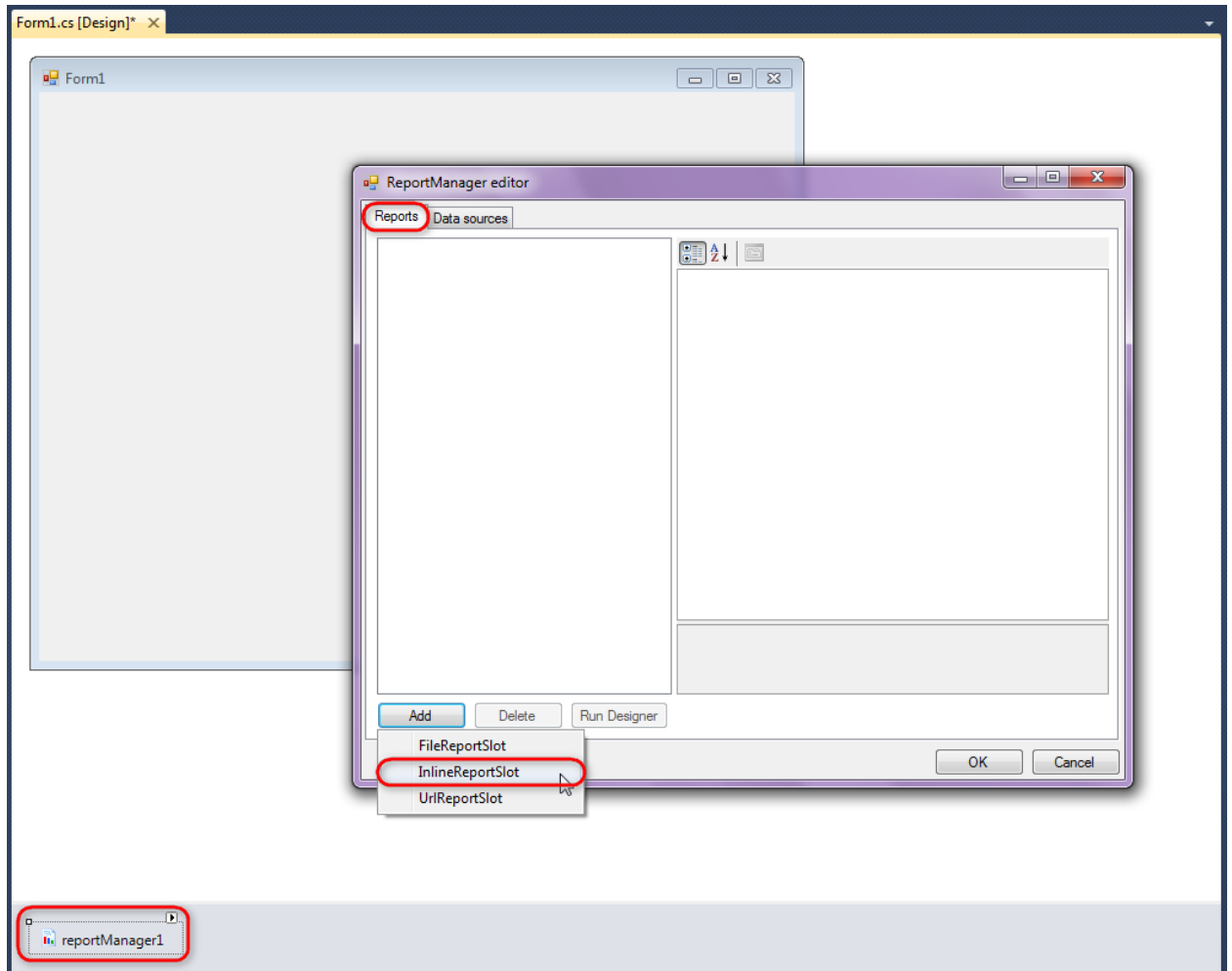
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

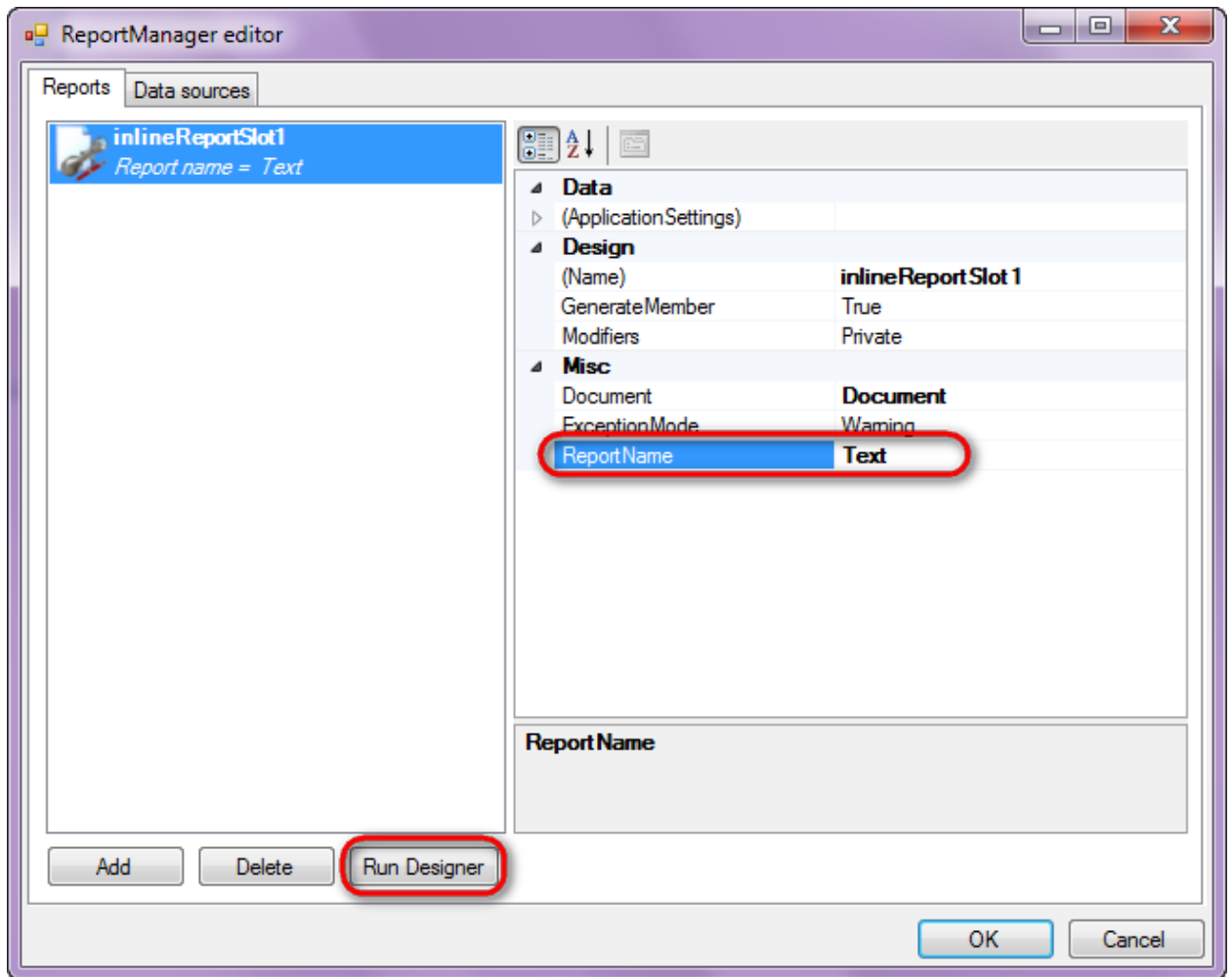


Go to "Reports tab", click "Add" and select "InlineReportSlot".

Step 6

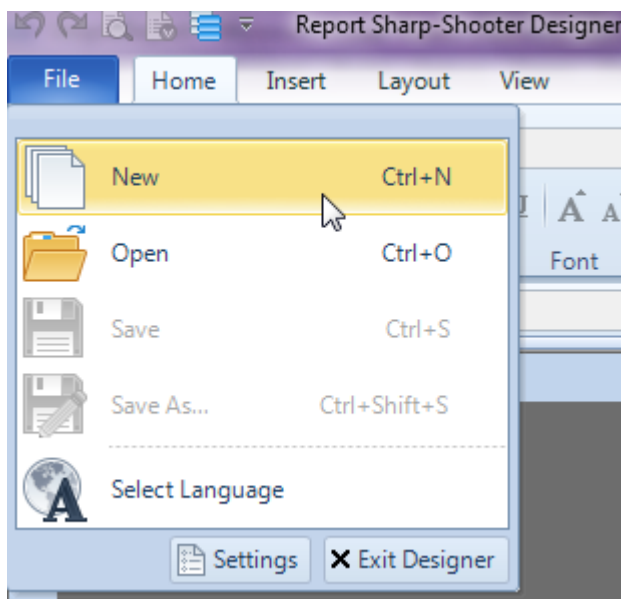
Set name of the report in the property ReportName - "Text".

Click "Run Designer" in order to open template editor - Report Designer.

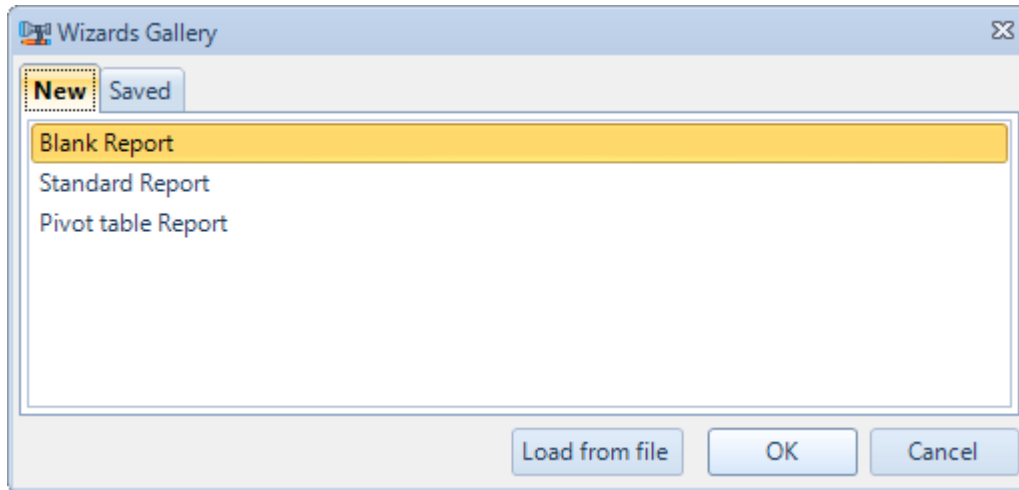


Step 7

Create new empty report – select File\New from the main menu.

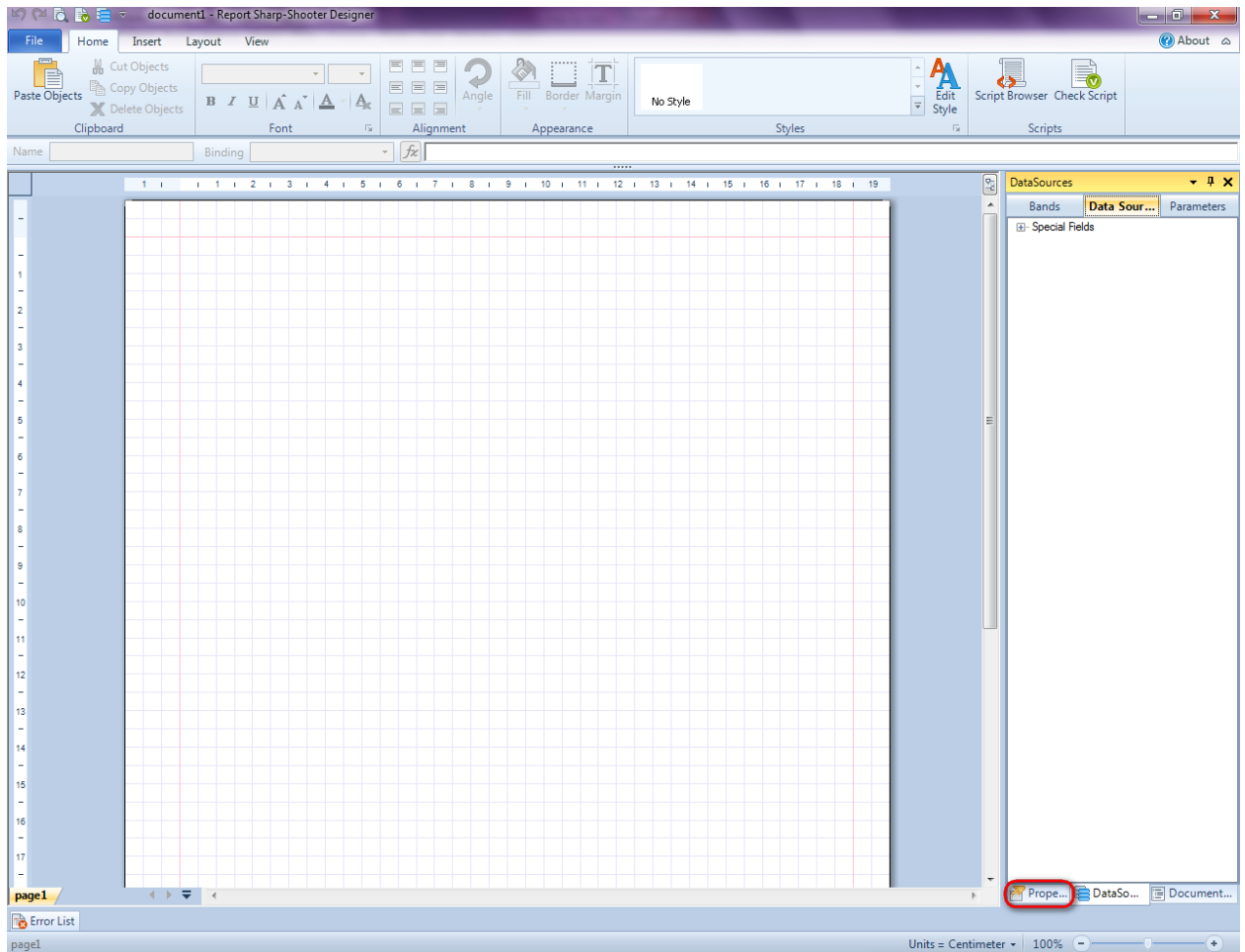


Select "Blank Report" in the Wizards Gallery and click "OK".

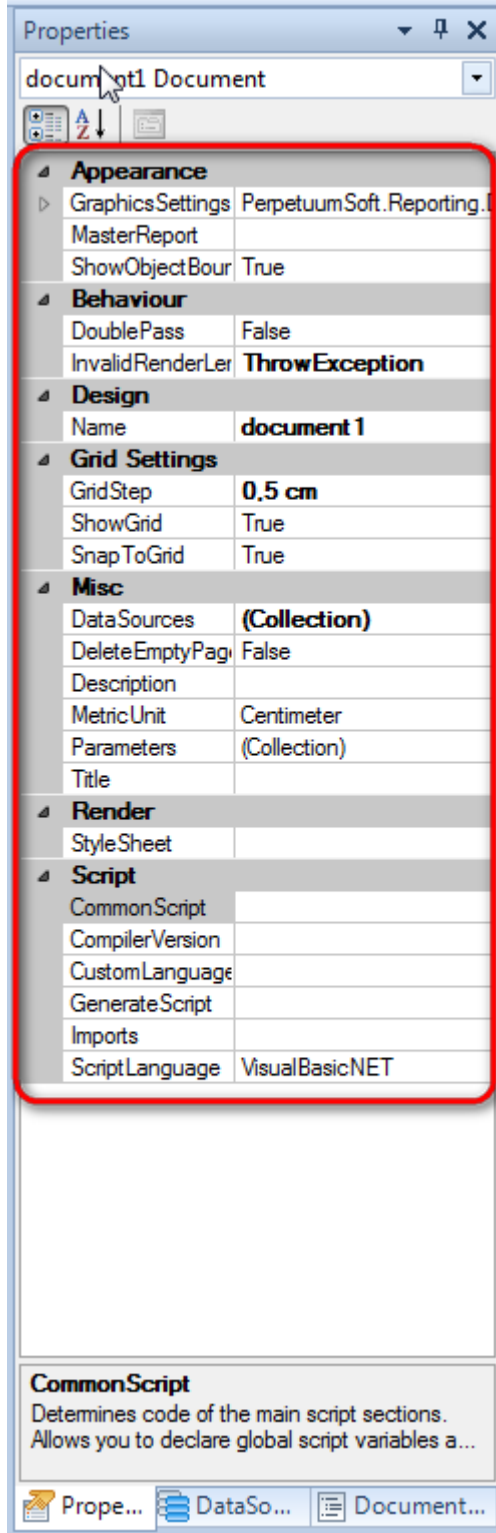


Step 8

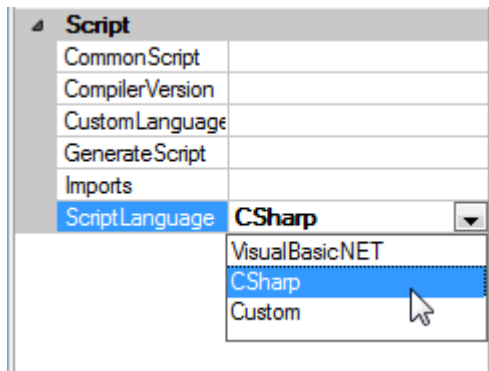
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

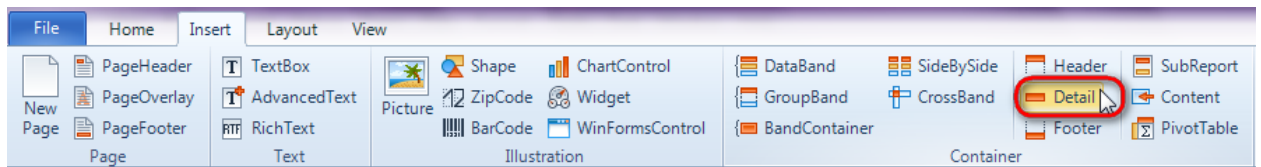


Set property ScriptLanguage = CSharp.

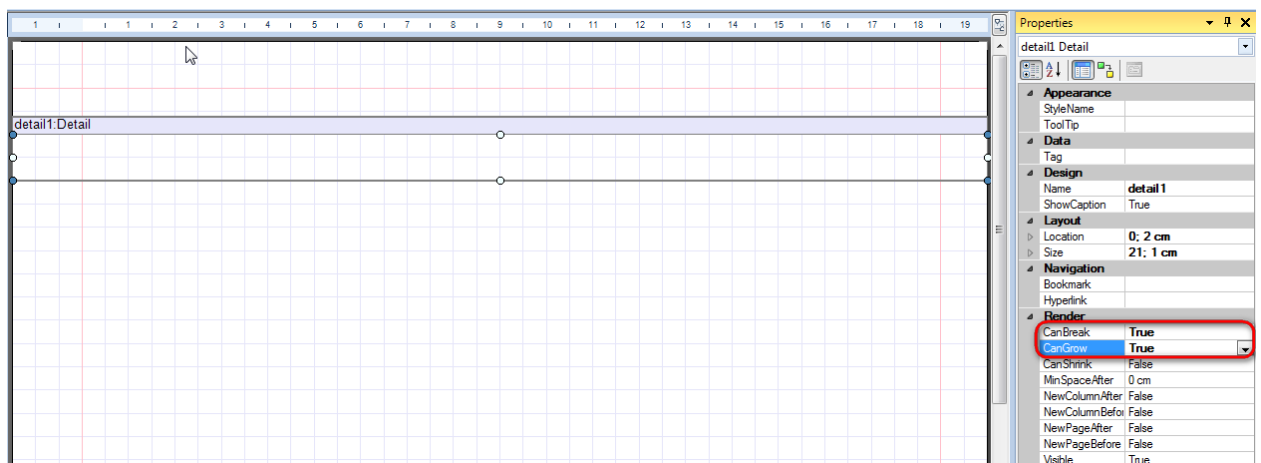


Step 9

Press "Detail" button on the Insert tab in the group Container.

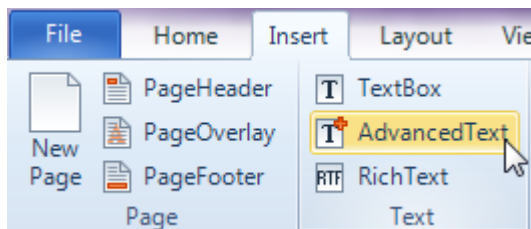


Click on the template area to add Detail band to the template. Set CanGrow and CanBreak properties to "True".



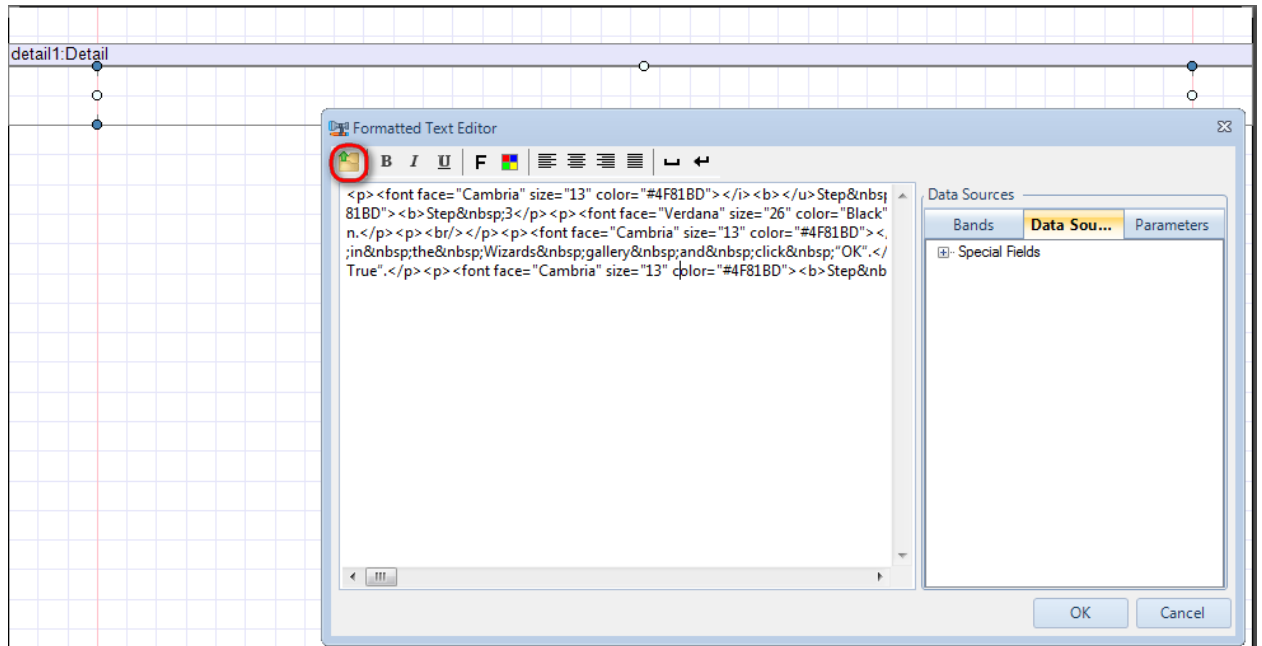
Step 10

Press button "AdvancedText" on the Insert tab in the group Text.

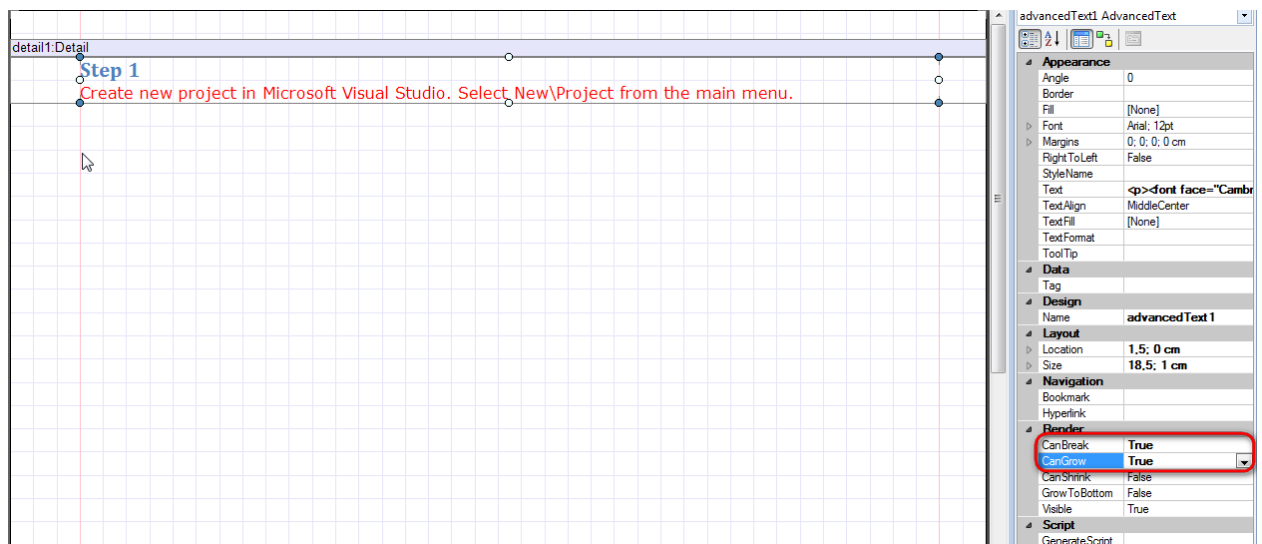


Click on the Detail band area to add AdvancedText element inside the Detail. Dragging the right border of the element make it fit page width. Double click on the AdvancedText band and open Formatted text editor.

Click "Open RTF document" button. Select RTF file containing some text.



On the property grid set CanBreak and CanGrow properties to "True".

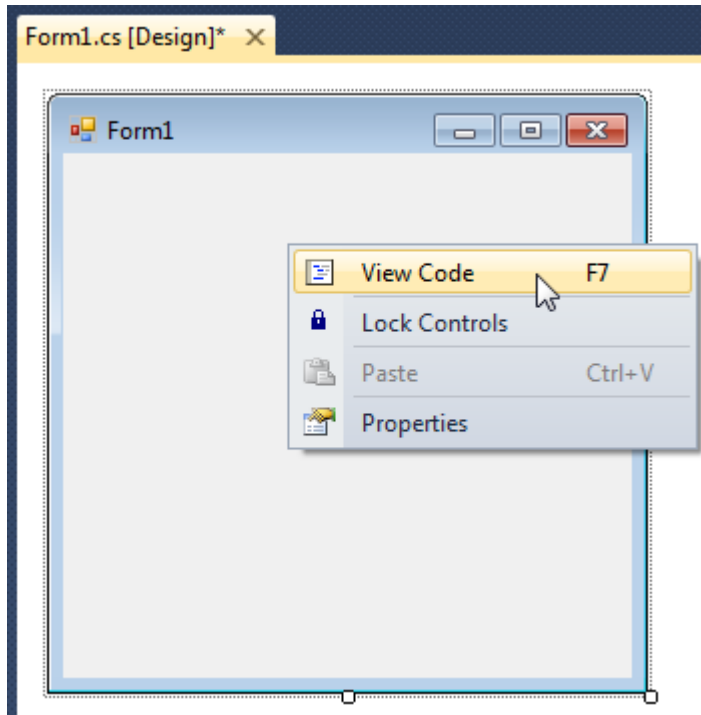


Step 11

Save report and close Report Designer.

Step 12

Right click on the form and select "View Code" in the context menu in order to view code.

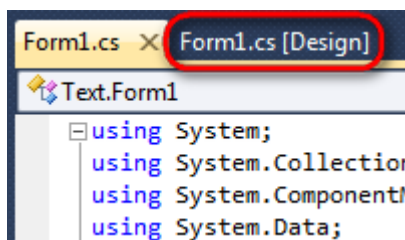


Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

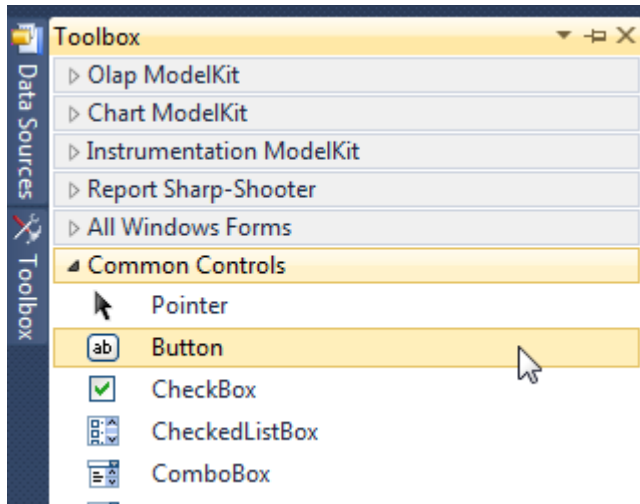
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 13

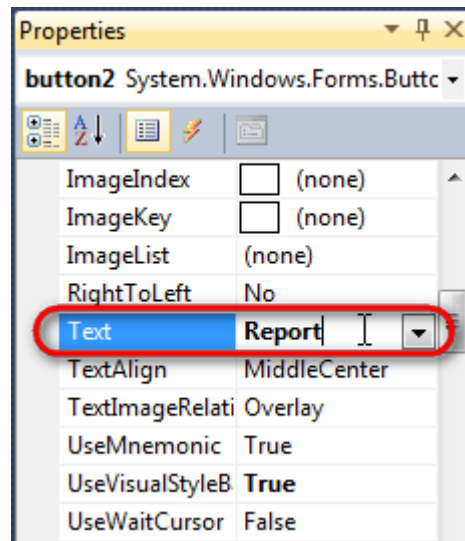
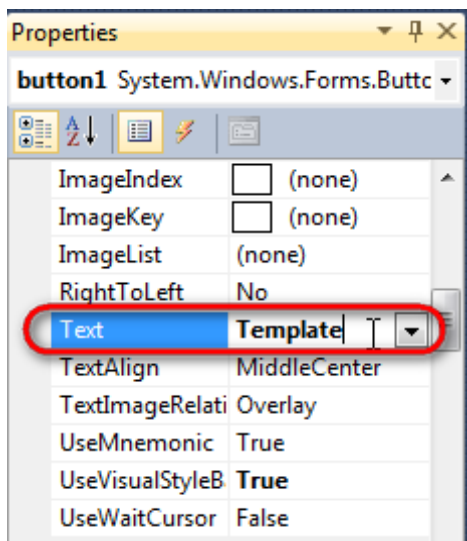
Go to the application form by clicking the "Form1.cs[Design]" tab.



Place two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the second one.



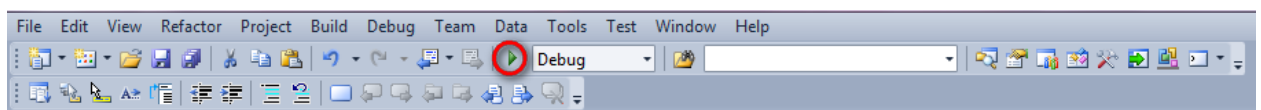
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

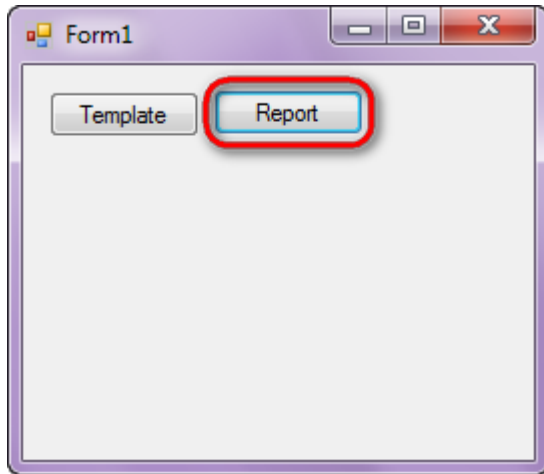
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 14

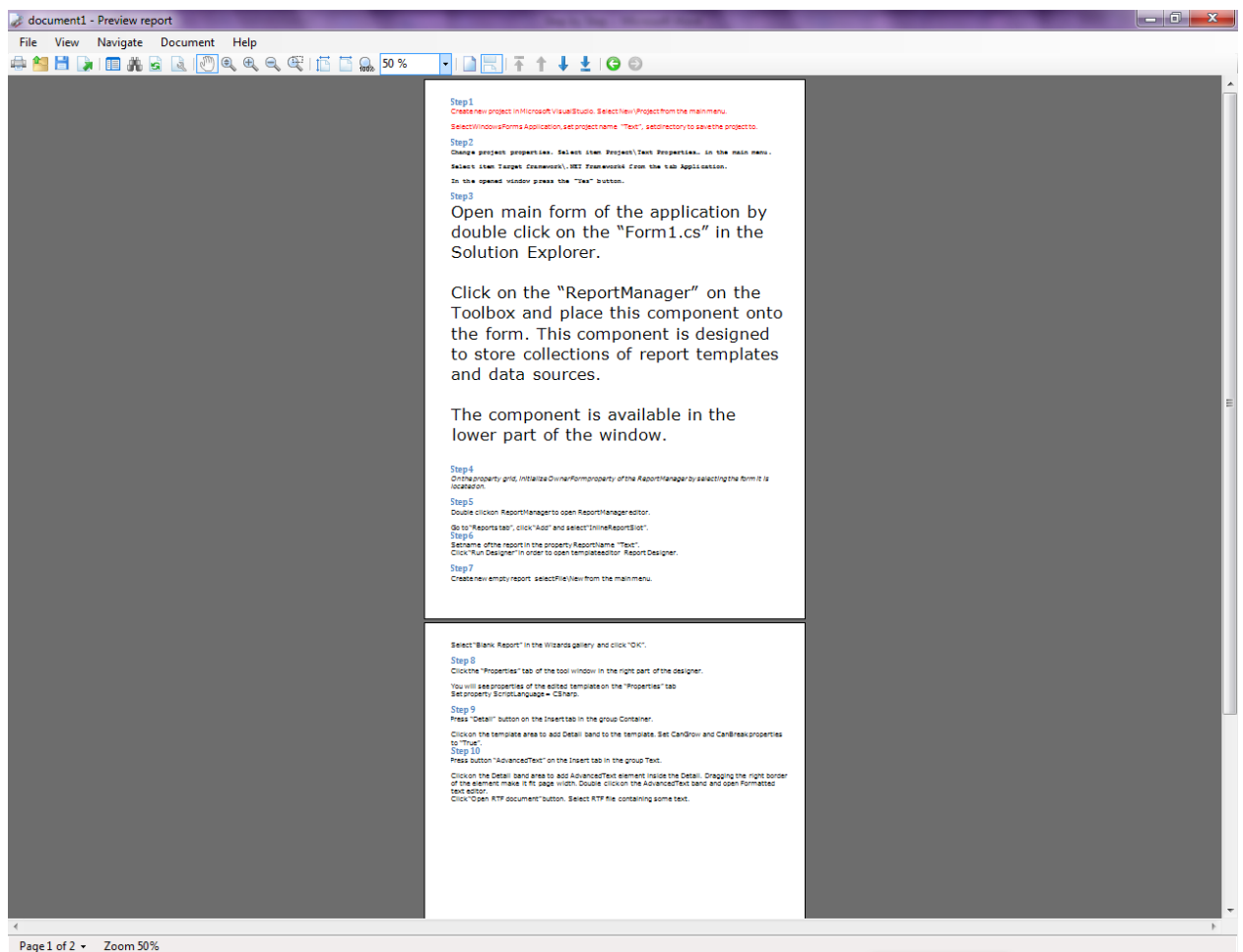
Click "Start Debugging" on the Visual Studio toolbar in order to run application.



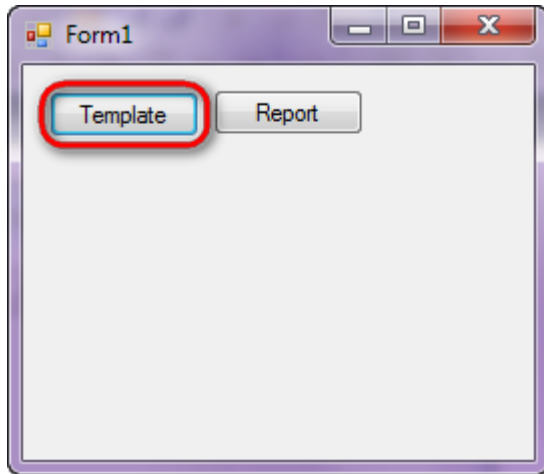
Click the "Report" button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



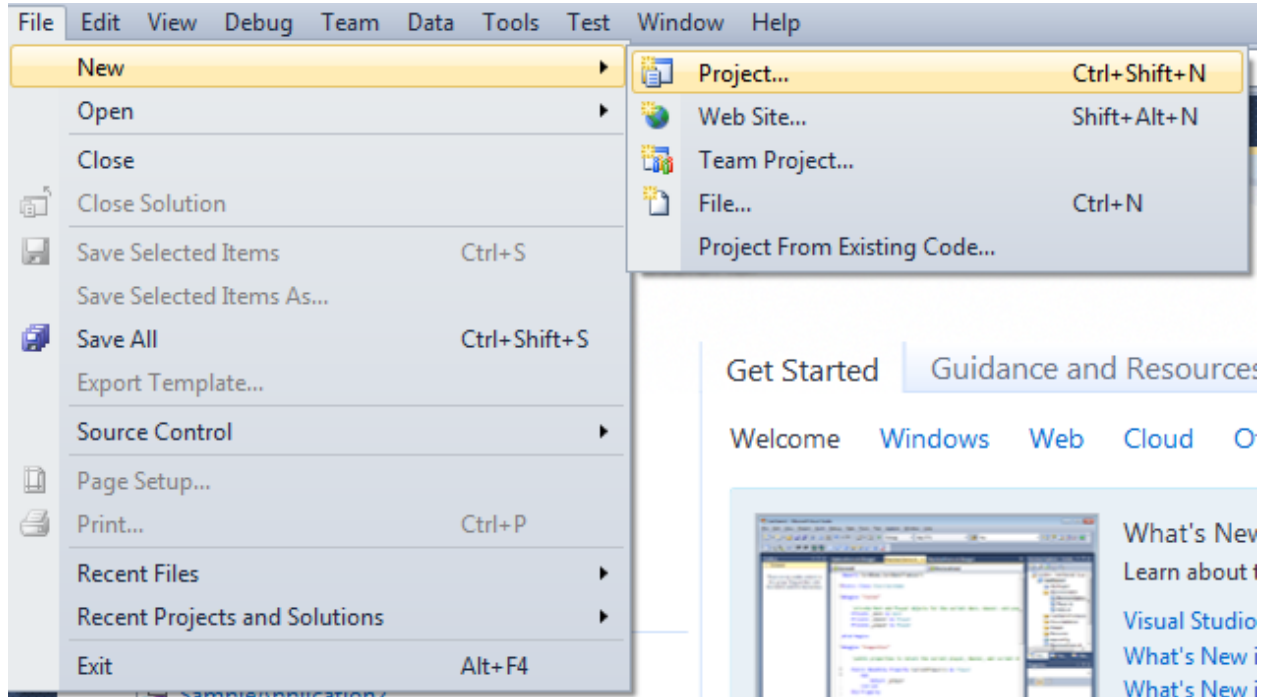


Watermark

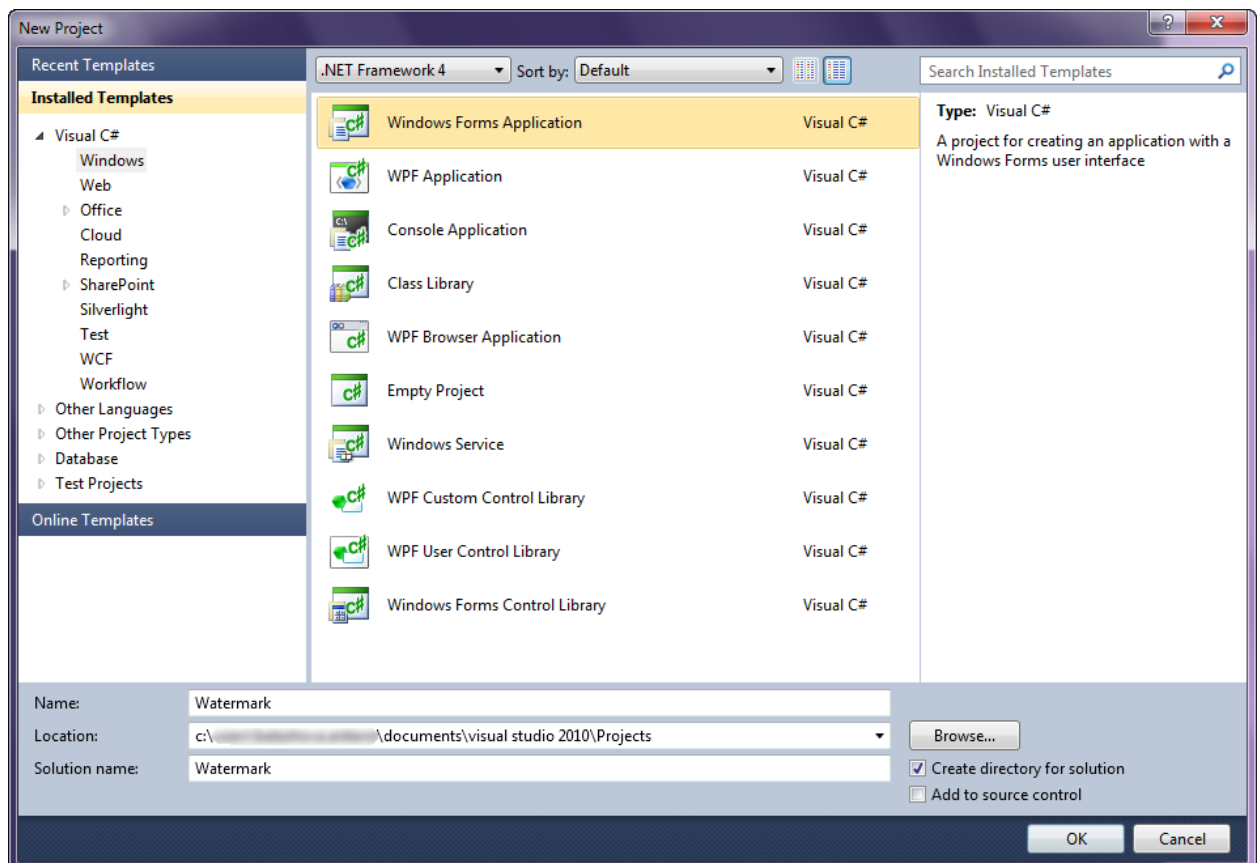
Template of a report containing main data that is located over the additional information – product logo and name.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

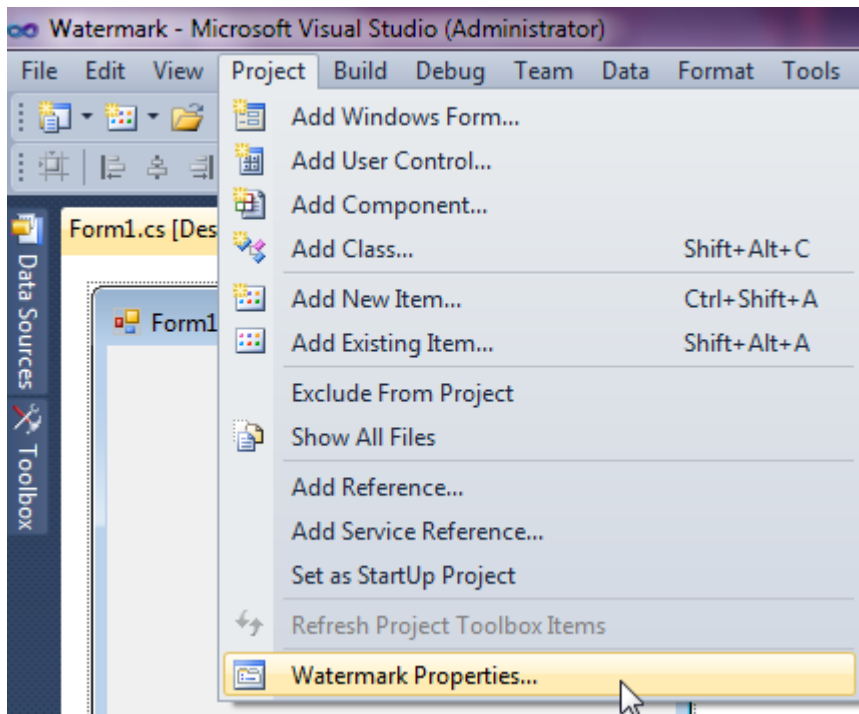


Select Windows Forms Application, set project name – “Watermark”, set directory to save the project to.

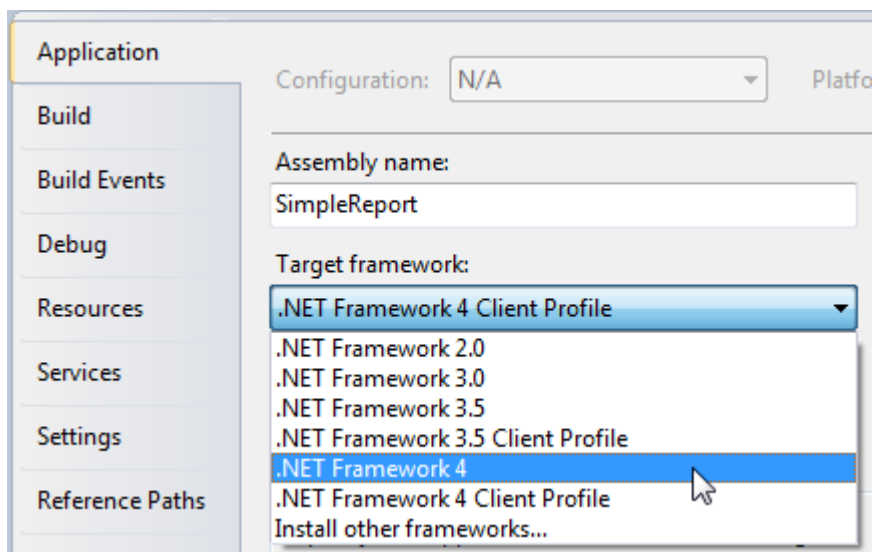


Step 2

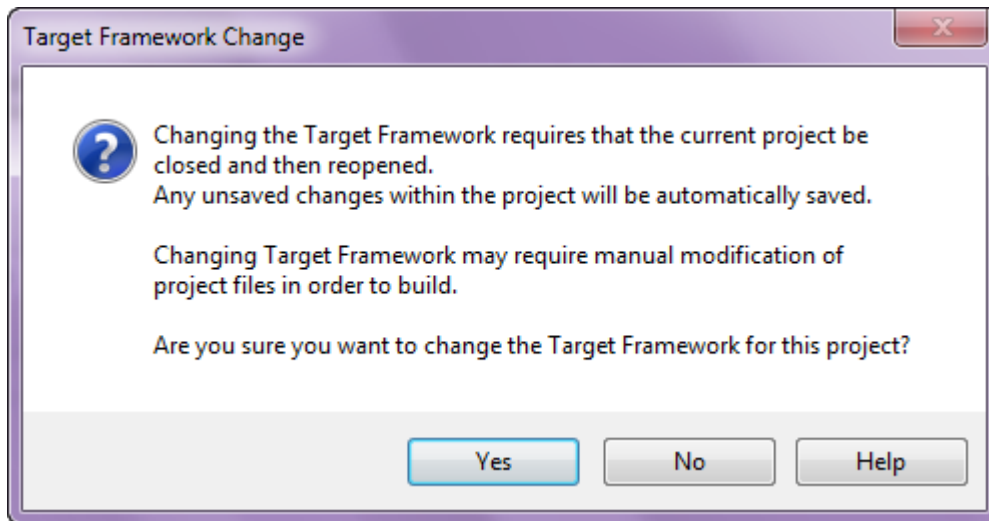
Change the project properties. Select the Project\Watermark Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

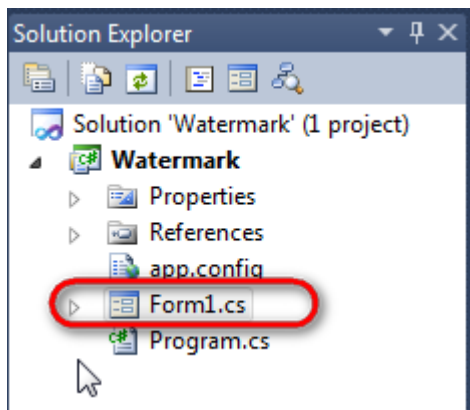


Press the "Yes" button in the opened window.

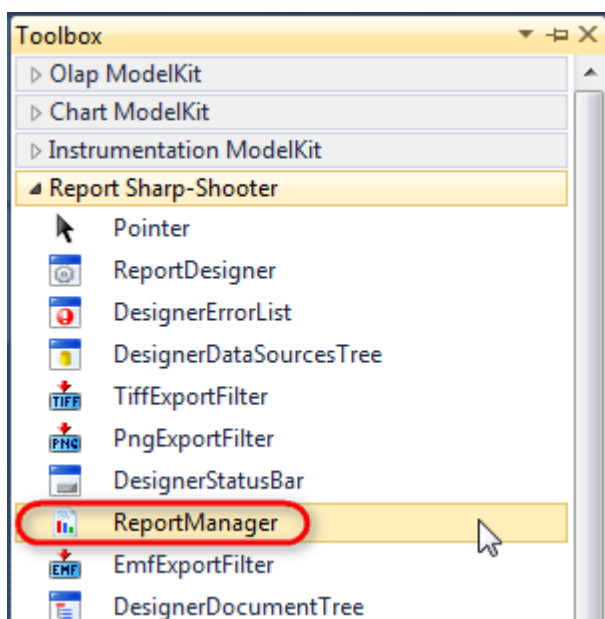


Step 3

Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

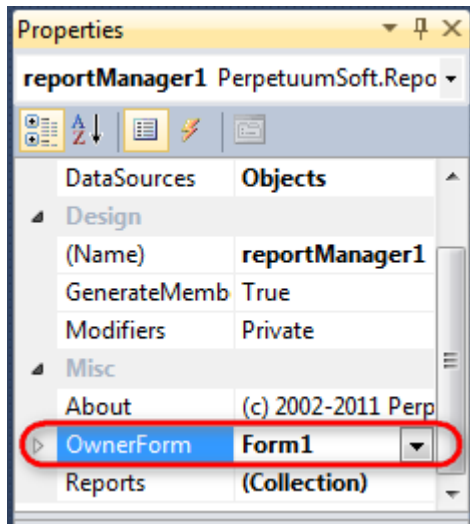


The component is available in the lower part of the window.



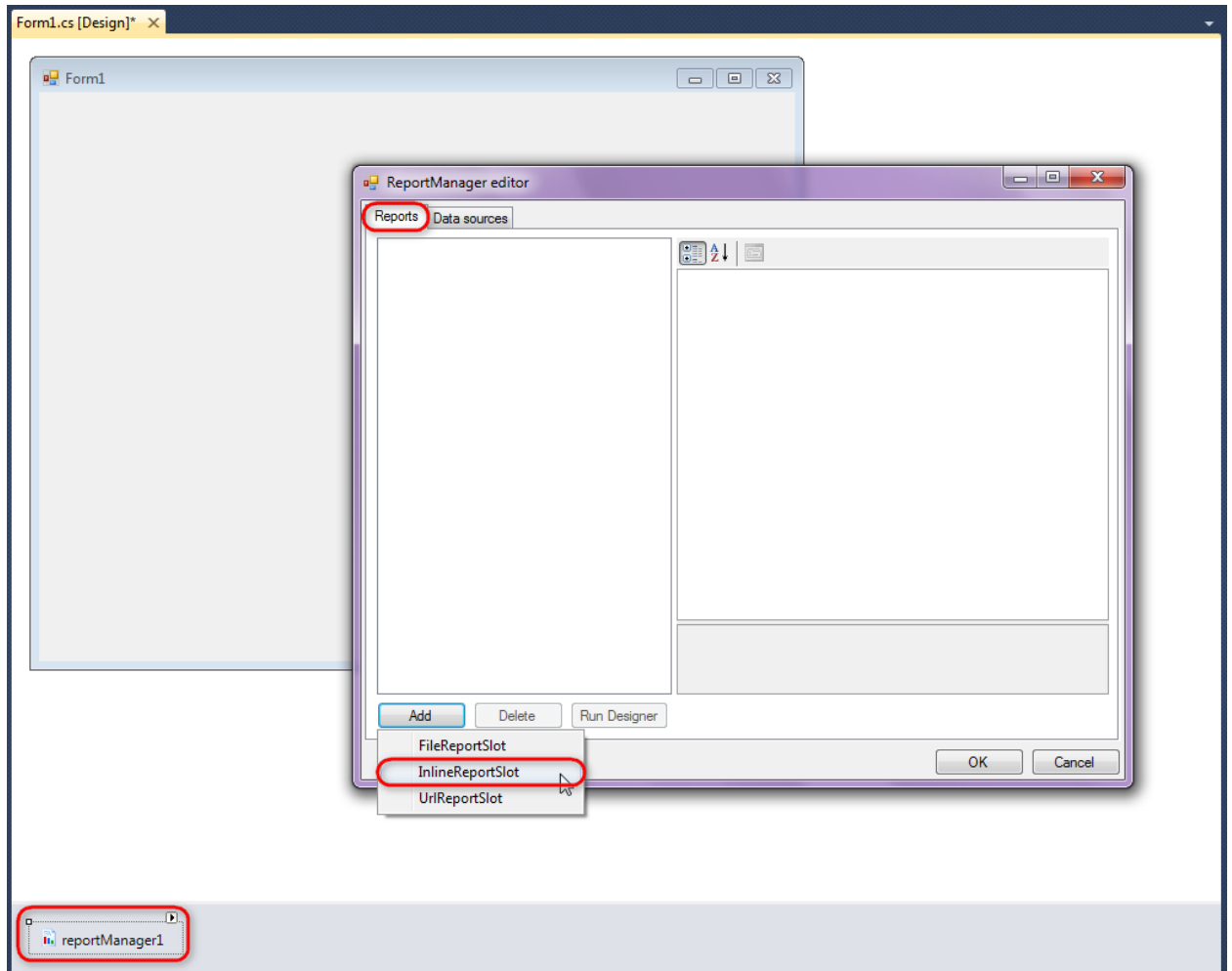
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

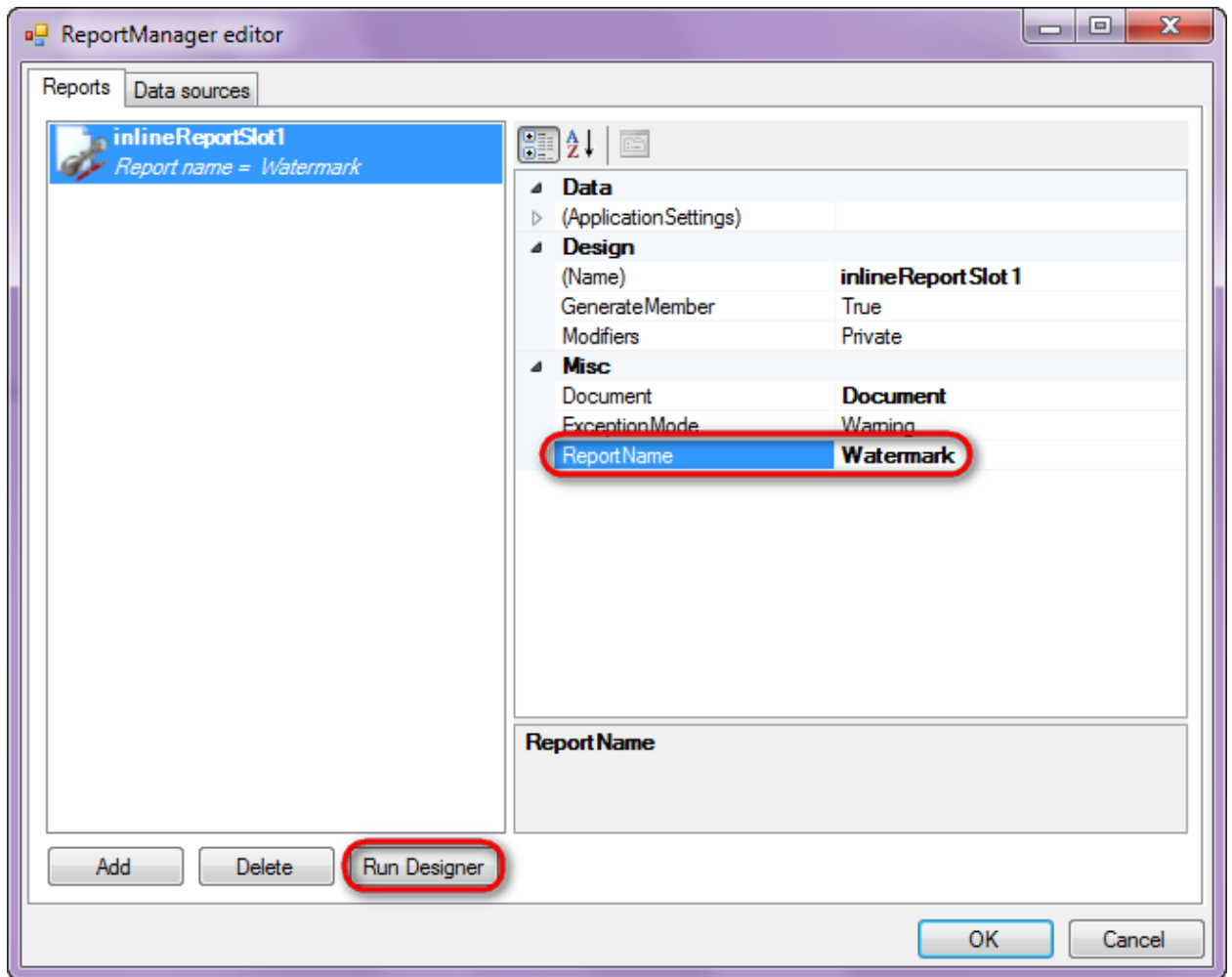


Go to "Reports tab", click "Add" and select "InlineReportSlot".

Step 6

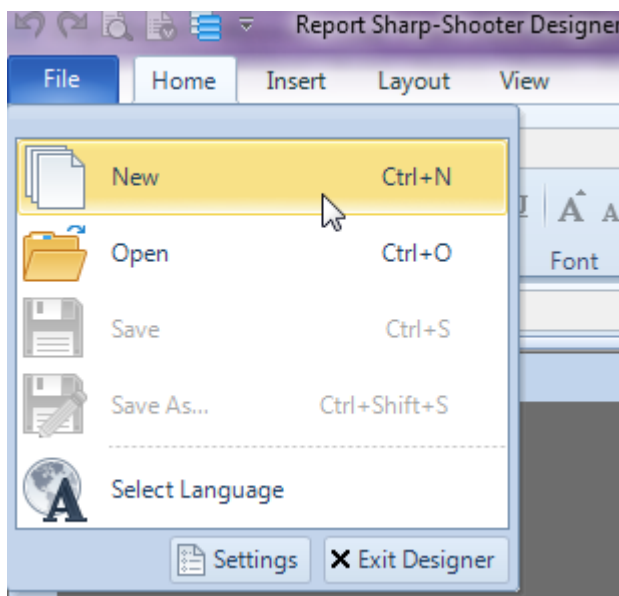
Set name of the report in the property ReportName - "Watermark".

Click "Run Designer" in order to open template editor - Report Designer.

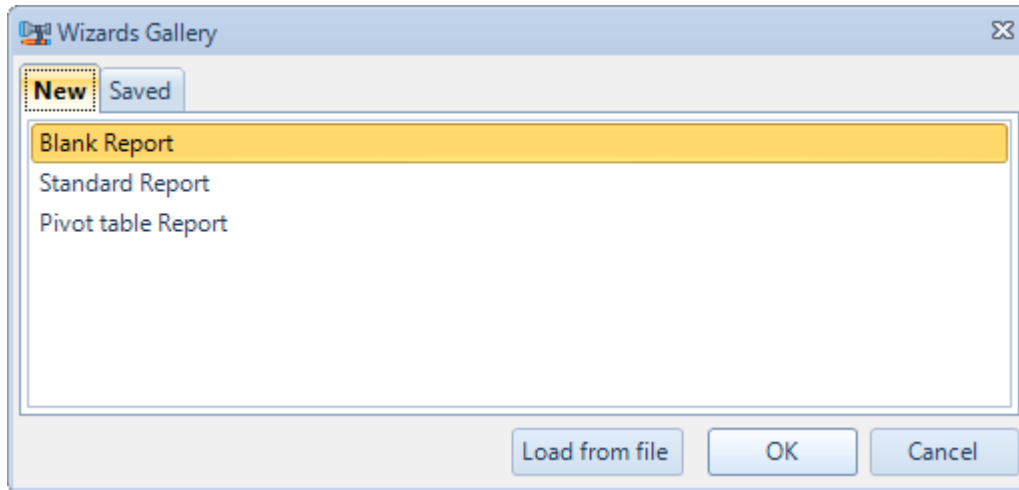


Step 7

Create new empty report – select File\New from the main menu.

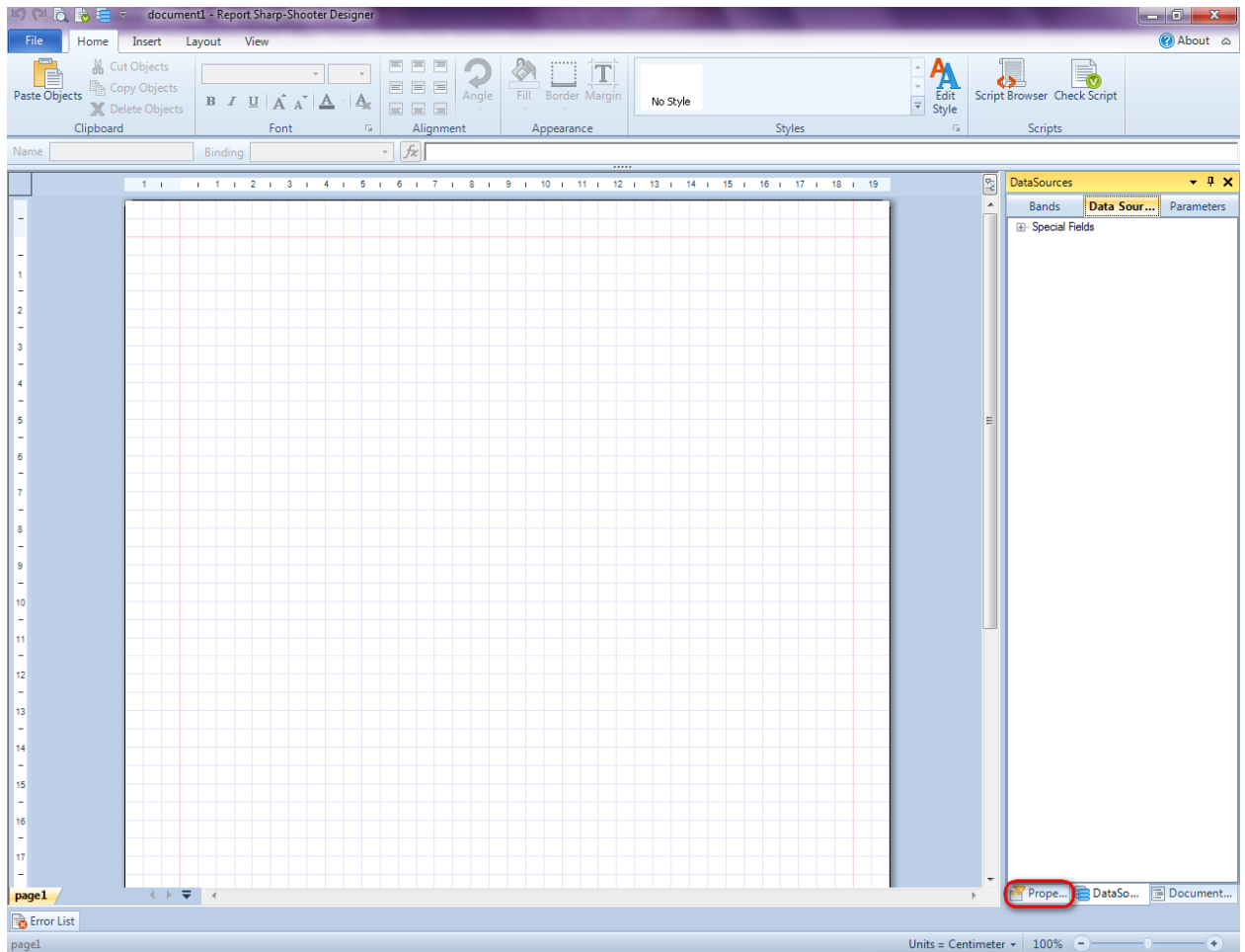


Select "Blank Report" in the Wizards Gallery and click "OK".

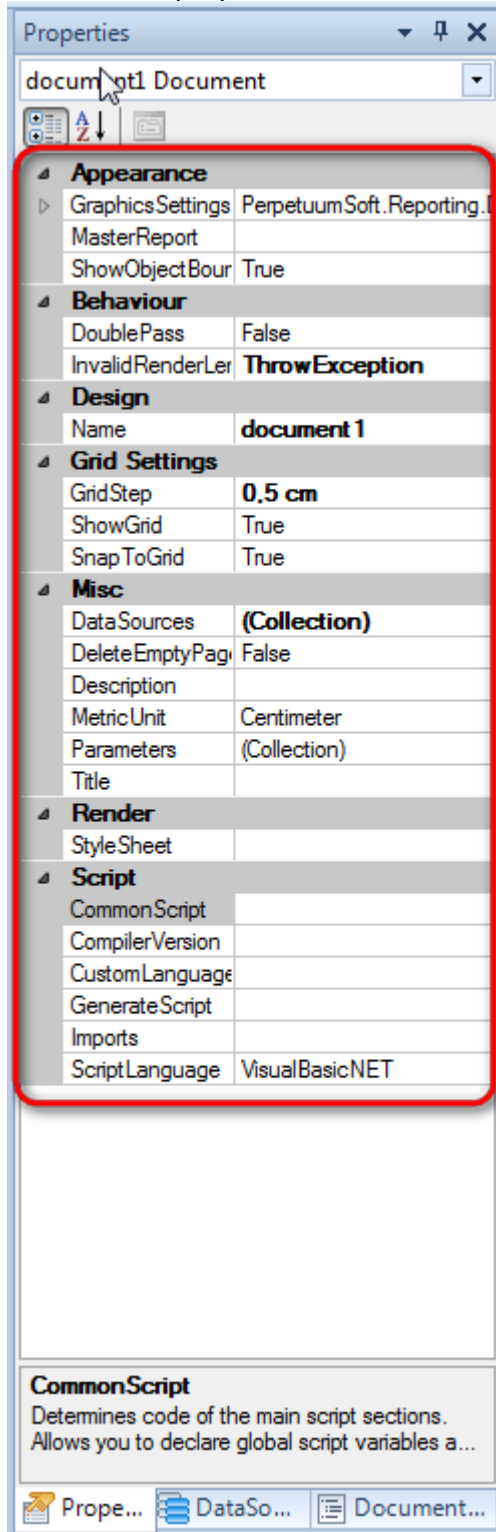


Step 8

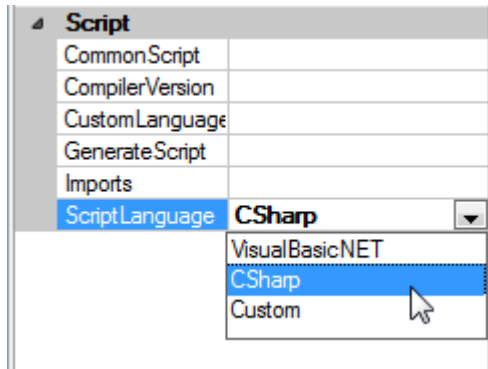
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

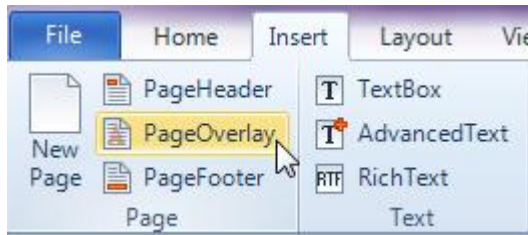


Set property ScriptLanguage = CSharp.



Step 9

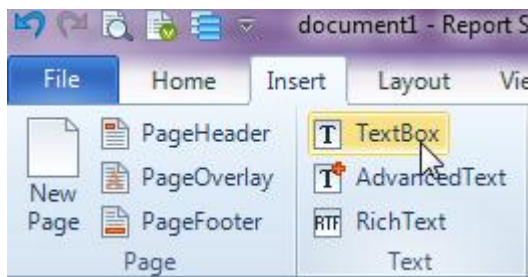
Press "PageOverlay" button on the Insert tab in the group Page.




Click on the template area to add PageOverlay band.

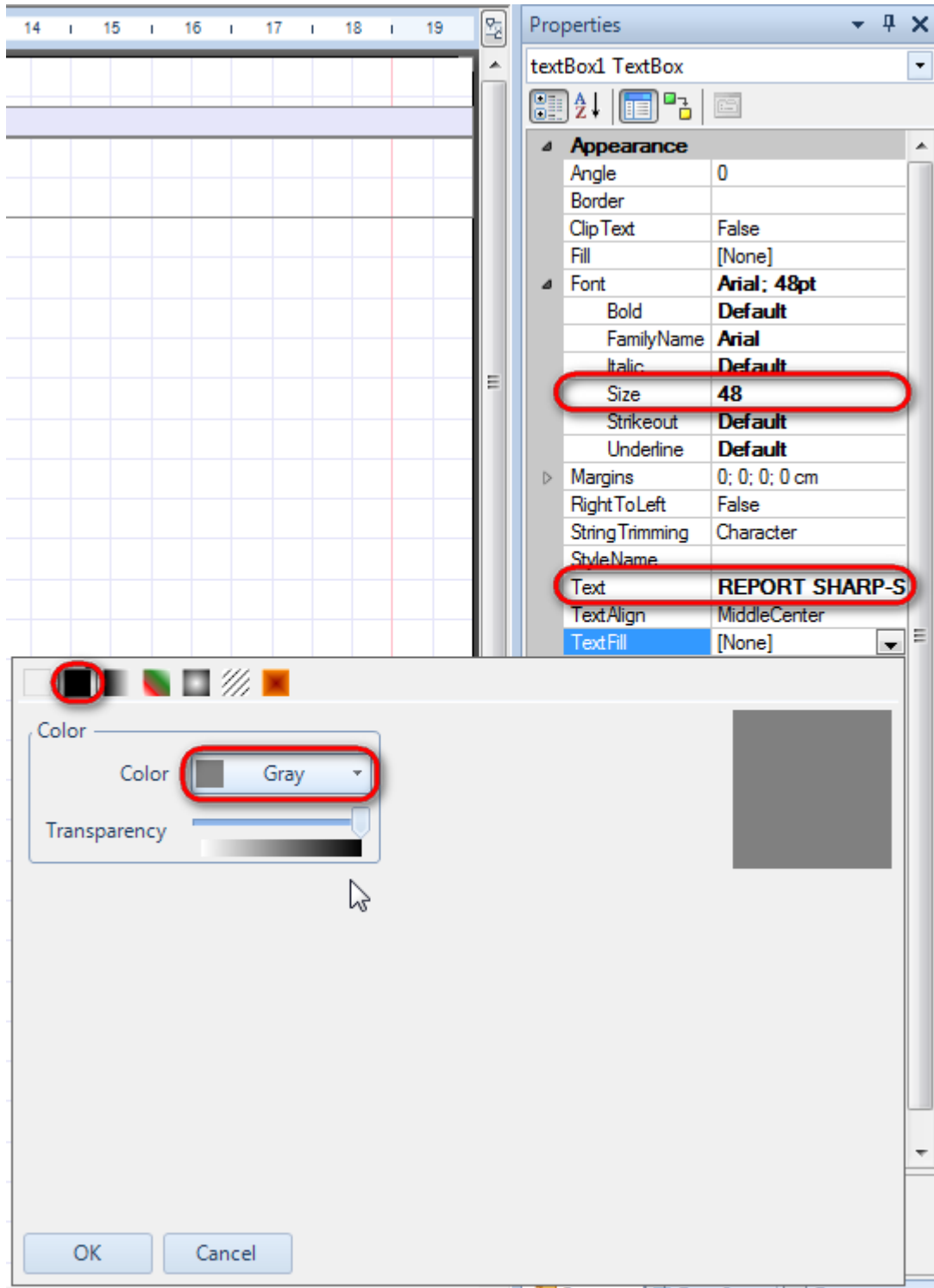
Step 10

Press button "TextBox" on the Insert tab in the group Text.



Click on the PageOverlay band area to add TextBox inside PageOverlay.

Set property Text = REPORT SHARP-SHOOTER. Set property Font.Size = 48. Select TextFill property, click button  select SolidFill, Color = Grey.



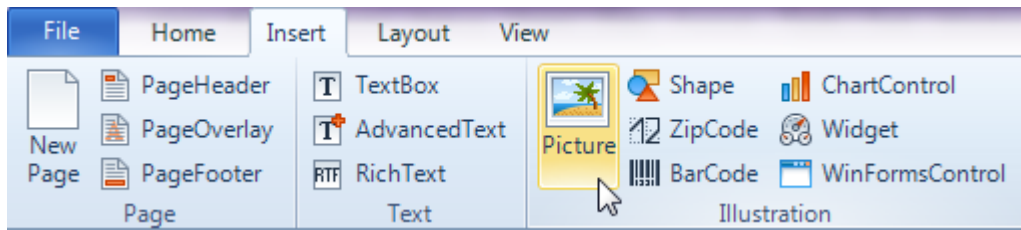
Change size of the PageOverlay band and the TextBox element so that the text is visible.





Step 11

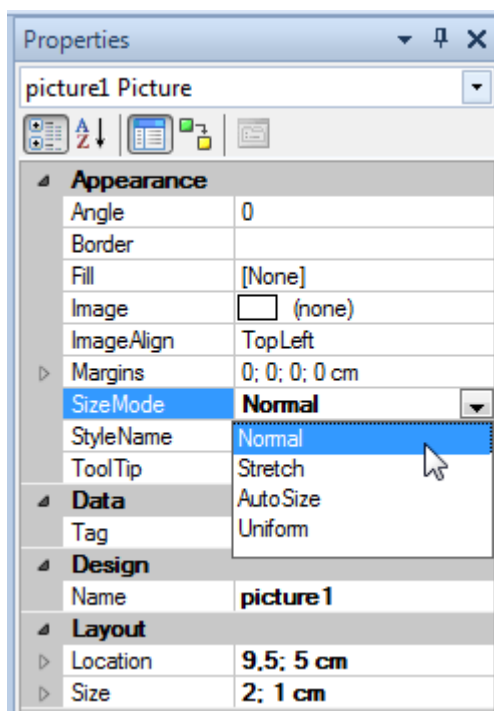
Press button "Picture" on the Insert tab in the group Illustration.



Click on the PageOverlay band area to add Picture element inside PageOverlay.

Double click on the Picture element to open dialog setting path to the picture. Select a picture and click "Open".

Set property SizeMode = Normal.



Change size of the PageOverlay band and Picture element so that the picture is visible.

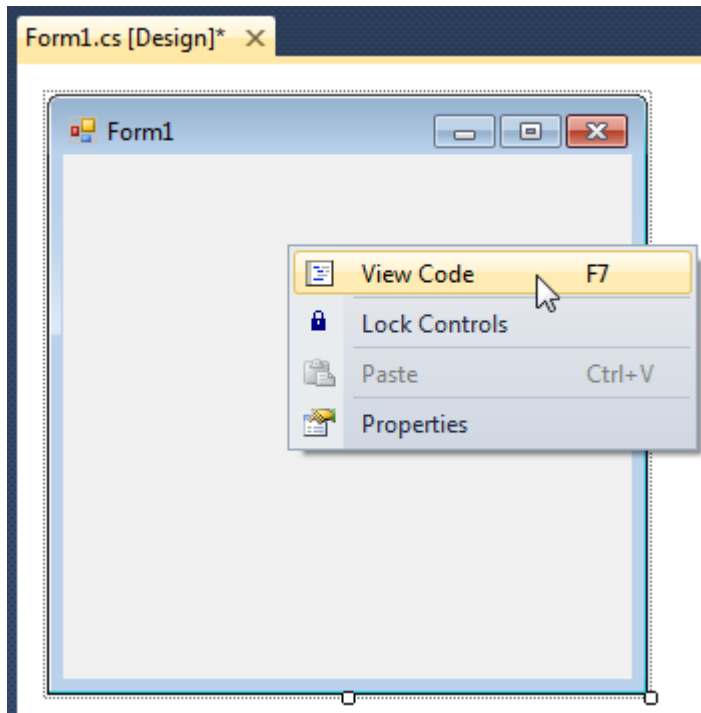


Step 12

Save template, close Report Designer

Step 13

Right click on the form and select "View Code" in the context menu in order to view code.

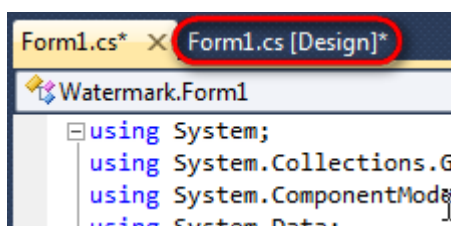


Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

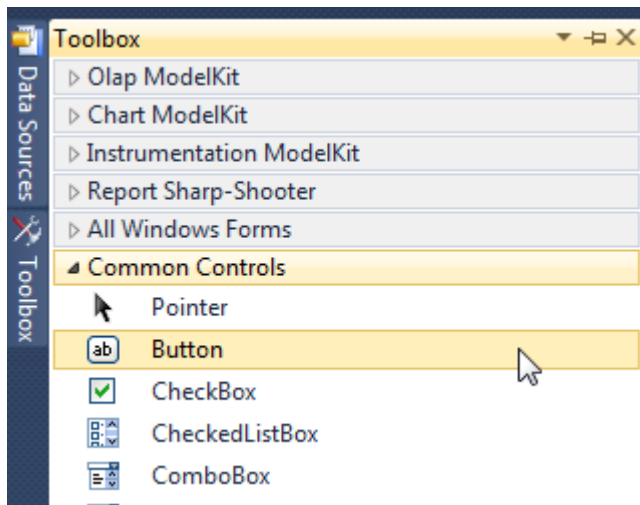
Step 14

Get back to the application form by clicking the "Form1.cs[Design]" tab.

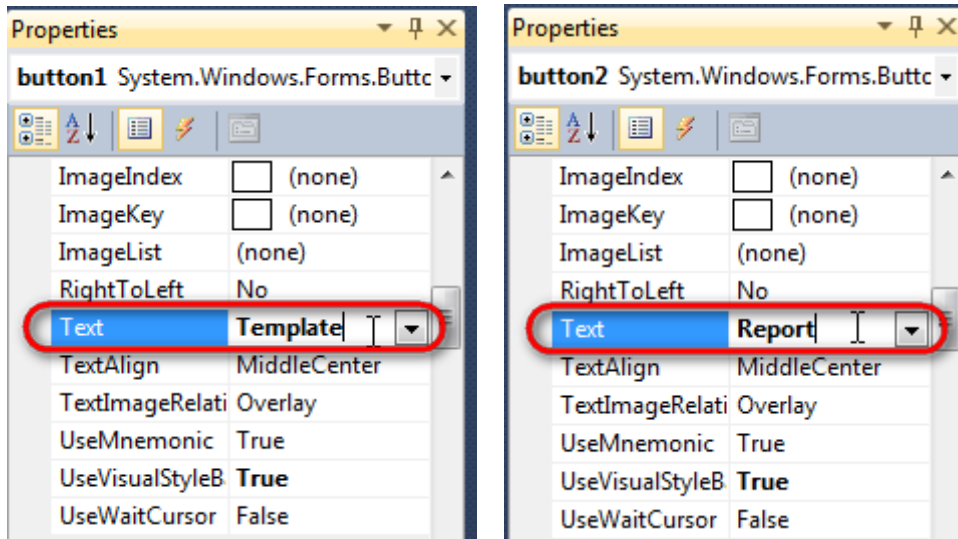




Place two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the second one.



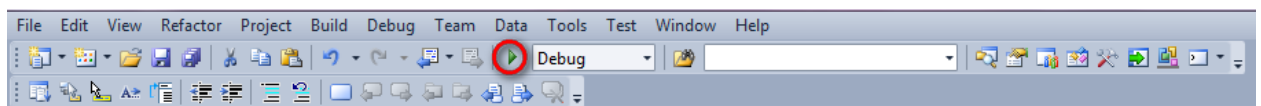
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

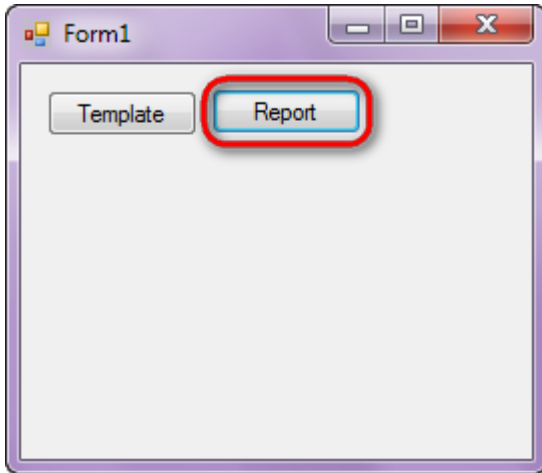
Step 15

Click "Start Debugging" on the Visual Studio toolbar in order to run application.

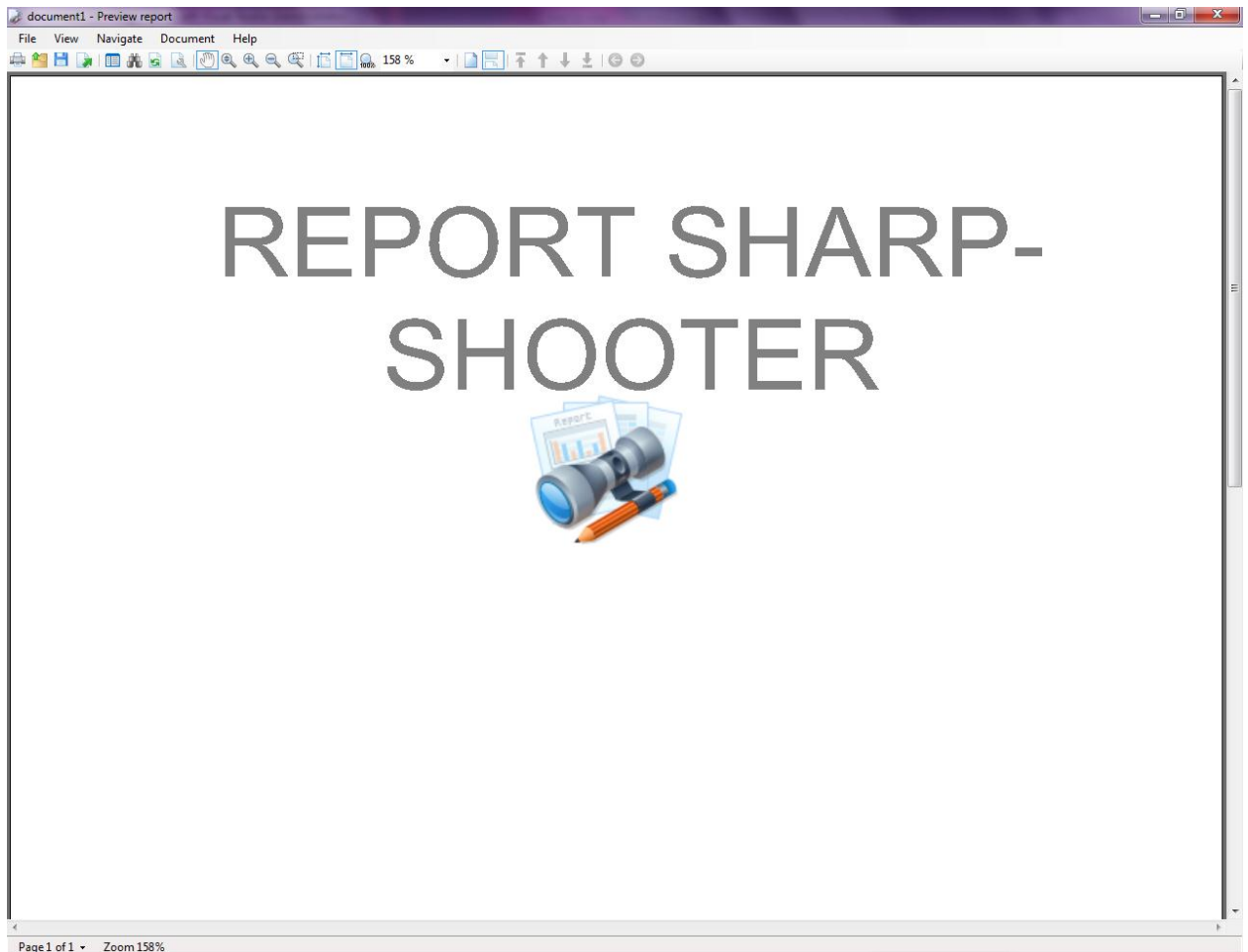




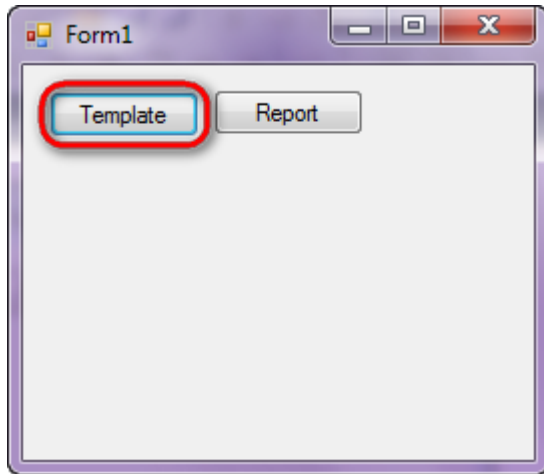
Click the "Report" button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar sample in the Samples Center is Reports\Special features\Watermarks.

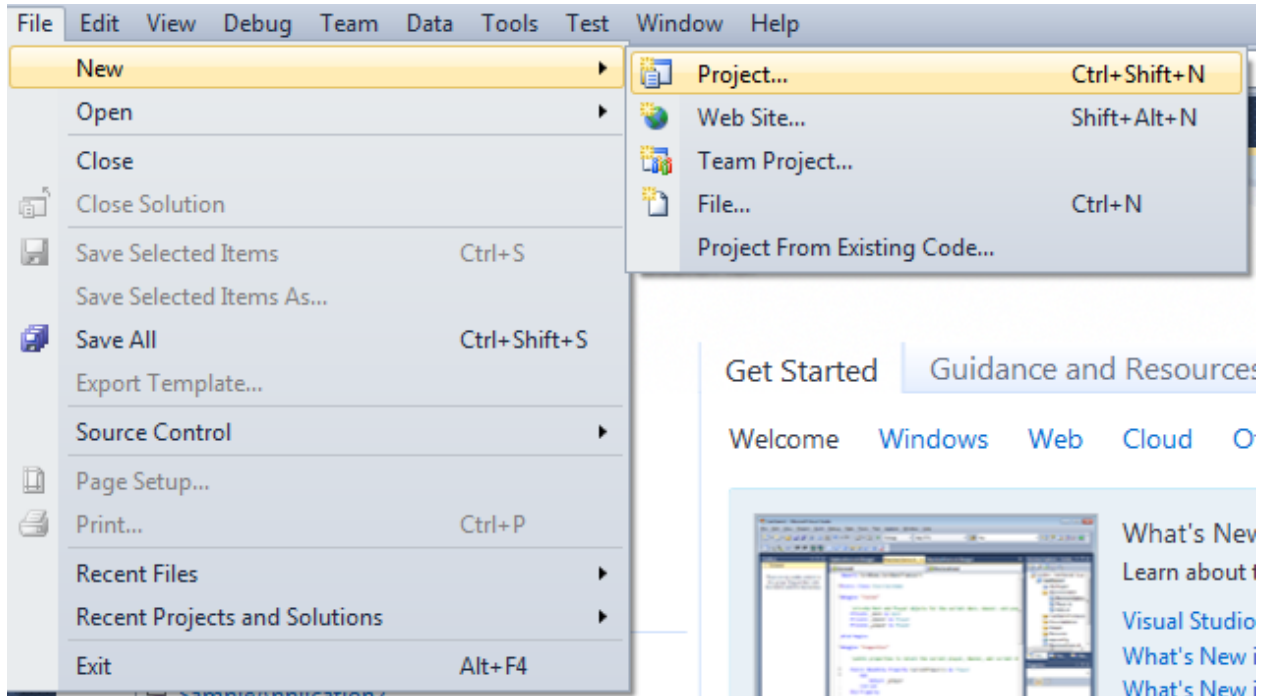


Master-Detail Report

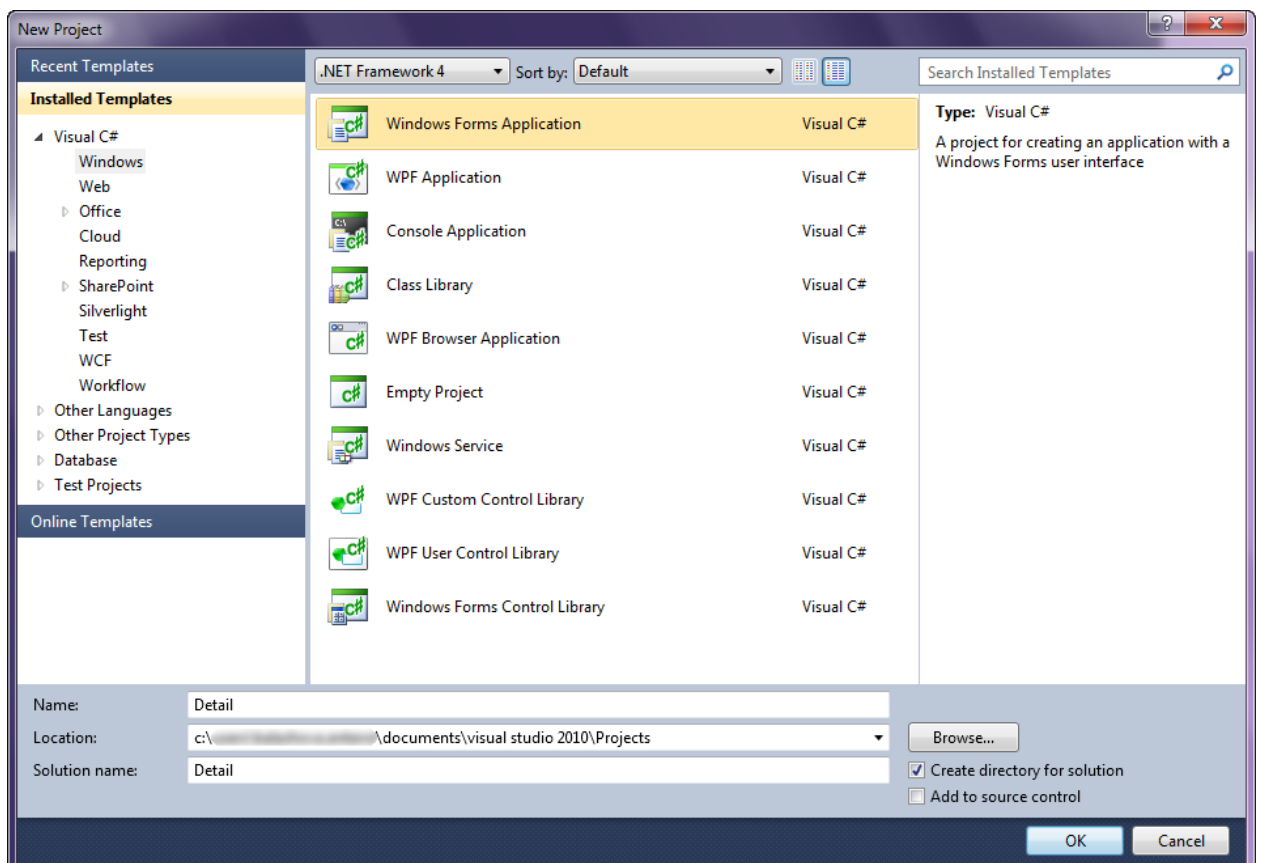
Template of a report containing a list of customers and a list of orders for every customer.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.



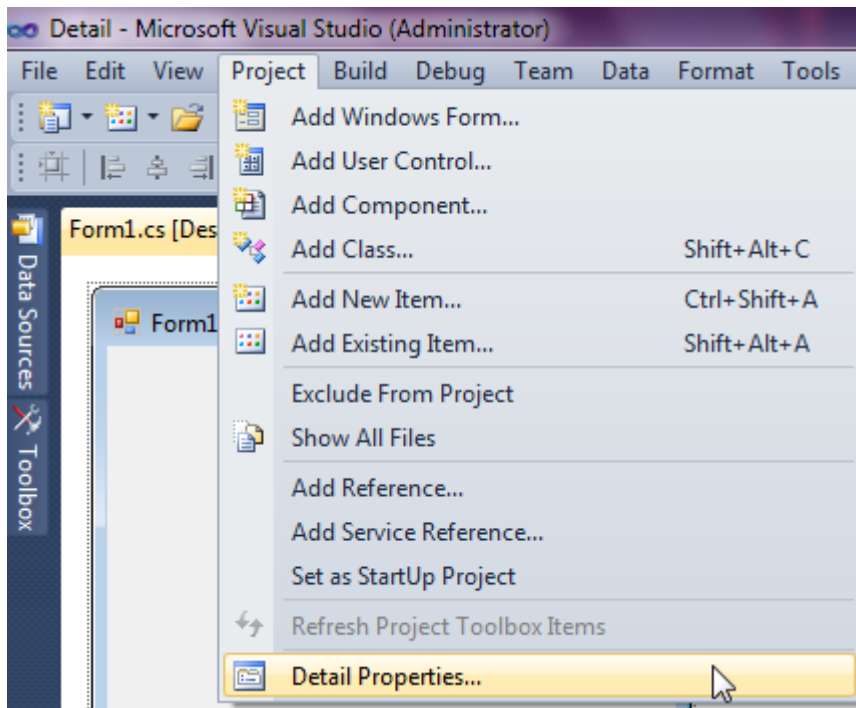
Select Windows Forms Application, set project name – “Detail”, set directory to save the project to.



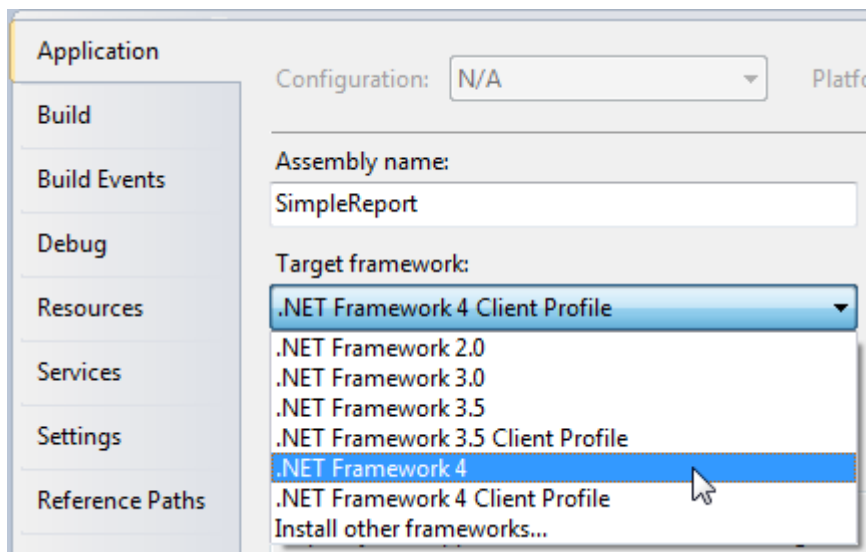


Step 2

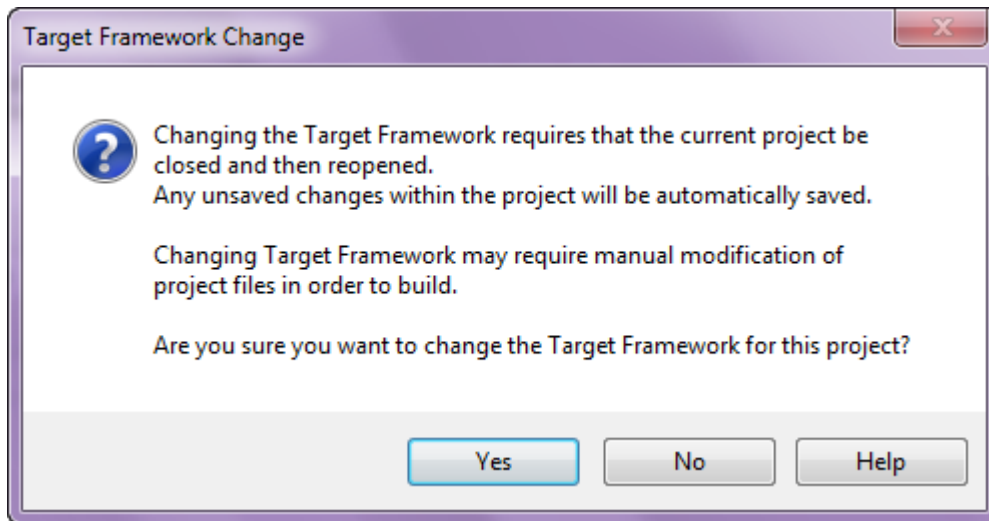
Change the project properties. Select the Project\Detail Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

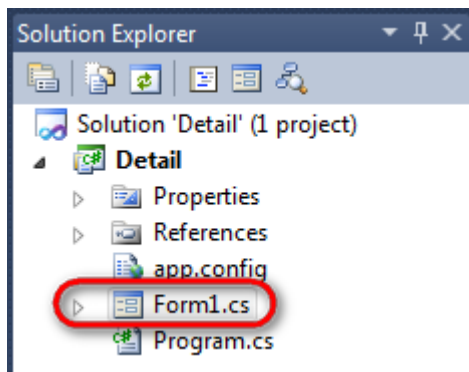


Press the "Yes" button in the opened window.

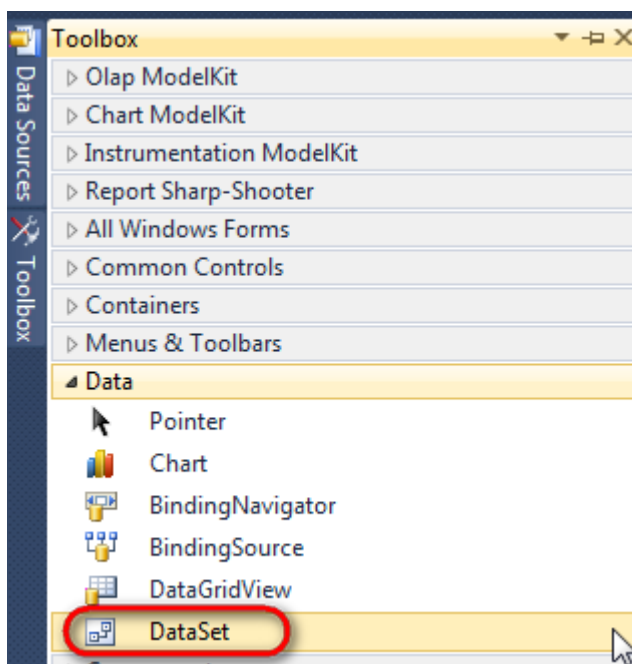


Step 3

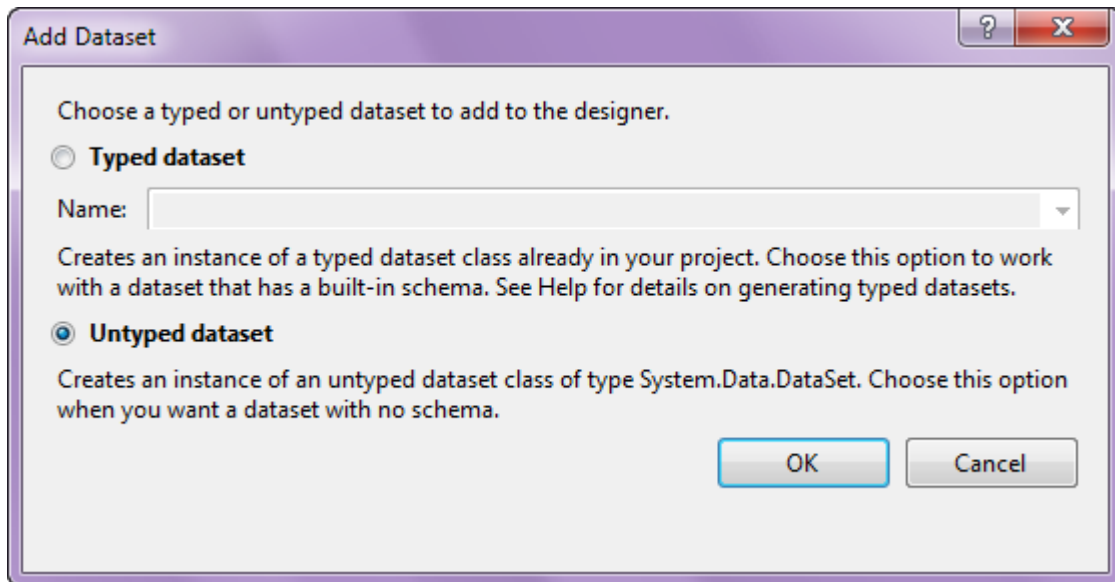
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



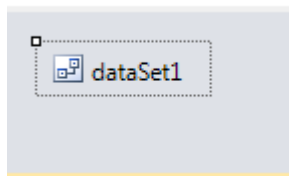
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

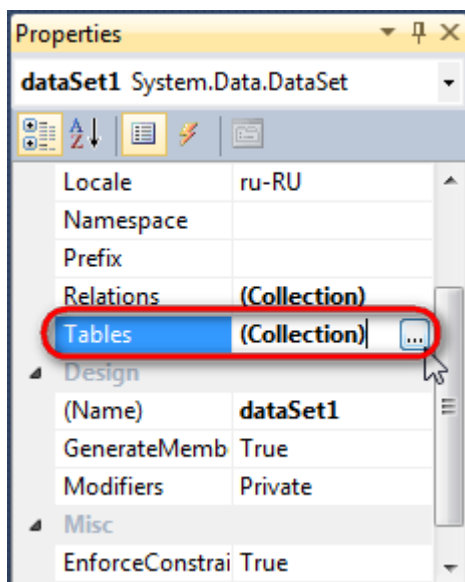


The component is available in the lower part of the window.

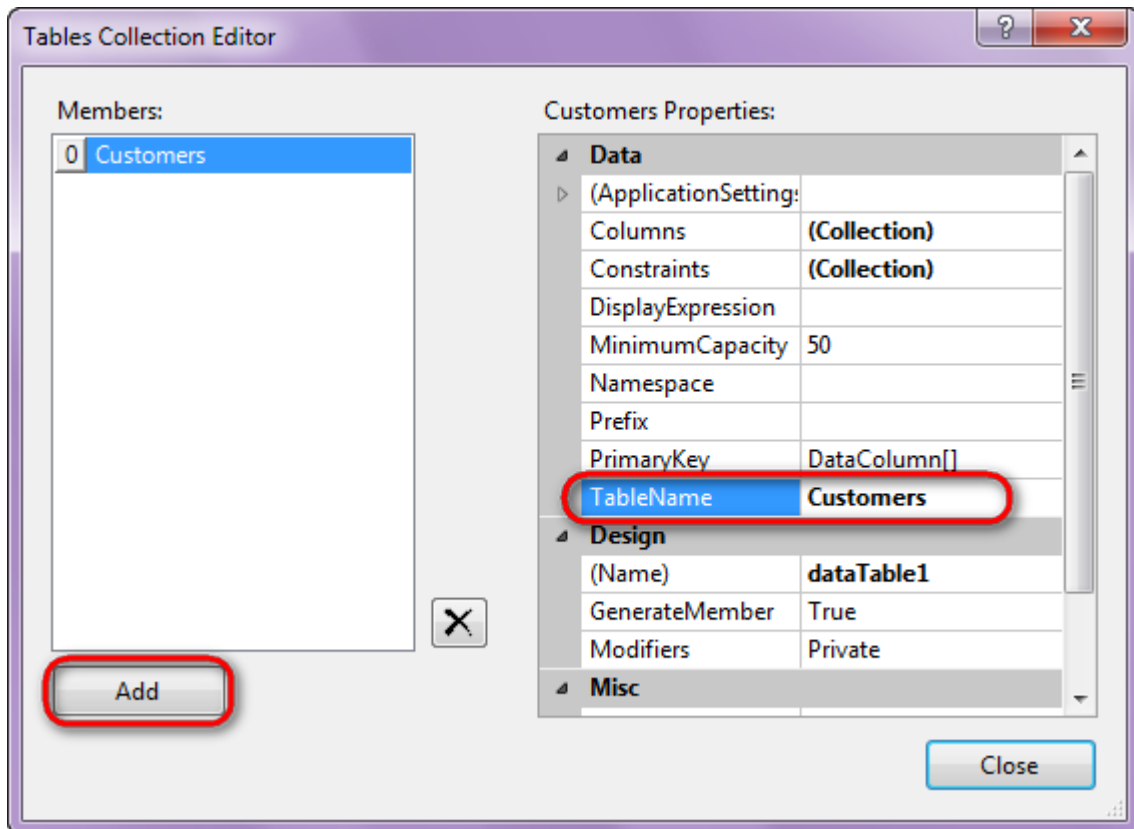


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

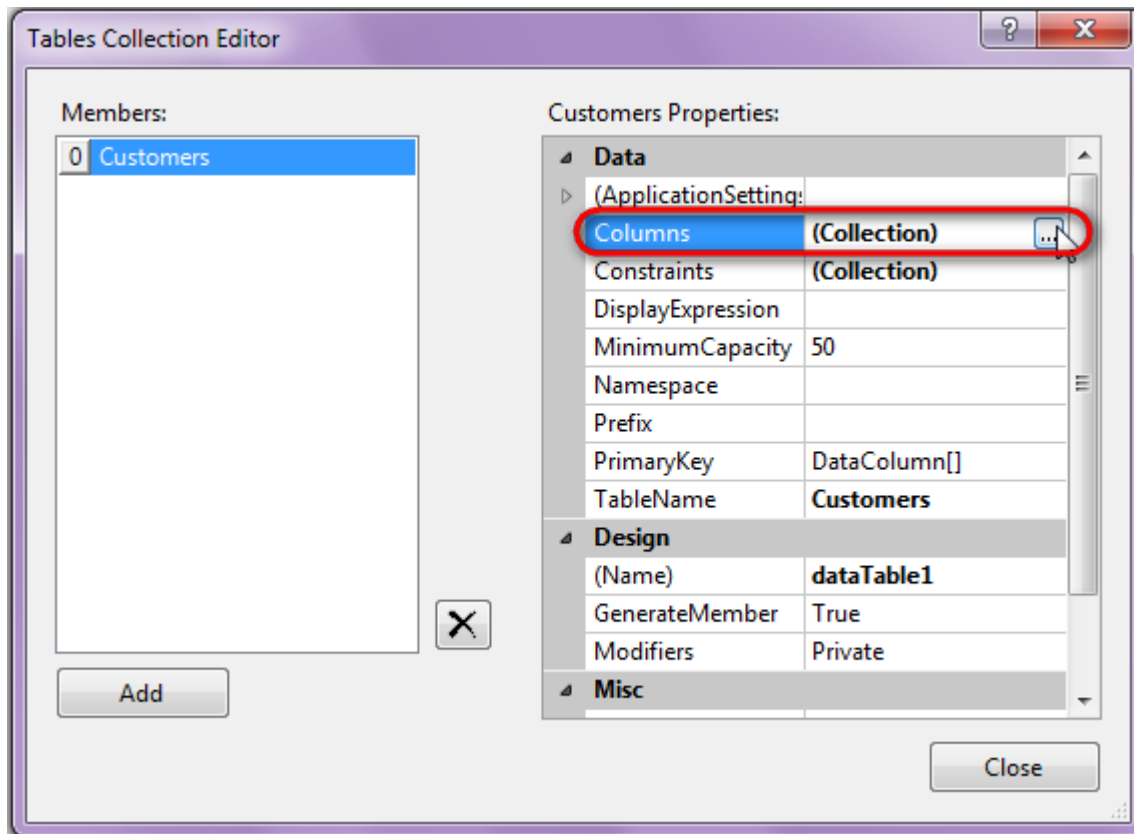


Click "Add" in order to add table. Set property TableName = Customers.

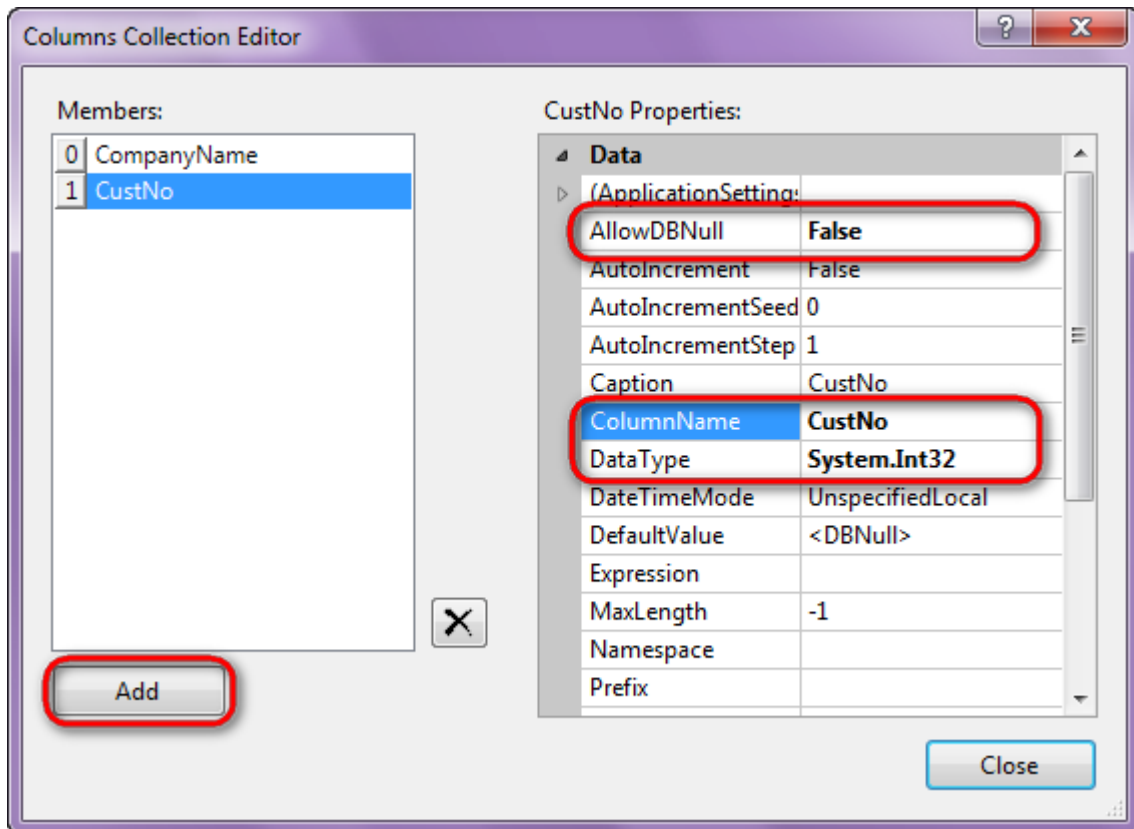


Step 5

Select Columns property, click button  in order to open property editor.

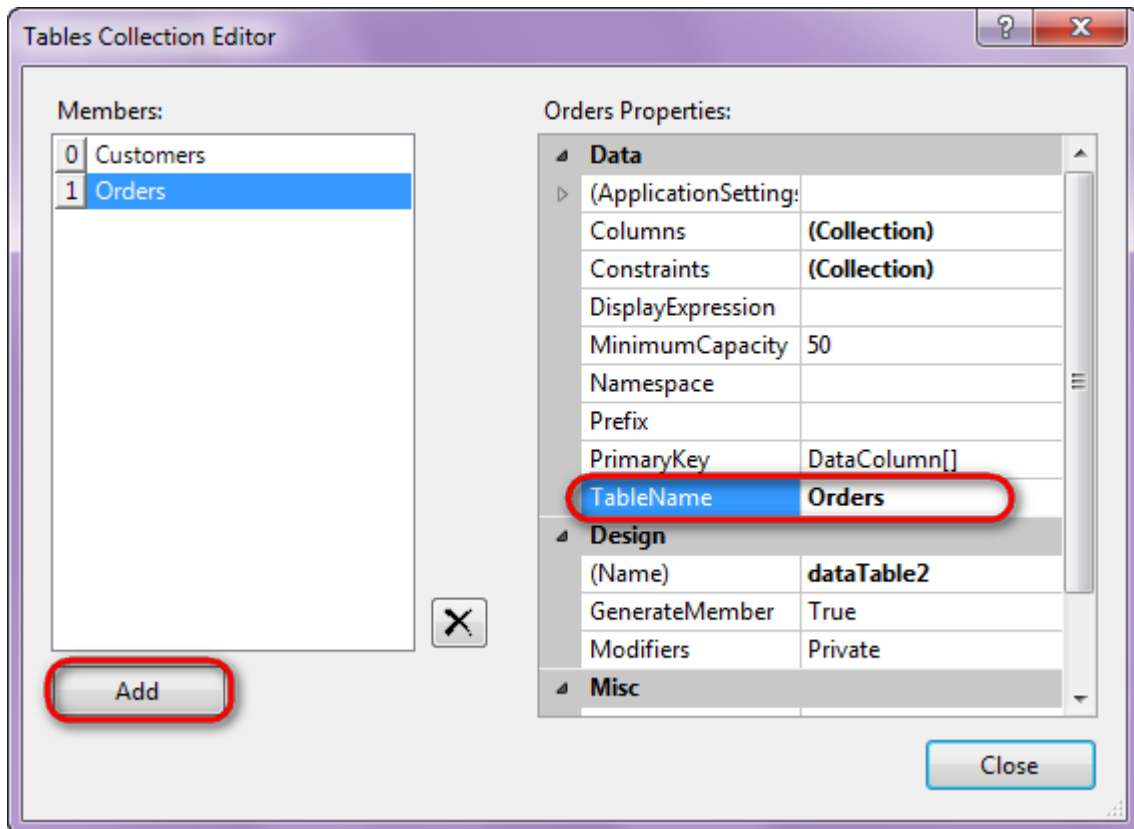


Click "Add" to add a new column. Add two columns. Set ColumnName property to "CompanyName", "CustNo". Set DataType = System.Int32, AllowDBNull = False for the CustNo column.



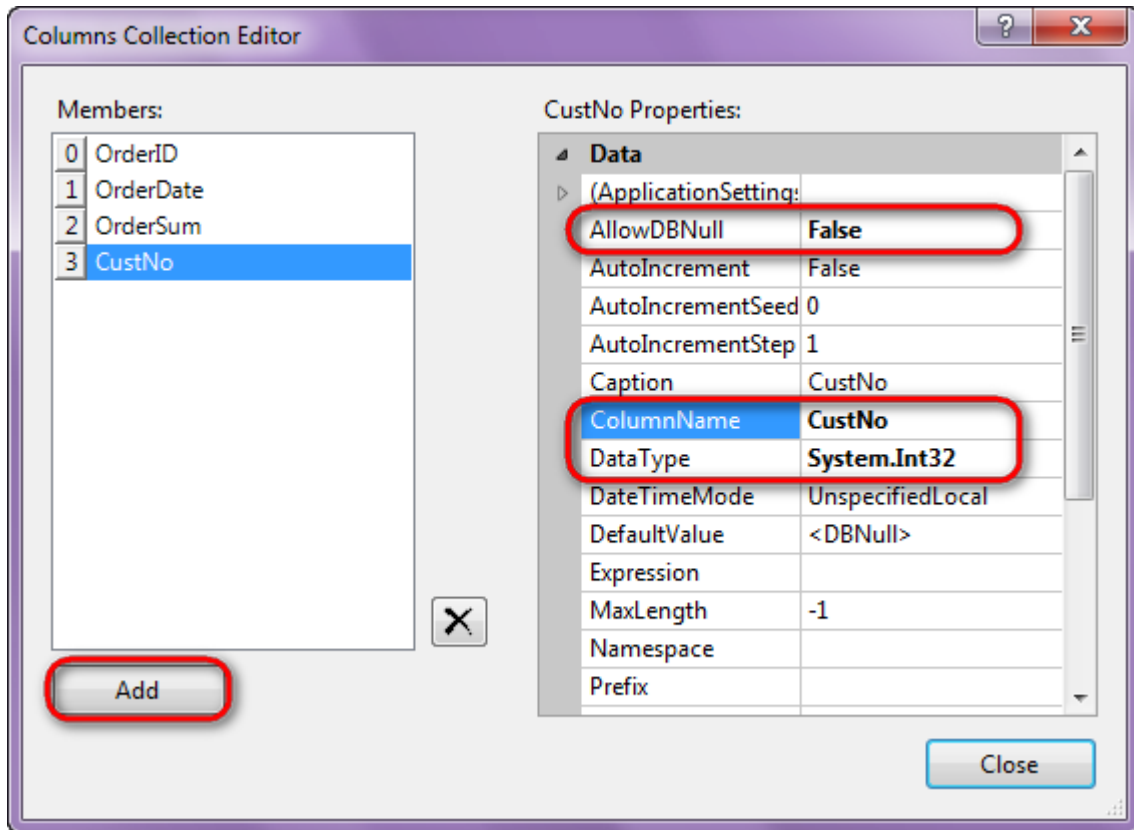
Step 6

In the Tables Collection Editor click "Add" in order to add one more table. Set TableName property to "Orders".




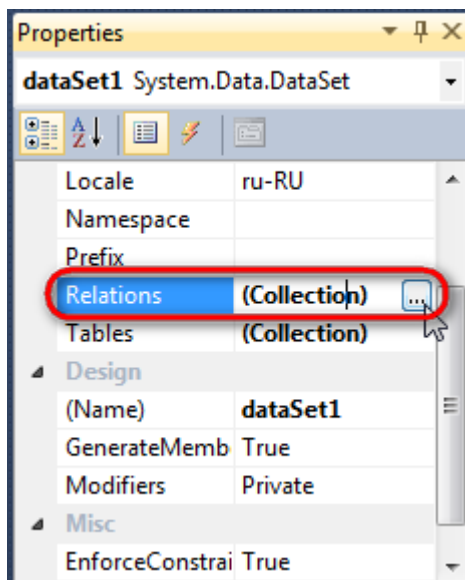


Open Columns property editor. Add four columns, set ColumnName property to "OrderID", "OrderDate", "OrderSum", "CustNo". Set DataType = System.Int32, AllowDBNull = False for the CustNo column.

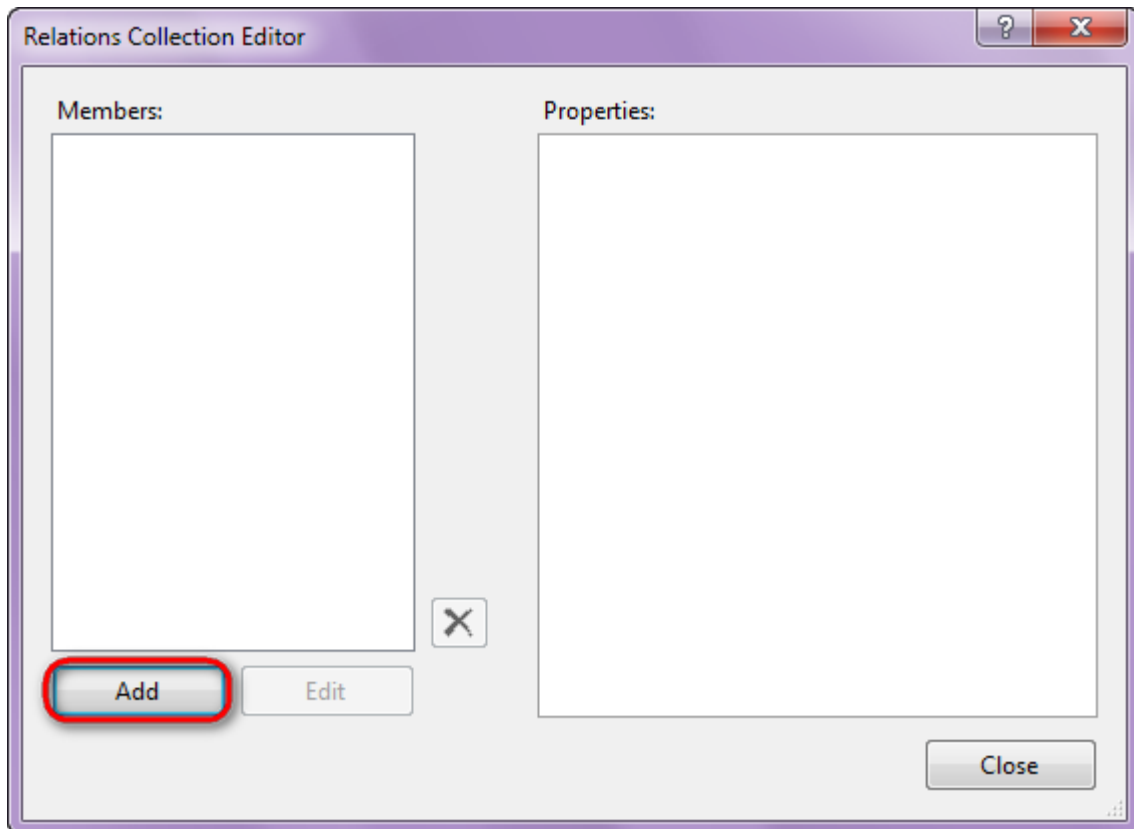


Step 7

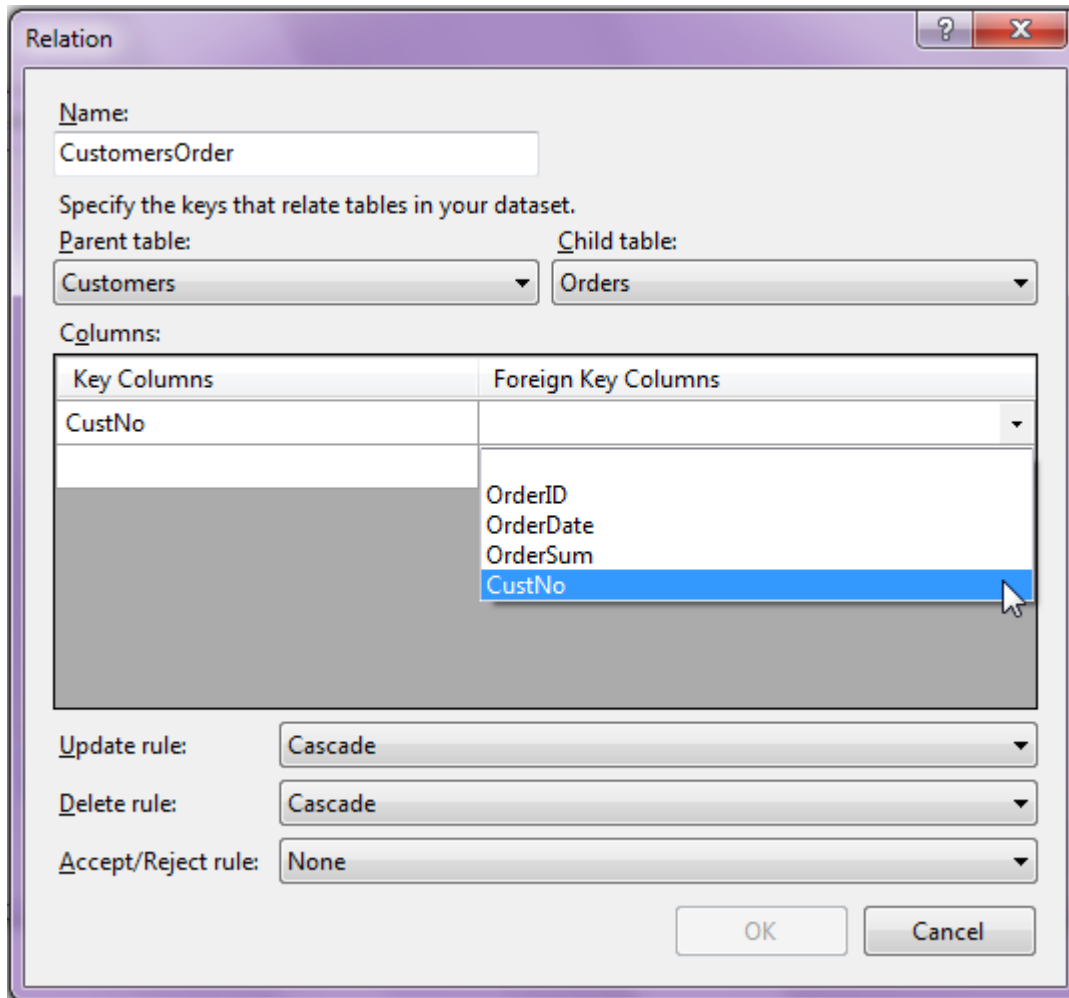
Select dataSet1 component in the form editor. On the property grid, select Relation property and click button  to open property editor.



Click "Add".

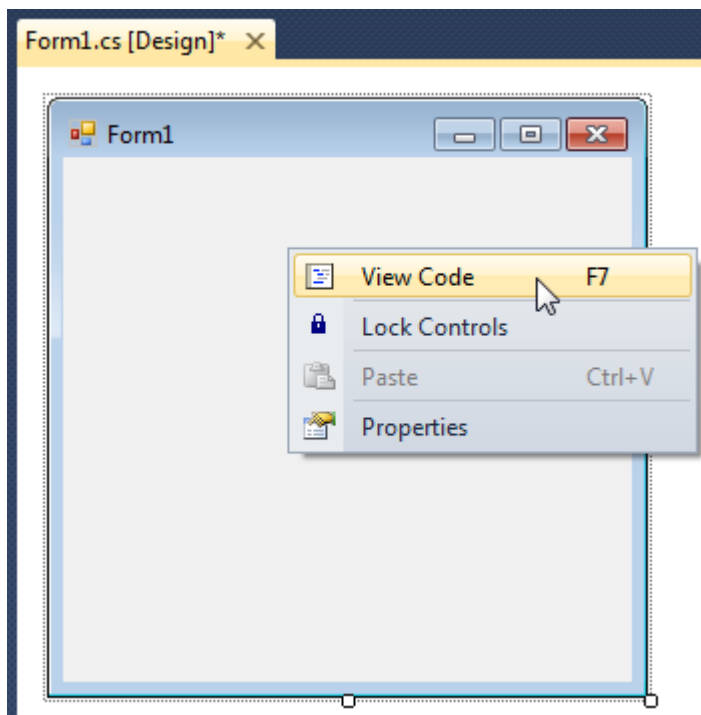


In the Relation form, set "CustomersOrder" in the Name field. Select "Customers" in the "Parent Table" and "Orders" in the "Child Table". In the "Columns table", set Key Columns - "CustNo", Foreign Key Columns - "CustNo".



Step 8

Right click on the form and select "View Code" in the context menu in order to view code.



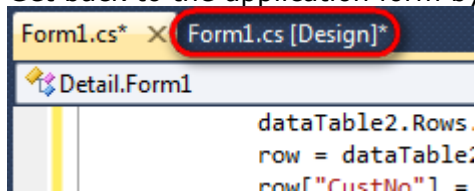
Add the following code in the class constructor to fill in a data source:



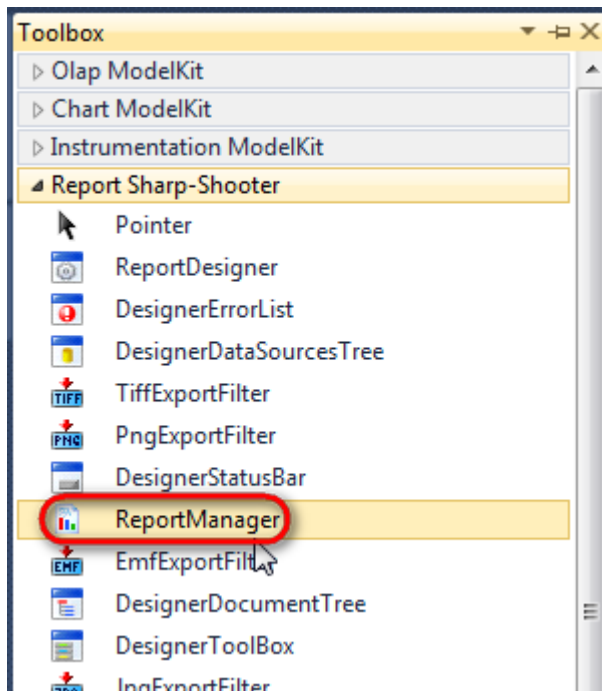
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CustNo"] = 1;
    row["CompanyName"] = "Bon App'";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CustNo"] = 2;
    row["CompanyName"] = "Chop-suey Chinese";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CustNo"] = 3;
    row["CompanyName"] = "Maison Dewey";
    dataTable1.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 1;
    row["OrderID"] = "00001";
    row["OrderDate"] = "21.03.2010";
    row["OrderSum"] = "50.00";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 1;
    row["OrderID"] = "00002";
    row["OrderDate"] = "15.02.2010";
    row["OrderSum"] = "14.50";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00010";
    row["OrderDate"] = "17.04.2010";
    row["OrderSum"] = "134.00";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00011";
    row["OrderDate"] = "24.01.2010";
    row["OrderSum"] = "45.45";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00013";
    row["OrderDate"] = "14.02.2010";
    row["OrderSum"] = "500.00";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00101";
    row["OrderDate"] = "13.03.2010";
    row["OrderSum"] = "6.03";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 3;
    row["OrderID"] = "00666";
    row["OrderDate"] = "06.06.2010";
    row["OrderSum"] = "66.66";
    dataTable2.Rows.Add (row);
}
```

Step 9

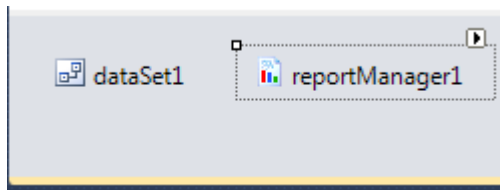
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

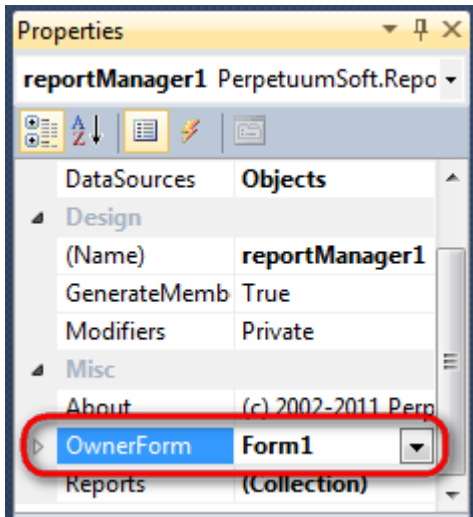


The component is available in the lower part of the window.

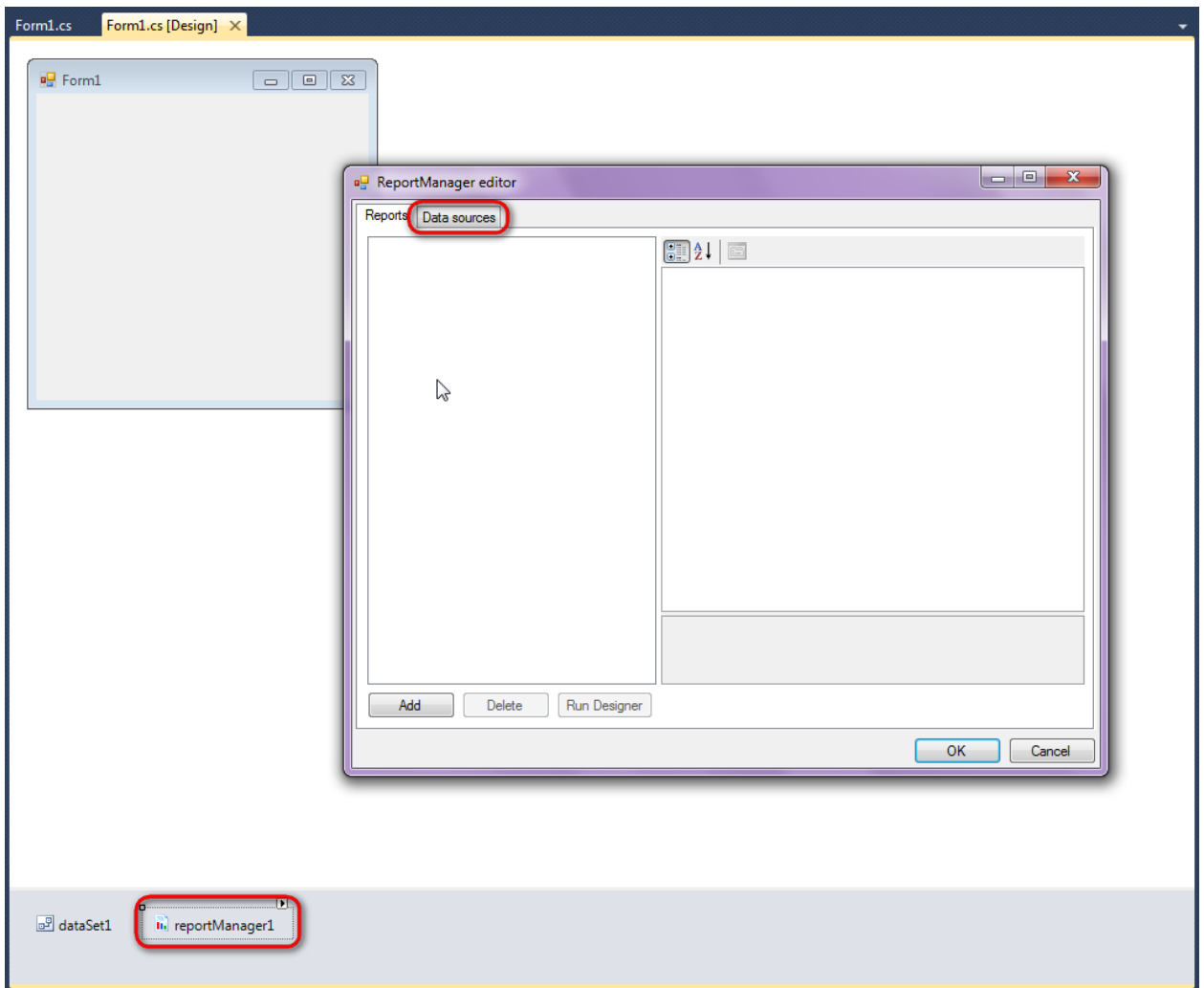


Step 10

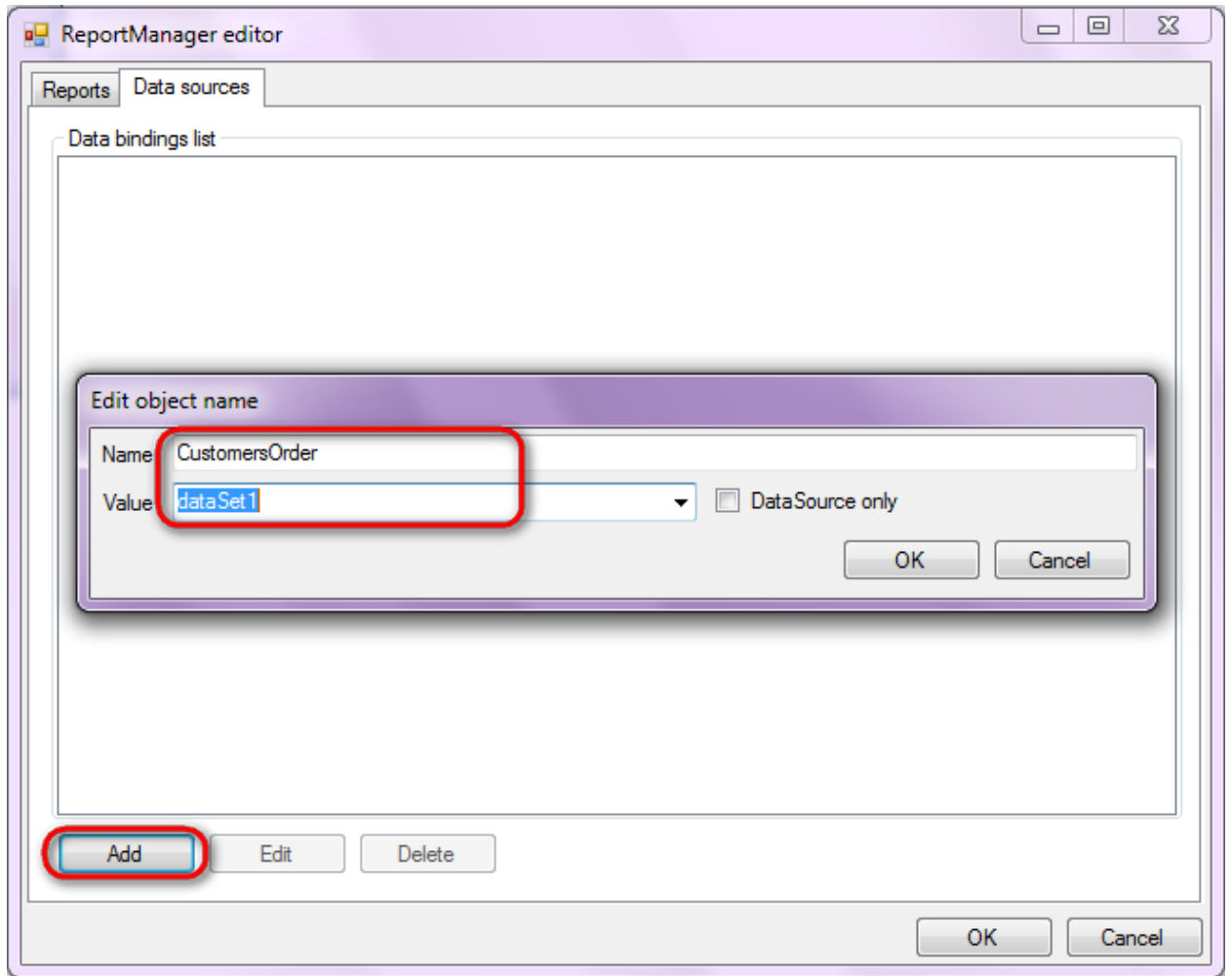
On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Double click on ReportManager to open ReportManager editor.



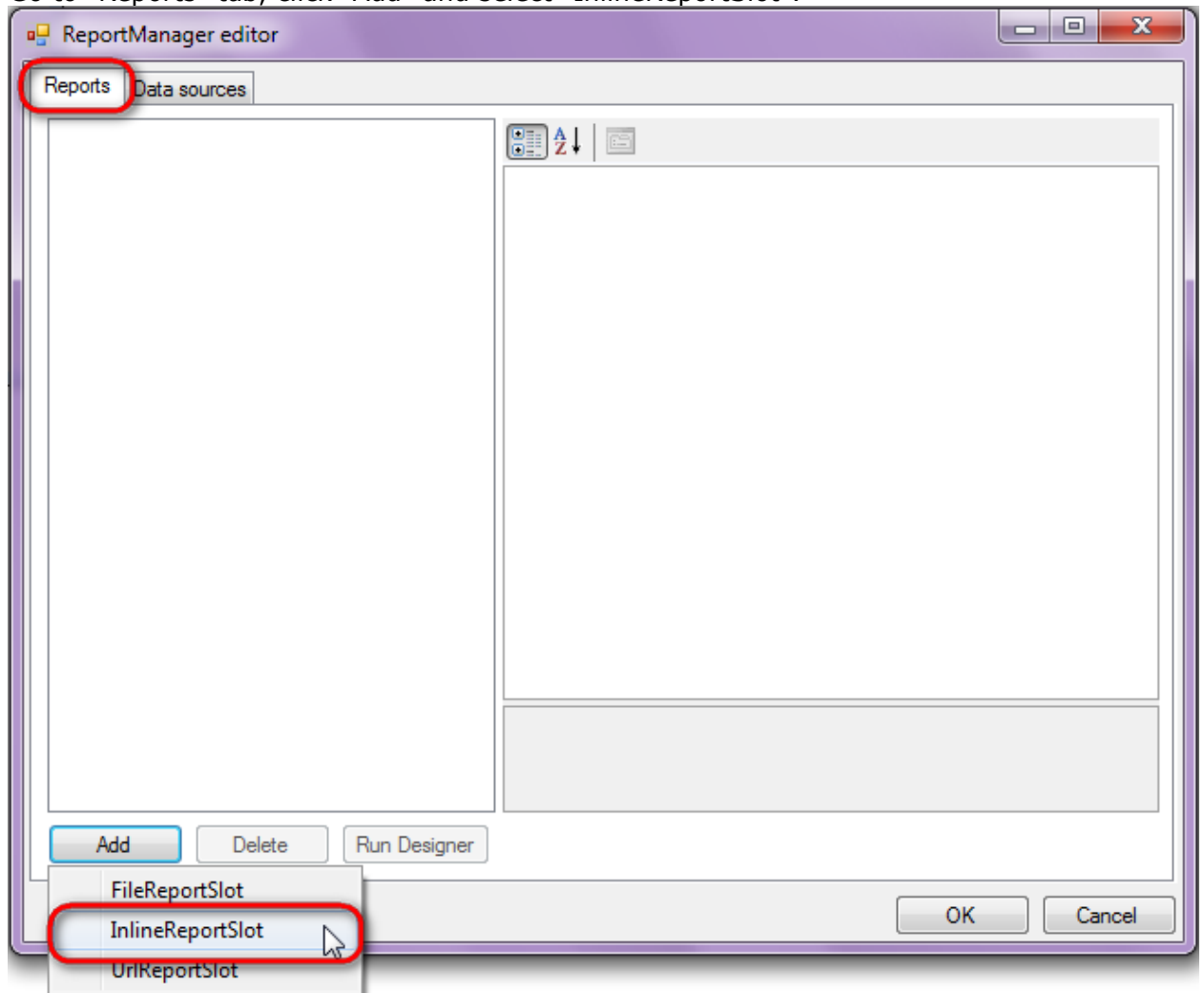
Go to Data sources tab, click "Add", set data source name – "CustomersOrder", select data source value – "dataSet1".





Step 11

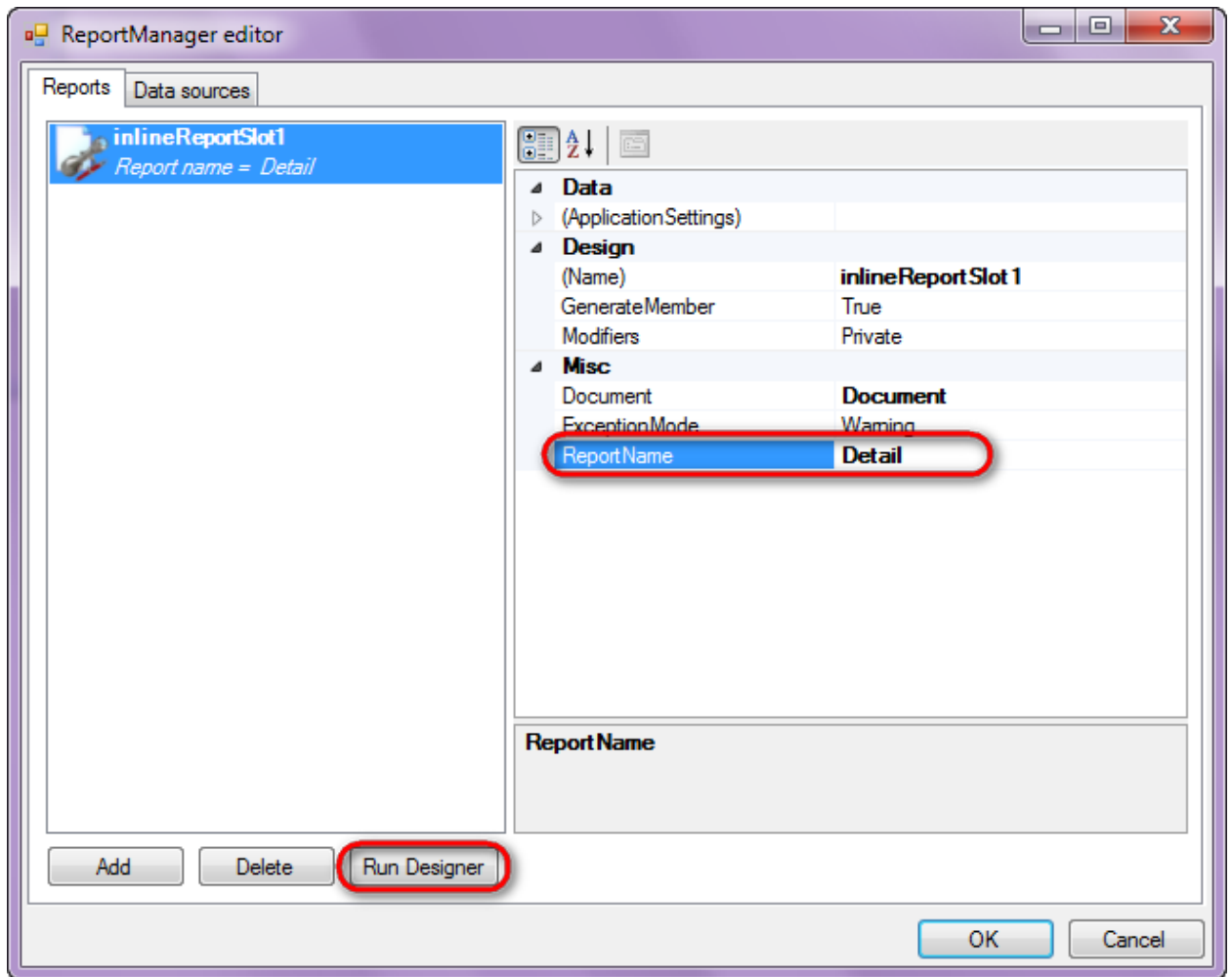
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 12

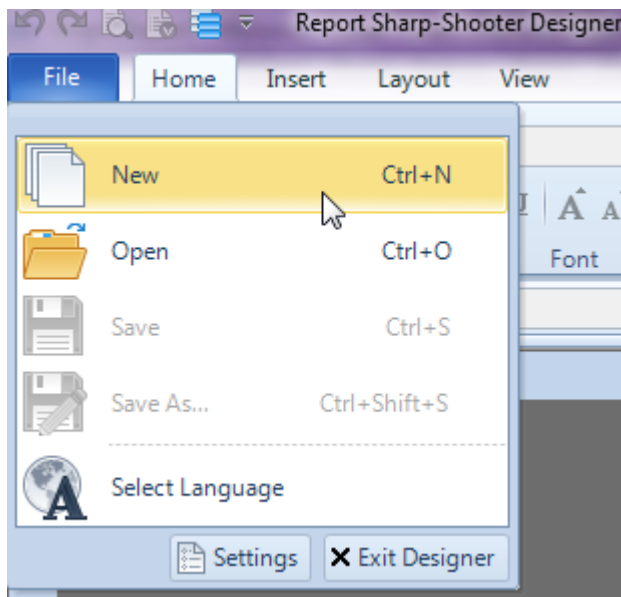
Set name of the report in the property ReportName - "Detail".

Click "Run Designer" in order to open template editor - Report Designer.

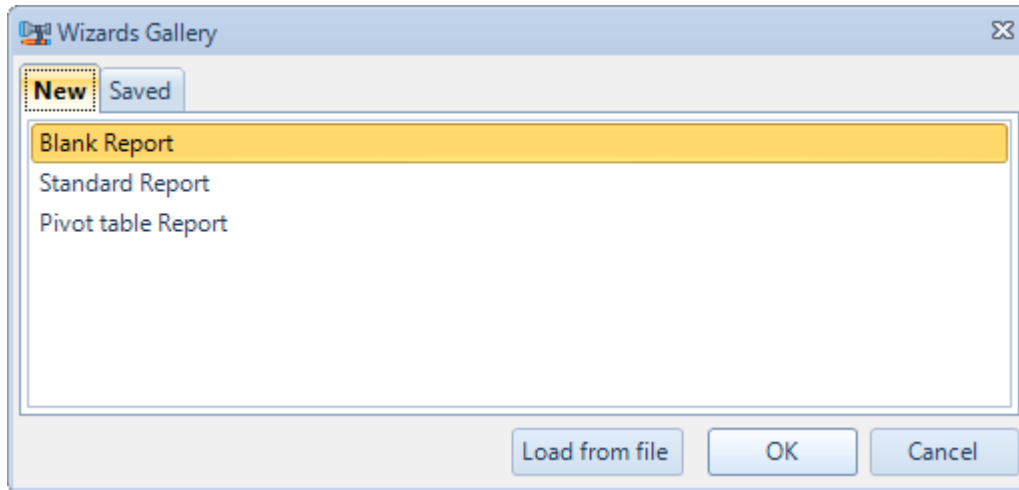


Step 13

Create new empty report – select File\New from the main menu.

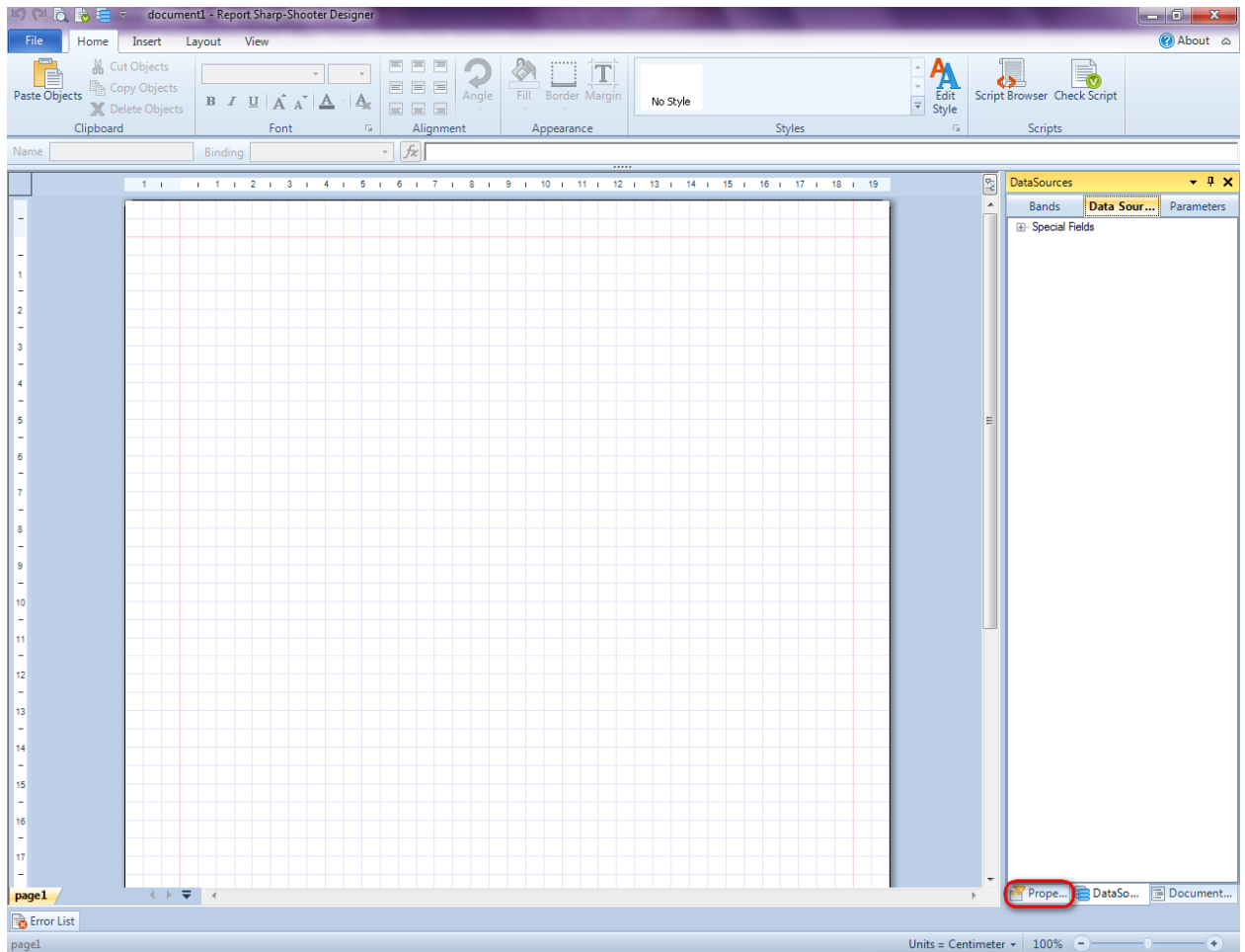


Select "Blank Report" in the Wizards Gallery and click "OK".



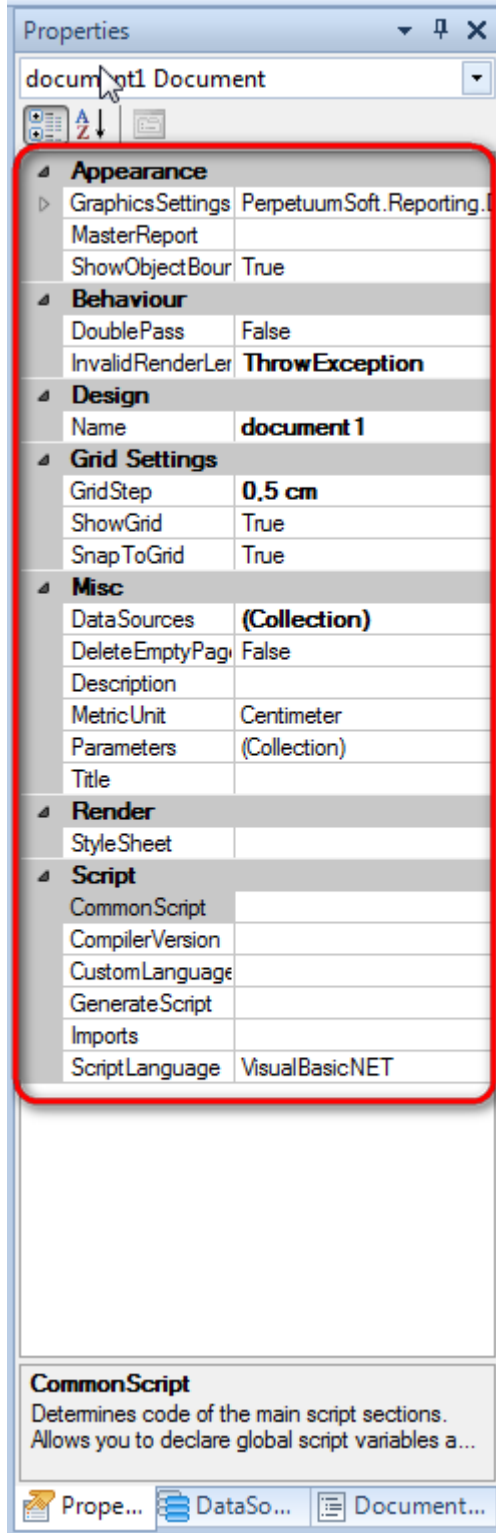
Step 14

Click the "Properties" tab of the tool window in the right part of the designer.

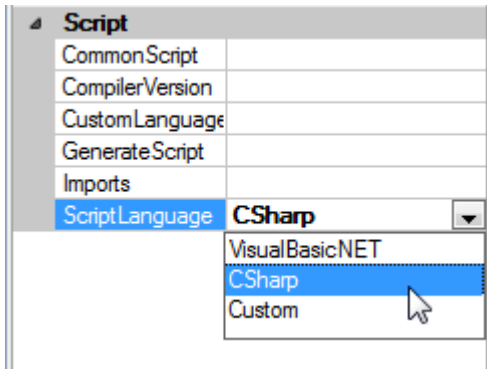




You will see properties of the edited template on the "Properties" tab

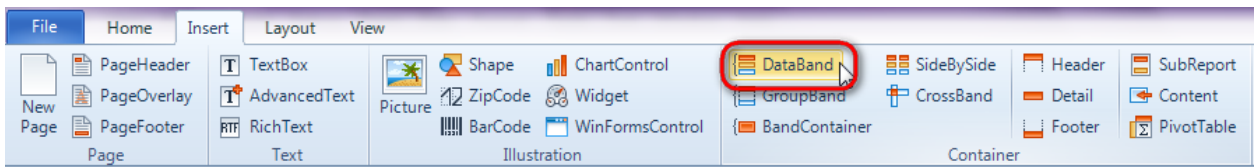


Set property ScriptLanguage = CSharp.



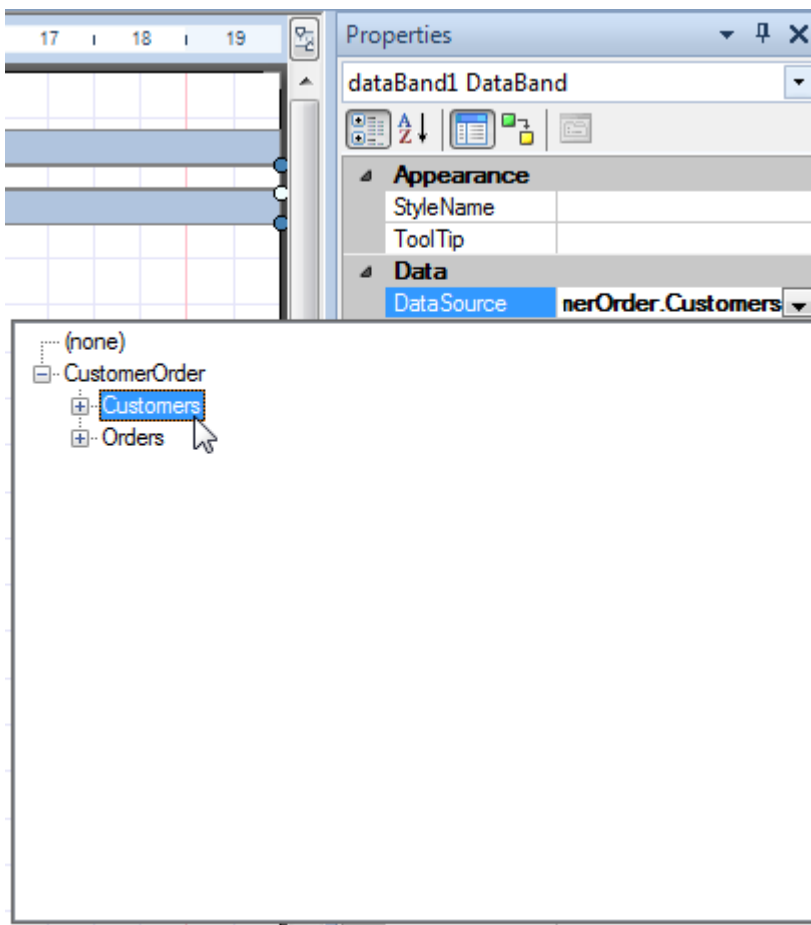
Step 15

Press "DataBand" button on the Insert tab in the group Container.



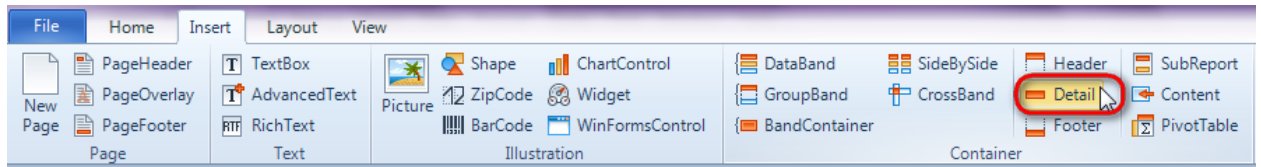
Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = CustomersOrder.Customers.

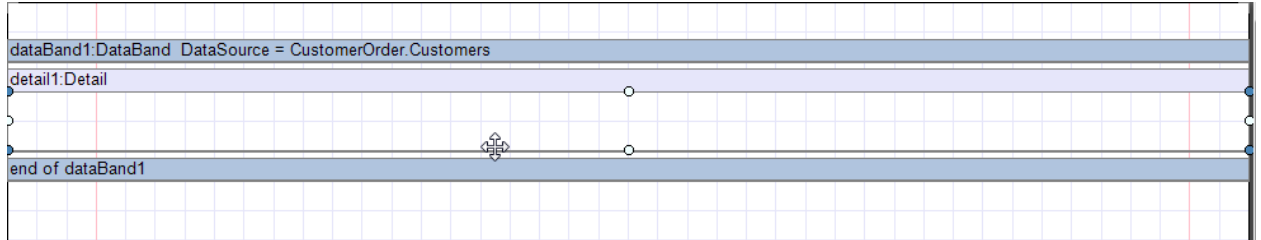


Step 16

Press "Detail" button on the Insert tab in the group Container.

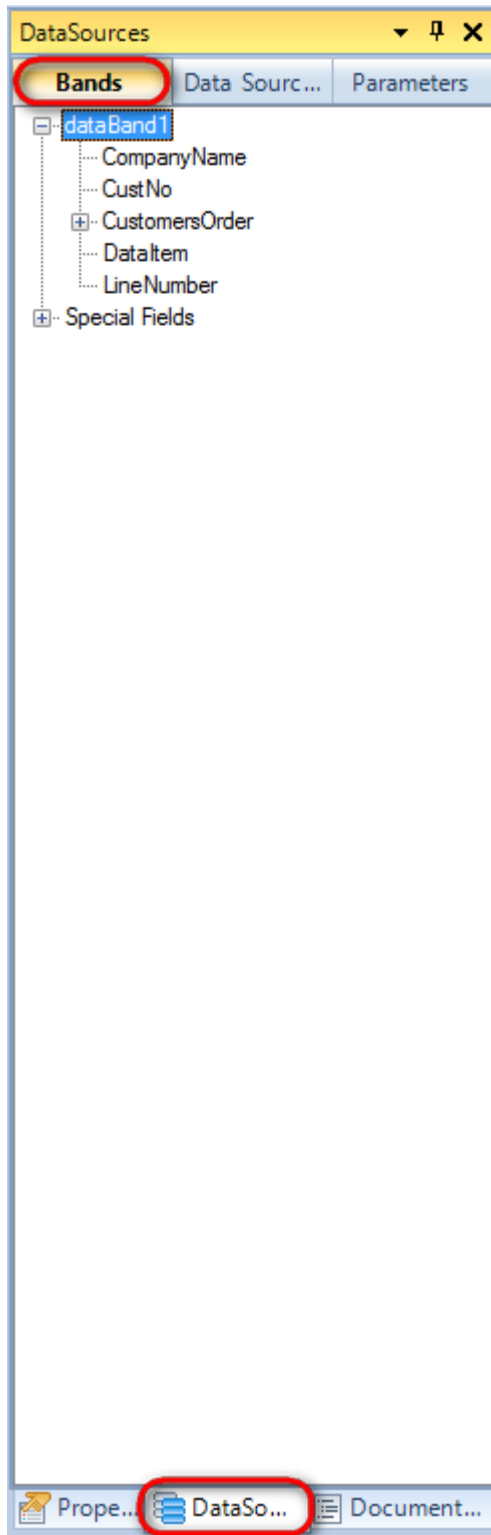


Click on the DataBand area to add Detail band inside DataBand.

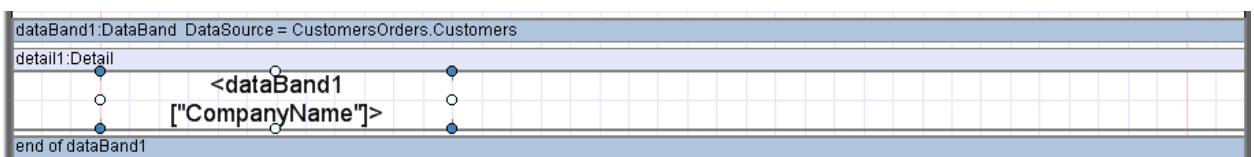


Step 17

Go to "DataSources" tab.



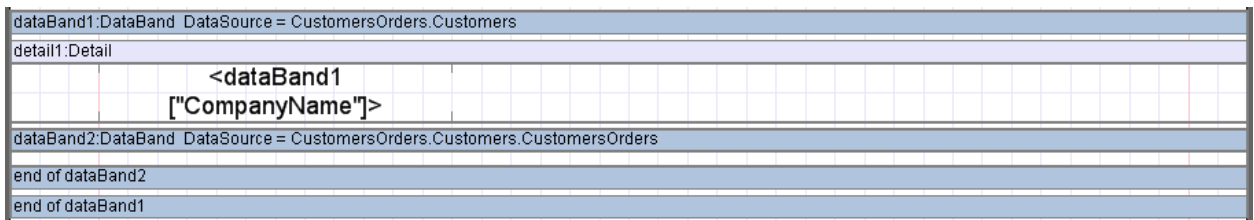
Drag and drop "CompanyName" field from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.





Step 18

Add another DataBand band to the dataBand1. Set "CustomersOrder.Customers.CustomersOrder" as a data source in the DataSource property.

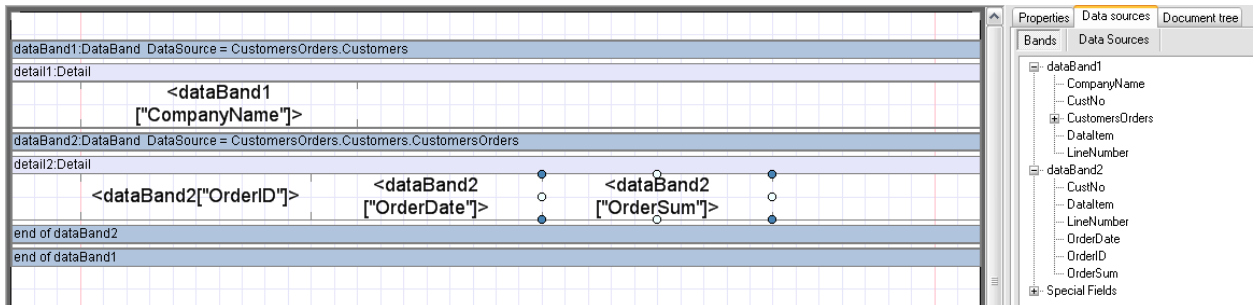


Step 19

Add Detail band to the dataBand2.

Step 20

On the "Data Sources" tab, drag and drop "OrderID", "OrderDate", "OrderSum" fields from the dataBand2 tree to the detail2 band.



Step 21

Save template and close Report Designer.

Step 22

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

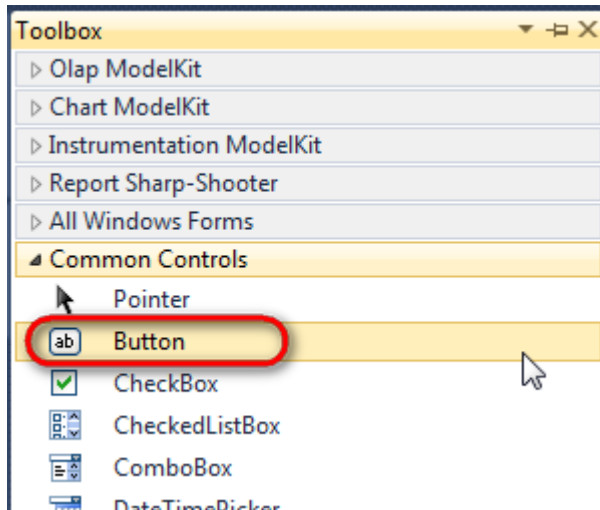
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CustNo"] = 1;
    row["CompanyName"] = "Bon App'";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CustNo"] = 2;
    row["CompanyName"] = "Chop-suey Chinese";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CustNo"] = 3;
    row["CompanyName"] = "Maison Dewey";
    dataTable1.Rows.Add(row);
    row = dataTable2.NewRow();
    row["CustNo"] = 1;
    row["OrderID"] = "00001";
    row["OrderDate"] = "21.03.2010";
    row["OrderSum"] = "50.00";
    dataTable2.Rows.Add(row);
    row = dataTable2.NewRow();
    row["CustNo"] = 1;
    row["OrderID"] = "00002";
```



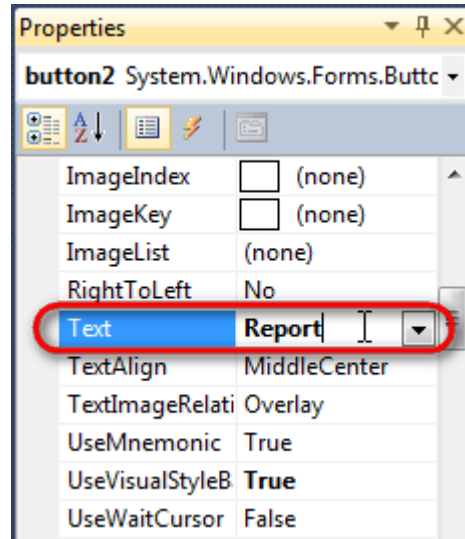
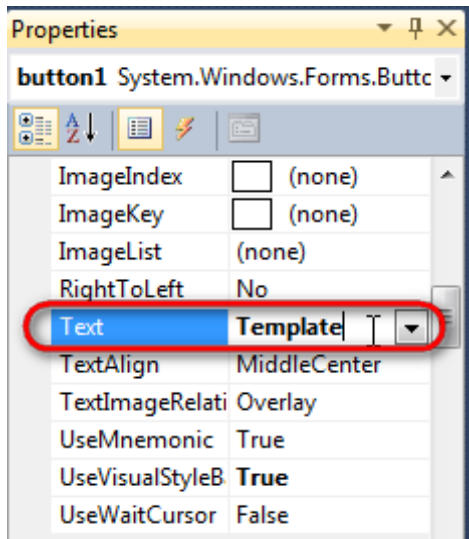
```
row["OrderDate"] = "15.02.2010";
row["OrderSum"] = "14.50";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00010";
row["OrderDate"] = "17.04.2010";
row["OrderSum"] = "134.00";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00011";
row["OrderDate"] = "24.01.2010";
row["OrderSum"] = "45.45";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00013";
row["OrderDate"] = "14.02.2010";
row["OrderSum"] = "500.00";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00101";
row["OrderDate"] = "13.03.2010";
row["OrderSum"] = "6.03";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 3;
row["OrderID"] = "00666";
row["OrderDate"] = "06.06.2010";
row["OrderSum"] = "66.66";
dataTable2.Rows.Add(row);
inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 23

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



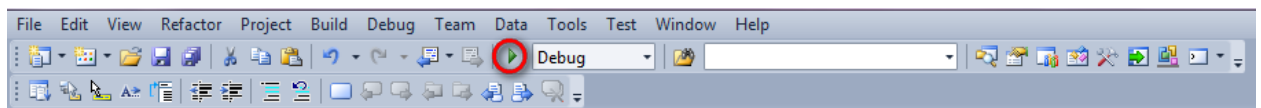
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

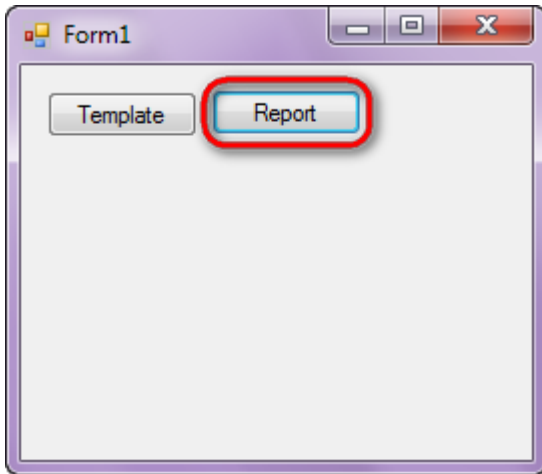
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 24

Click “Start Debugging” on the Visual Studio toolbar in order to run application.



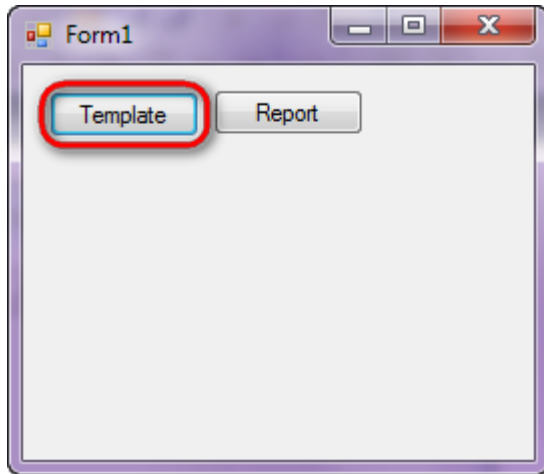
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.

Item Name	ID	Date	Price
Bon App'			
	00001	21.03.2010	50.00
	00002	15.02.2010	14.50
Chop-suey Chinese			
	00010	17.04.2010	134.00
	00011	24.01.2010	45.45
	00013	14.02.2010	500.00
	00101	13.03.2010	6.03
Maison Dewey			
	00666	06.06.2010	66.66

In order to edit template, close Report Viewer and press "Template" on the application form.



Similar application sample is located in the following folder "\\Perpetuum Software\Net ModelKit Suite\ Samples\Report Sharp-Shooter\CSharp\MasterDetail".

Similar sample in the Samples Center is Reports\Master-Detail\Master-Detail.

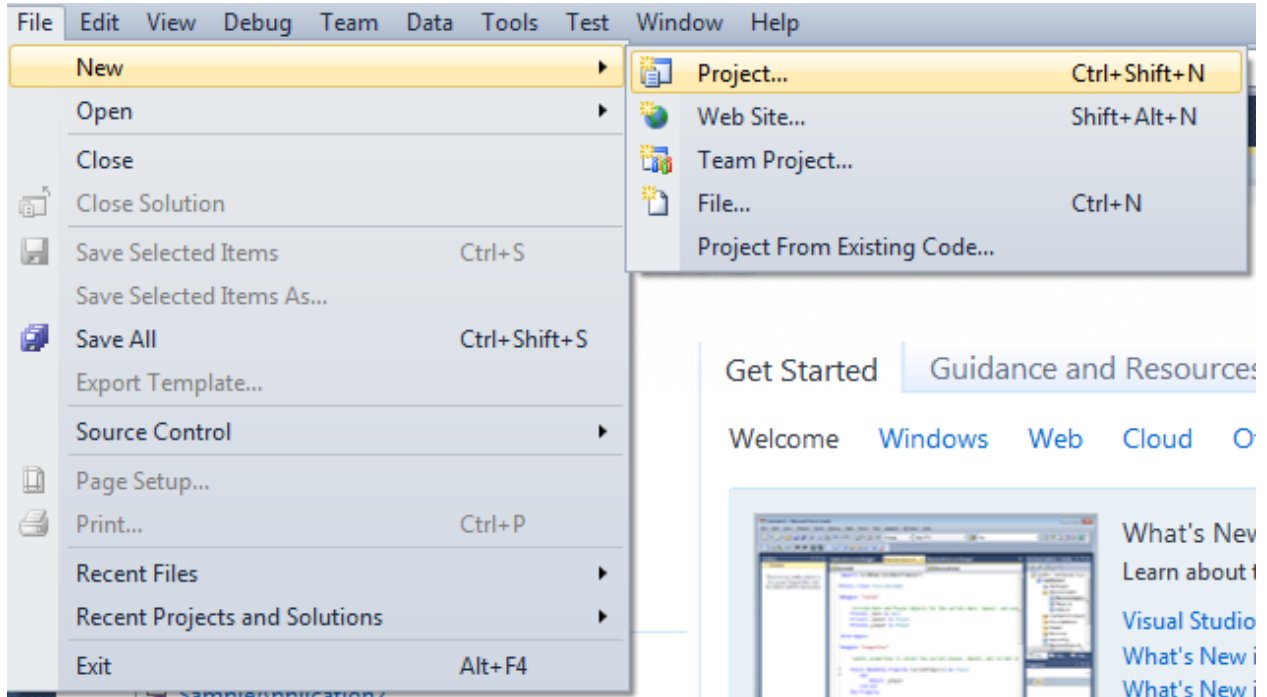


Chart

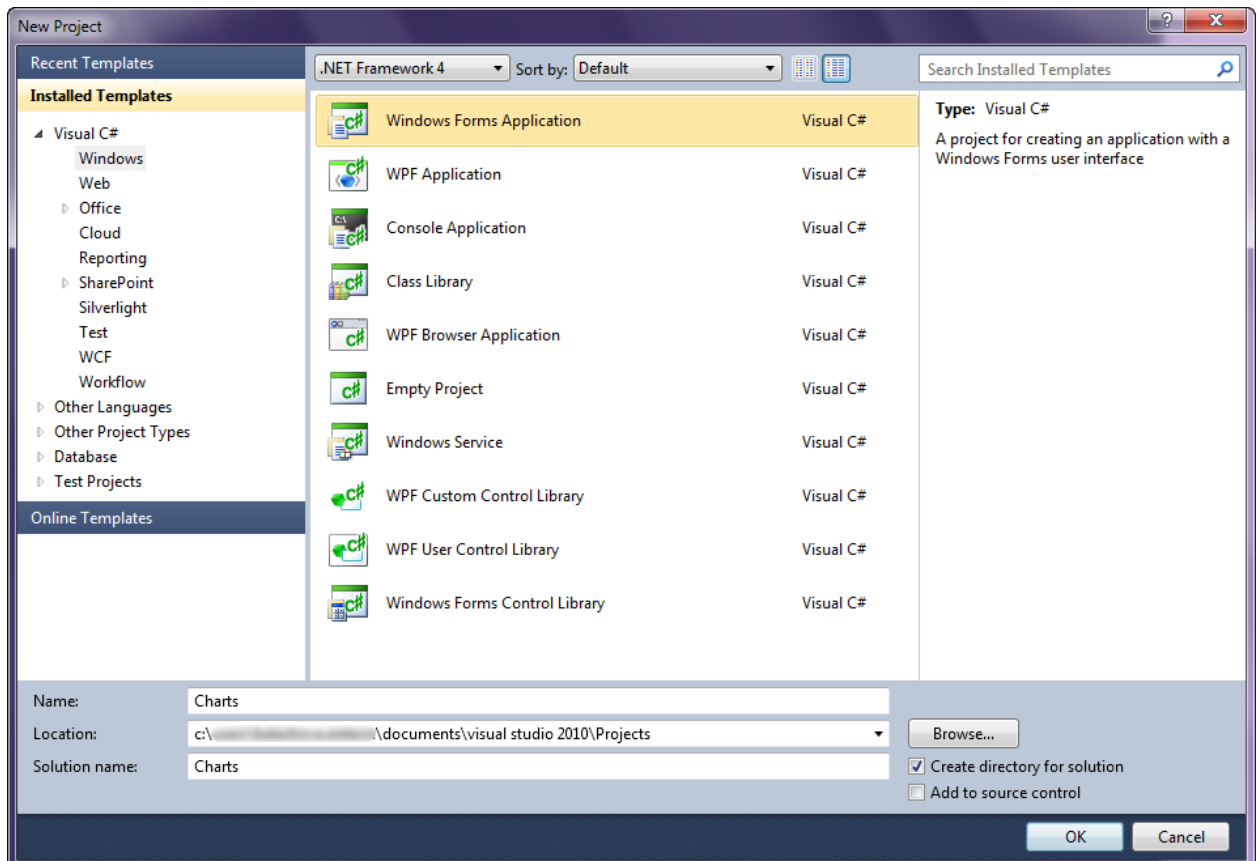
Template of a report containing charts for every category of goods including information on price of the goods in this category.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

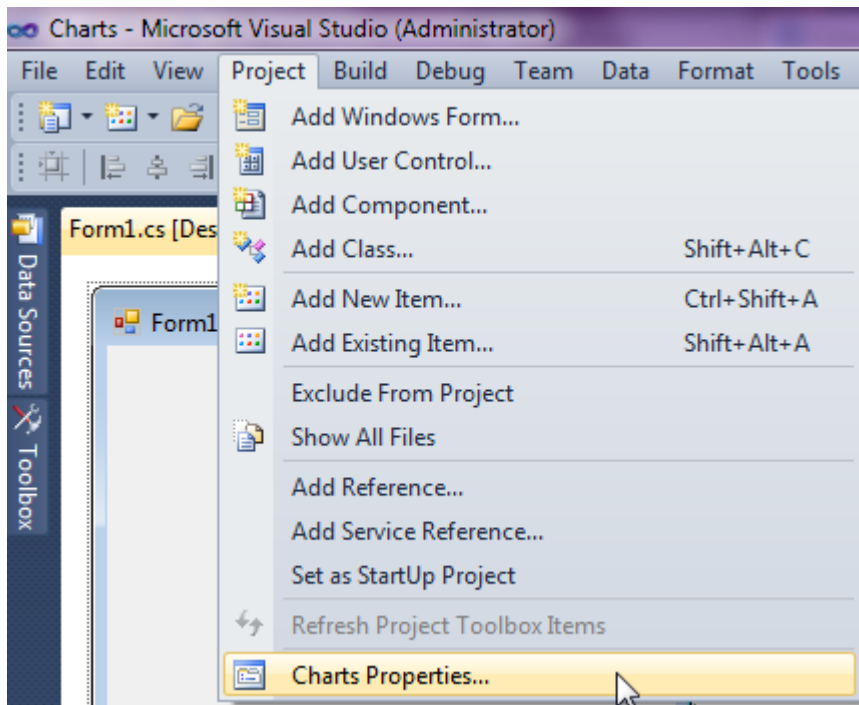


Select Windows Forms Application, set project name – “Charts”, set directory to save the project to.

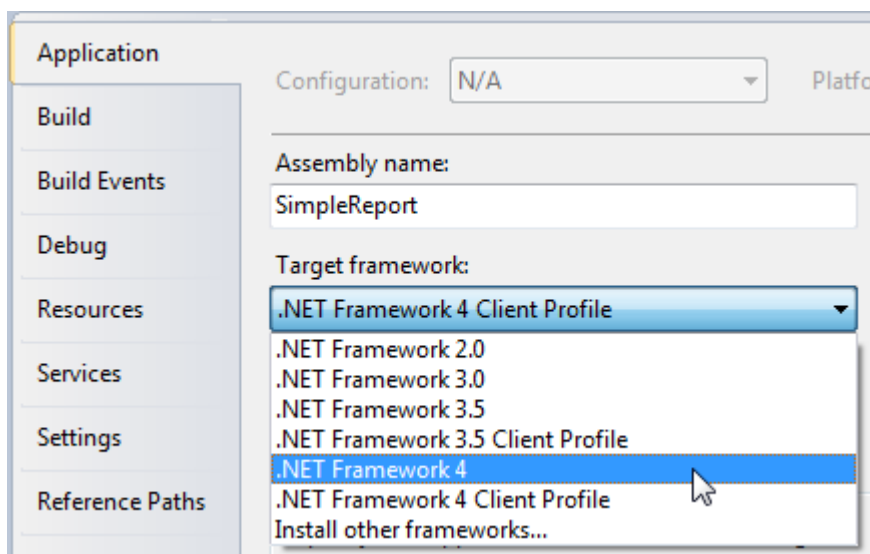


Step 2

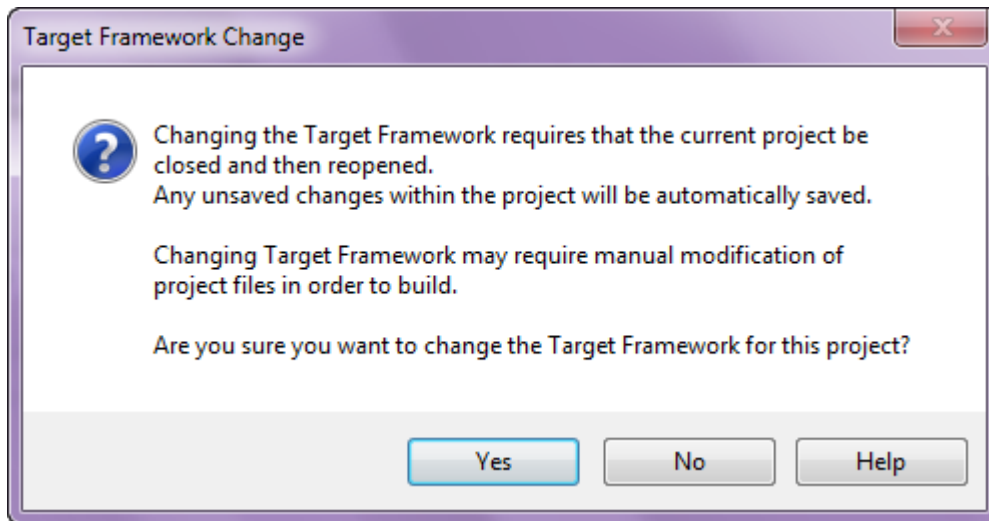
Change the project properties. Select the Project\Charts Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

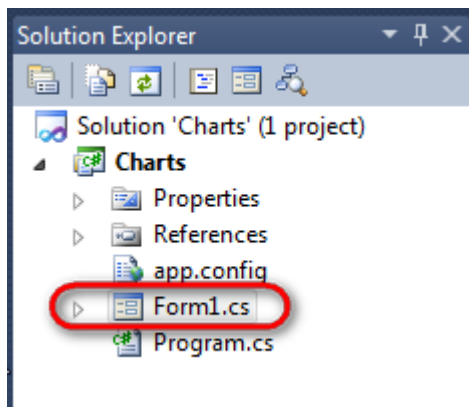


Press the "Yes" button in the opened window.

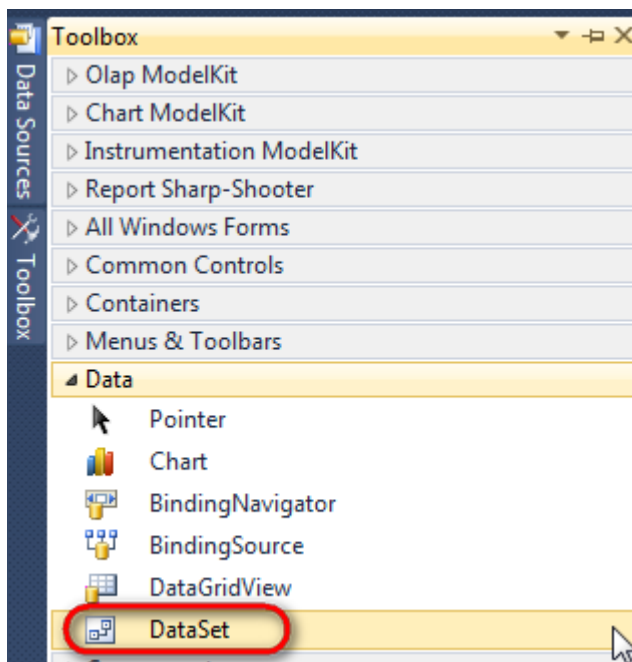


Step 3

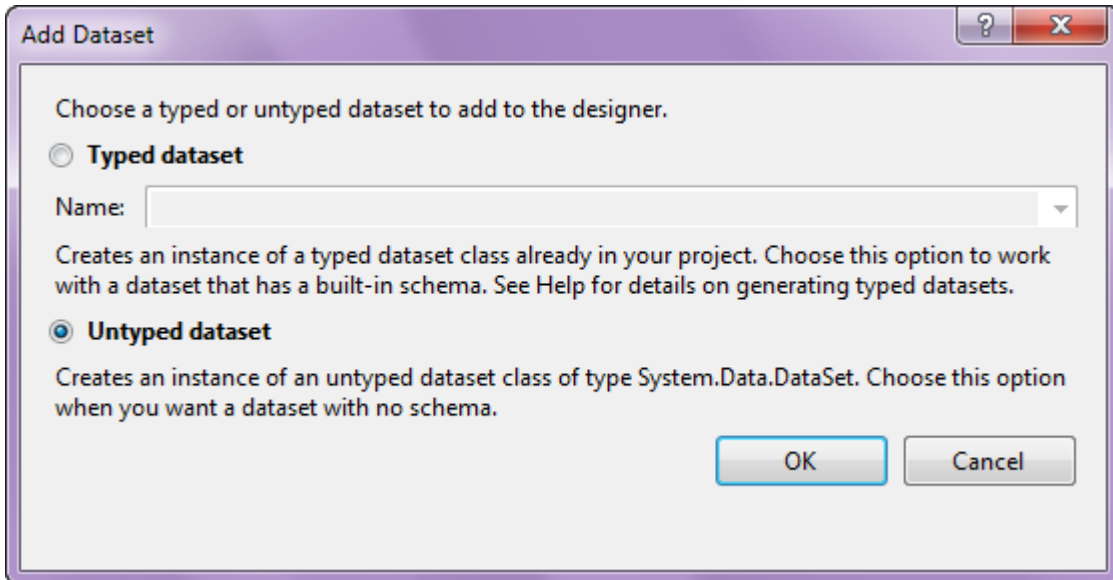
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



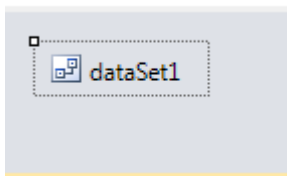
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

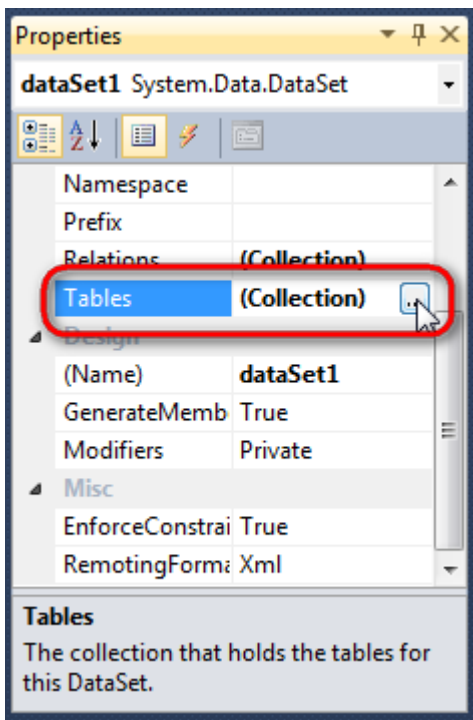


The component is available in the lower part of the window.

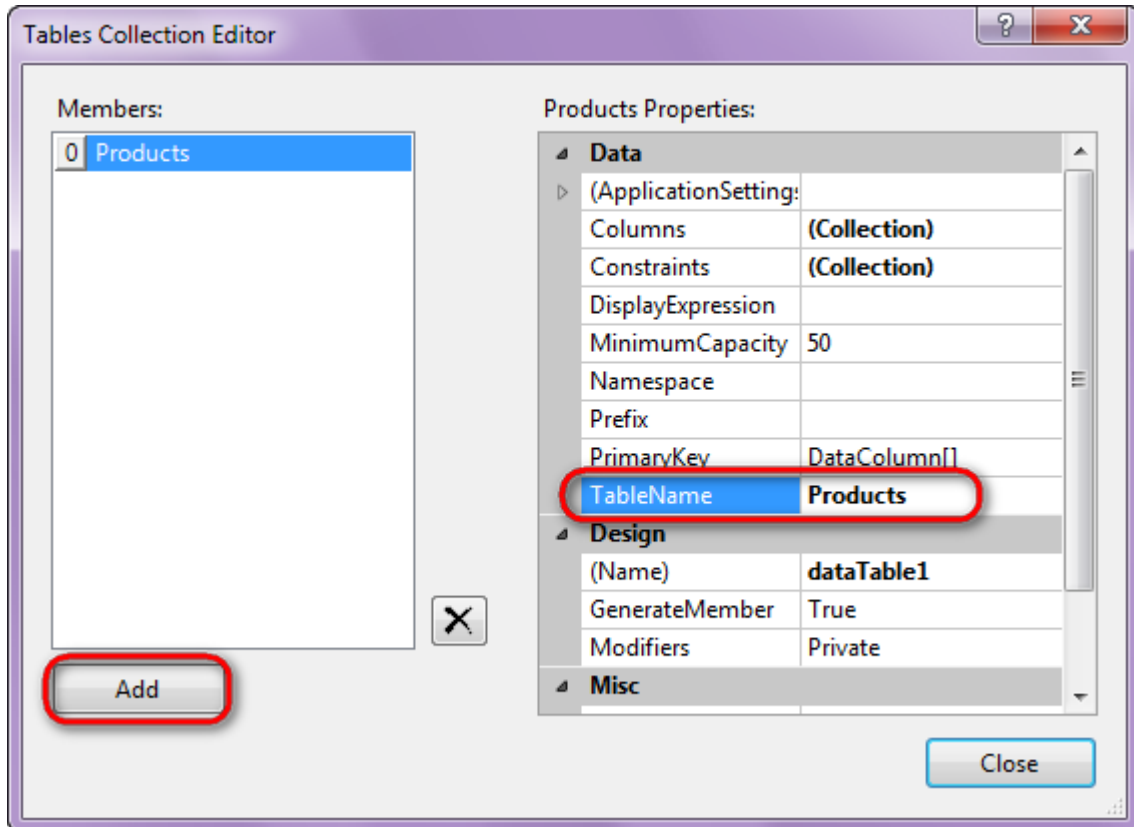


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

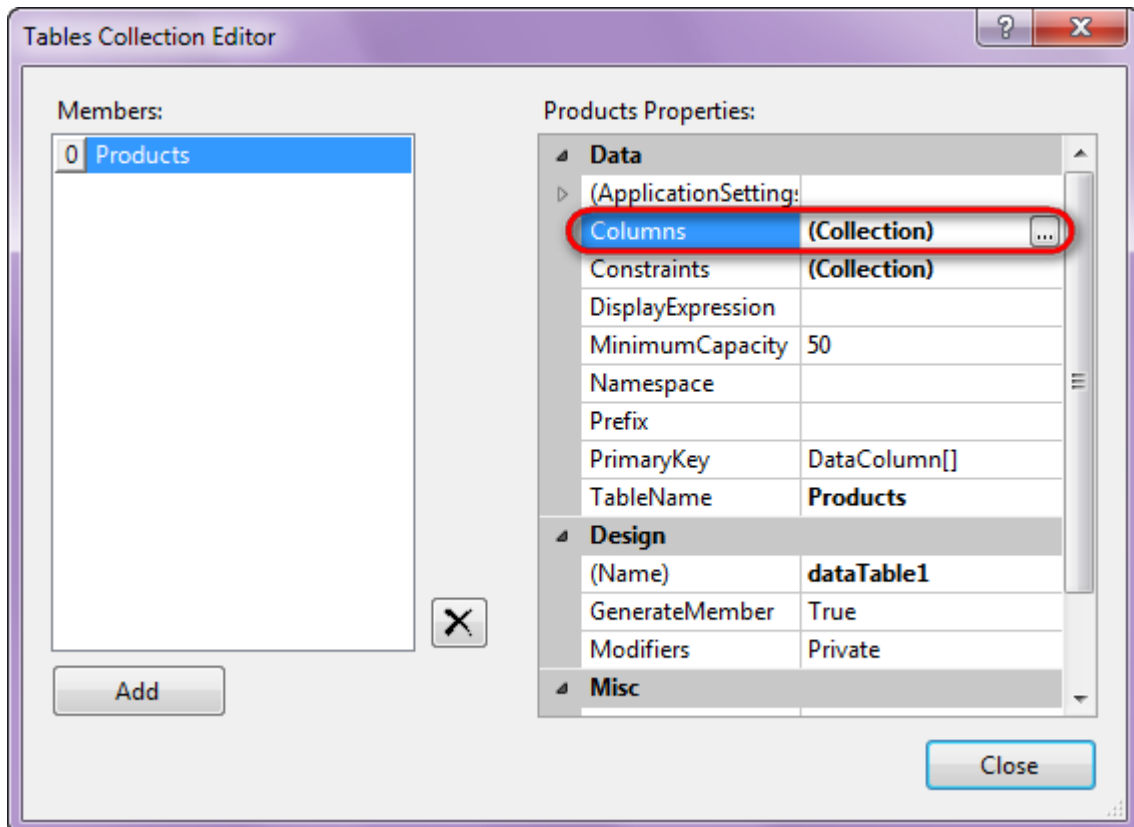


Click "Add" in order to add table. Set property TableName = Products.

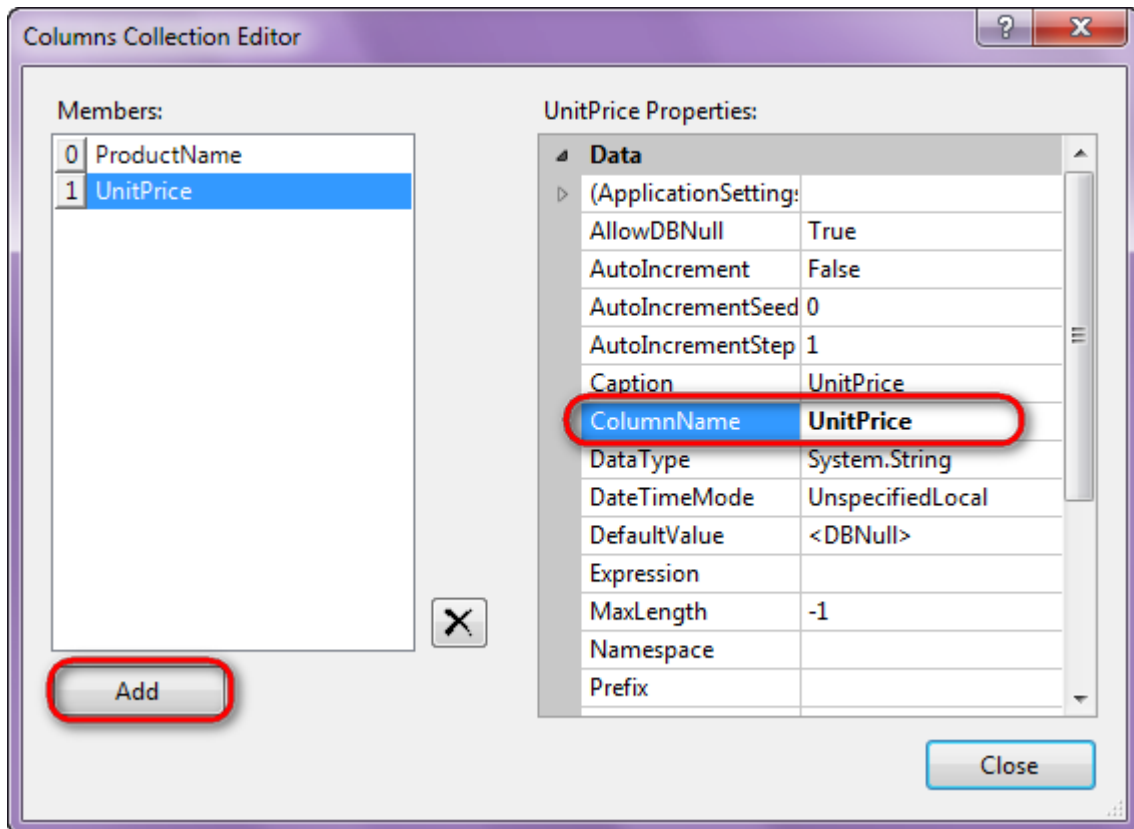


Step 5

Select Columns property, click button  in order to open property editor.

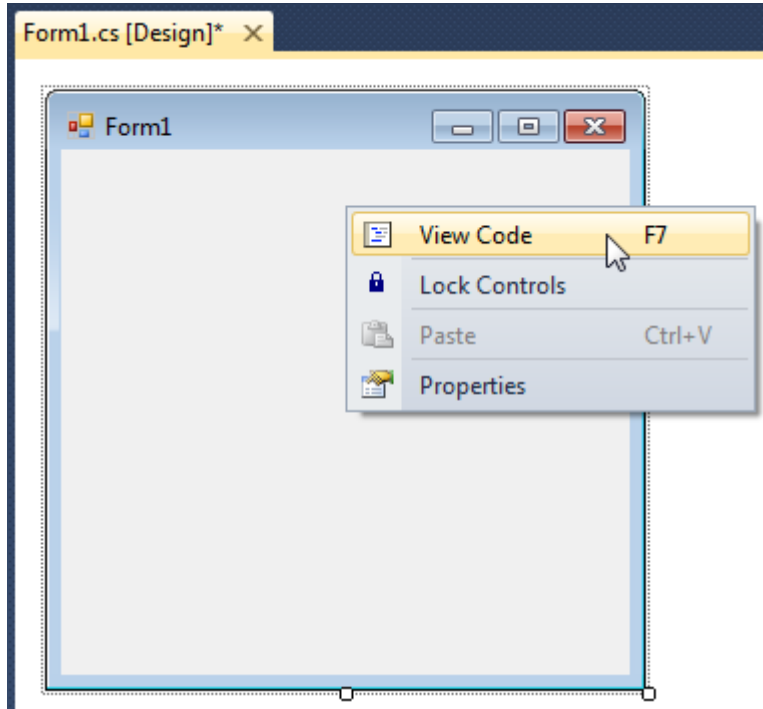


Click "Add" to add a new column. Add two columns. Set ColumnName properties to "ProductName", "UnitPrice".



Step 6

Right click on the form and select "View Code" in the context menu in order to view code.



Add the following code in the class constructor to fill in a data source:

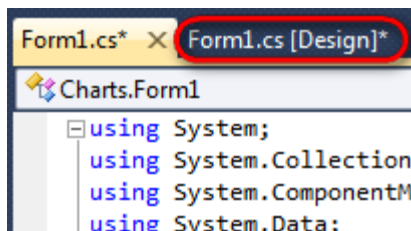
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["ProductName"] = "Chai";
}
```



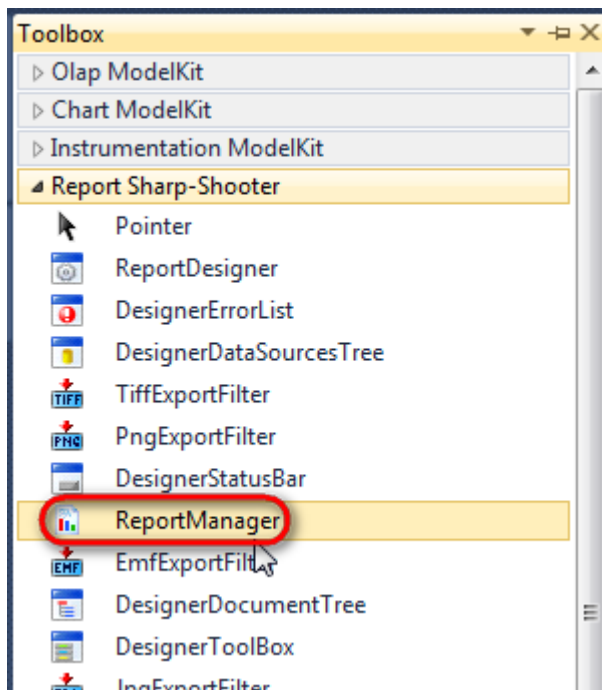
```
row["UnitPrice"] = "18";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["ProductName"] = "Chang";  
row["UnitPrice"] = "19";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["ProductName"] = "Aniseed Syrup";  
row["UnitPrice"] = "10";  
dataTable1.Rows.Add(row);  
}
```

Step 7

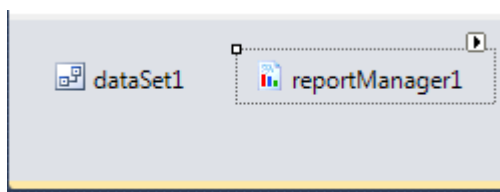
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

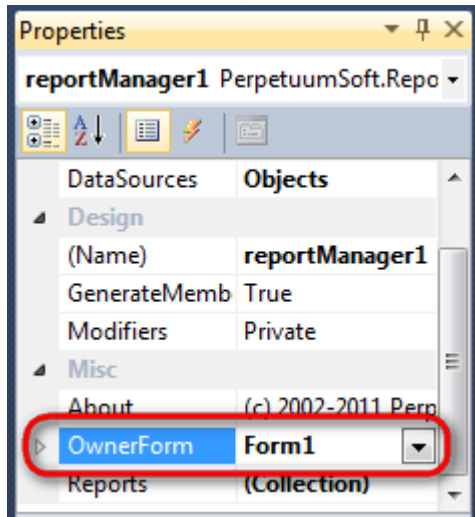


The component is available in the lower part of the window.



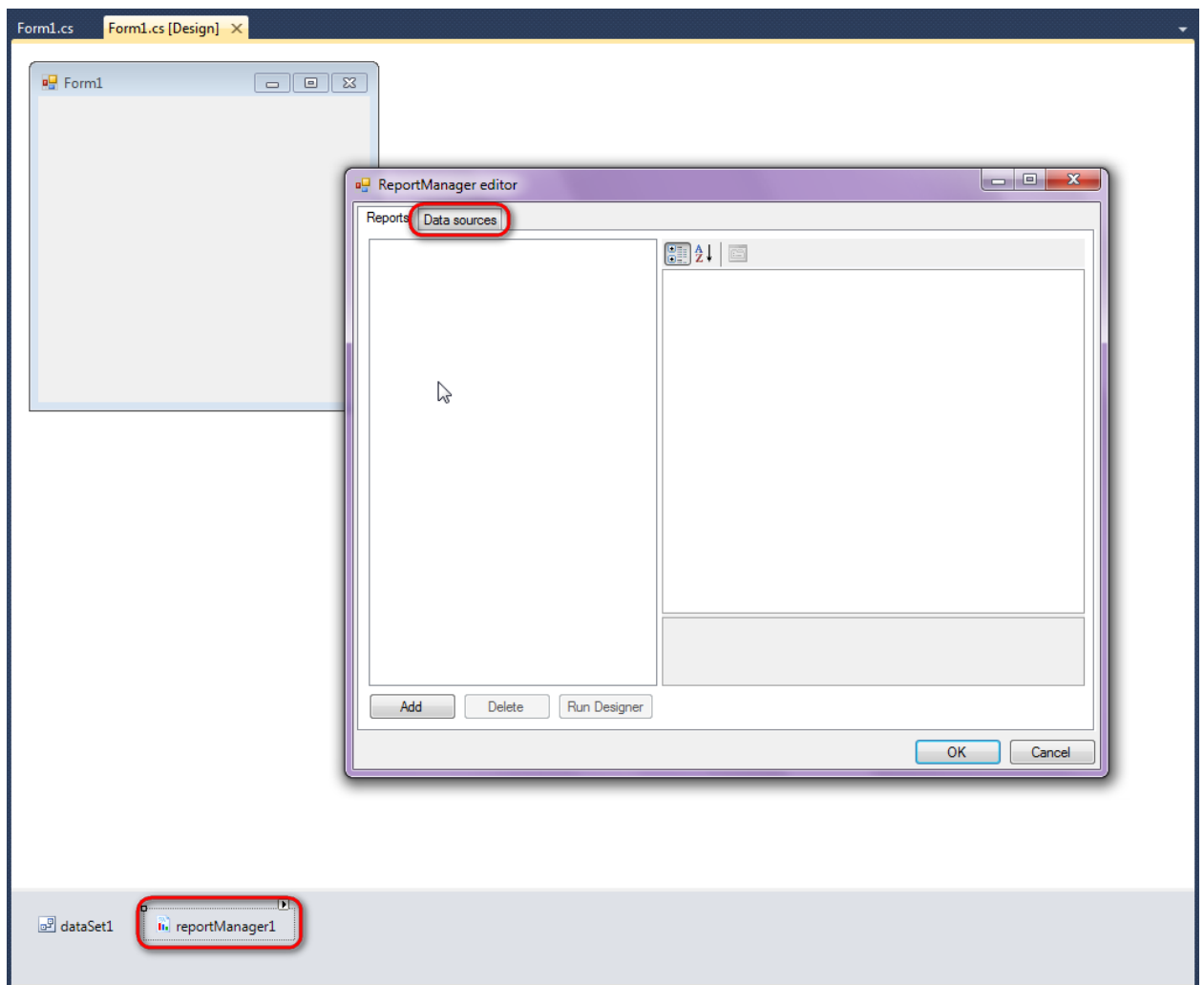
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

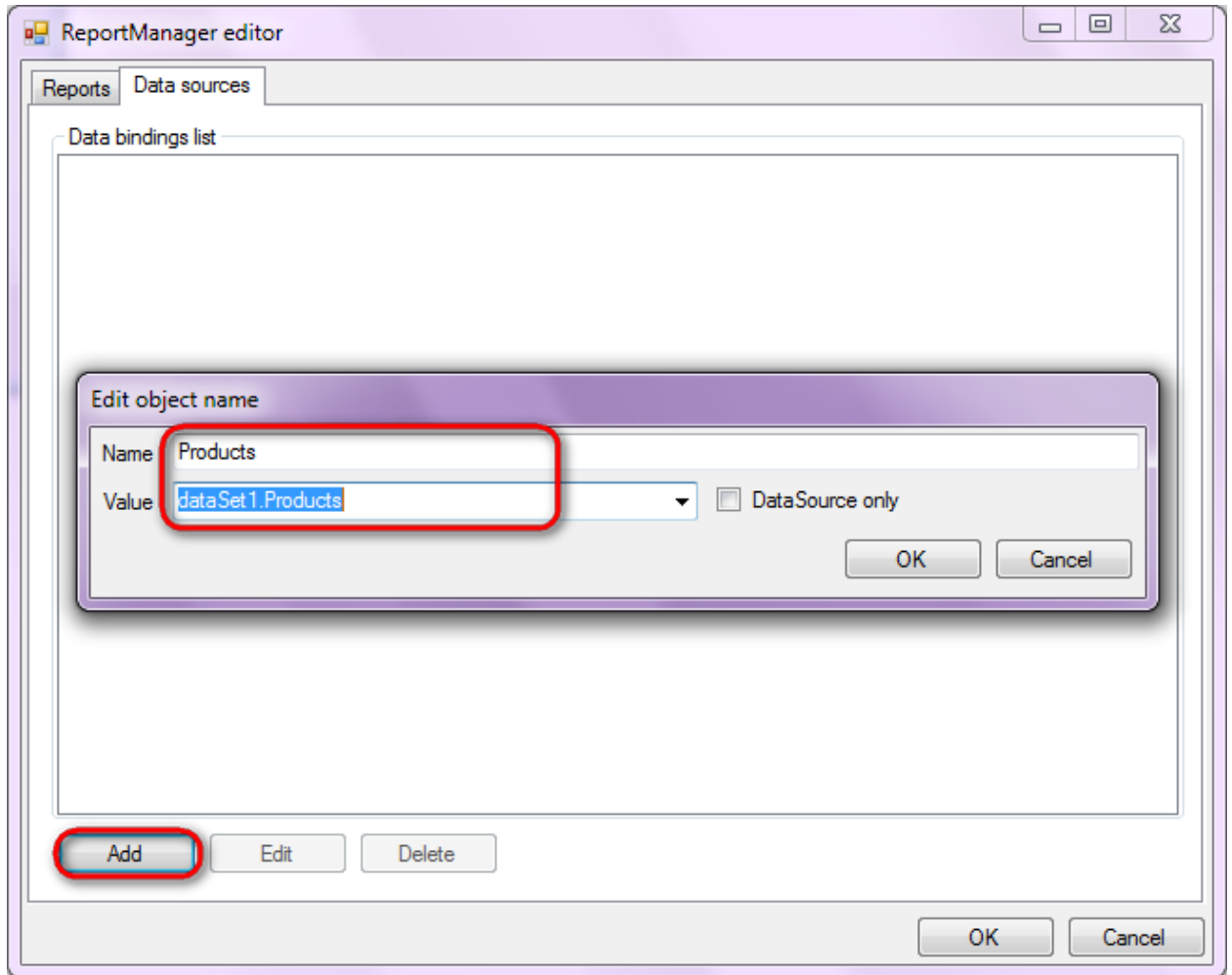


Step 9

Double click on ReportManager to open ReportManager editor.

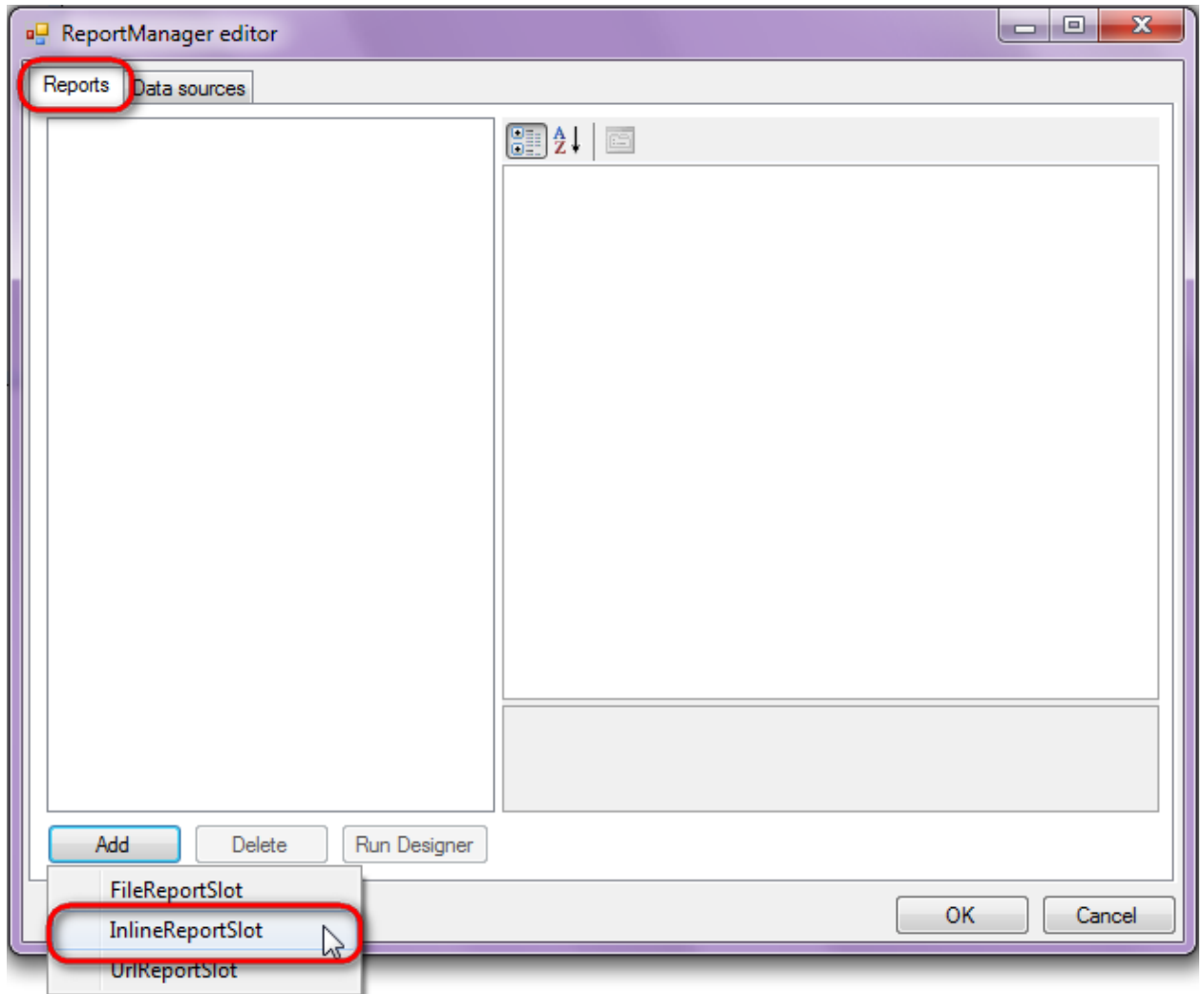


Go to "Data Sources" tab and click "Add", set data source name – "Products", select data source value – "dataSet1.Products".



Step 10

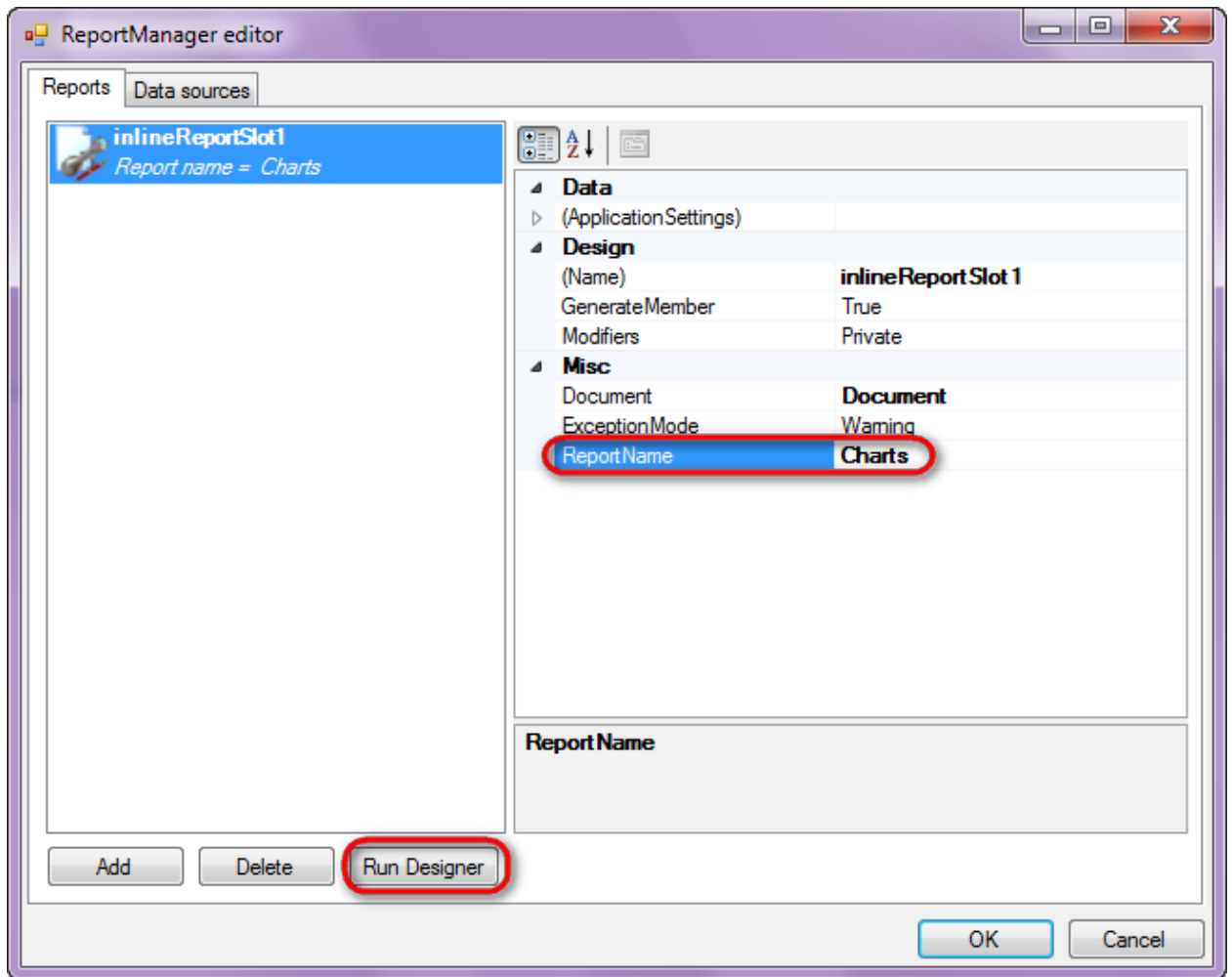
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

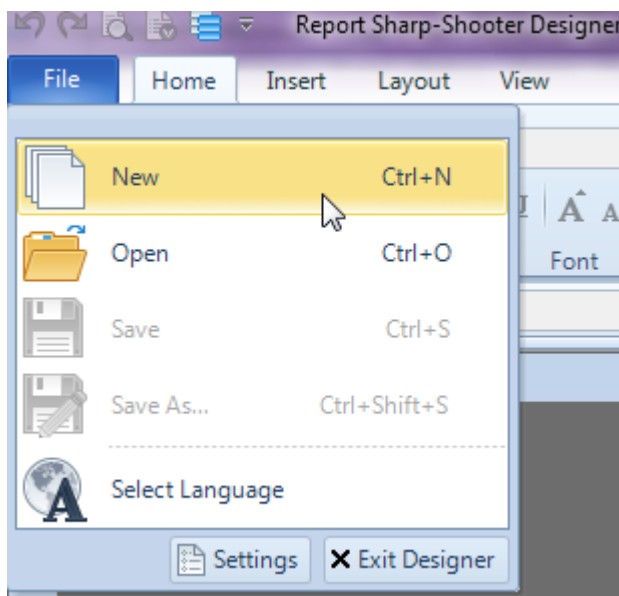
Set name of the report in ReportName – “Charts”.

Click “Run Designer” to open template editor Report Designer.

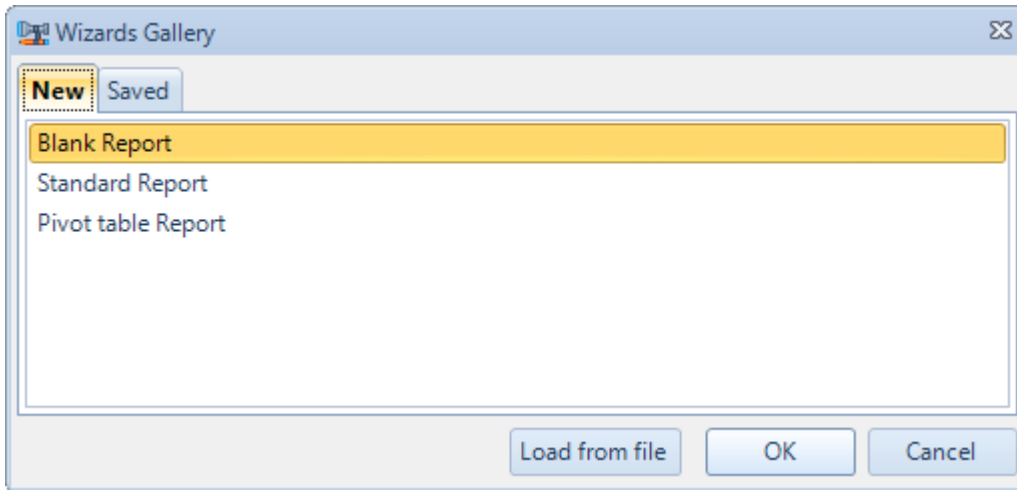


Step 12

Create new empty report – select File\New from the main menu.

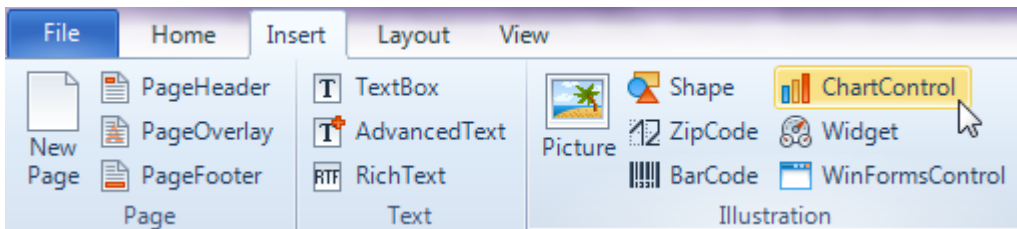


Select "Blank Report" in the Wizards Gallery and click "OK".



Step 13

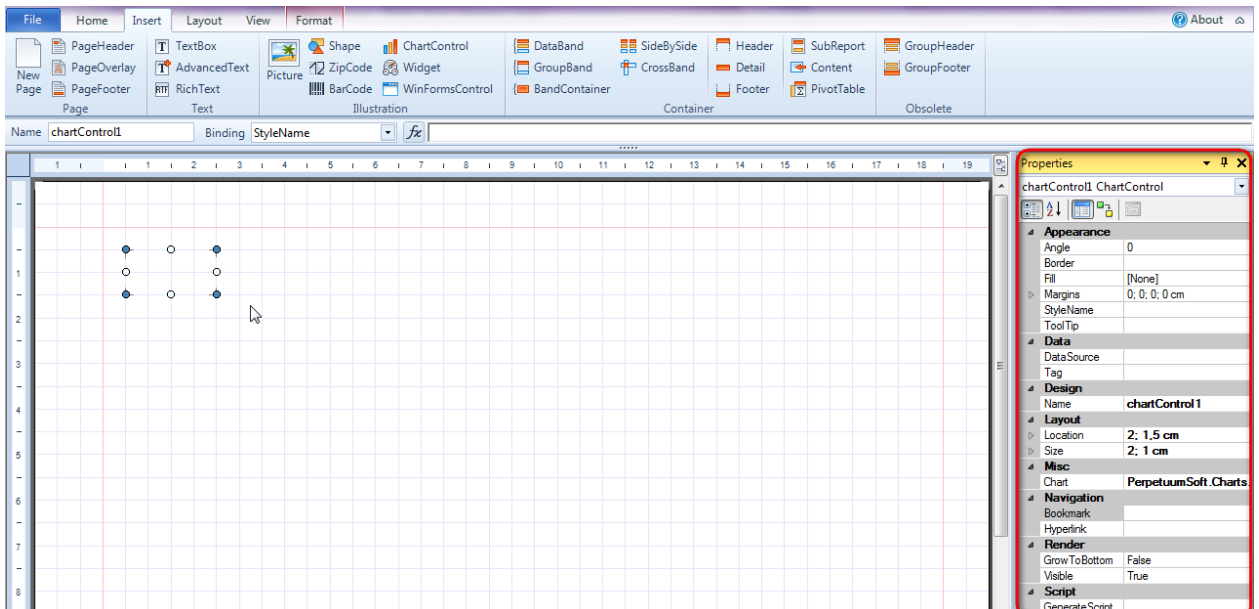
Press "ChartControl" button on the Insert tab in the group Illustration.



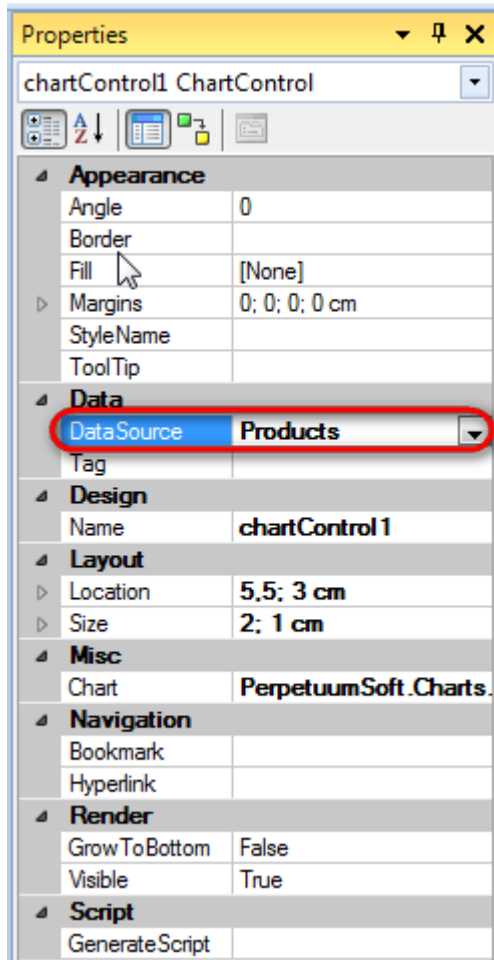
Click on the template area to add ChartControl to the template.

Step 14

Select ChartControl element. You will see component properties on the property grid of the Tool Window in the right part of the editor.



Select DataSource property, click button , and select value "Products".



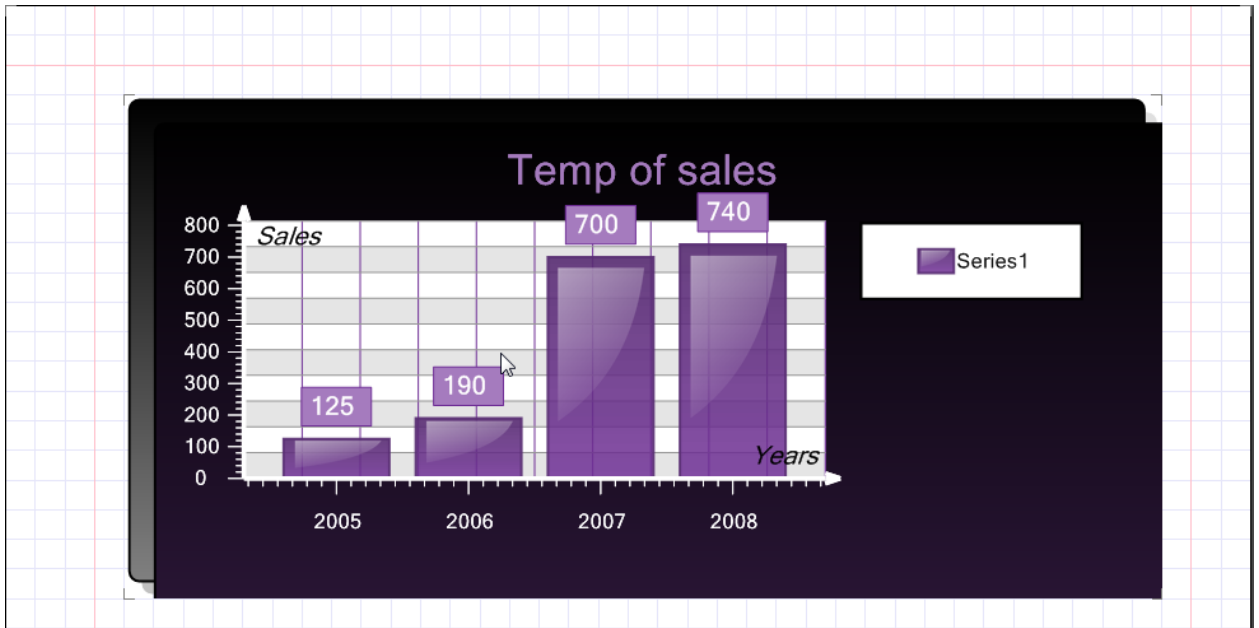
Step 15

Double click on the ChartControl area to open Chart ModelKit designer. Create a chart. Find more information on creating charts in Chart ModelKit designer in Section Chart ModelKit hereof.

Step 16

Close Chart ModelKit designer by clicking "OK" button.

Report template should look as follows:



Step 17

Save template, close Report Designer.

Step 18

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

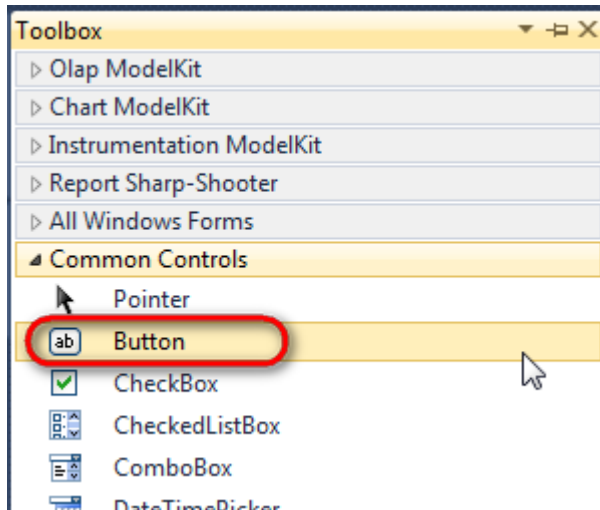
```

public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row ["ProductName"] = "Chai";
    row ["UnitPrice"] = "18";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["ProductName"] = "Chang";
    row ["UnitPrice"] = "19";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["ProductName"] = "Aniseed Syrup";
    row ["UnitPrice"] = "10";
    dataTable1.Rows.Add (row);
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted (object sender, EventArgs
e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm =
new PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}

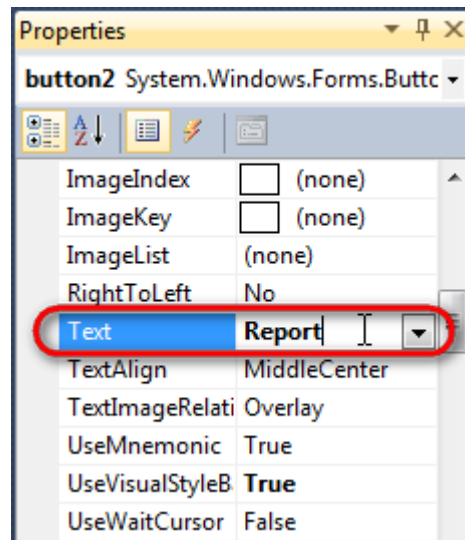
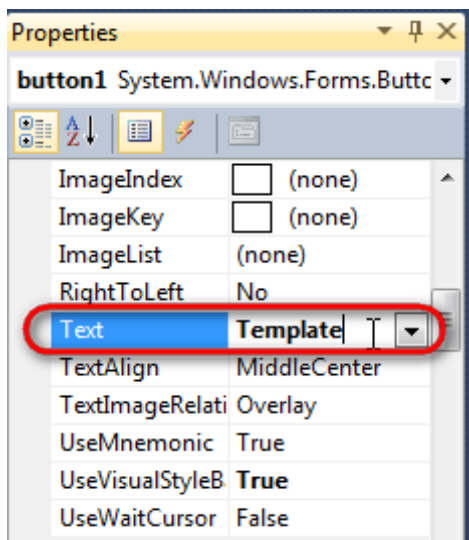
```

Step 19

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



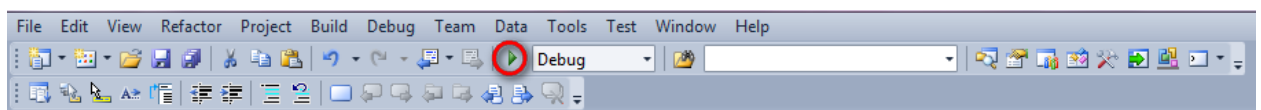
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

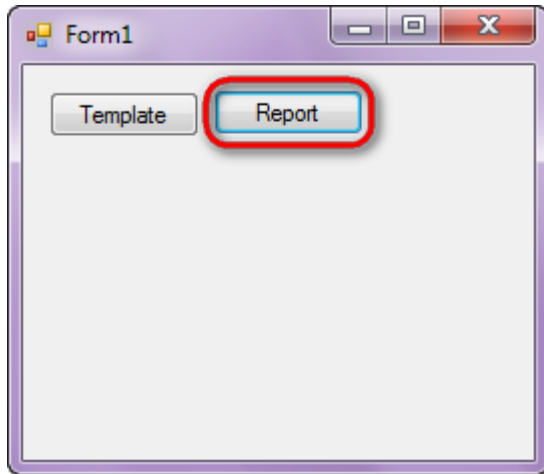
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 20

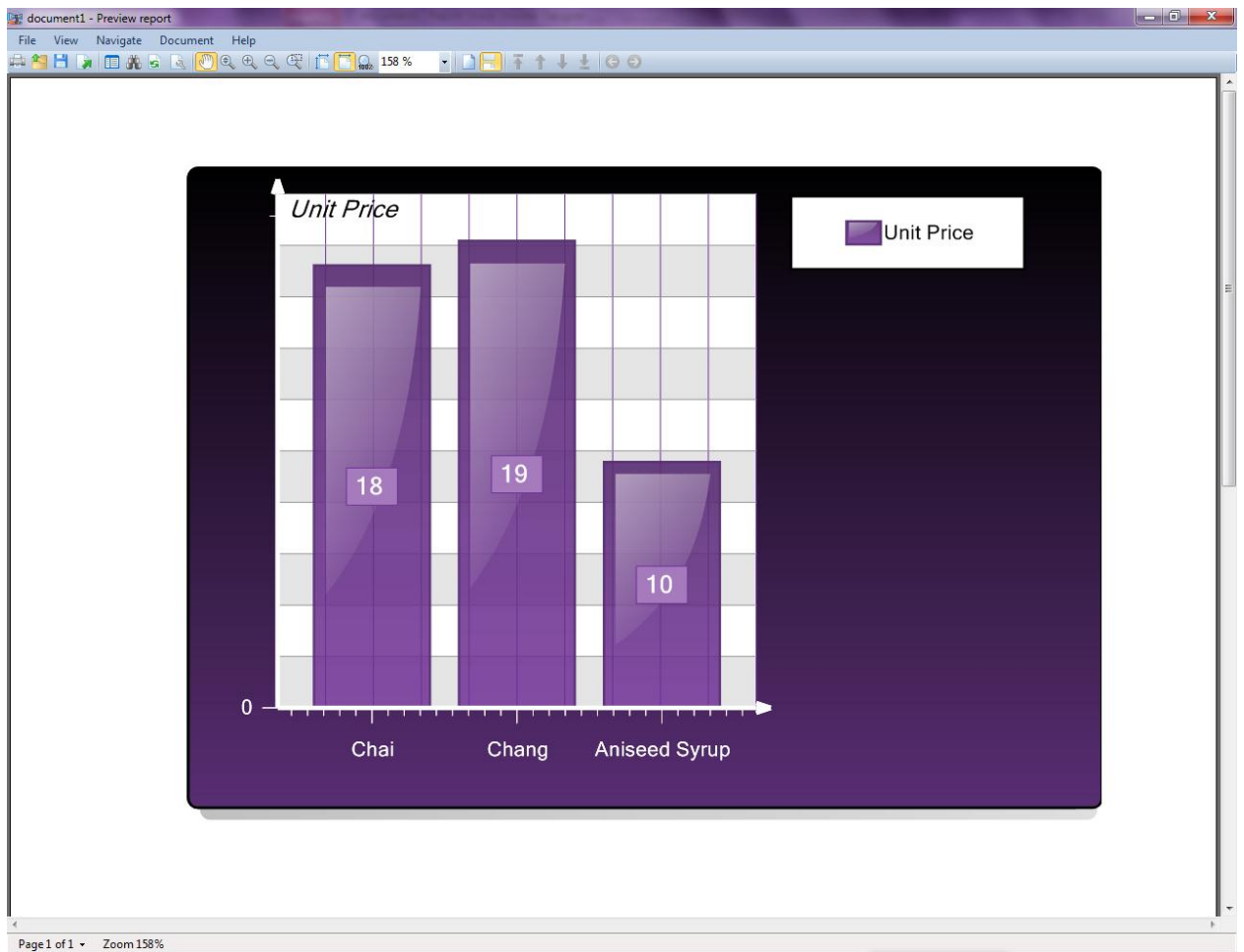
Click “Start Debugging” on the Visual Studio toolbar in order to run application.



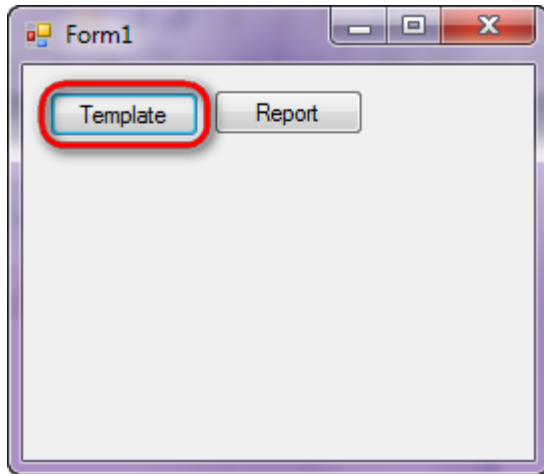
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar sample in the Samples Center is Report Controls\Charts\Chart on DataBand.

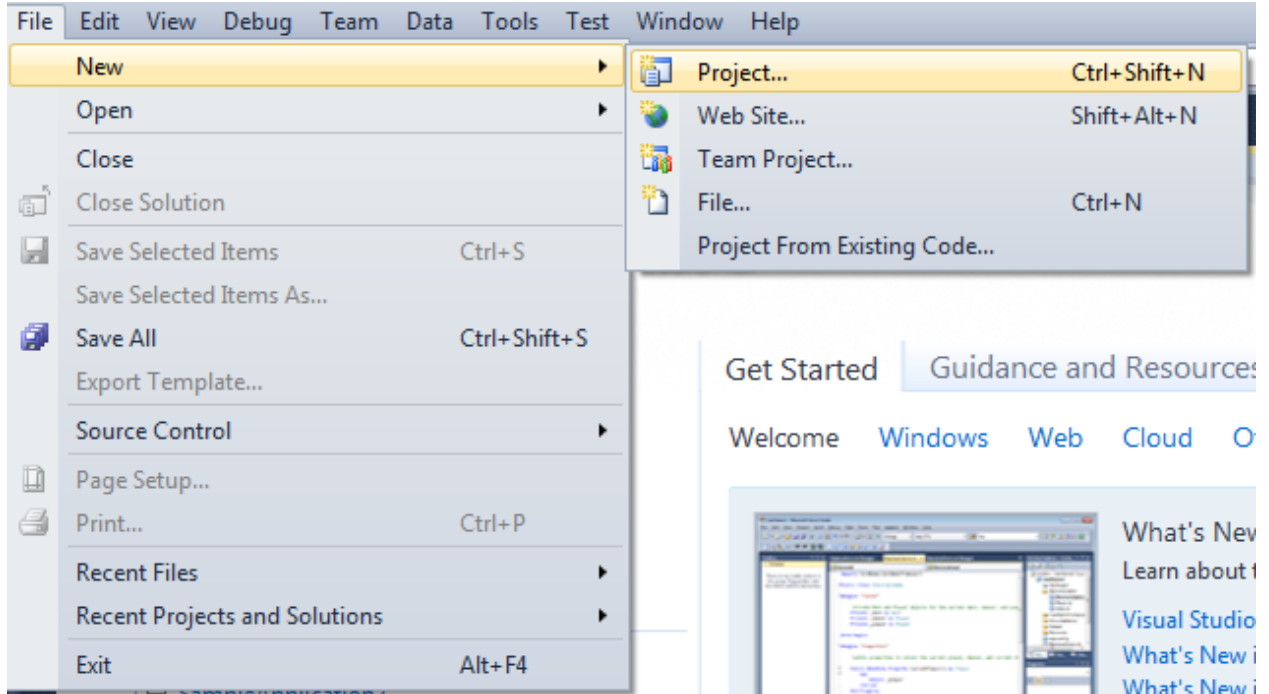


Widget

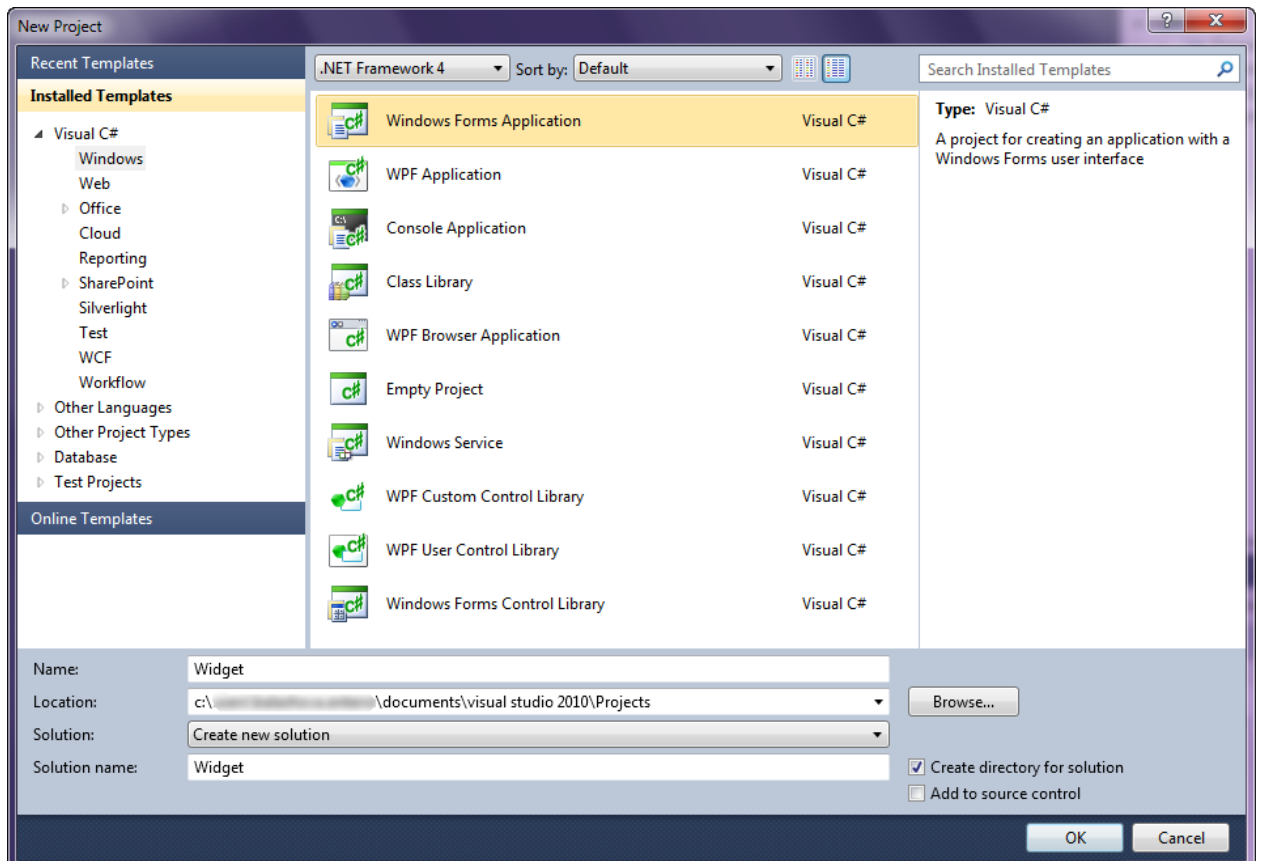
Template of a report containing widget.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

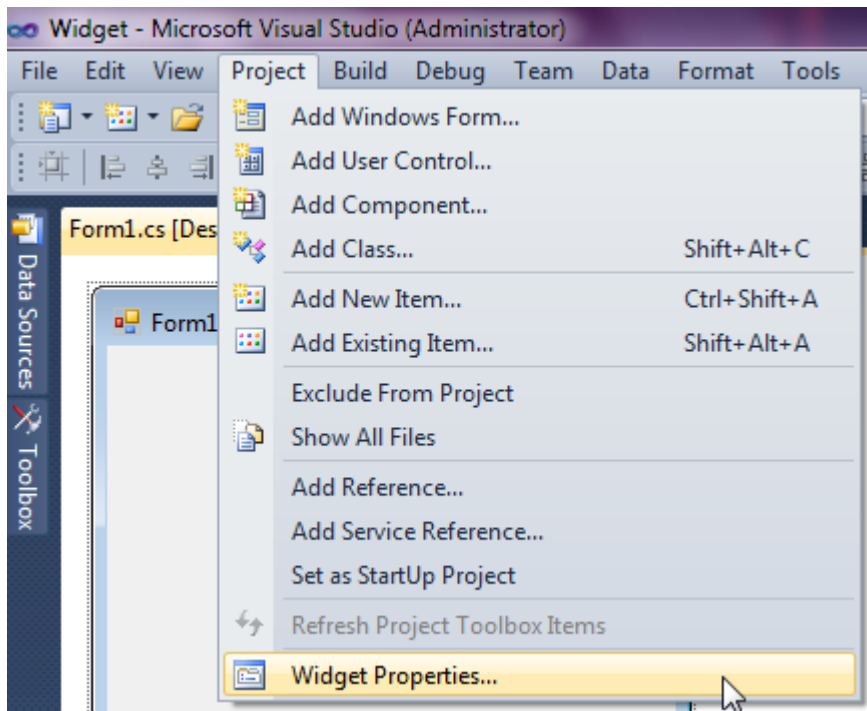


Select Windows Forms Application, set project name – “Widget”, set directory to save the project to.

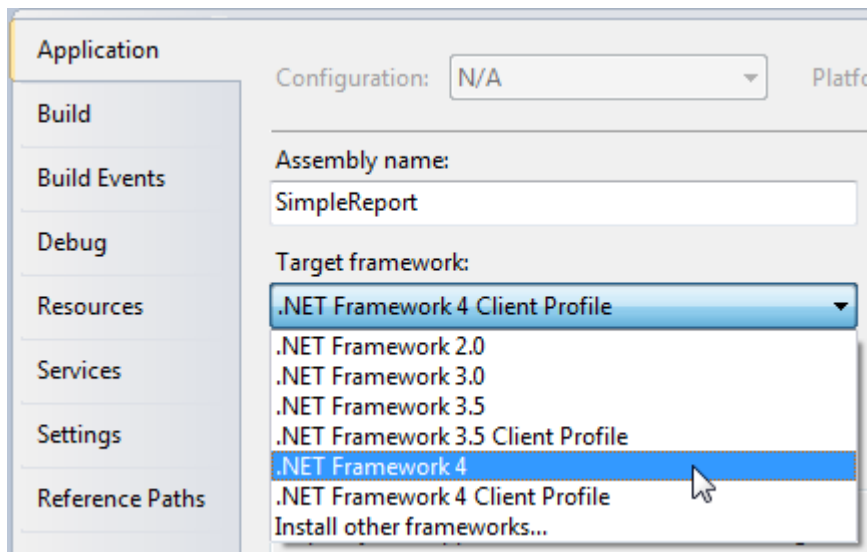


Step 2

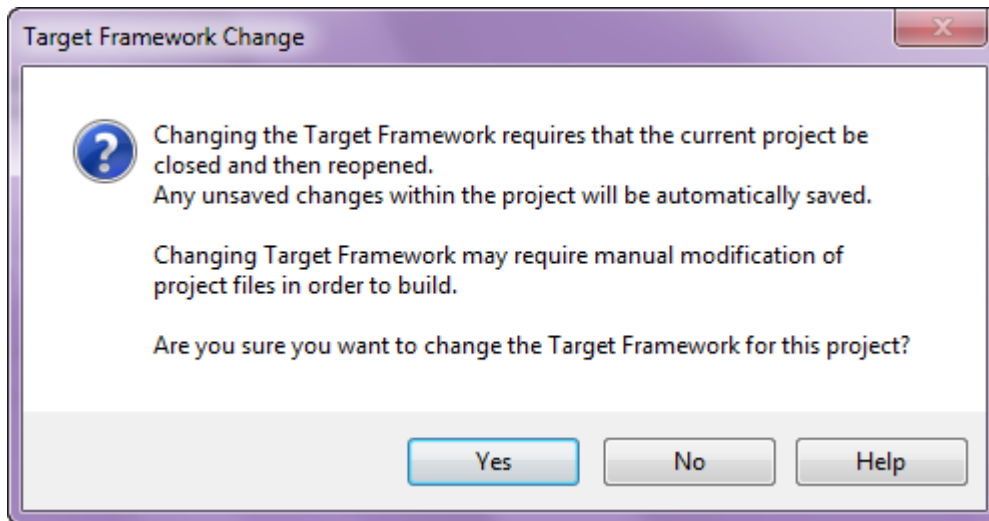
Change the project properties. Select the Project\Widget Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

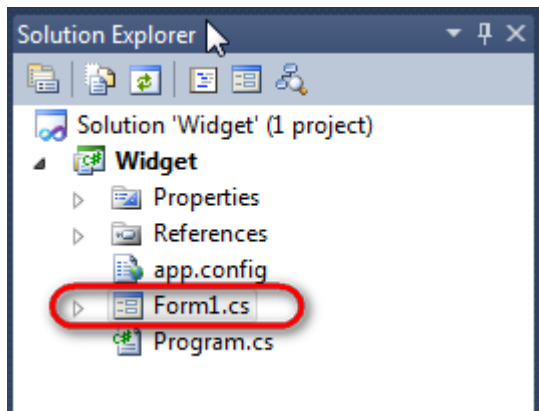


Press the "Yes" button in the opened window.

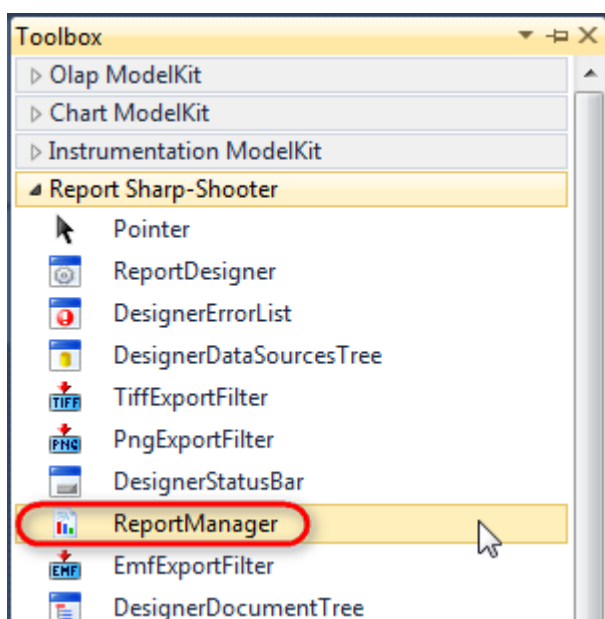


Step 3

Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

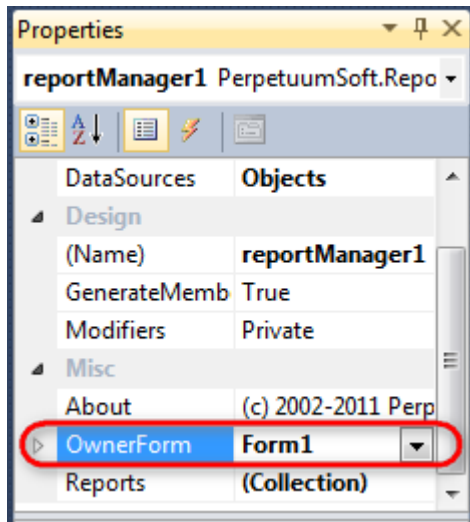


The component is available in the lower part of the window.



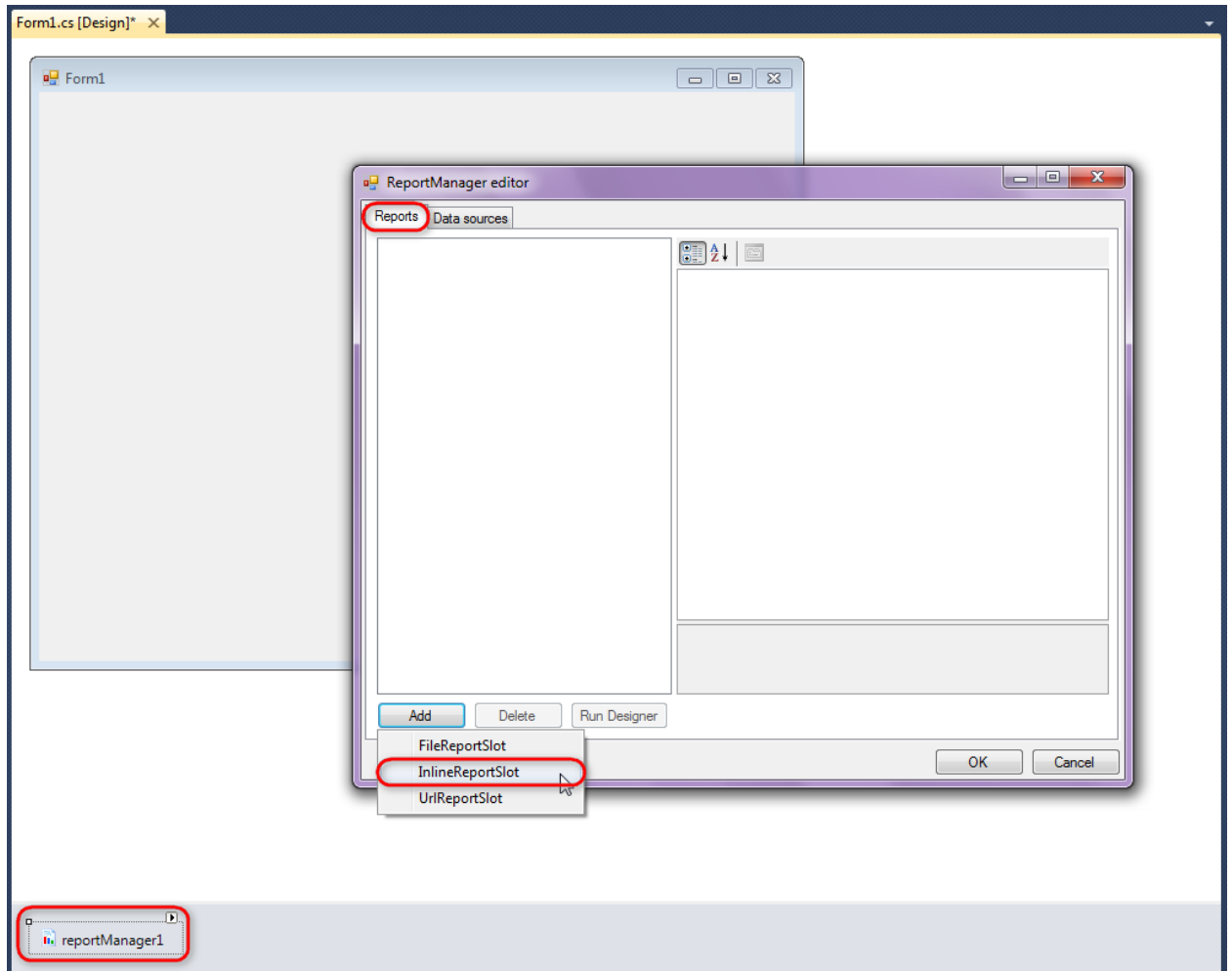
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

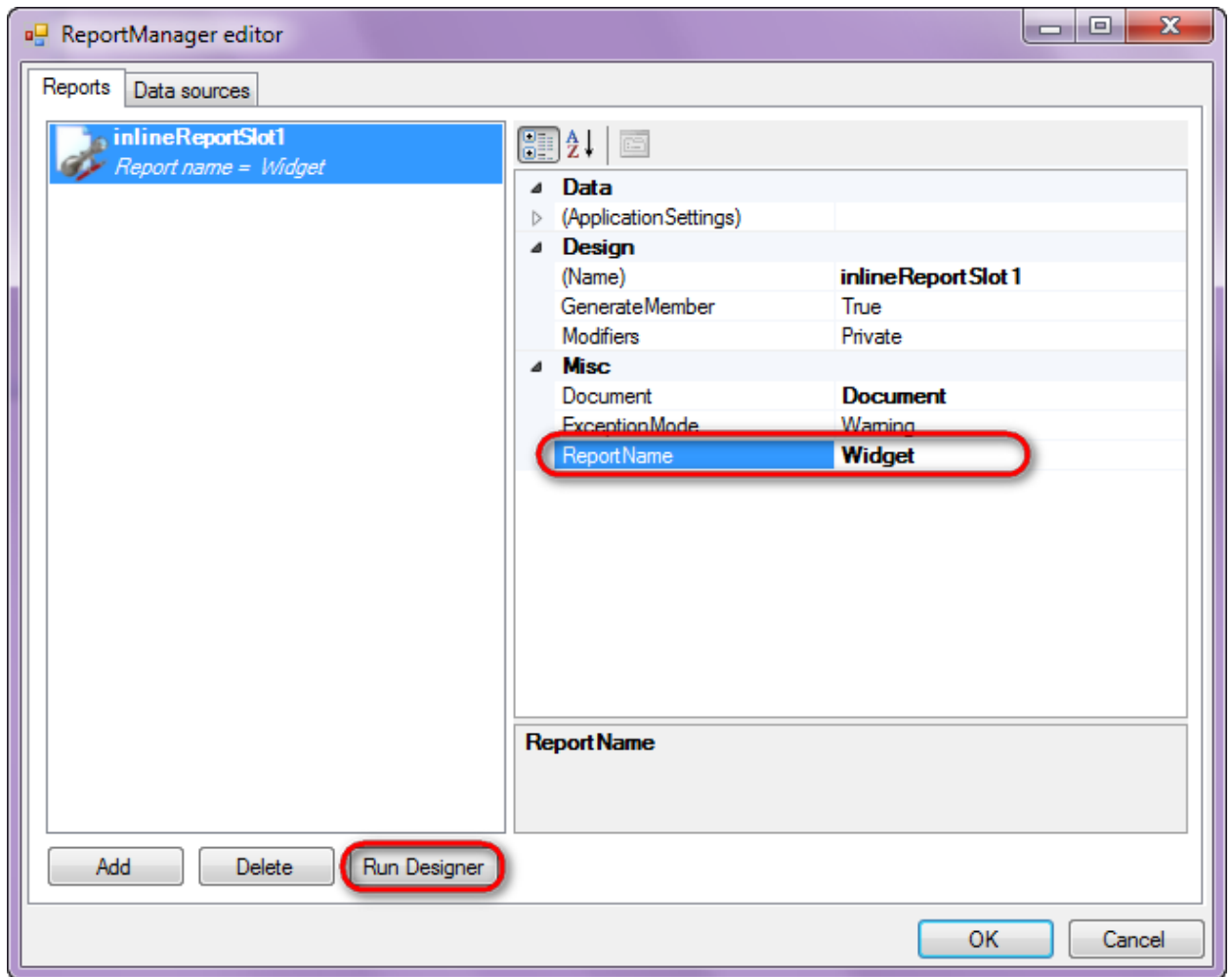


Go to "Reports" tab, click "Add" and select "InlineReportSlot".

Step 6

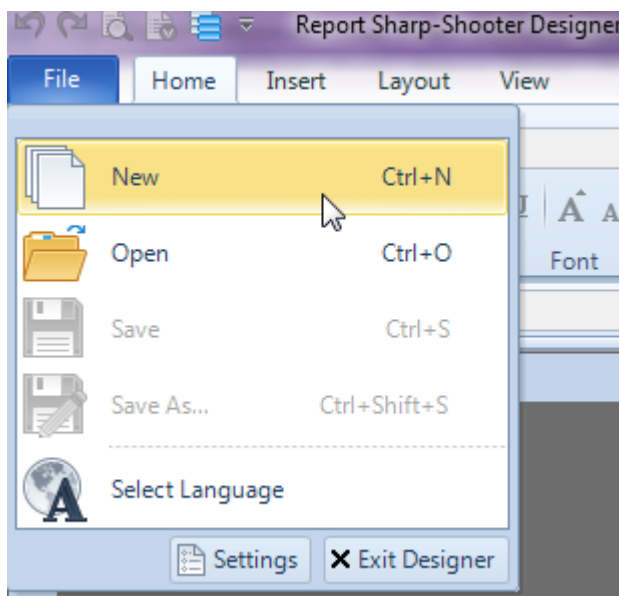
Set name of the report in the property ReportName - "Widget".

Click "Run Designer" in order to open template editor - Report Designer.

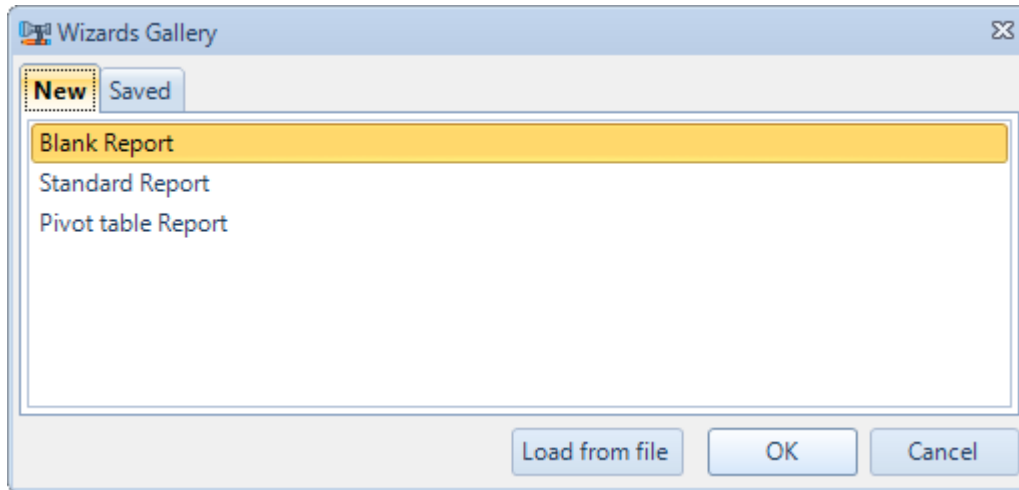


Step 7

Create new empty report – select File\New from the main menu.

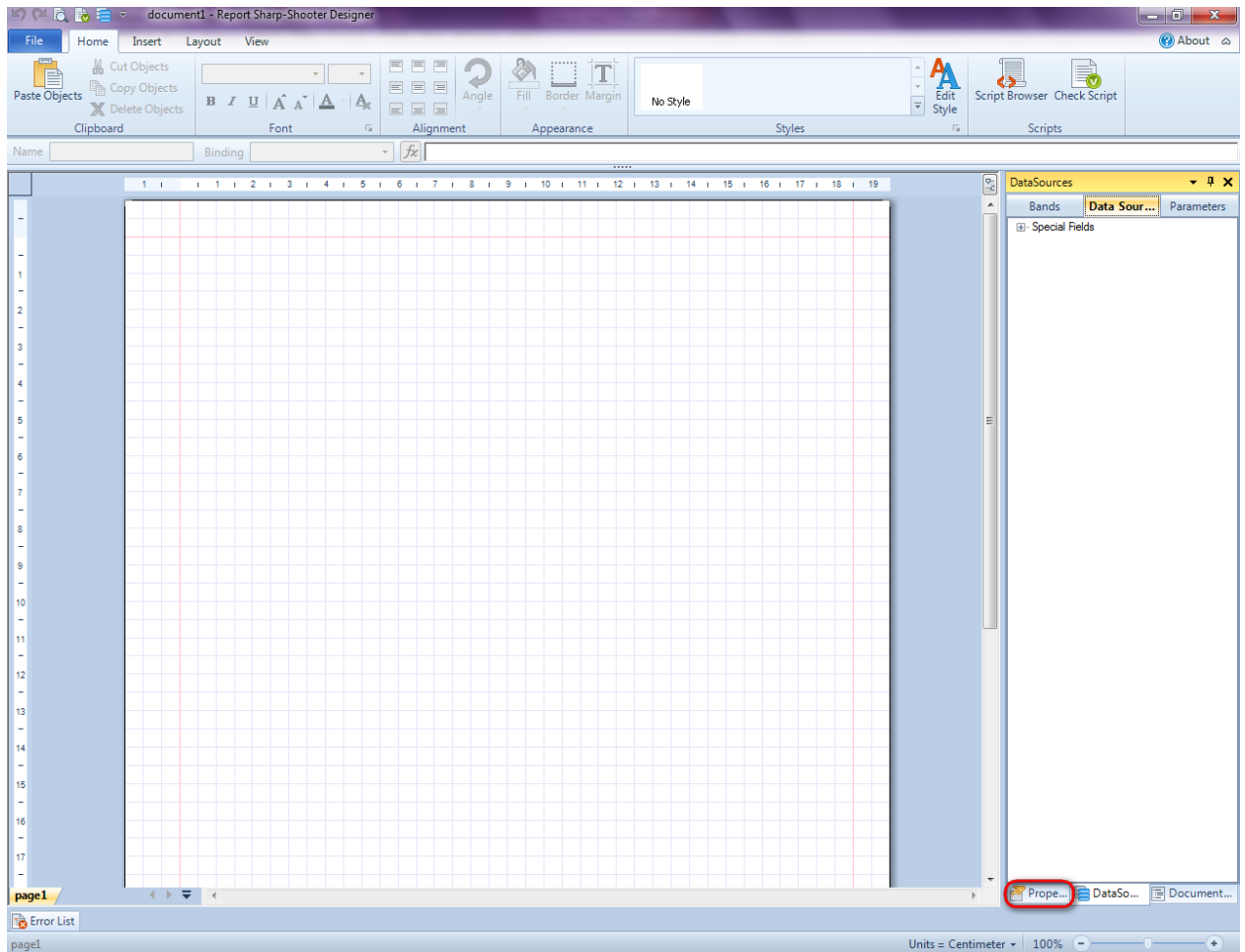


Select "Blank Report" in the Wizards Gallery and click "OK".



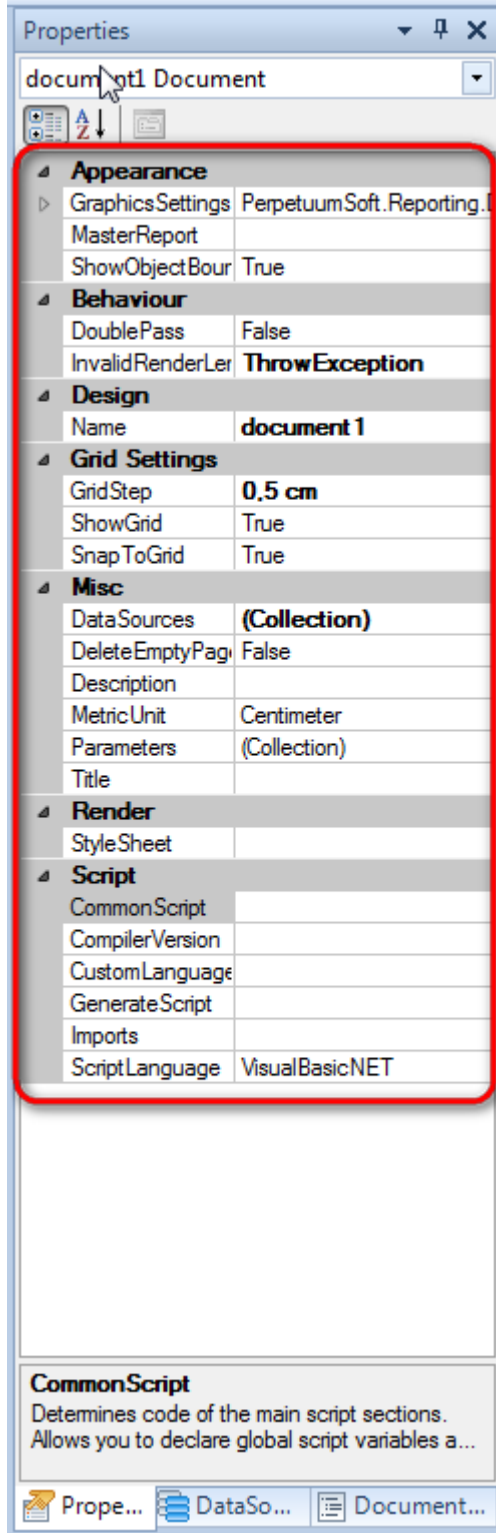
Step 8

Click the "Properties" tab of the tool window in the right part of the designer.

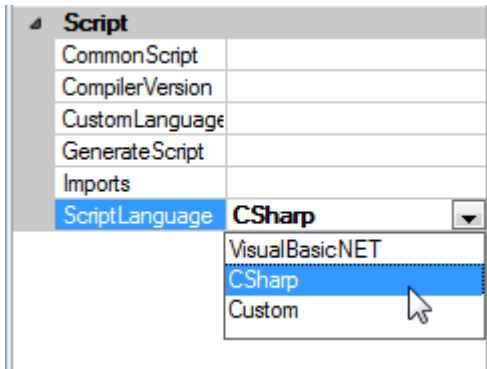




You will see properties of the edited template on the "Properties" tab

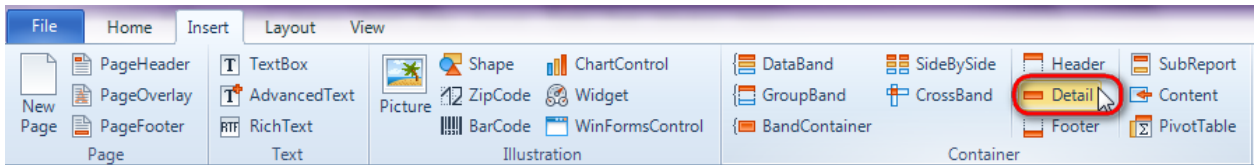


Set property ScriptLanguage = CSharp.



Step 9

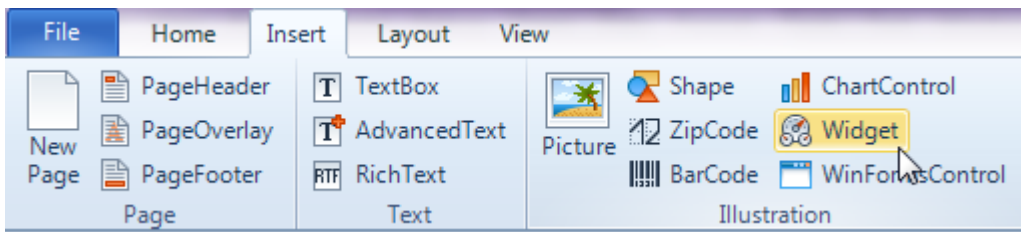
Press "Detail" button on the Insert tab in the group Container.



Click on the template area to add the band to the template.

Step 10

Press "Widget" button on the Insert tab in the group Illustration.

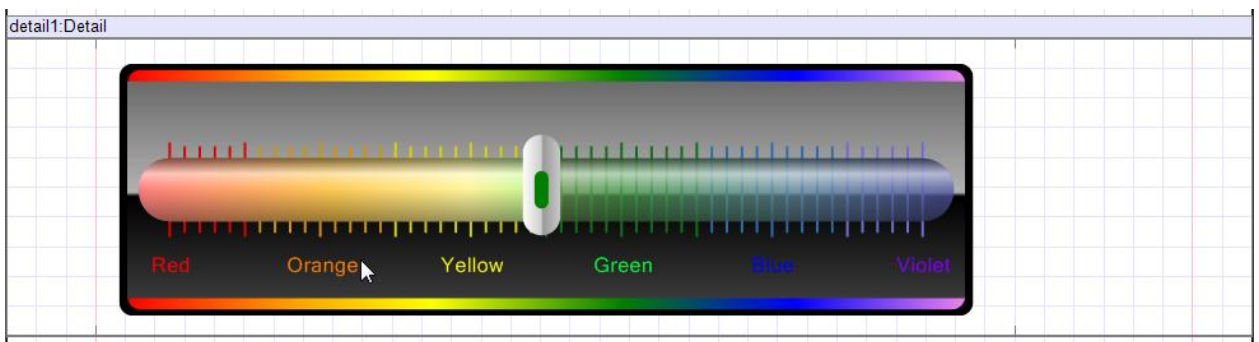


Click on the Detail band area to add Widget component inside Detail.

Double click on the Widget element to open Instrument designer in order to design a widget. Find more information on creating widgets in Instrument designer in Section Instrument designer hereof.

After the Widget is created close Instrument designer by clicking "OK"

Report template should as follows:

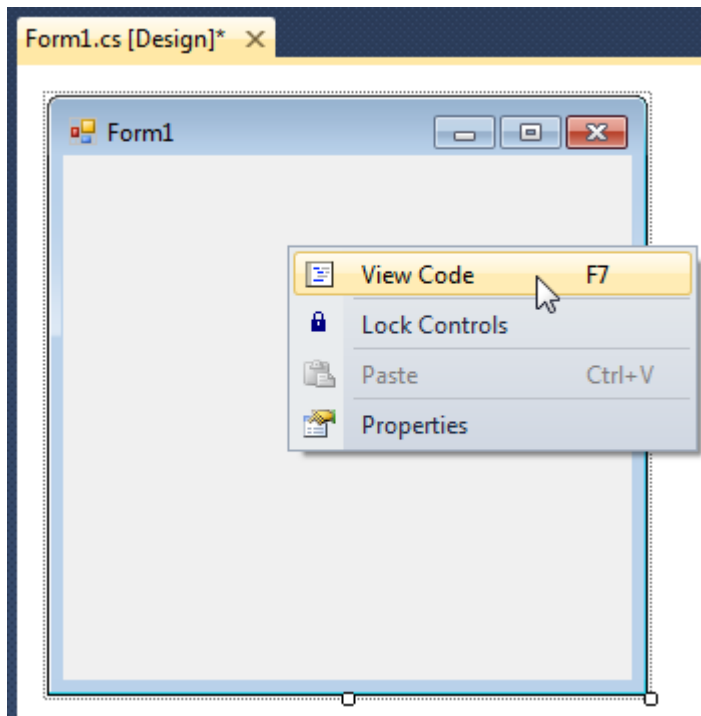


Step 11

Save template and close Report Designer.

Step 12

Right click on the application form and select "View Code" in the context menu in order to view code.

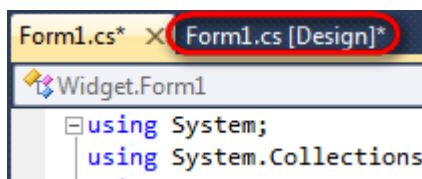


Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

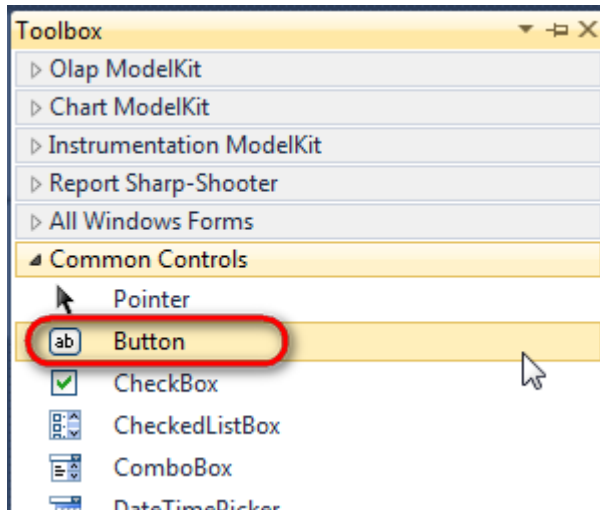
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted (object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 13

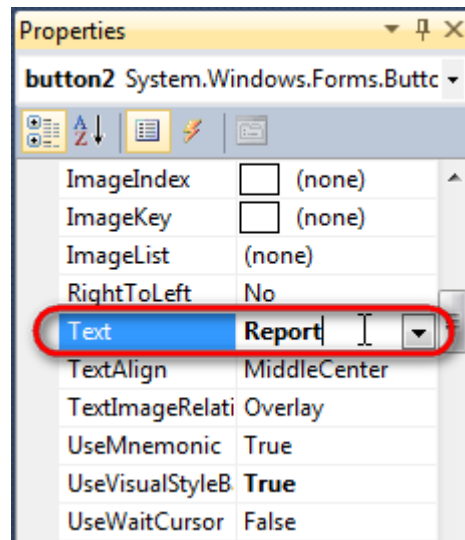
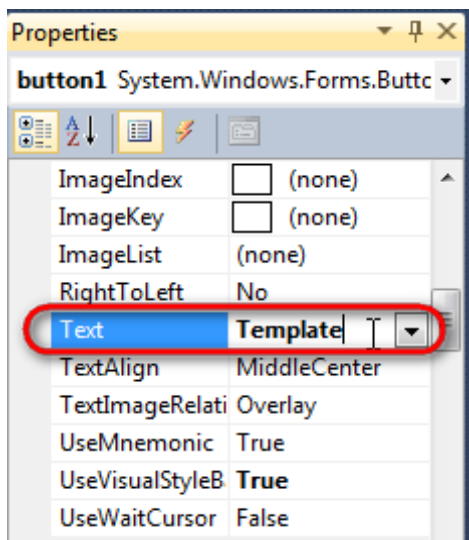
Get back to application form by clicking "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



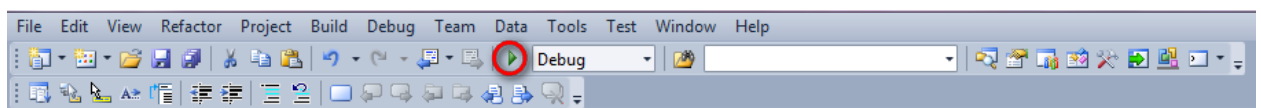
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

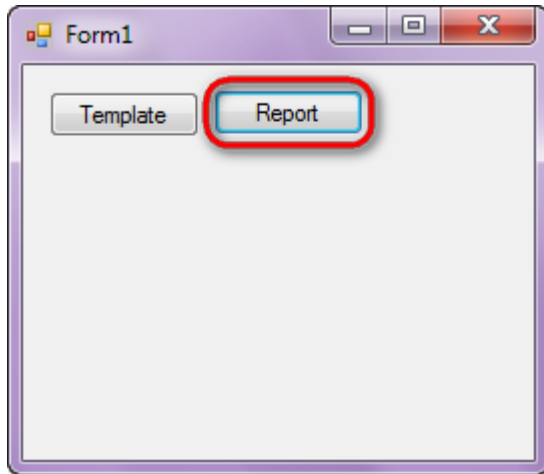
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 14

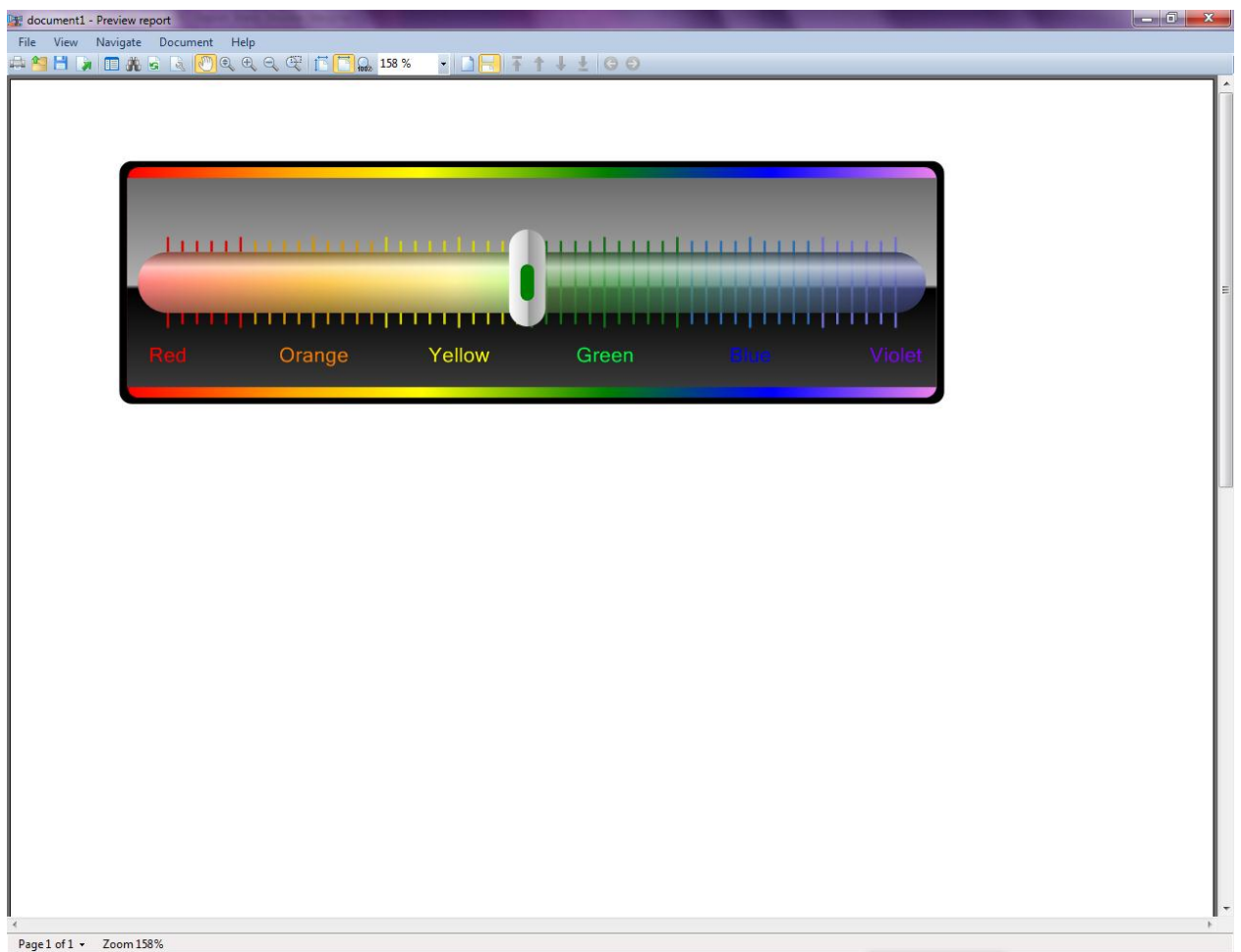
Click “Start Debugging” on the Visual Studio toolbar in order to run application.



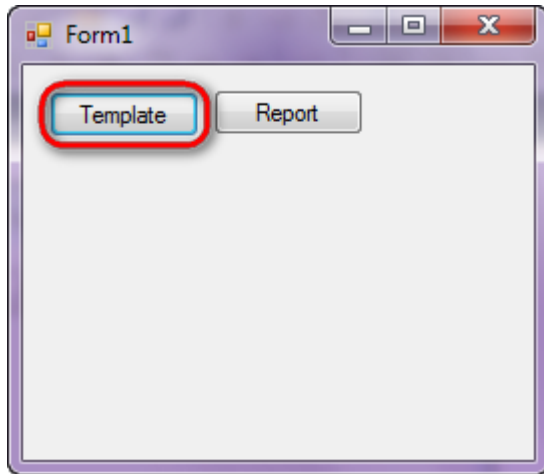
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar sample in the Samples Center is Report Controls\Instruments\Instruments types.

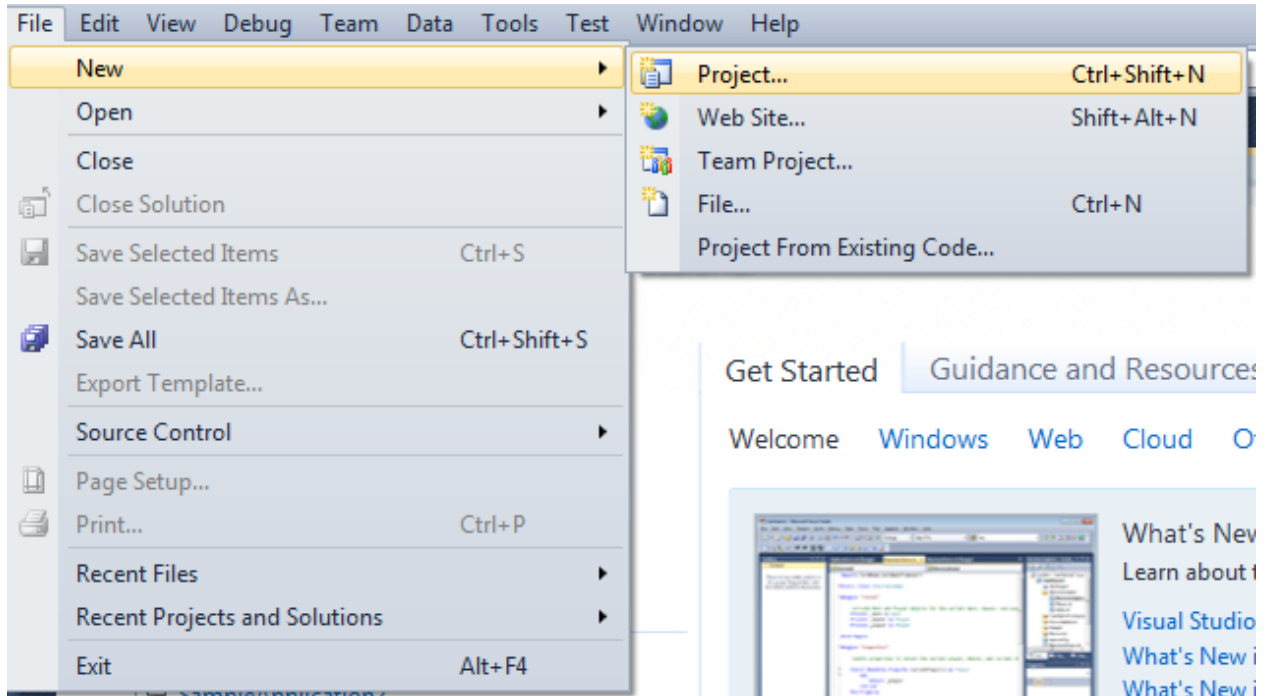


Advanced Text

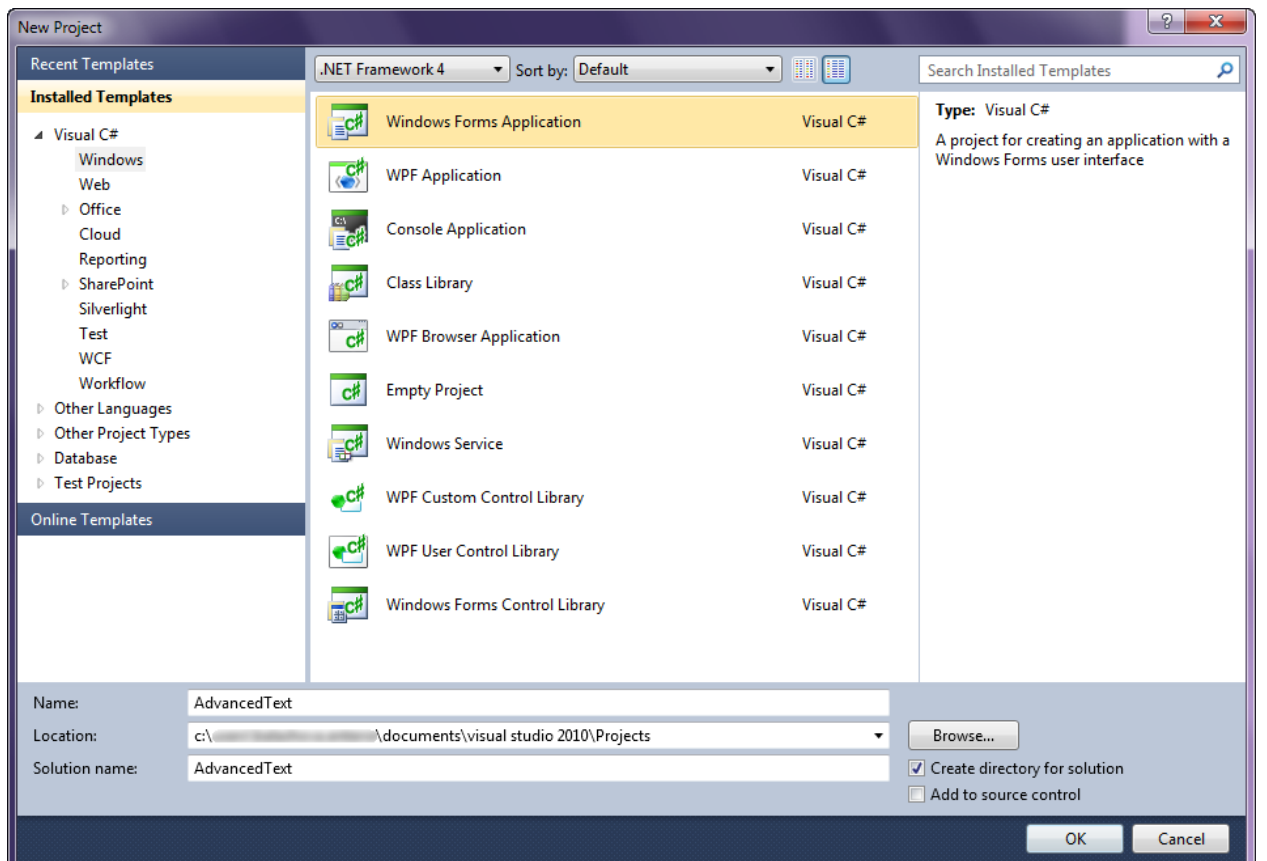
Template of a report containing formatted text.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

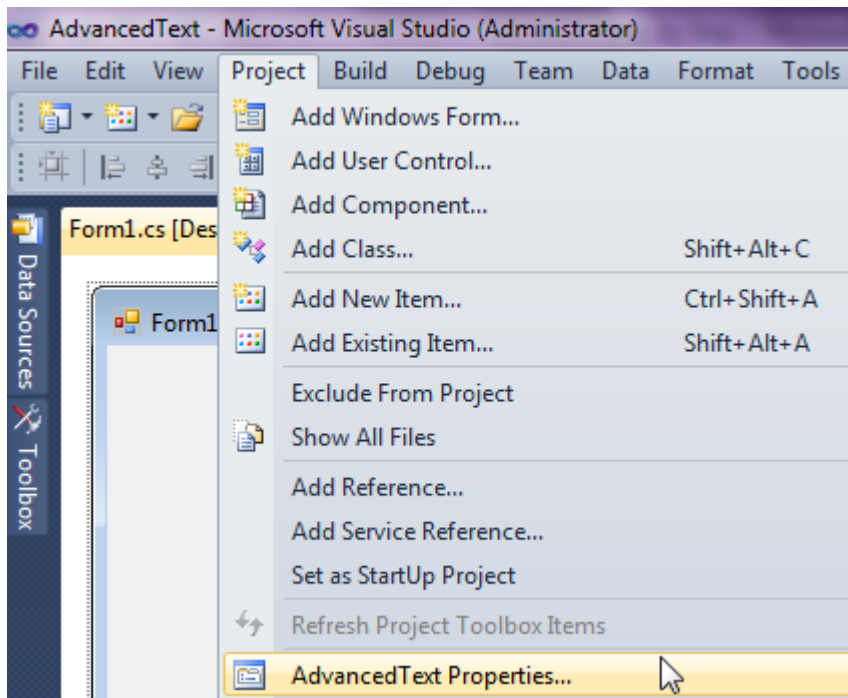


Select Windows Forms Application, set project name – “AdvancedText”, set directory to save the project to.

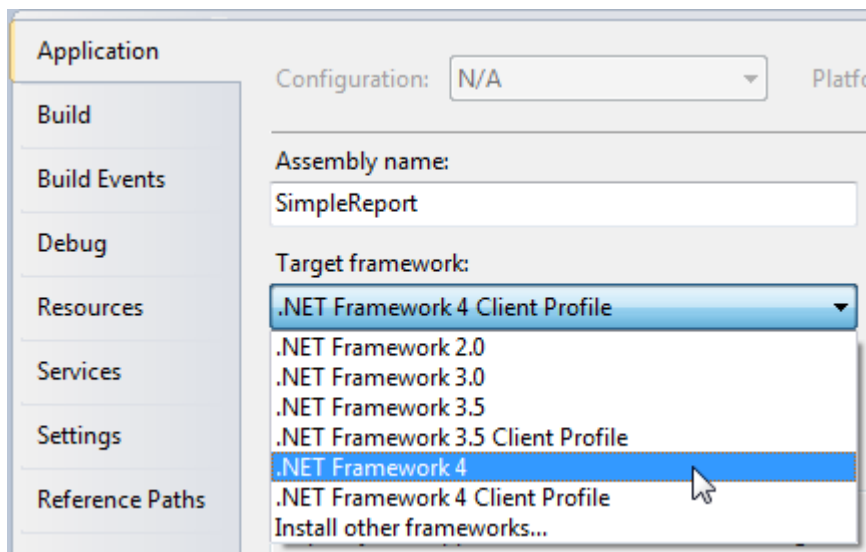


Step 2

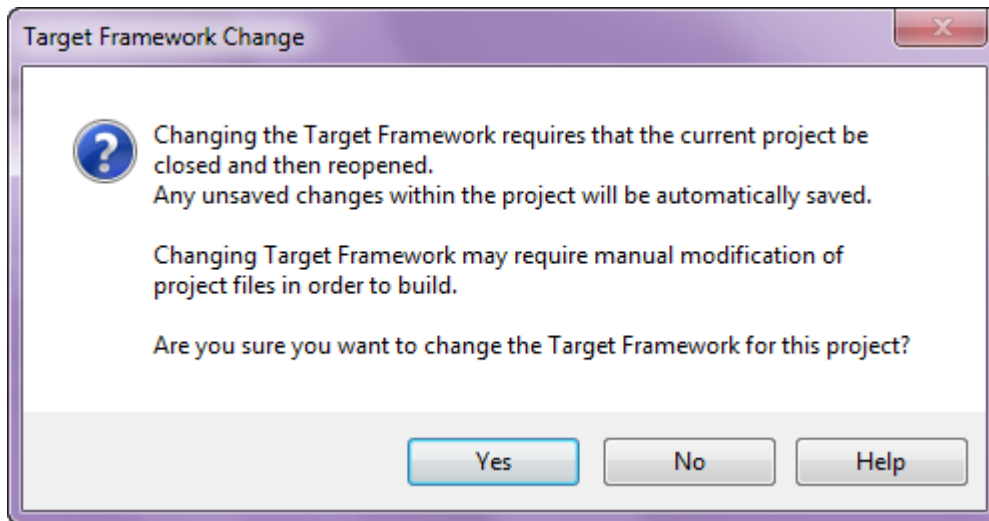
Change the project properties. Select the Project\AdvancedText Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

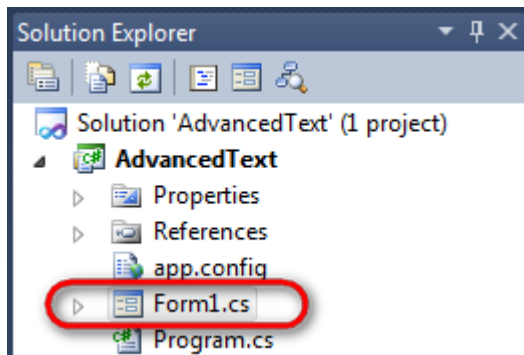


Press the "Yes" button in the opened window.

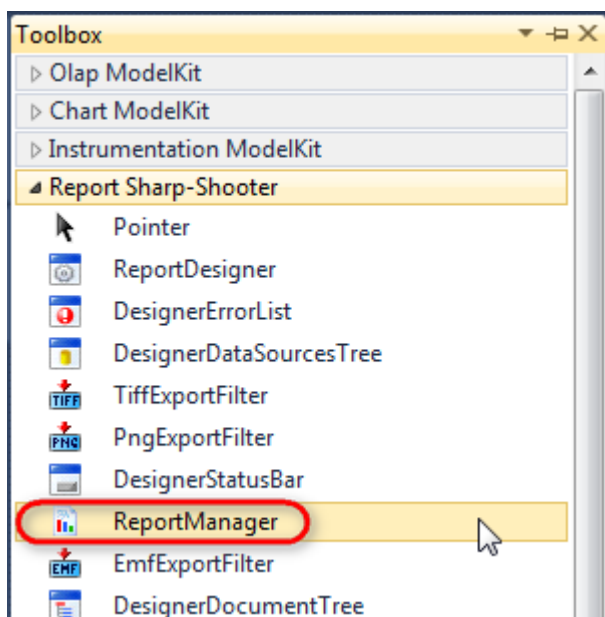


Step 3

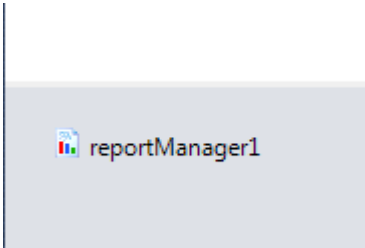
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" element on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

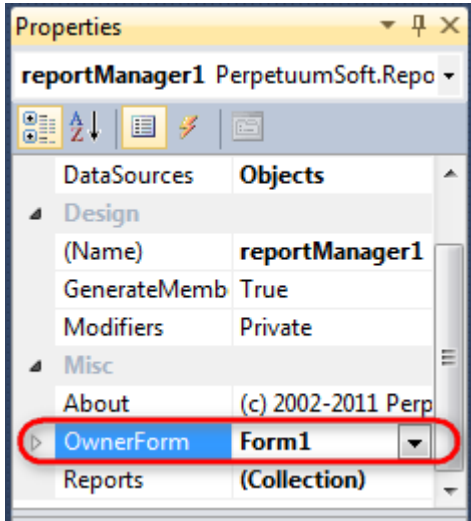


The component is available in the lower part of the window.



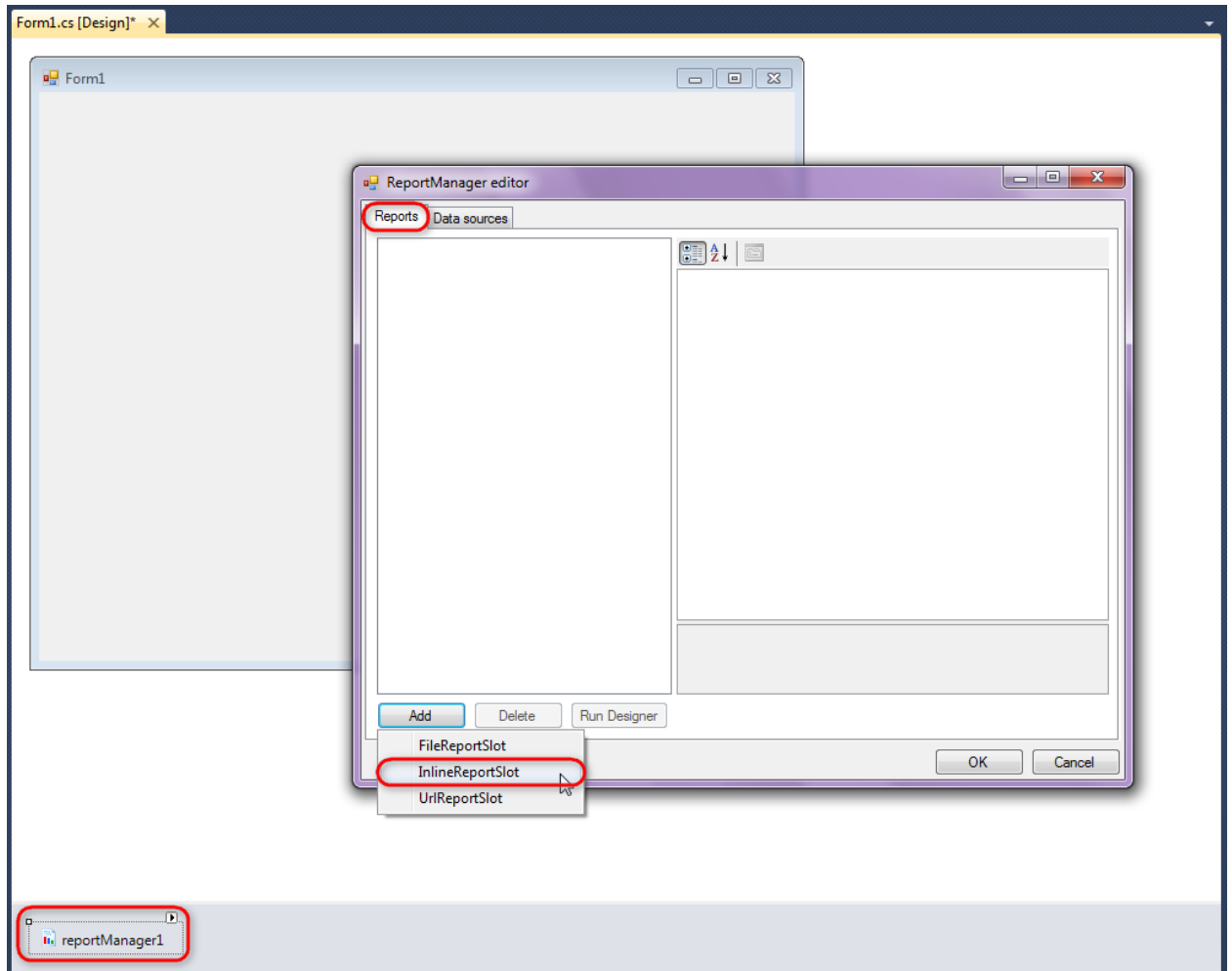
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

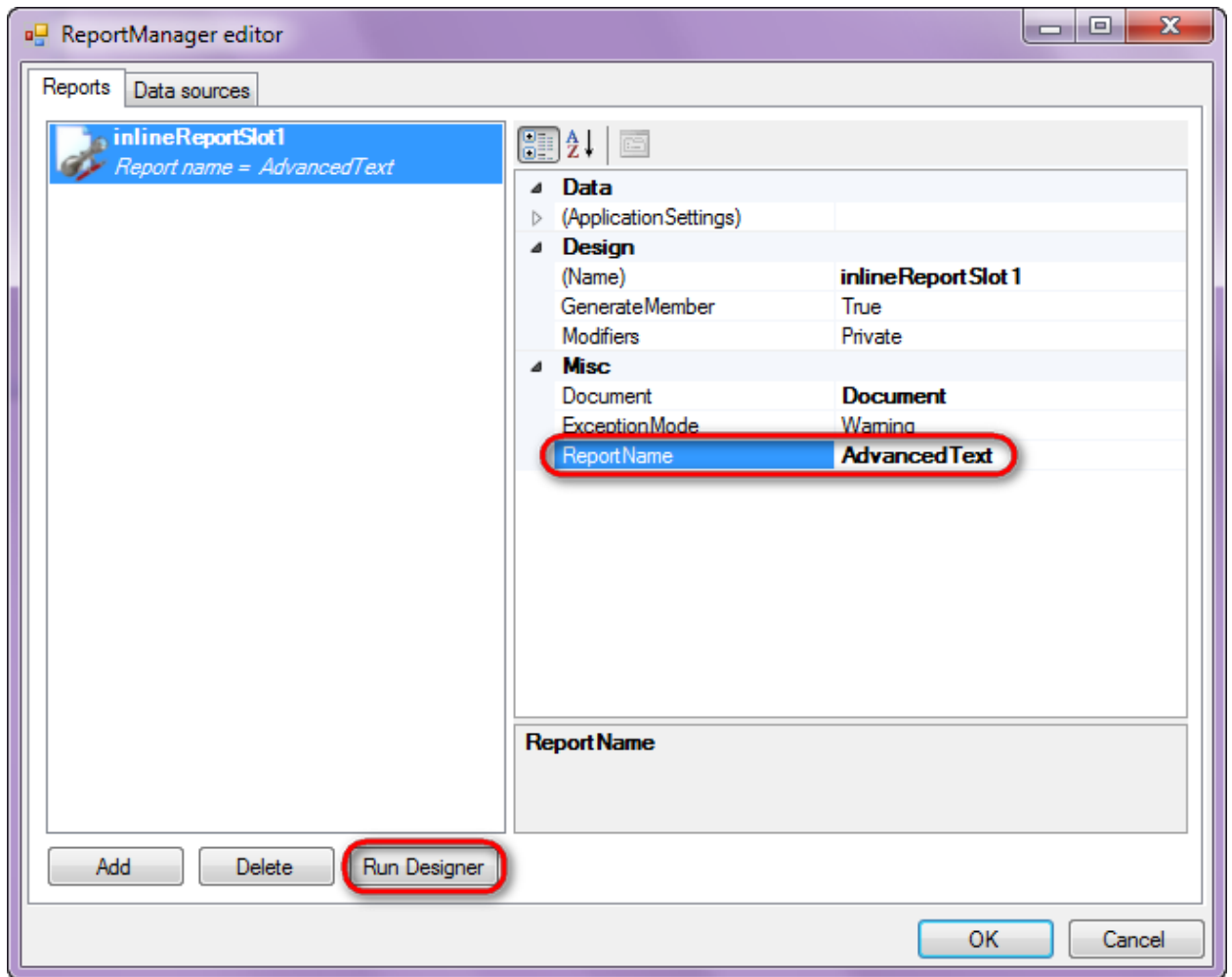


Go to "Reports tab", click "Add" and select "InlineReportSlot".

Step 6

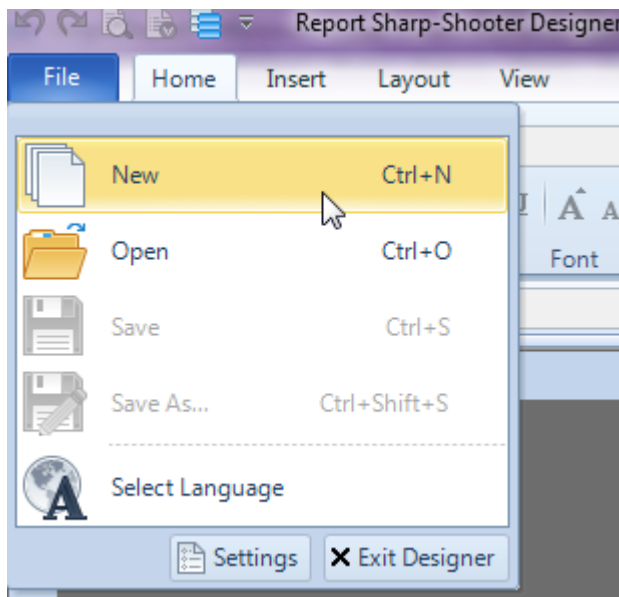
Set name of the report in the property ReportName – "AdvancedText".

Click "Run Designer" in order to open template editor - Report Designer.

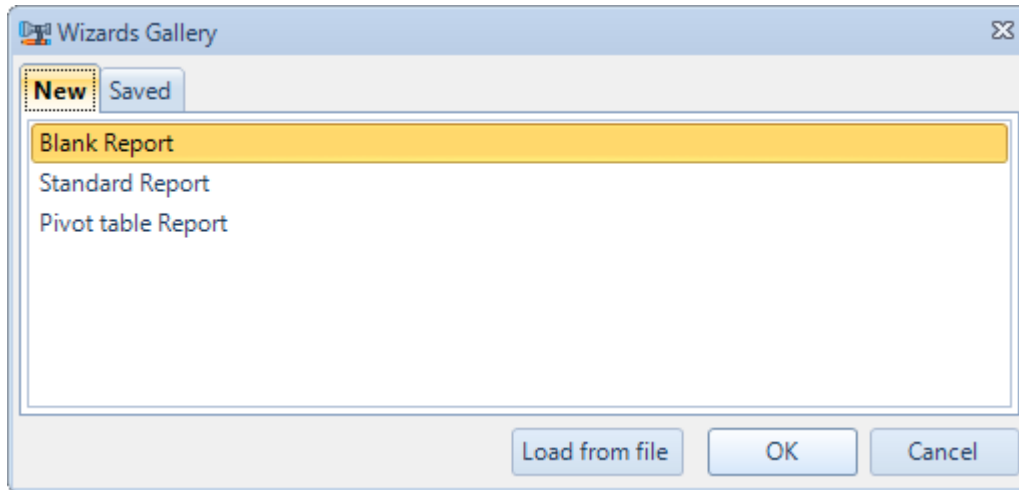


Step 7

Create new empty report – select File\New from the main menu.

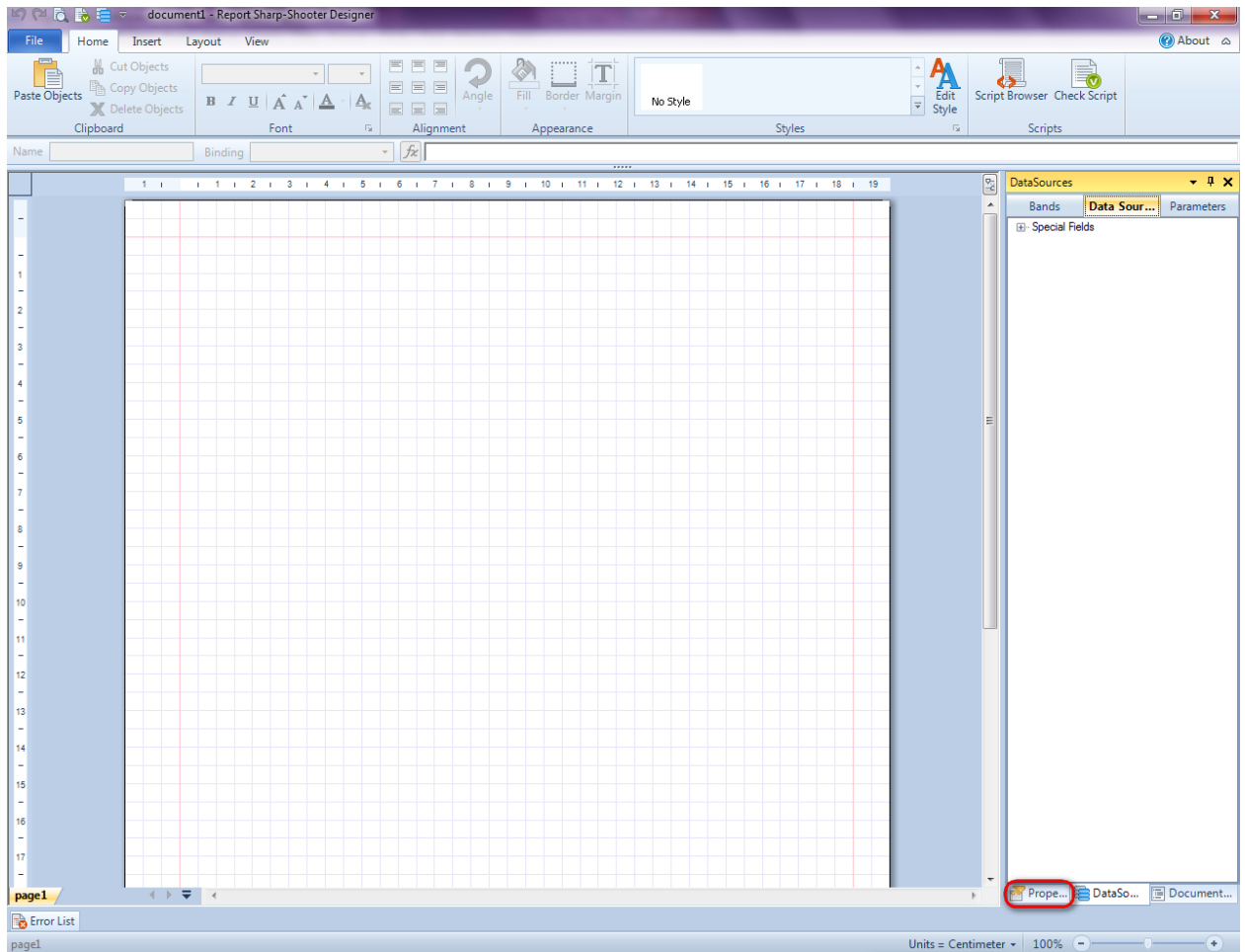


Select "Blank Report" in the Wizards Gallery and click "OK".

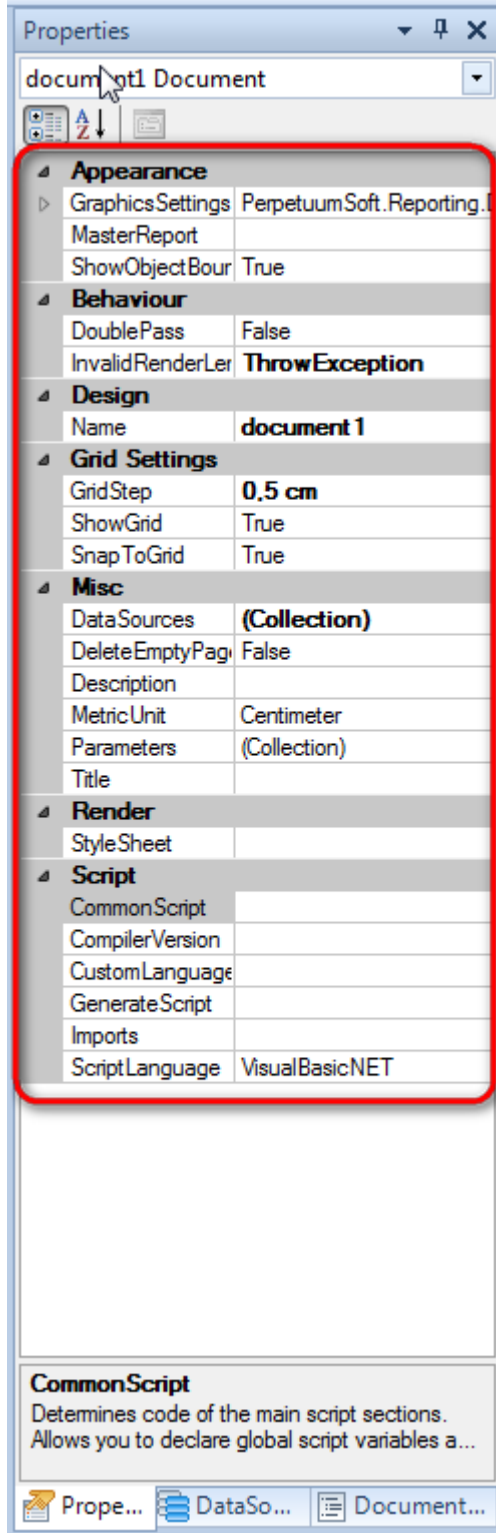


Step 8

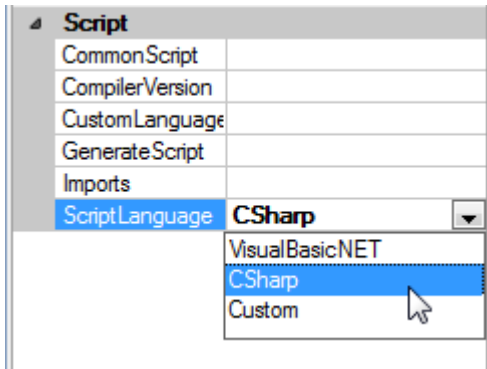
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

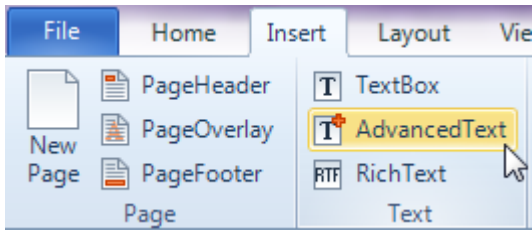


Set property ScriptLanguage = CSharp.



Step 9

Press "AdvancedText" button on the Insert tab in the group Text.

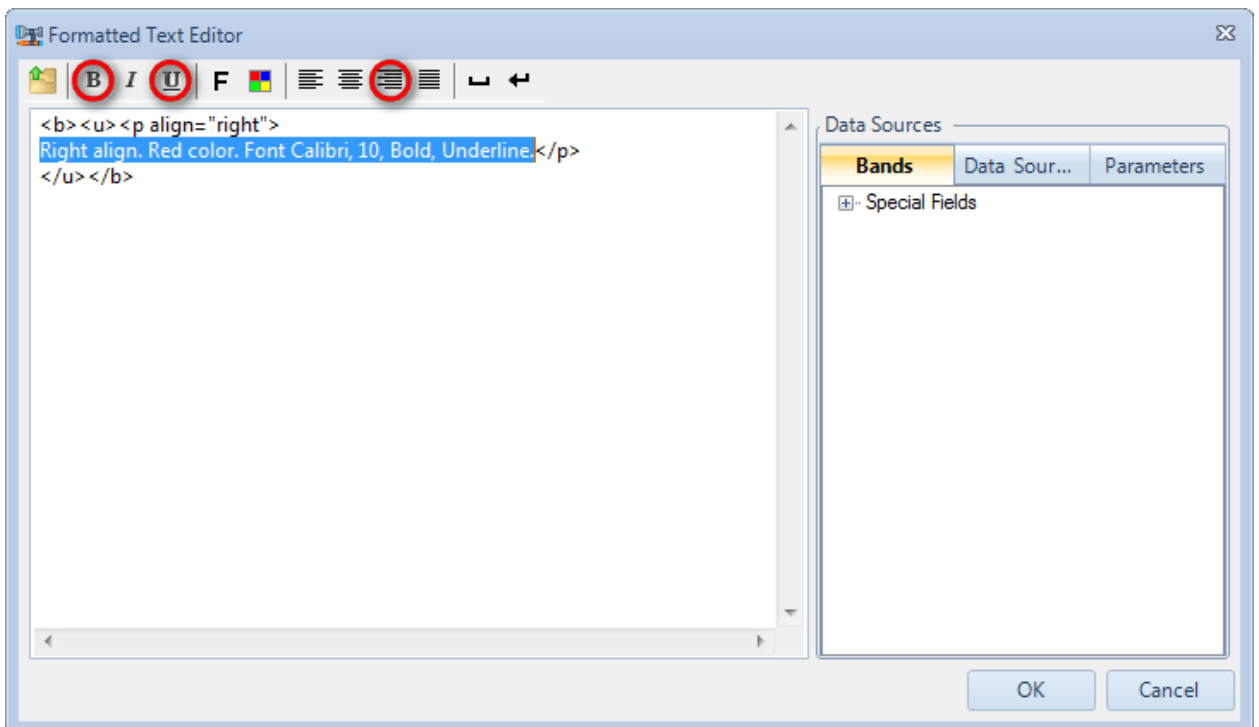


Click on the template area to add AdvancedText to the template.

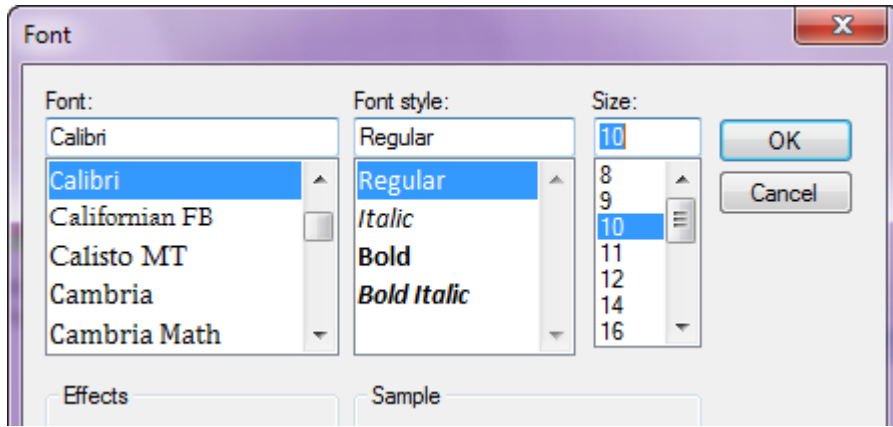
Step 10

Double click on the AdvancedText element area and open Formatted Text Editor. Add the following text "Right align. Red color. Font Calibri, 10, Bold, Underline." Select it.

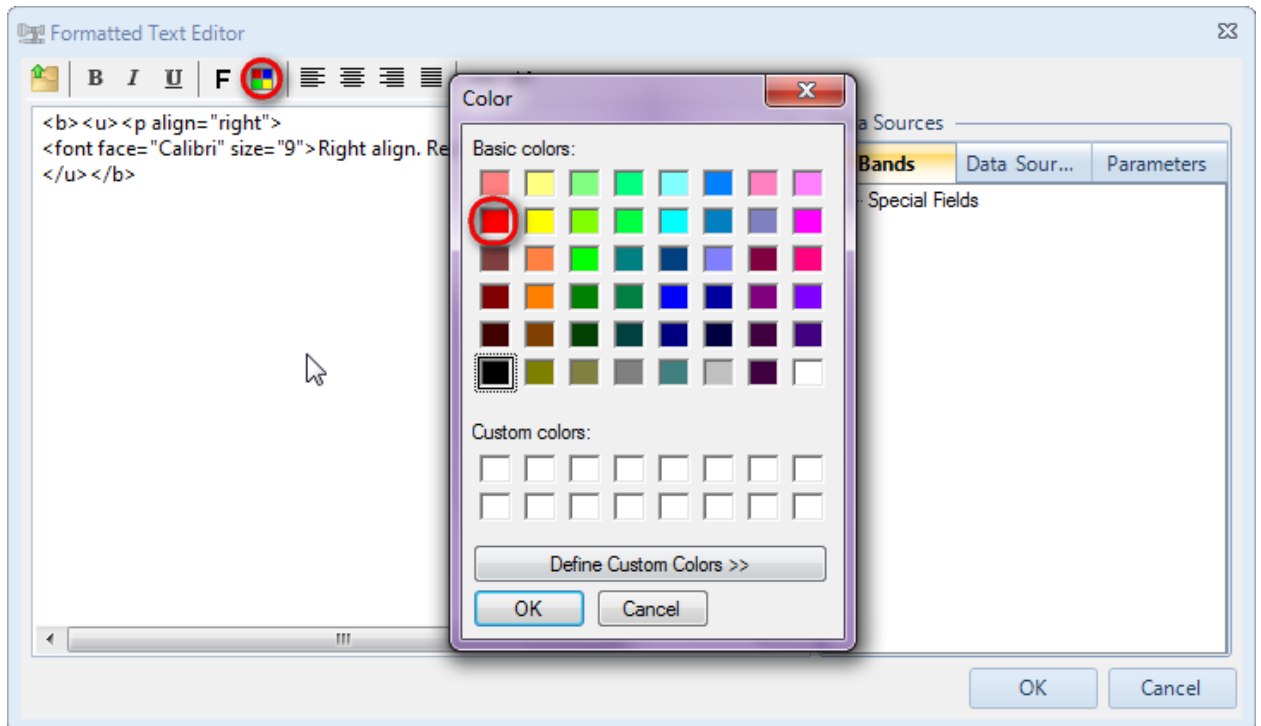
Click those buttons "Bold", "Underline" and "Right align". Corresponding tags are added to the text.



Click "Font" button to open font editor. In the Fonts list, select "Calibri", set Size to "10".



Click "Color" button, select red color on the Color form.



Step 11

Set cursor at the end of the text in the Formatted Text Editor. Click "Insert line break" button.

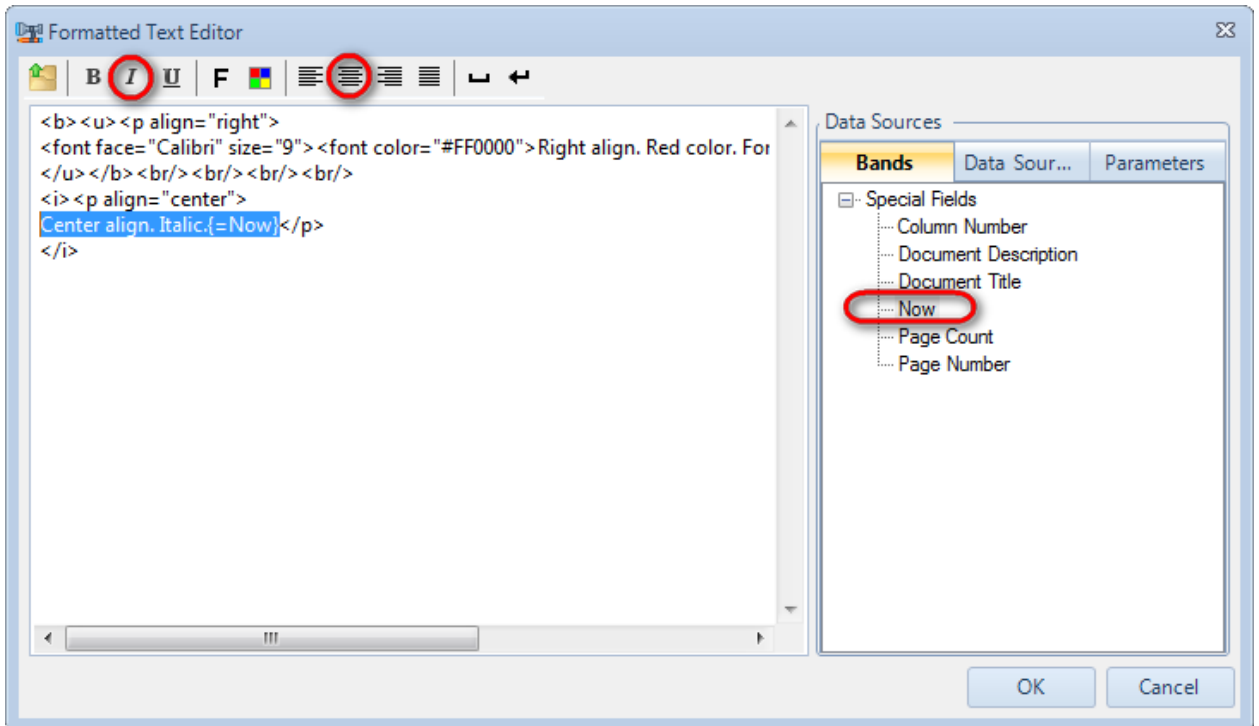


Add several line breaks in this way.

Step 12

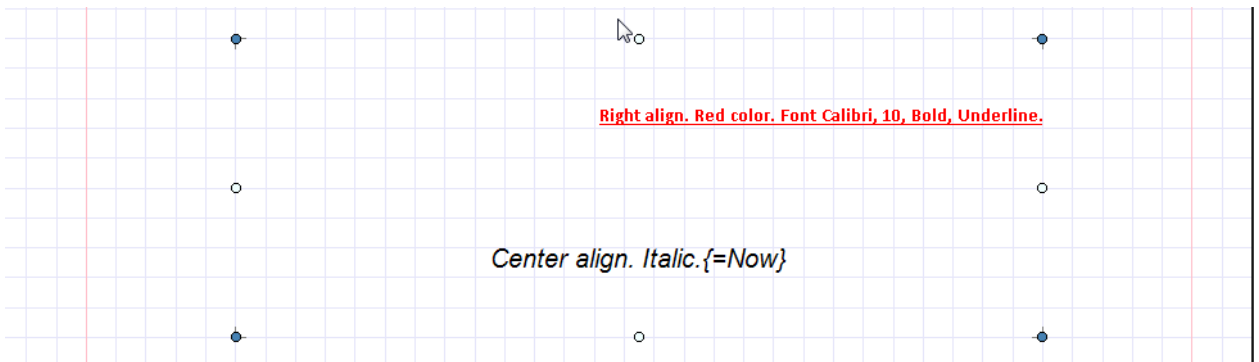
In the Formatted Text Editor, add the following text "Center align. Italic." Add by double click "Now" field from the Special Fields tree (in the Data Sources area).

Select the added text and click "Italic" and "Center align" buttons.



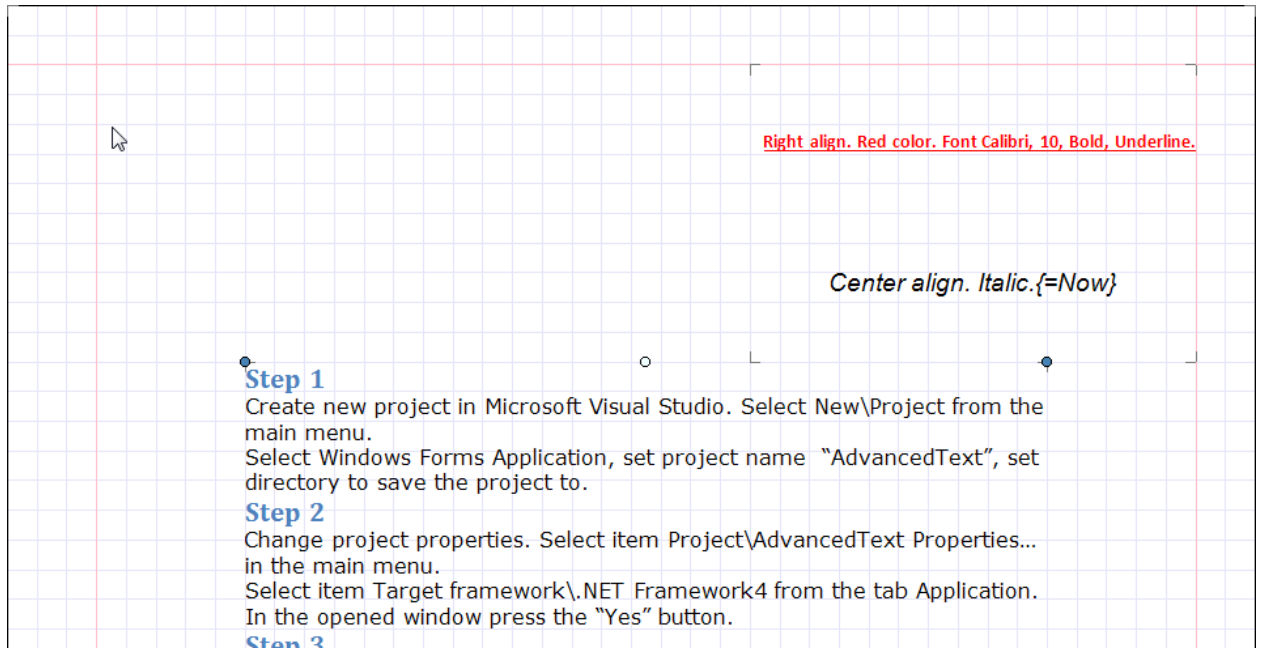
Step 13

Close Formatted Text Editor by clicking "OK". Change size of the AdvancedText element so that the text is completely visible.



Step 14

Add one more AdvancedText element. Open Formatted Text Editor. Click "Open RFT Document" button. Select RTF file. Close editor, change size of the AdvancedText element to view the whole text.

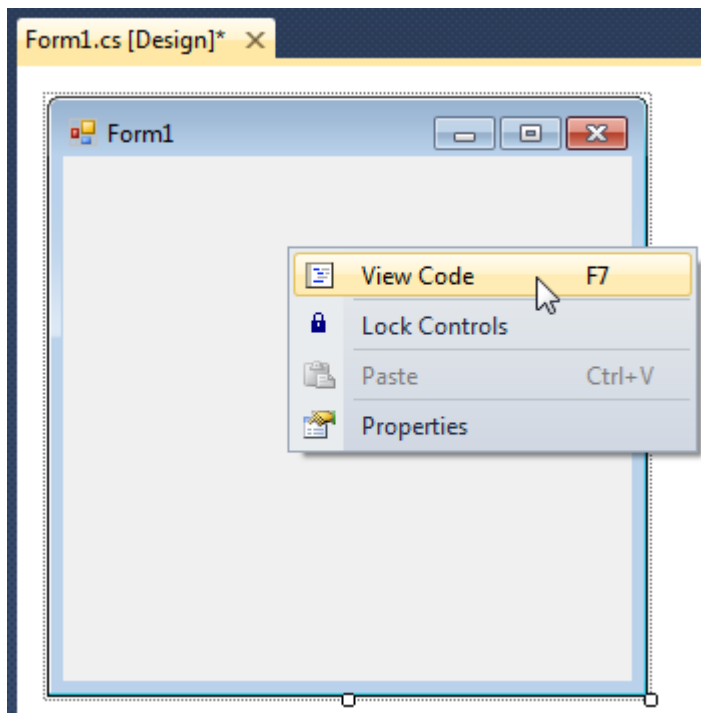


Step 15

Save template, close Report Designer.

Step 16

Right click on the application form and select "View Code" in the context menu in order to view code.



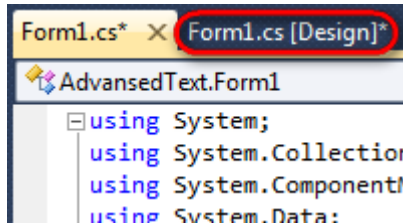
Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()  
{  
    InitializeComponent ();  
    inlineReportSlot1.RenderCompleted += new  
EventHandler (reportSlot_RenderCompleted);  
}
```

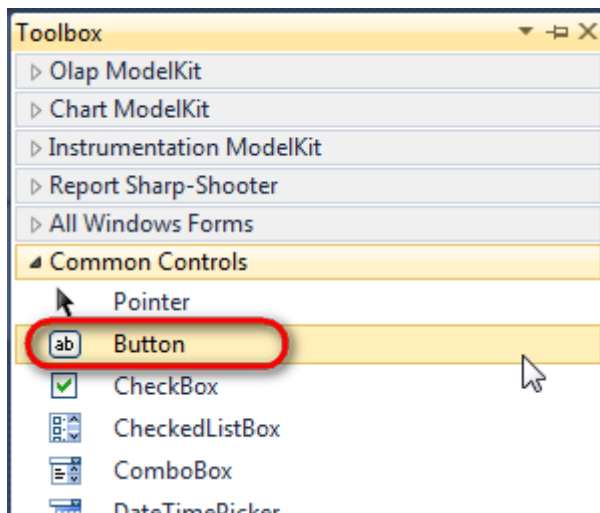
```
}  
private void reportSlot_RenderCompleted(object sender, EventArgs e)  
{  
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new  
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))  
    {  
        previewForm.WindowState = FormWindowState.Maximized;  
        previewForm.ShowDialog(this);  
    }  
}
```

Step 17

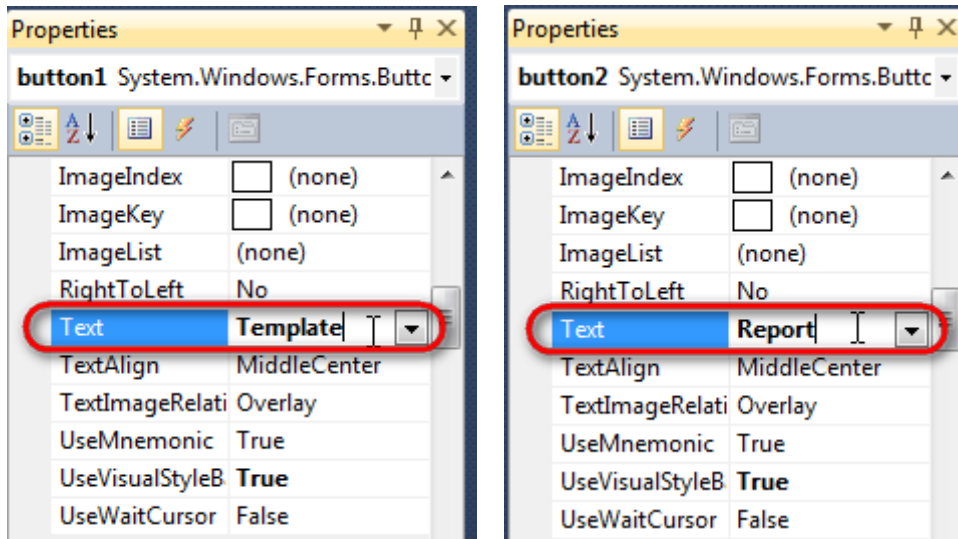
Get back to the application form by clicking "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



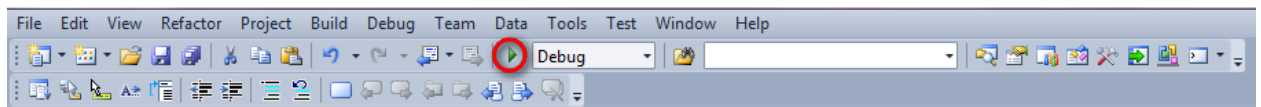
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

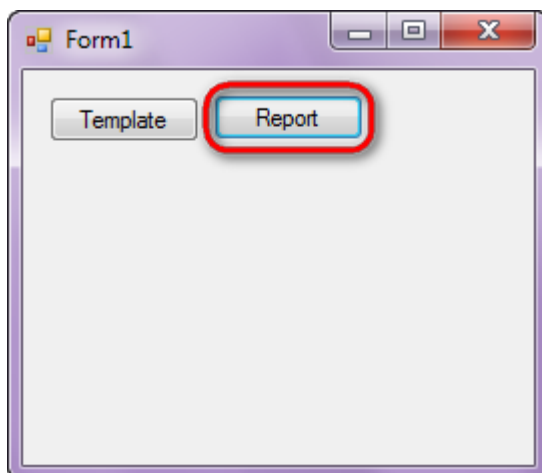
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 18

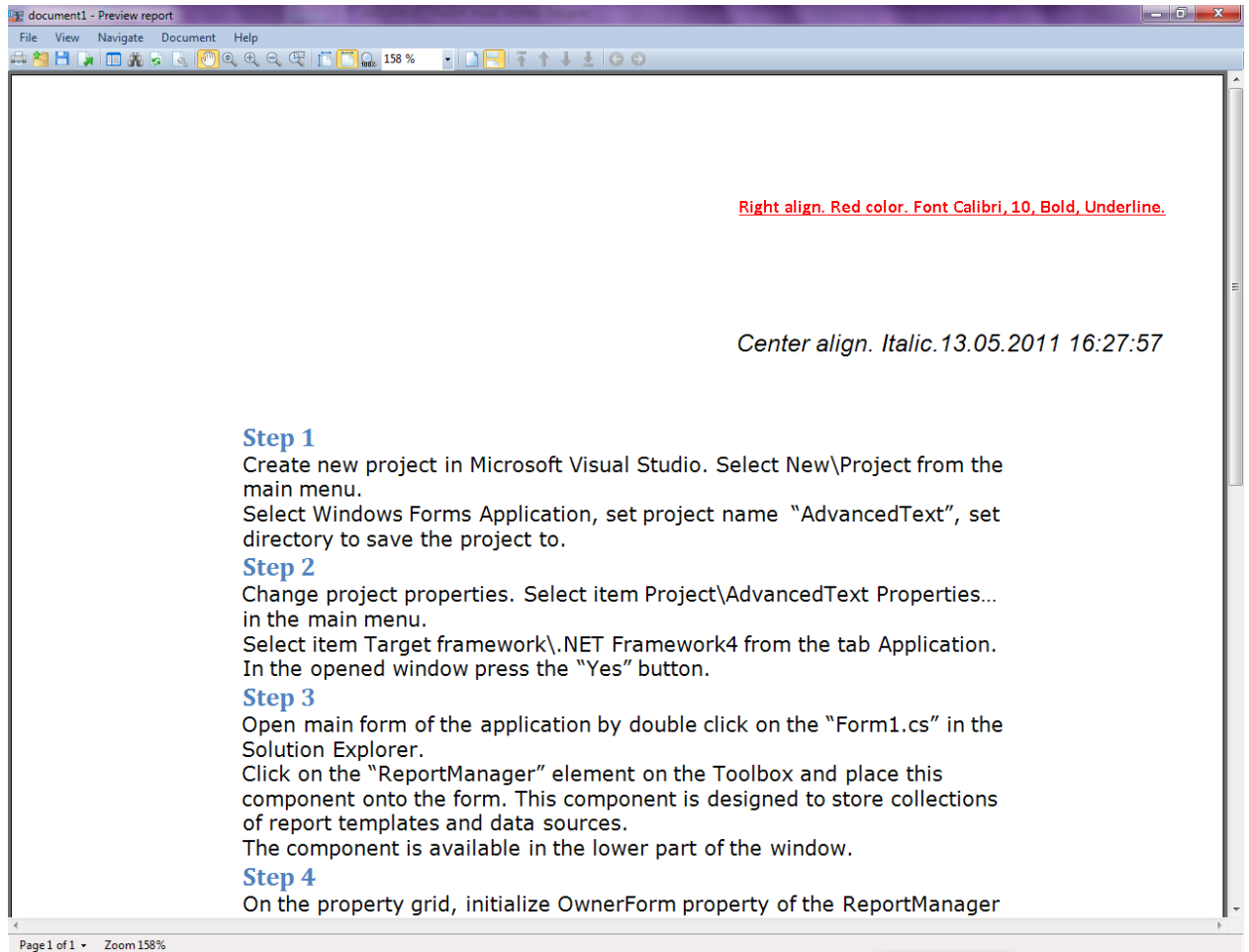
Click “Start Debugging” on the Visual Studio toolbar in order to run application.



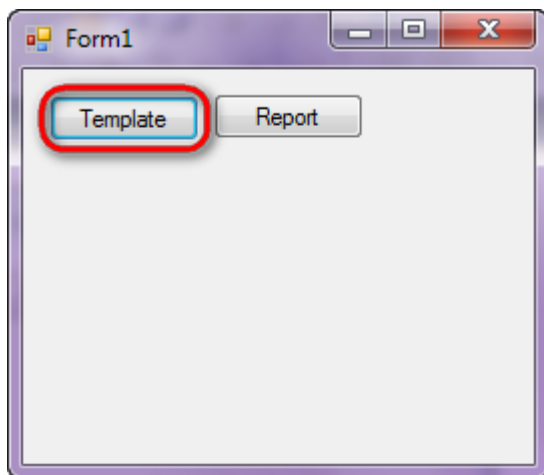
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar sample in the Samples Center is Report Controls\Basic\Advanced text.

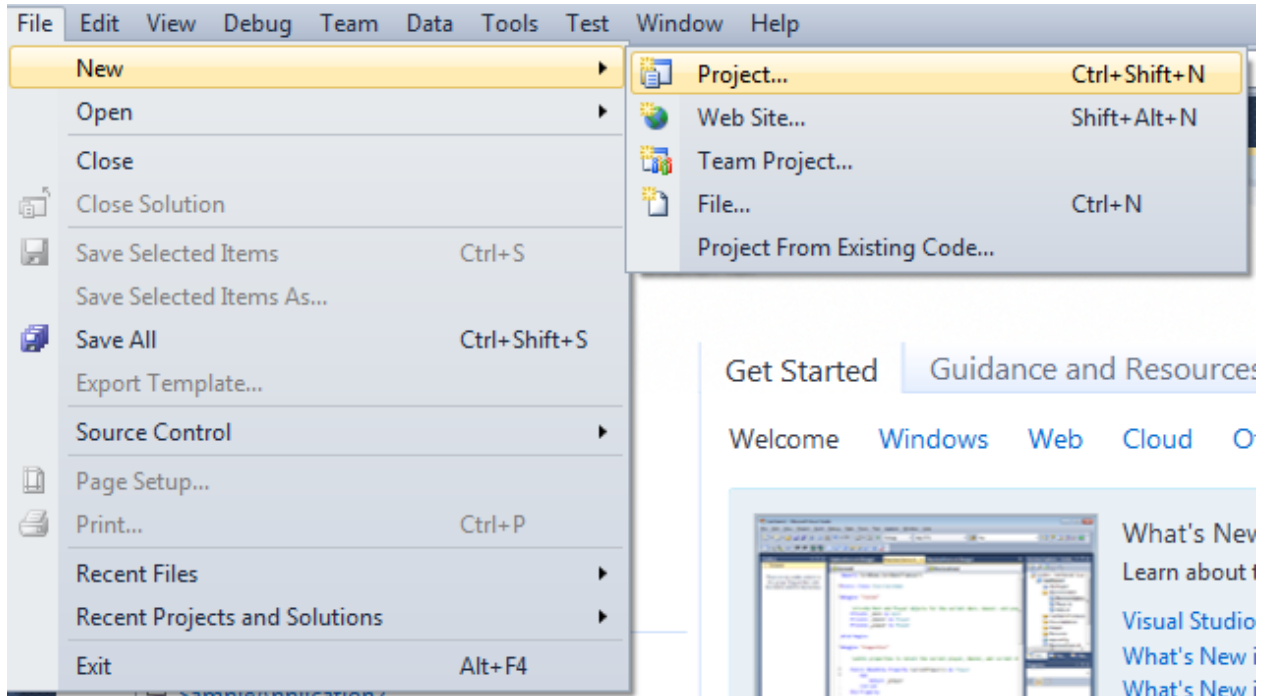


BarCode

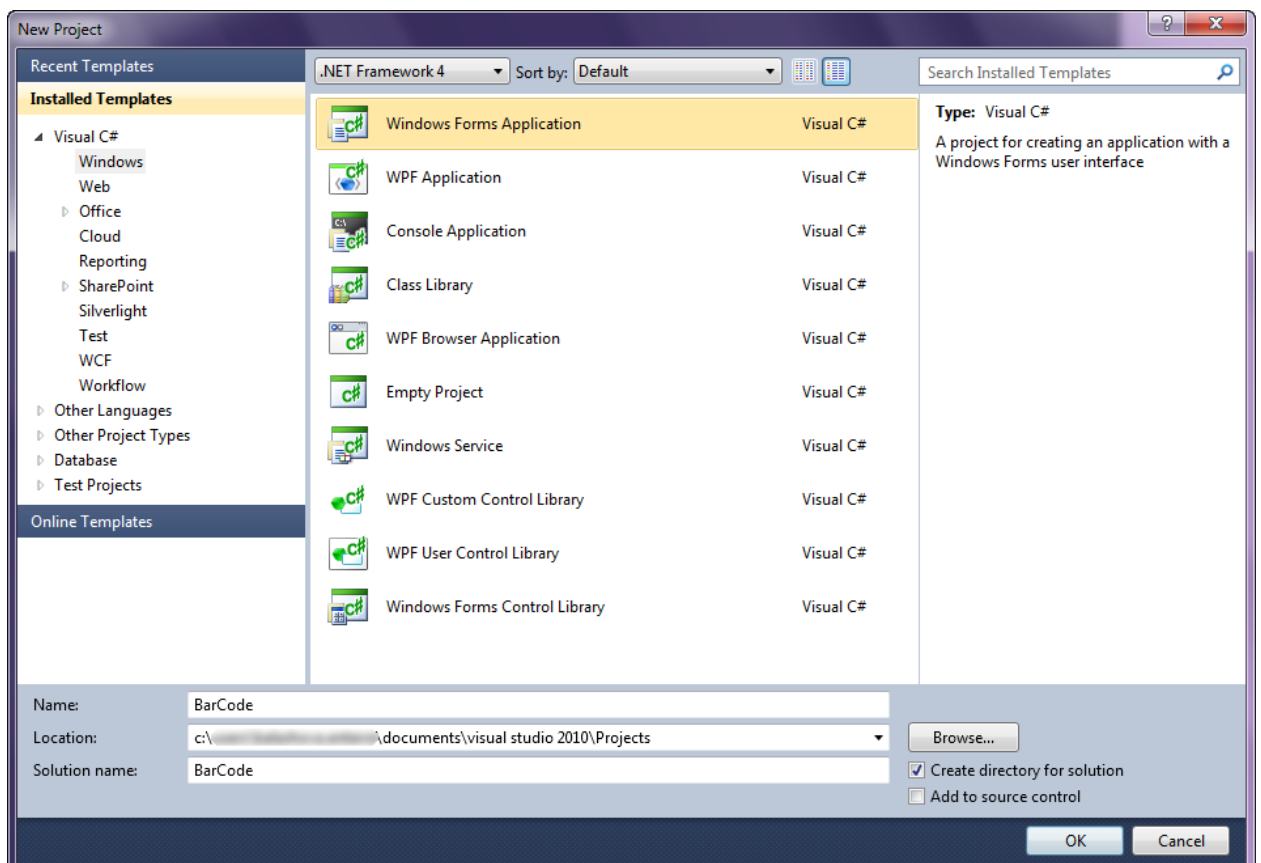
Template of a report designed to print labels with bar code.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

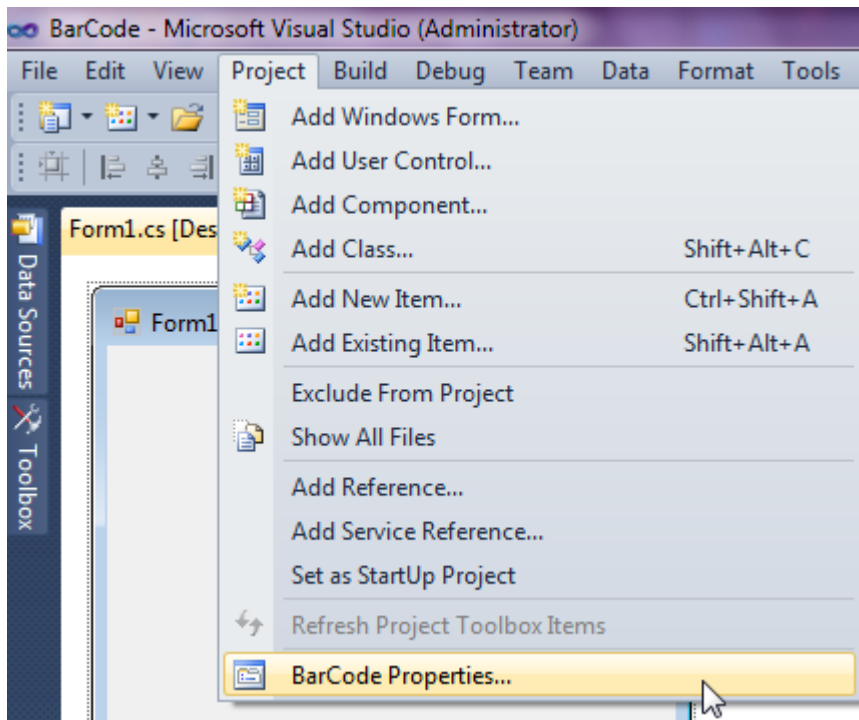


Select Windows Forms Application, set project name – “BarCode”, set directory to save the project to.

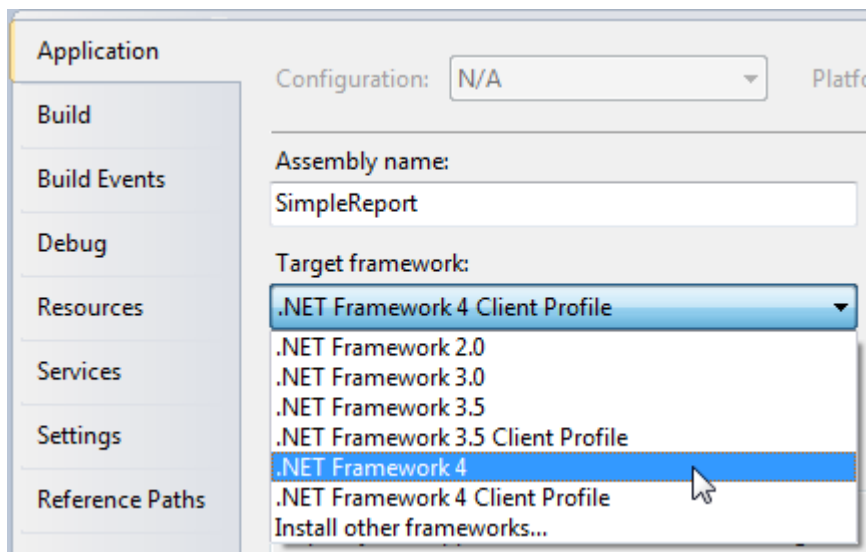


Step 2

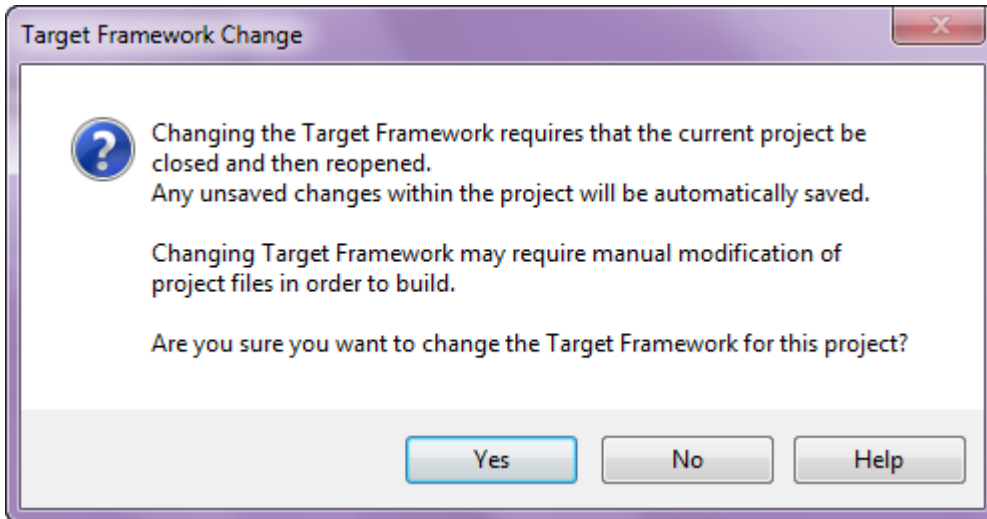
Change the project properties. Select the Project\BarCode Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

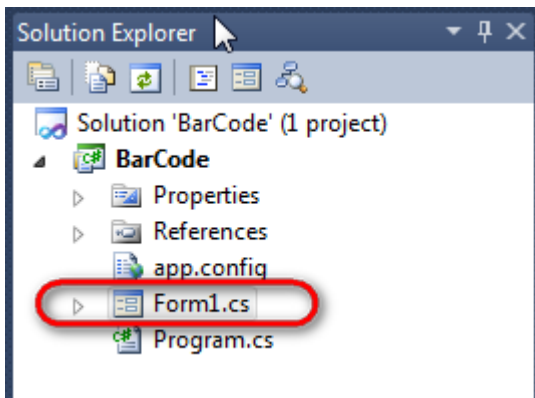


Press the "Yes" button in the opened window.

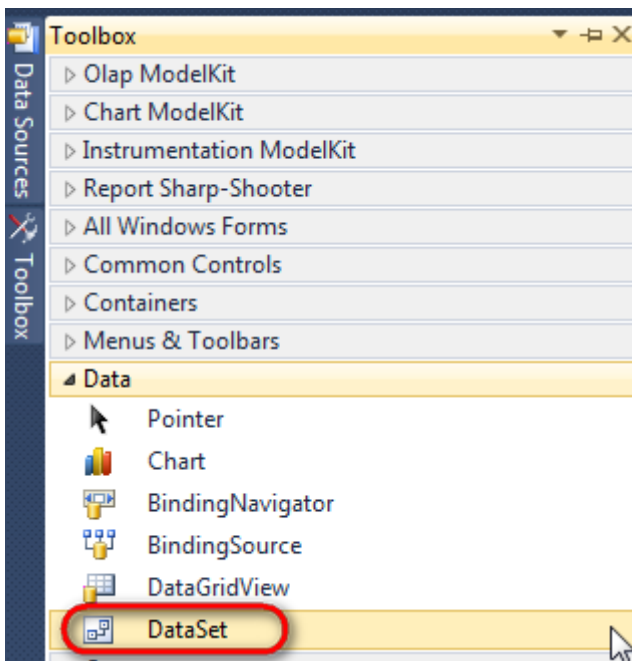


Step 3

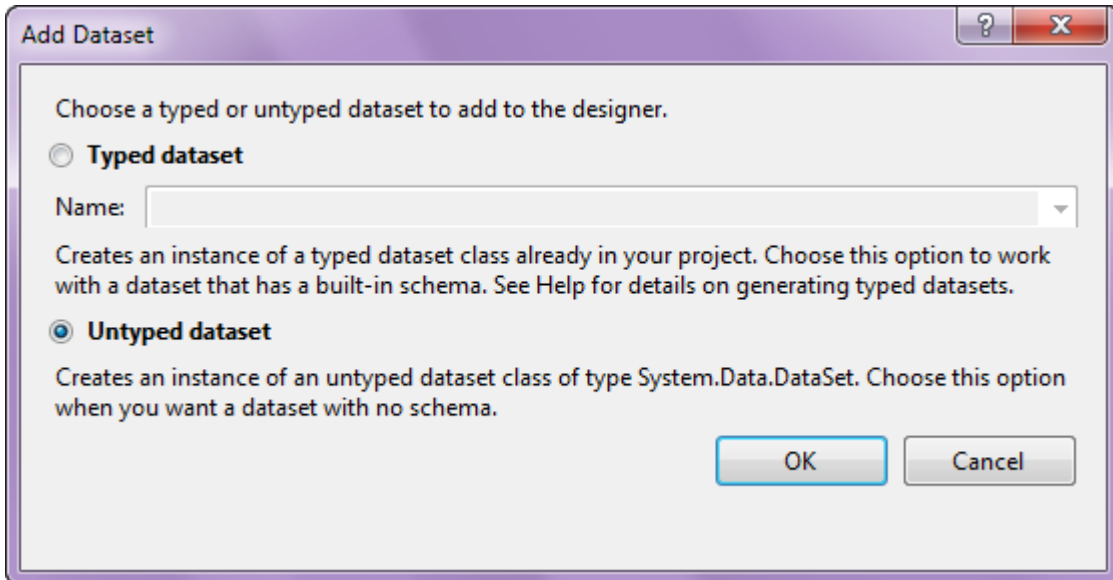
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



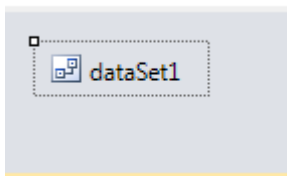
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK".

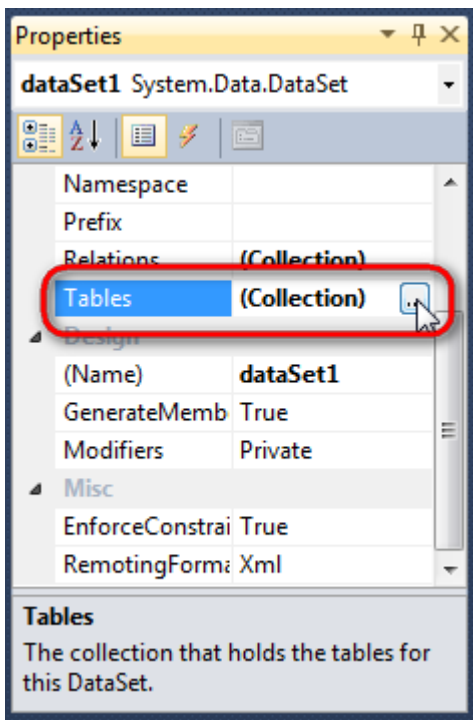


The component is available in the lower part of the window.

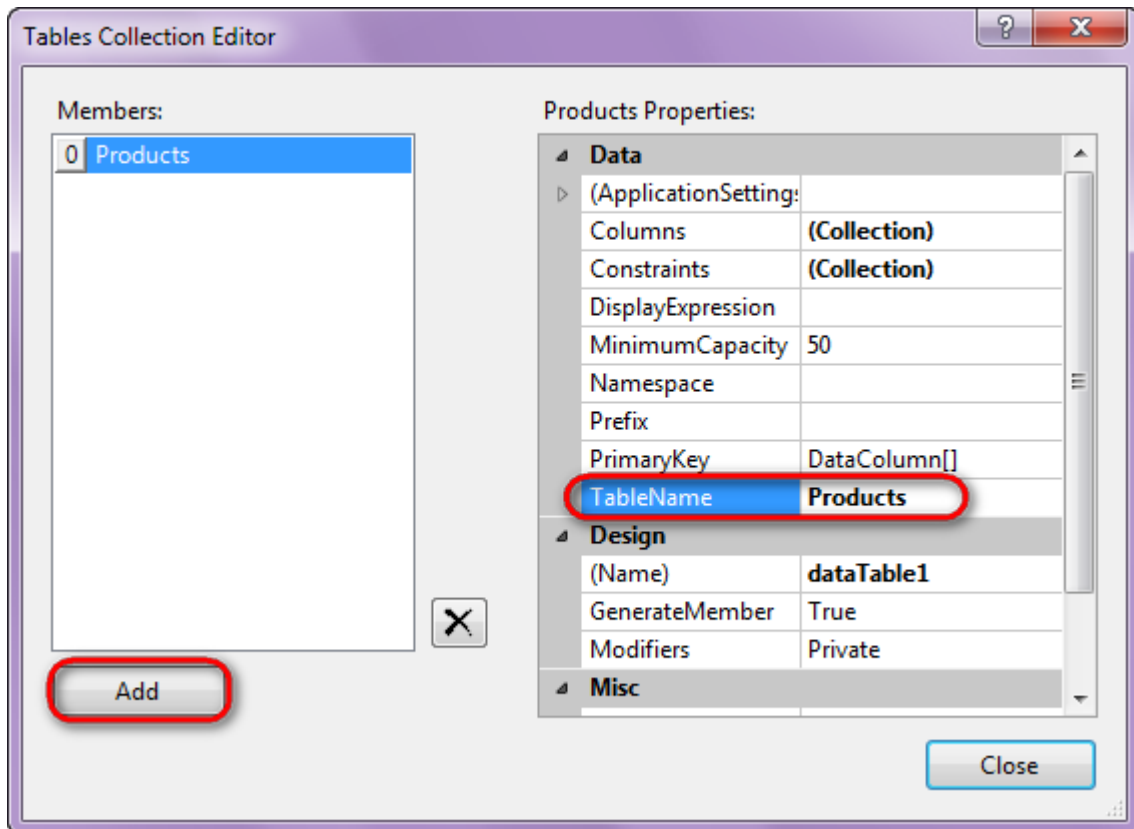


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

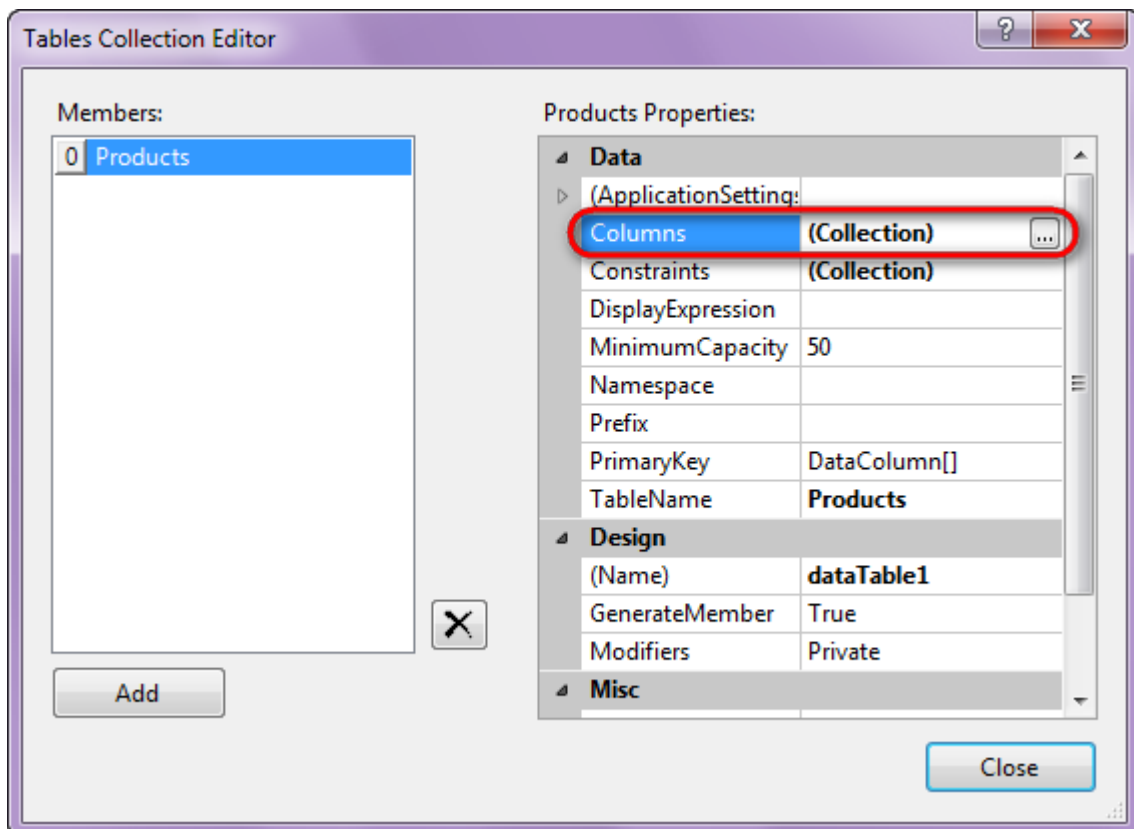


Click "Add" in order to add table. Set property TableName = Products.

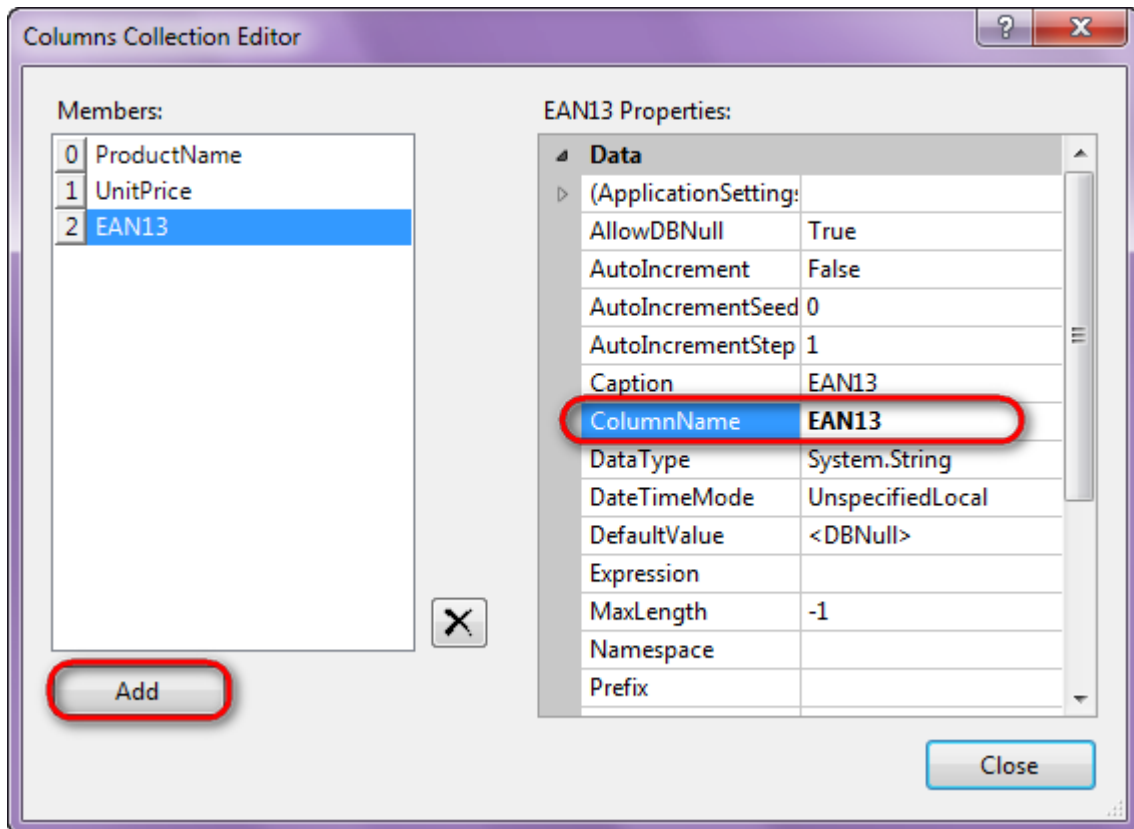


Step 5

Select Columns property, click button  in order to open property editor.

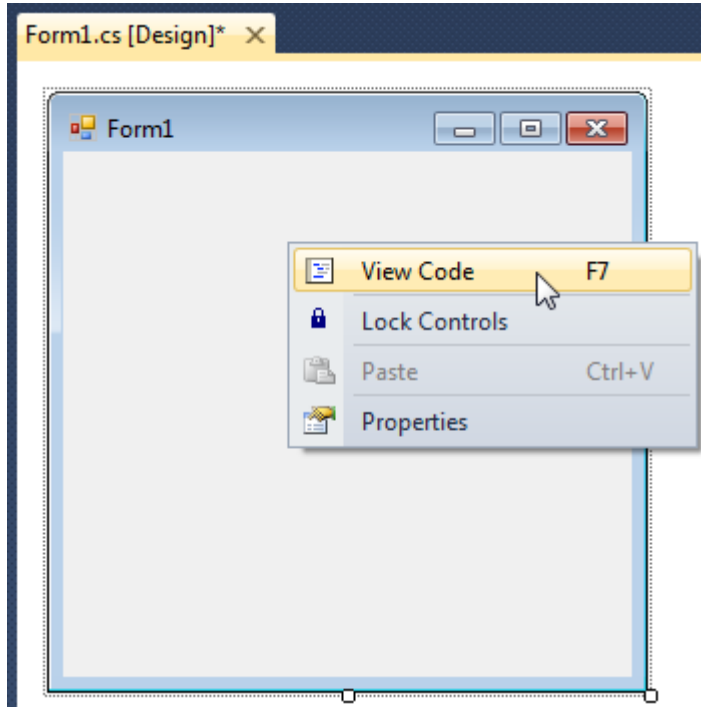


Click "Add" to add a new column. Add three columns. Set ColumnName property to "ProductName", "UnitPrice", "EAN13".



Step 6

Right click on the form and select "View Code" in the context menu to view code.



Add the following code to the class constructor in order to fill data source.

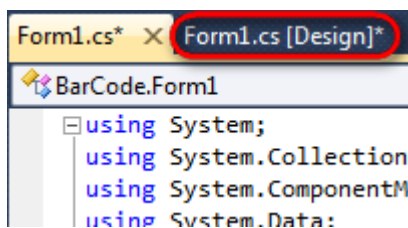
```
public Form1 ()  
{  
    InitializeComponent ();  
    DataRow row = dataTable1.NewRow ();  
    row["ProductName"] = "Chai";  
}
```



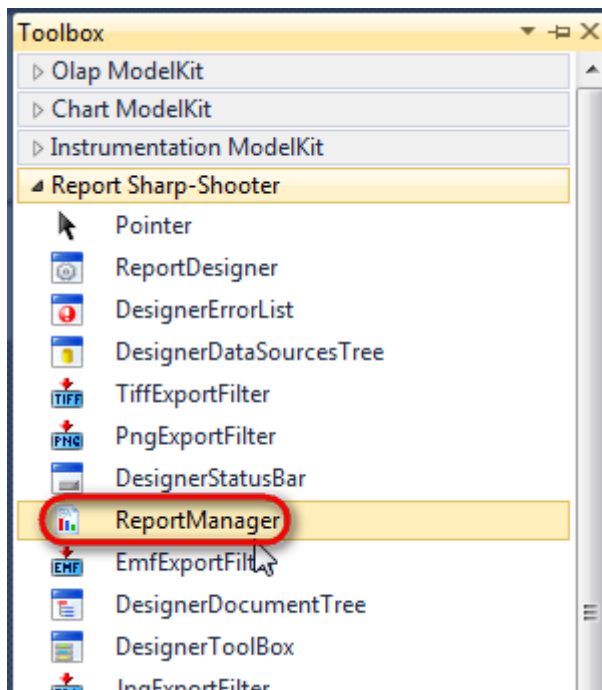
```
row["UnitPrice"] = "18";  
row["EAN13"] = "0706849000019";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["ProductName"] = "Chang";  
row["UnitPrice"] = "19";  
row["EAN13"] = "0706849000026";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["ProductName"] = "Aniseed Syrup";  
row["UnitPrice"] = "10";  
row["EAN13"] = "0706849000033";  
dataTable1.Rows.Add(row);  
}
```

Step 7

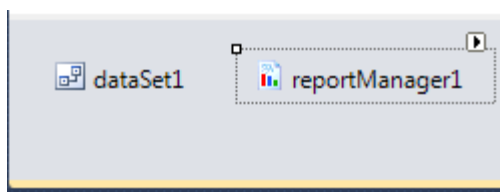
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

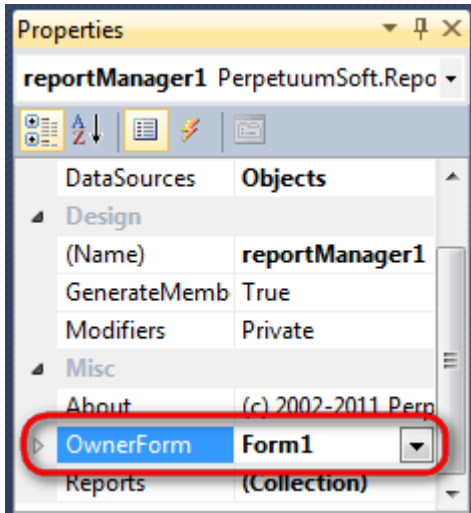


The component is available in the lower part of the window.



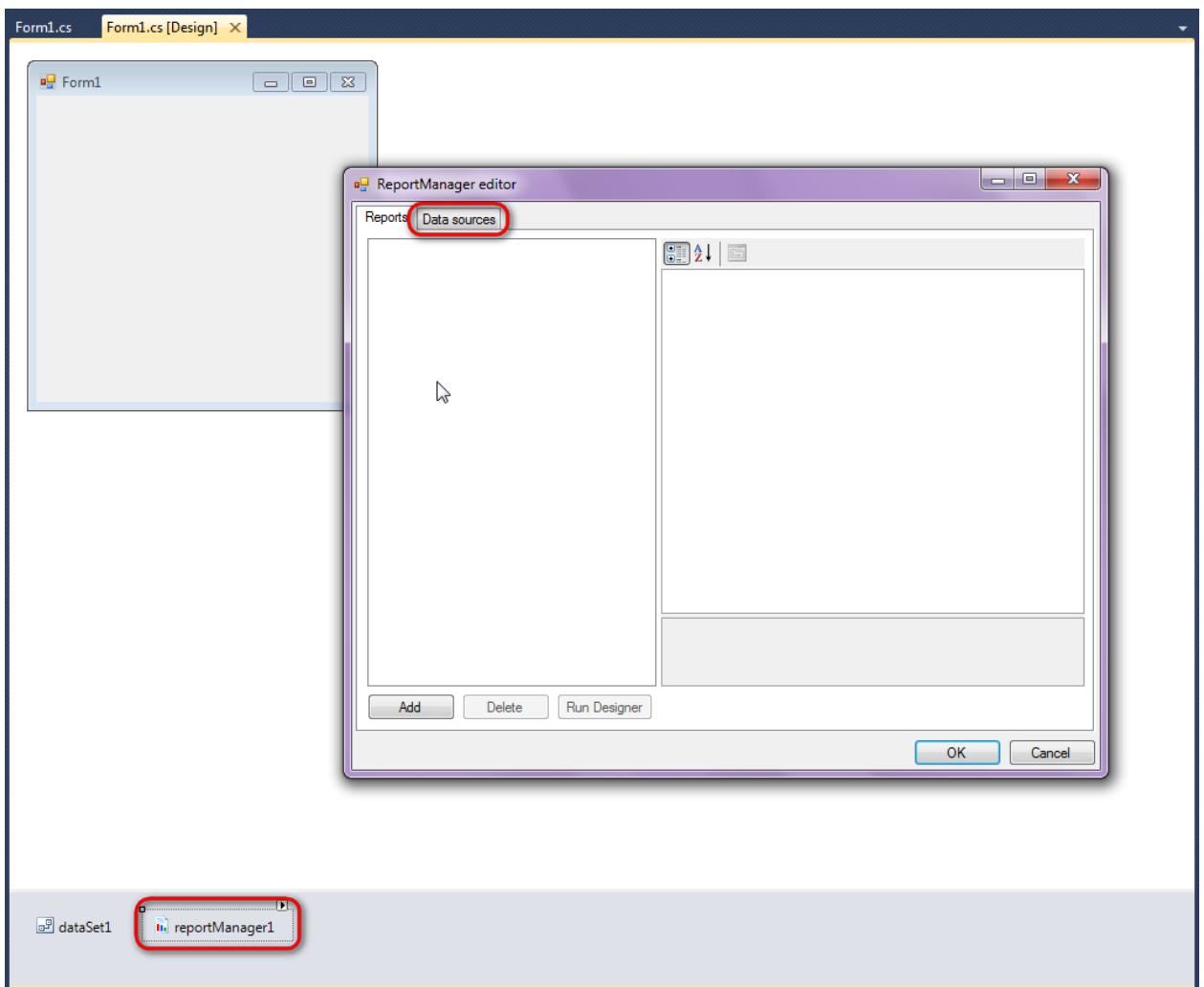
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

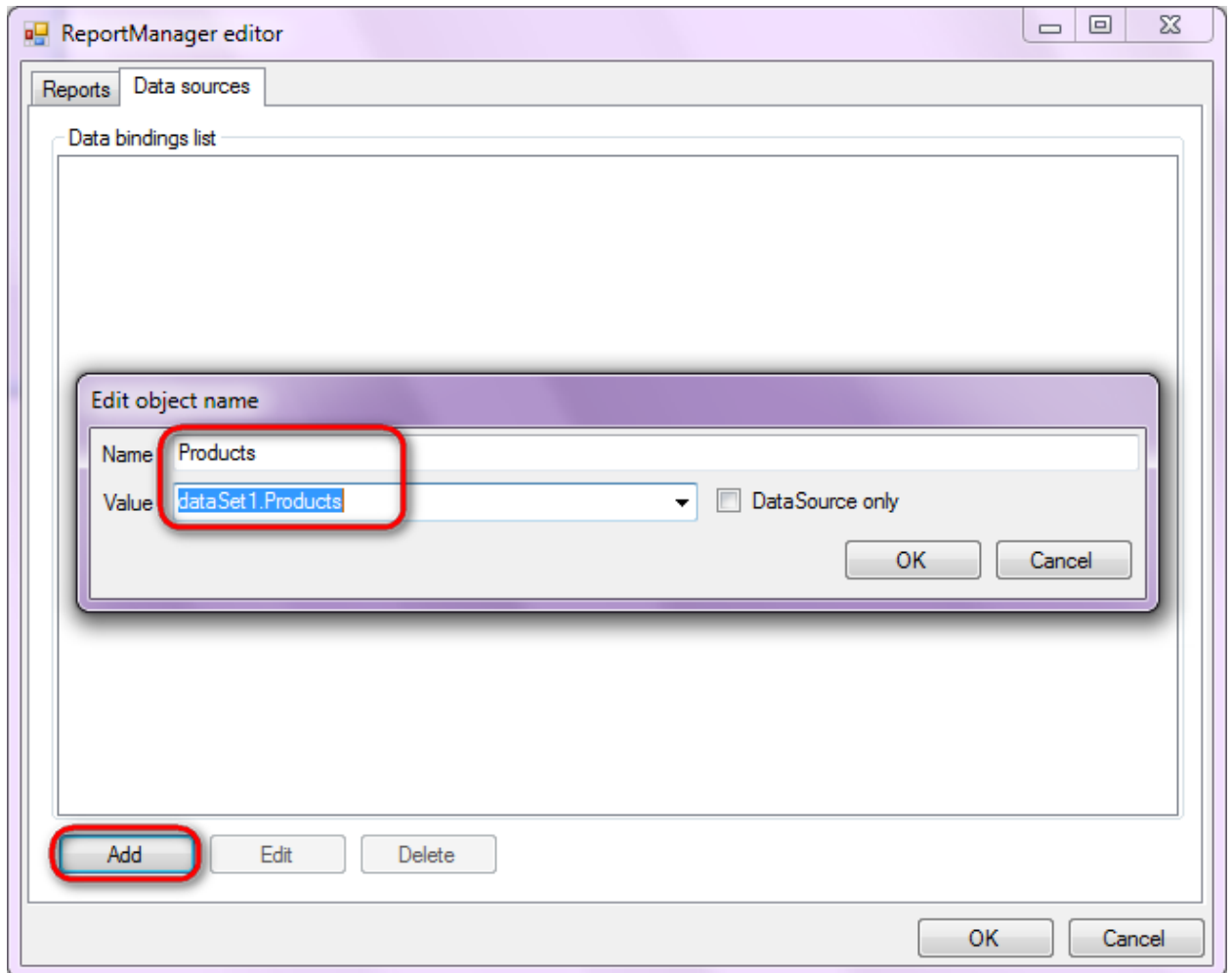


Step 9

Double click on ReportManager to open ReportManager editor.

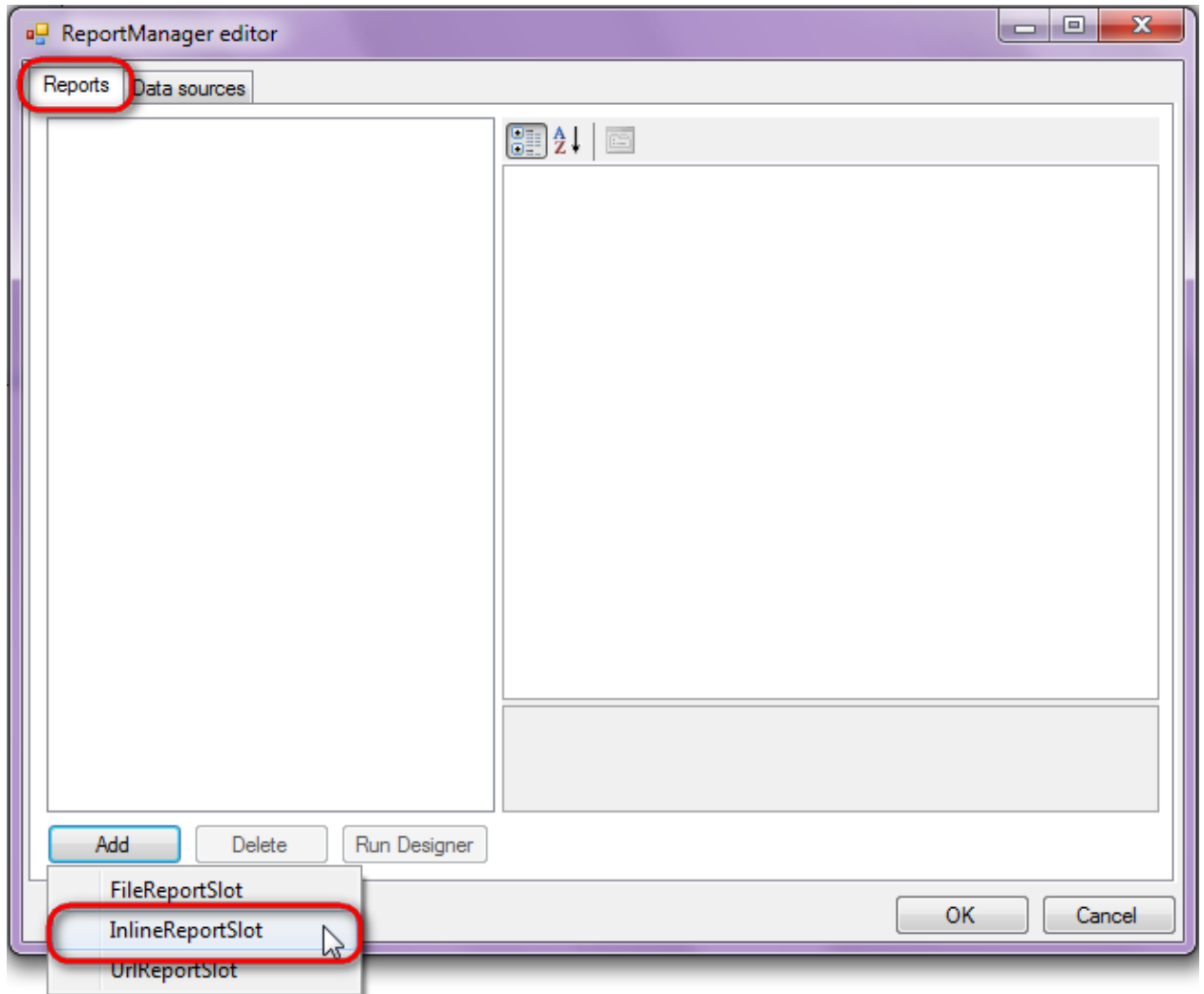


Go to "Data Sources" tab, click "Add", set data source name – "Products", select data source value – "dataSet1.Products".



Step 10

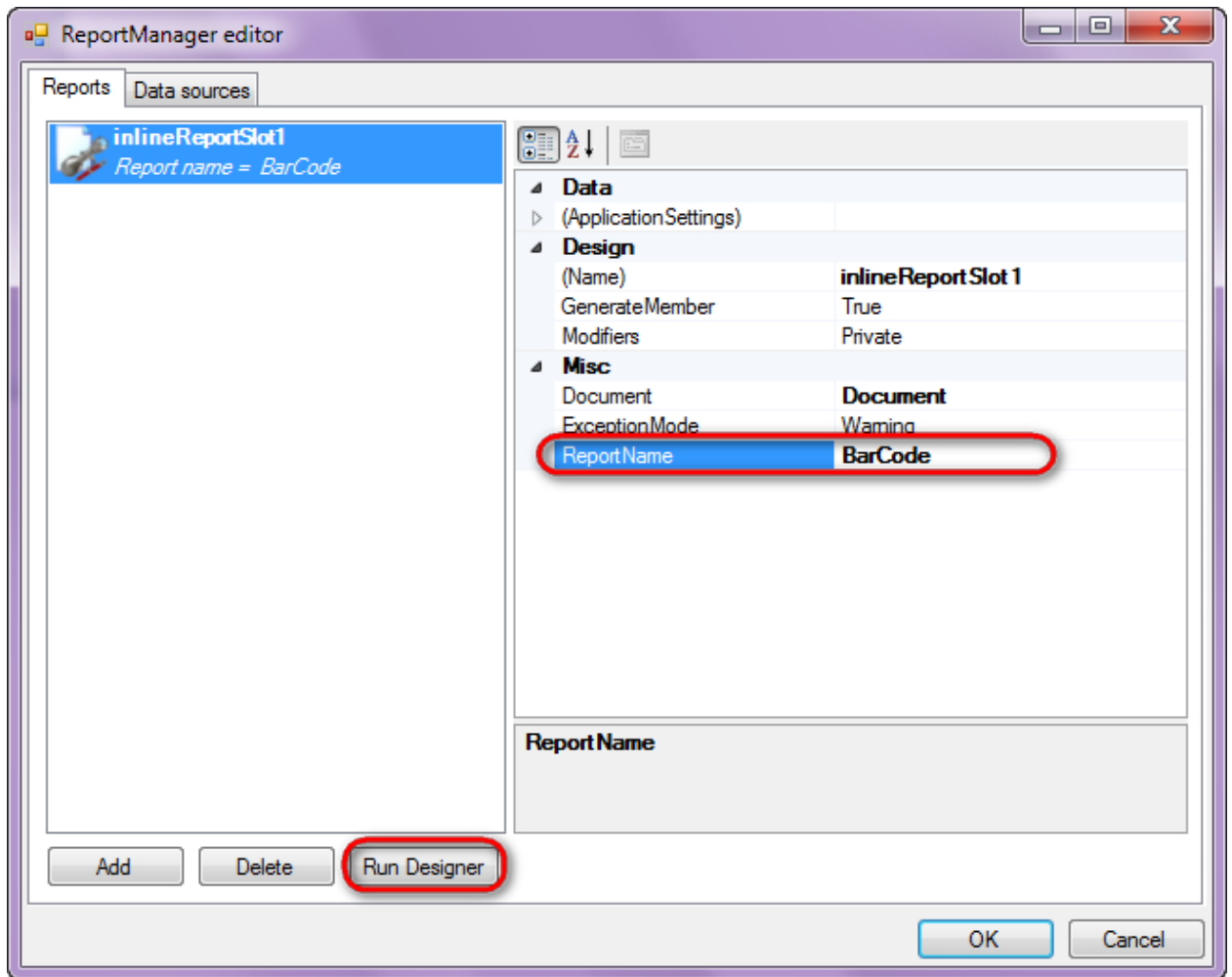
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

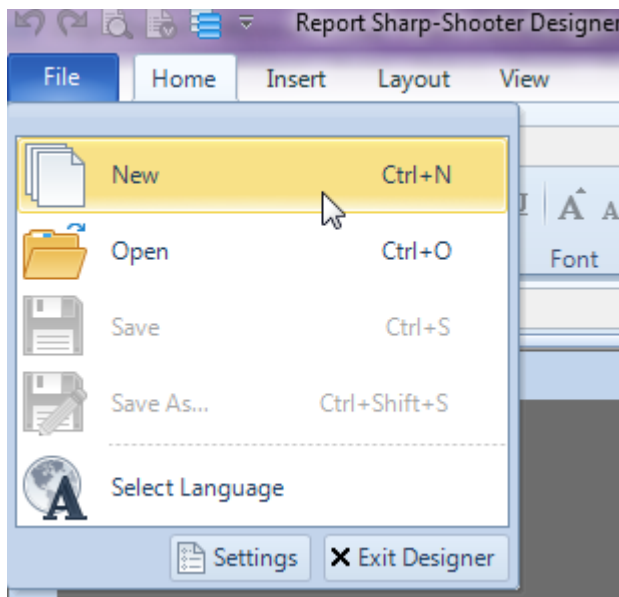
Set name of the report in the property ReportName – “BarCode”.

Click “Run Designer” in order to open template editor - Report Designer.

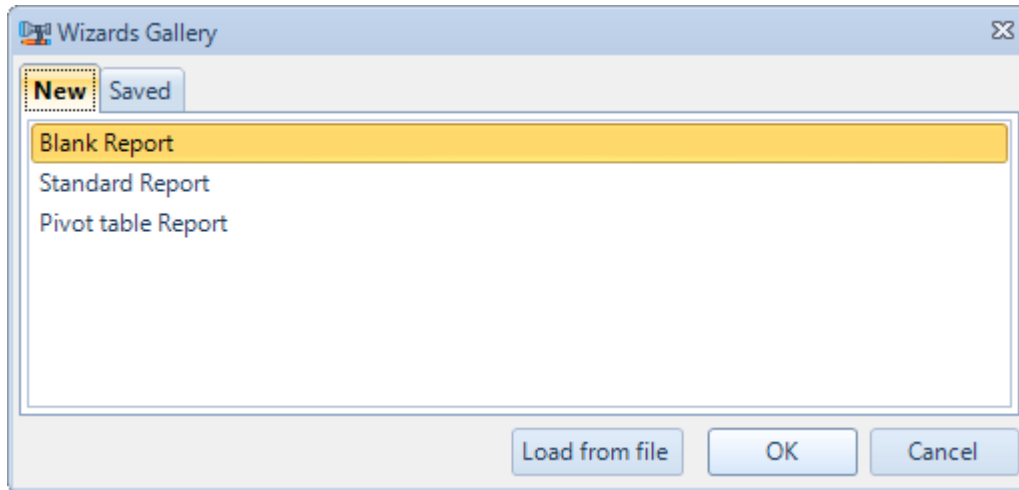


Step 12

Create new empty report – select File\New from the main menu.

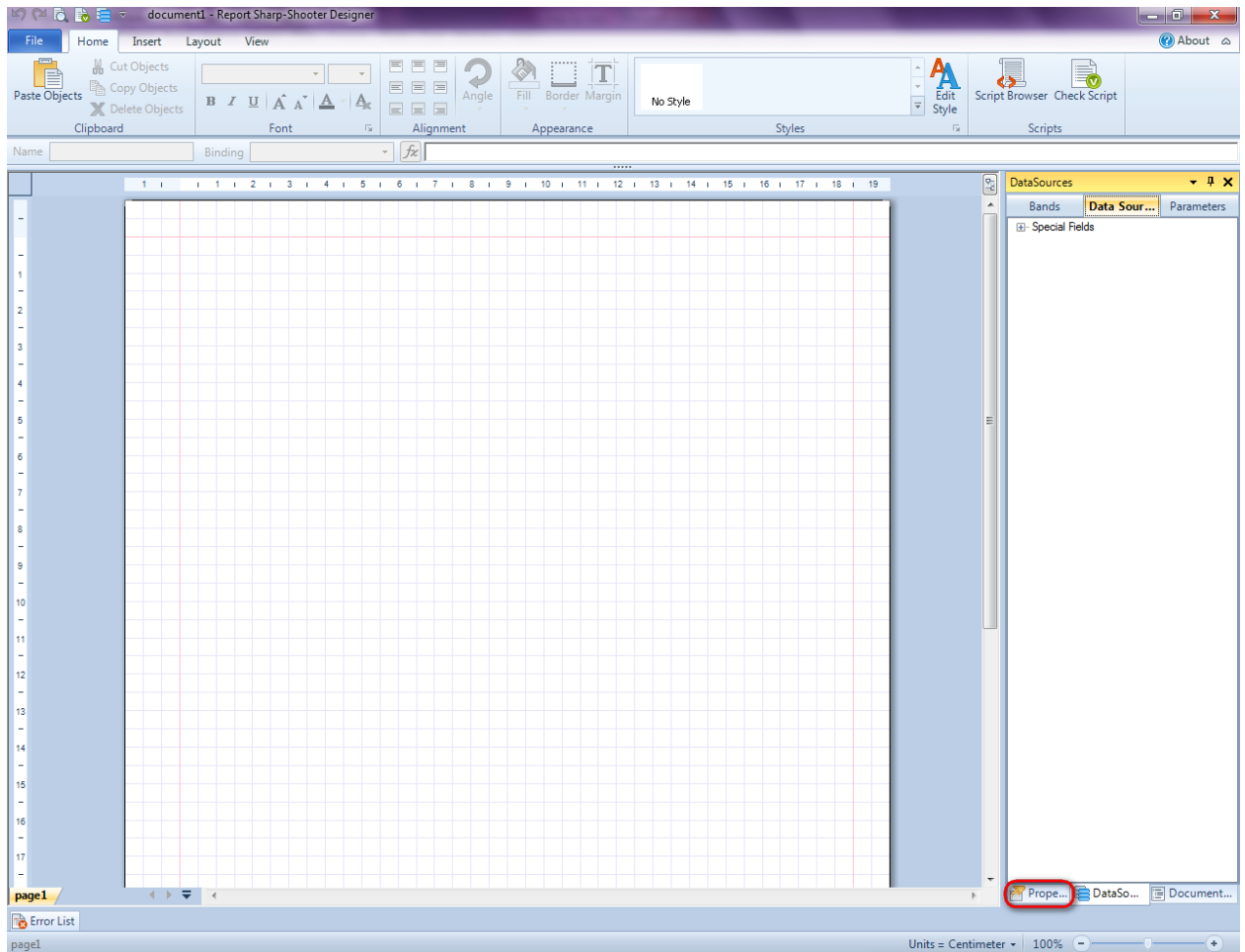


Select "Blank Report" in the Wizards Gallery and click "OK".



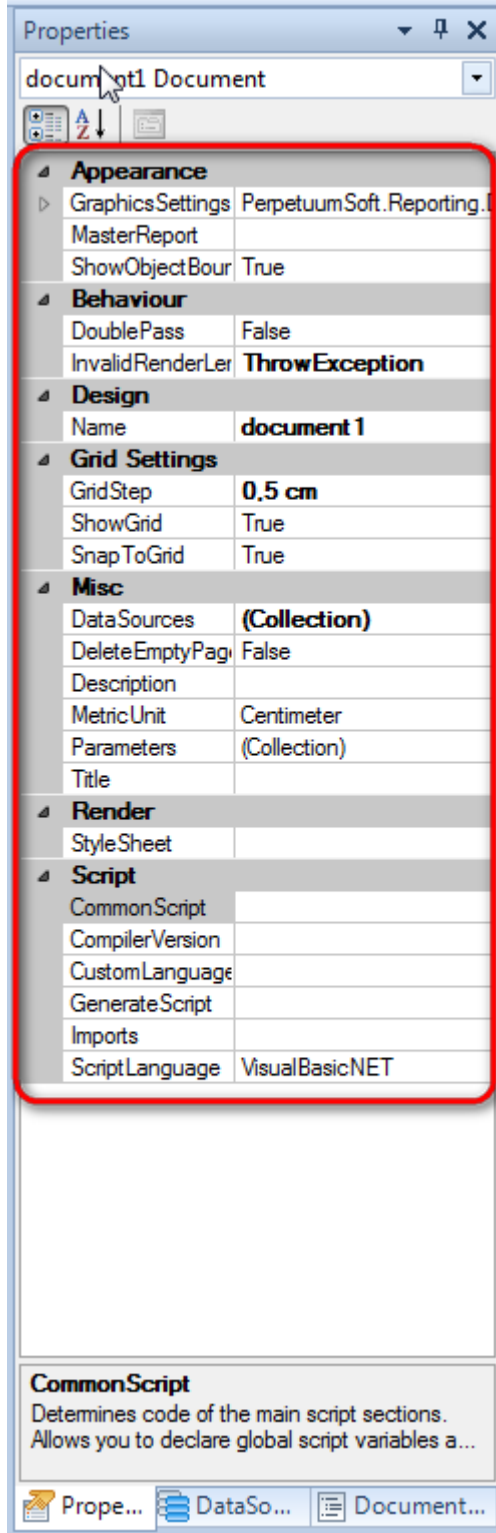
Step 13

Click the "Properties" tab of the tool window in the right part of the designer.

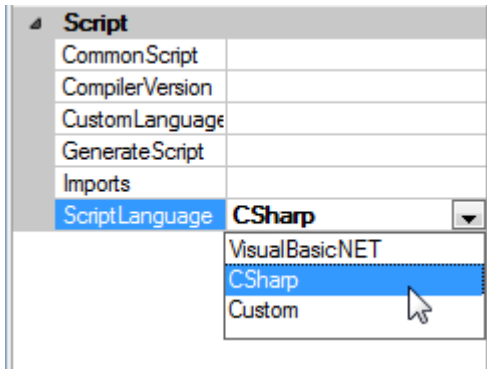





You will see properties of the edited template on the "Properties" tab

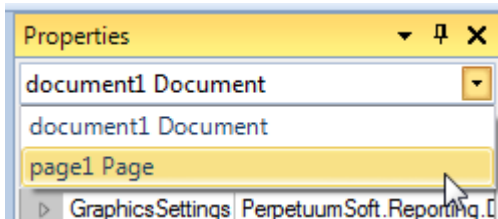


Set property ScriptLanguage = CSharp.

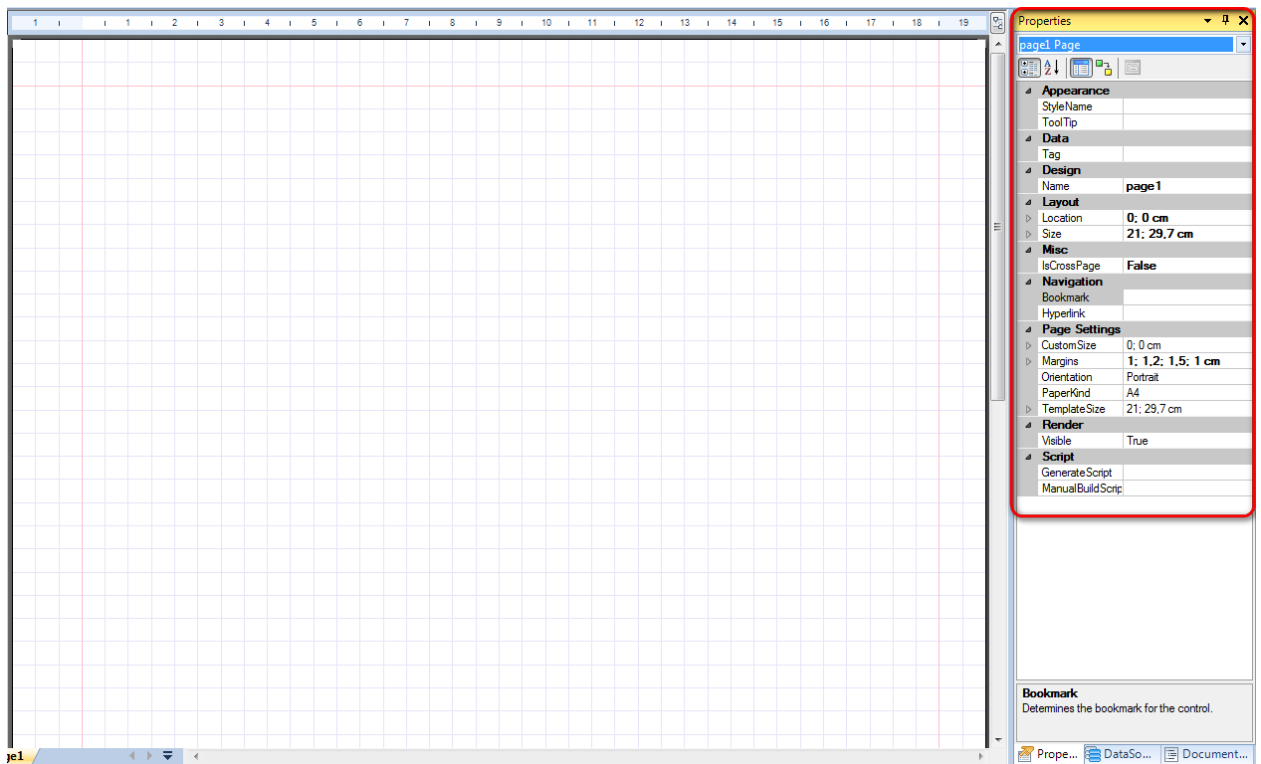



Step 14

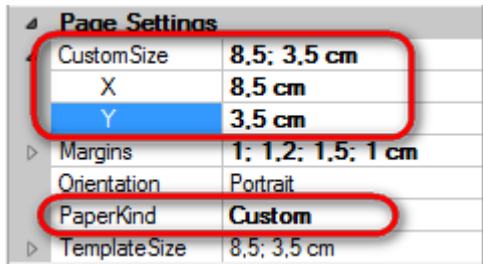
Press the  button in Properties. Select the "page1 Page" item in the combo box.



You will see properties of the edited page in the "Properties" tab of Tool Window in the right part of the designer.

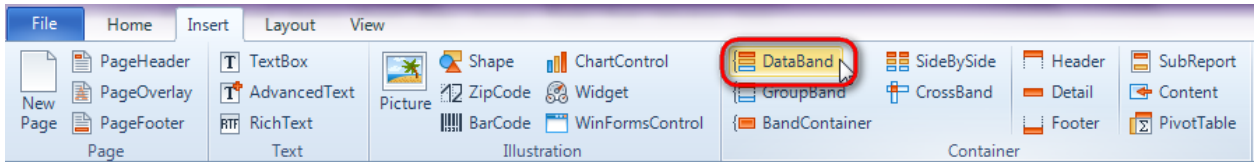


Set the PaperKind = Custom property. Click the  button to the left from the CustomSize property and set the following properties: X = 8,5 cm; Y = 3,5 cm.



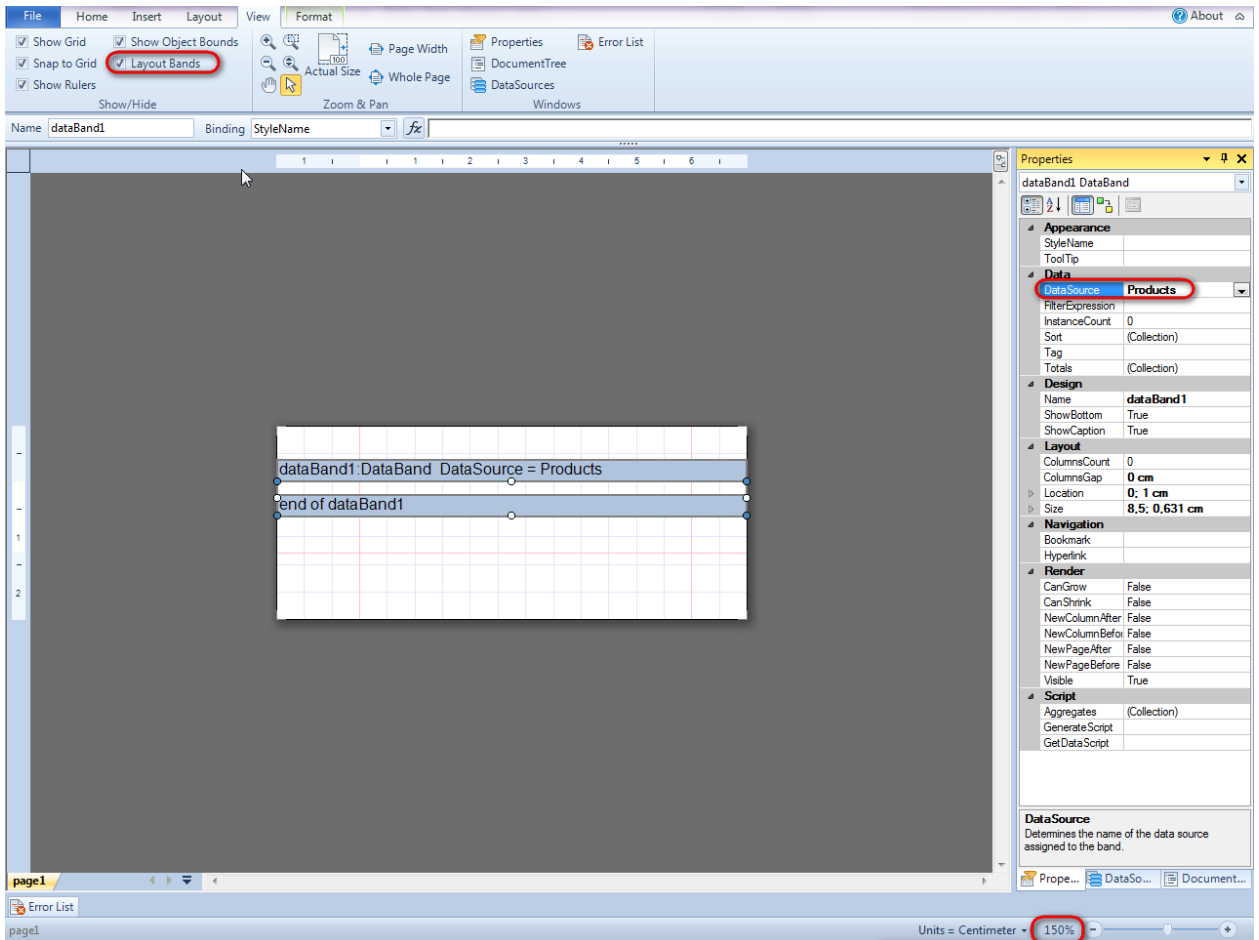
Step 15

Press the "DataBand" button in the Insert tab in the group Container.



Click on the template area to add DataBand to the template.

Resize the component in order it is fitted the page width.

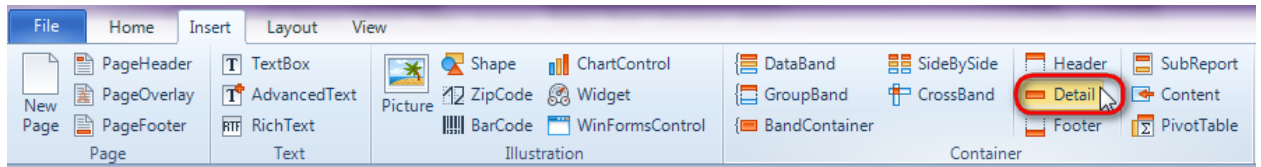


Pay attention to the zoom value – 150%. Check "Layout Bands" in the View tab in the Show\Hide group. The "Layout Bands" option, which automatically sets size of the bands, should be disabled.

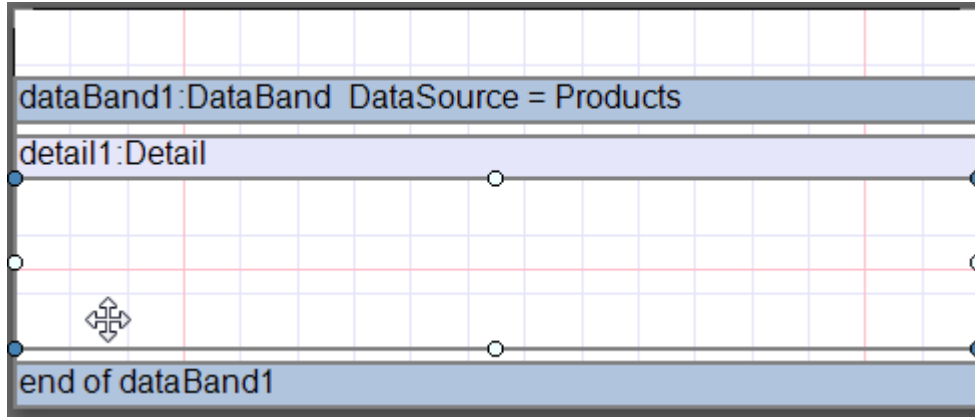
Set DataSource = Products.

Step 16

Press the "Detail" button in the Insert tab in the group Container.

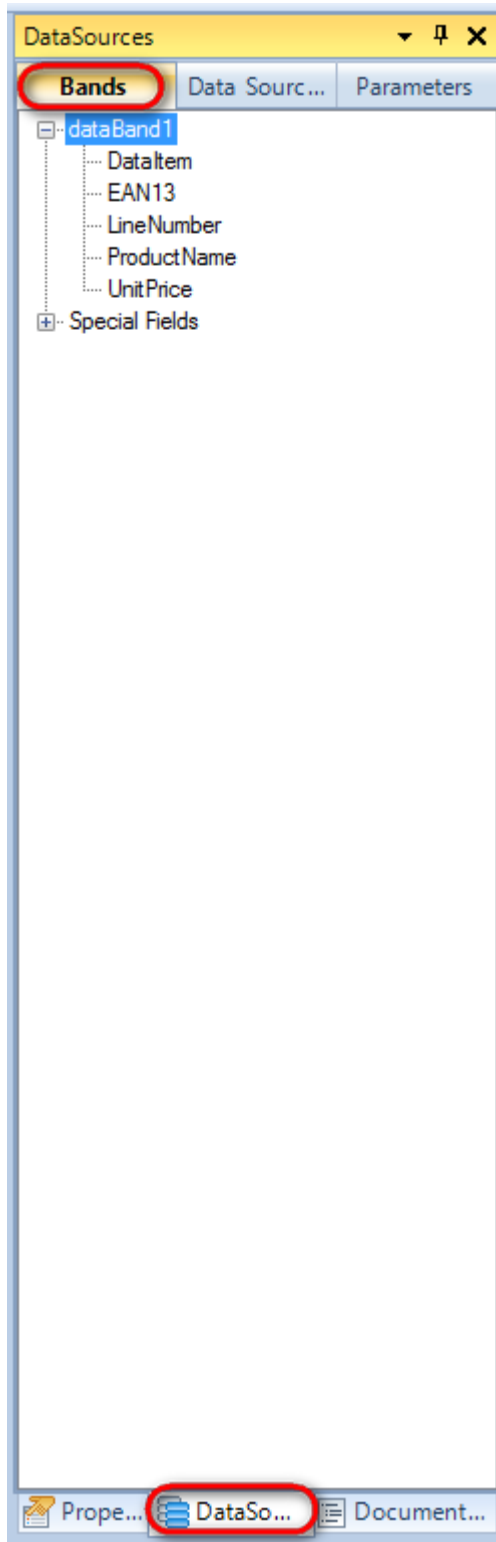


Click on the DataBand area to add Detail band inside DataBand. Change its size so that it covers the whole DataBand area.

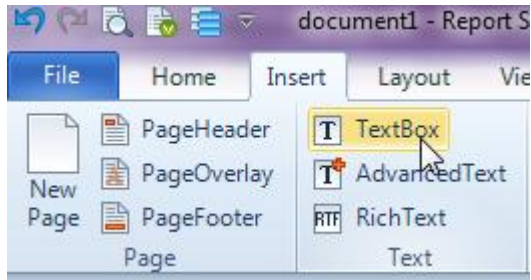


Step 17

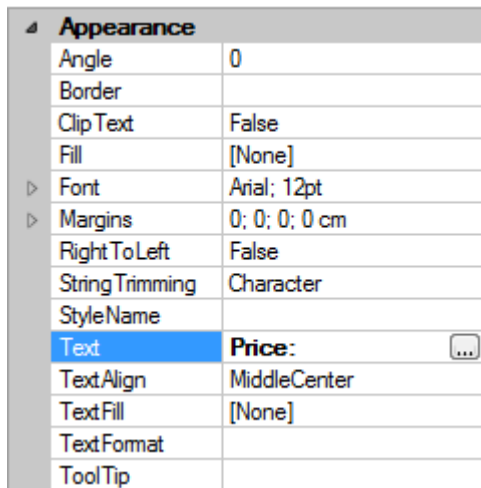
Go to the "DataSources" tab.



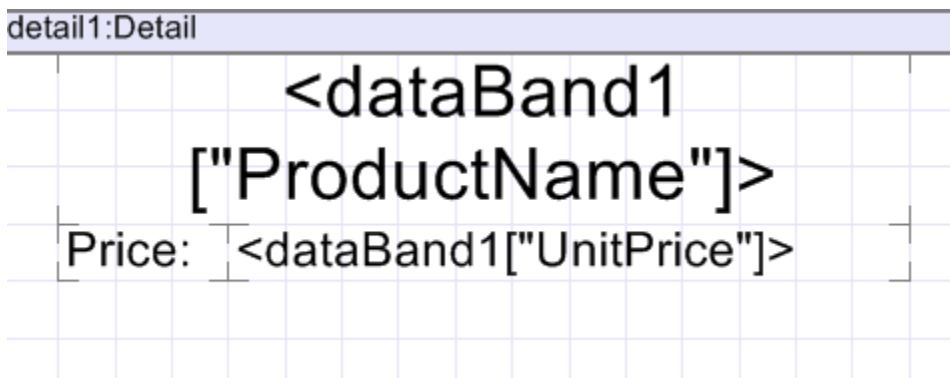
Drag and drop the "ProductName", "UnitPrice" fields from the dataBand1 tree to the detail1 band. Add one more TextBox element (press the "TextBox" button in the Insert tab in the Text group).



Set property Text = "Price:"

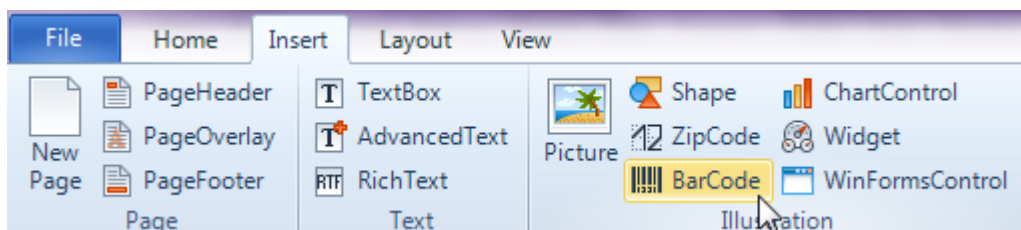


Locate the TextBox elements in the following way:

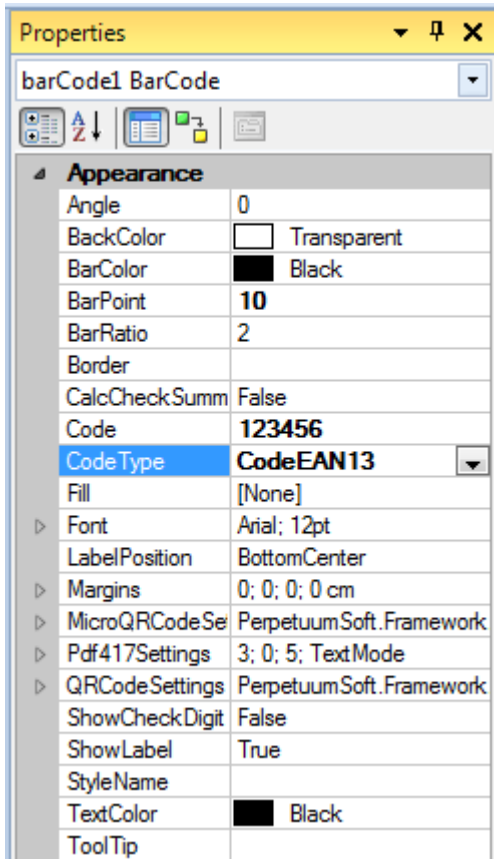


Step 18

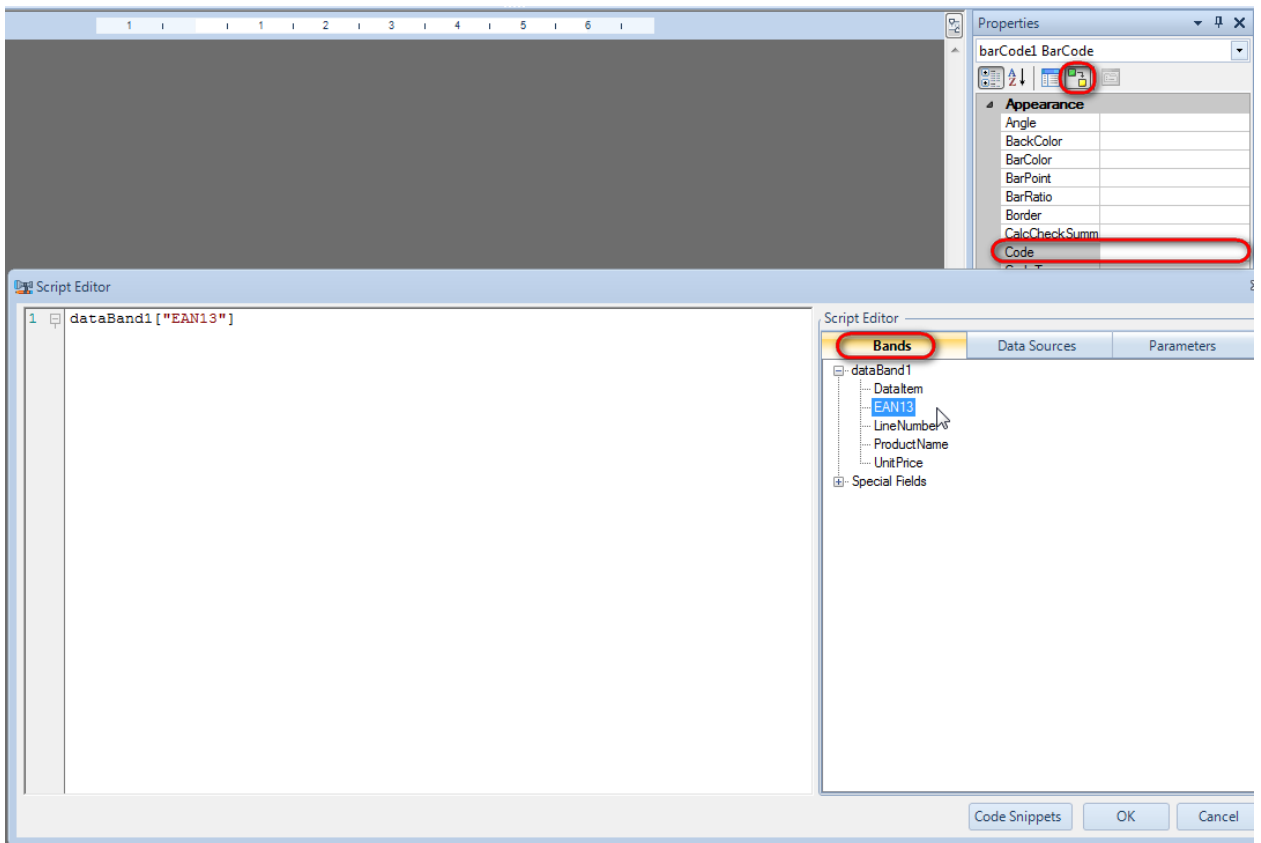
Press the "BarCode" button in the Insert tab in the group Illustration.



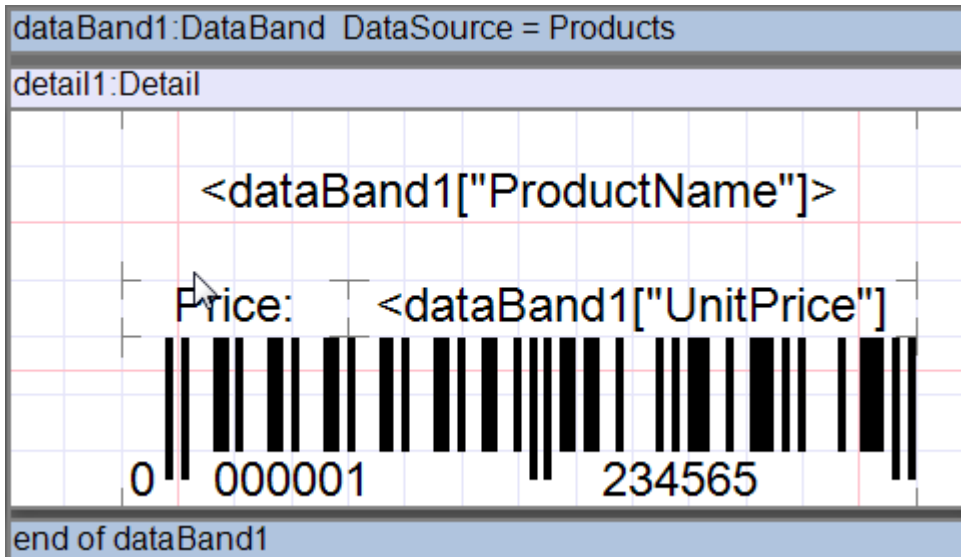
Place BarCode onto the detail1 band area. Set property CodeType = CodeEAN1.



To set Barcode code, open the Bindings properties in the "Properties" tab and set `barCode1.Code = dataBand1["EAN13"]`.



Report template should look as follows:



Step 19

Save template, close Report Designer.

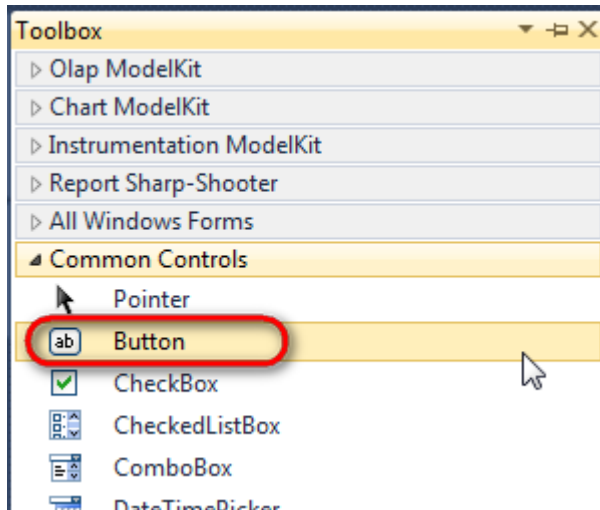
Step 20

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

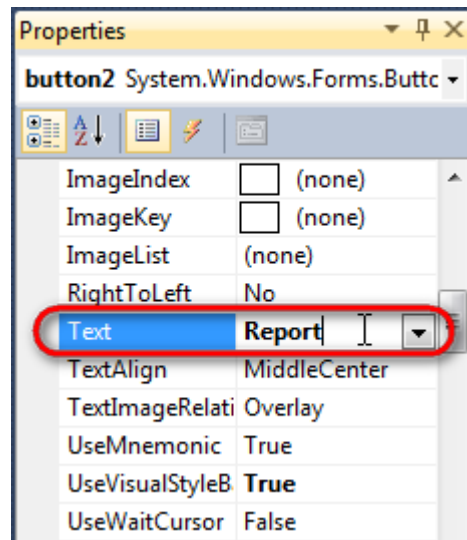
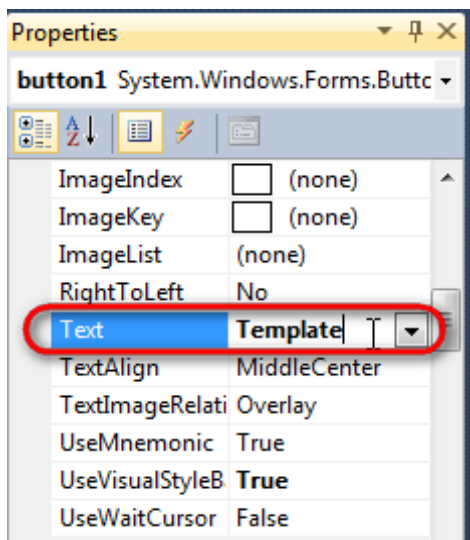
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["ProductName"] = "Chai";
    row["UnitPrice"] = "18";
    row["EAN13"] = "0706849000019";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["ProductName"] = "Chang";
    row["UnitPrice"] = "19";
    row["EAN13"] = "0706849000026";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["ProductName"] = "Aniseed Syrup";
    row["UnitPrice"] = "10";
    row["EAN13"] = "0706849000033";
    dataTable1.Rows.Add (row);
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted (object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
}
```

Step 21

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



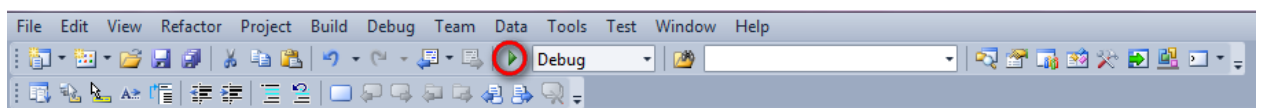
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

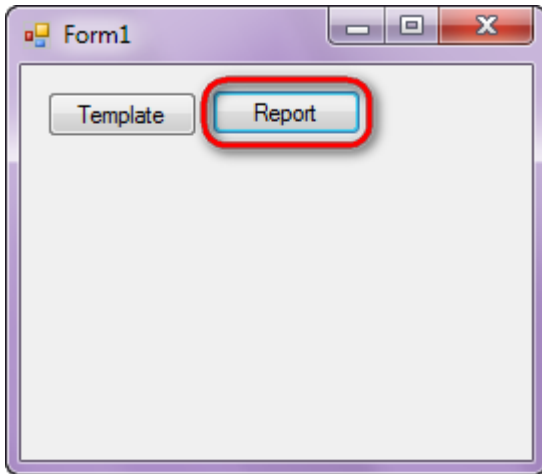
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

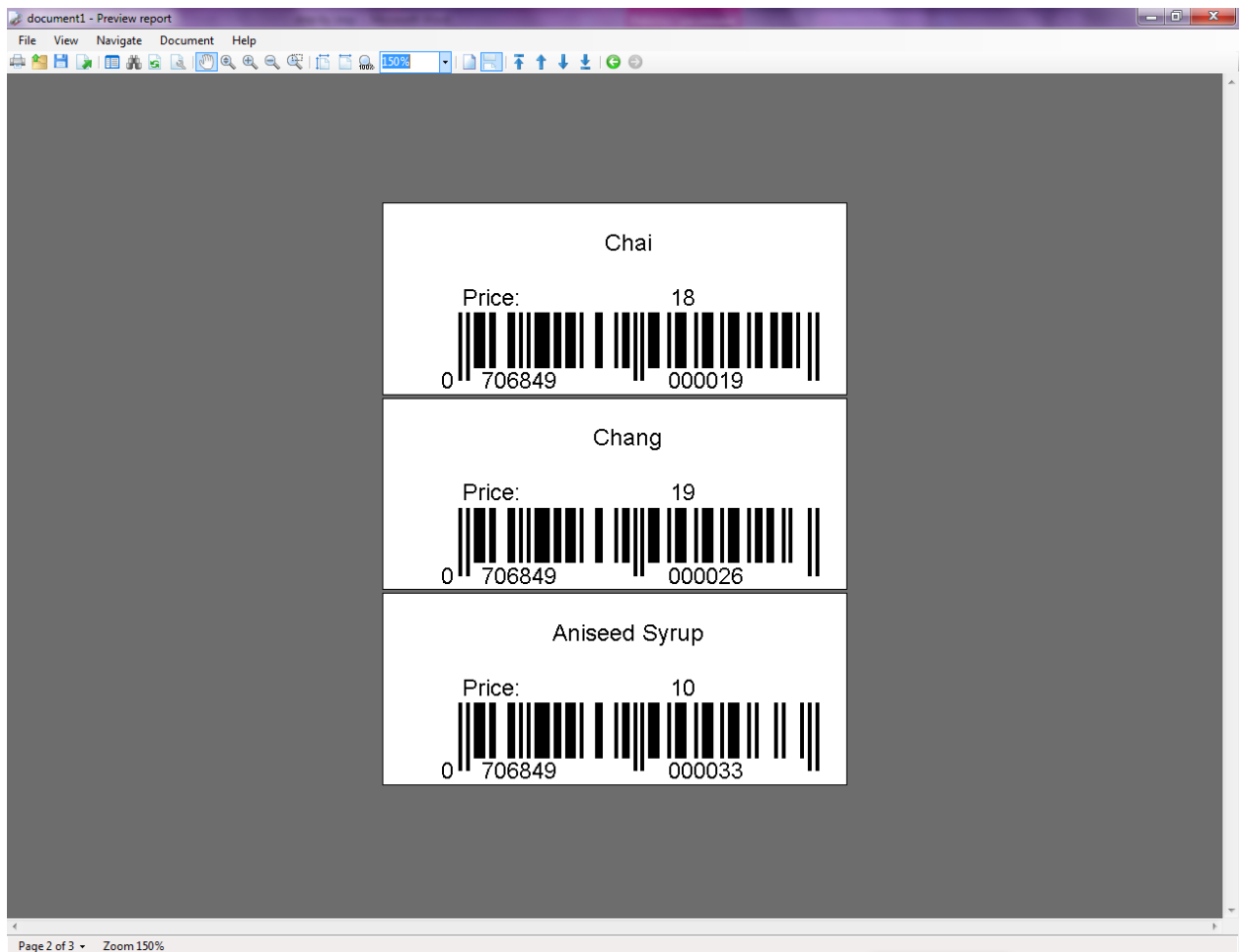
Click “Start Debugging” on the Visual Studio toolbar in order to run application.



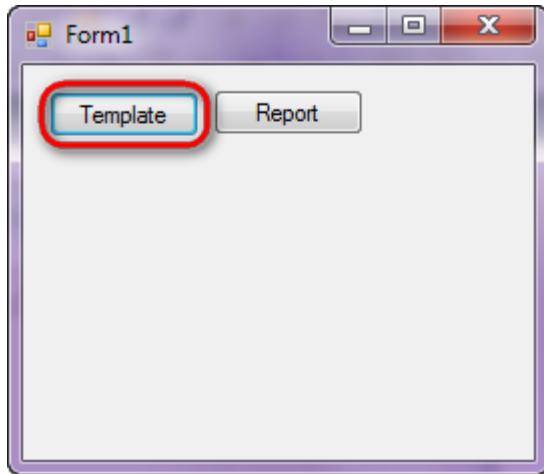
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar application sample is located in the following folder "\\Perpetuum Software\Net ModelKit Suite\ Samples\Report Sharp-Shooter\CSharp\LabelExample".

Similar sample in the Samples Center is Report Controls\Basic\Bar codes.

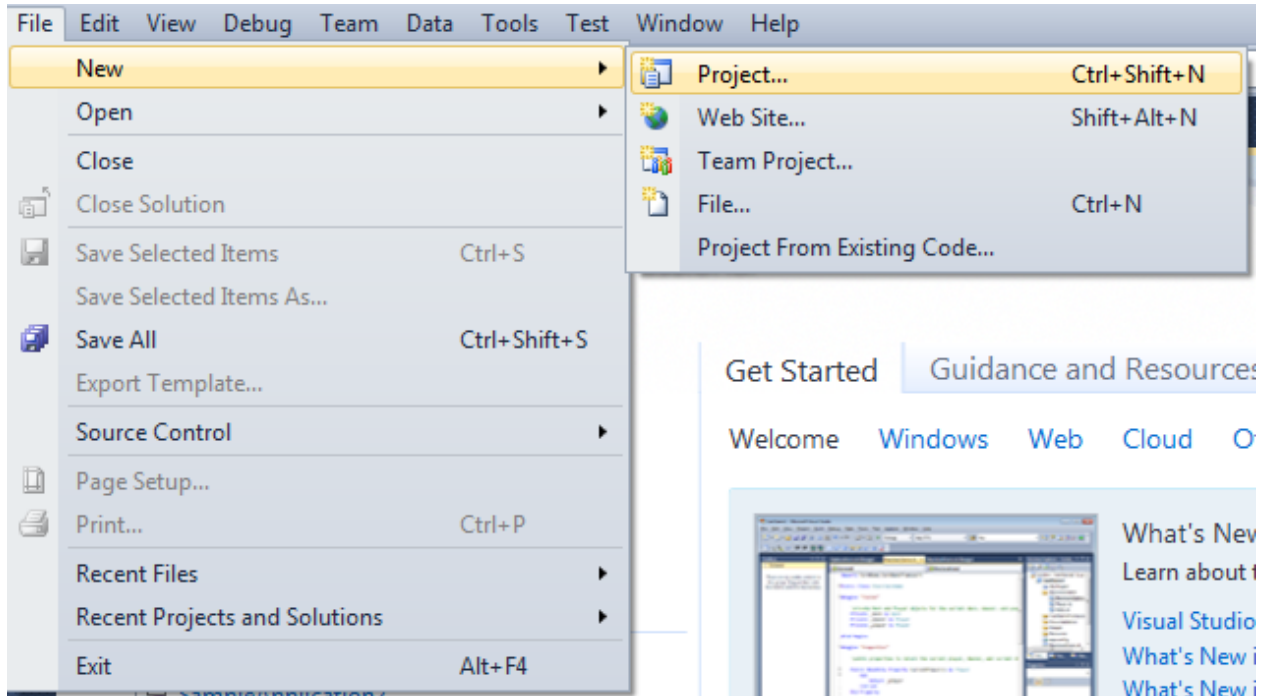


Picture

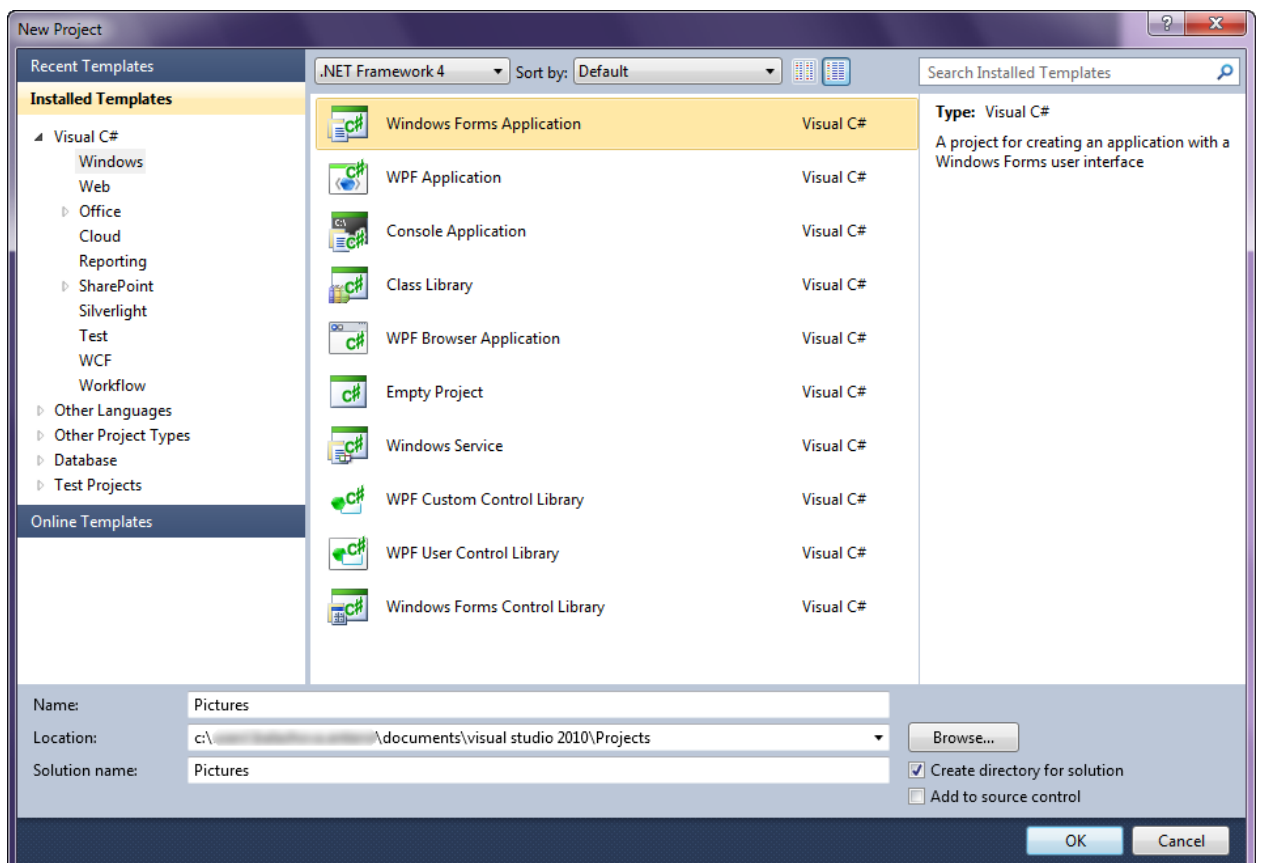
Template of a report containing employees' photos.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

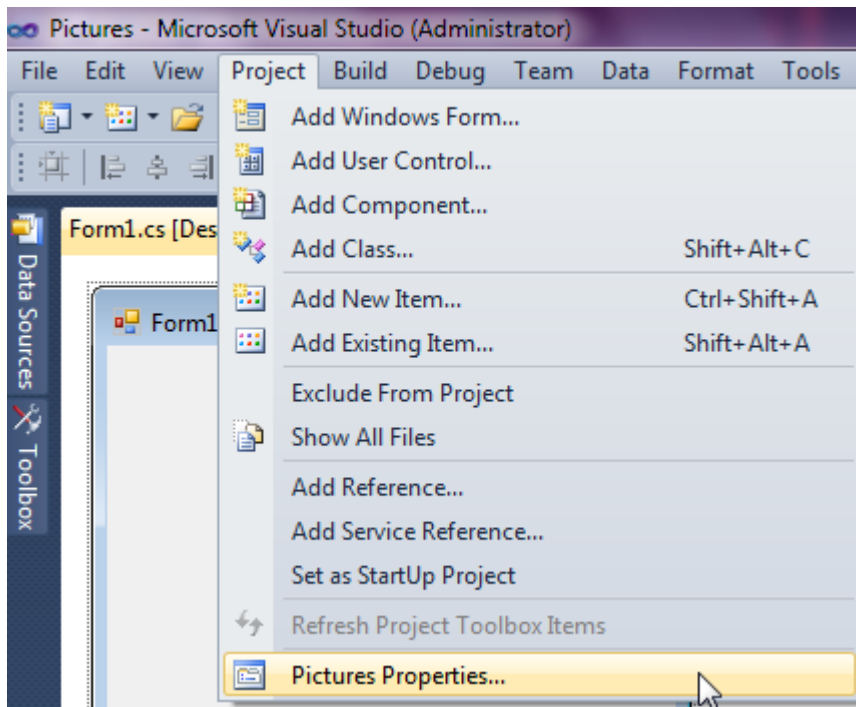


Select Windows Forms Application, set project name – "Pictures", set directory to save the project to.

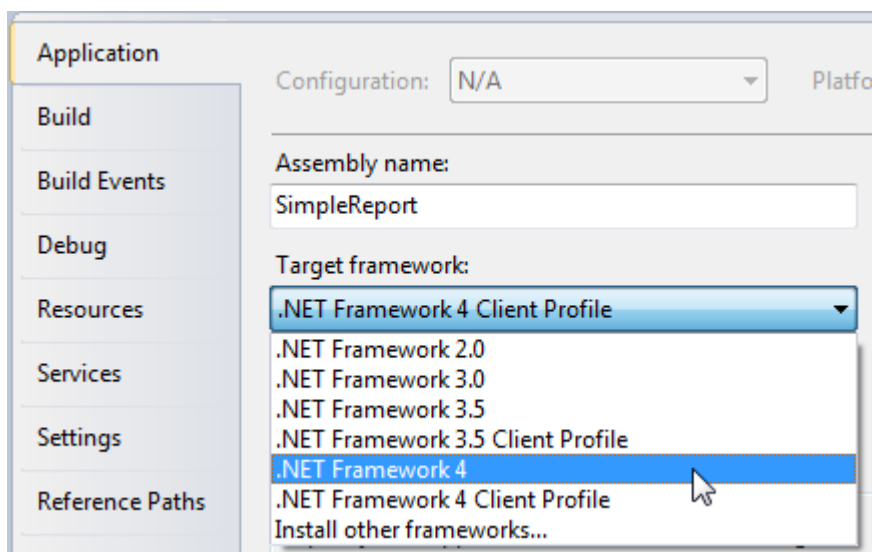


Step 2

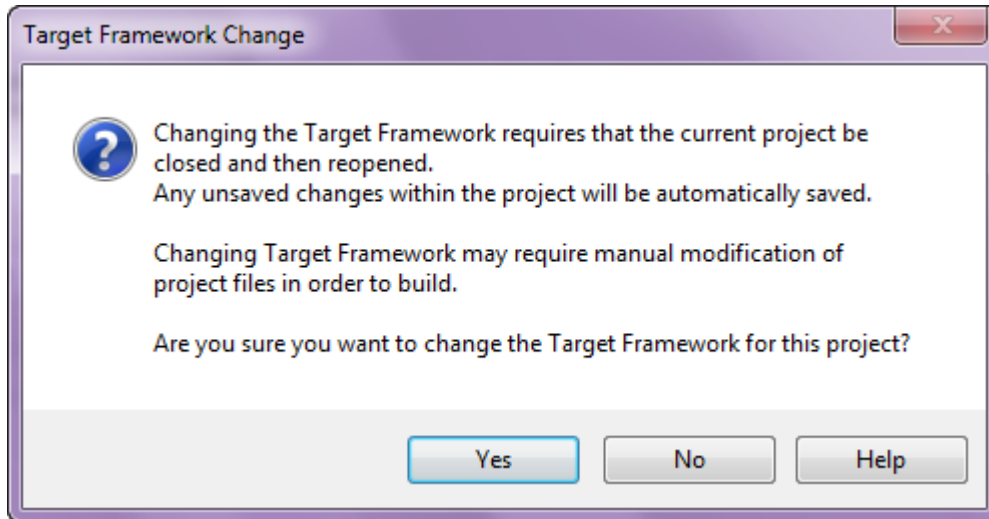
Change the project properties. Select the Project\Pictures Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

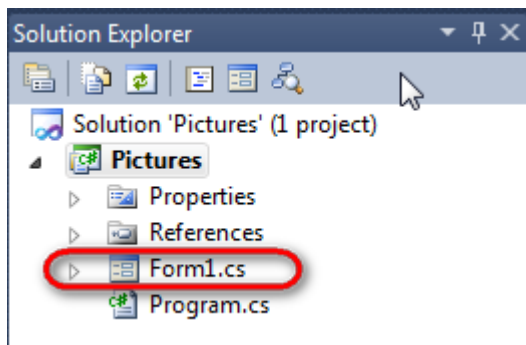


Press the "Yes" button in the opened window.

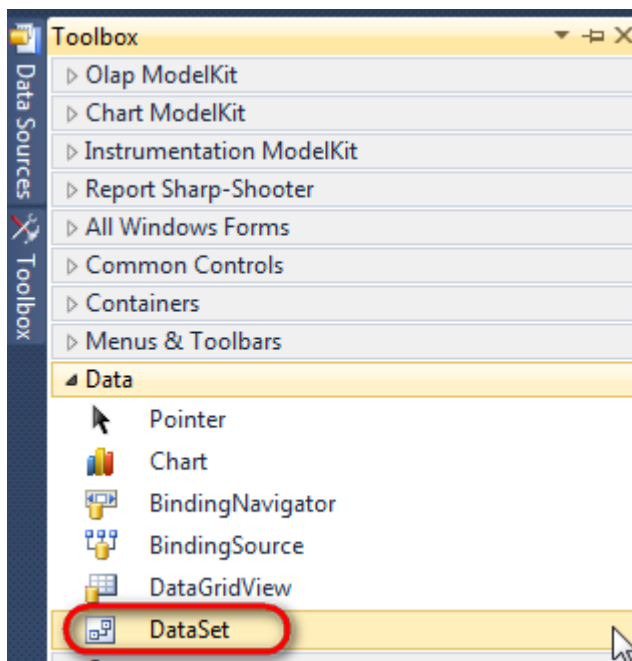


Step 3

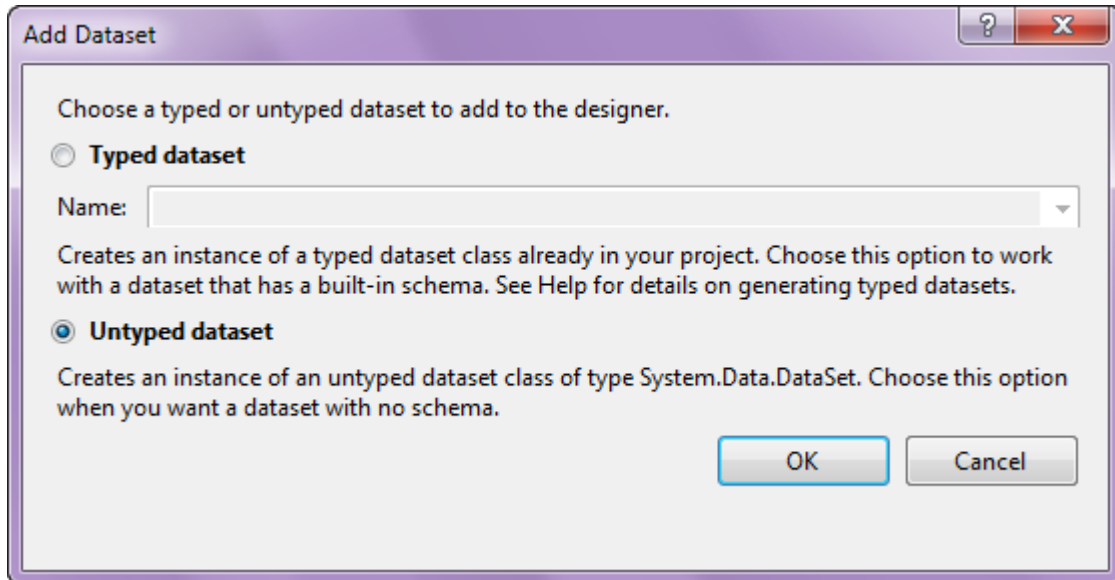
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



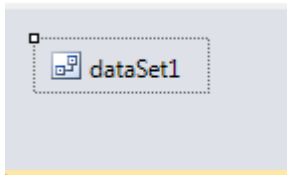
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

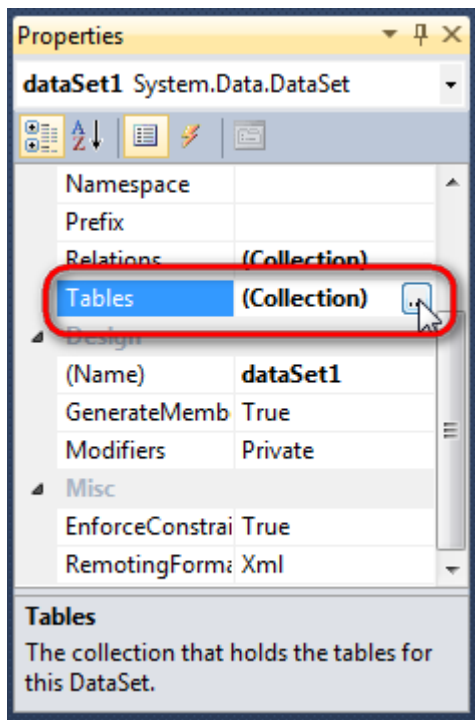


The component is available in the lower part of the window.

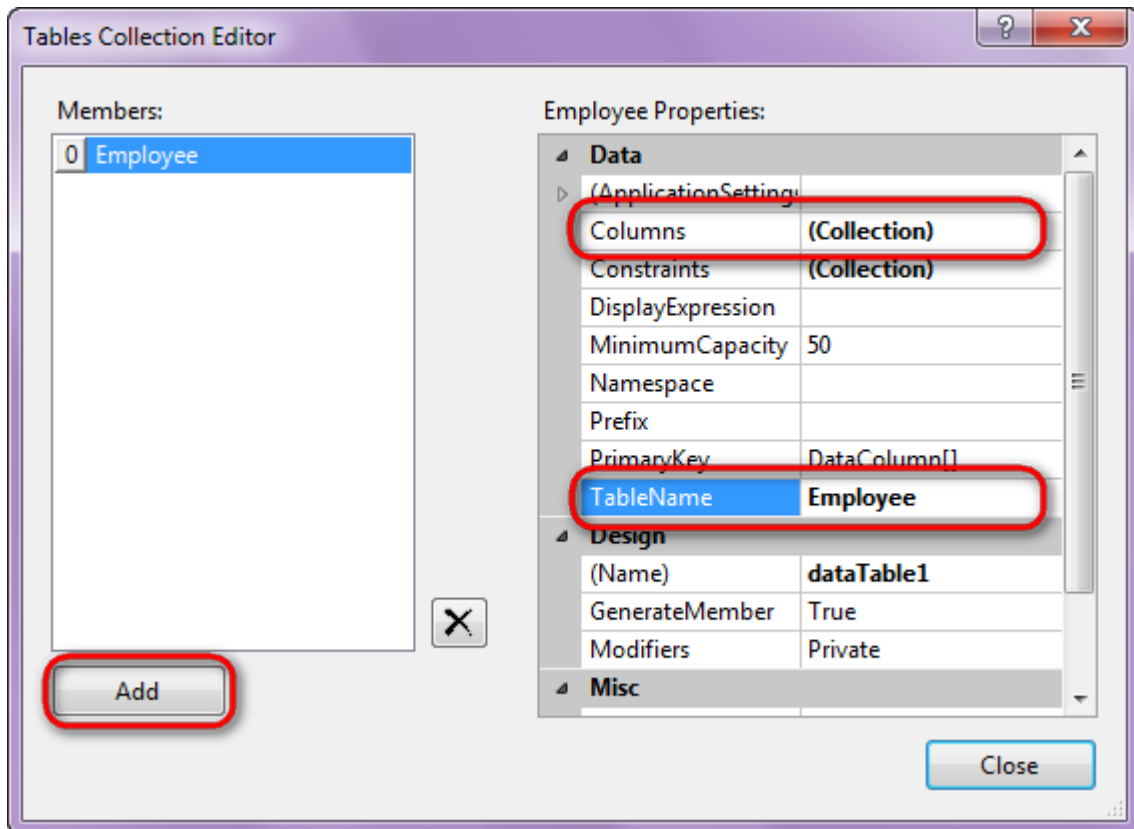


Step 4



Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

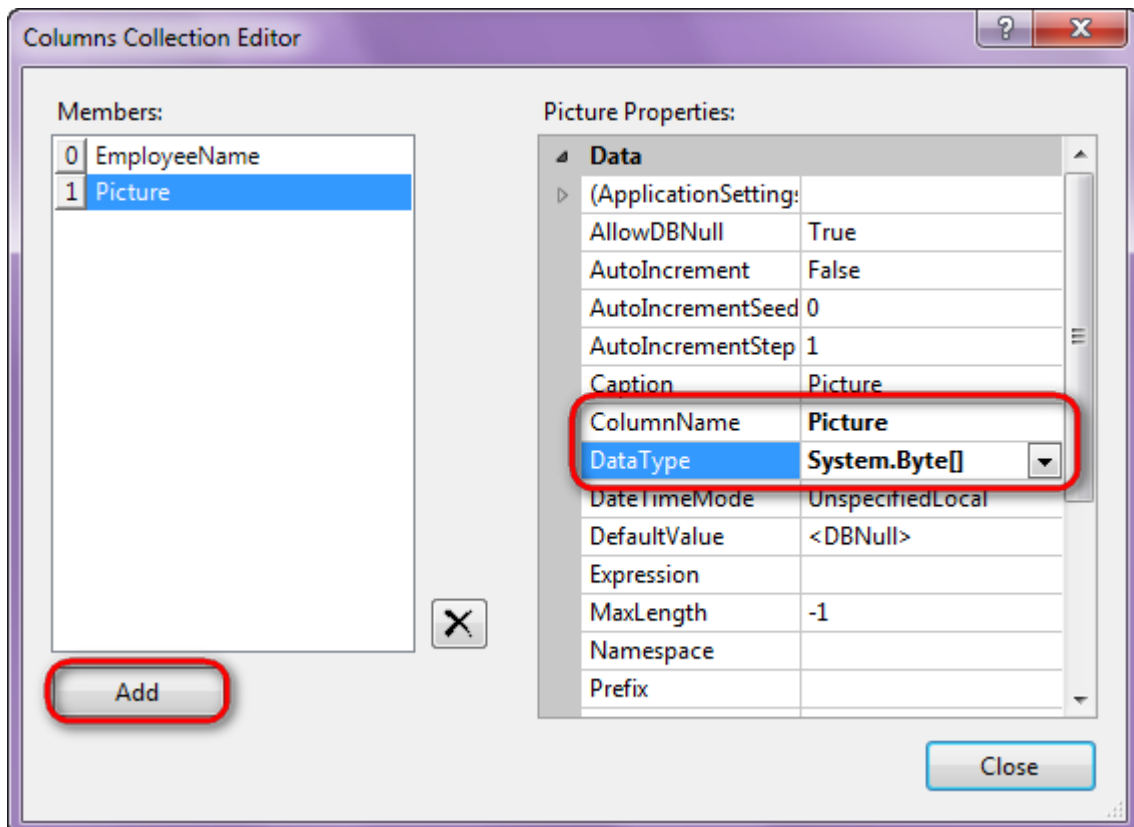


Click "Add" in order to add table. Set property TableName = Employee.



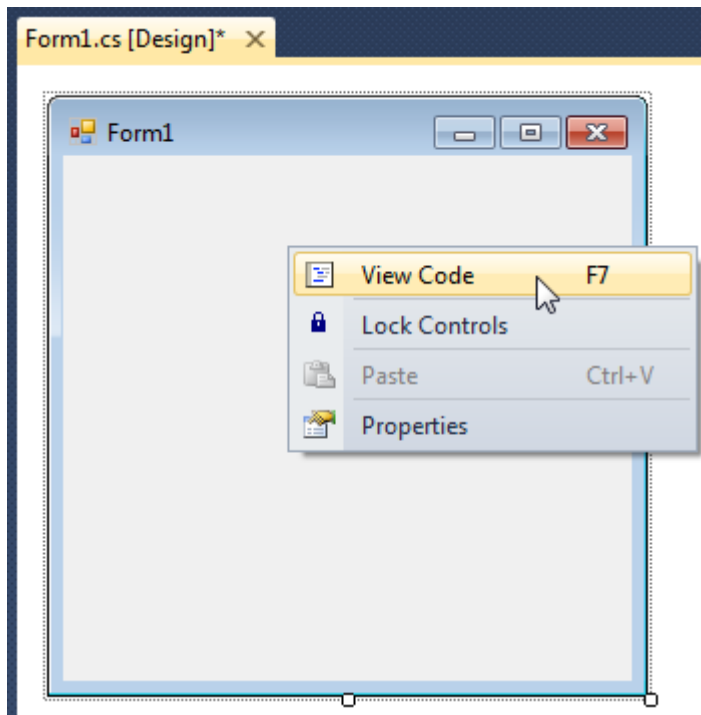
Step 5

Select Columns property, click button  to open Columns Collection Editor. Click "Add" to add a new column. Add two columns. Set ColumnName properties to "EmployeeName" and "Picture". For the Picture column, select DataType property, click button , select "System.Byte[]" value in the list.



Step 6

Right click on the form and select "View Code" in the context menu to view code.

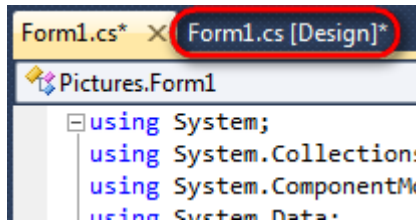


Add the following code to the class constructor in order to fill data source.

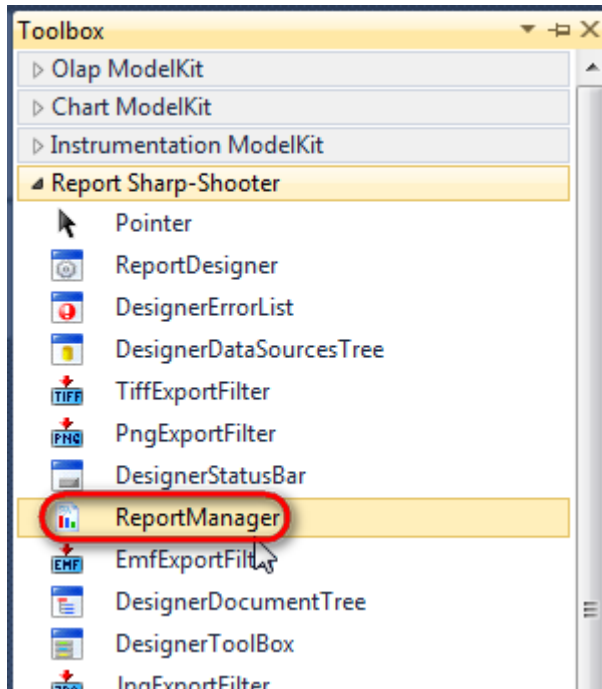
```
public Form1()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["EmployeeName"] = "Nancy Davolio";
    Image img =
Image.FromFile("C:\\data\\pictures\\NancyDavolio.png");
    Byte[] pic = imageToByteArray(img);
    row["Picture"] = pic;
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["EmployeeName"] = "Andrew Fuller";
    img = Image.FromFile("C:\\data\\pictures\\AndrewFuller.png");
    pic = imageToByteArray(img);
    row["Picture"] = pic;
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["EmployeeName"] = "Robert King";
    img = Image.FromFile("C:\\data\\pictures\\RobertKing.png");
    pic = imageToByteArray(img);
    row["Picture"] = pic;
    dataTable1.Rows.Add(row);
}
//Function designed to transform image to byte code:
public byte[] imageToByteArray(System.Drawing.Image imageIn)
{
    System.IO.MemoryStream ms = new System.IO.MemoryStream();
    imageIn.Save(ms, System.Drawing.Imaging.ImageFormat.Png);
    return ms.ToArray();
}
```

Step 7

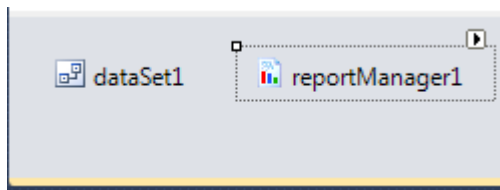
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

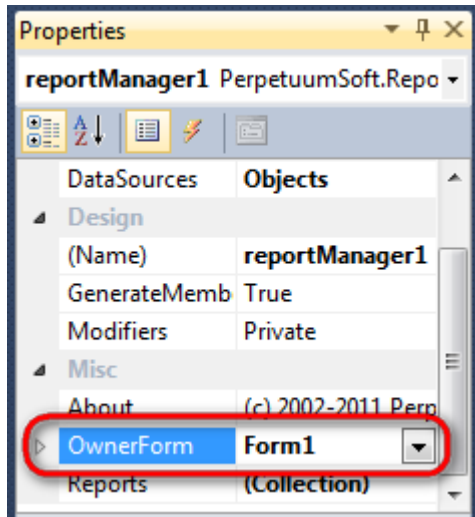


The component is available in the lower part of the window.



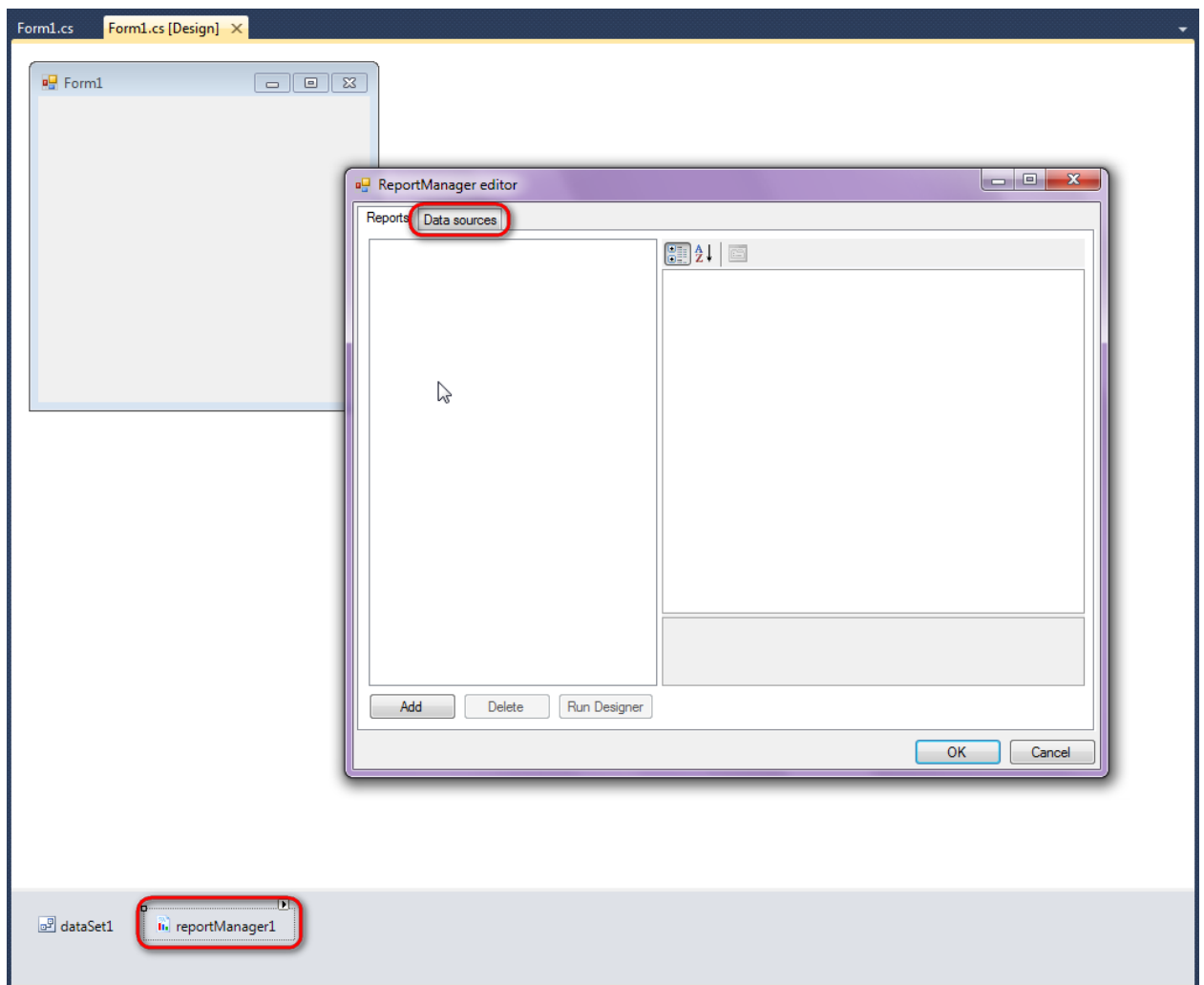
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

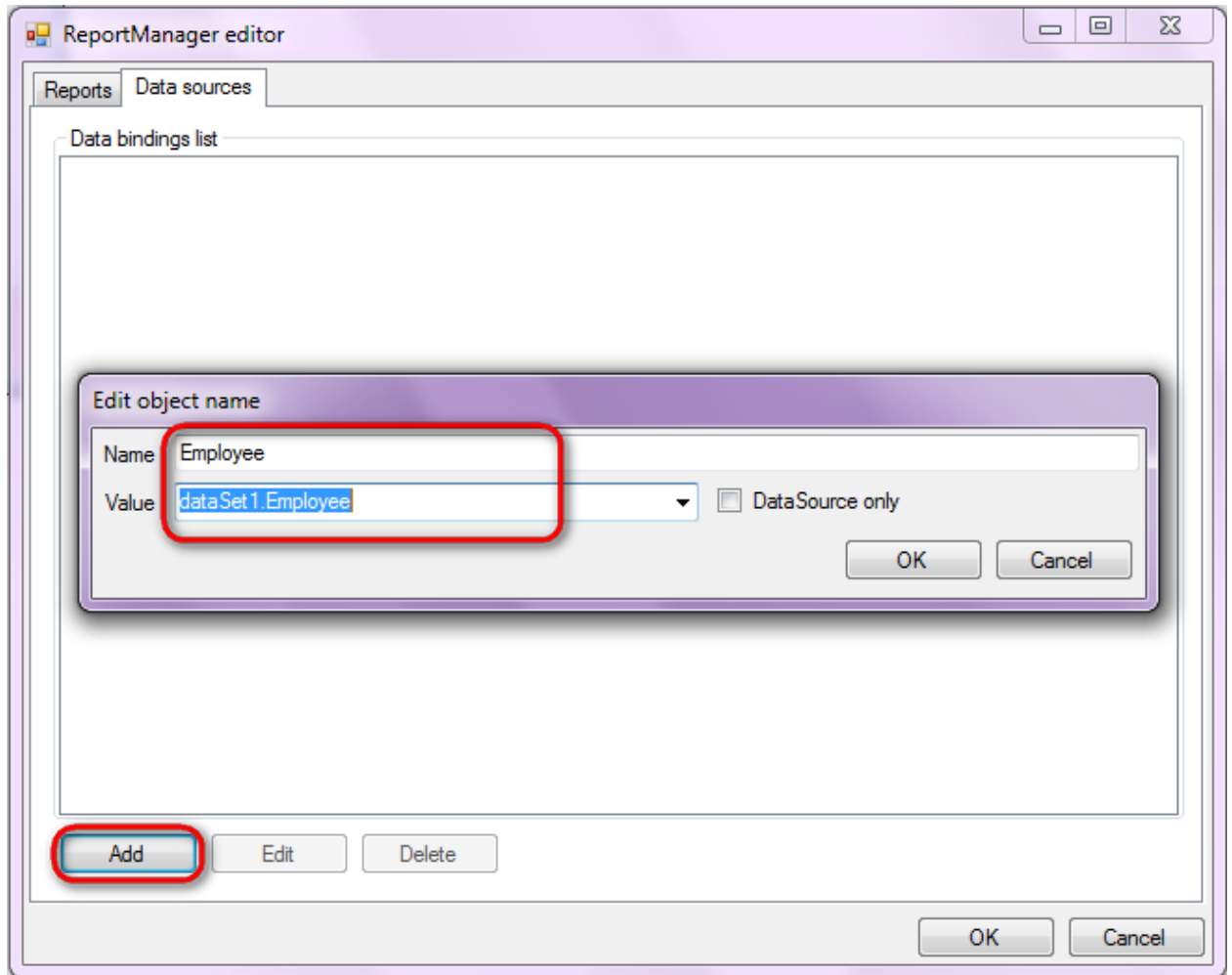


Step 9

Double click on ReportManager to open ReportManager editor.

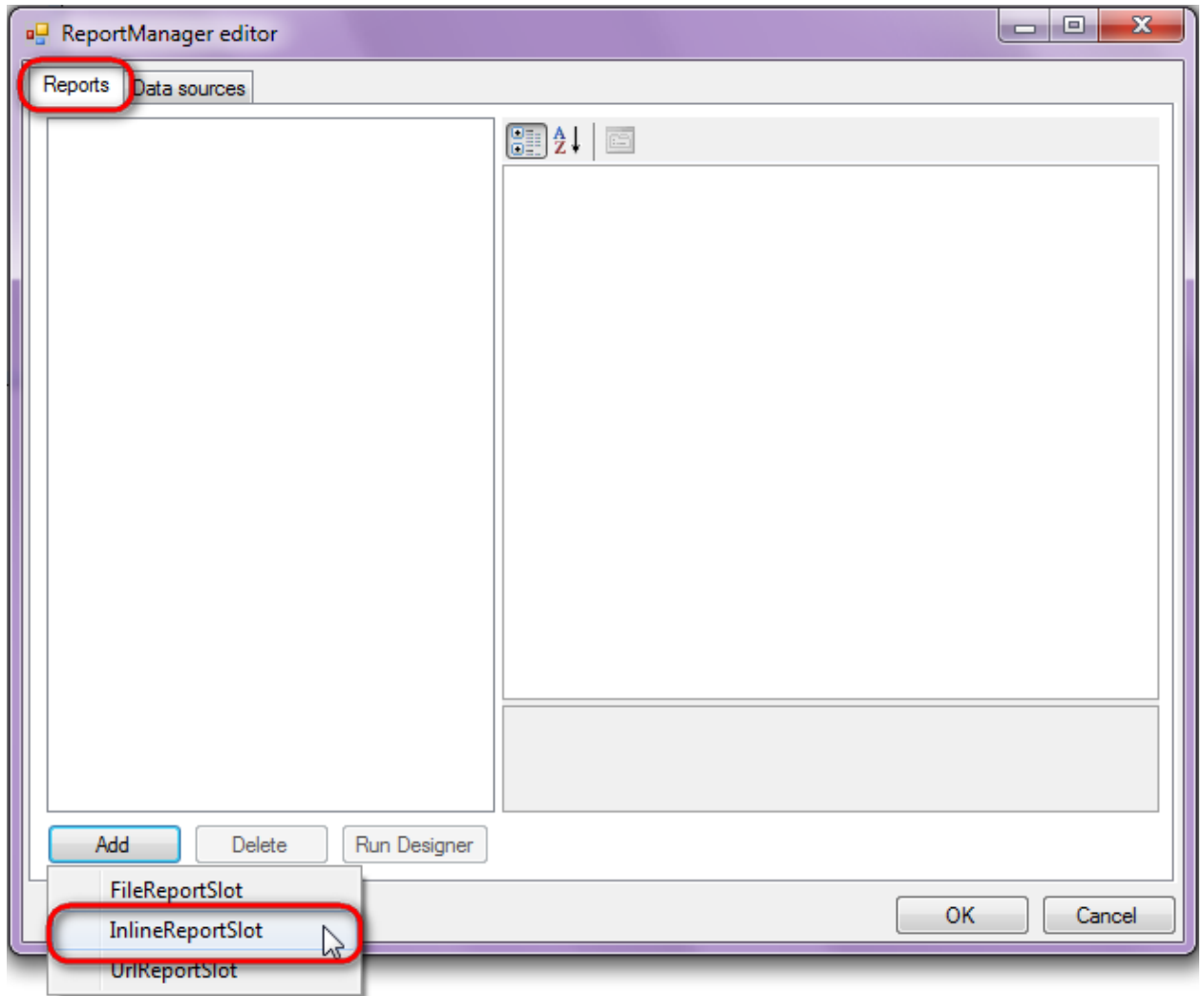


Go to "Data Sources" tab, click "Add", set data source name – "Employee", select data source value – "dataSet1.Employee".



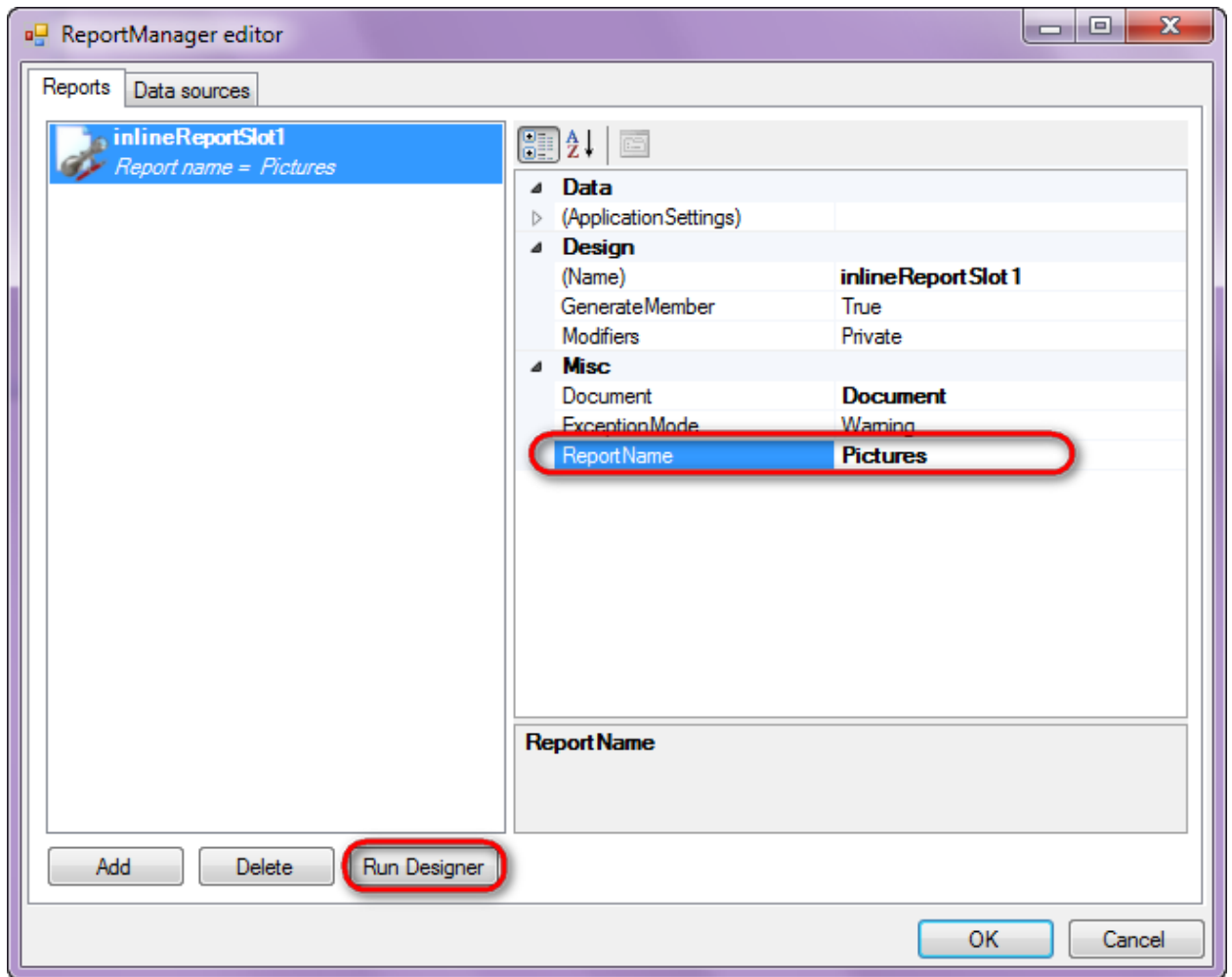
Step 10

Go to "Reports" tab, click "Add" and select "InlineReportSlot".



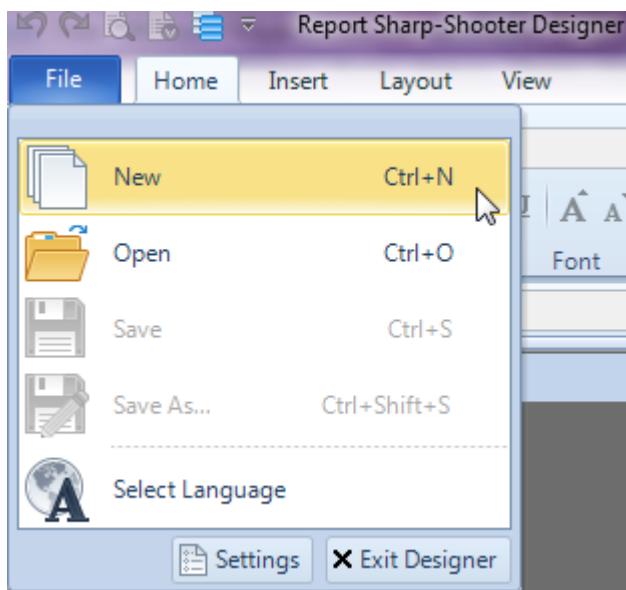
Step 11

Set name of the report in the property ReportName - "Pictures". Click "Run Designer" in order to open template editor - Report Designer.

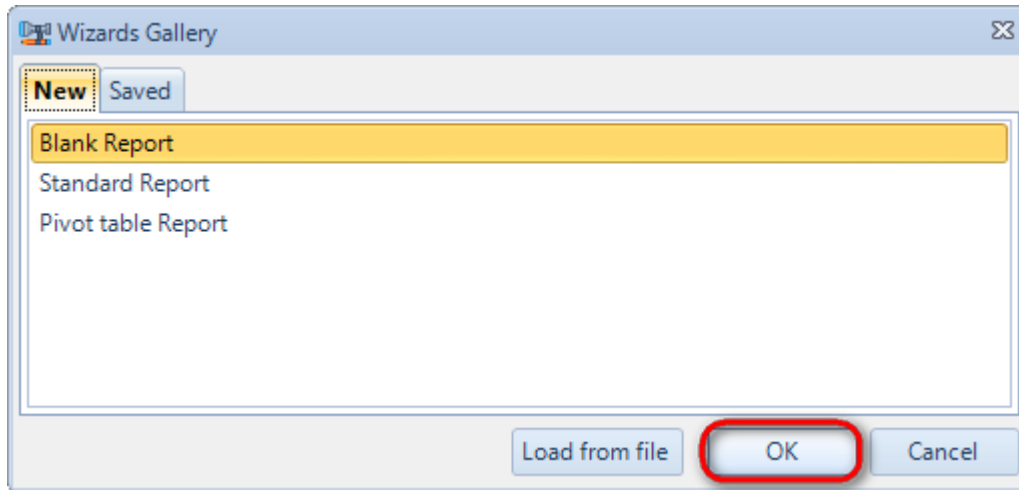


Step 12

Create new empty template – select File\New in the main menu.

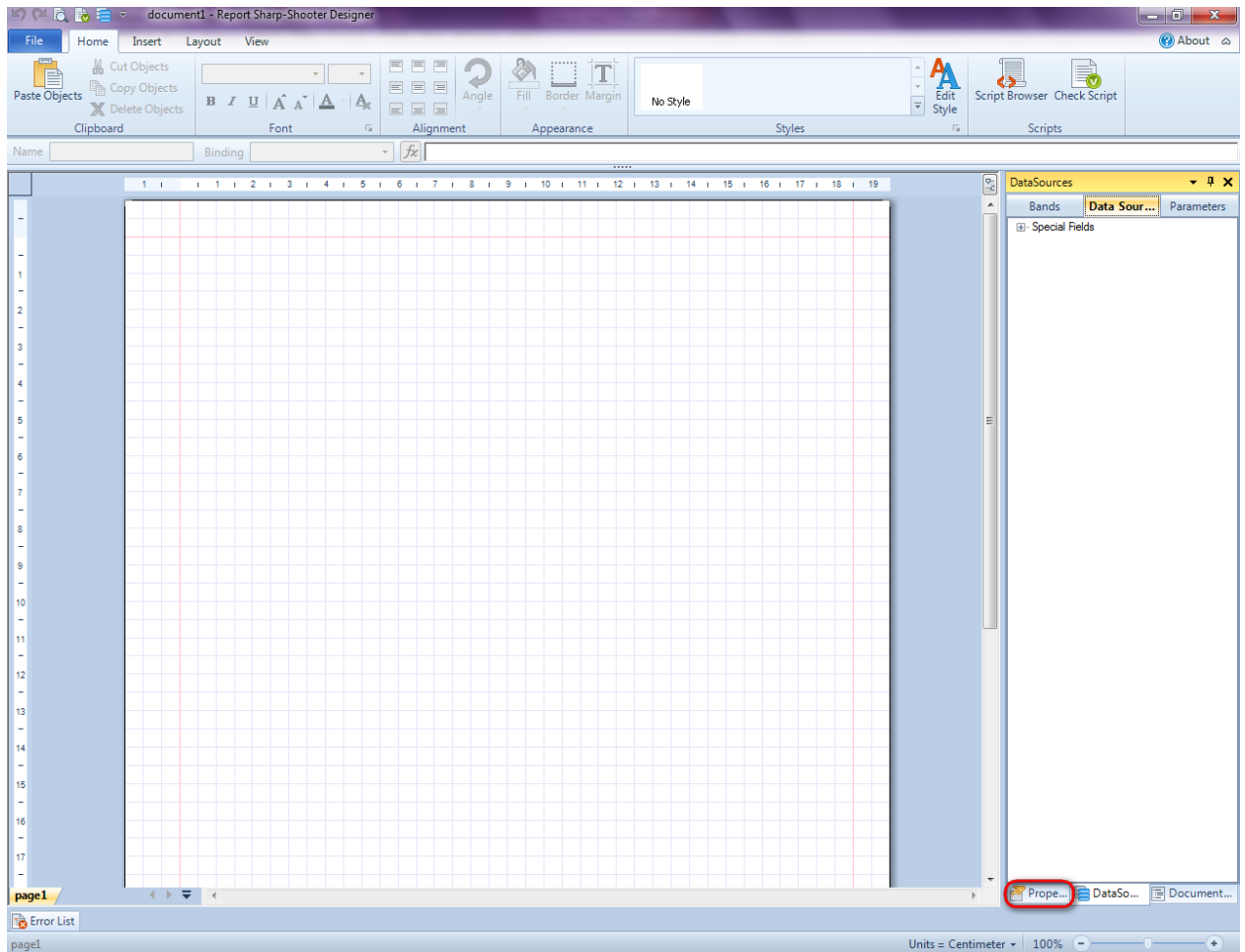


Select "Blank Report" in the Wizards Gallery and click "OK".



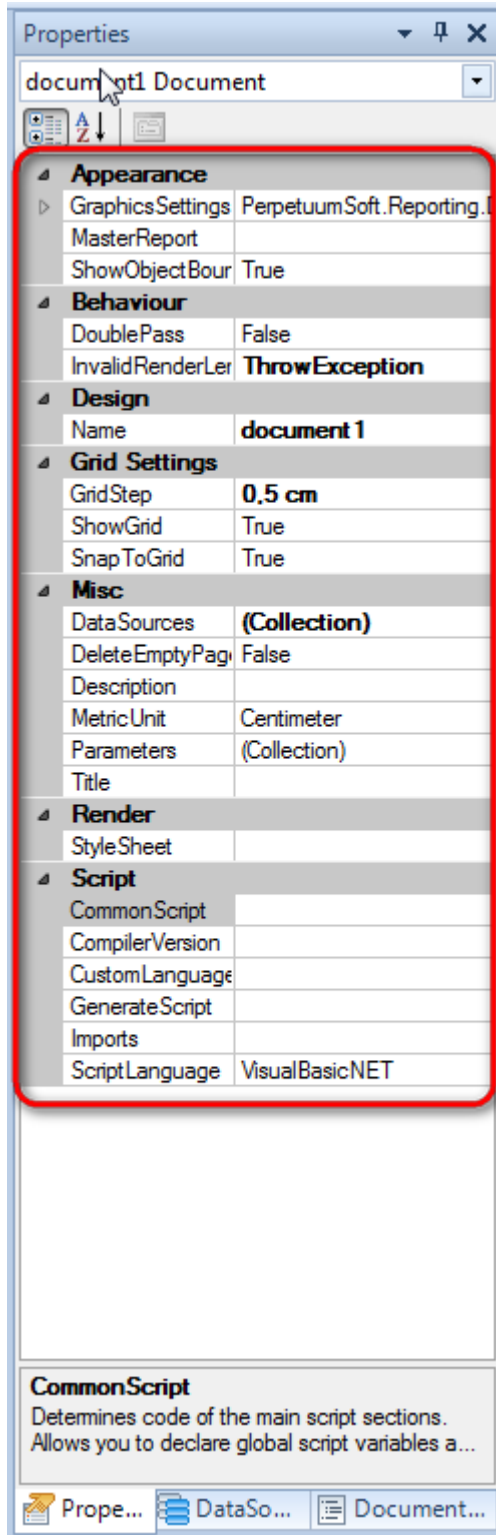
Step 13

Click the "Properties" tab of the tool window in the right part of the designer.

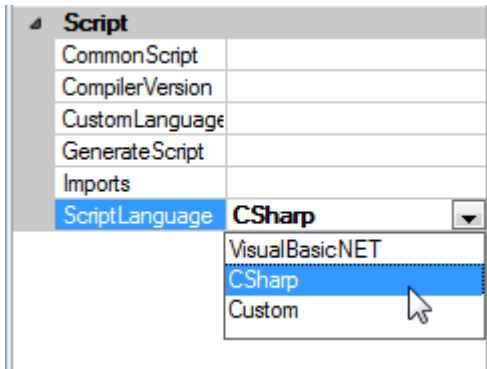




You will see properties of the edited template on the "Properties" tab

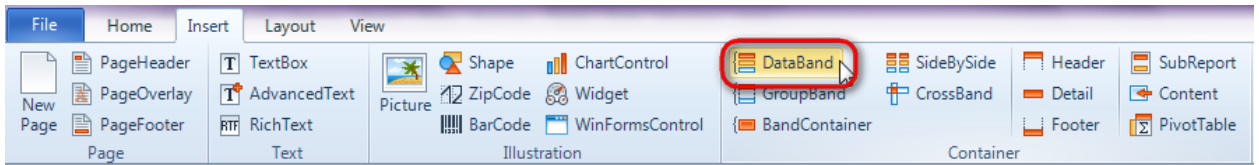


Set property ScriptLanguage = CSharp.



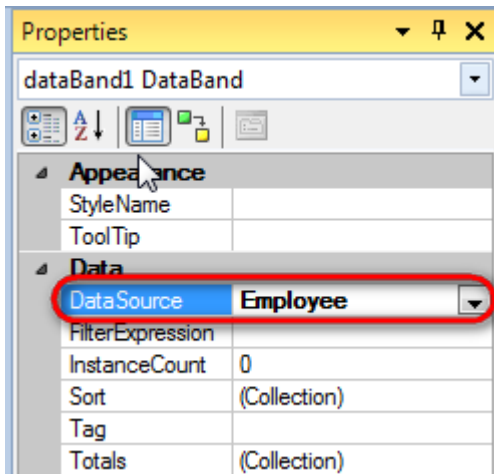
Step 14

Press "DataBand" button on the Insert tab in the group Container.

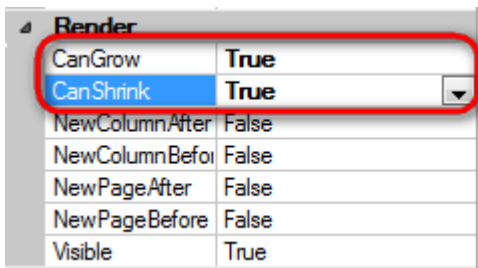


Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = Employee.

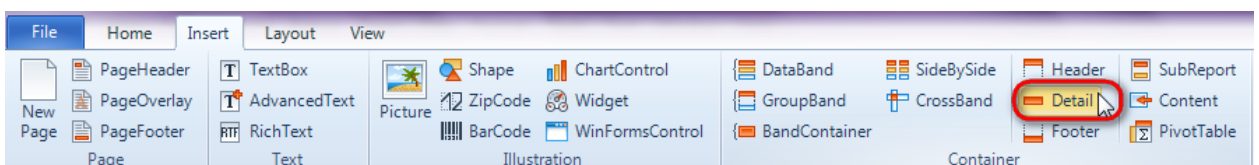


On the "Properties" tab, set CanShrink and CanGrow properties to "True".



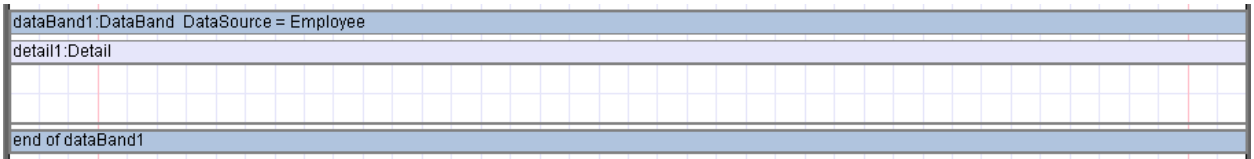
Step 15

Press "Detail" button on the Insert tab in the group Container.





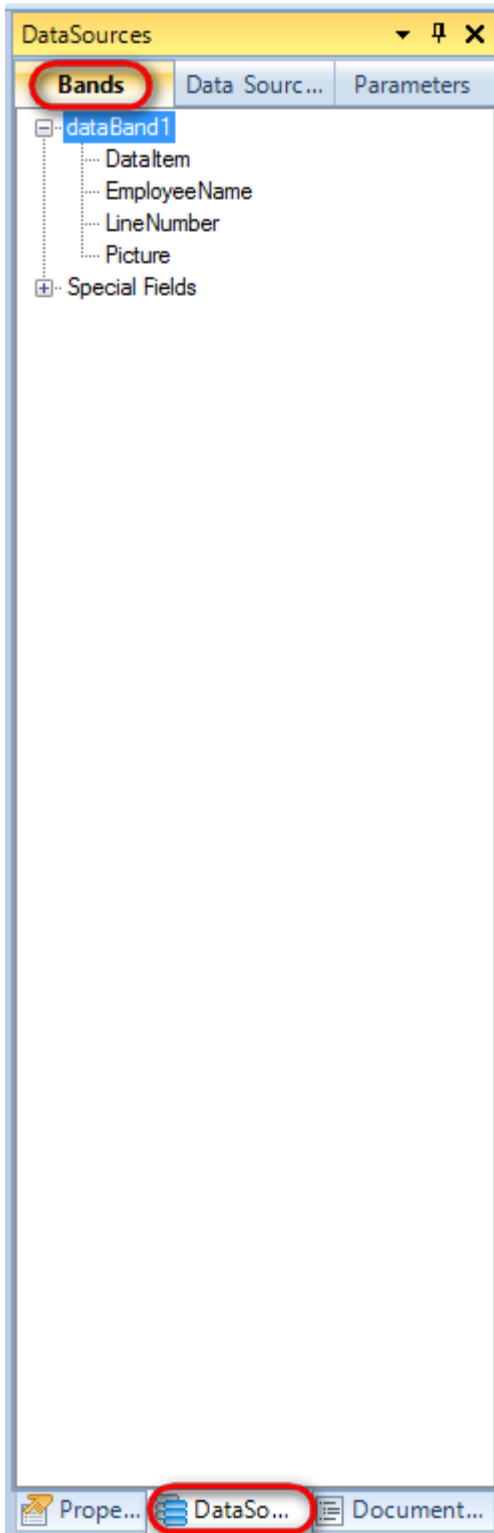
Click on the DataBand area to add Detail band inside DataBand.



On the "Properties" tab, set CanShrink and CanGrow properties to "True".

Step 16

Go to "DataSources" tab.



Drag and drop "EmployeeName" field from the dataBand1 tree o the detail1 band. As a result TextBox is created. Script loading data from the data source is added to the Value property.

```
dataBand1:DataBand DataSource = New_Name
detail1:Detail
<dataBand1
  ["EmployeeName"]>
end of dataBand1
```

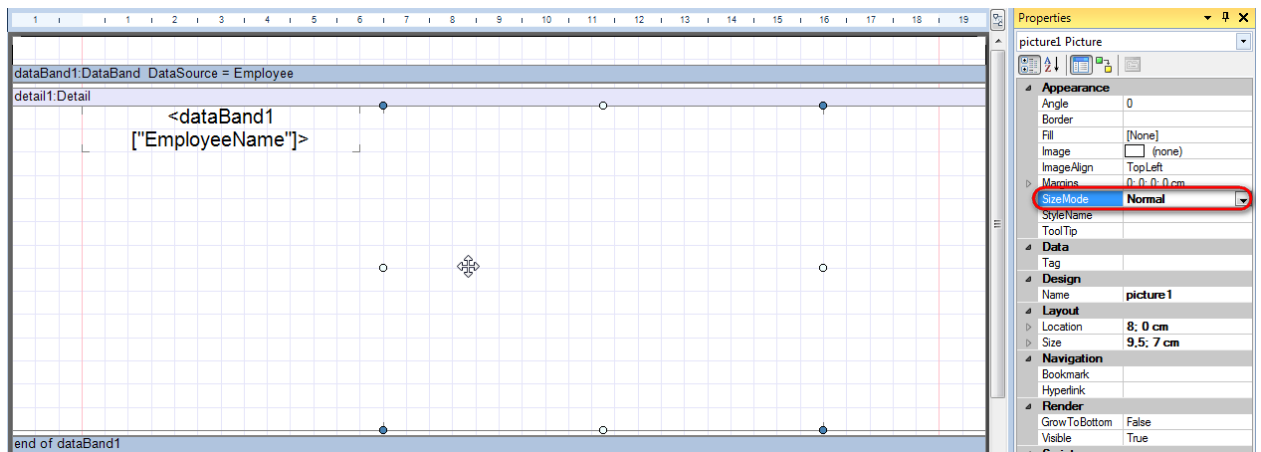


Step 17

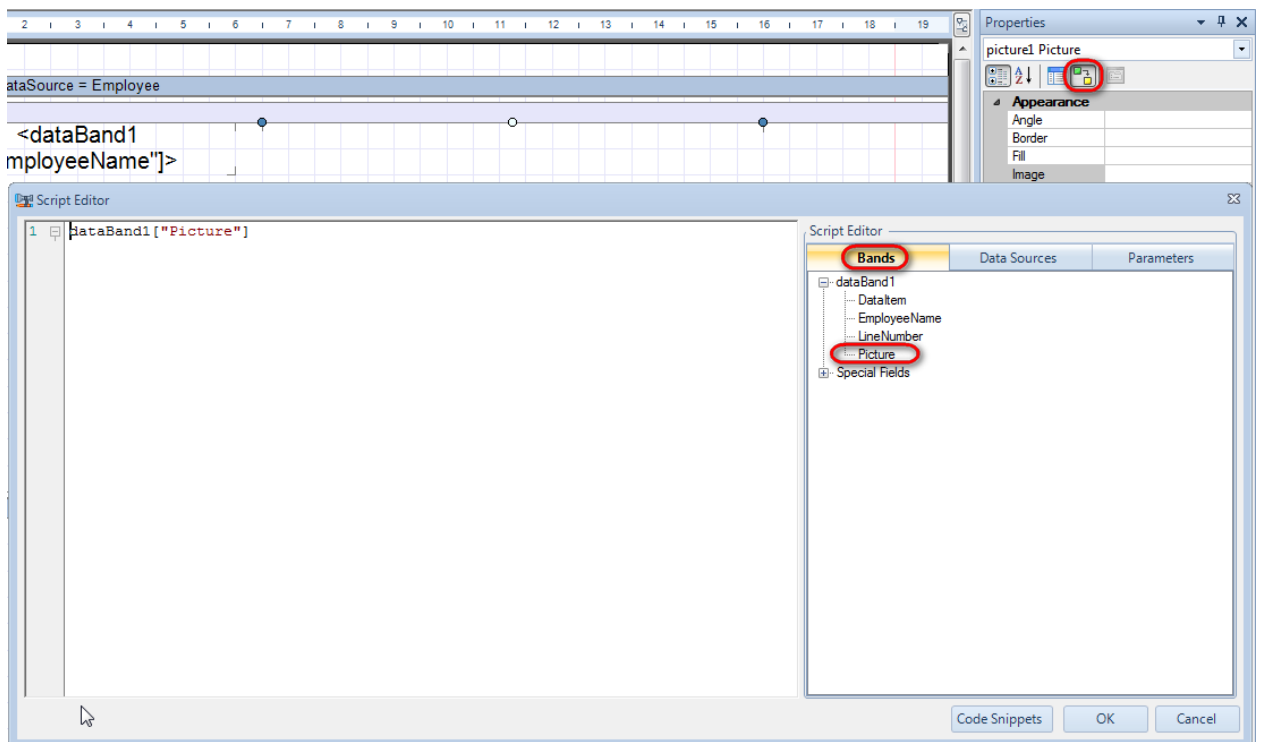
Press "Picture" button on the Insert tab in the group Illustration.



Click on the Detail band area to add Picture element inside Detail. Change size of the bands and the Picture element. Set SizeMode = Normal on the "Properties" tab.



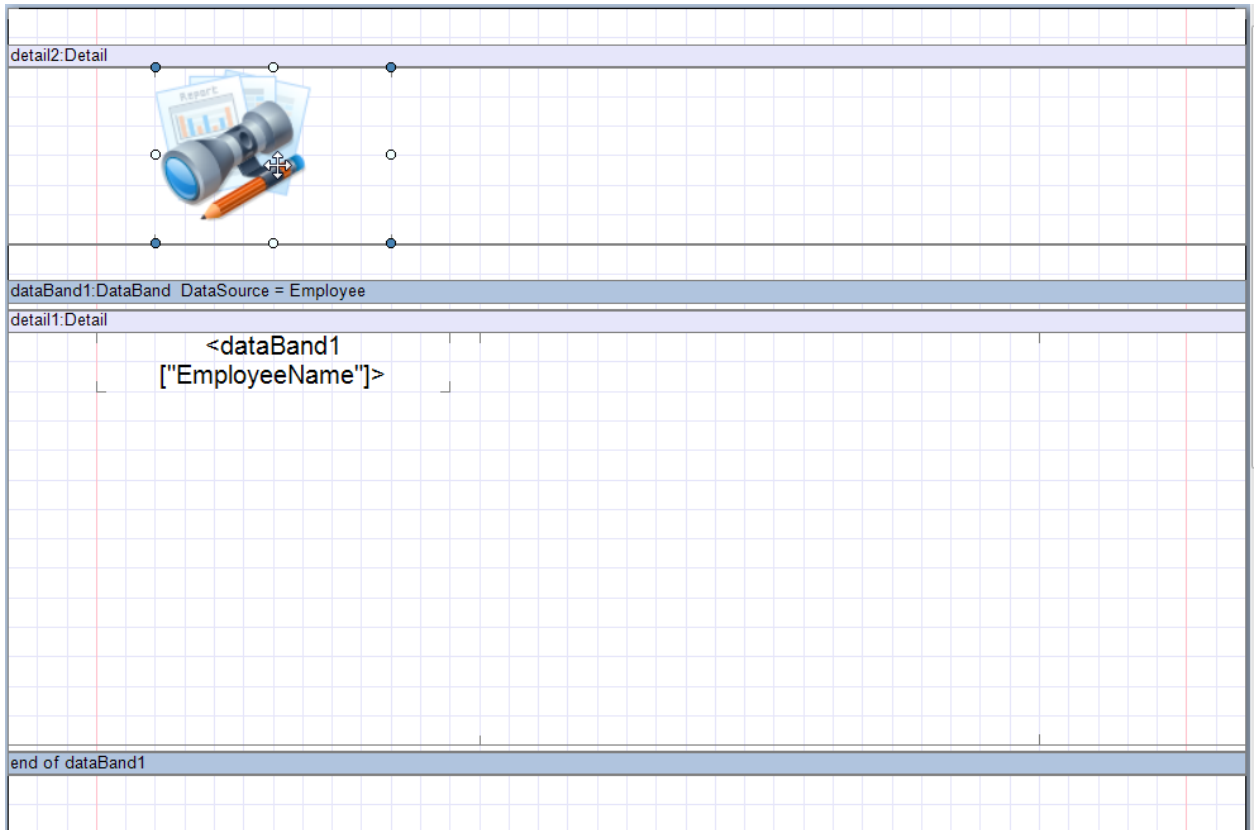
Click "Bindings" button, set Image property to "dataBand1[\"Picture\"]".



Step 18

Add one more Detail band, add there Picture element. Double click on the element to open dialog to select an image. Select picture and click "OK".

Report template should look as follows:



Step 19

Save template, close Report Designer.

Step 20

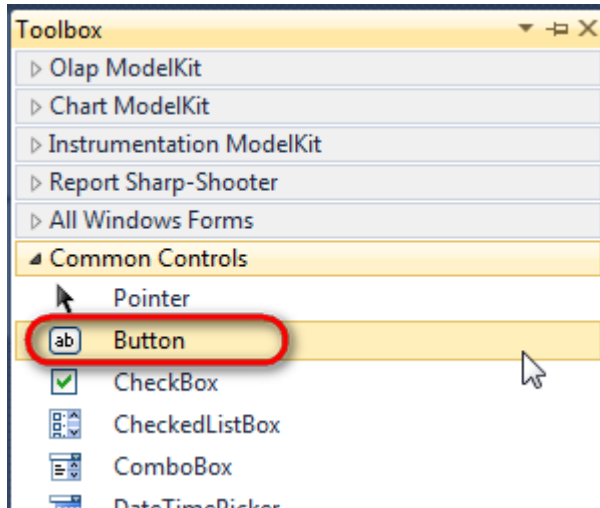
Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["EmployeeName"] = "Nancy Davolio";
    Image img =
Image.FromFile("C:\\data\\pictures\\NancyDavolio.png");
    Byte[] pic = imageToByteArray(img);
    row["Picture"] = pic;
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["EmployeeName"] = "Andrew Fuller";
    img = Image.FromFile("C:\\data\\pictures\\AndrewFuller.png");
    pic = imageToByteArray(img);
    row["Picture"] = pic;
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["EmployeeName"] = "Robert King";
    img = Image.FromFile("C:\\data\\pictures\\RobertKing.png");
    pic = imageToByteArray(img);
    row["Picture"] = pic;
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
```

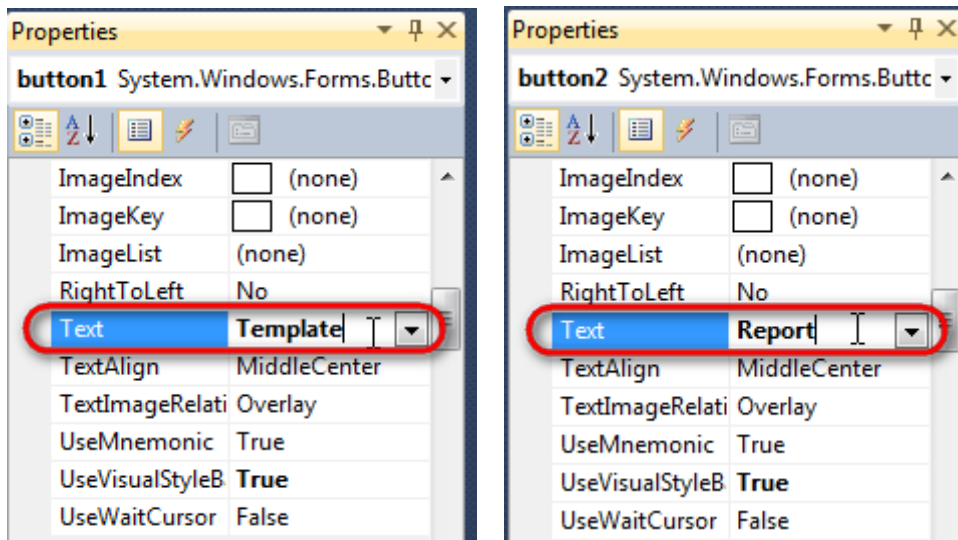
```
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 21

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



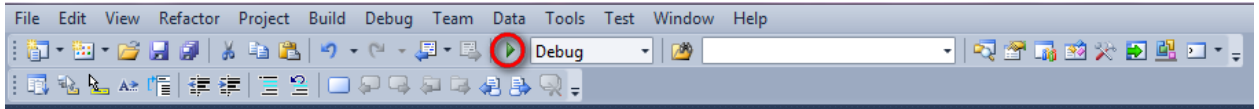
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}
```

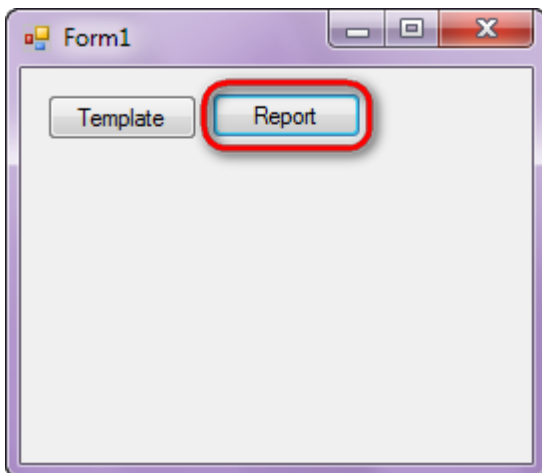
```
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

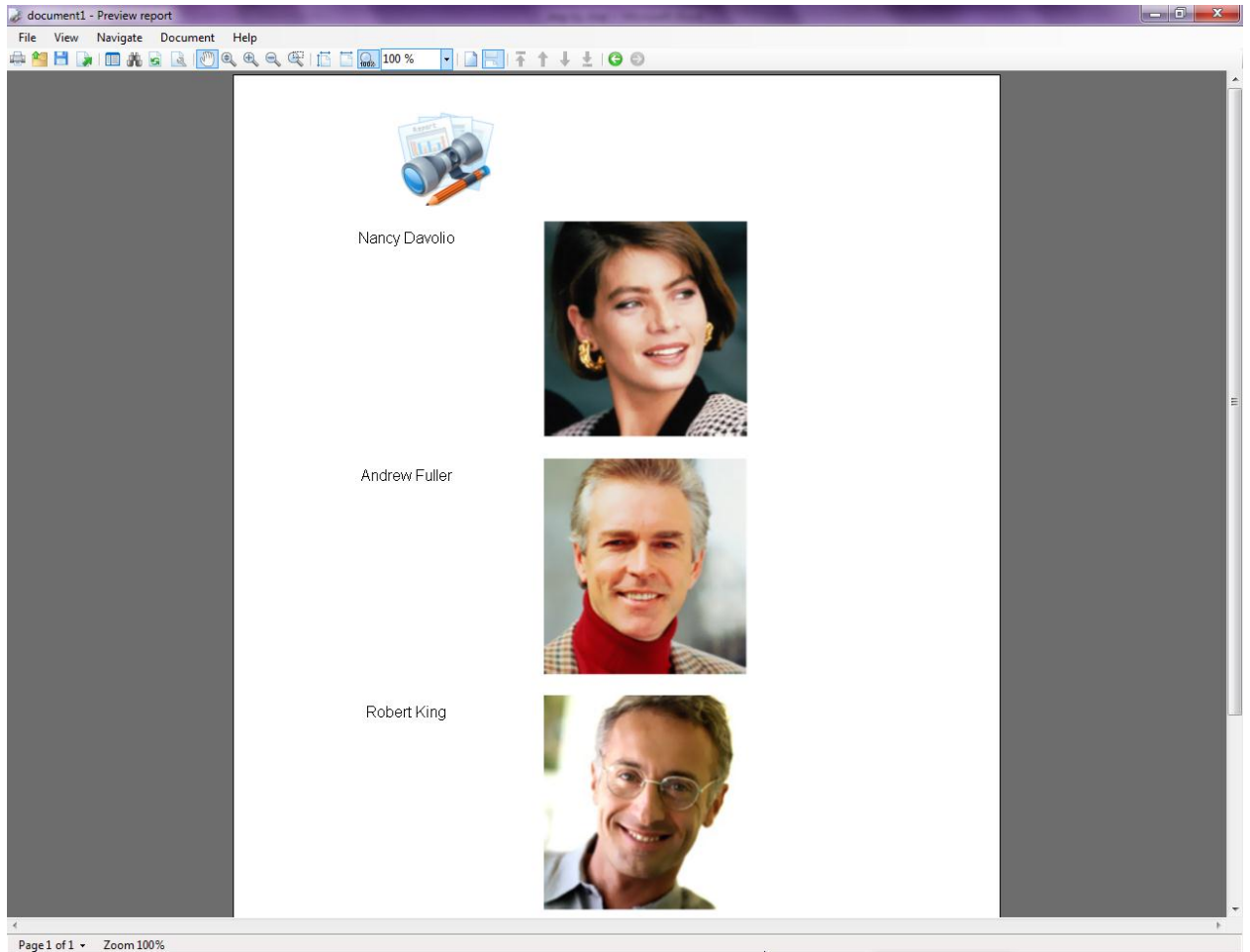
Click "Start Debugging" on the Visual Studio toolbar in order to run application.



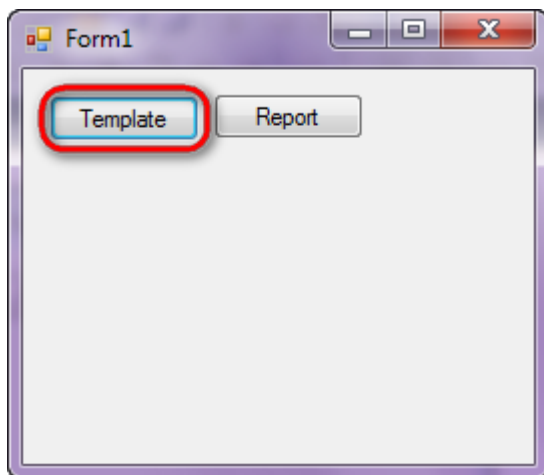
Click the "Report" button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Report Controls\Basic\Pictures.

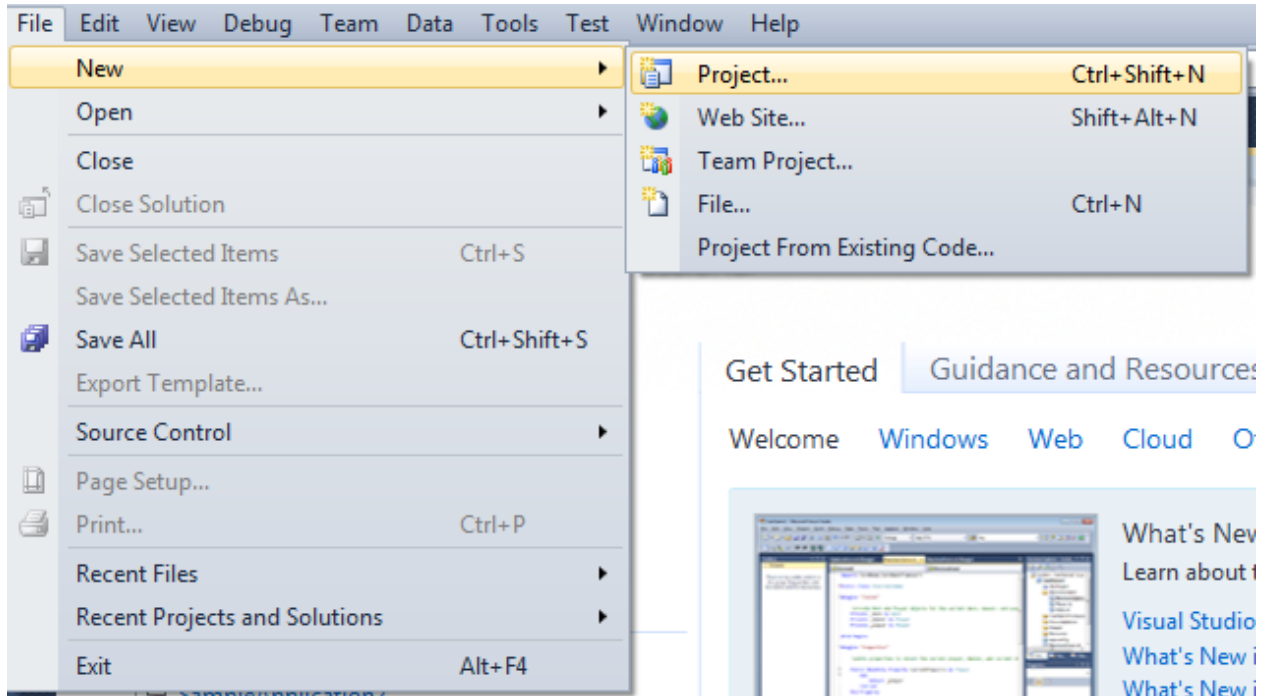


Shape

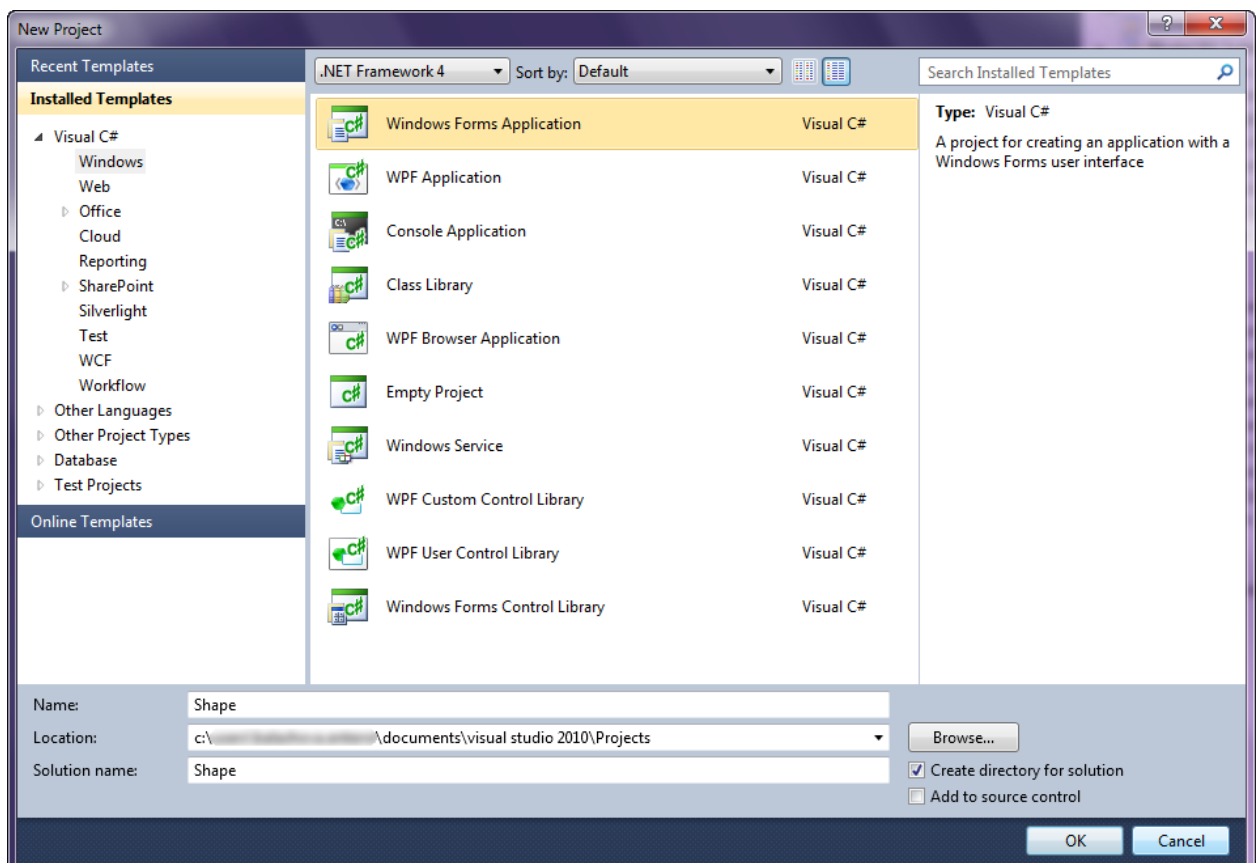
Template of a report containing various shapes.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

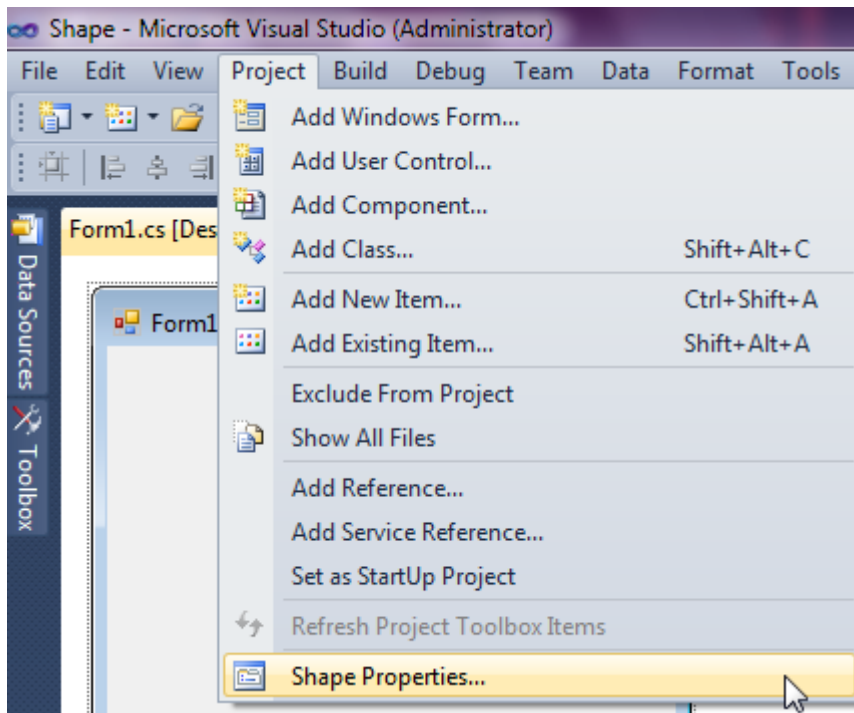


Select Windows Forms Application, set project name – “Shape”, set directory to save the project to.

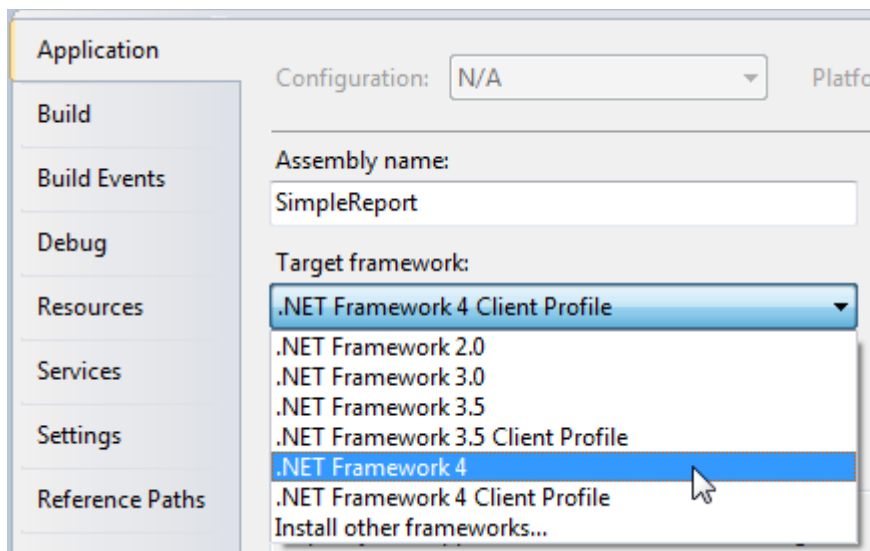


Step 2

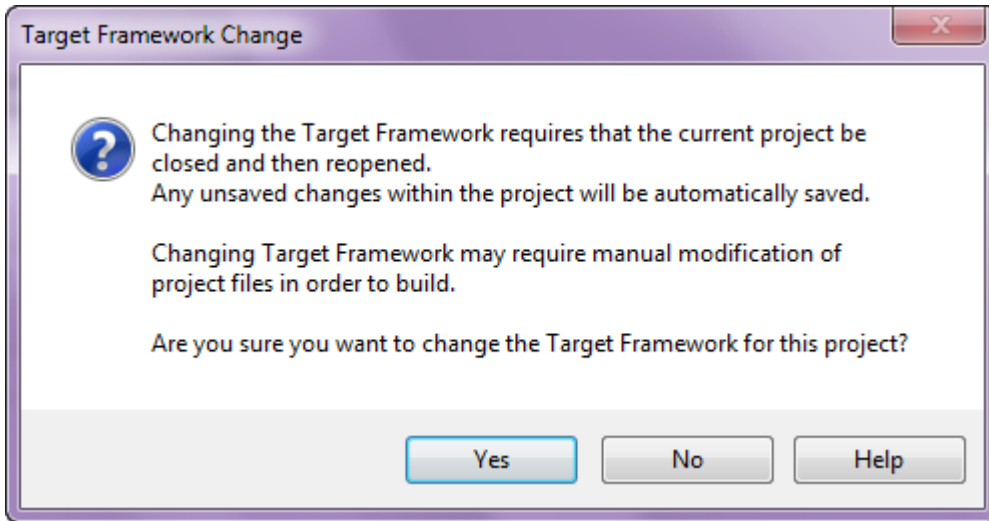
Change the project properties. Select the Project\Shape Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

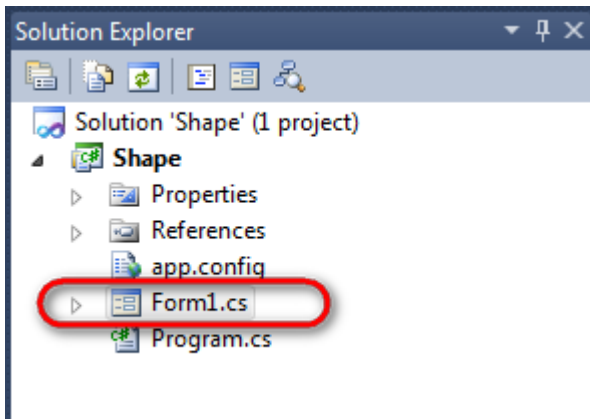


Press the "Yes" button in the opened window.

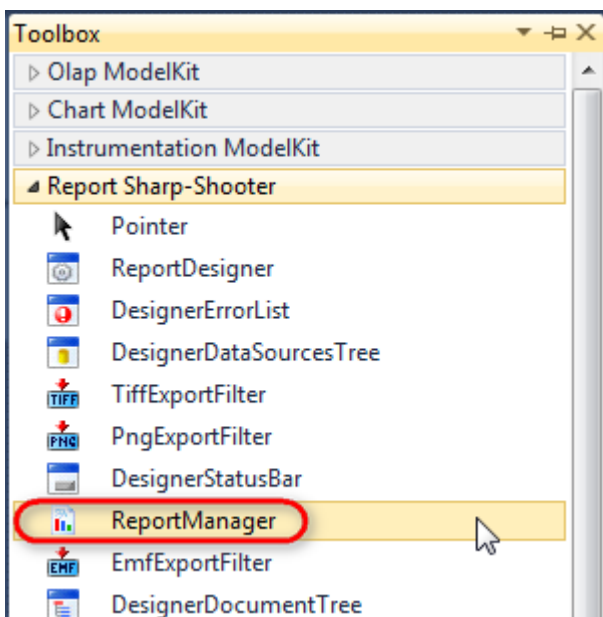


Step 3

Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

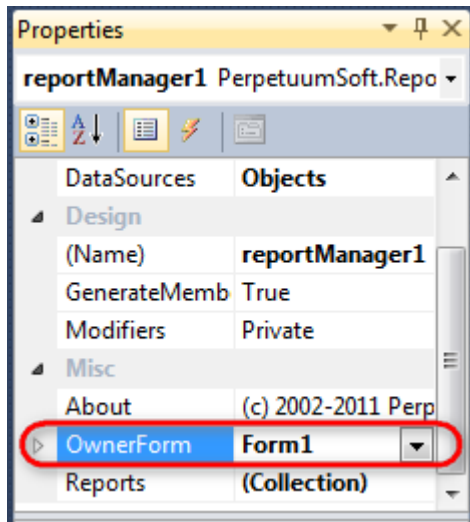


The component is available in the lower part of the window.



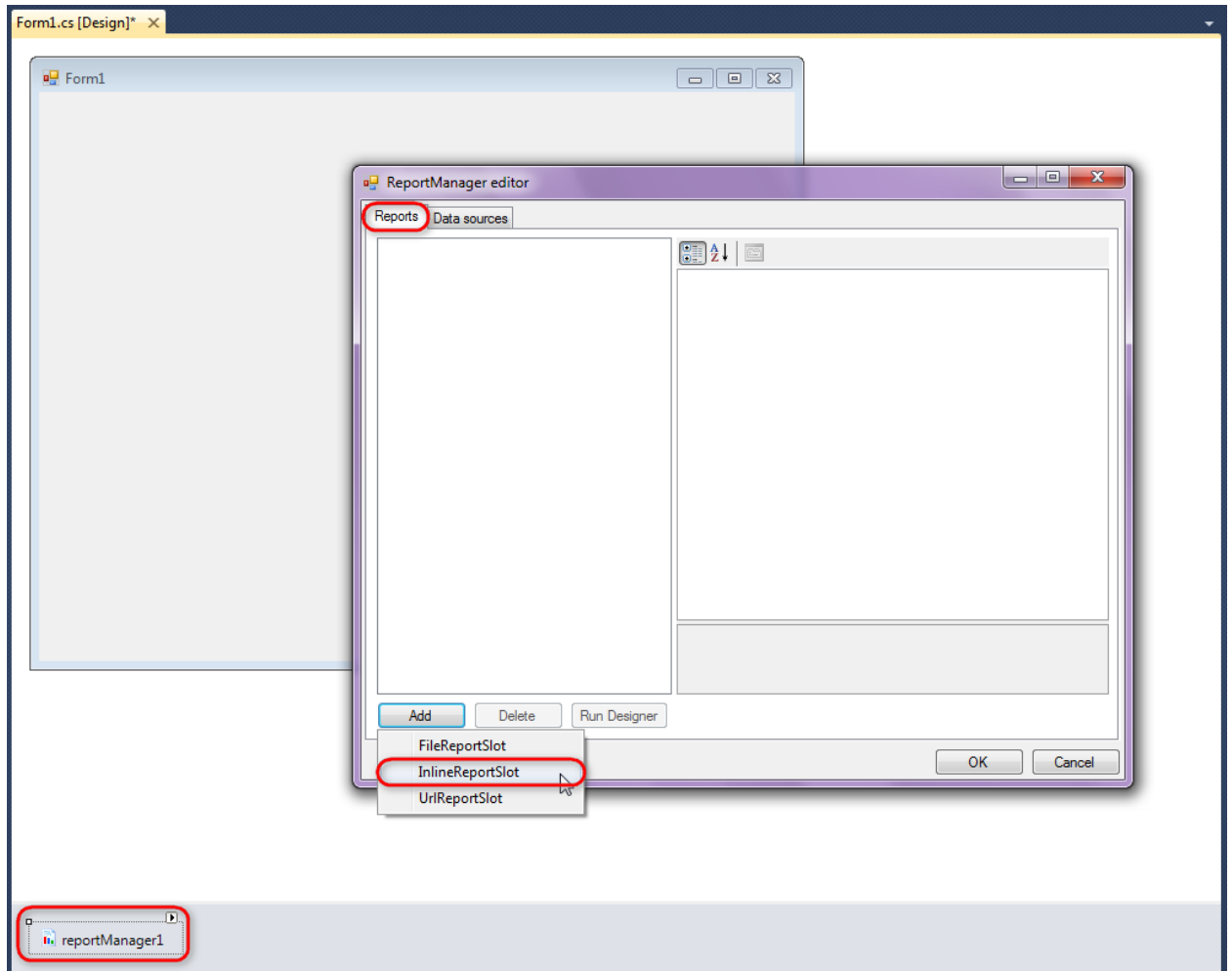
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



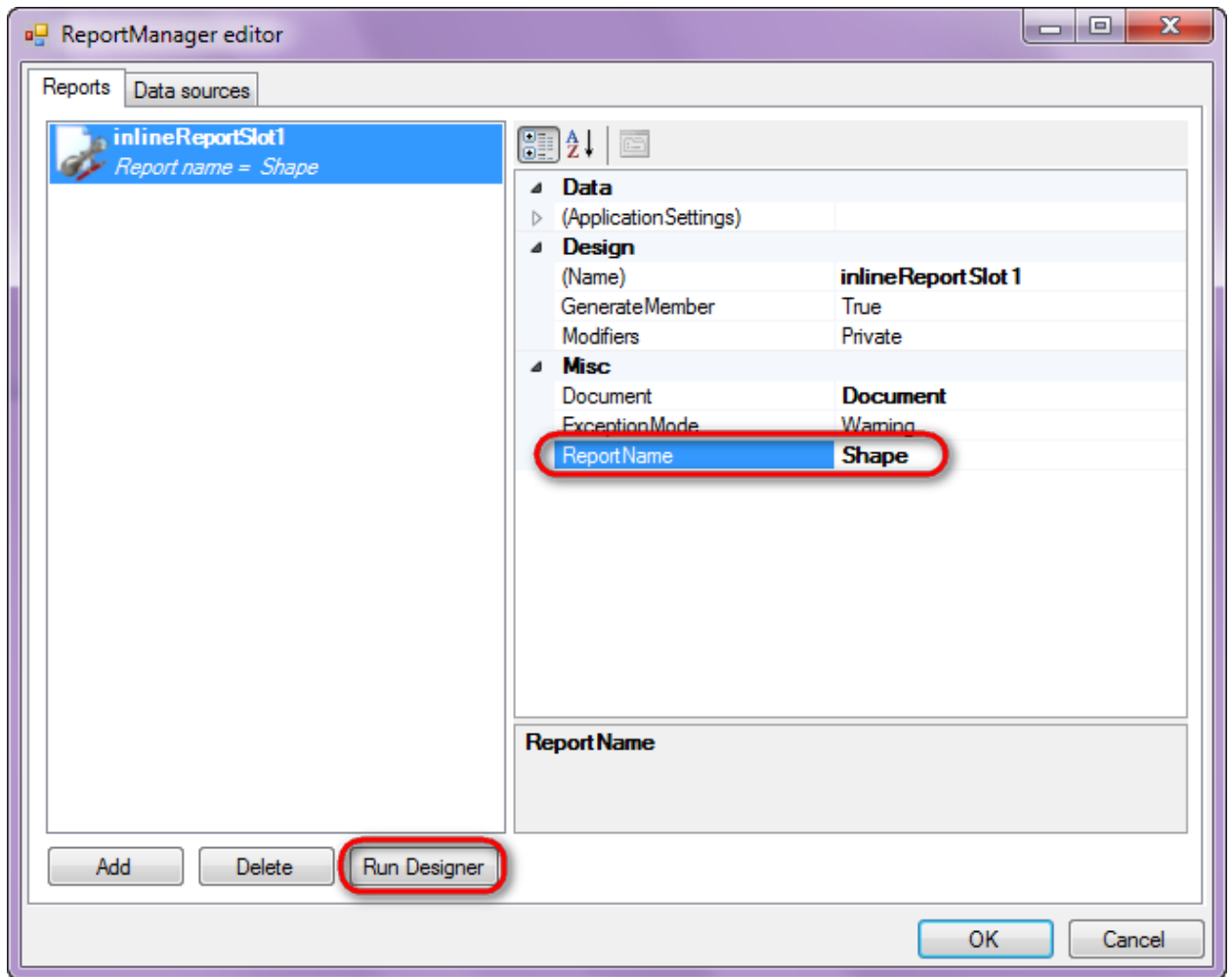
Step 5

Double click on ReportManager to open ReportManager editor.



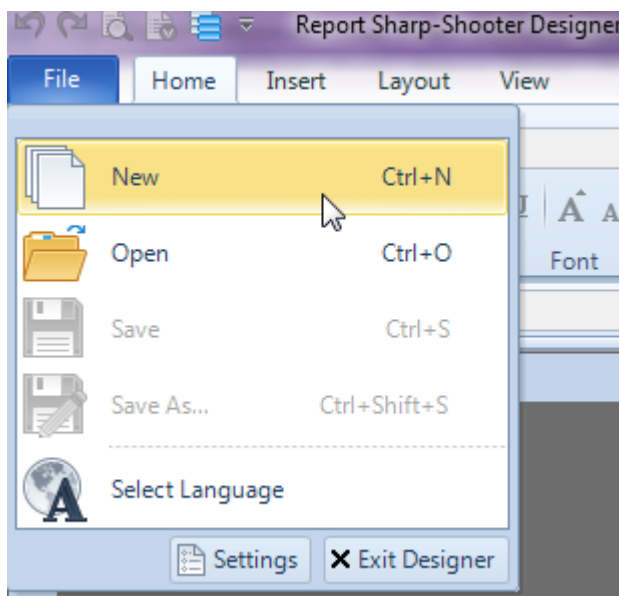
Step 6

Go to "Reports" tab, click "Add" and select "InlineReportSlot". Set name of the report in the property ReportName - "Shape". Click "Run Designer" in order to open template editor - Report Designer.

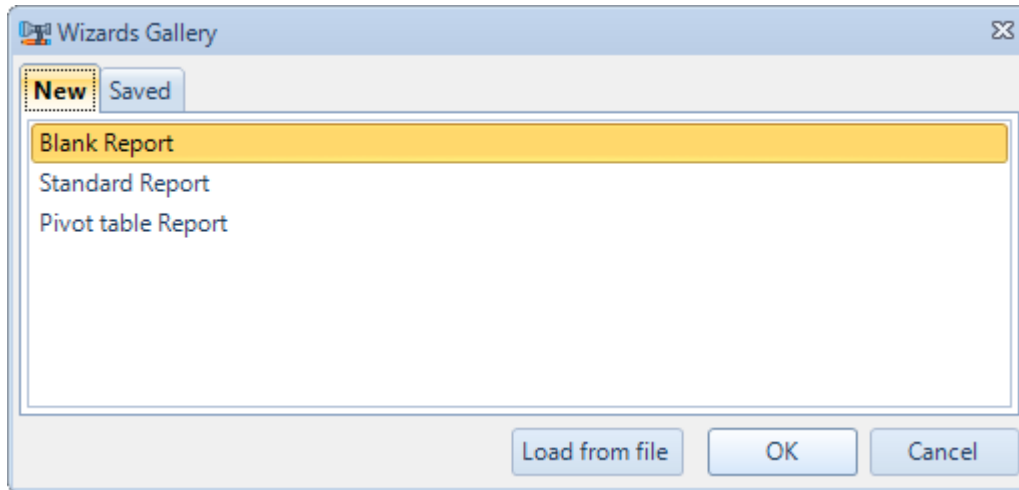


Step 7

Create new empty template – select File\New from the main menu.

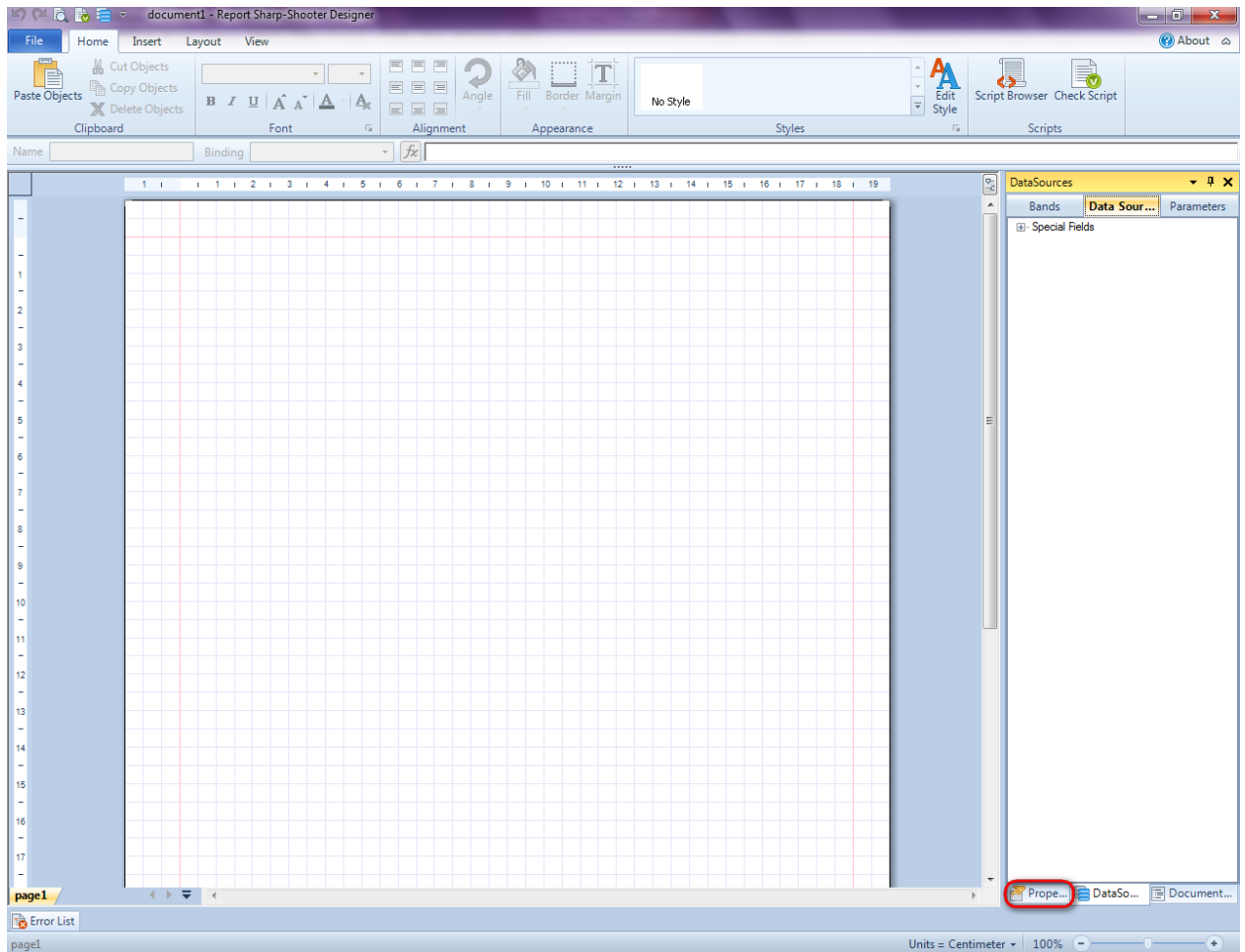


Select "Blank Report" in the Wizards Gallery and click "OK".

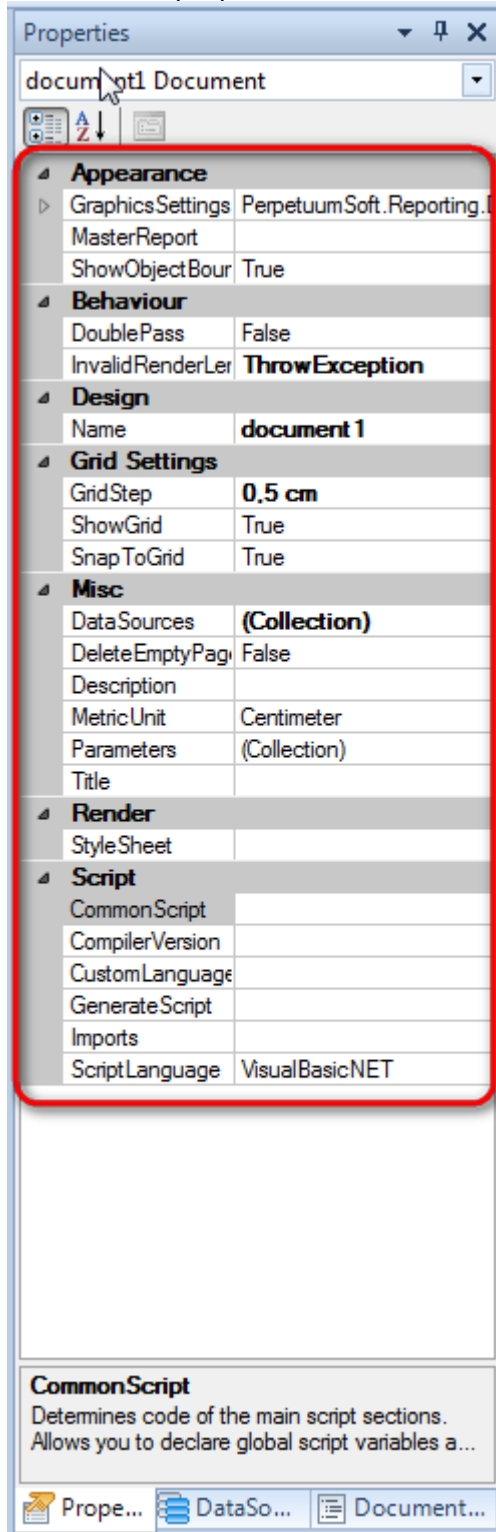


Step 8

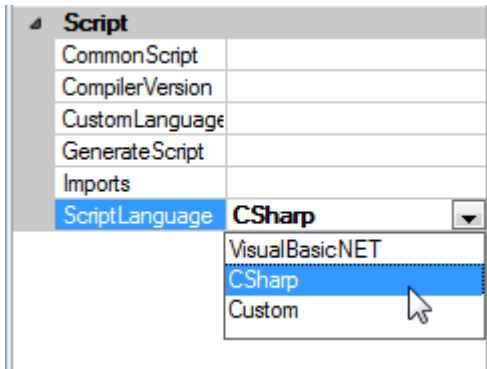
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

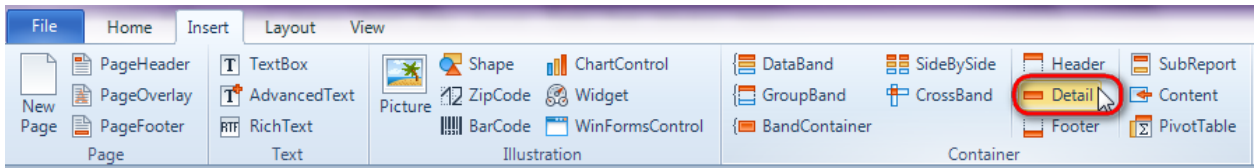


Set property ScriptLanguage = CSharp.

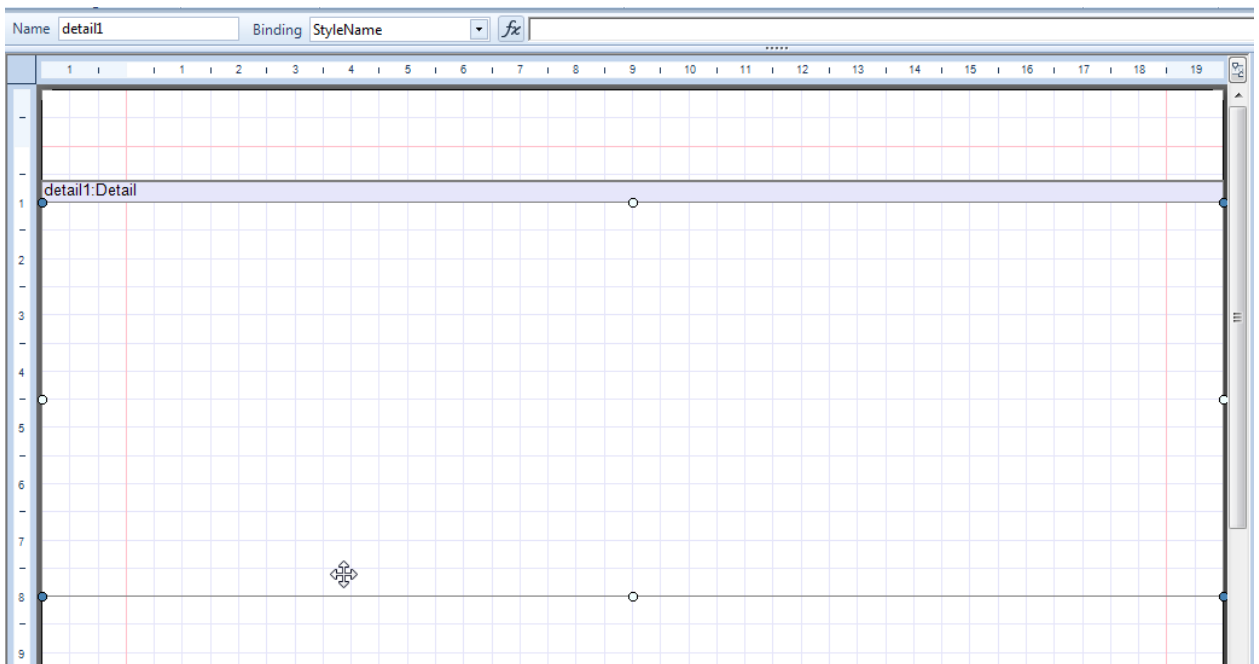


Step 9

Press the "Detail" button in the Insert tab in the group Container.

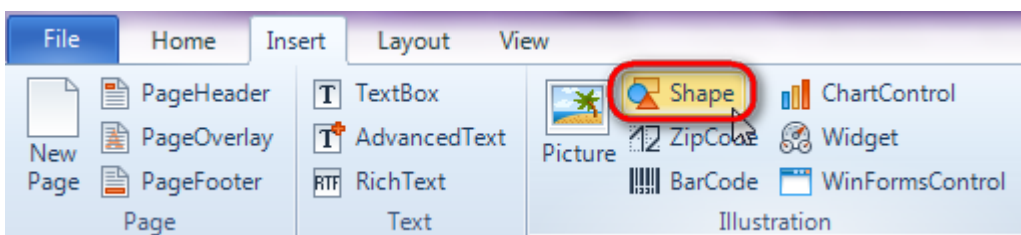



Click on the template area to add band to the template. Change the size of the band by stretching its border.

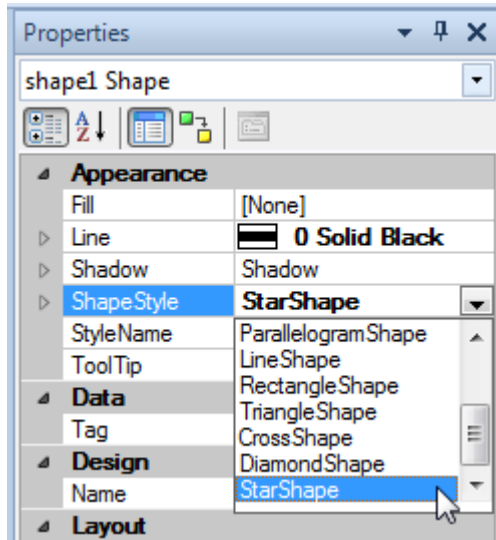


Step 10

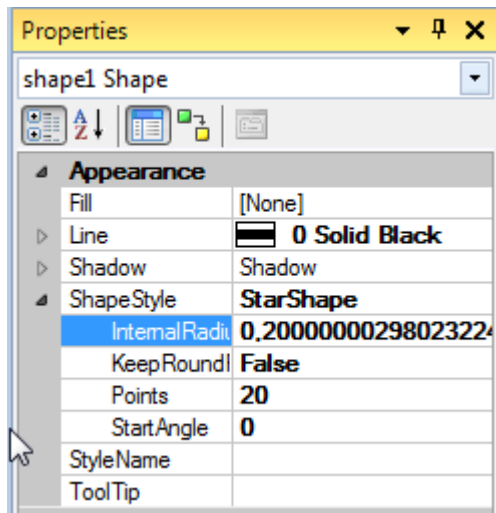
Press the "Shape" button in the Insert tab in the group Illustration.



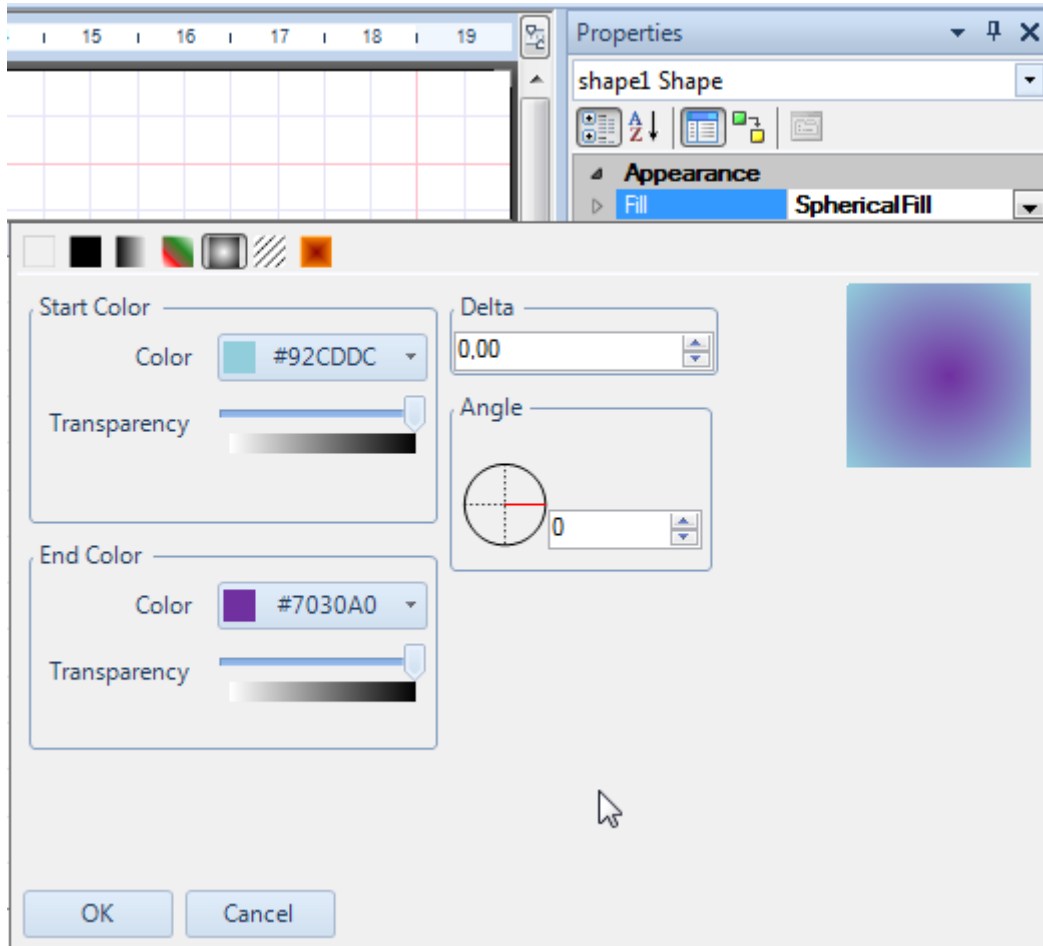
Click on the Detail band to add the Shape element inside Detail. Change the shape size. Select the ShapeStyle property, click the  button, select "StarShape".




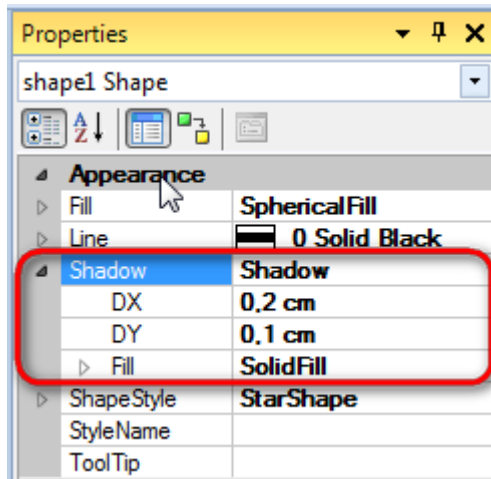
Click the  button to the left from the ShapeStyle property to open additional options. Set Points = 20, InternalRadius = 0,2.




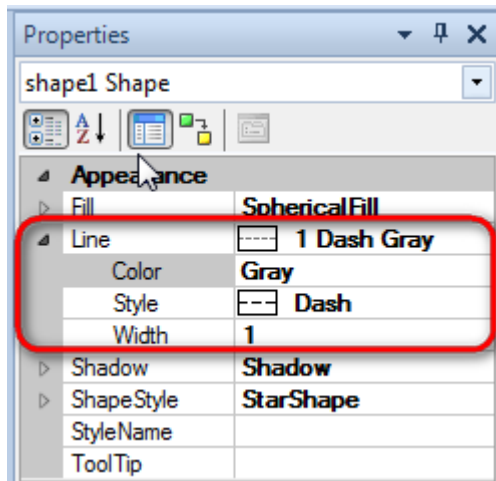
Select the Fill property, click the  button, select "SphericalFill", set two colors.



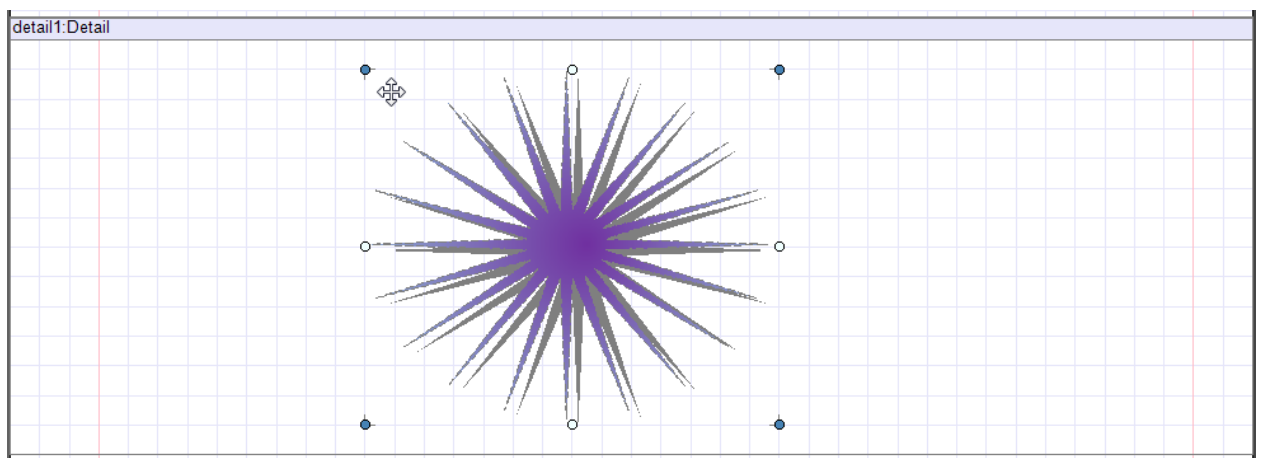
Click the  button to the left from the Shadow property, set DX = 0,2 cm, DY = 0,1 cm, Fill = SolidFill.



Click the  button to the left from the Line property, set Color = Gray, Style = Dash, Width = 1.



The report template should look as follows:

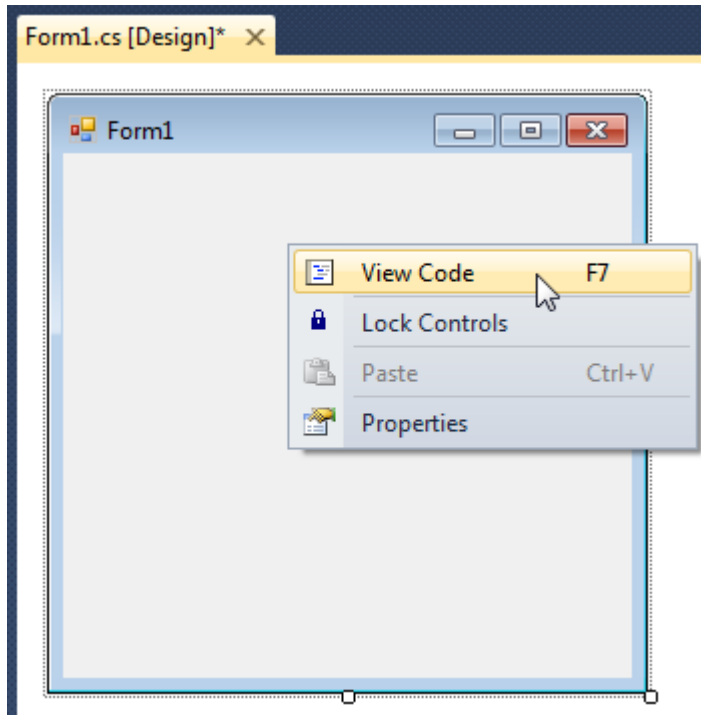


Step 11

Save template, close Report Designer.

Step 12

Right click on the application form and select "View Code" in the context menu to view code.

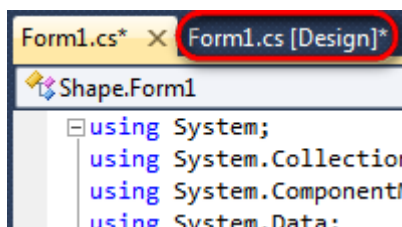


Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

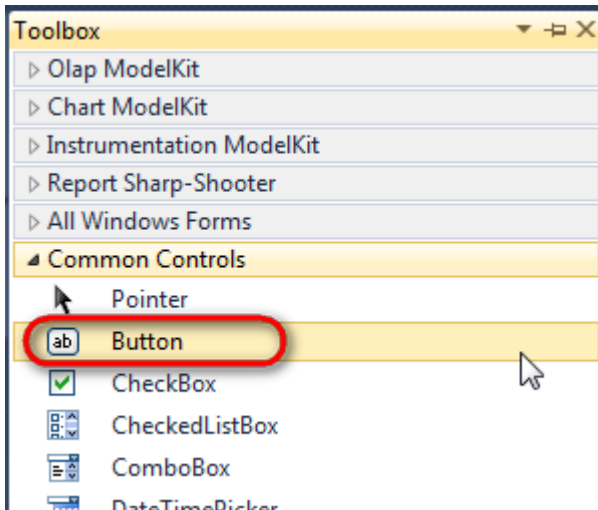
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 13

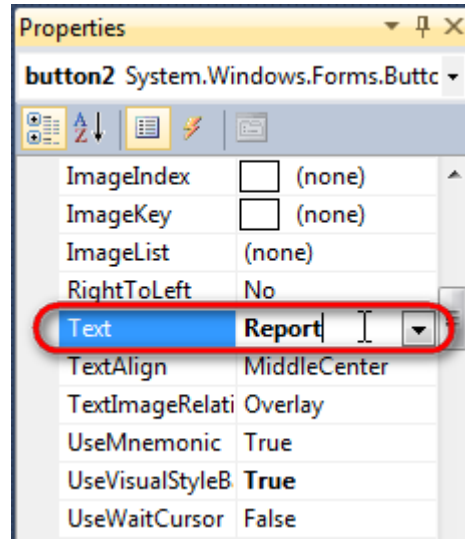
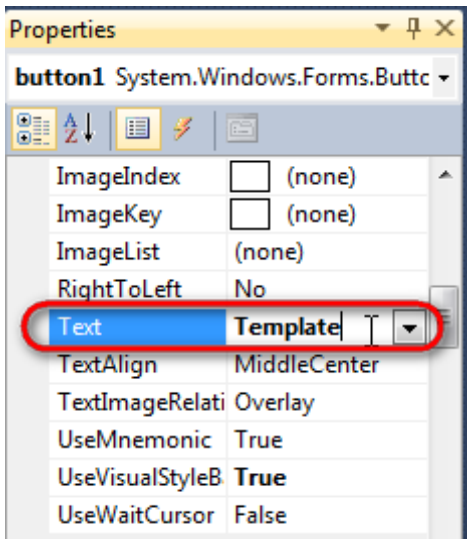
Get back to the application form by clicking "Form1.cs [Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



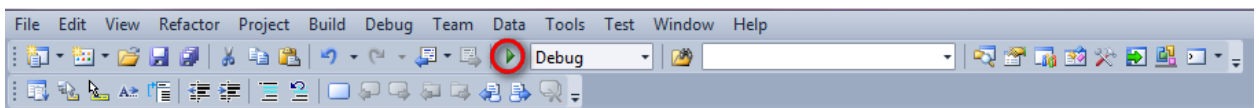
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

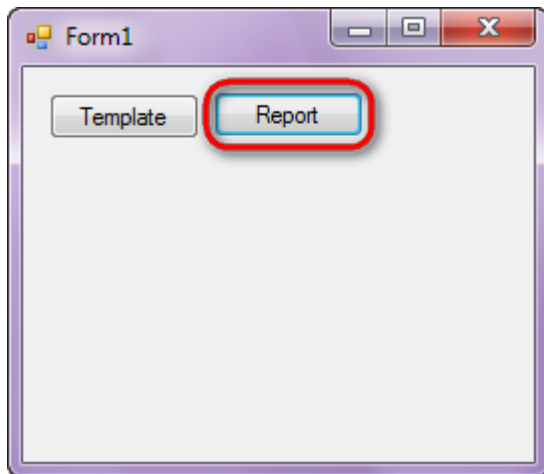
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 14

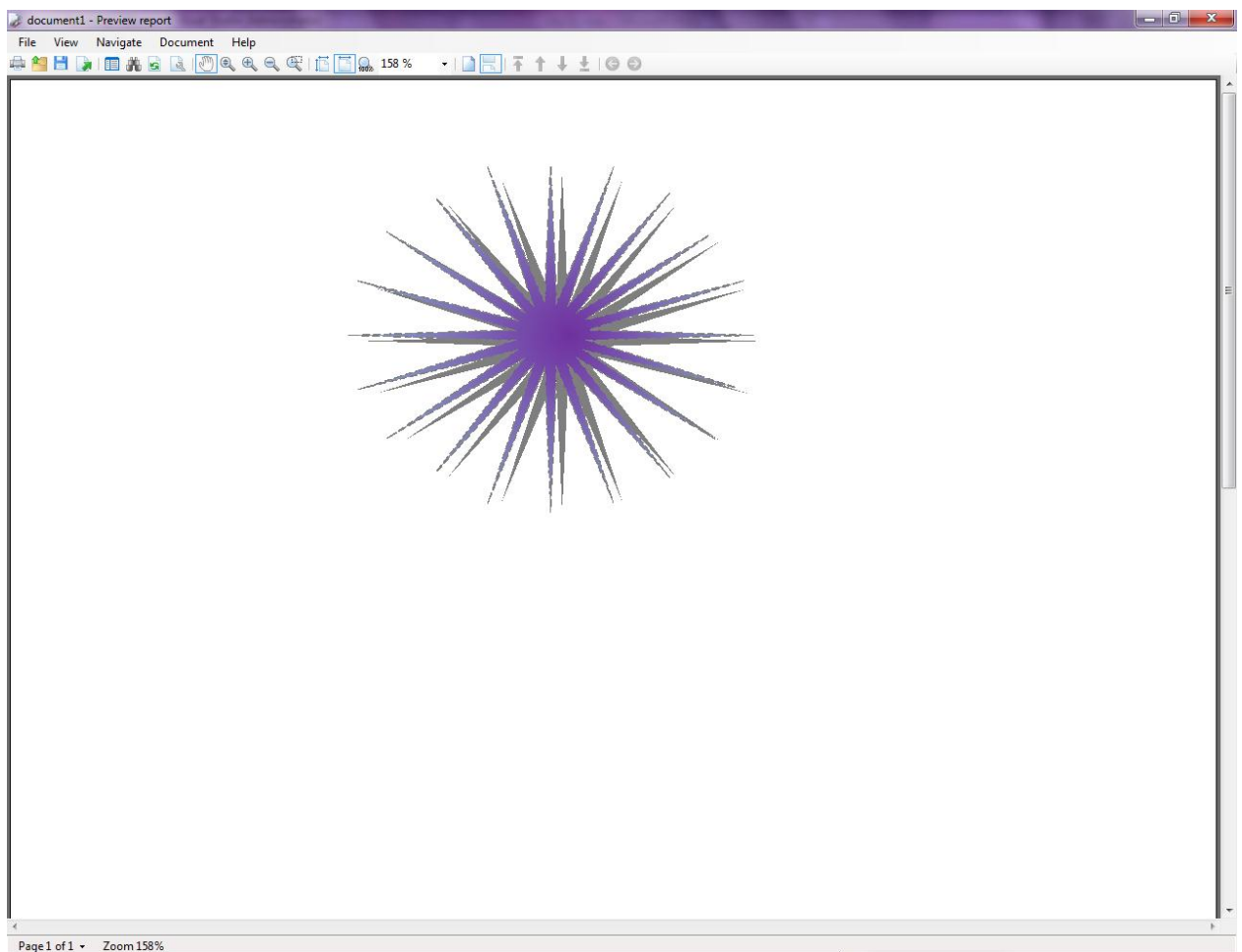
Click “Start Debugging” on the Visual Studio toolbar in order to run application.



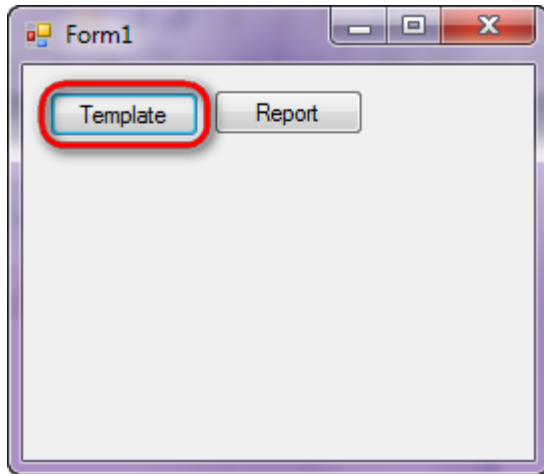
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Report Controls\Basic\Shapes.

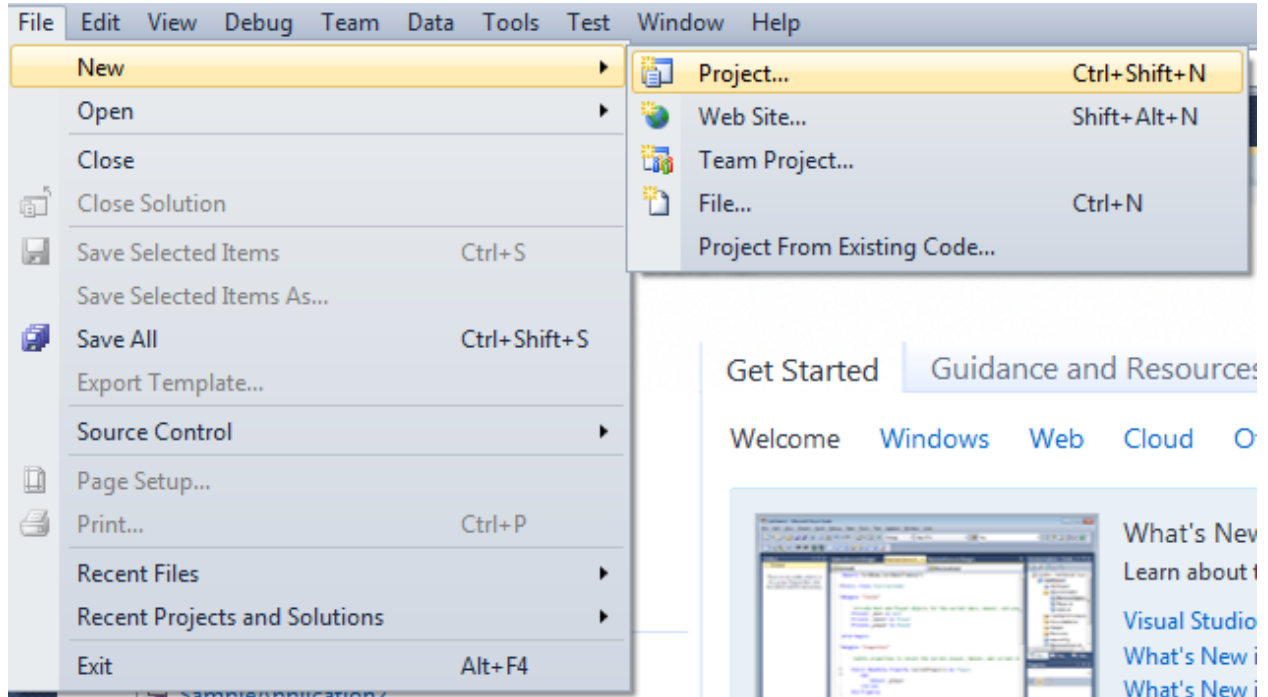


Styles

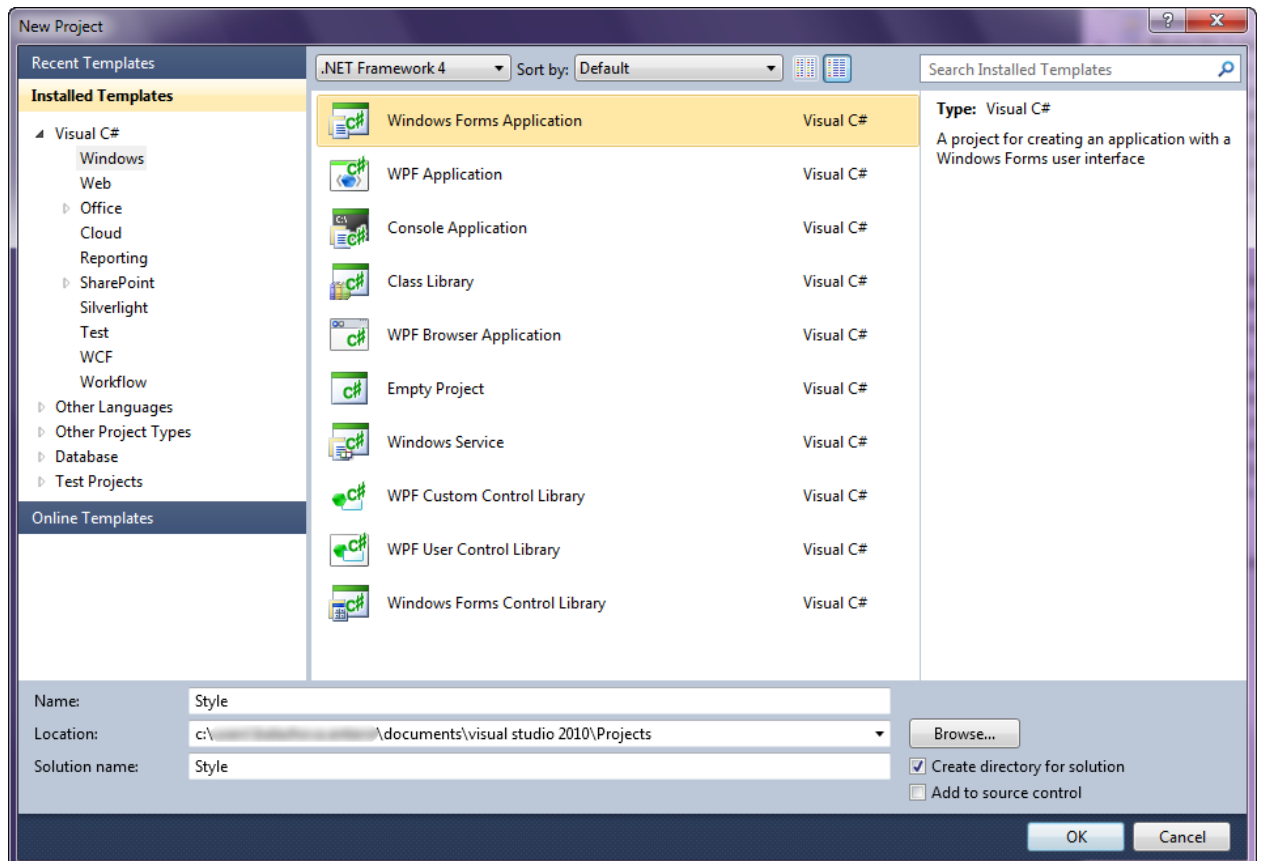
Template of a report containing a list of customers with address, contact person and phone. Different styles are applied to even and odd rows to make easier to view the report.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

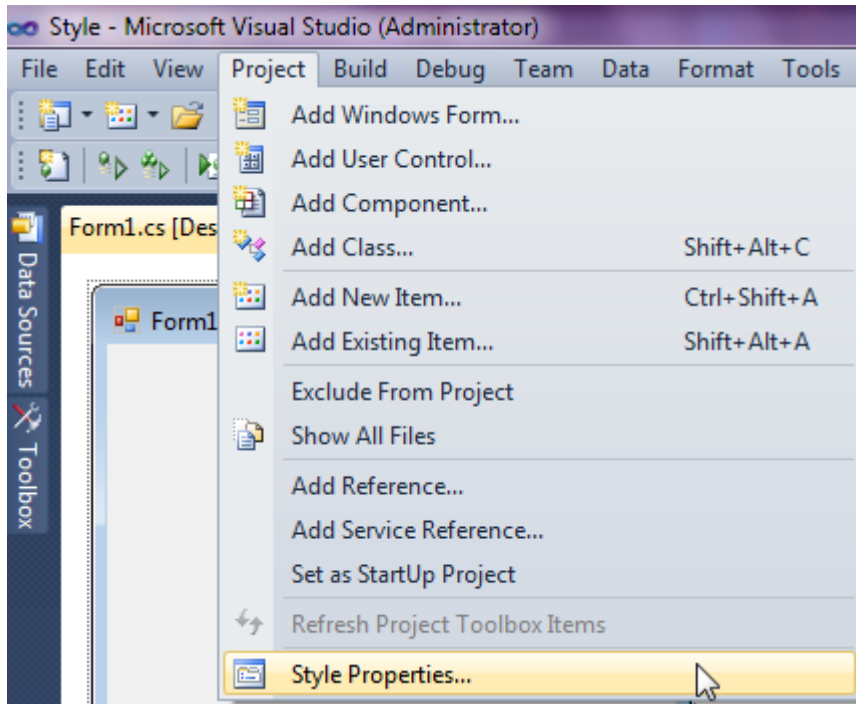


Select Windows Forms Application, set project name – "Style", set directory to save the project to.

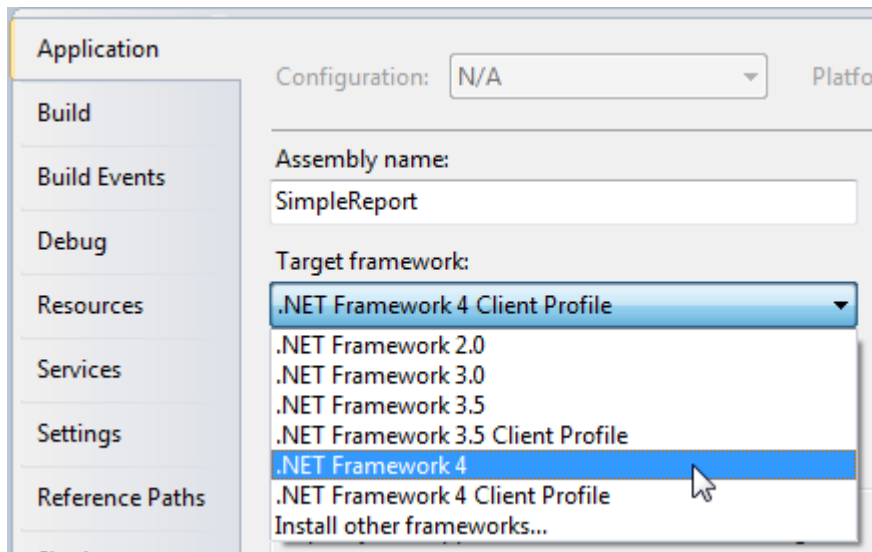


Step 2

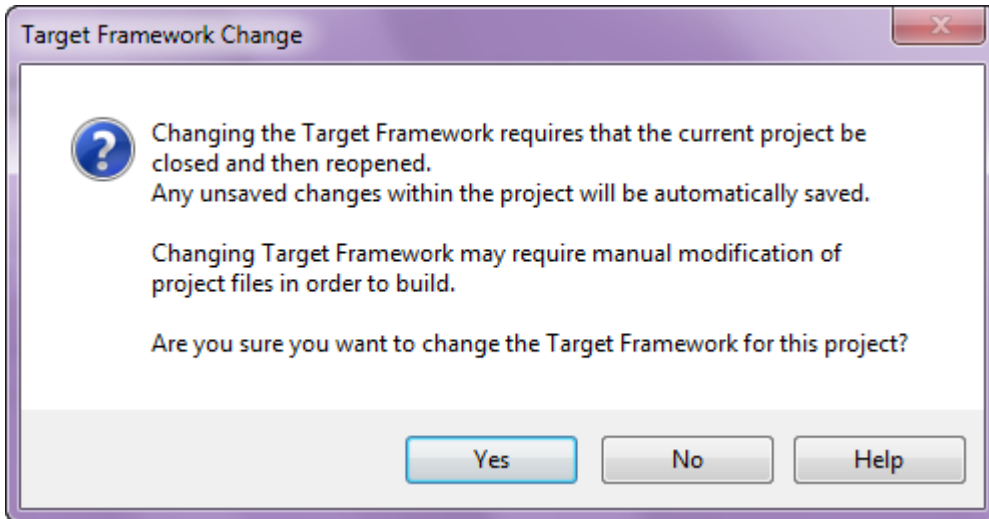
Change the project properties. Select the Project\Style Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

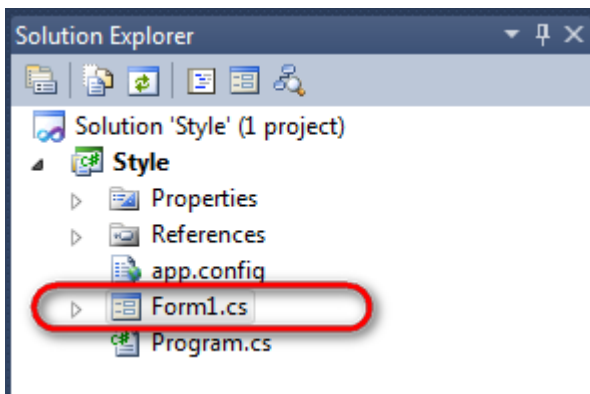


Press the "Yes" button in the opened window.

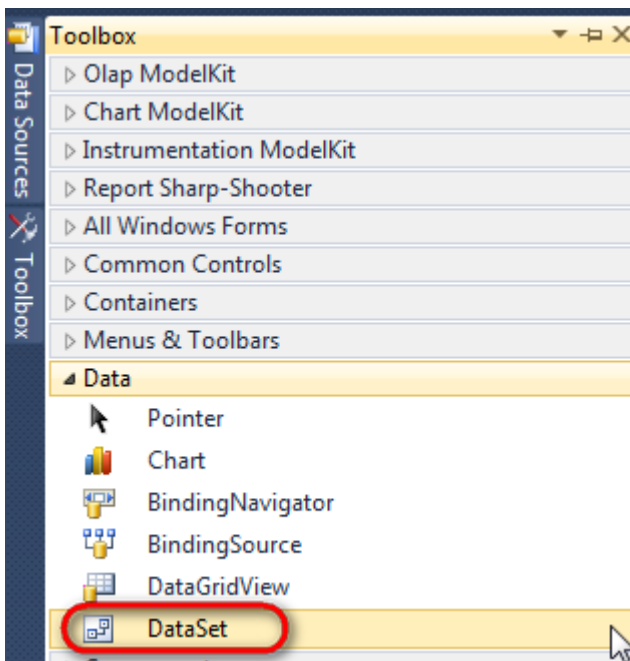


Step 3

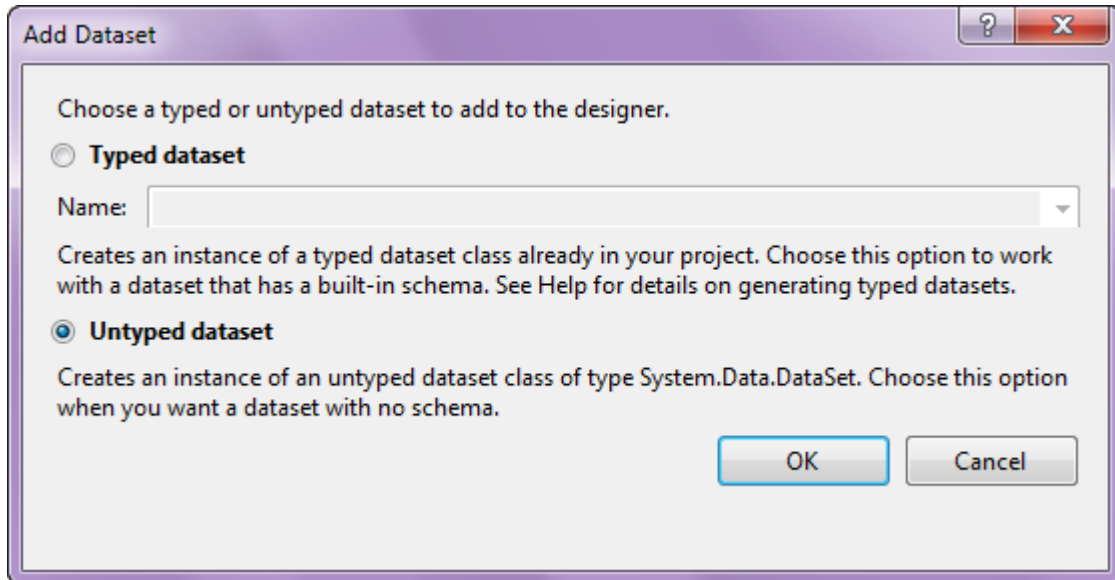
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



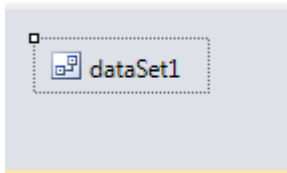
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

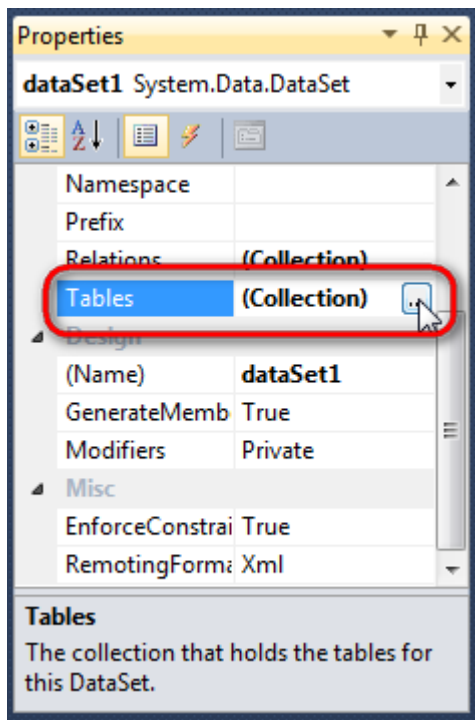


The component is available in the lower part of the window.

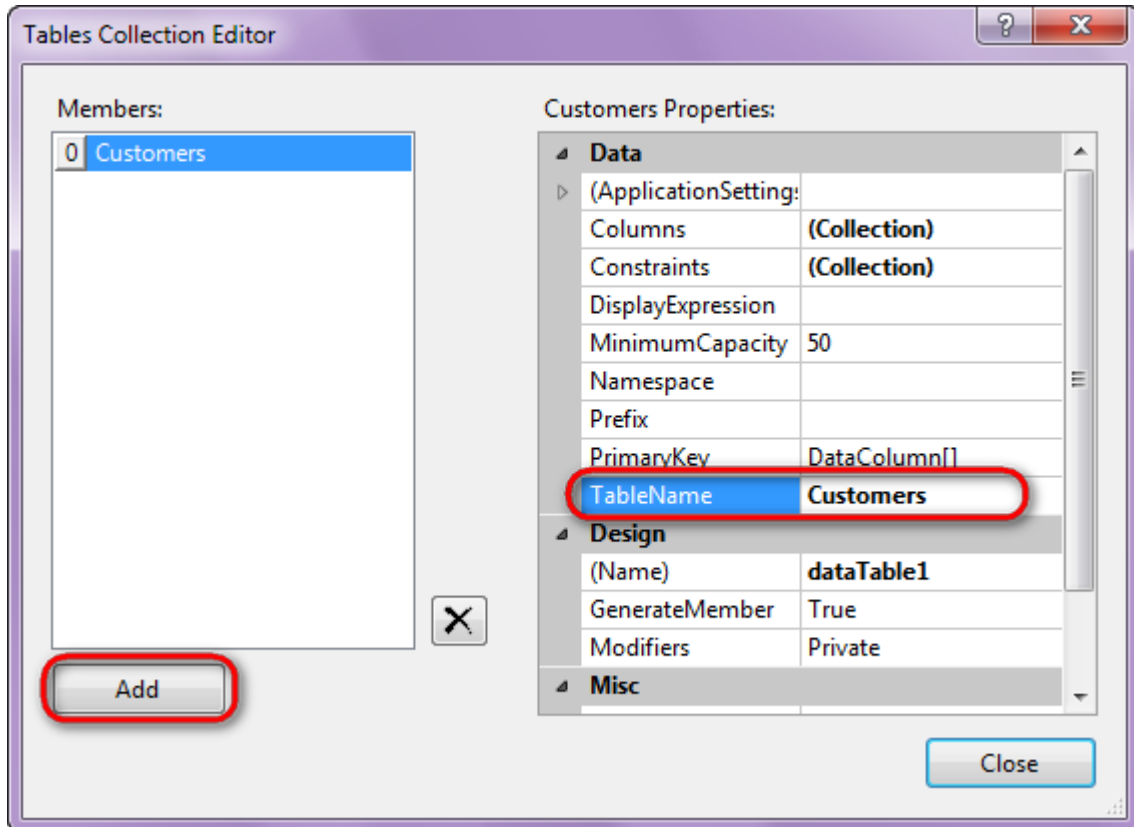


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

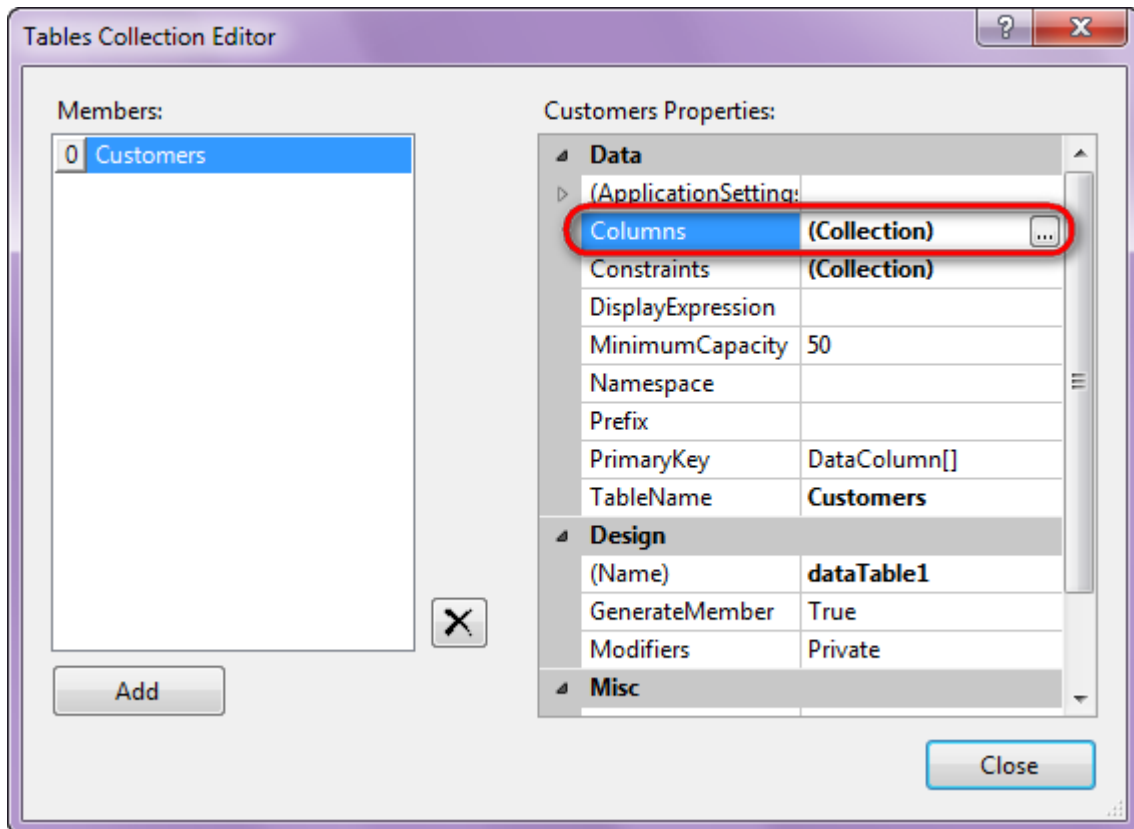


Click "Add" in order to add table. Set property TableName = Customers.

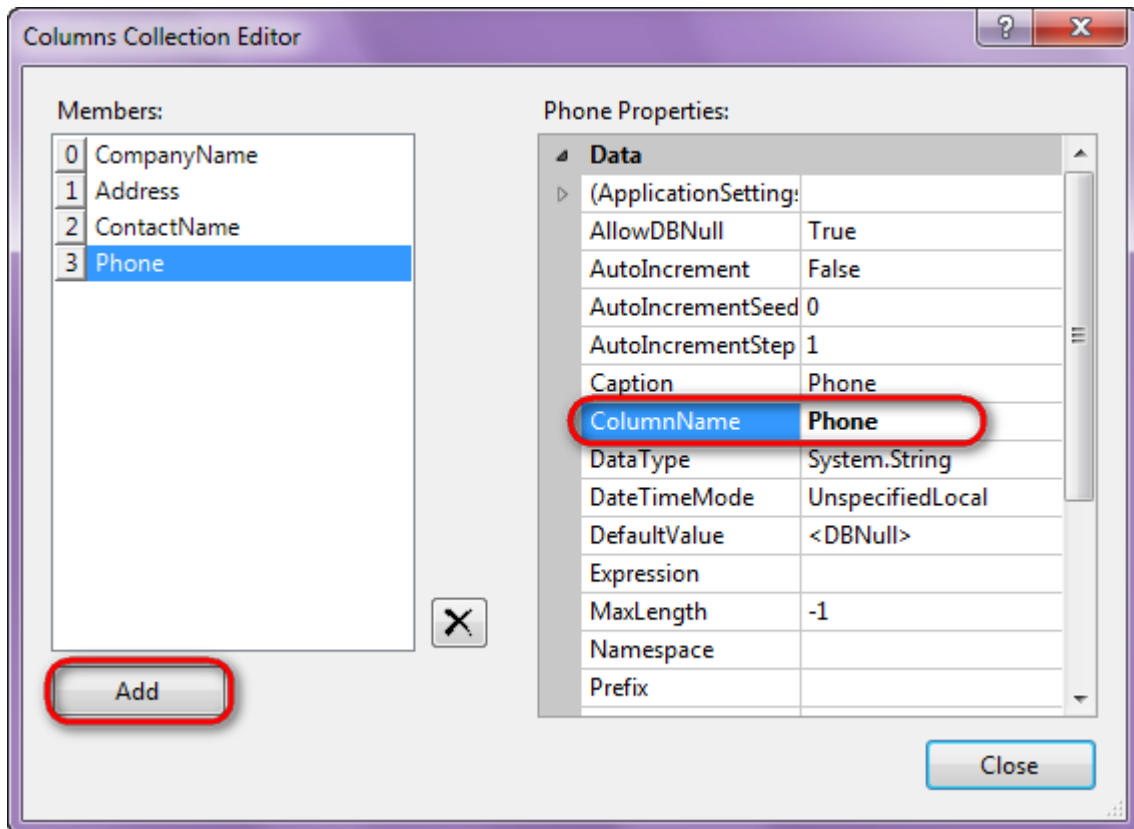


Step 5

Select Columns property, click button  in order to open property editor.

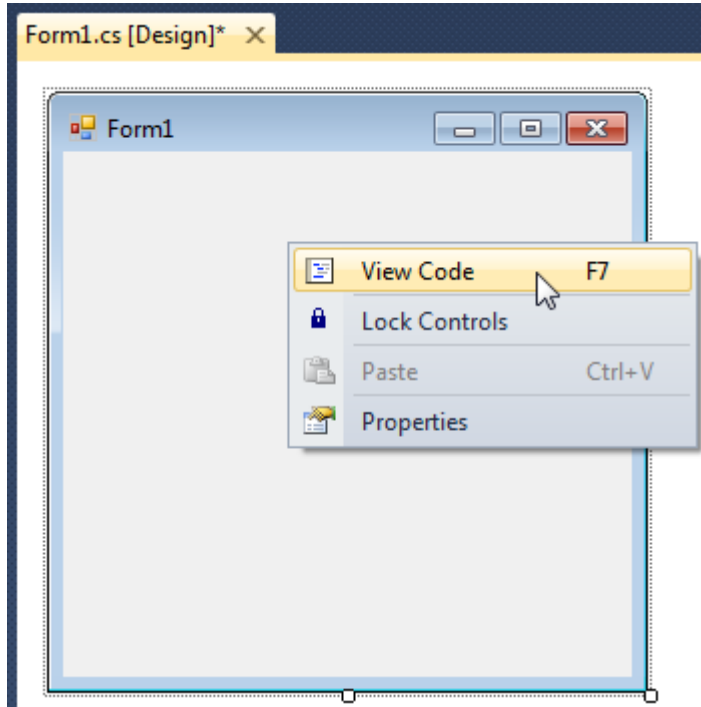


Click "Add" to add a new column. Add four columns. Set ColumnName property to "CompanyName", "Address", "ContactName", "Phone" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu to view code.



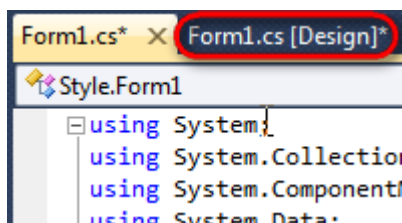
Add the following code to the class constructor in order to fill data source.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
}
```

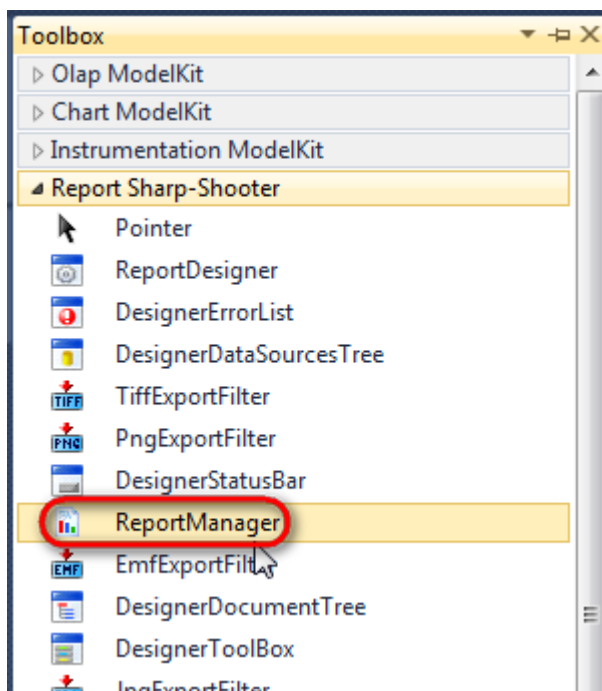
```
row["Address"] = "Obere Str. 57";  
row["ContactName"] = "Maria Anders";  
row["Phone"] = "030-0074321";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["Address"] = "Avda. de la Constitución 2222";  
row["ContactName"] = "Ana Trujillo";  
row["Phone"] = "(5) 555-4729";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["Address"] = "Kirchgasse 6";  
row["ContactName"] = "Roland Mendel";  
row["Phone"] = "7675-3425";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["Address"] = "Luisenstr. 48";  
row["ContactName"] = "Karin Josephs";  
row["Phone"] = "0251-031259";  
dataTable1.Rows.Add(row);  
}
```

Step 7

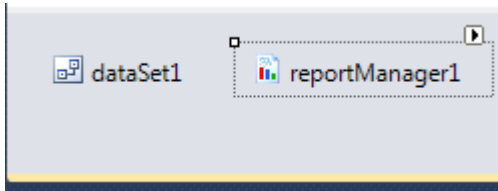
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

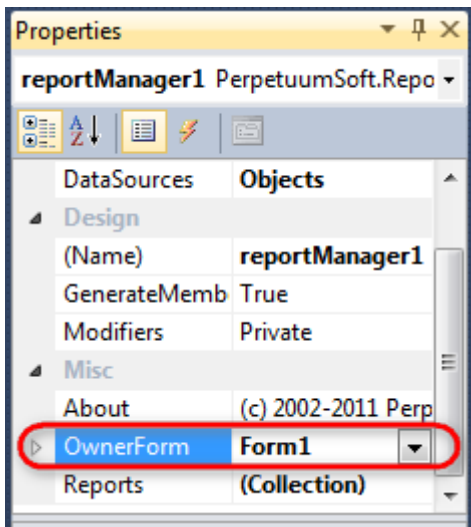


The component is available in the lower part of the window.



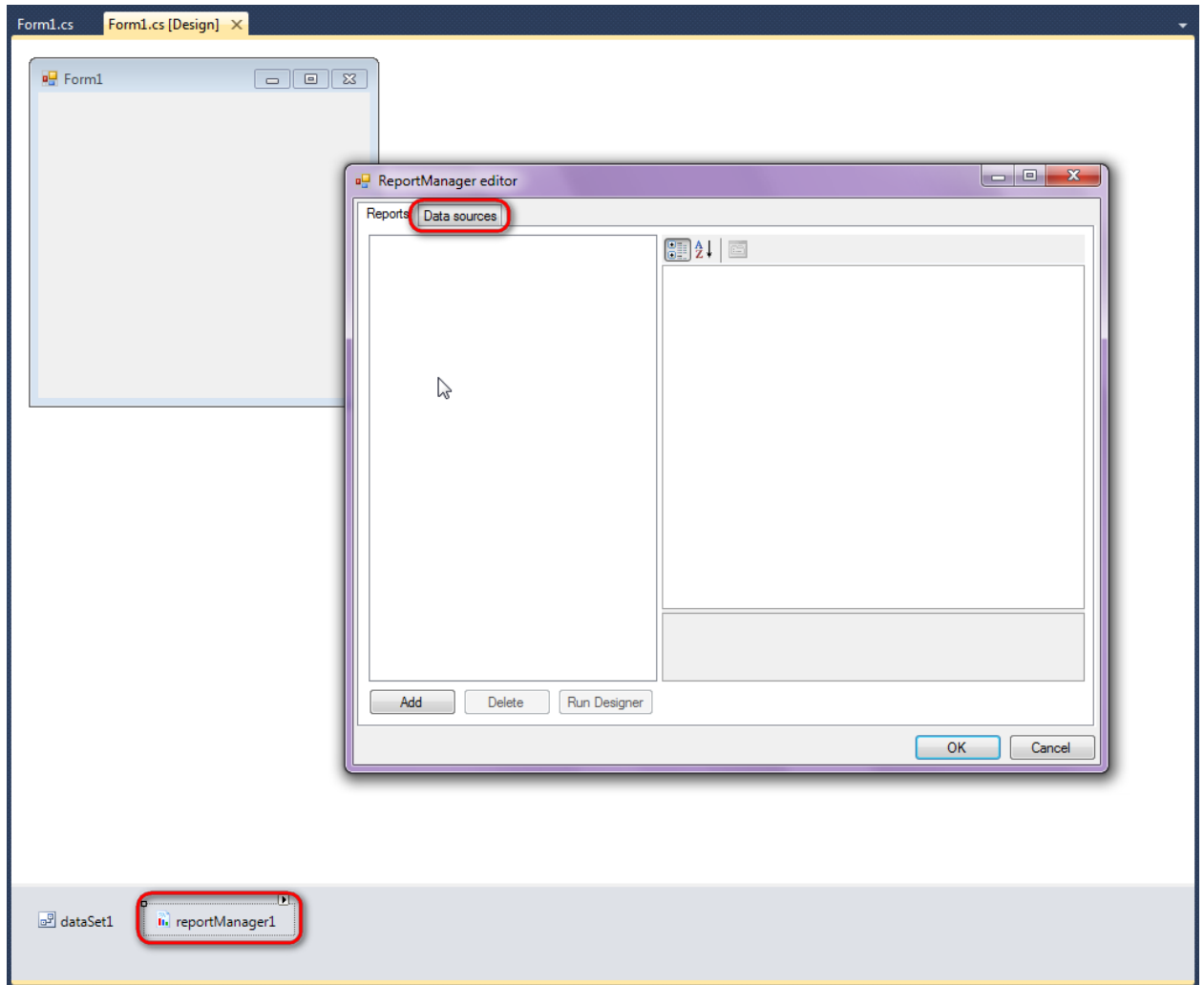
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

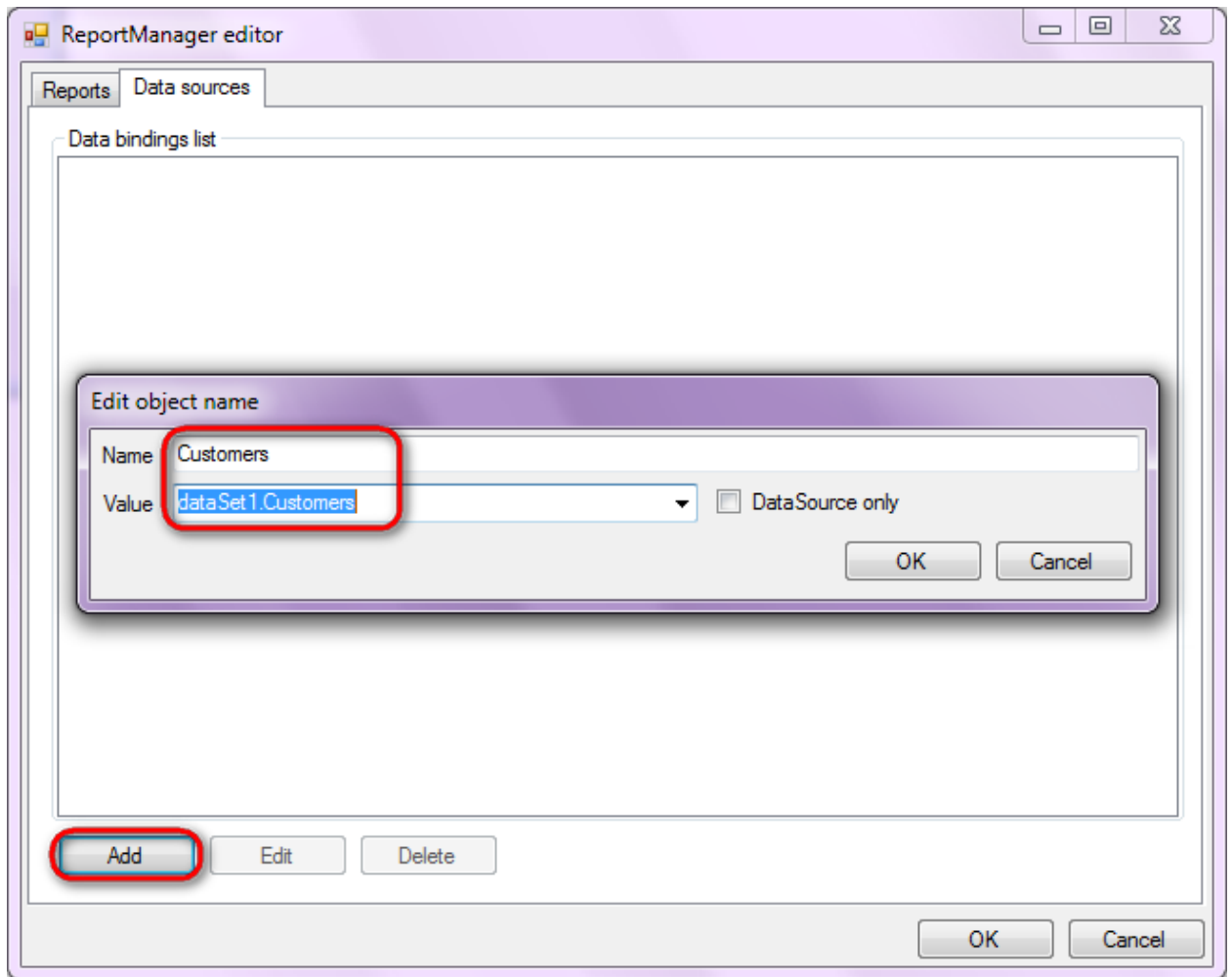


Step 9

Double click on ReportManager to open ReportManager editor.

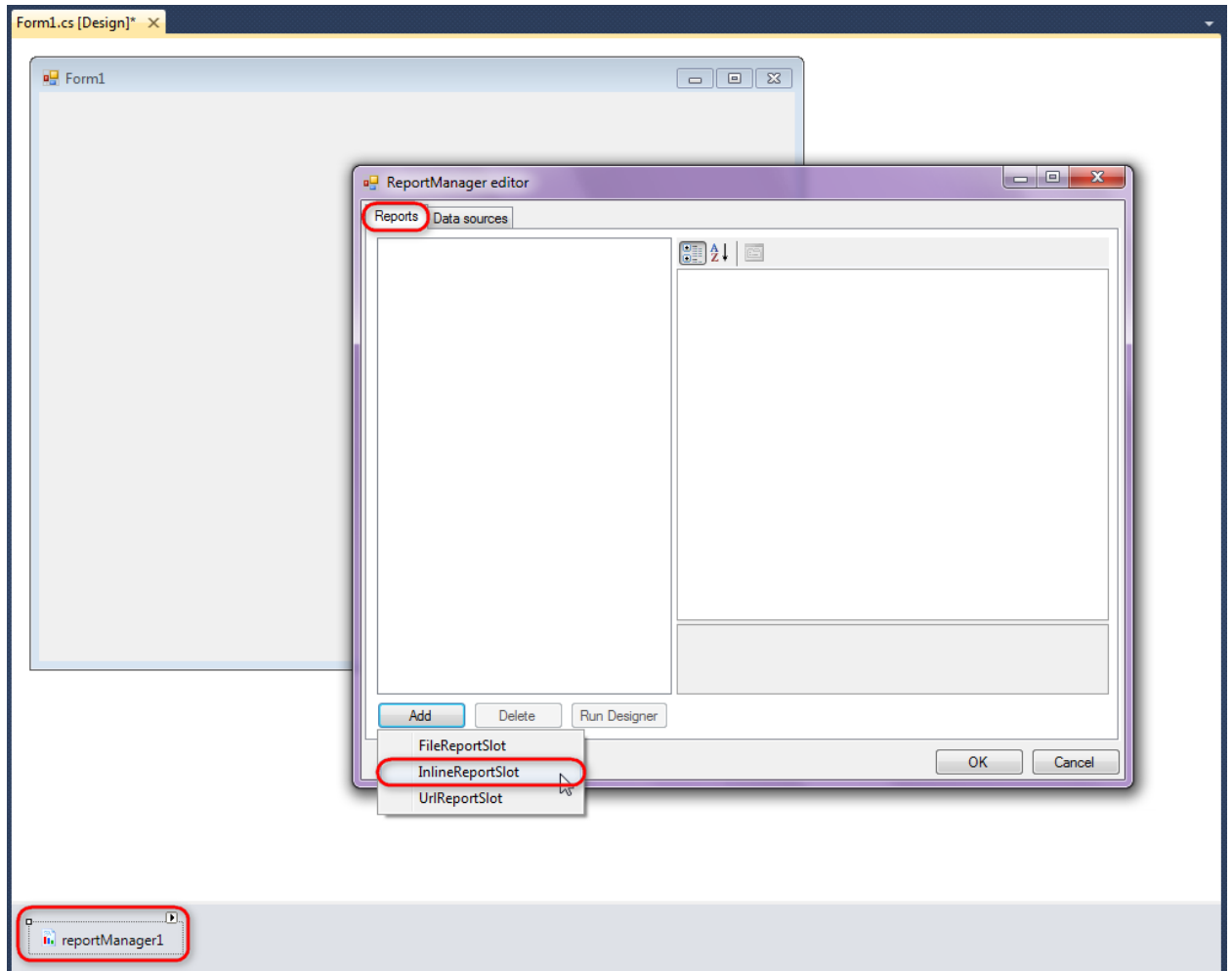


Go to "Data sources" tab, click "Add", set data source name – "Customers", select data source value – "dataSet1.Customers".



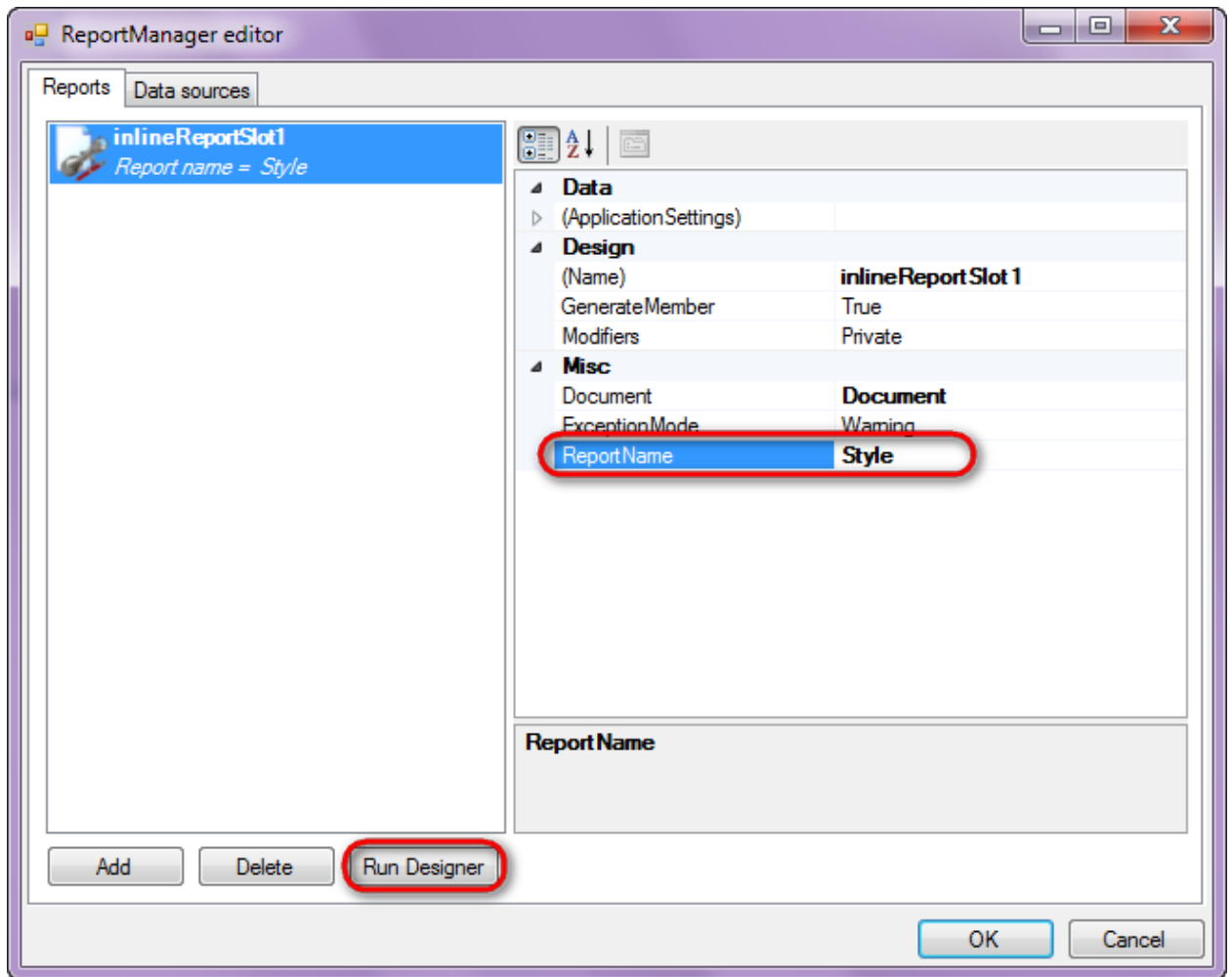
Step 10

Go to "Reports" tab, click "Add" and select "InlineReportSlot".



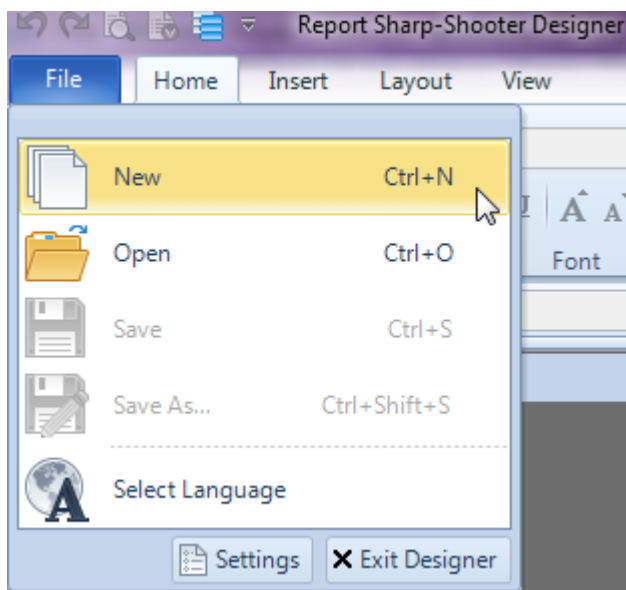
Step 11

Set name of the report in the property ReportName - "List". Click "Run Designer" in order to open template editor - Report Designer.

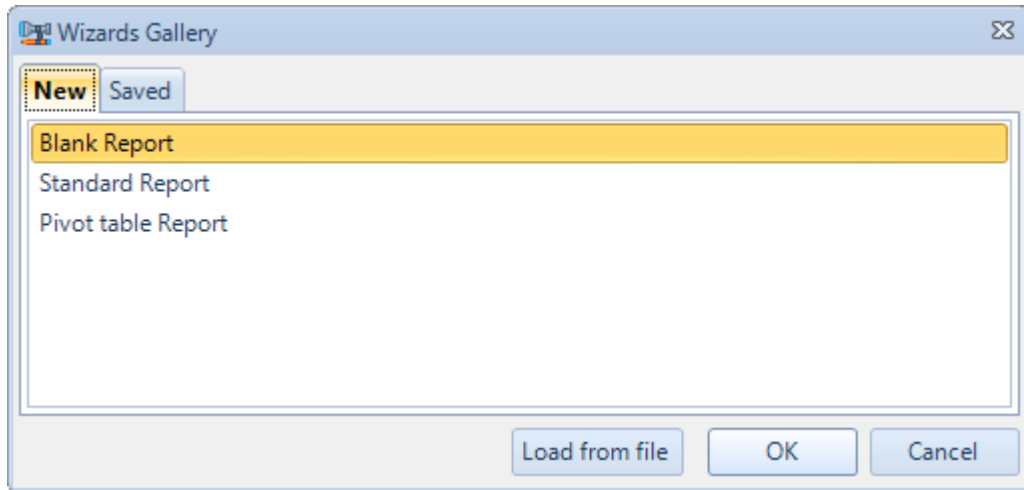


Step 12

Create new empty template – select File\New from the main menu.

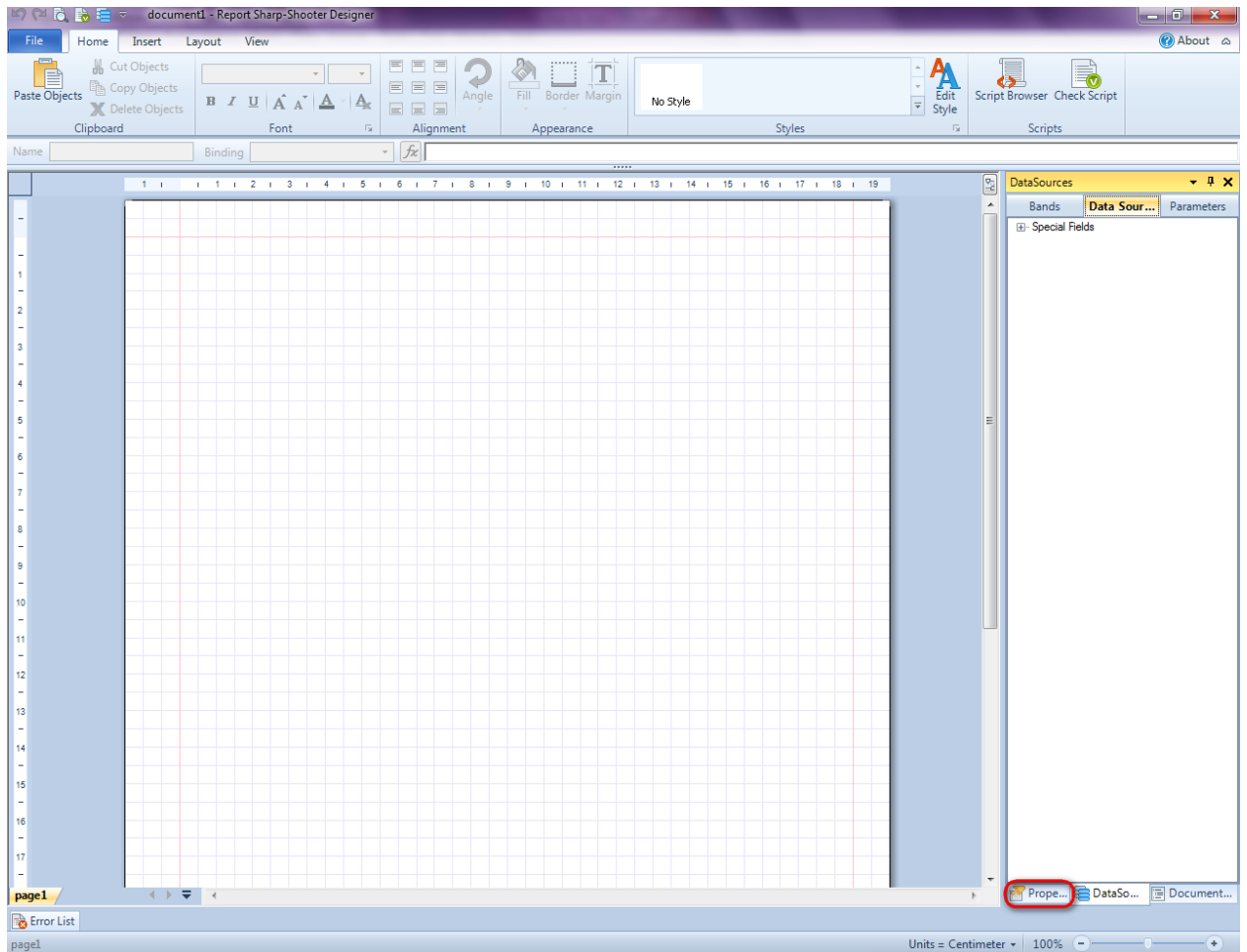


Select "Blank Report" in the Wizards Gallery and click "OK".

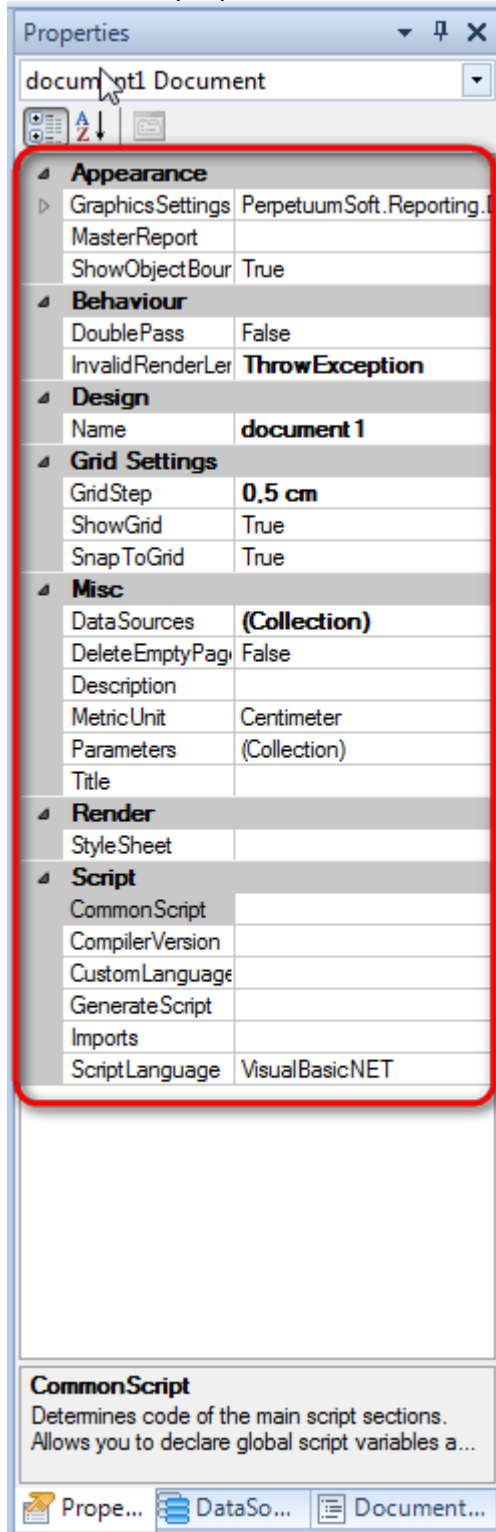


Step 13

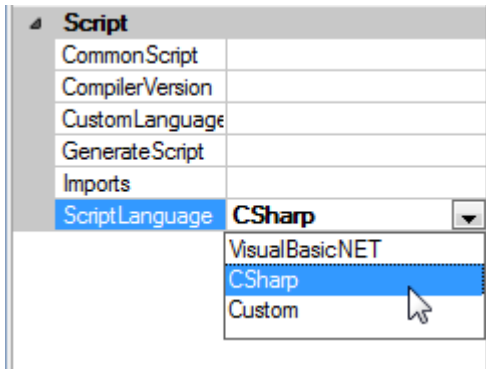
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

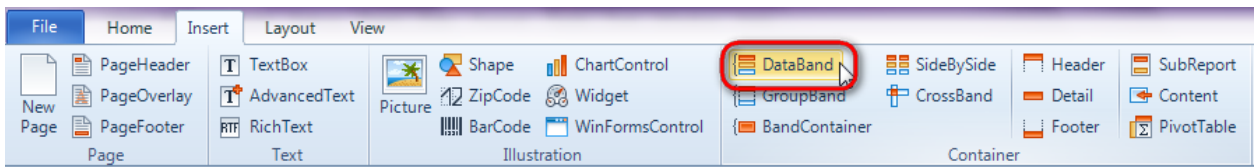


Set property ScriptLanguage = CSharp.



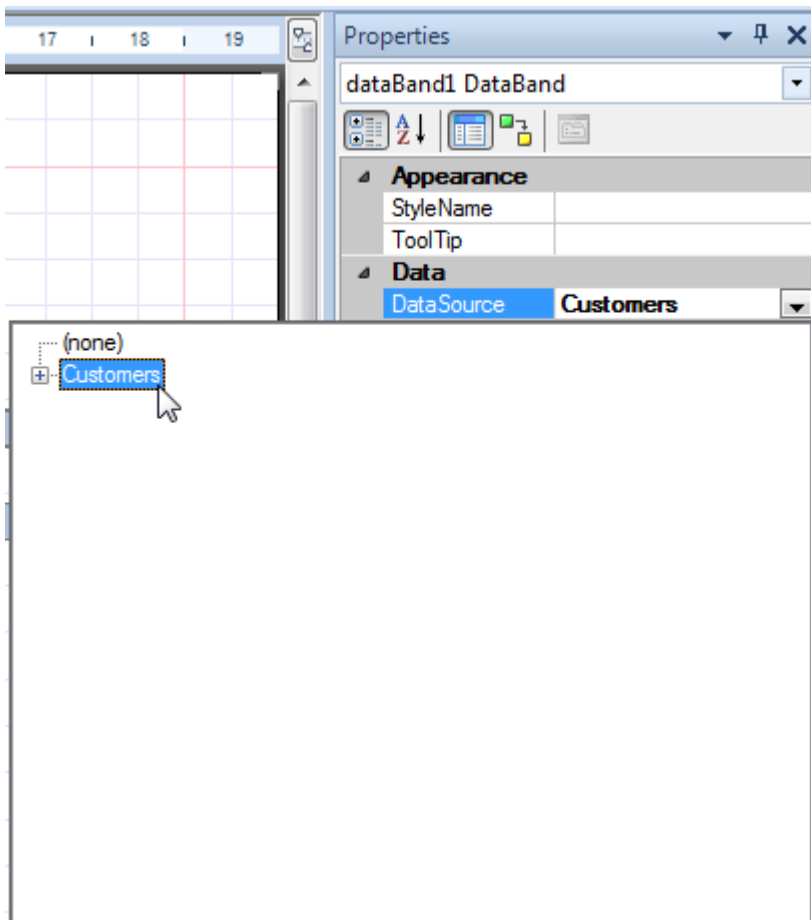
Step 14

Press "DataBand" button on the Insert tab in the group Container.



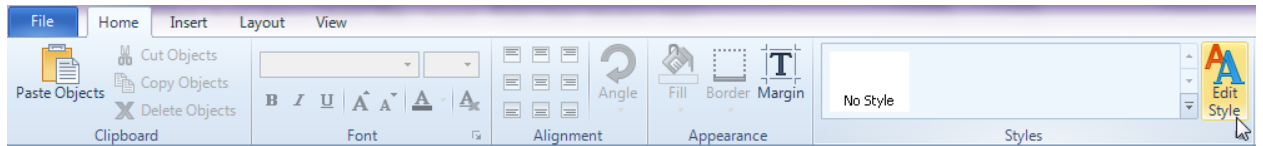
Click on the template area to add DataBand band to the template.


Set data source in the property DataSource = Customers.

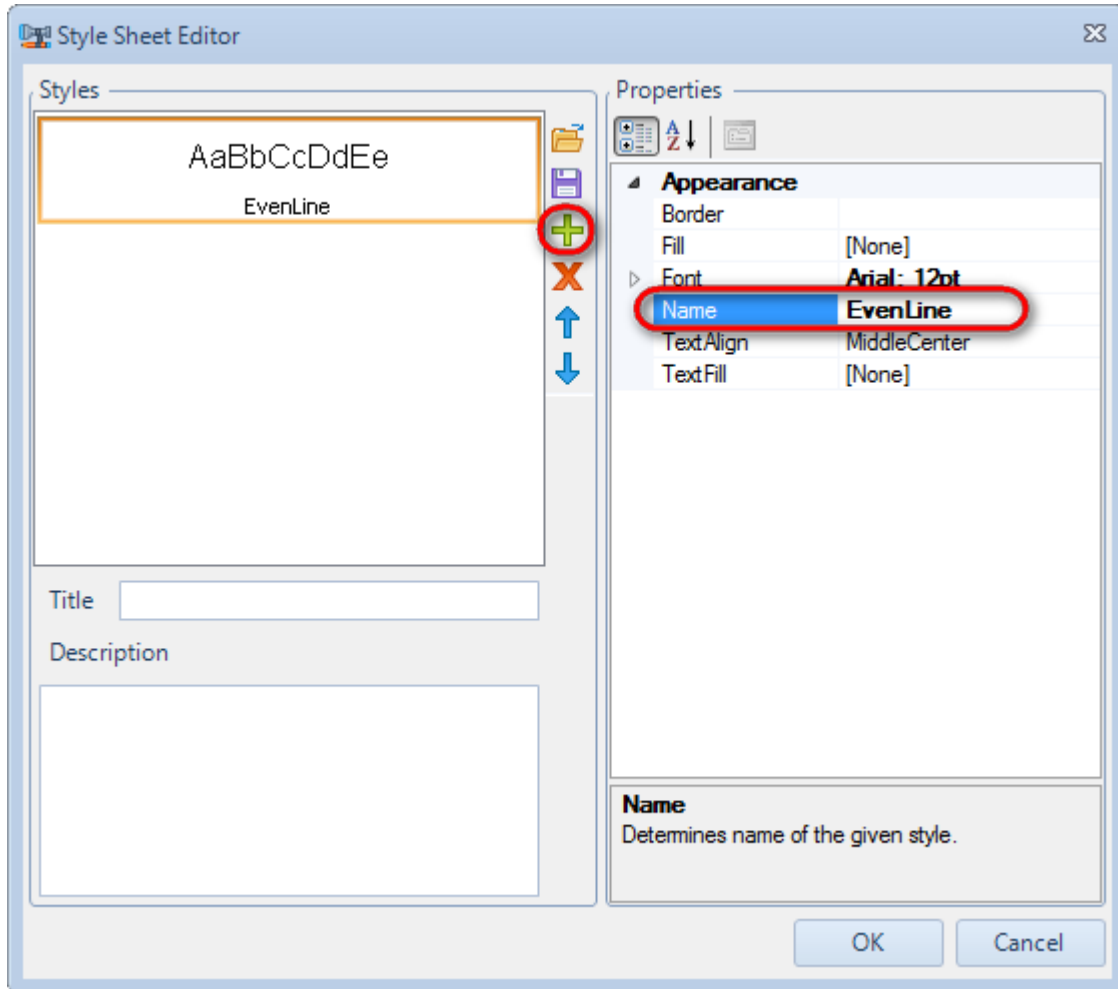



Step 15

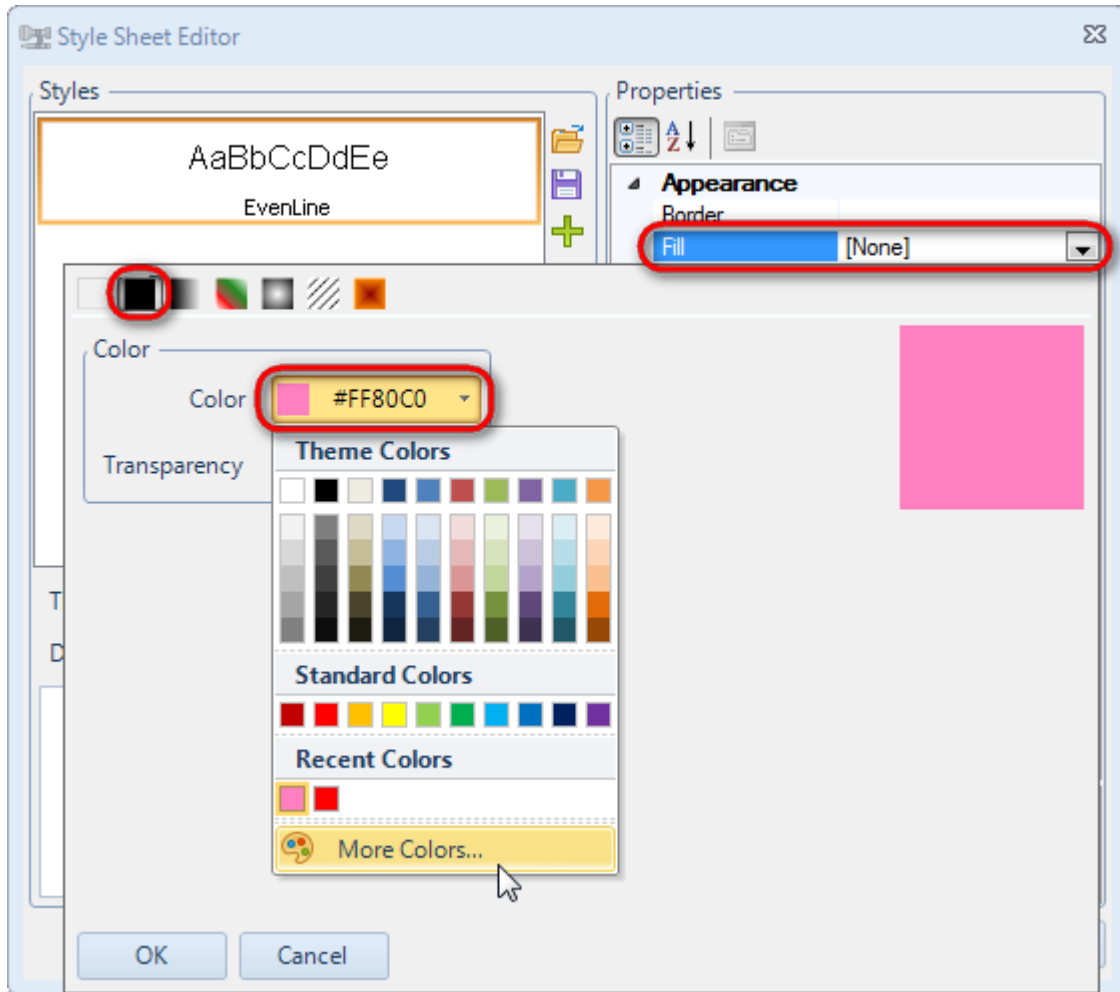
Press the "Edit Style" button in the Home tab in the group Styles.



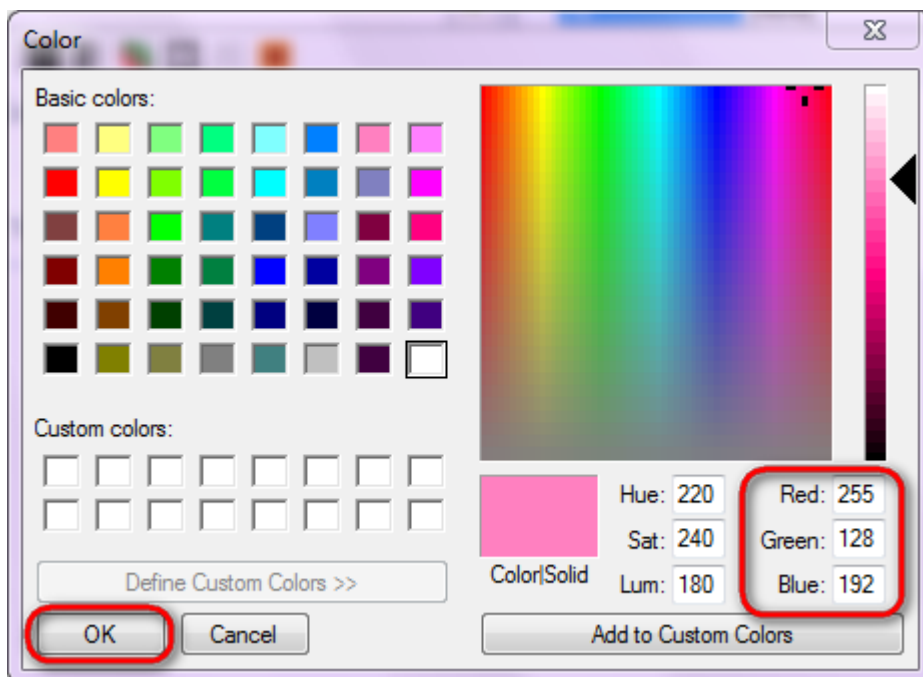
Click the  button to add a new style in the Style Sheet Editor. Set Name = EvenLine.



Select the Fill property, click the  button to open the fill editor. Select SolidFill; press the "Color" button. Select the "More Colors..." item in the appeared combo box.



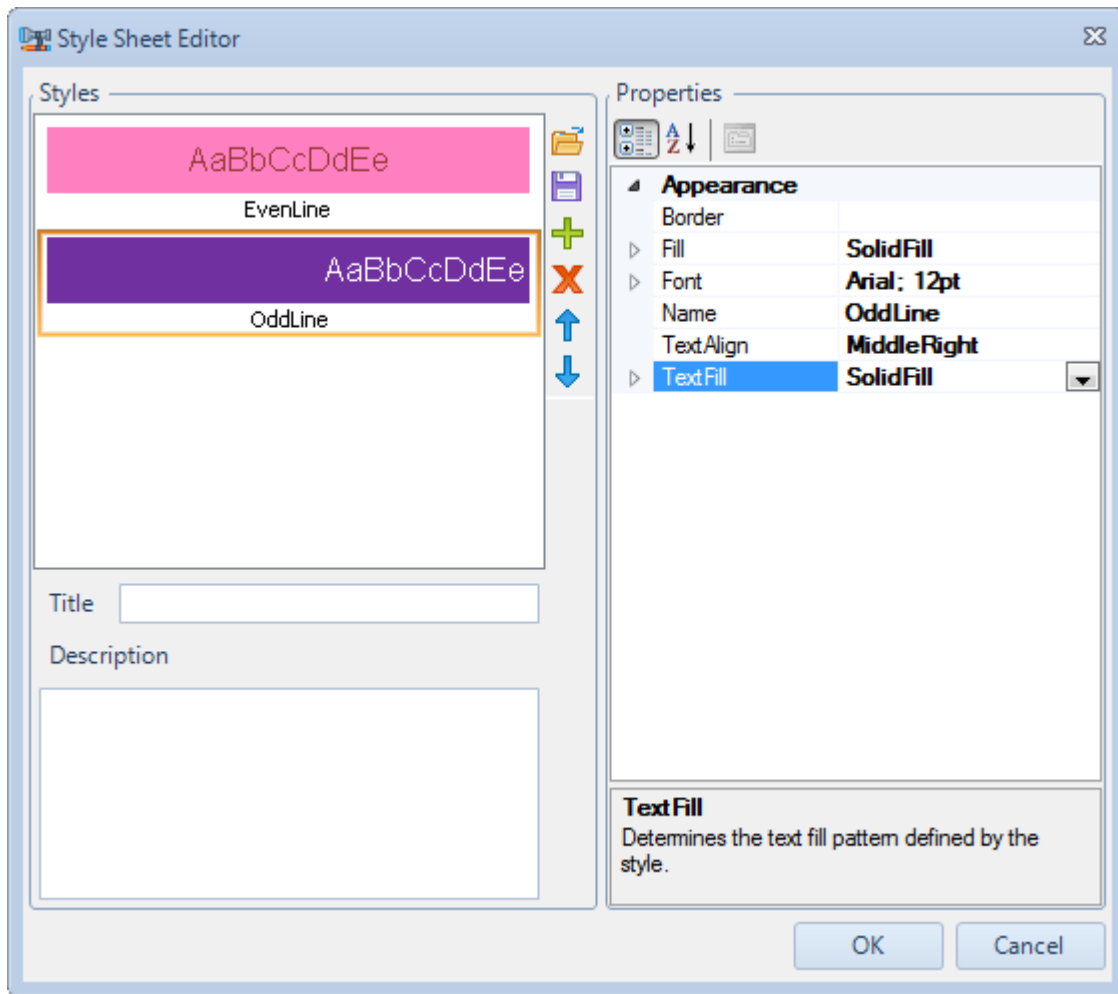
Enter the following values in the Color editor window: Red = 255, Green = 128, Blue = 192. Press "Ok".



Set the TextFill property in the same way.

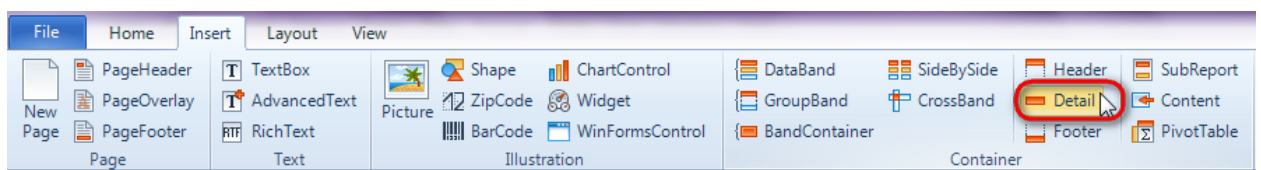


Add one more style. Set Name = OddLine. Set the Fill and TextFill properties different from the EvenLine.

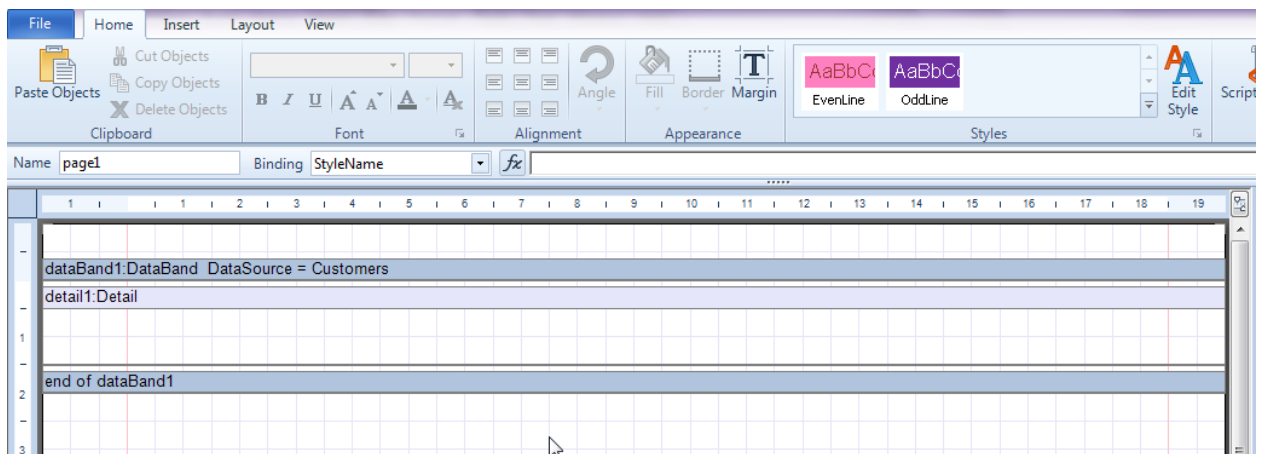


Step 16

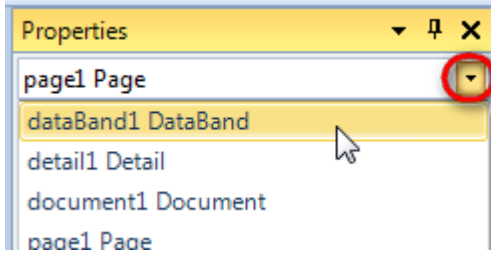
Press the "Detail" button in the Insert tab in the group Container.



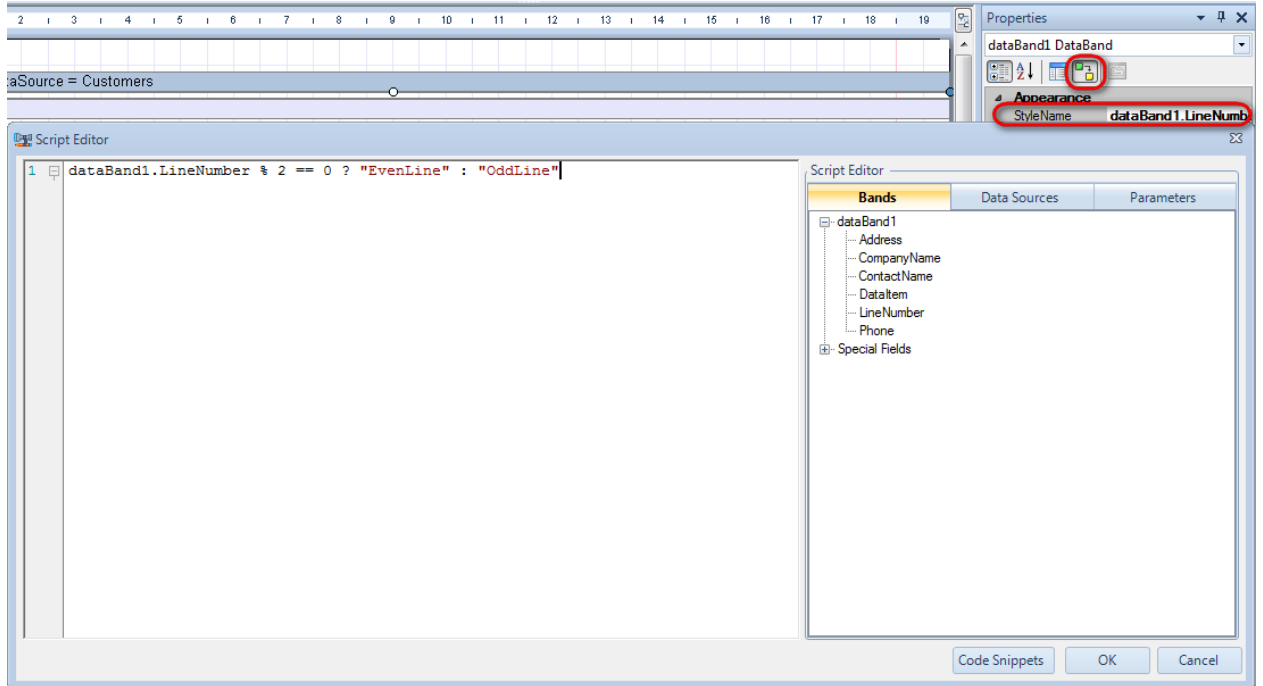
Click on the DataBand area to add Detail band inside DataBand.



Select dataBand1 DataBand in the combo box in the "Properties" type.

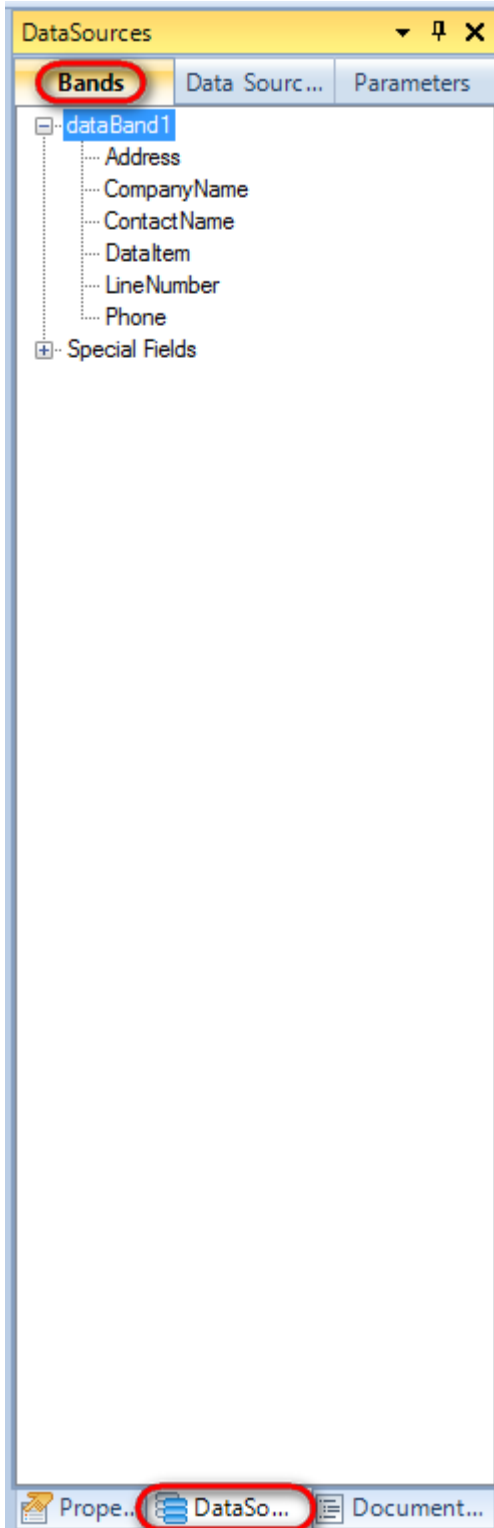


Click the "Bindings" button in the "Properties" tab, set the StyleName property: `StyleName = dataBand1.LineNumber % 2 == 0 ? "EvenLine" : "OddLine"`.



Step 17

Go to "DataSources" tab.



Drag and drop fields "CompanyName", "Address", "ContactName", "Phone" from the dataBand1 tree to the detail1 band. As a result TextBox elements are created. Script loading data from the data source is added to the Value property.

Change size of the elements and located them in the following way:

dataBand1:DataBand DataSource = Customers			
detail1:Detail			
<dataBand1 ["CompanyName"]>	<dataBand1 ["ContactName"]>	<dataBand1 ["Address"]>	<dataBand1["Phone"]>
end of dataBand1			



Step 18

Save template, close Report Designer.

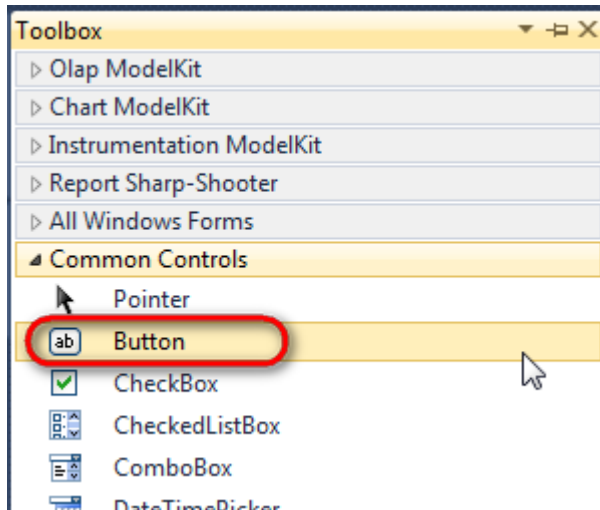
Step 19

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

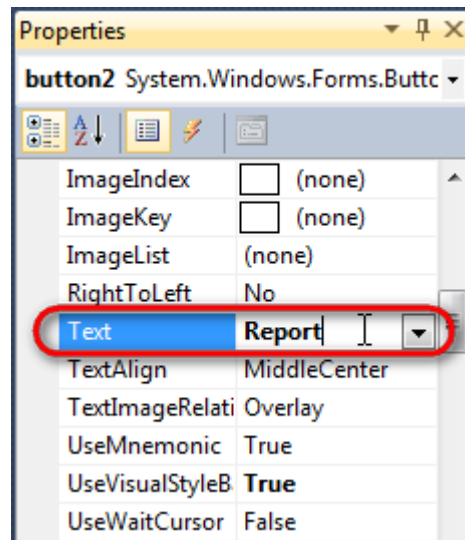
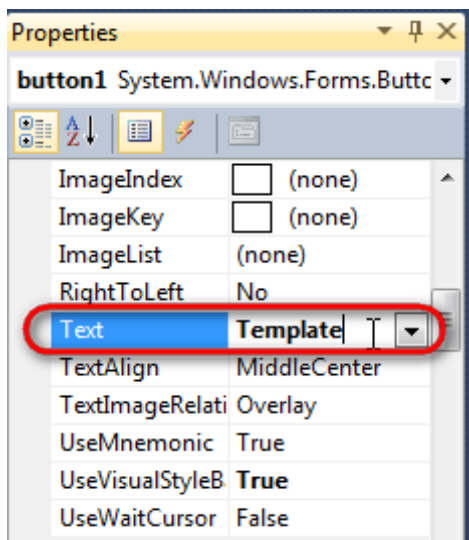
```
public Form1()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["Address"] = "Obere Str. 57";
    row["ContactName"] = "Maria Anders";
    row["Phone"] = "030-0074321";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["Address"] = "Avda. de la Constitución 2222";
    row["ContactName"] = "Ana Trujillo";
    row["Phone"] = "(5) 555-4729";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["Address"] = "Kirchgasse 6";
    row["ContactName"] = "Roland Mendel";
    row["Phone"] = "7675-3425";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["Address"] = "Luisenstr. 48";
    row["ContactName"] = "Karin Josephs";
    row["Phone"] = "0251-031259";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 20

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



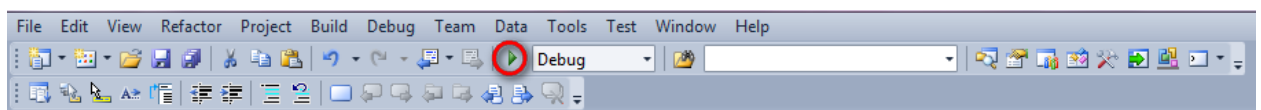
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

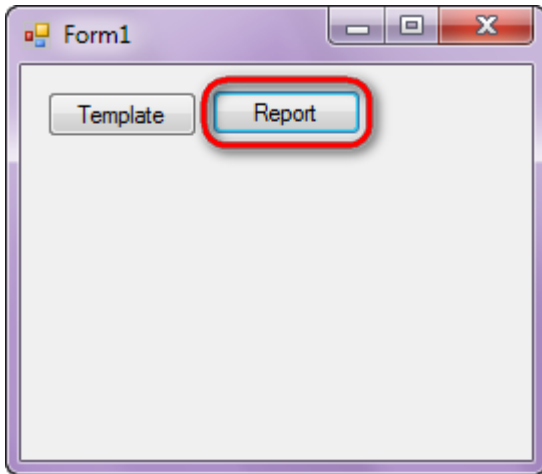
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 21

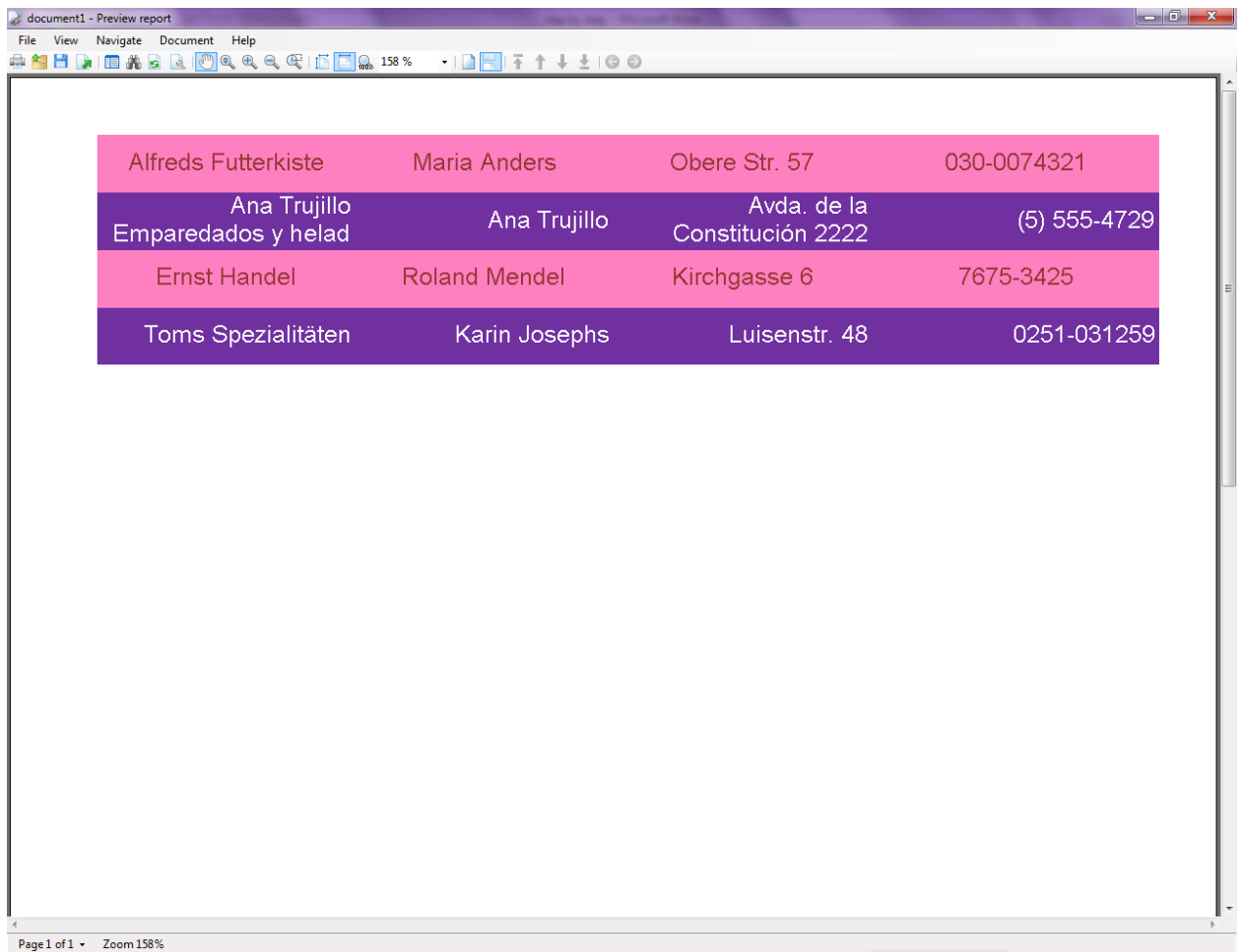
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



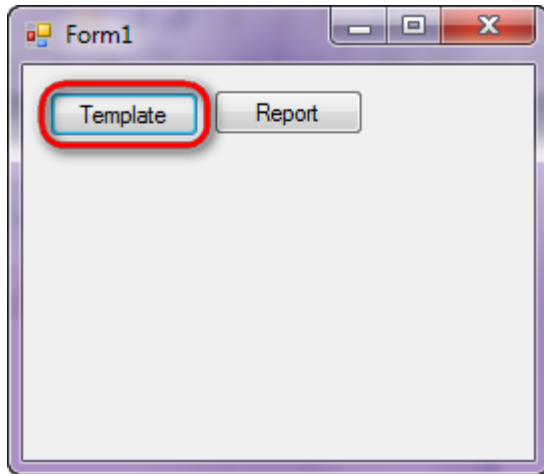
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



Similar sample in the Samples Center is Scripting & Binding\Odd/Even lines highlighting.

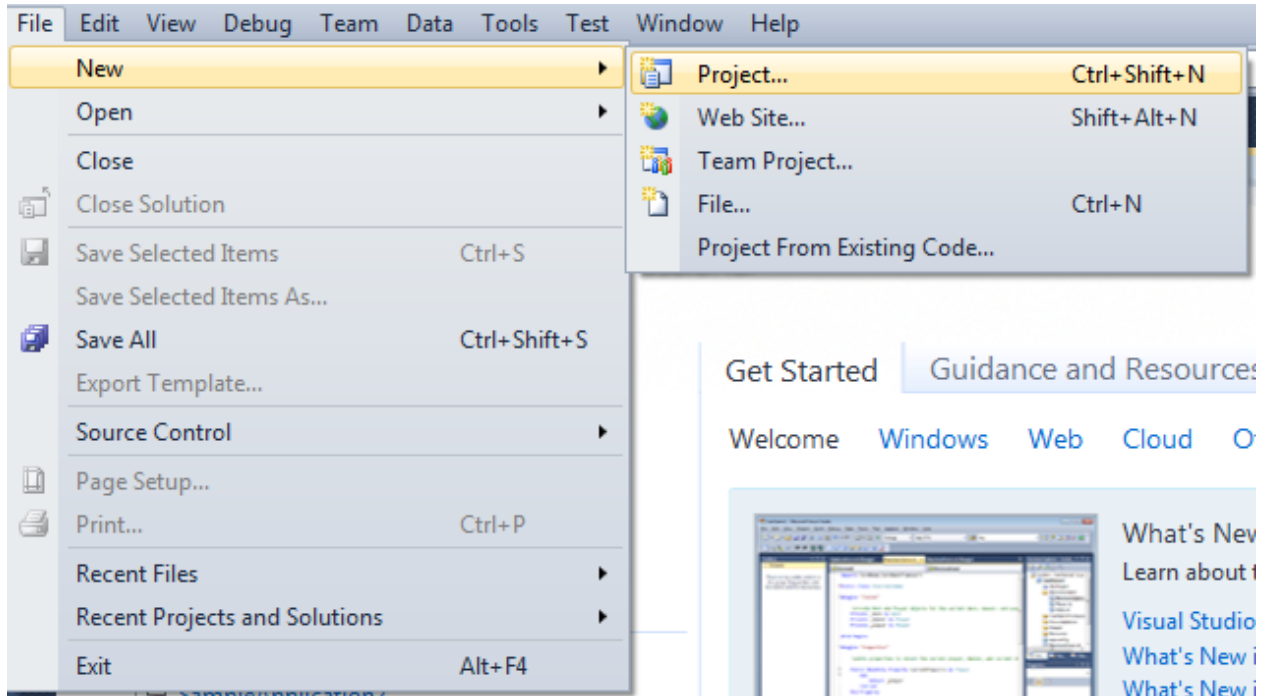


Sorting

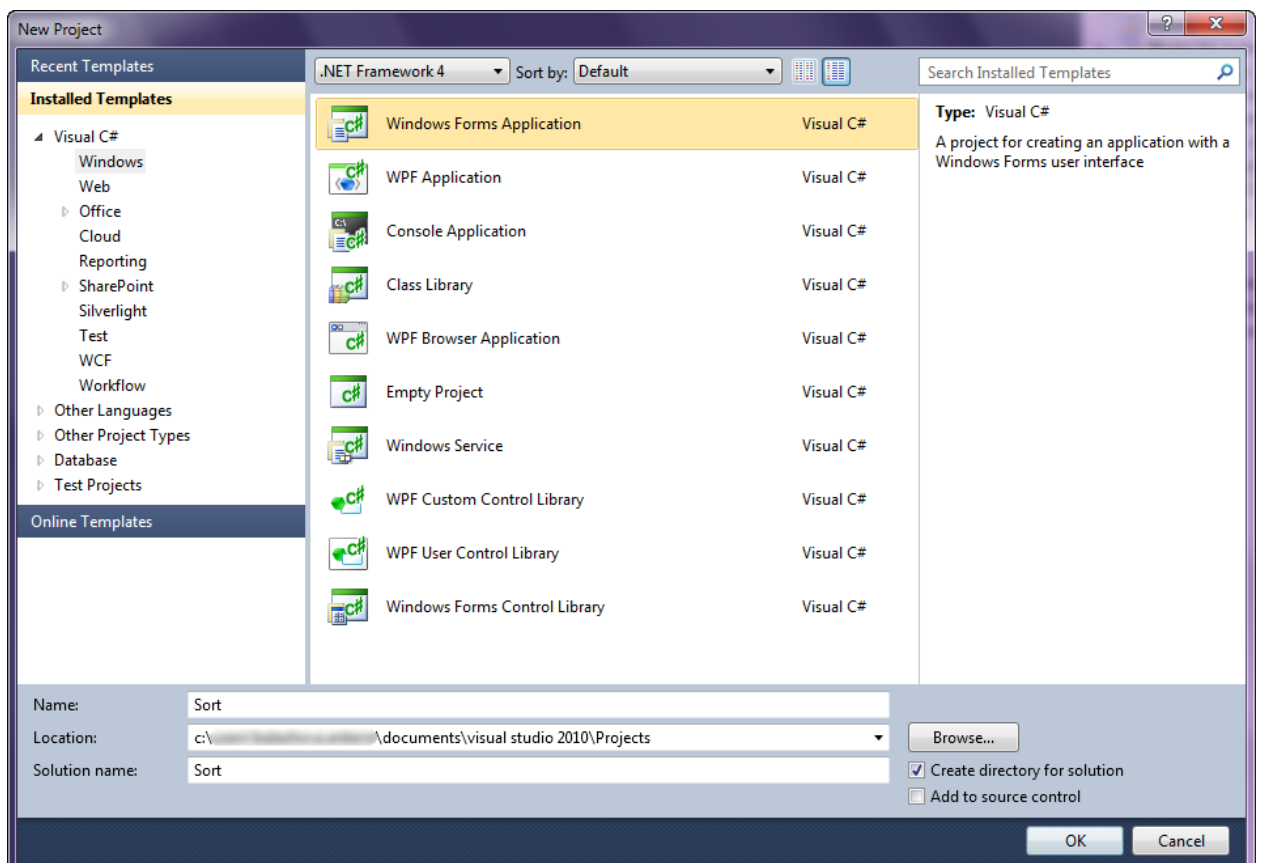
Template of a report containing a list of customers sorted by city and company name.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.



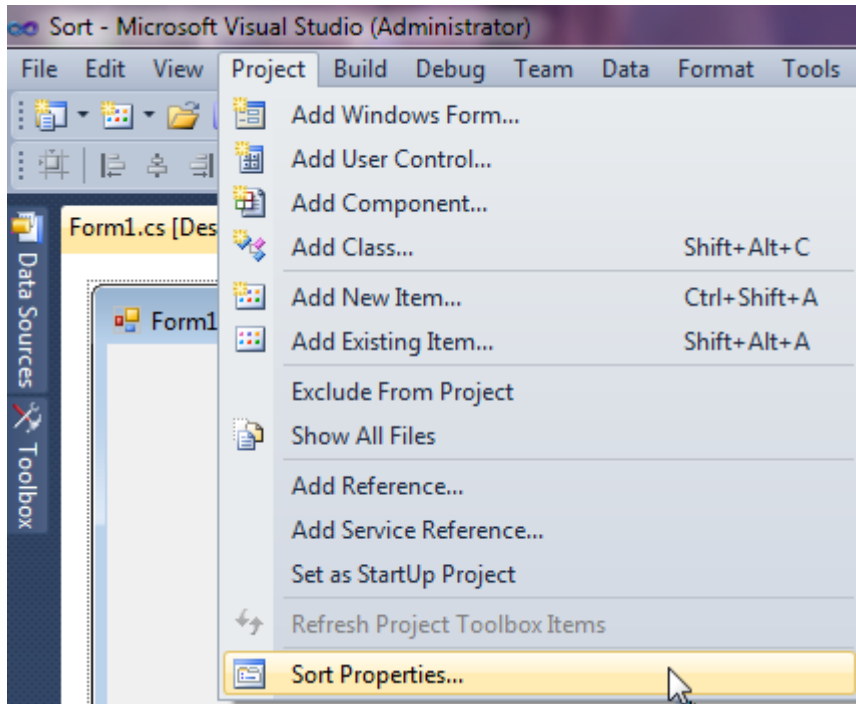
Select Windows Forms Application, set project name – “Sort”, set directory to save the project to.



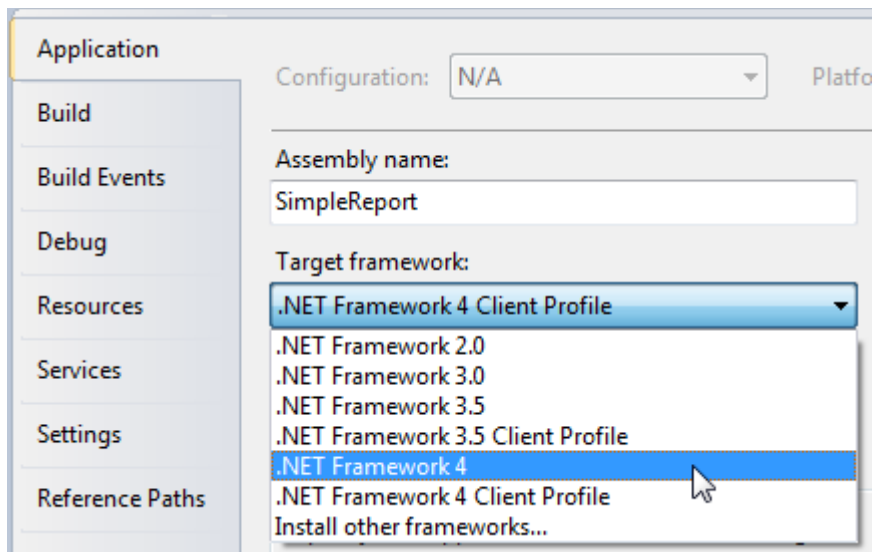


Step 2

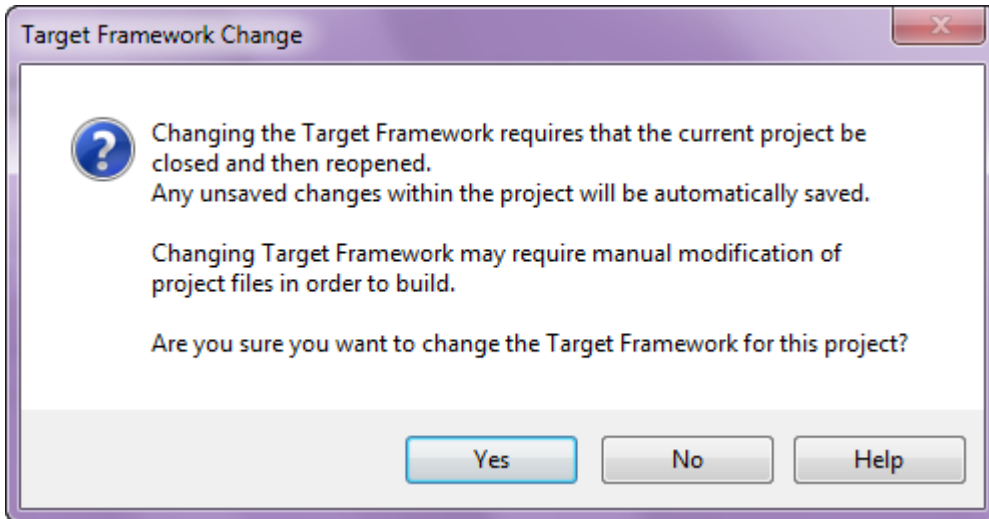
Change the project properties. Select the Project\Sort Properties... item in the main menu.



Select the Target framework*.NET Framework4 item in the Application tab.

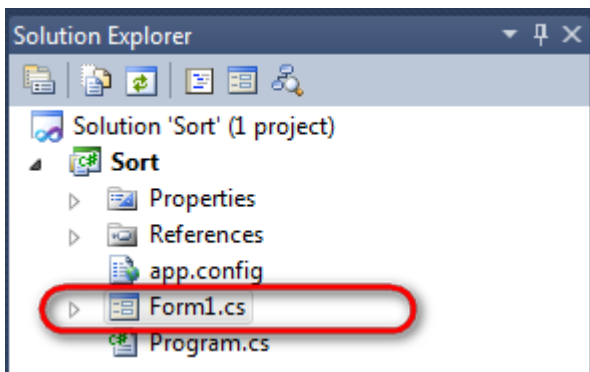


Press the "Yes" button in the opened window.

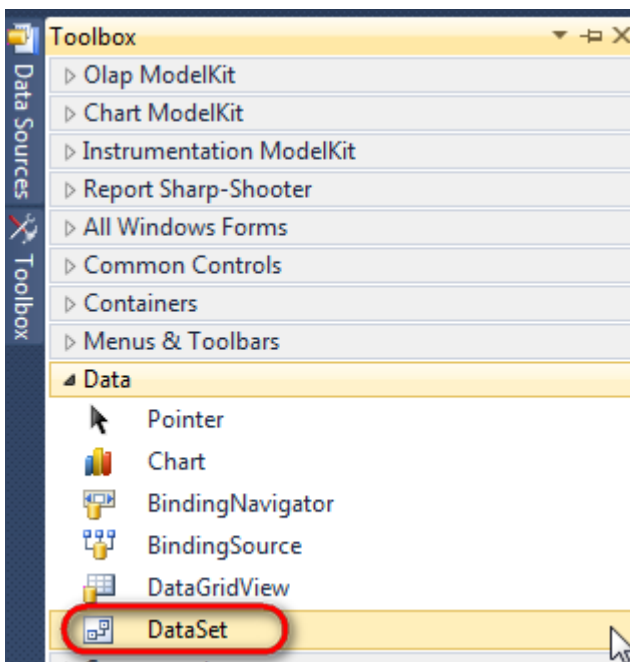


Step 3

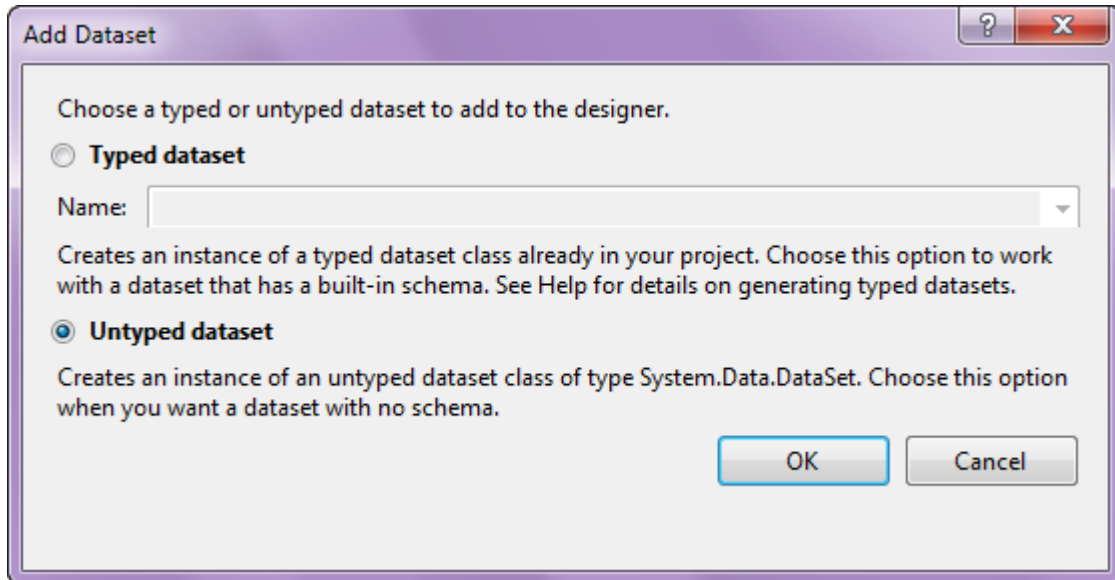
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



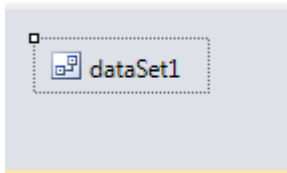
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

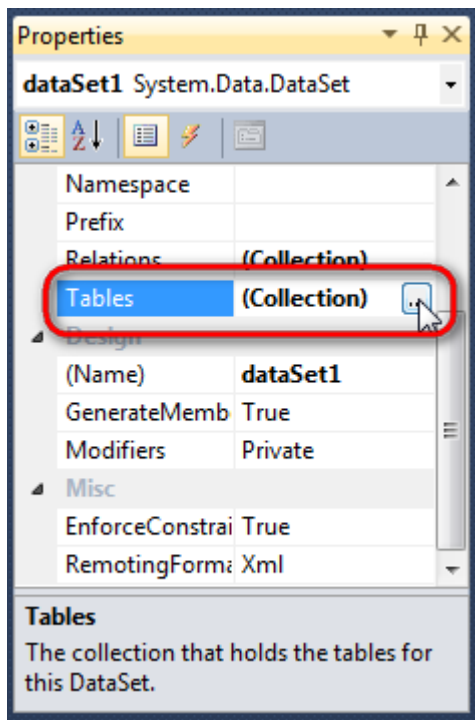


The component is available in the lower part of the window.

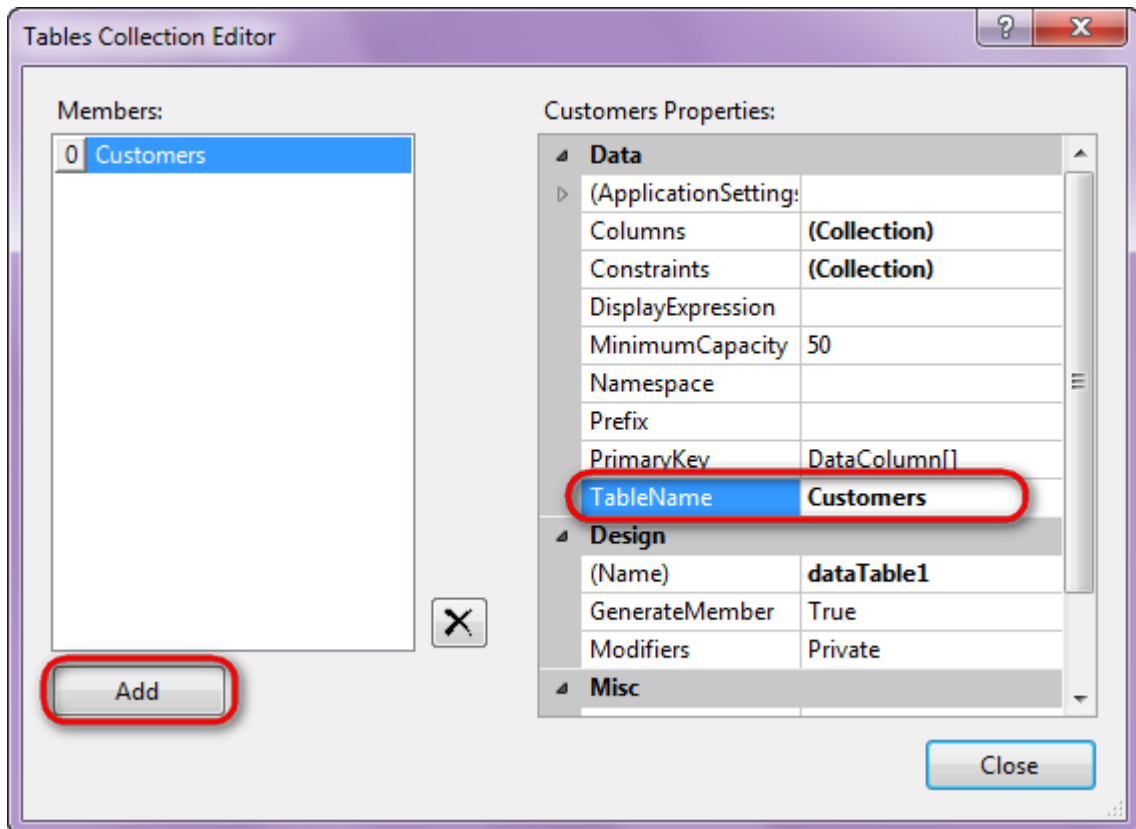


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

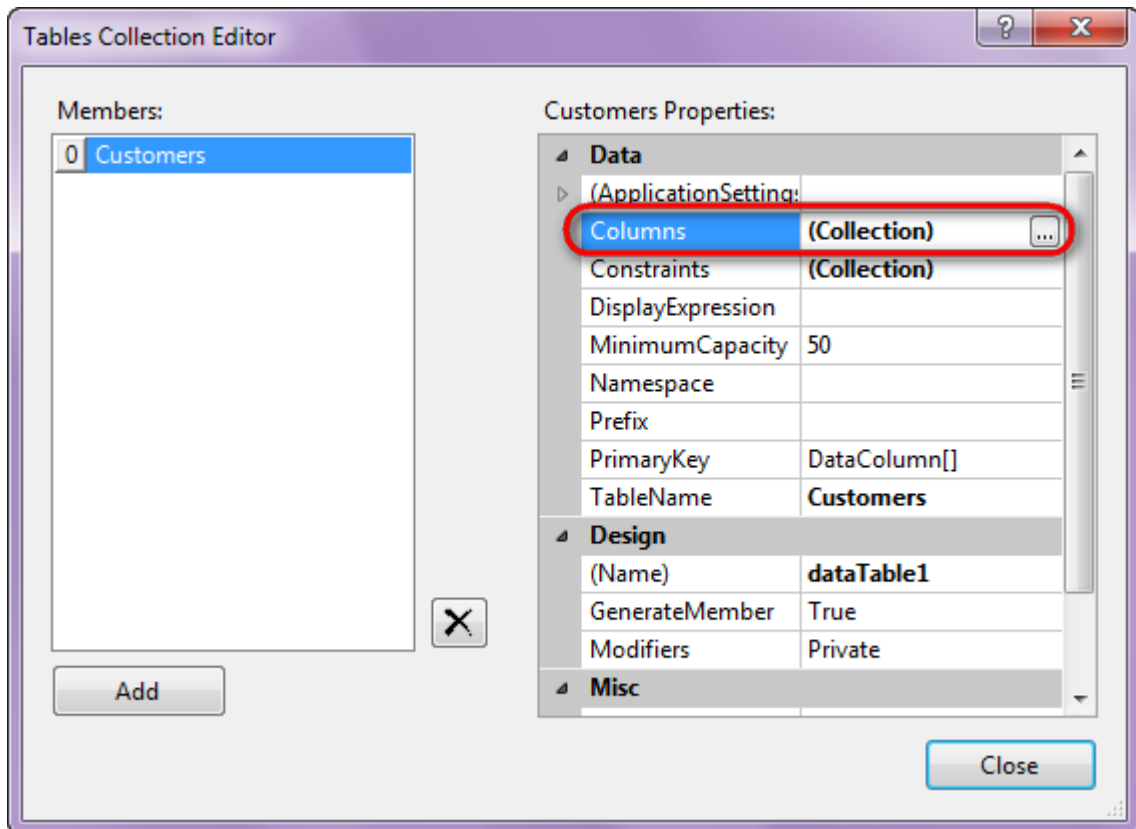


Click "Add" in order to add table. Set property TableName = Customers.

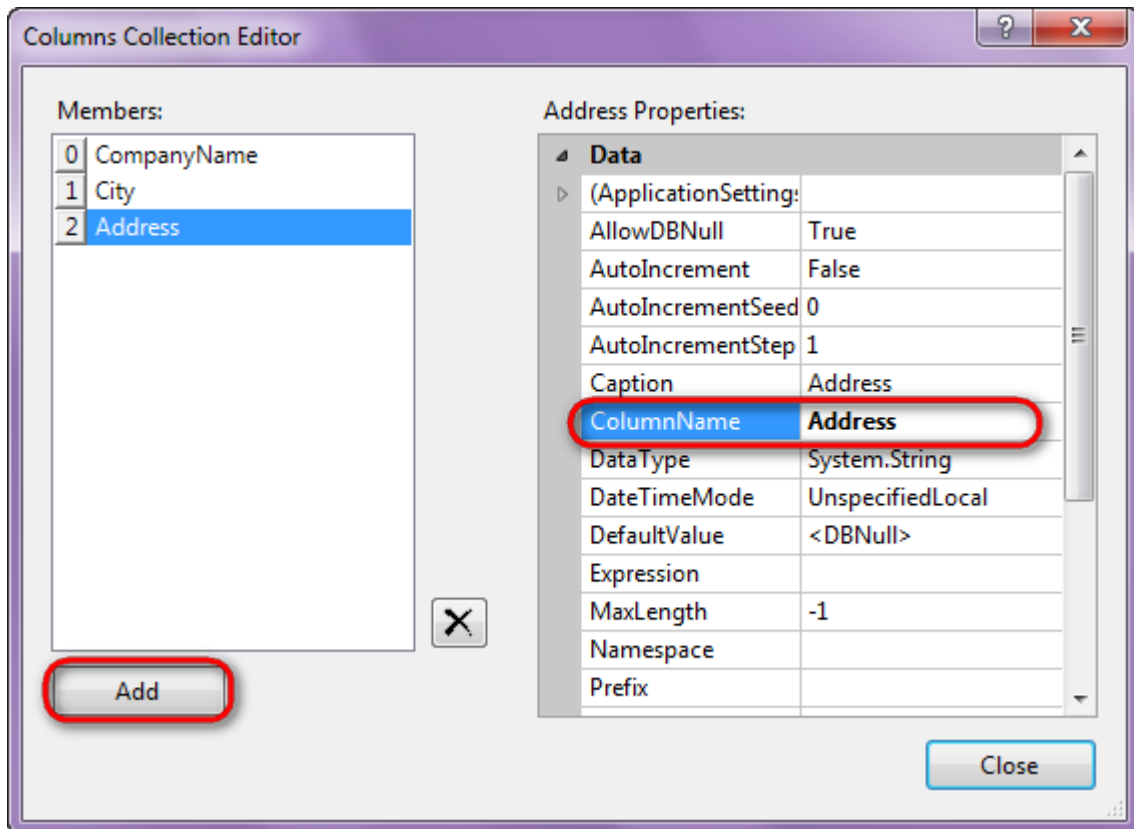


Step 5

Select Columns property, click button  in order to open property editor.

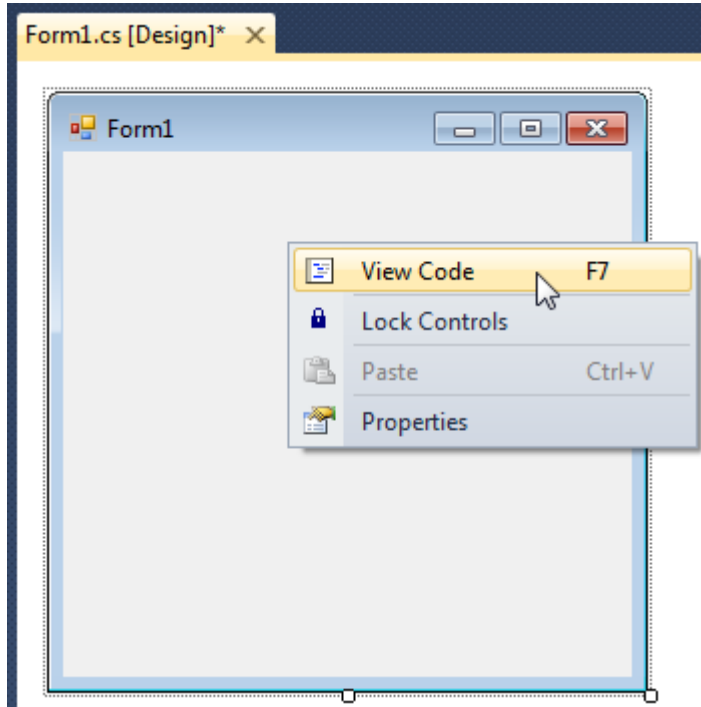


Click "Add" to add a new column. Add three columns. Set ColumnName property to "CompanyName", "City", and "Address".



Step 6

Right click on the form and select "View Code" in the context menu to view code.



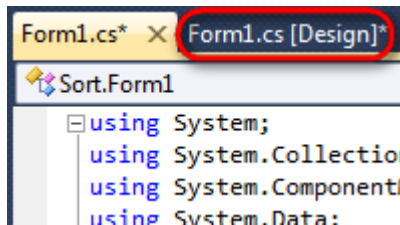
Add the following code to the class constructor in order to fill data source.

```
public Form1 ()  
{  
    InitializeComponent ();  
    DataRow row = dataTable1.NewRow ();  
    row["CompanyName"] = "Alfreds Futterkiste";  
}
```

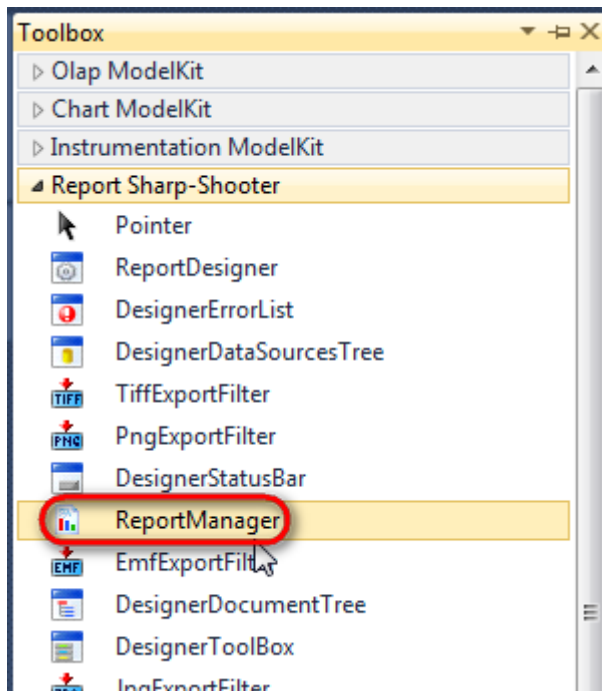
```
row["City"] = "London";  
row["Address"] = "Obere Str. 57";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["City"] = "Paris";  
row["Address"] = "Avda. de la Constitución 2222";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["City"] = "London";  
row["Address"] = "Kirchgasse 6";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["City"] = "New York";  
row["Address"] = "Luisenstr. 48";  
dataTable1.Rows.Add(row);  
}
```

Step 7

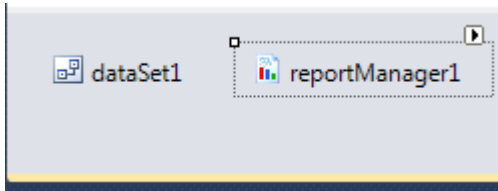
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

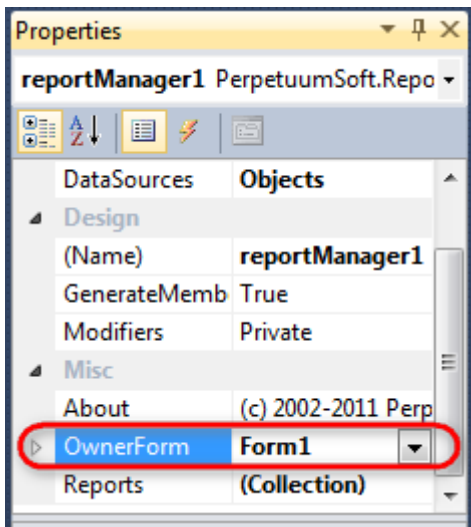


The component is available in the lower part of the window.



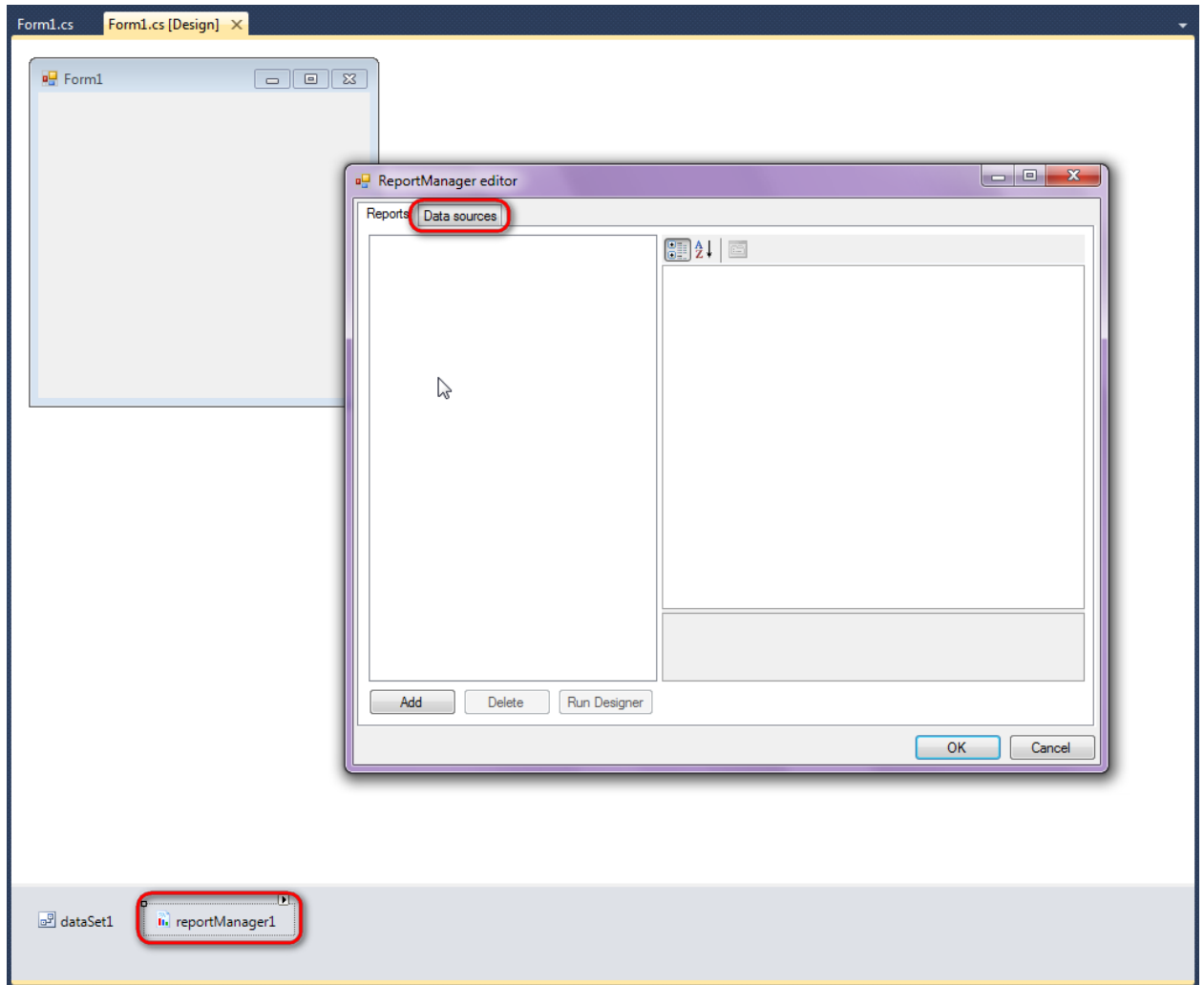
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

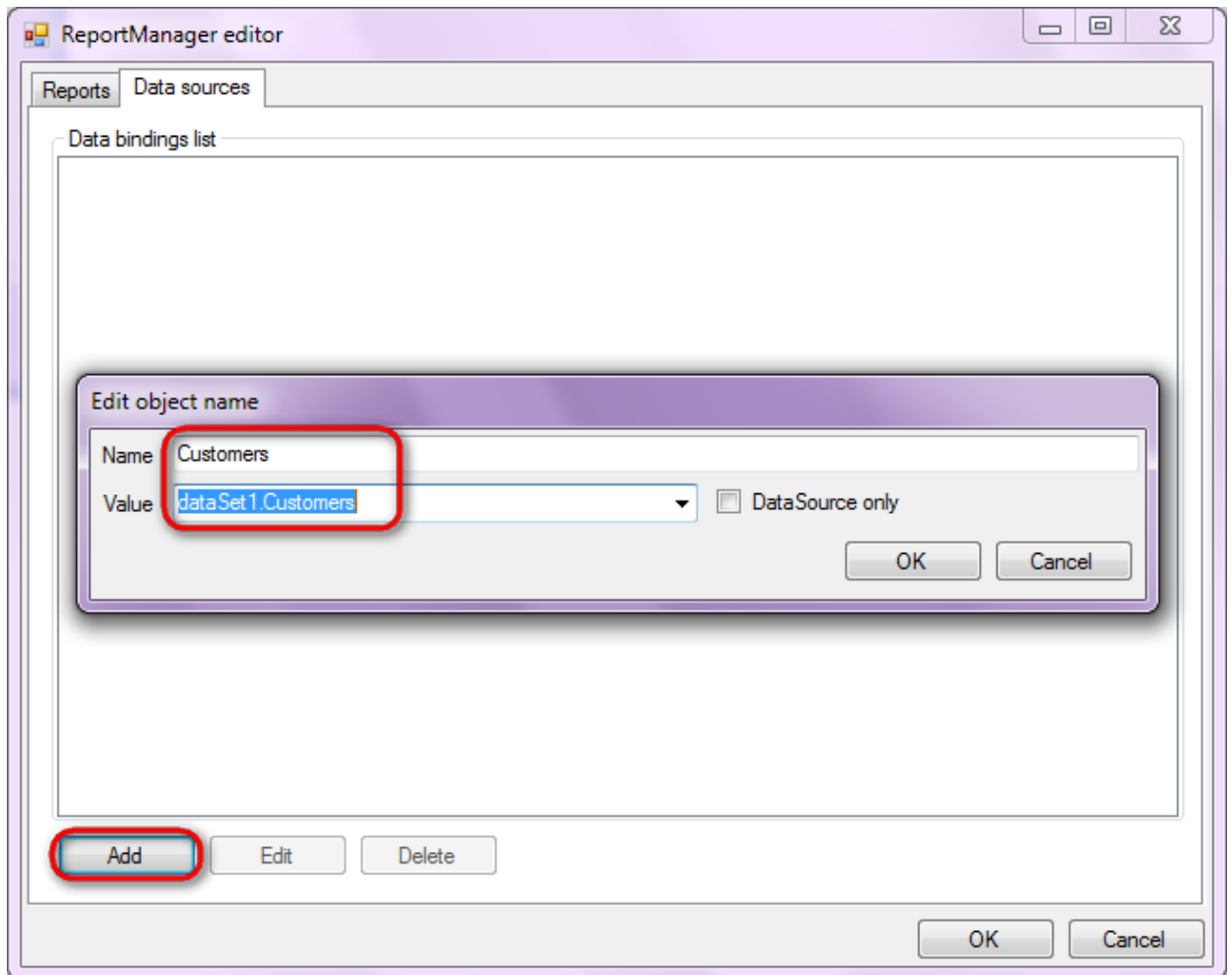


Step 9

Double click on ReportManager to open ReportManager editor.

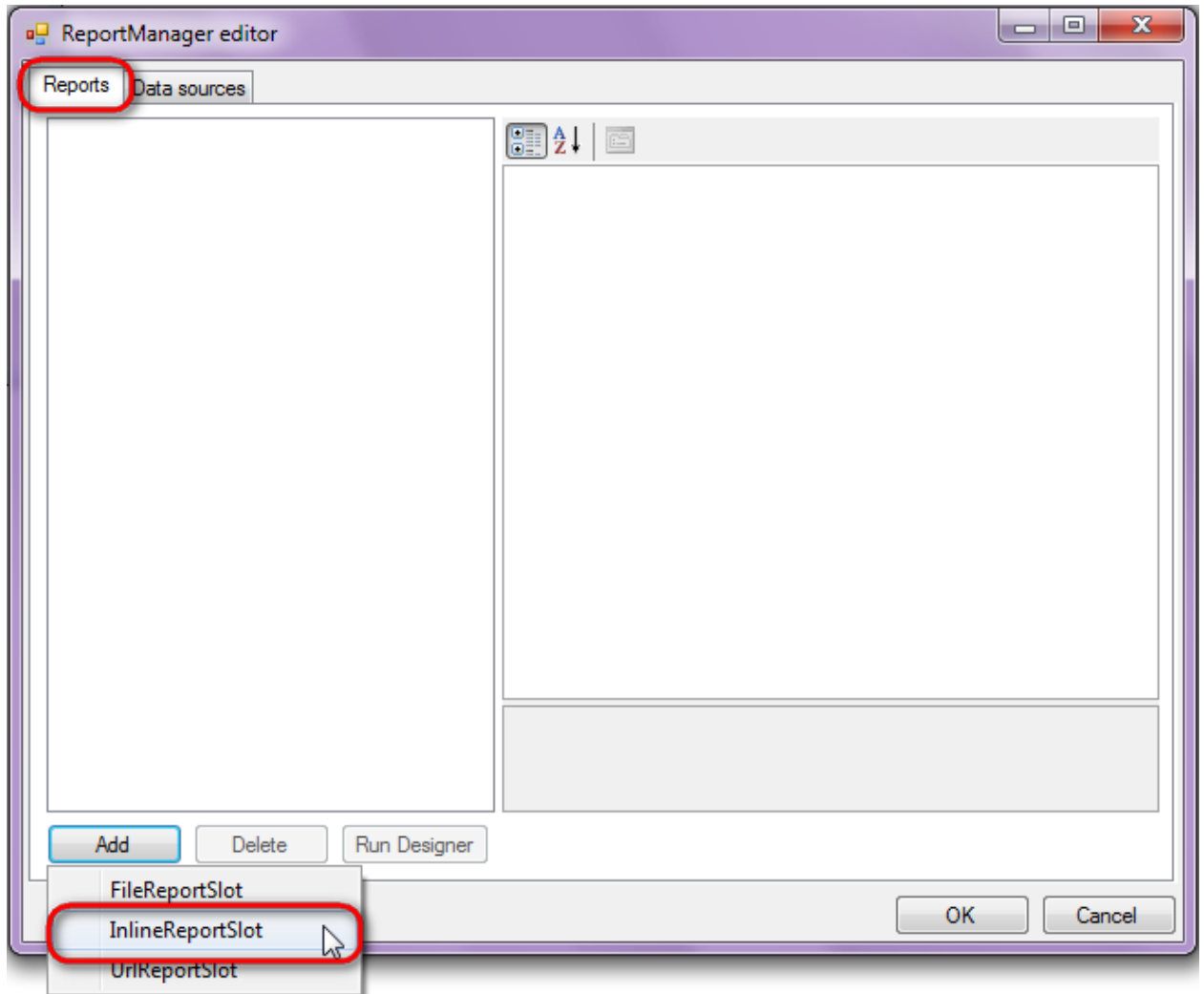


Go to "Data sources" tab, click "Add", set data source name – "Customers", select data source value – "dataSet1.Customers".



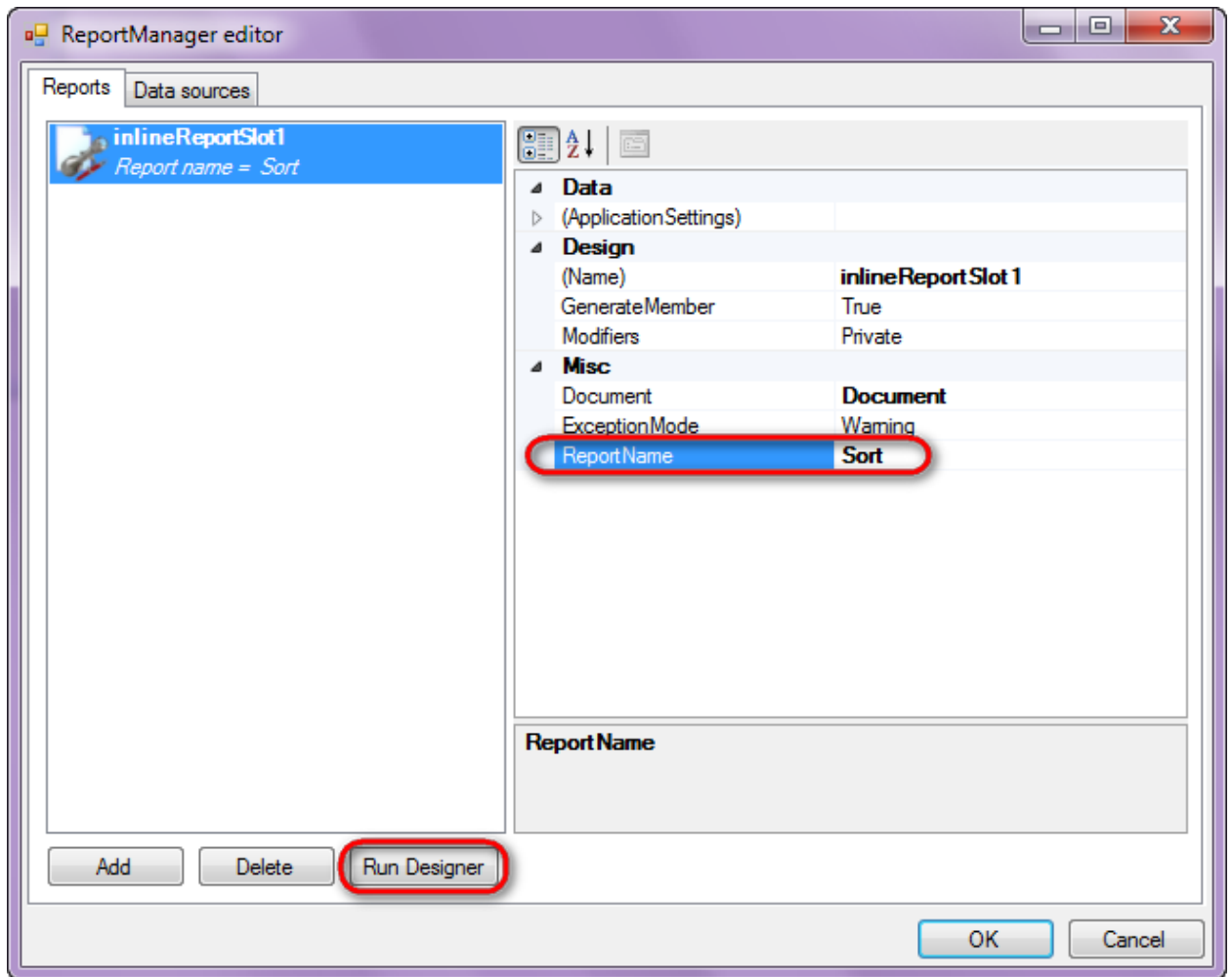
Step 10

Go to "Reports" tab, click "Add" and select "InlineReportSlot".



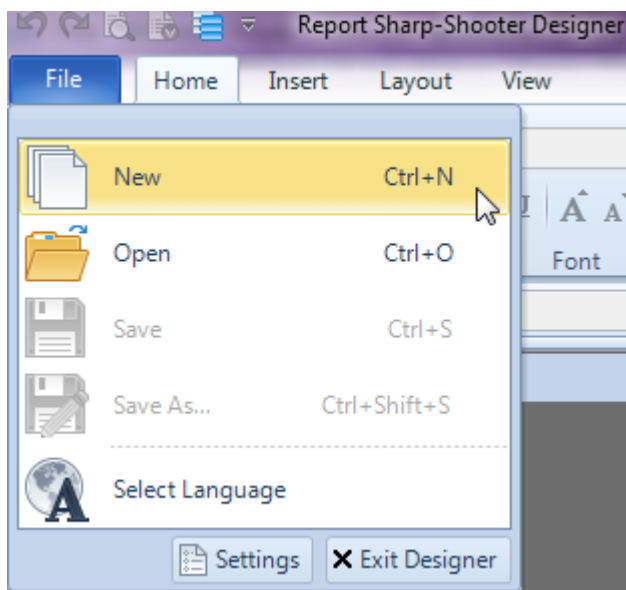
Step 11

Set name of the report in the property ReportName - "Sort". Click "Run Designer" in order to open template editor - Report Designer.

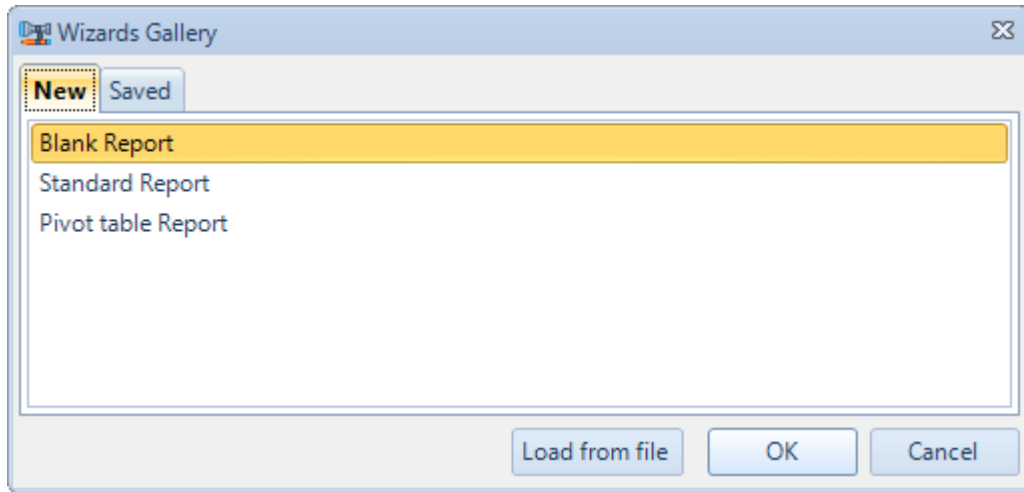


Step 12

Create new empty template – select File\New from the main menu.

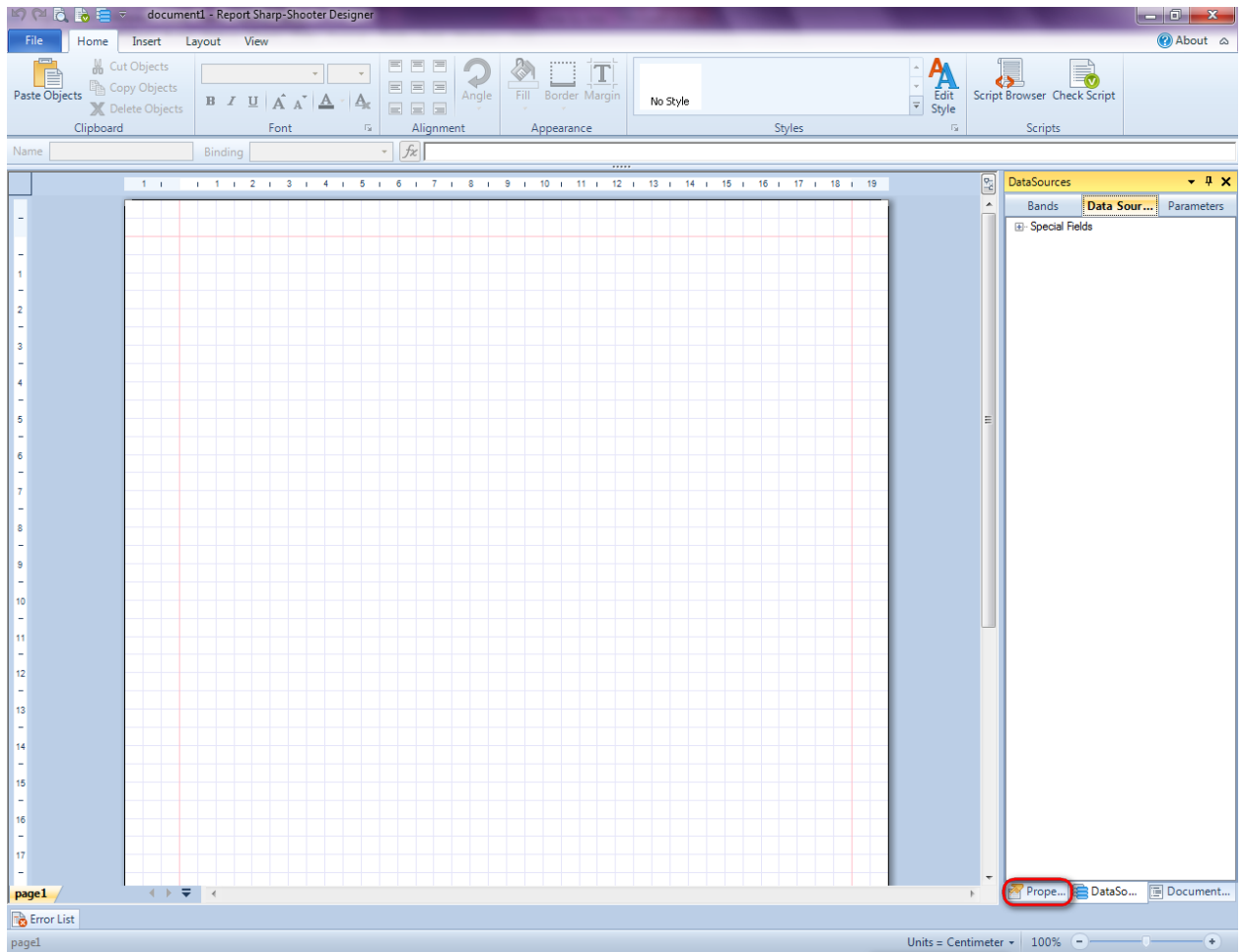


Select "Blank Report" in the Wizards Gallery and click "OK".

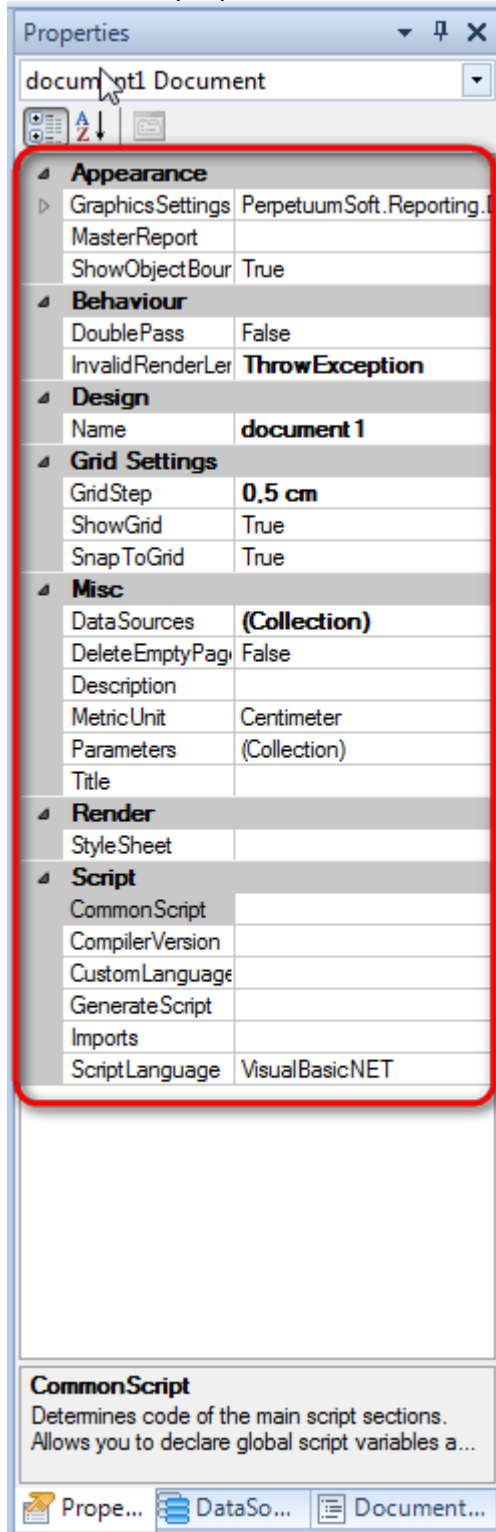


Step 13

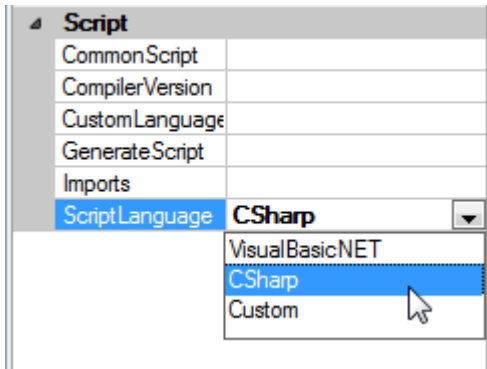
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

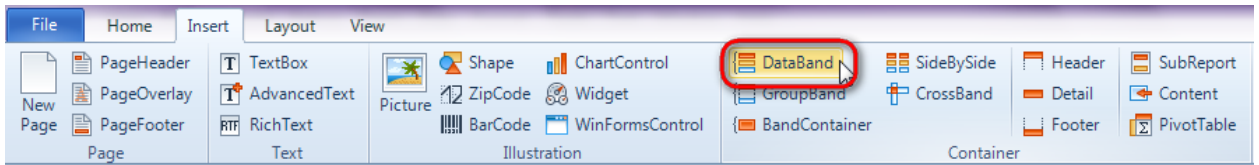


Set property ScriptLanguage = CSharp.



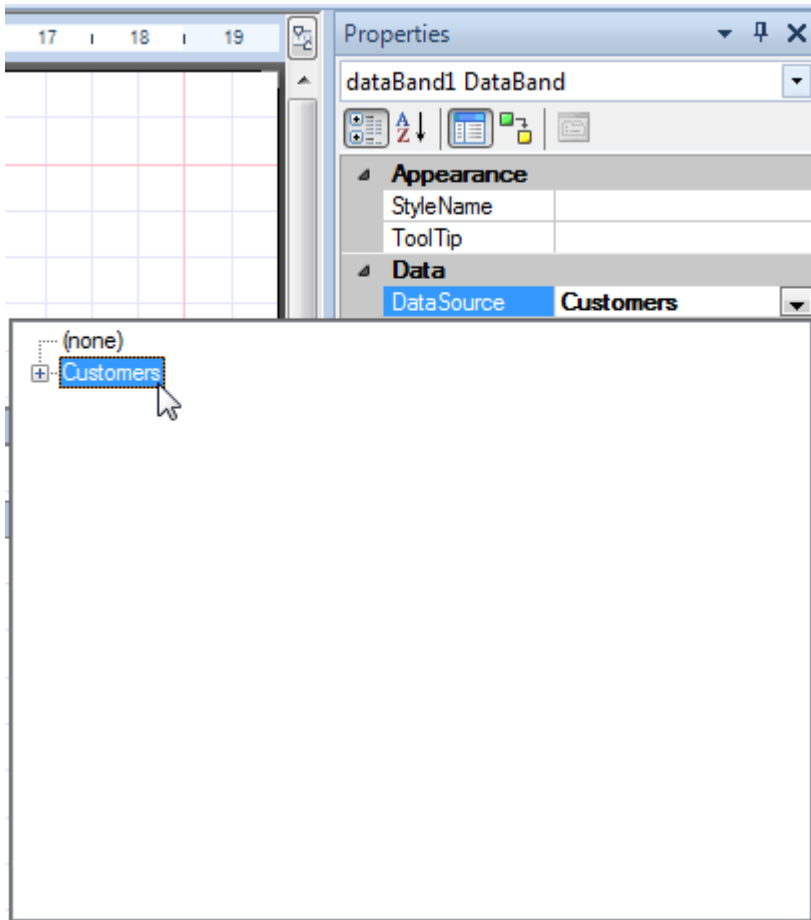
Step 14

Press "DataBand" button on the Insert tab in the group Container.

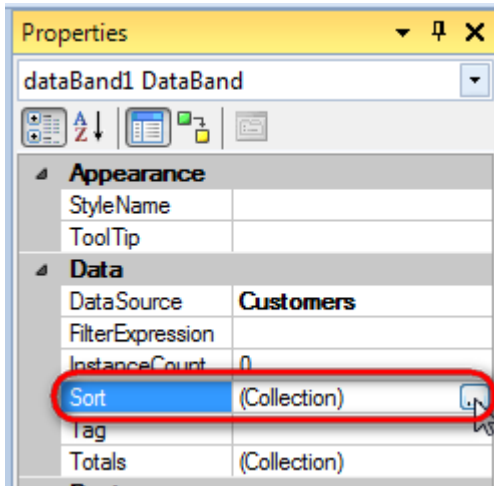



Click on the template area to add DataBand band to the template.

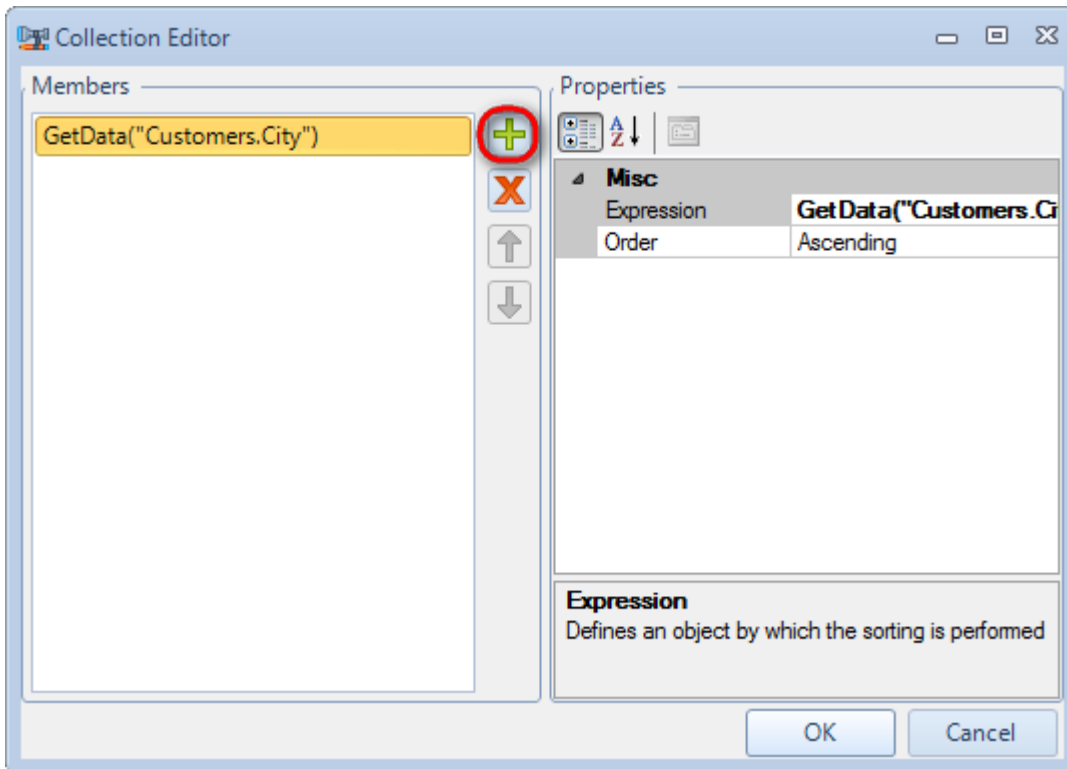
Set data source in the property DataSource = Customers.



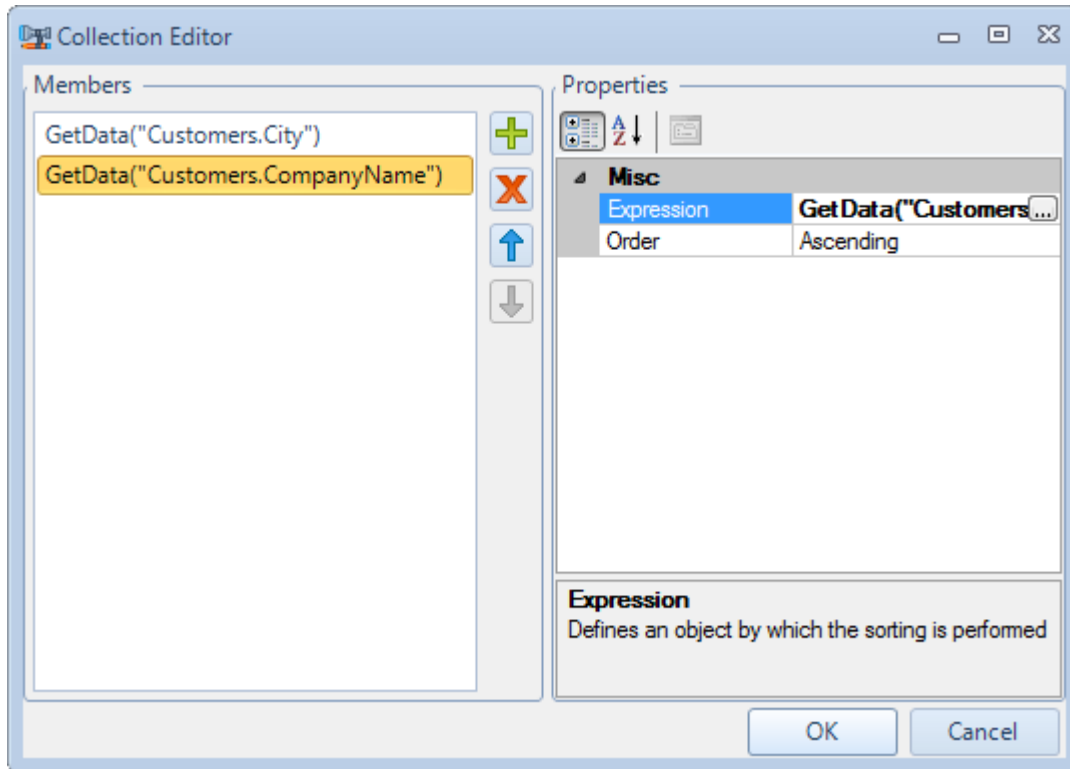
Select Sort property, click button  to open property collection editor.



Click  button to add new sorting condition. Set Expression property to "GetData("Customers.City")", and property Order = Ascending.

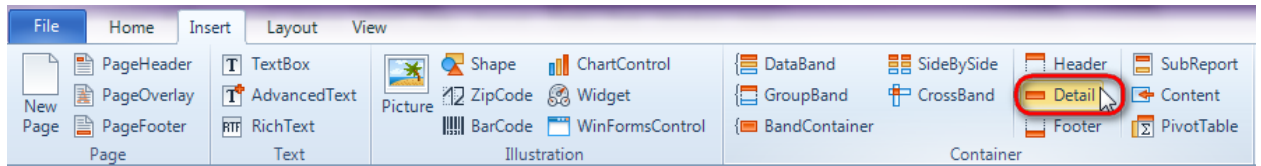


Add one more condition, set Expression = GetData("Customers.CompanyName"), Order = Ascending.

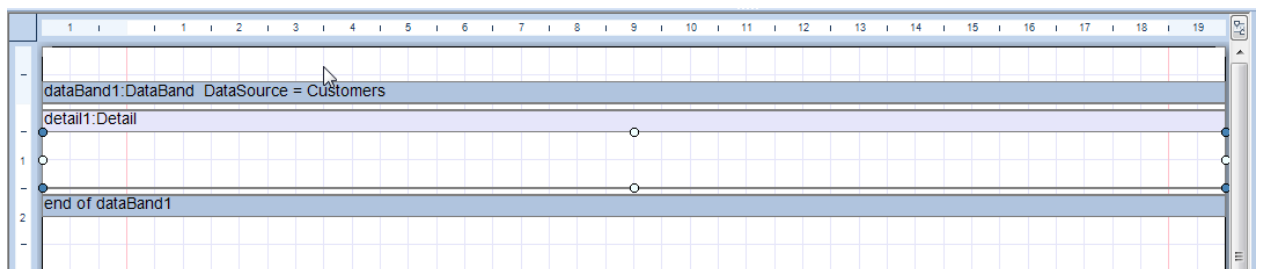


Step 15

Press "Detail" button on the Insert tab in the group Container.

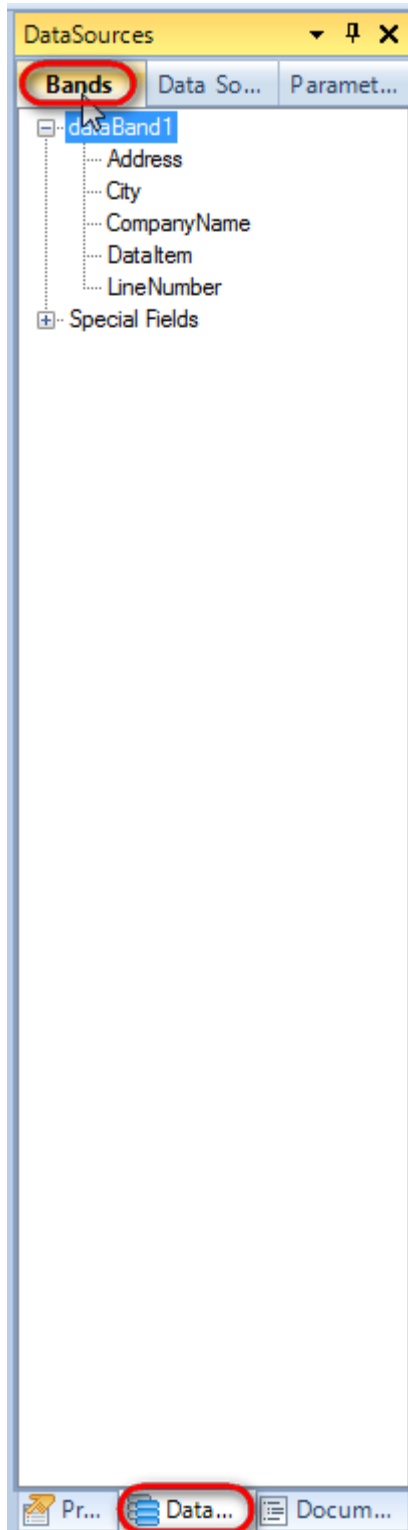


Click on the DataBand area to add Detail band inside DataBand.



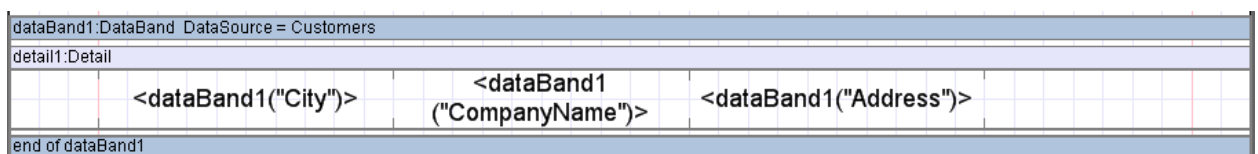
Step 16

Go to "Data Source" tab.



Drag and drop "CompanyName", "City", "Address" fields from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.

Change size of the elements and locate them in the way shown in the picture below.





Step 17

Save template, close Report Designer.

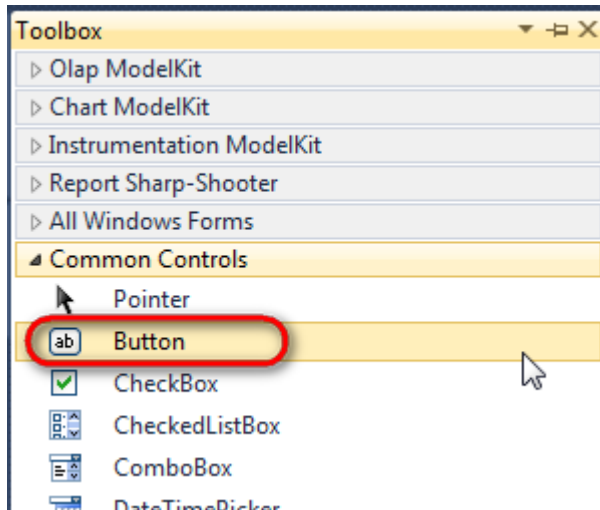
Step 18

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

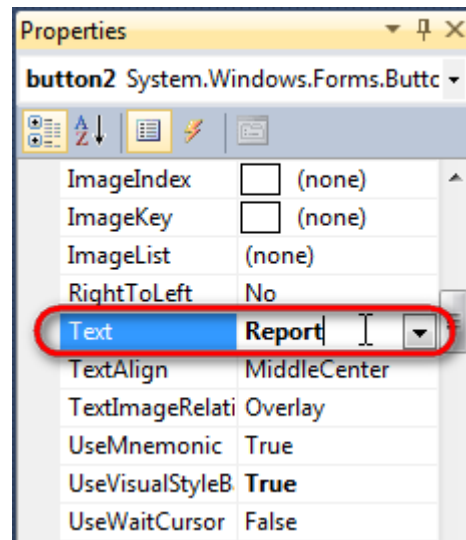
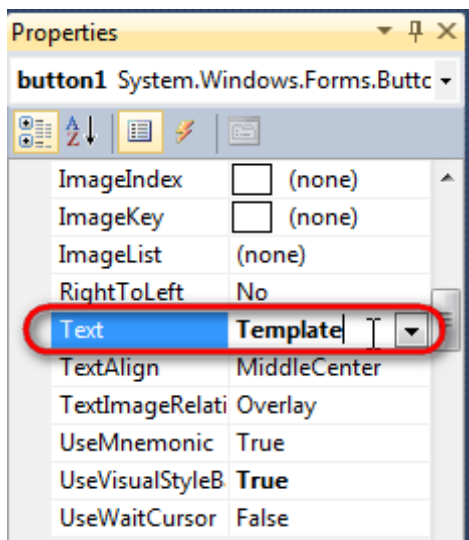
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["City"] = "London";
    row["Address"] = "Obere Str. 57";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["City"] = "Paris";
    row["Address"] = "Avda. de la Constitución 2222";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["City"] = "London";
    row["Address"] = "Kirchgasse 6";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["City"] = "New York";
    row["Address"] = "Luisenstr. 48";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 19

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



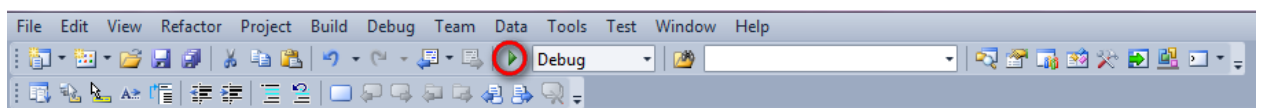
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

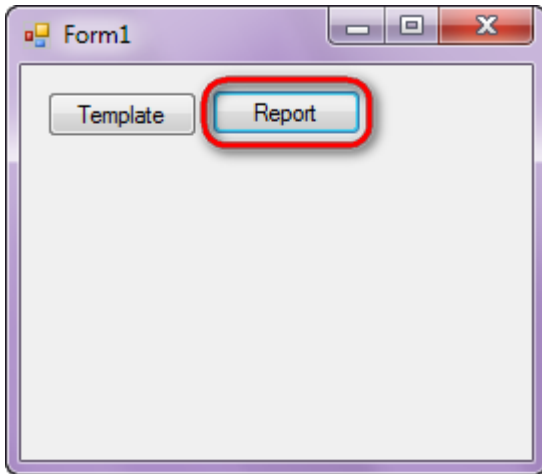
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 20

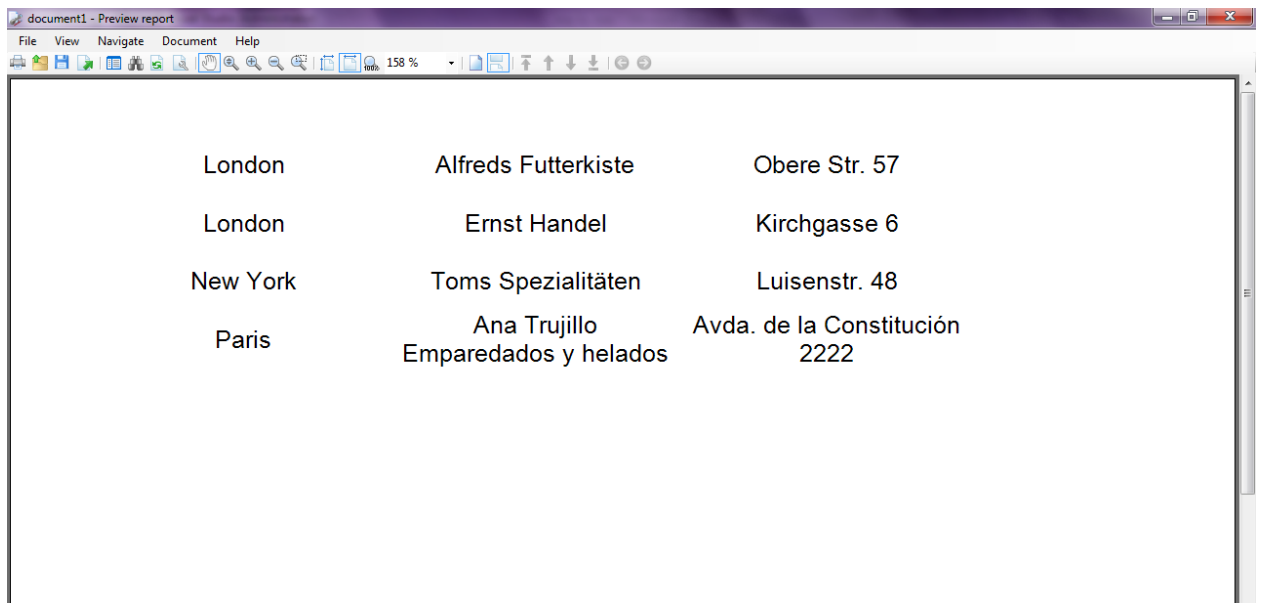
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



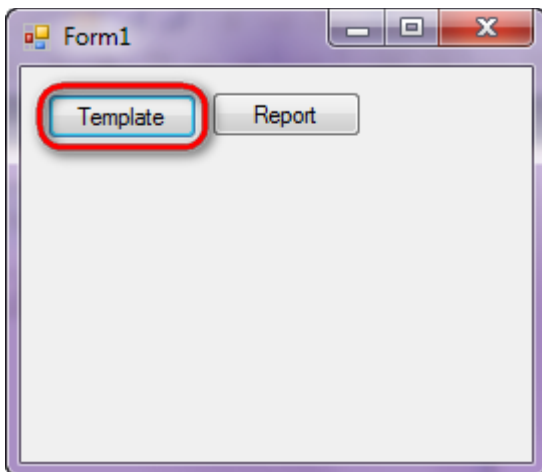
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



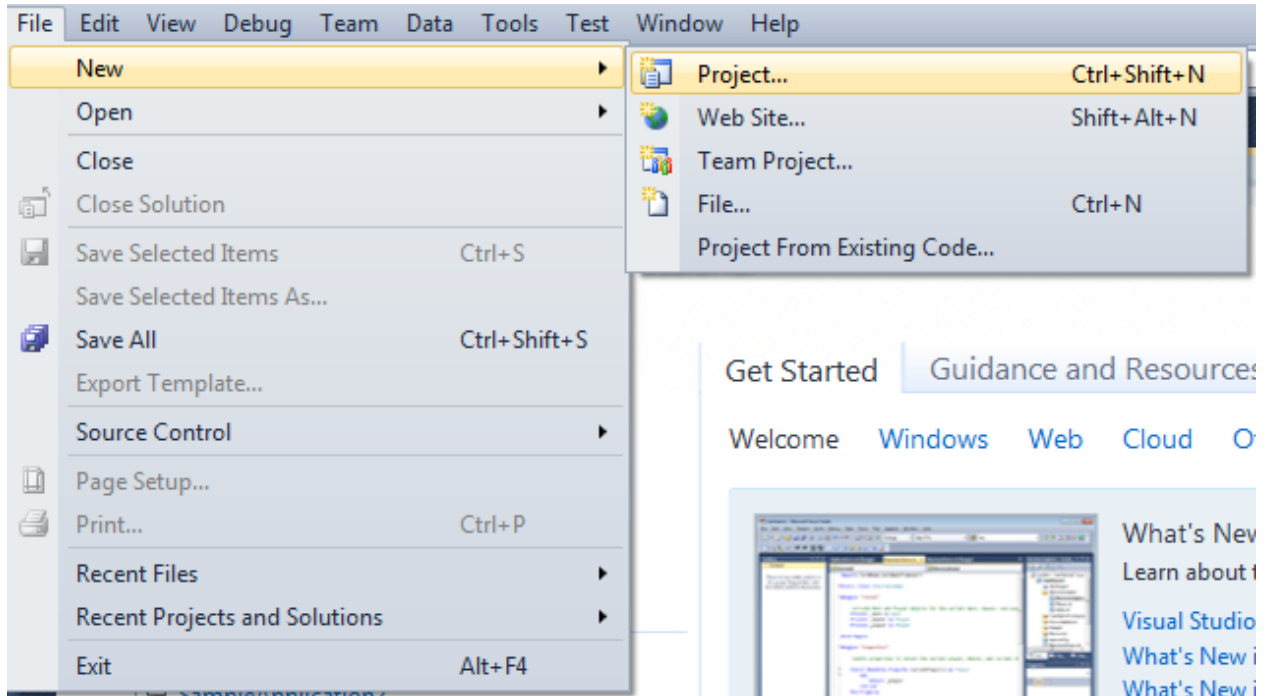


Filtering

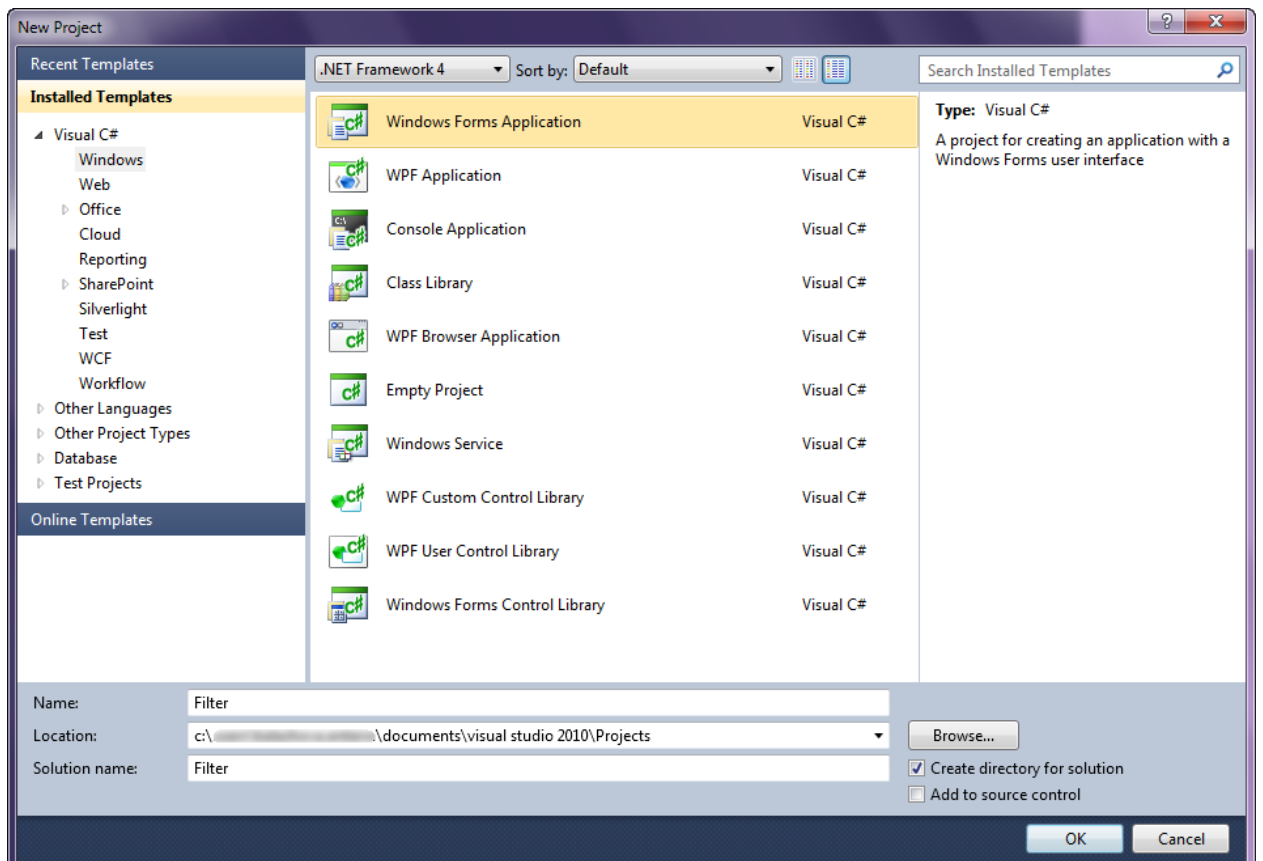
Template of a report containing odd numbers from the set of integers.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.



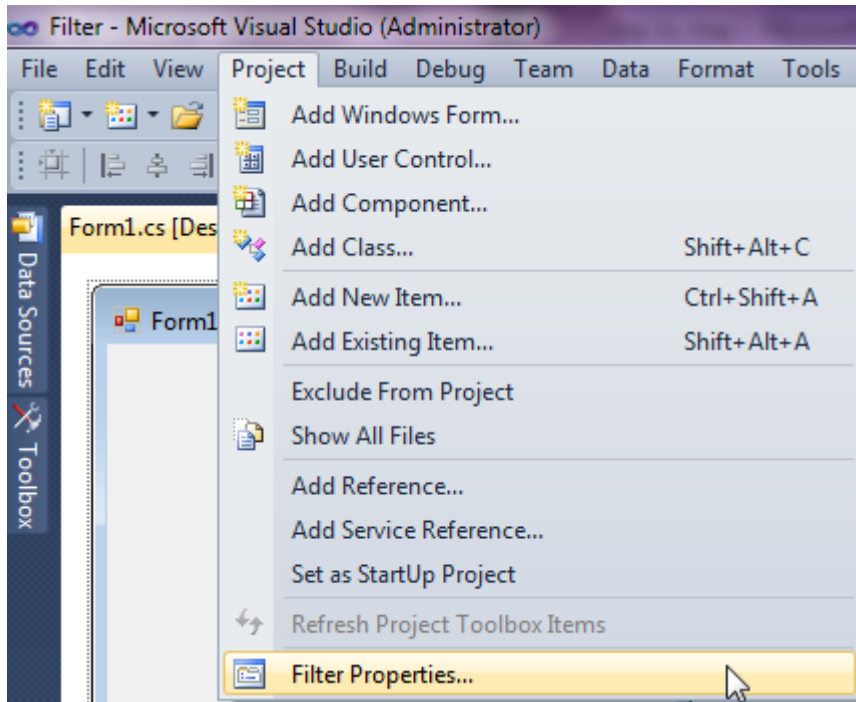
Select Windows Forms Application, set project name – “Filter”, set directory to save the project to.



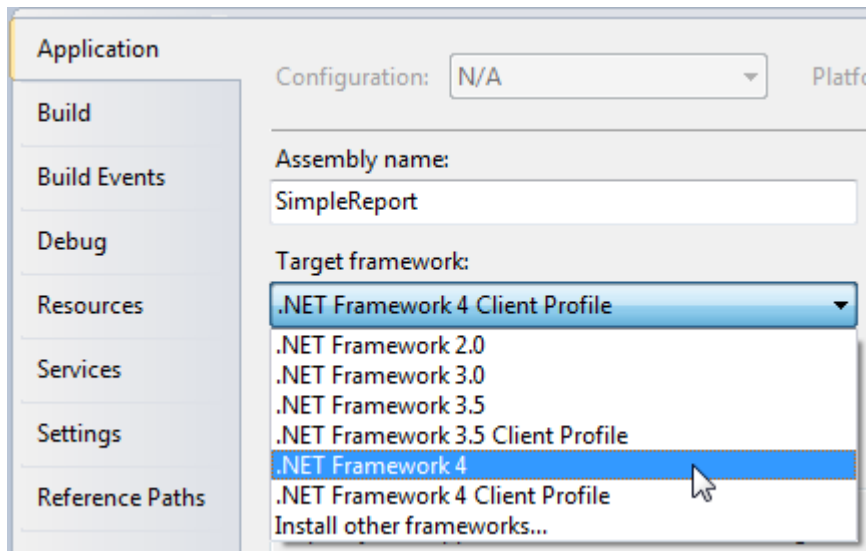


Step 2

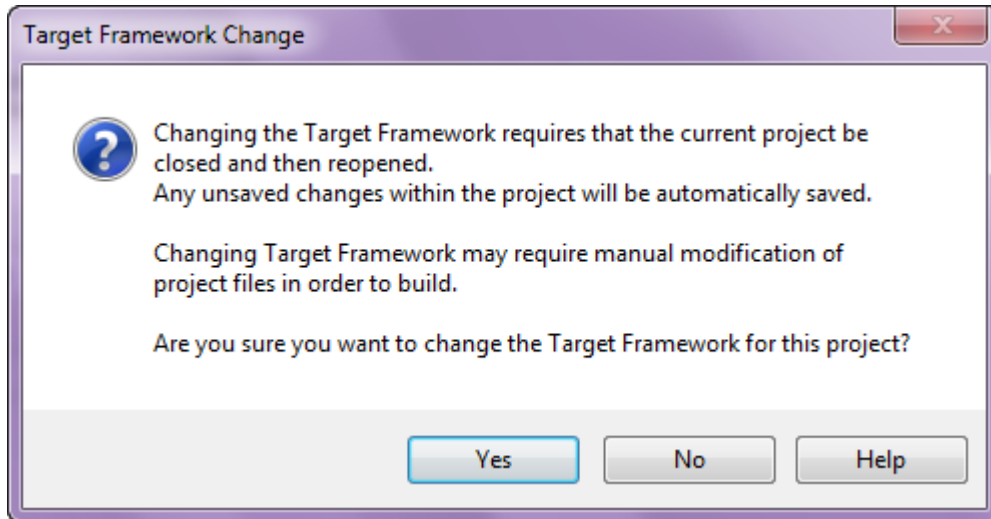
Change the project properties. Select the Project\Filter Properties... item in the main menu.



Select the Target framework*.NET Framework4 item in the Application tab.

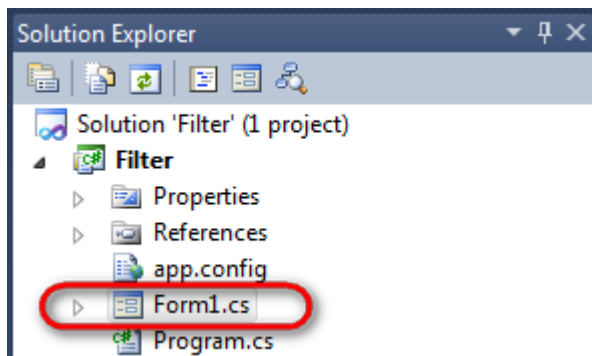


Press the "Yes" button in the opened window.

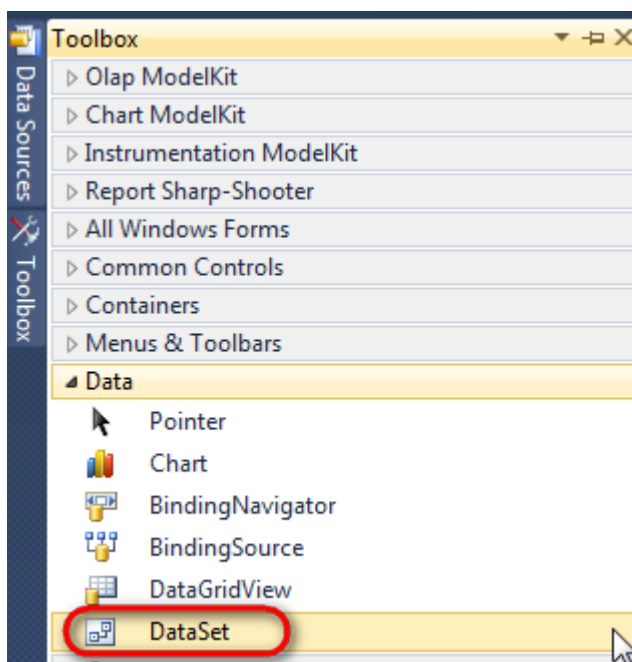


Step 3

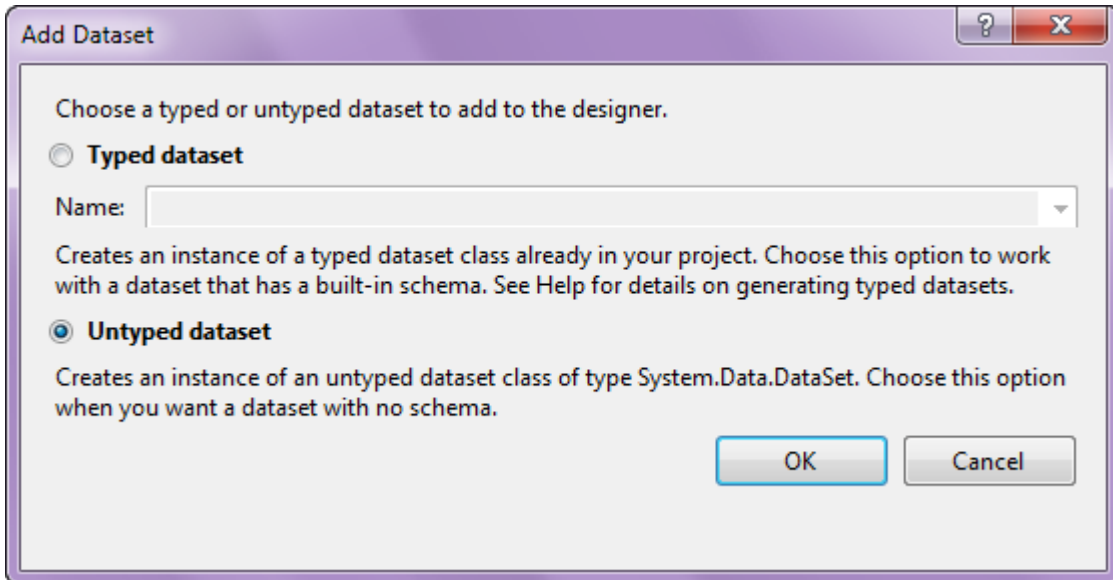
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



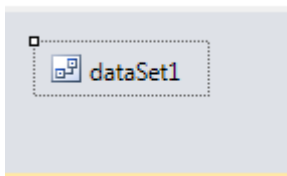
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

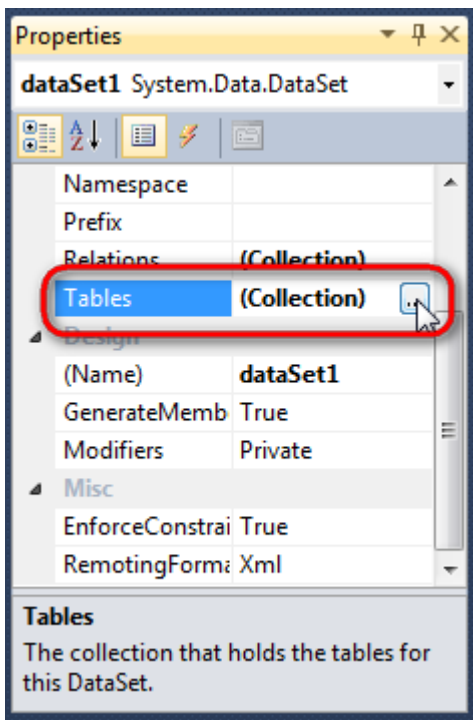


The component is available in the lower part of the window.

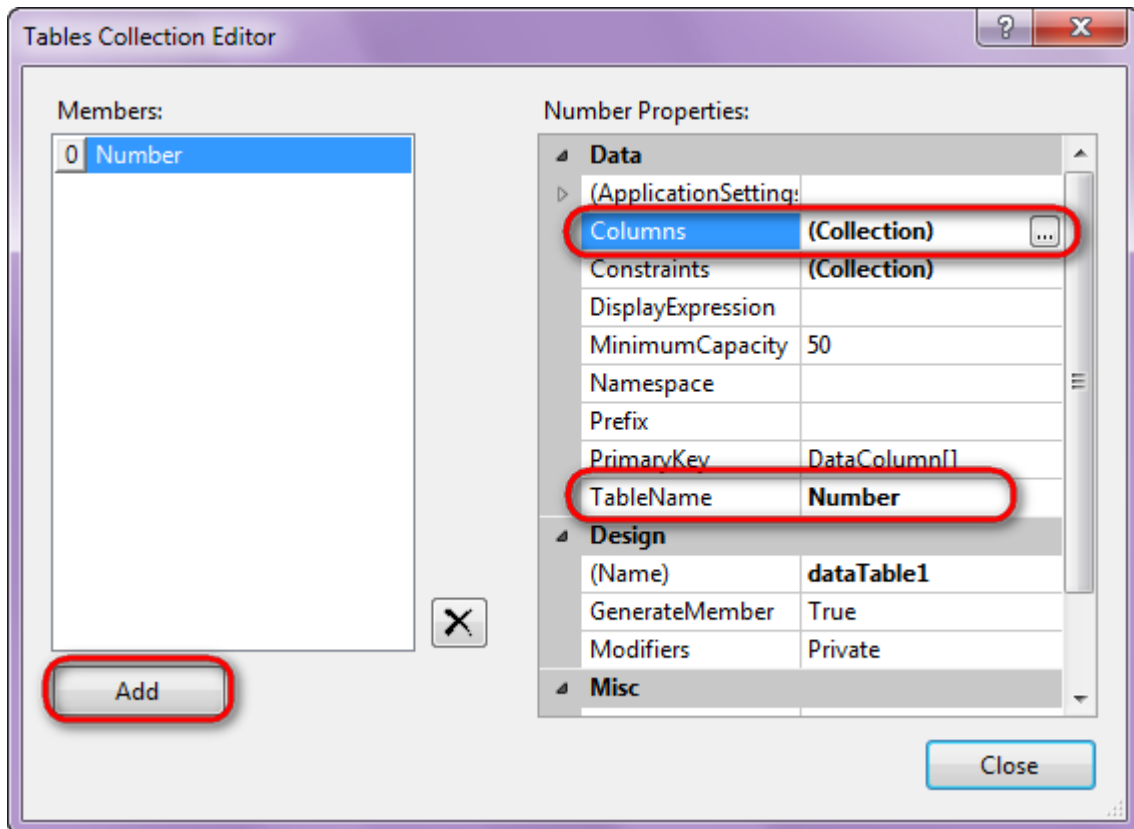


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.



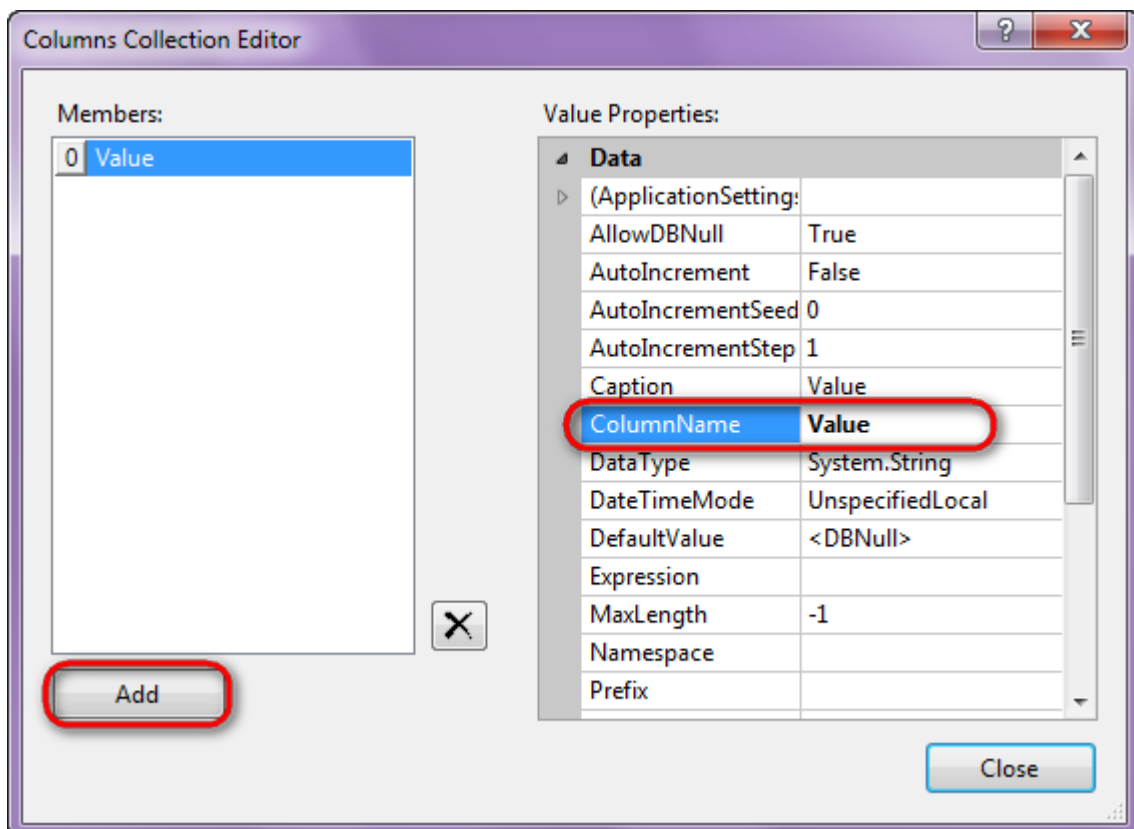
Click "Add" in order to add table. Set property TableName = Number.



Select Columns property, click button  in order to open property editor.

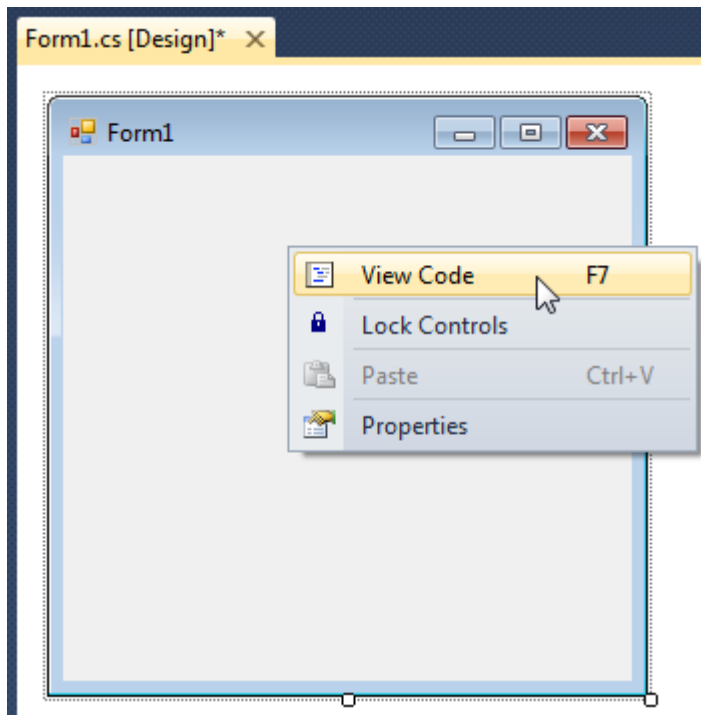
Step 5

Click "Add" to add a new column. Set ColumnName property to "Value".



Step 6

Right click on the form and select "View Code" in the context menu to view code.

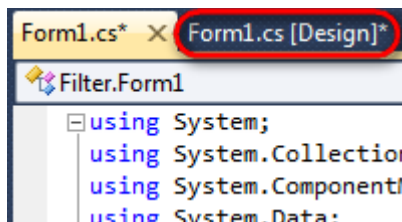


Add the following code to the class constructor in order to fill data source.

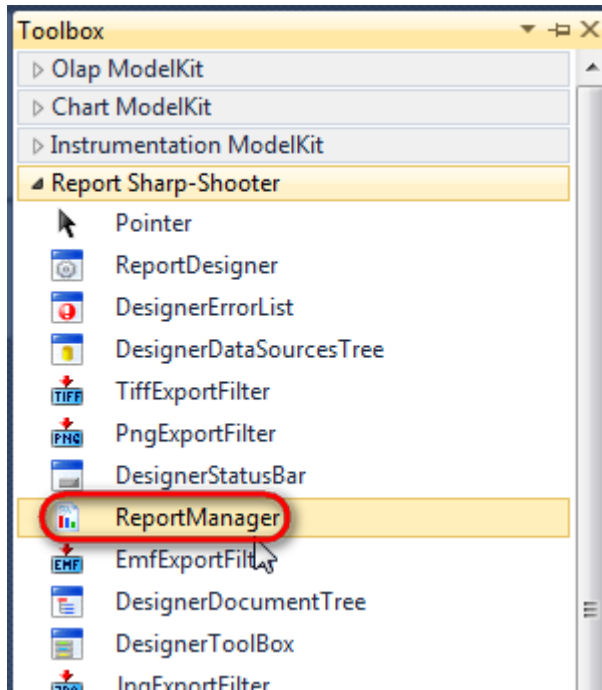
```
public Form1()
{
    InitializeComponent();
    DataRow row;
    for (int i = 0; i < 50; i++)
    {
        row = dataTable1.NewRow();
        row["Value"] = i;
        dataTable1.Rows.Add(row);
    }
}
```

Step 7

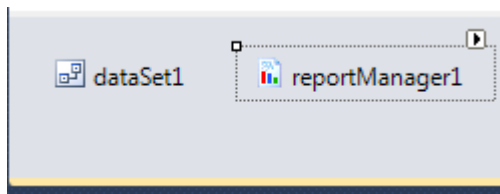
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

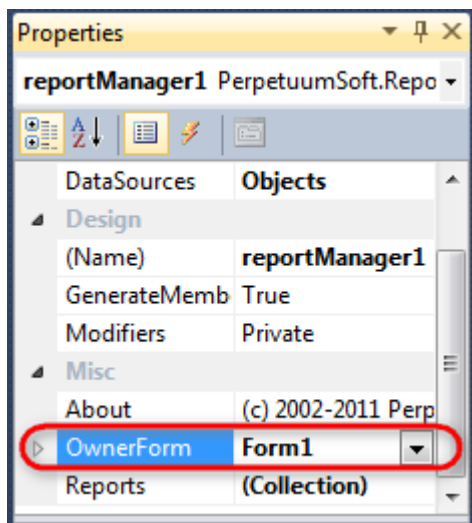


The component is available in the lower part of the window.



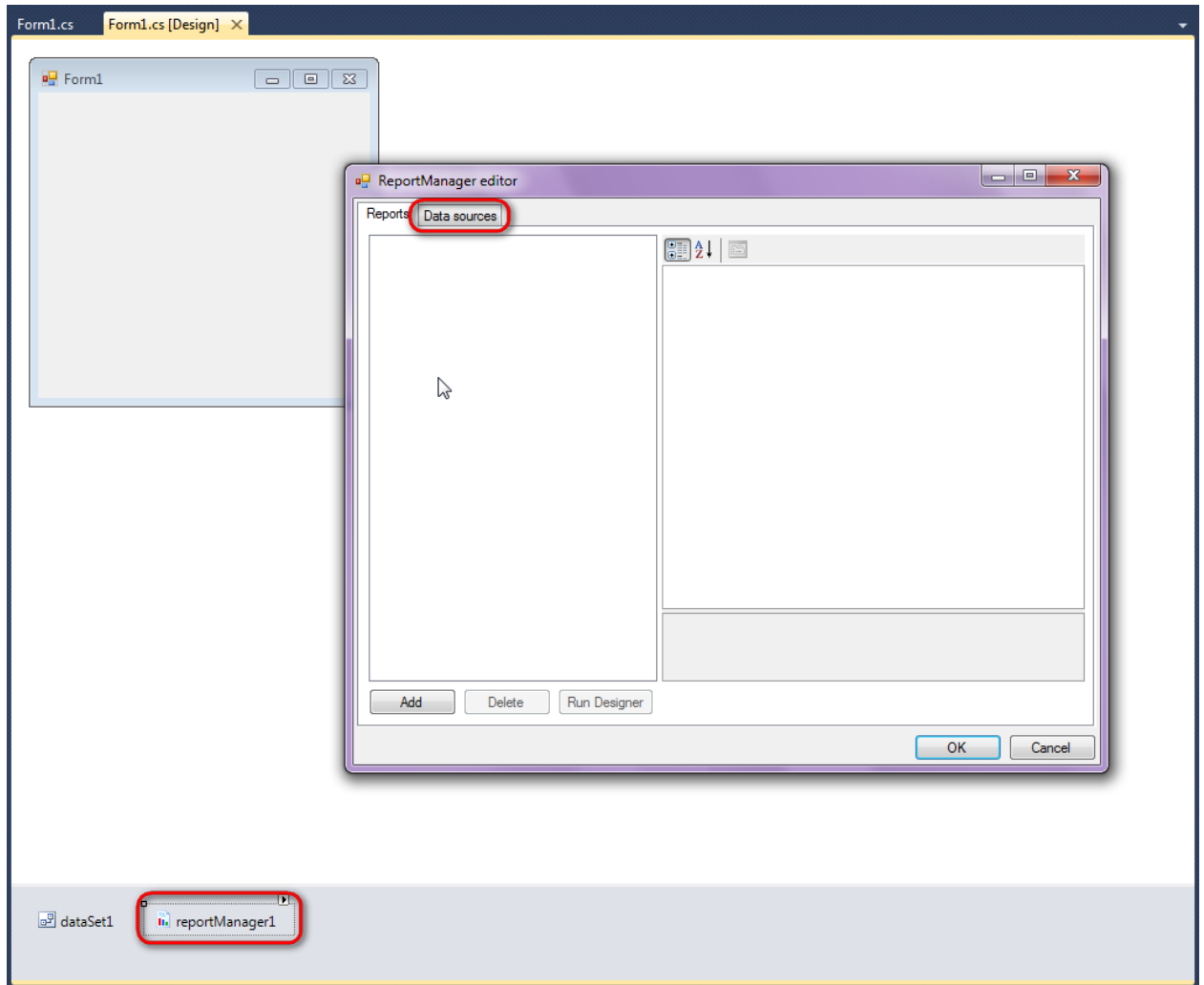
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

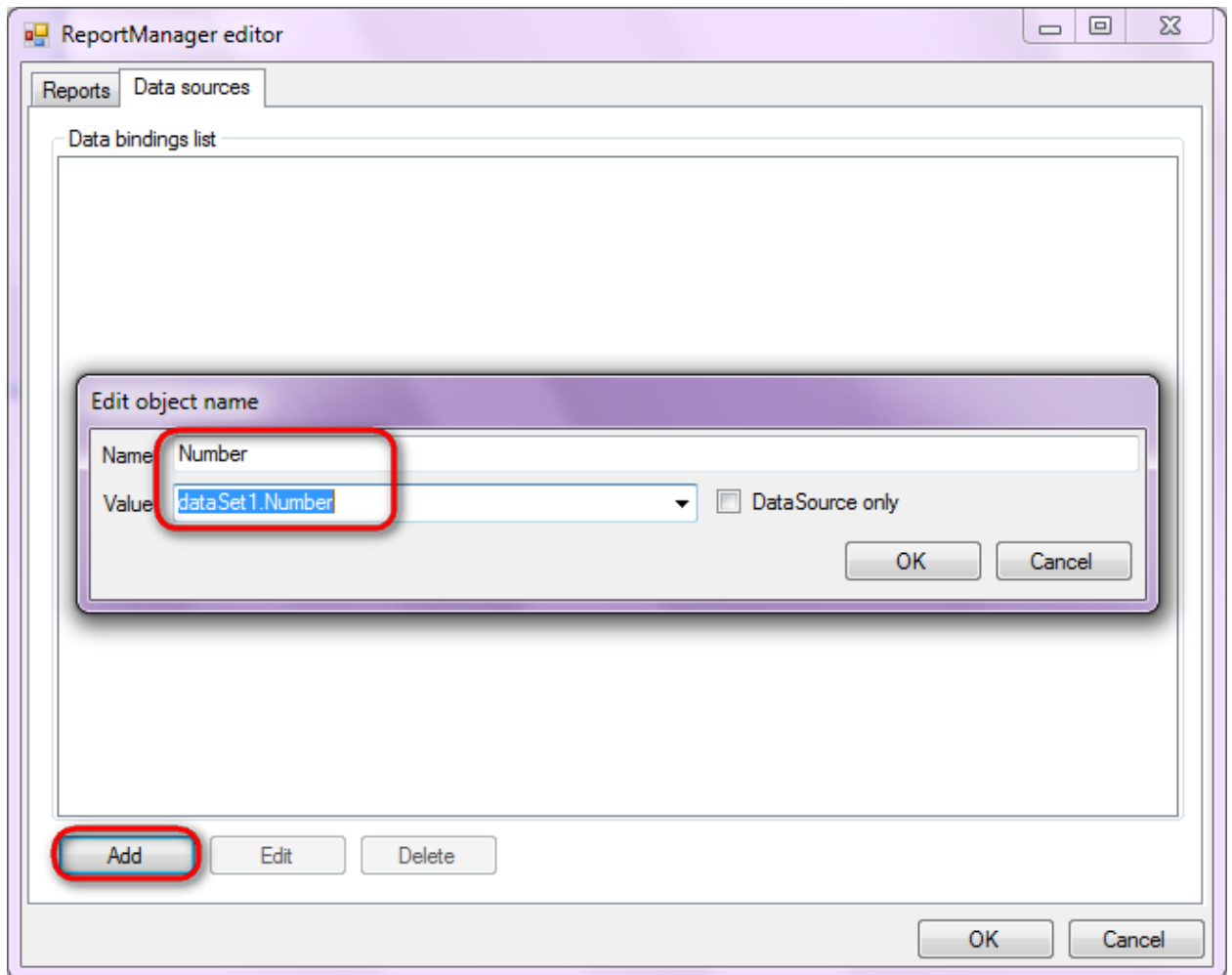


Step 9

Double click on ReportManager to open ReportManager editor.

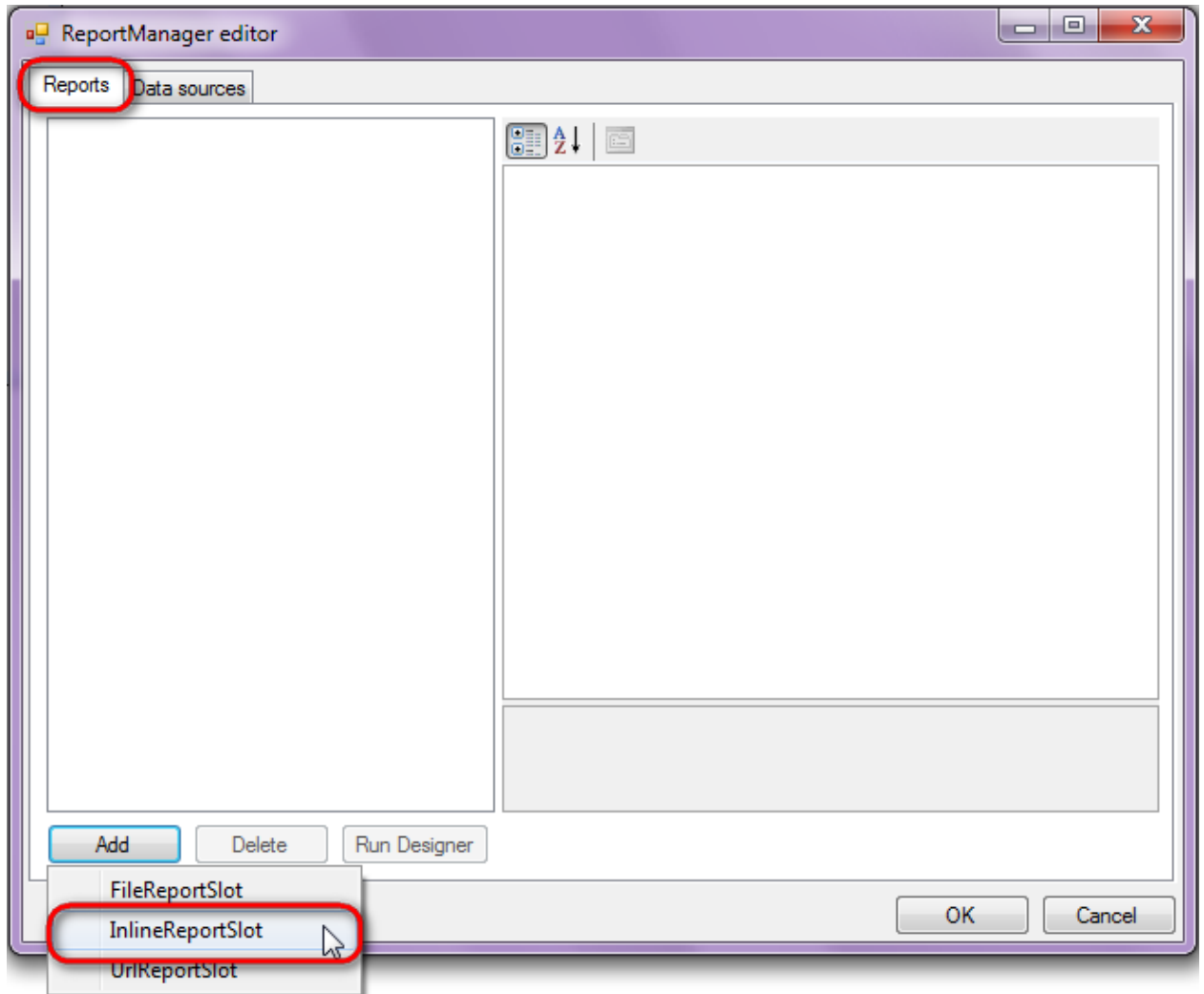


Go to "Data sources" tab, click "Add", and set data source name – "Number", select data source value – "dataSet1.Number".



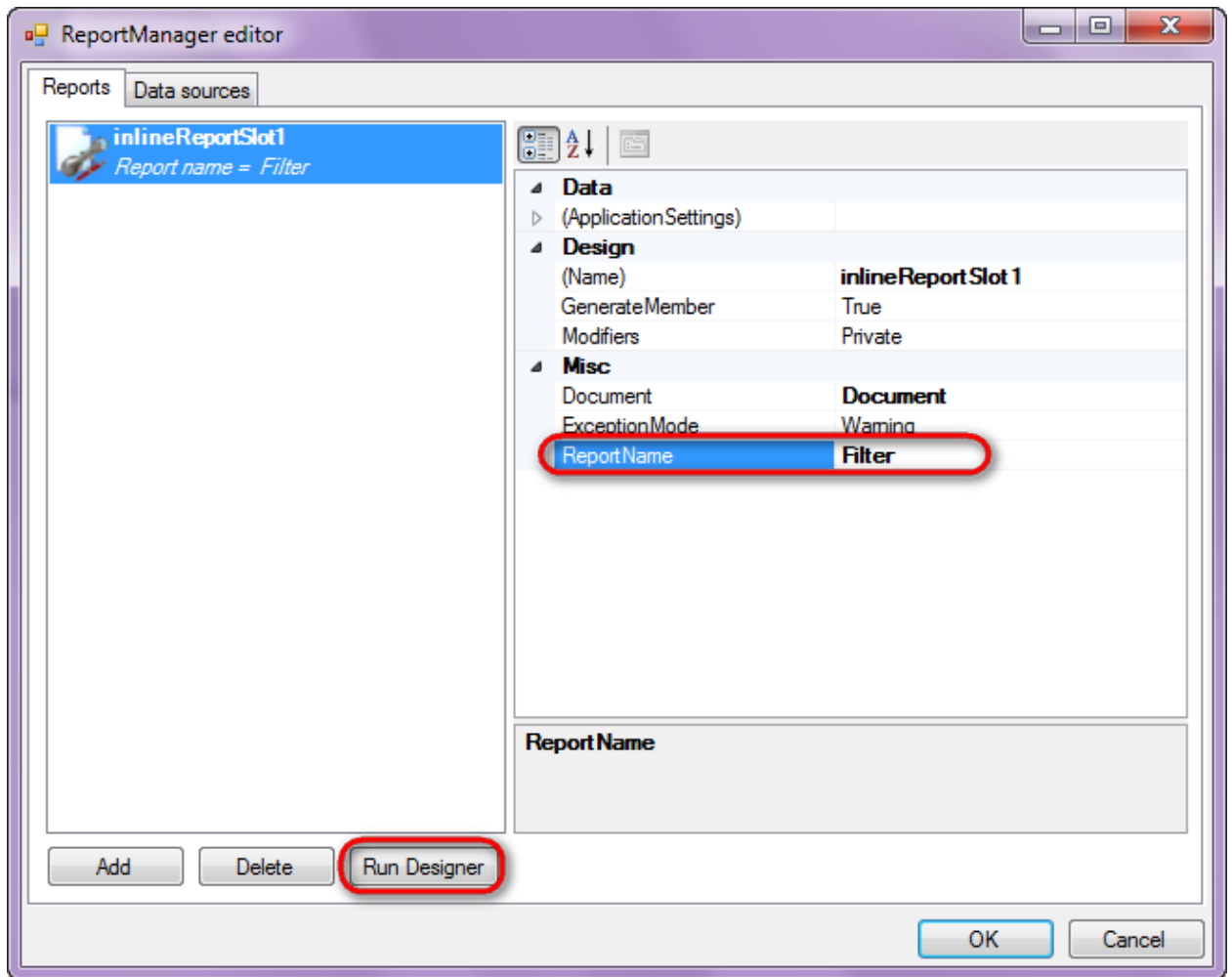
Step 10

Go to "Reports" tab, click "Add" and select "InlineReportSlot".



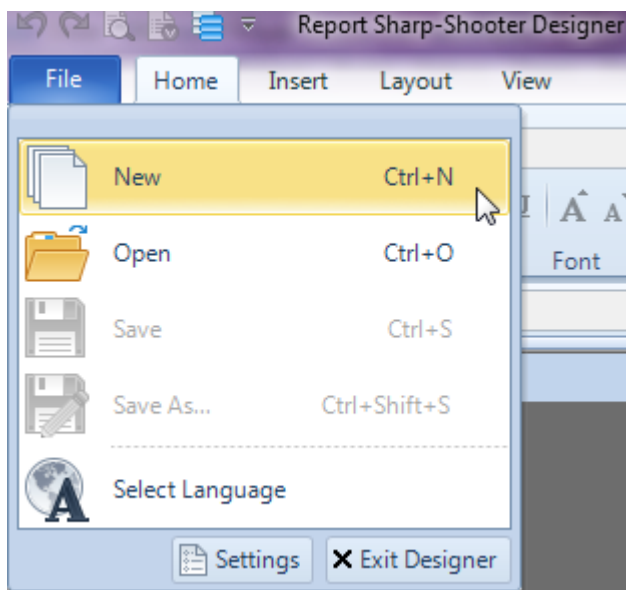
Step 11

Set name of the report in the property ReportName – “Filter”. Click “Run Designer” in order to open template editor - Report Designer.

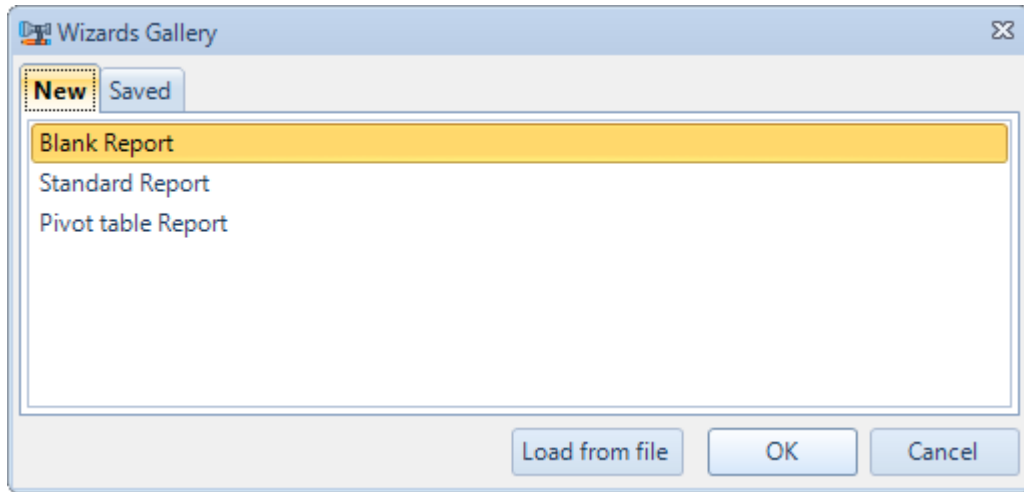


Step 12

Create new empty template – select File\New from the main menu.

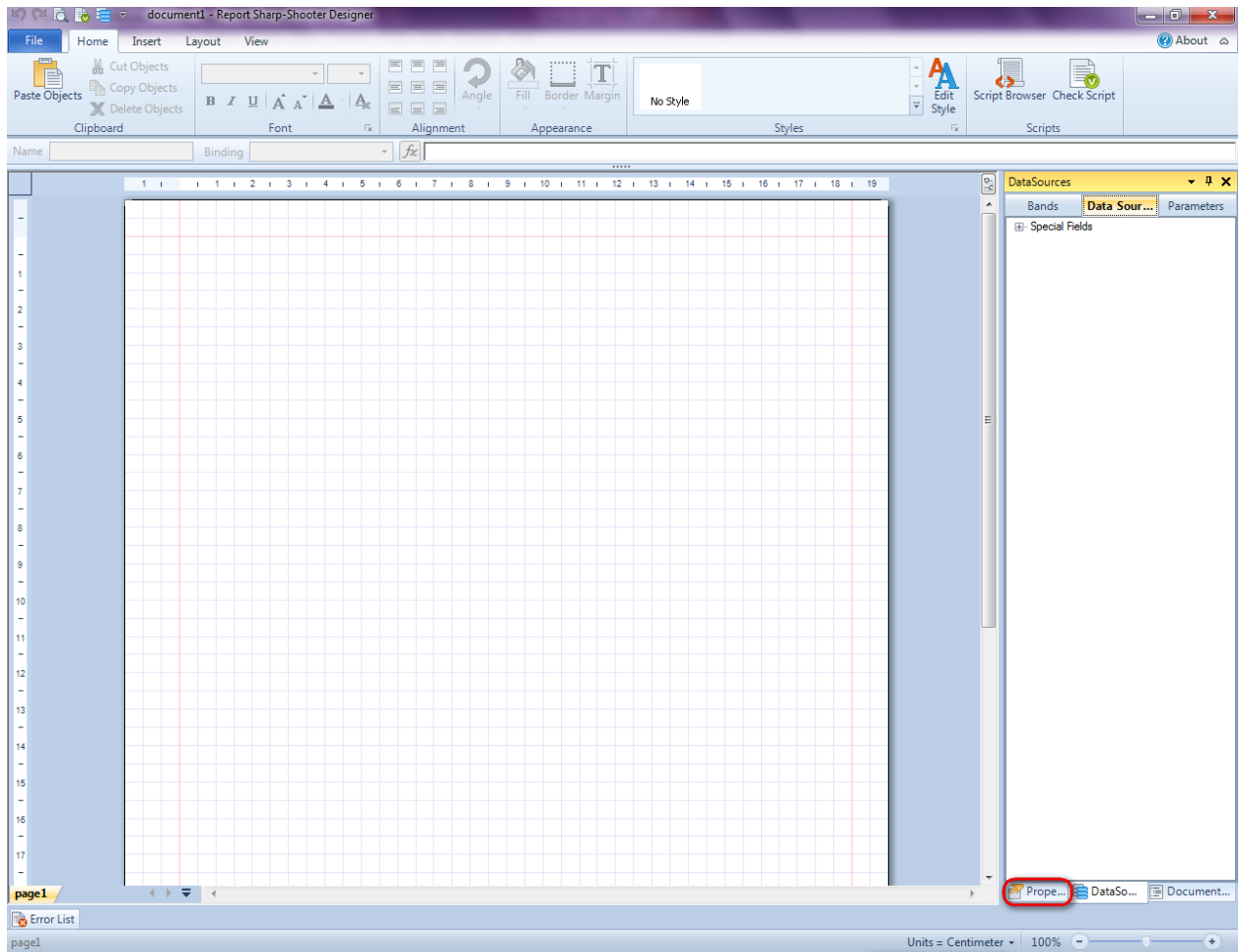


Select "Blank Report" in the Wizards Gallery and click "OK".

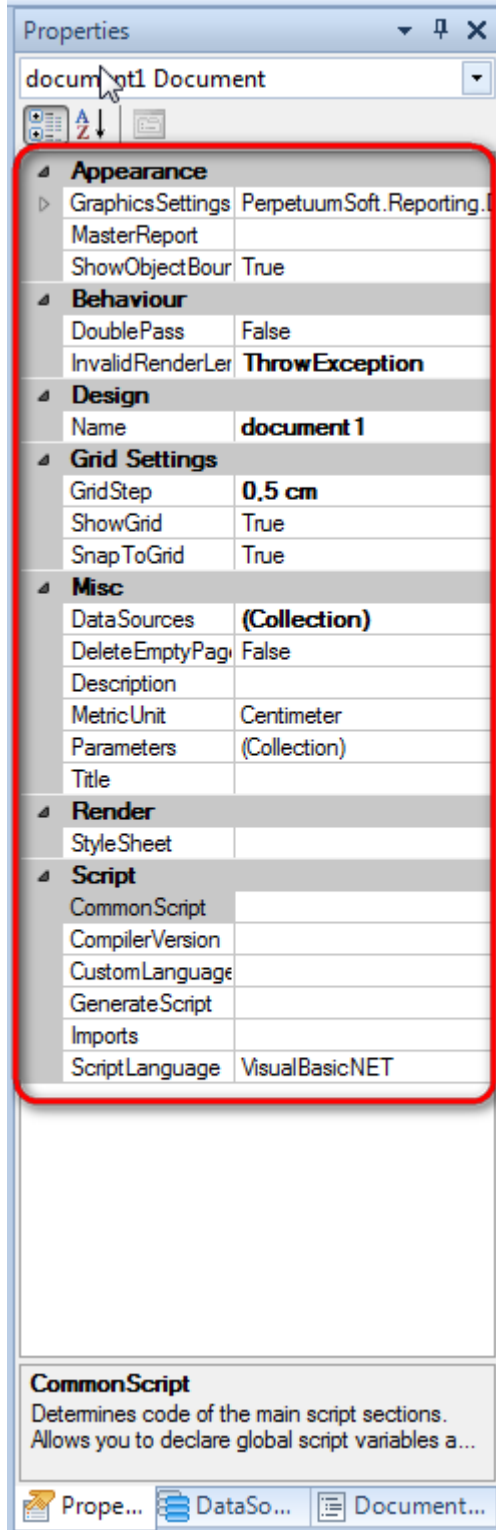


Step 13

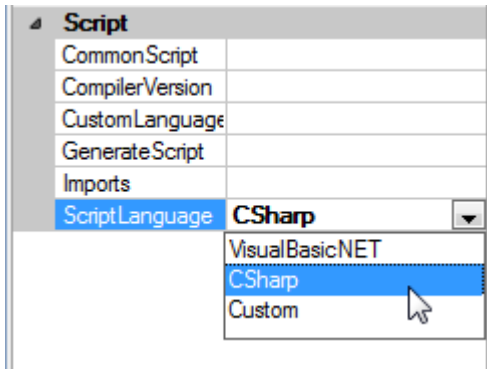
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

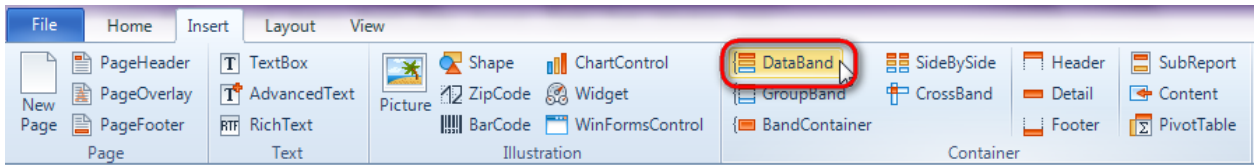


Set property ScriptLanguage = CSharp.



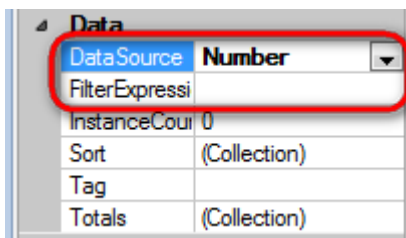
Step 14


Press "DataBand" button on the Insert tab in the group Container.

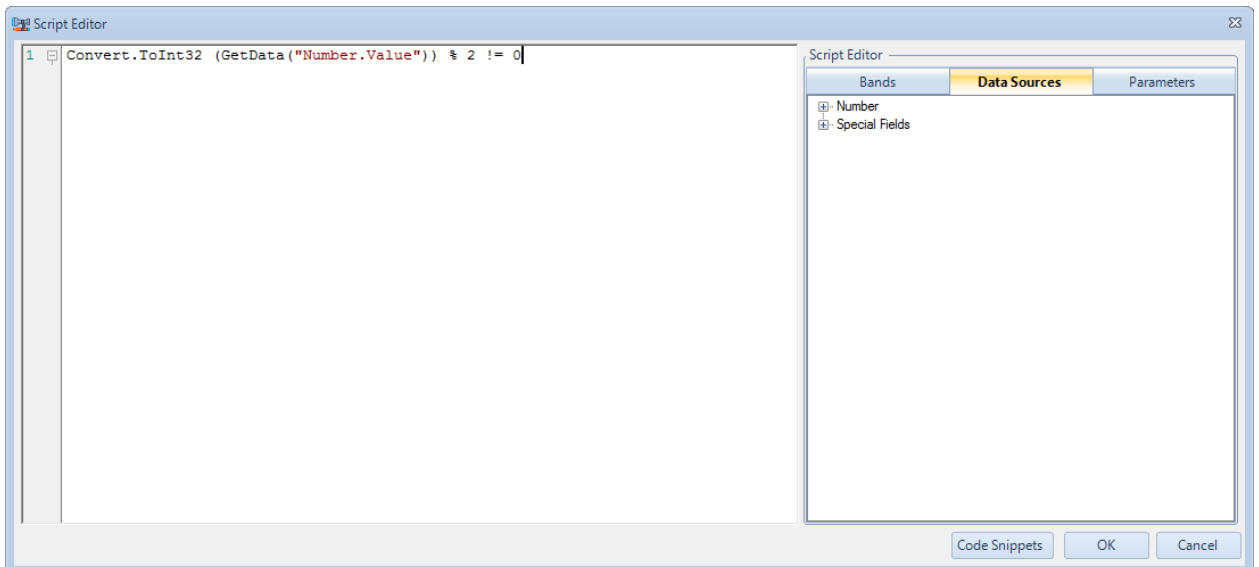


Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = Number.

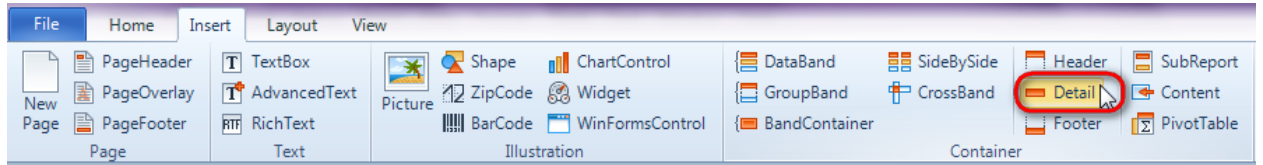


Select FilterExpression property, click button  to open Script Editor. Enter filtering condition to display only odd numbers: `Convert.ToInt32 (GetData("Number.Value")) % 2 != 0`



Step 15

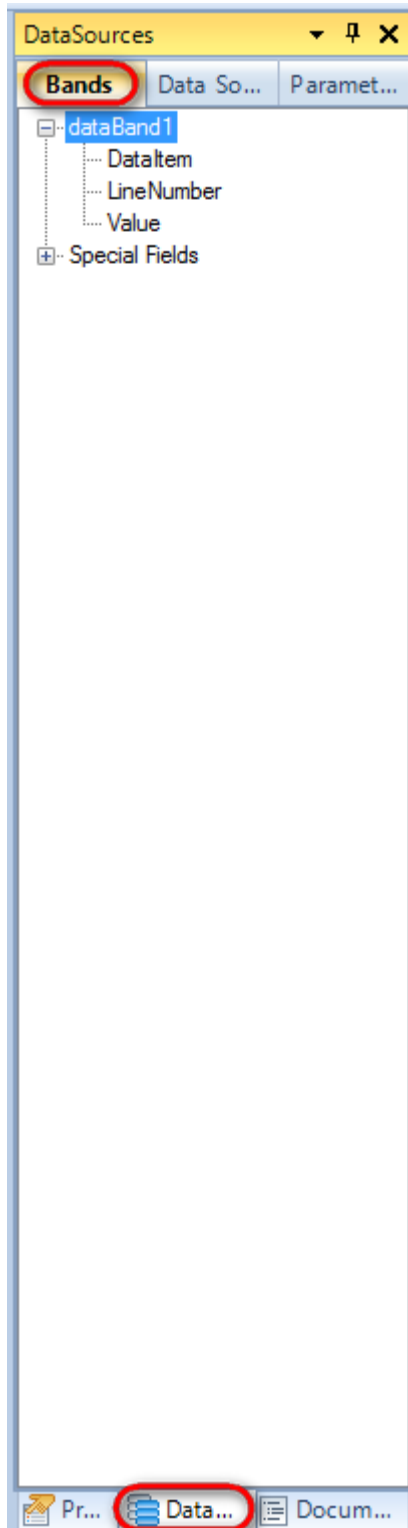
Press "Detail" button on the Insert tab in the group Container.



Click on the DataBand area to add Detail band inside DataBand.

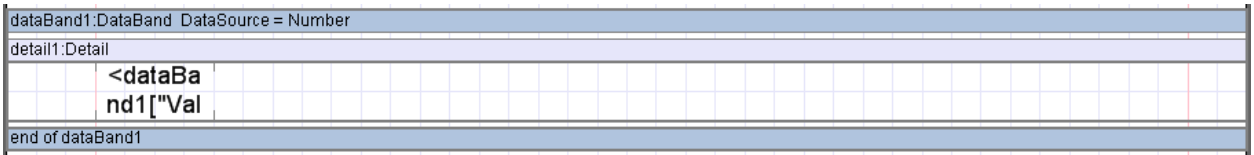
Step 16

Go to "DataSources" tab.





Drag and drop "Value" field from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.



Step 17

Save template, close Report Designer.

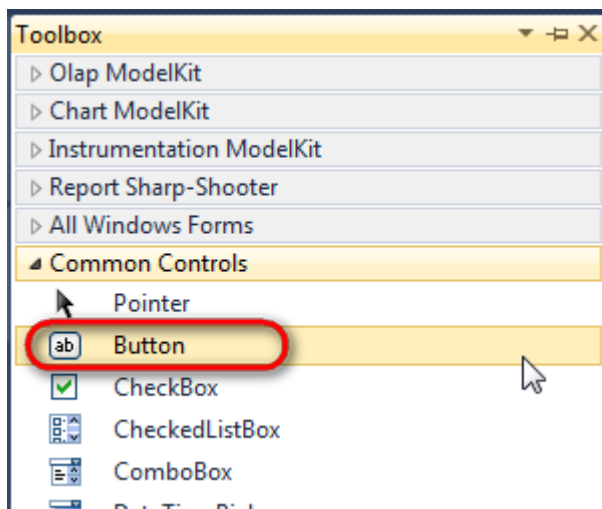
Step 18

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row;
    for (int i = 0; i < 50; i++)
    {
        row = dataTable1.NewRow ();
        row["Value"] = i;
        dataTable1.Rows.Add (row);
    }
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted (object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

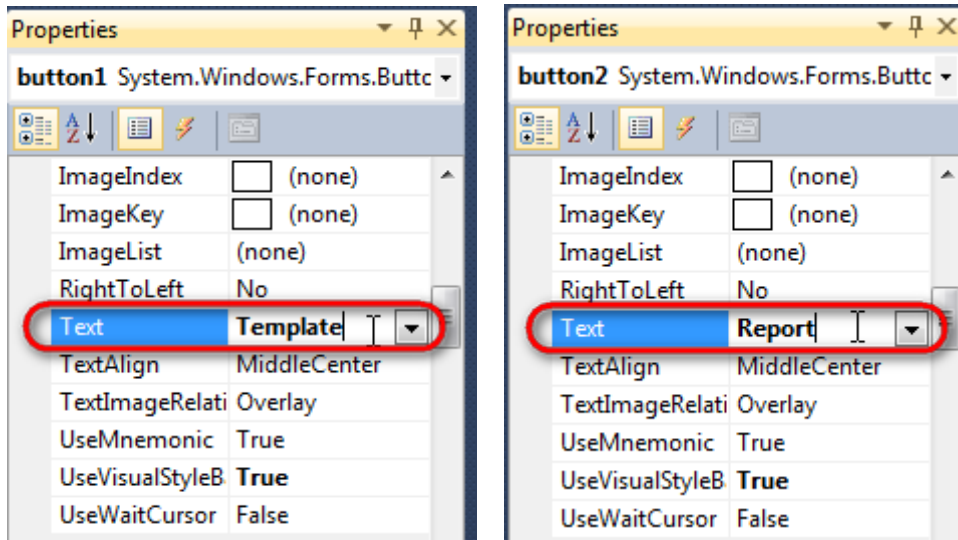
Step 19

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).





Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



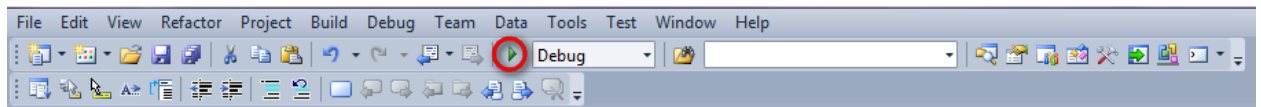
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

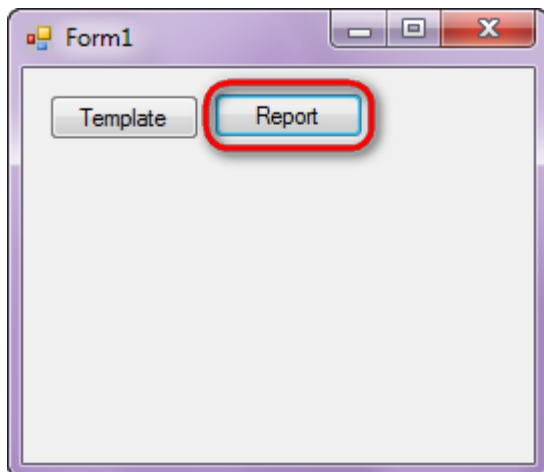
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 20

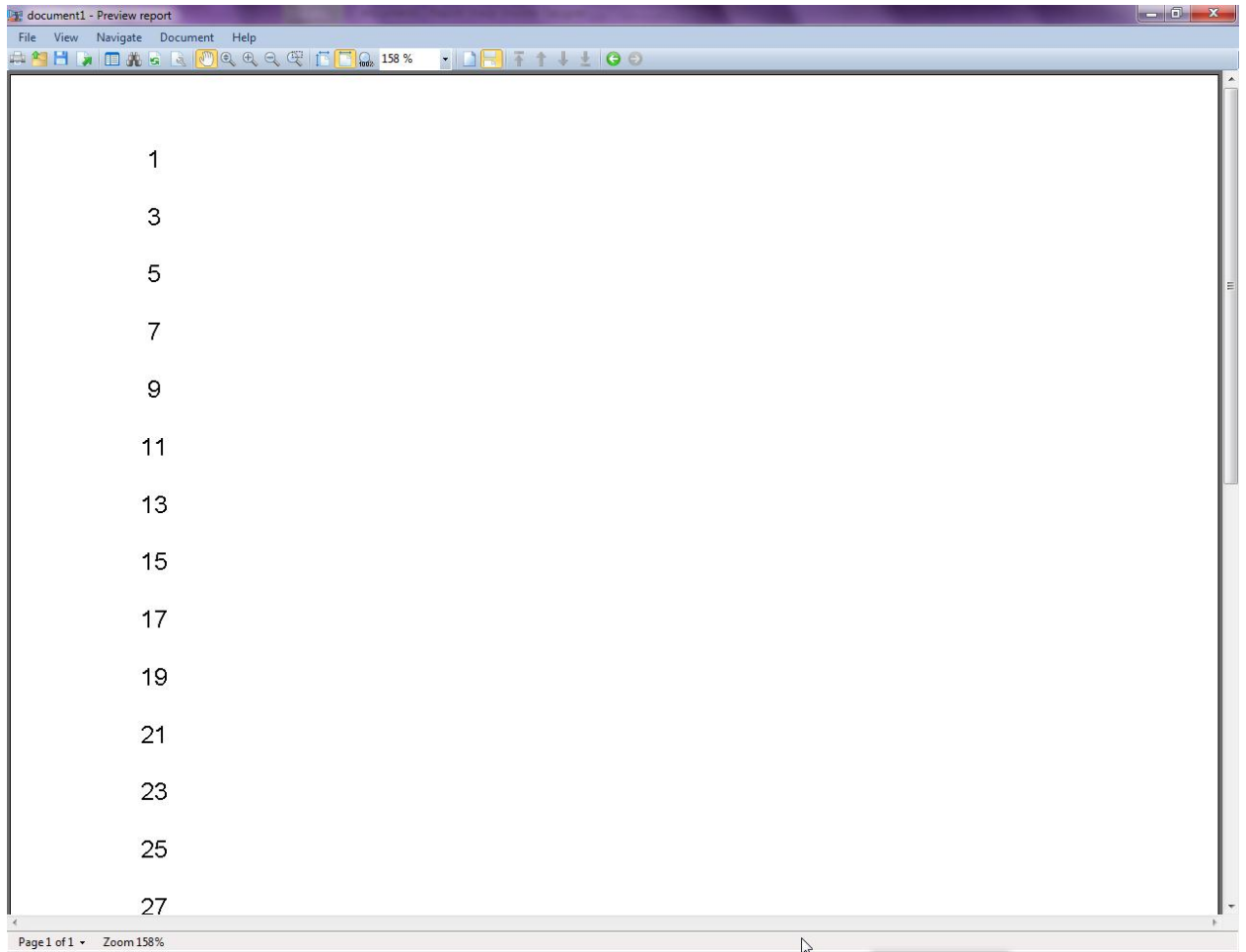
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



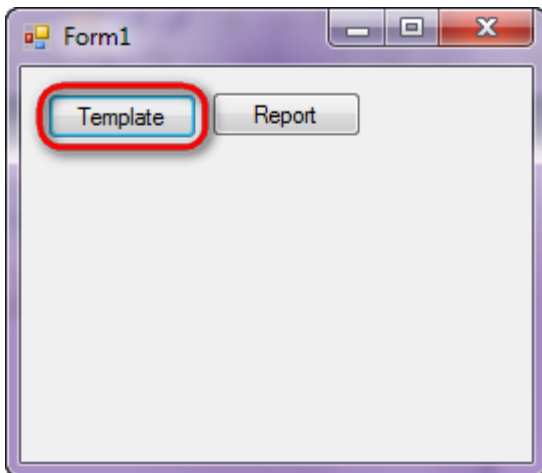
Click the “Report” button in the opened application window.



Generated report will open with Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



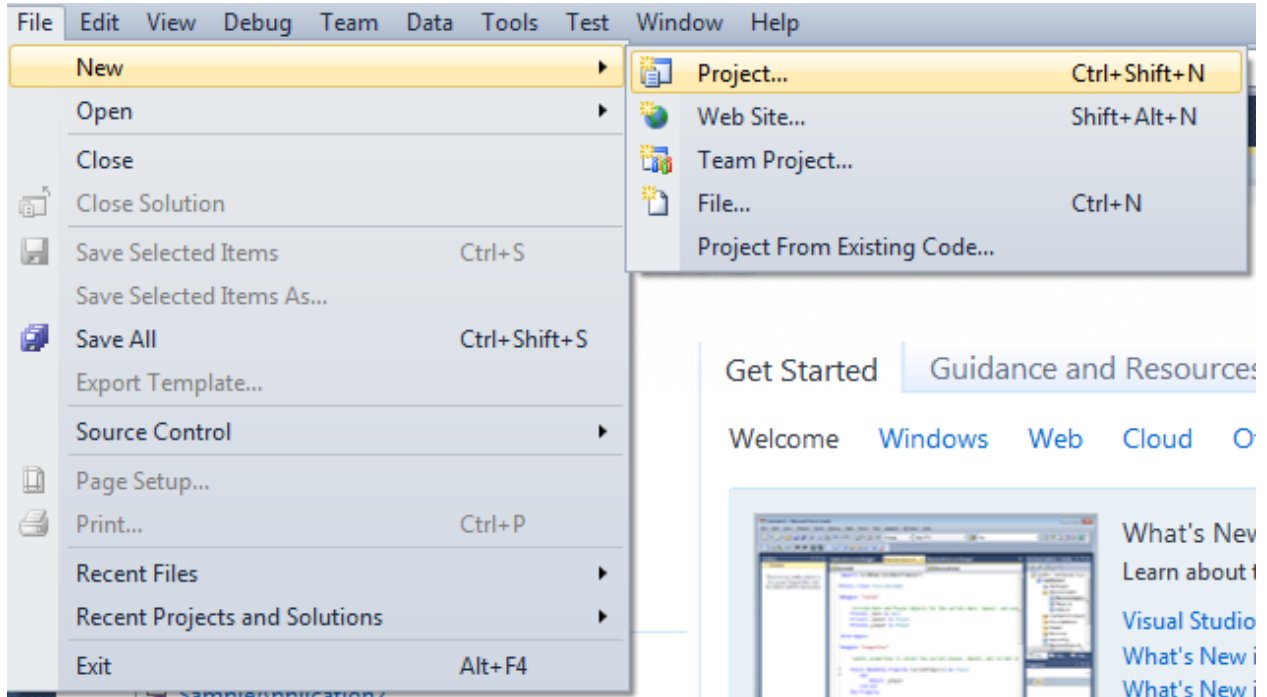


Grouping

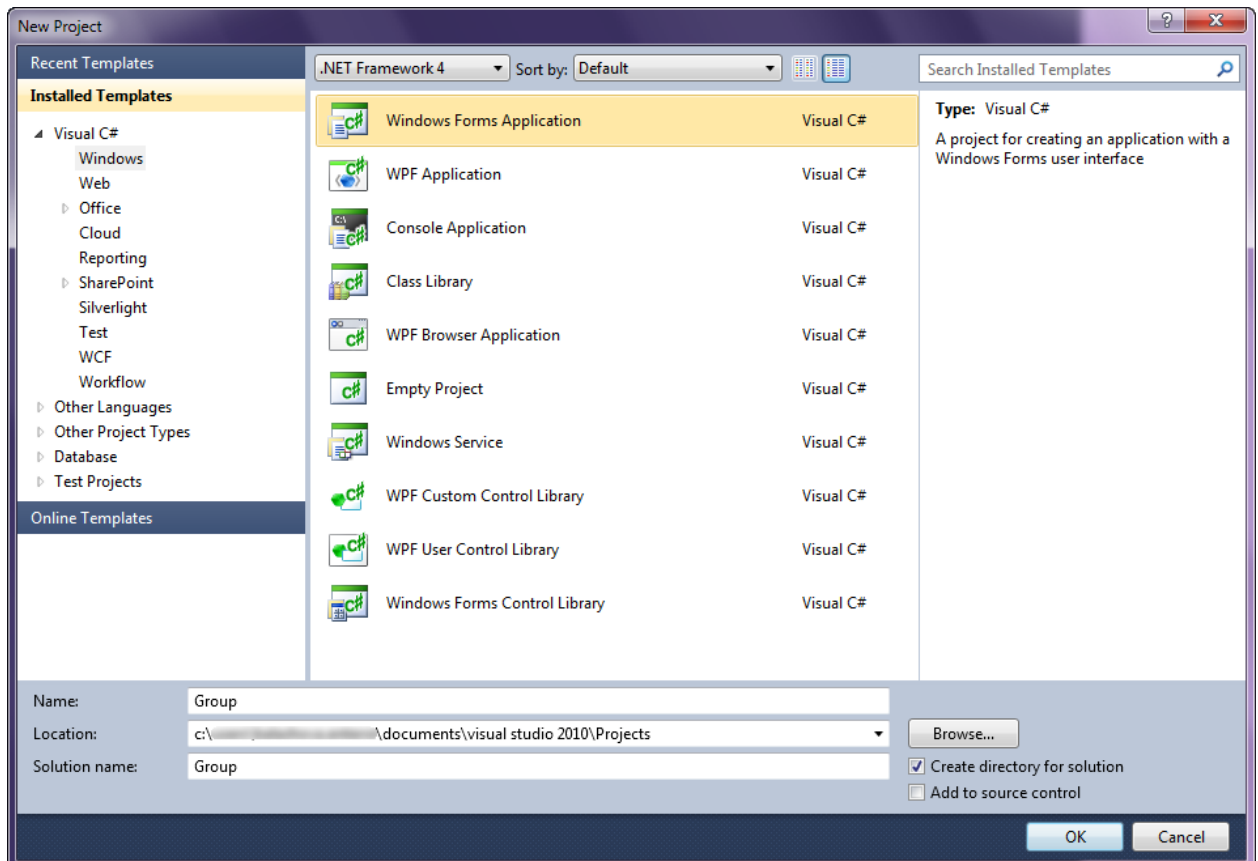
Template of a report containing information on customers grouped by the first letter of the company name.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

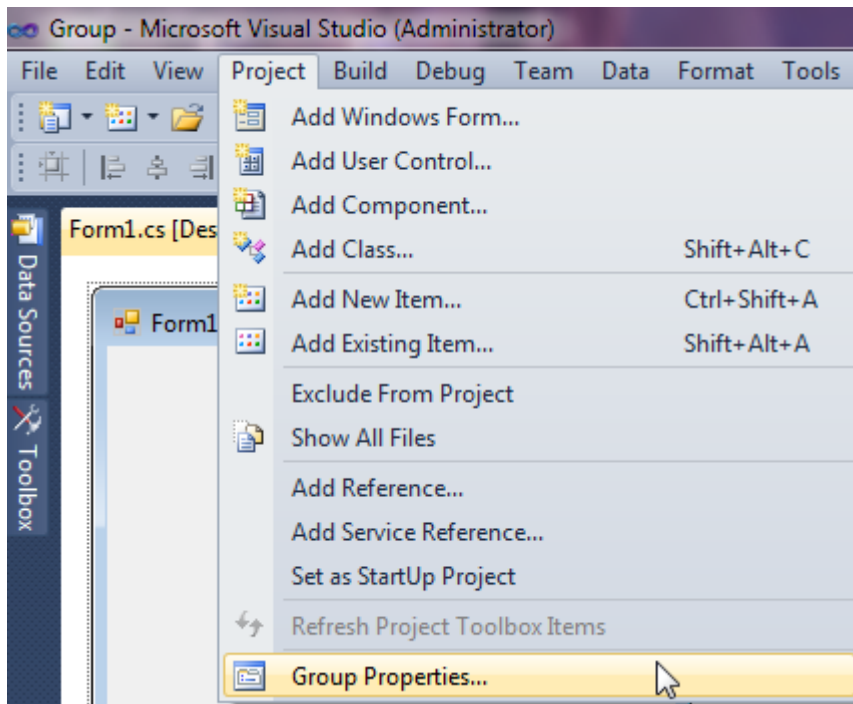


Select Windows Forms Application, set project name – “Group”, set directory to save the project to.

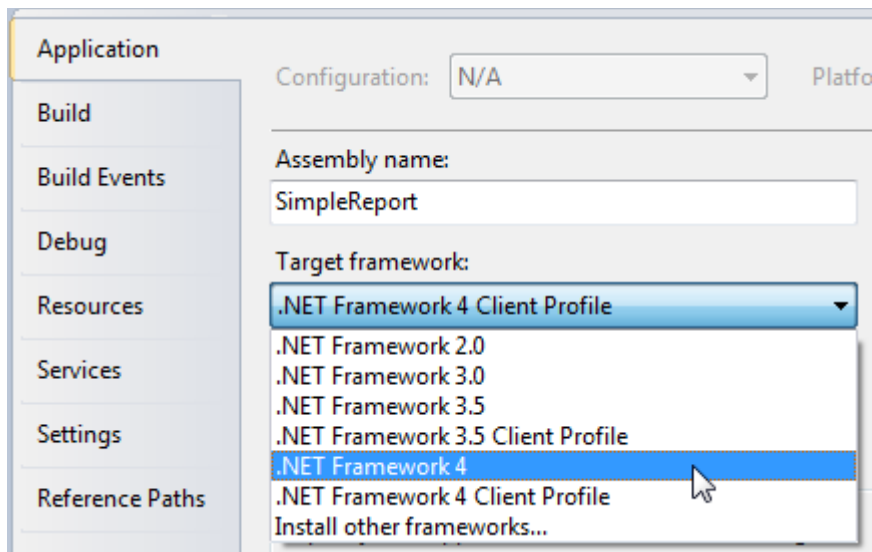


Step 2

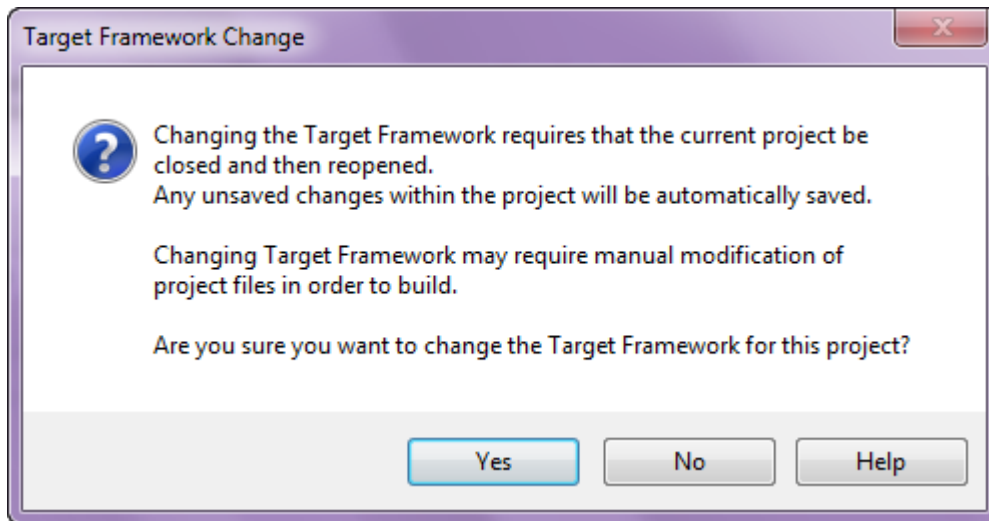
Change the project properties. Select the Project\Group Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

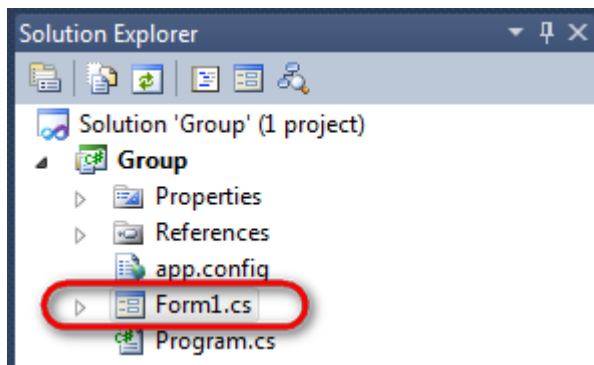


Press the "Yes" button in the opened window.

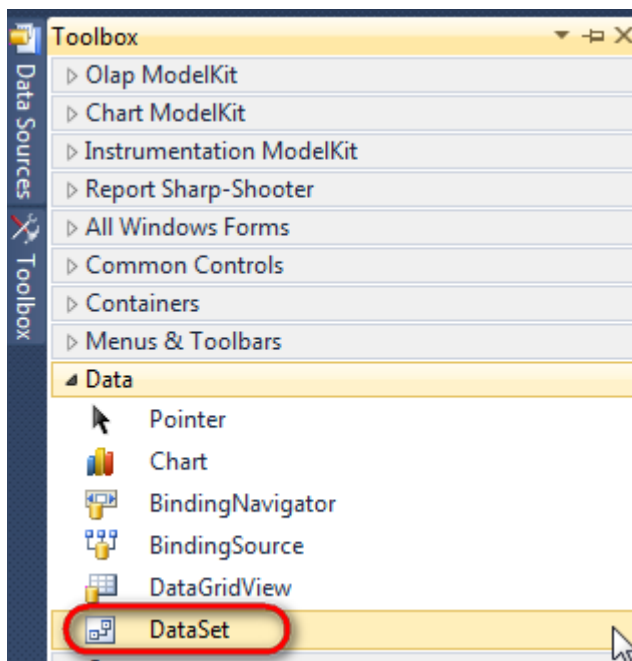


Step 3

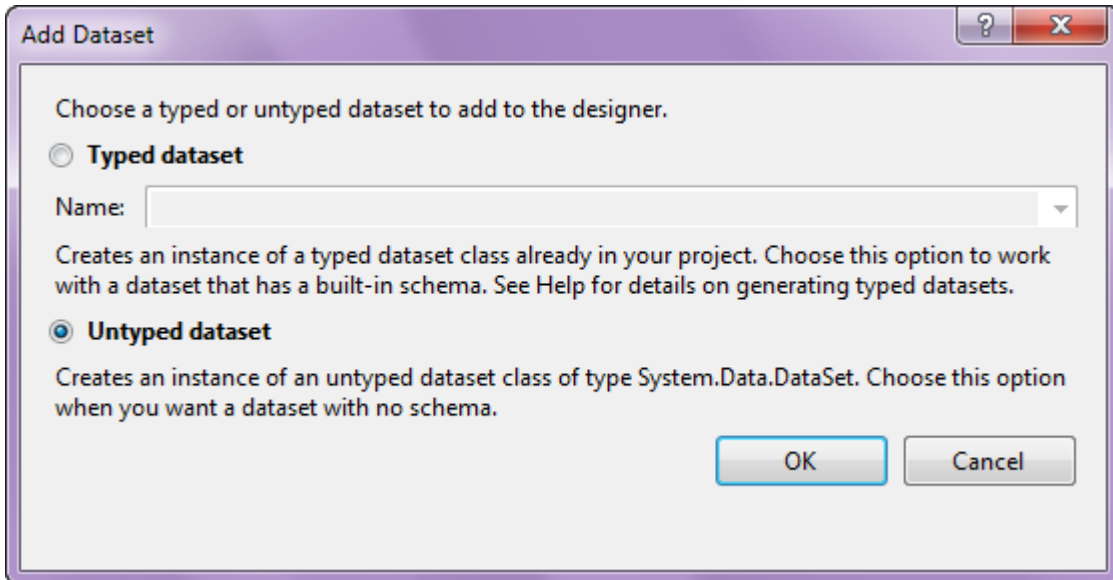
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



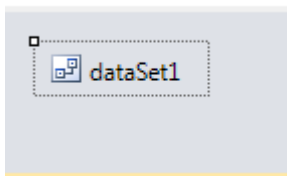
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

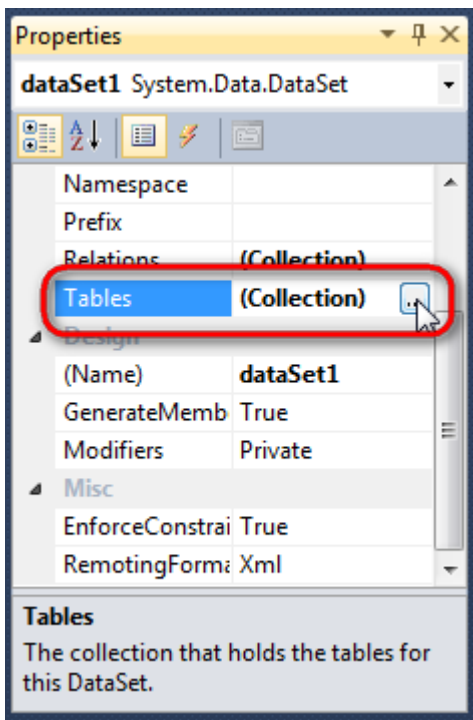


The component is available in the lower part of the window.

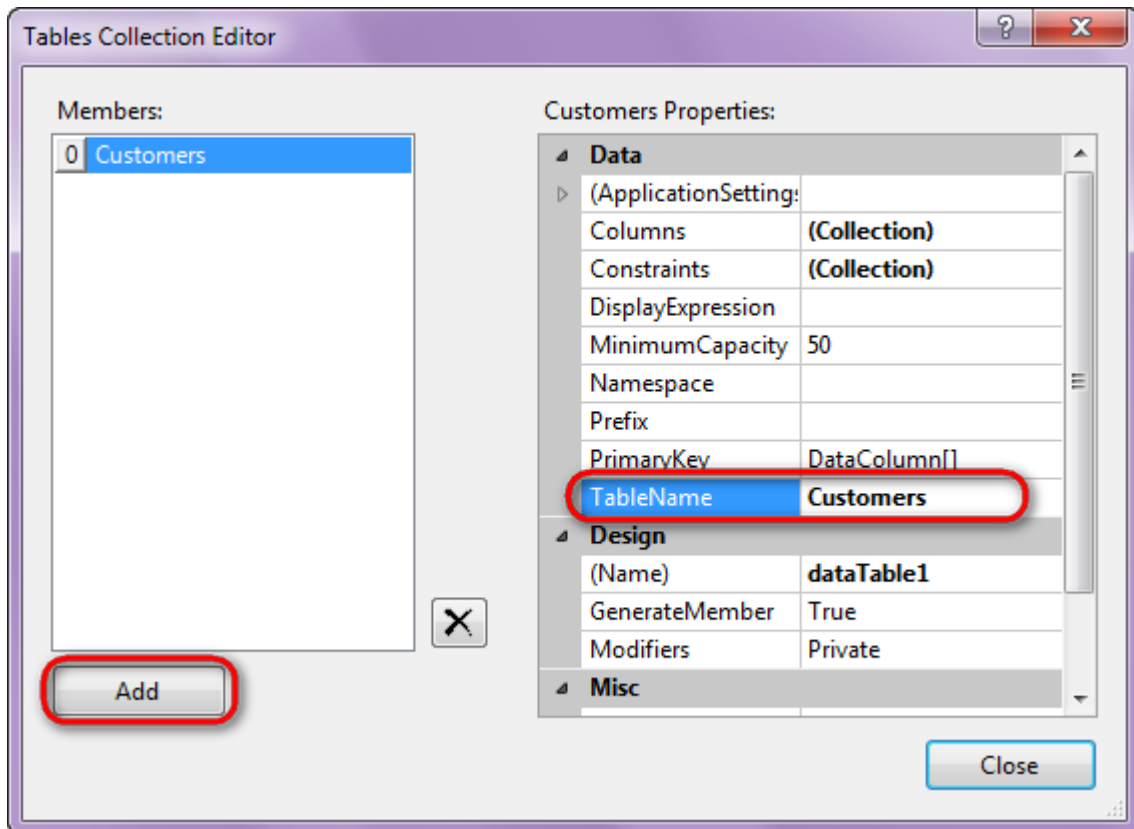


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

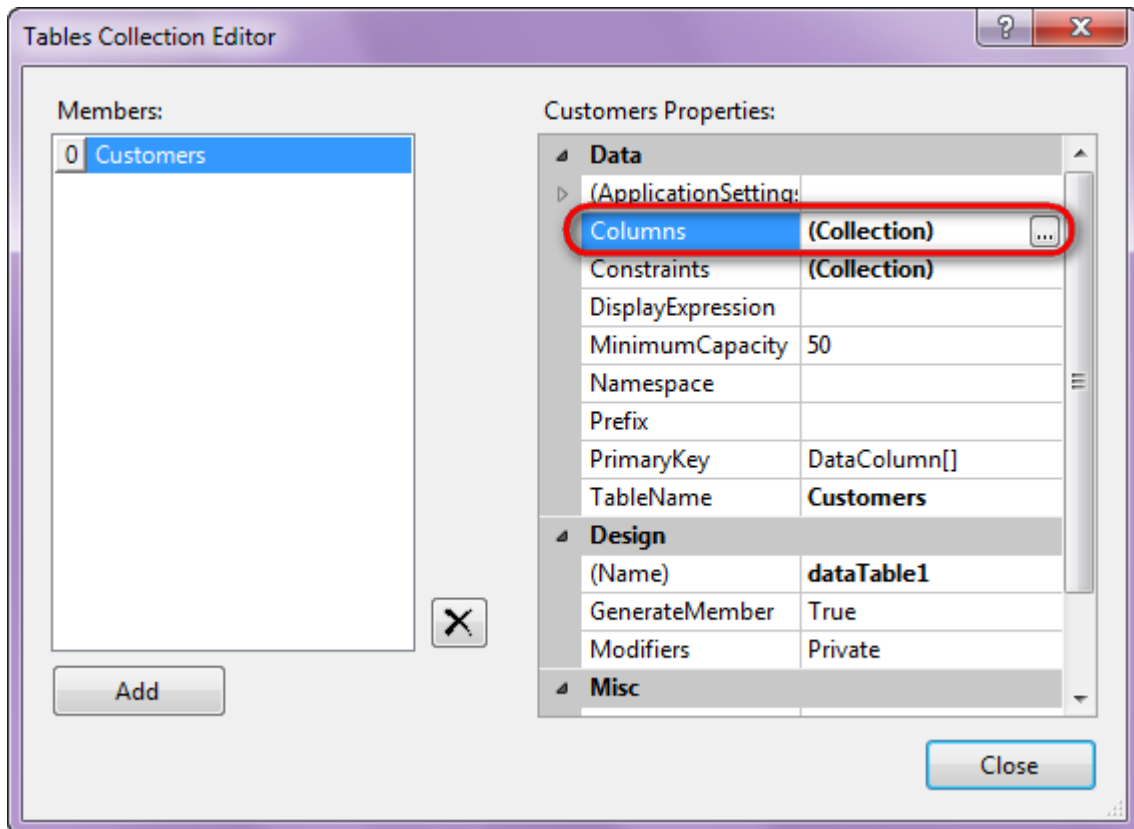


Click "Add" in order to add table. Set property TableName = Customers.

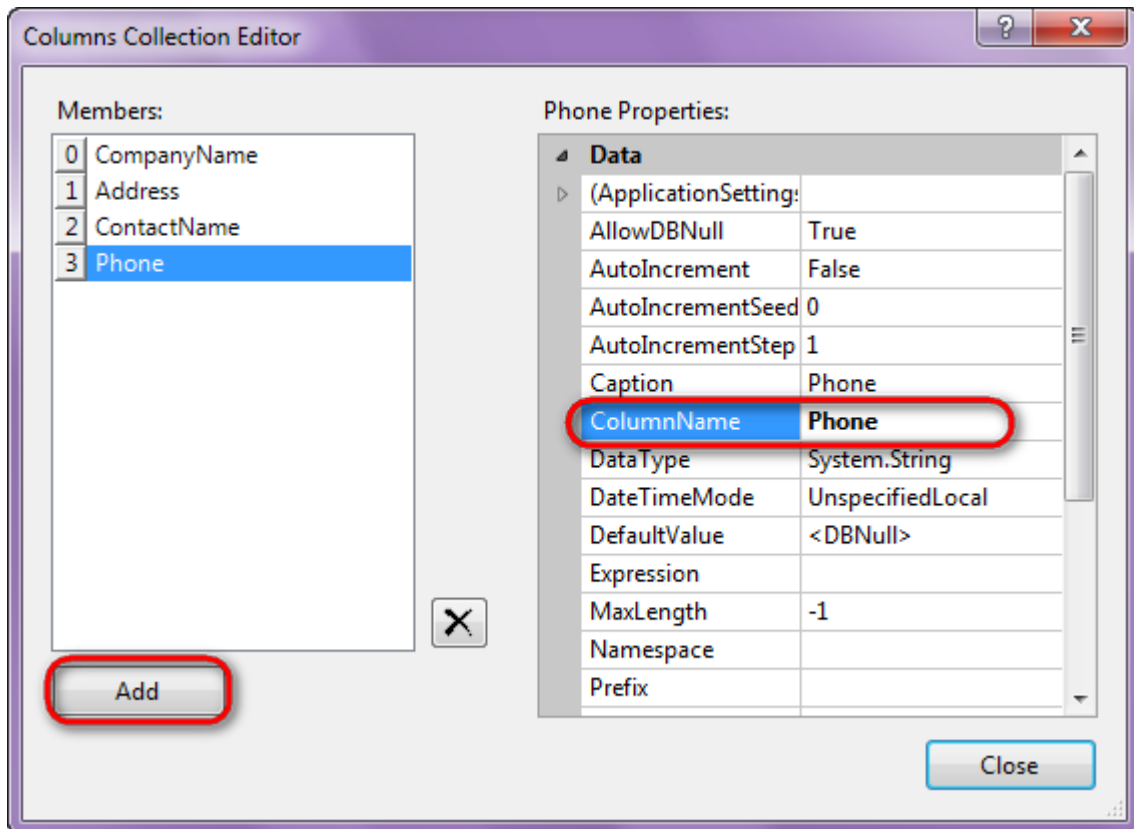


Step 5

Select Columns property, click button  in order to open property editor.

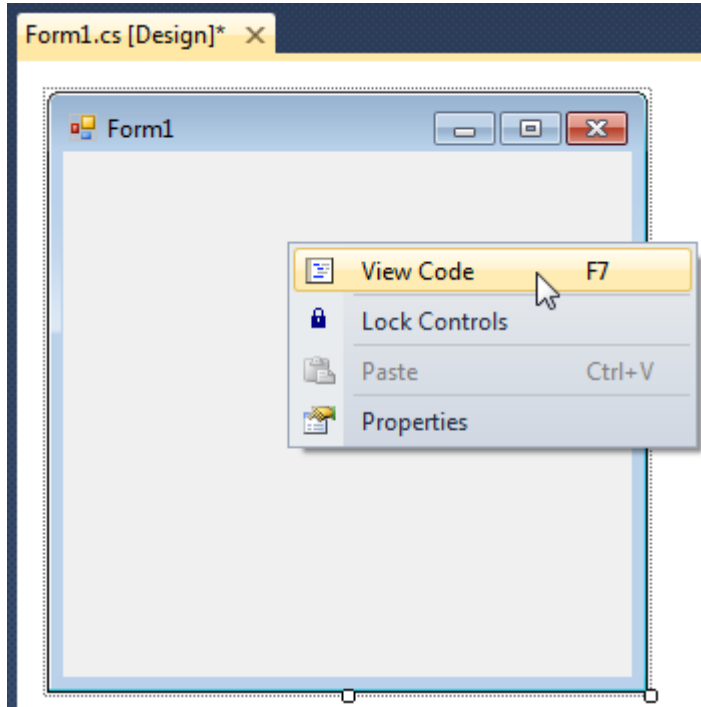


Click "Add" to add a new column. Add four columns. Set ColumnName property to "CompanyName", "Address", "ContactName", "Phone".



Step 6

Right click on the form and select "View Code" in the context menu to view code.



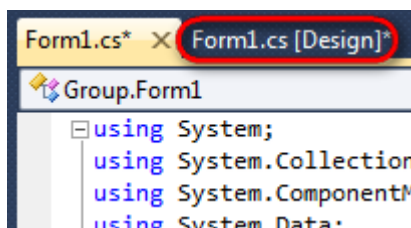
Add the following code to the class constructor in order to fill data source.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
}
```

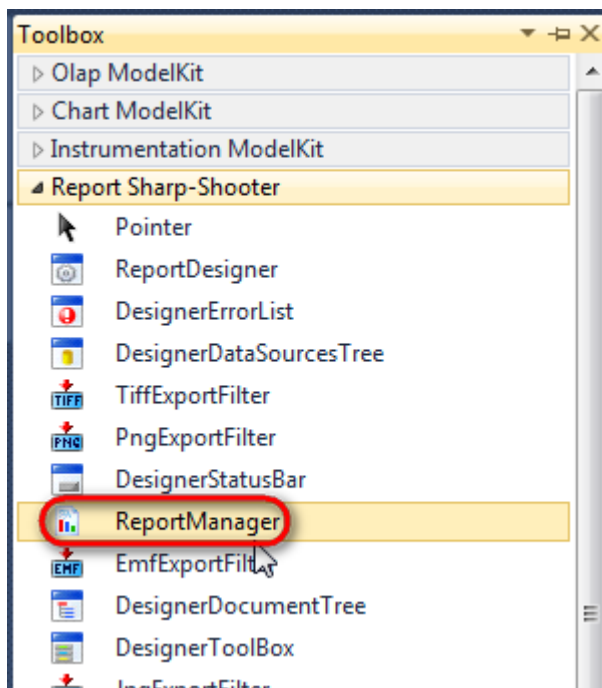
```
row["Address"] = "Obere Str. 57";  
row["ContactName"] = "Maria Anders";  
row["Phone"] = "030-0074321";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["Address"] = "Avda. de la Constitución 2222";  
row["ContactName"] = "Ana Trujillo";  
row["Phone"] = "(5) 555-4729";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["Address"] = "Kirchgasse 6";  
row["ContactName"] = "Roland Mendel";  
row["Phone"] = "7675-3425";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["Address"] = "Luisenstr. 48";  
row["ContactName"] = "Karin Josephs";  
row["Phone"] = "0251-031259";  
dataTable1.Rows.Add(row);  
}
```

Step 7

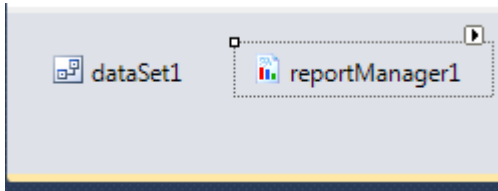
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

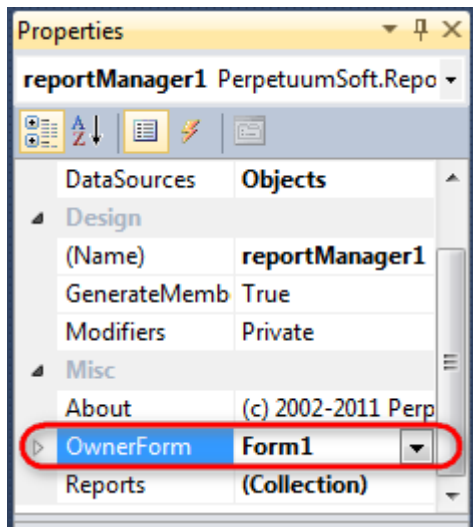


The component is available in the lower part of the window.



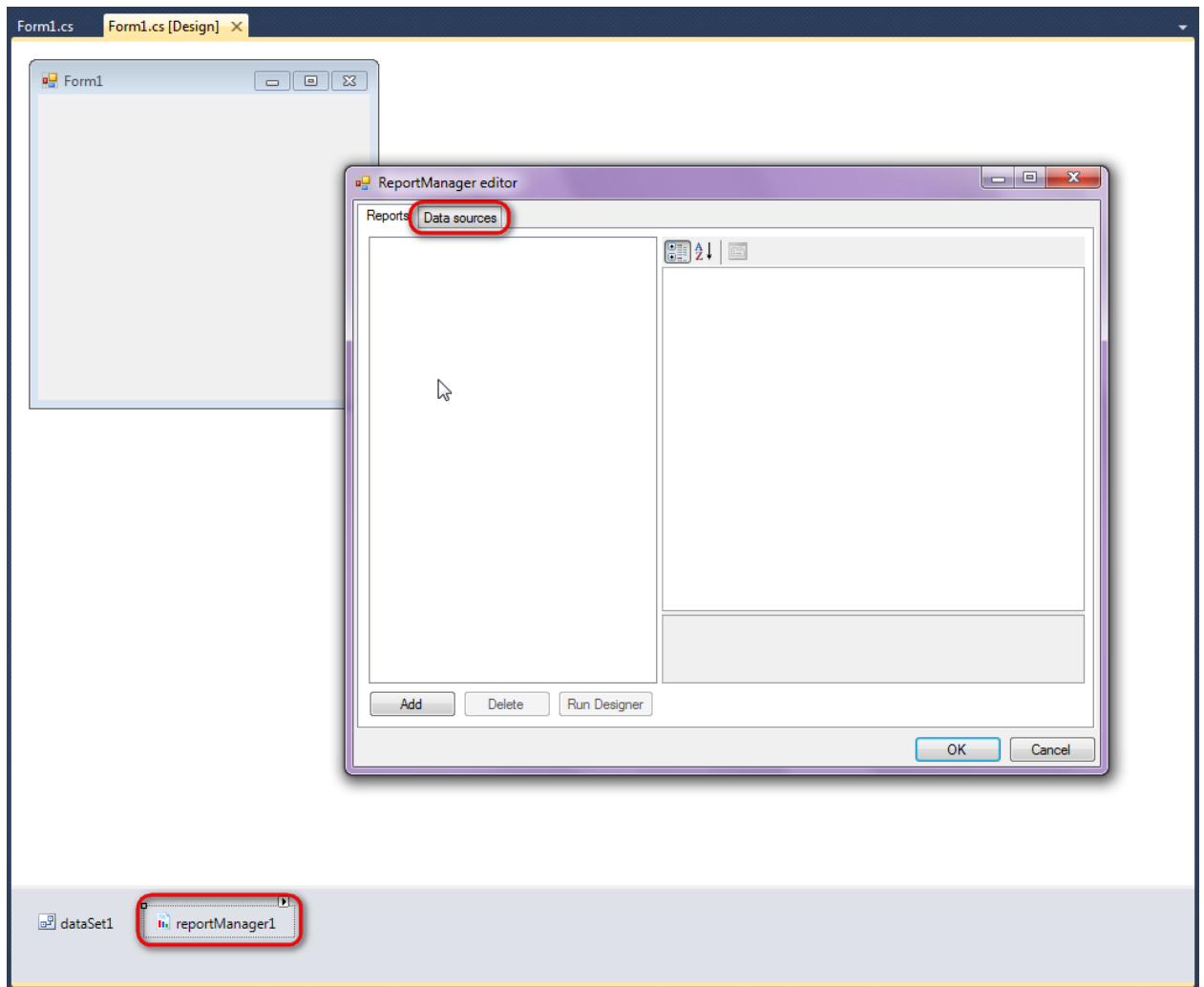
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

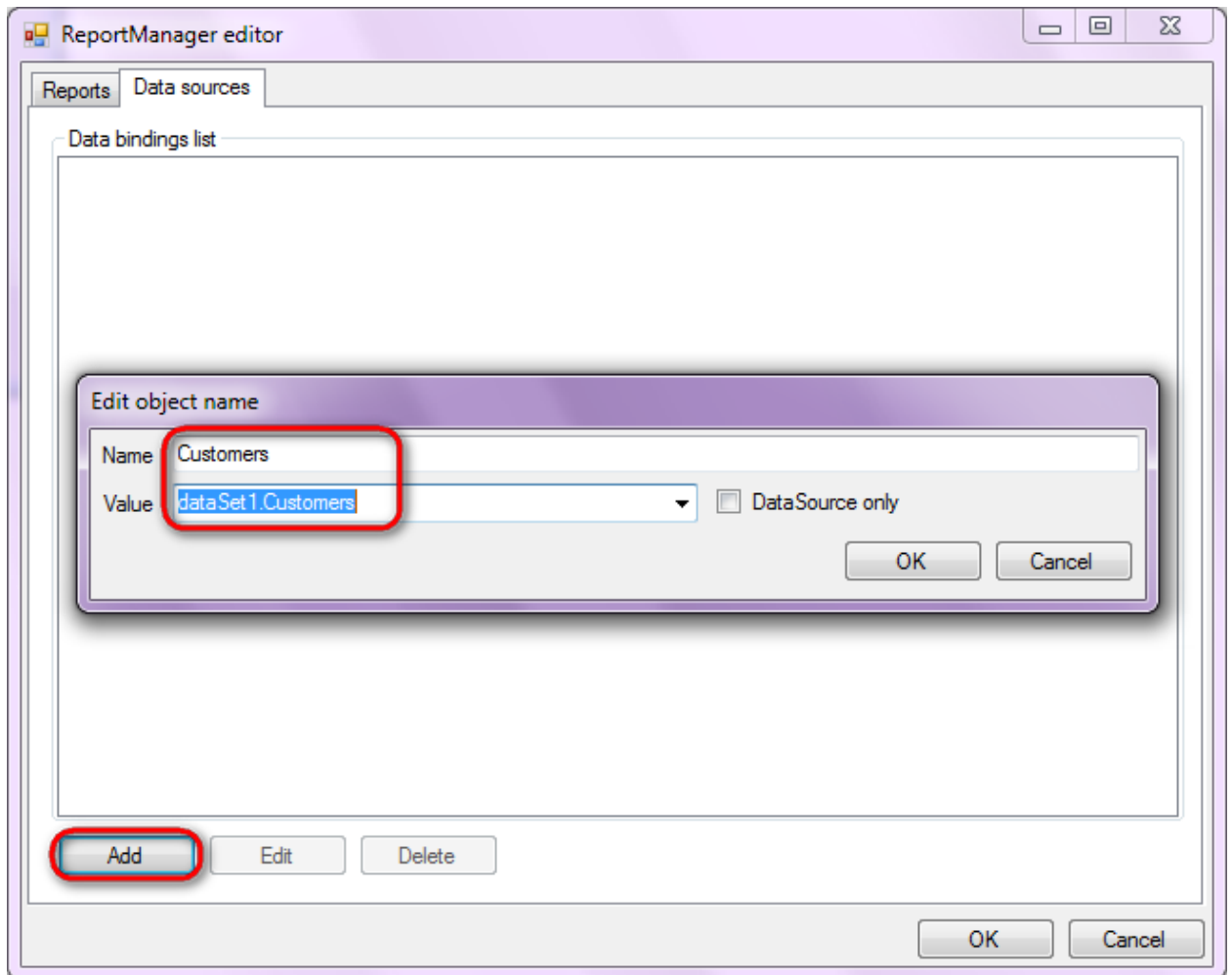


Step 9

Double click on ReportManager to open ReportManager editor.

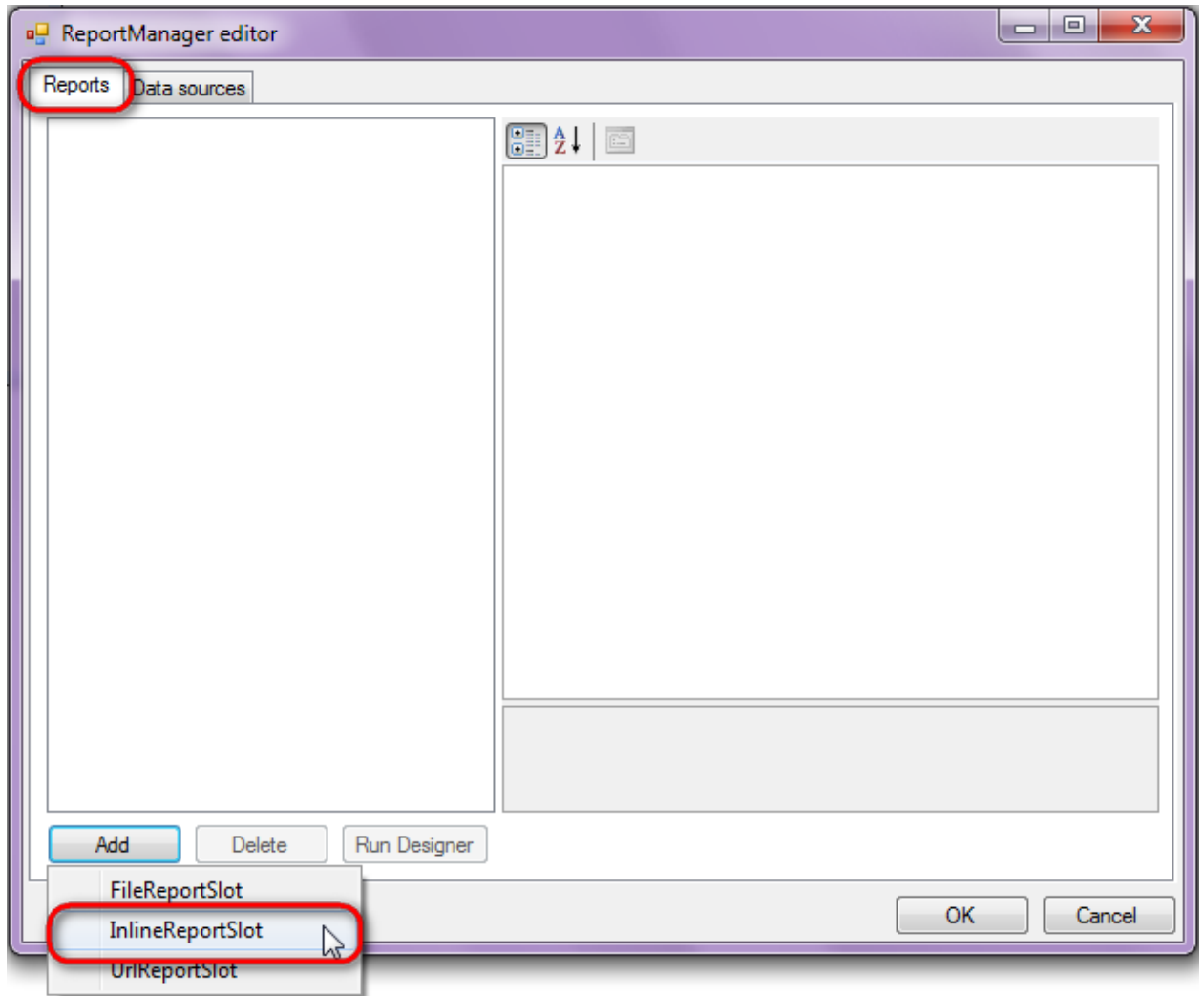


Go to "Data sources" tab, click "Add", set data source name – "Customers", select data source value – "dataSet1.Customers".



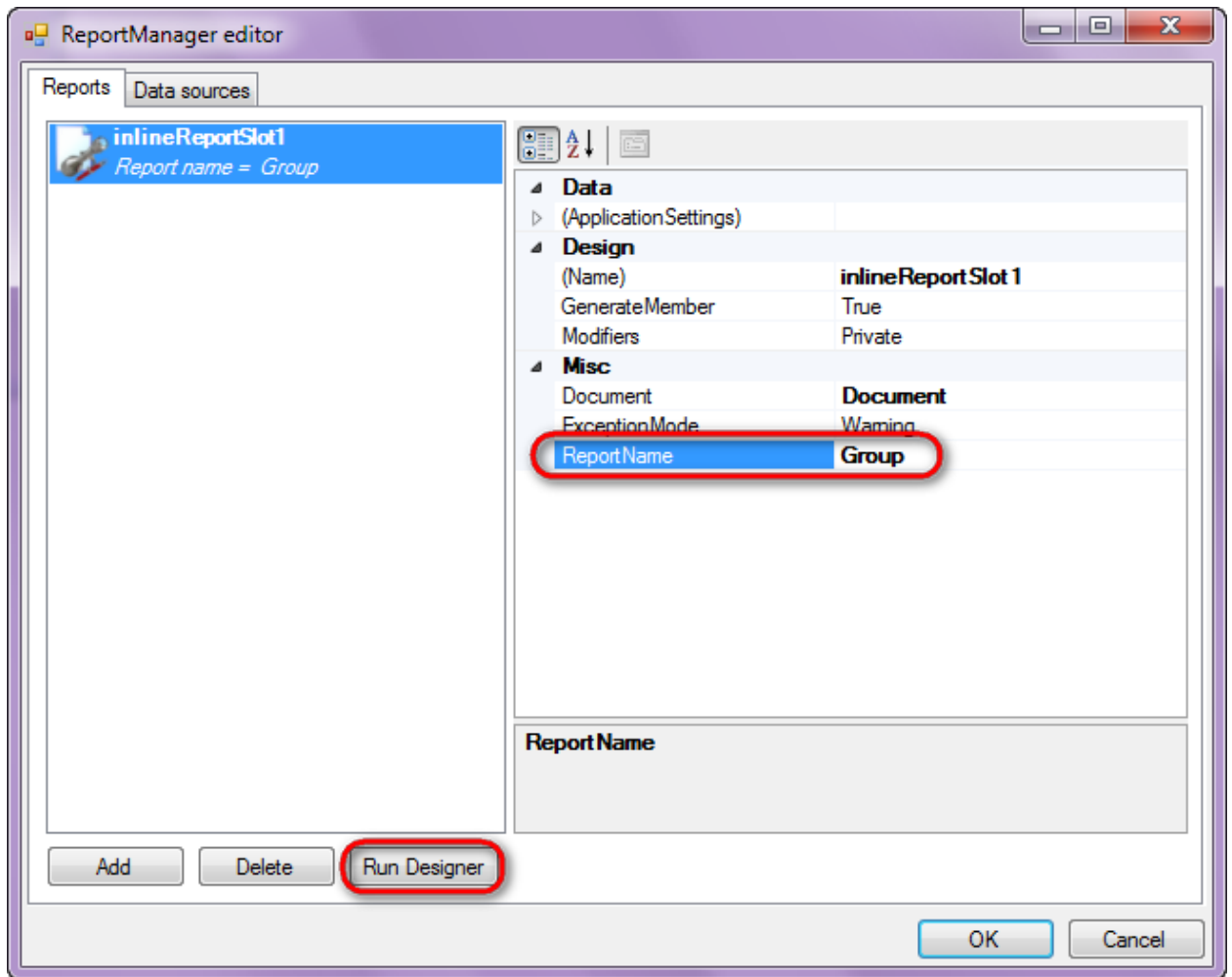
Step 10

Go to "Reports" tab, click "Add" and select "InlineReportSlot".



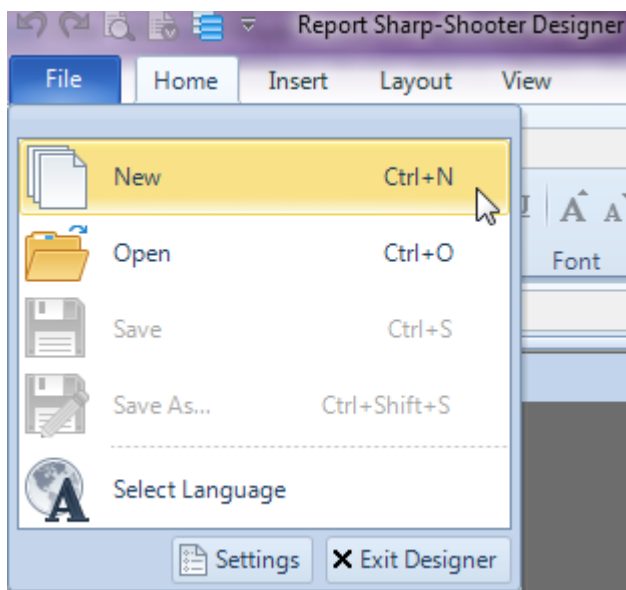
Step 11

Set name of the report in the property ReportName - "Group". Click "Run Designer" in order to open template editor - Report Designer.

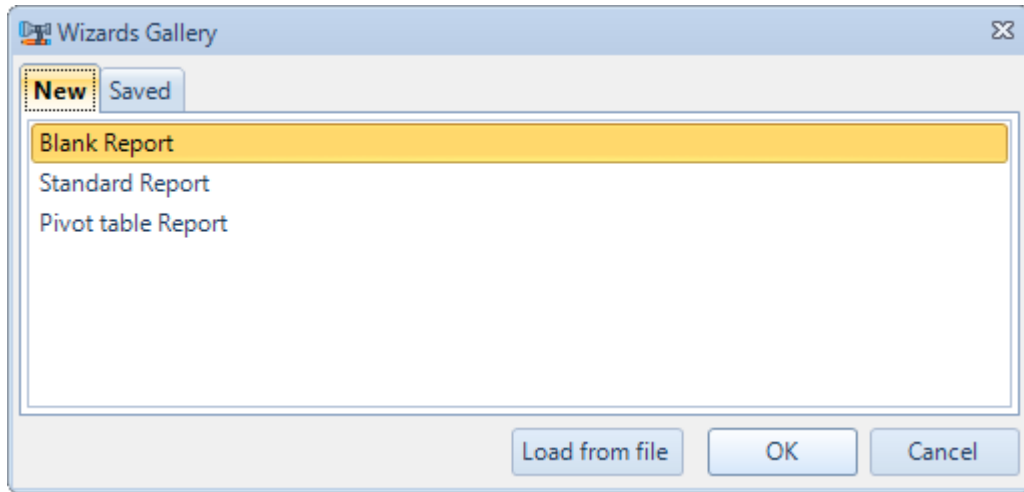


Step 12

Create new empty template – select File\New from the main menu.

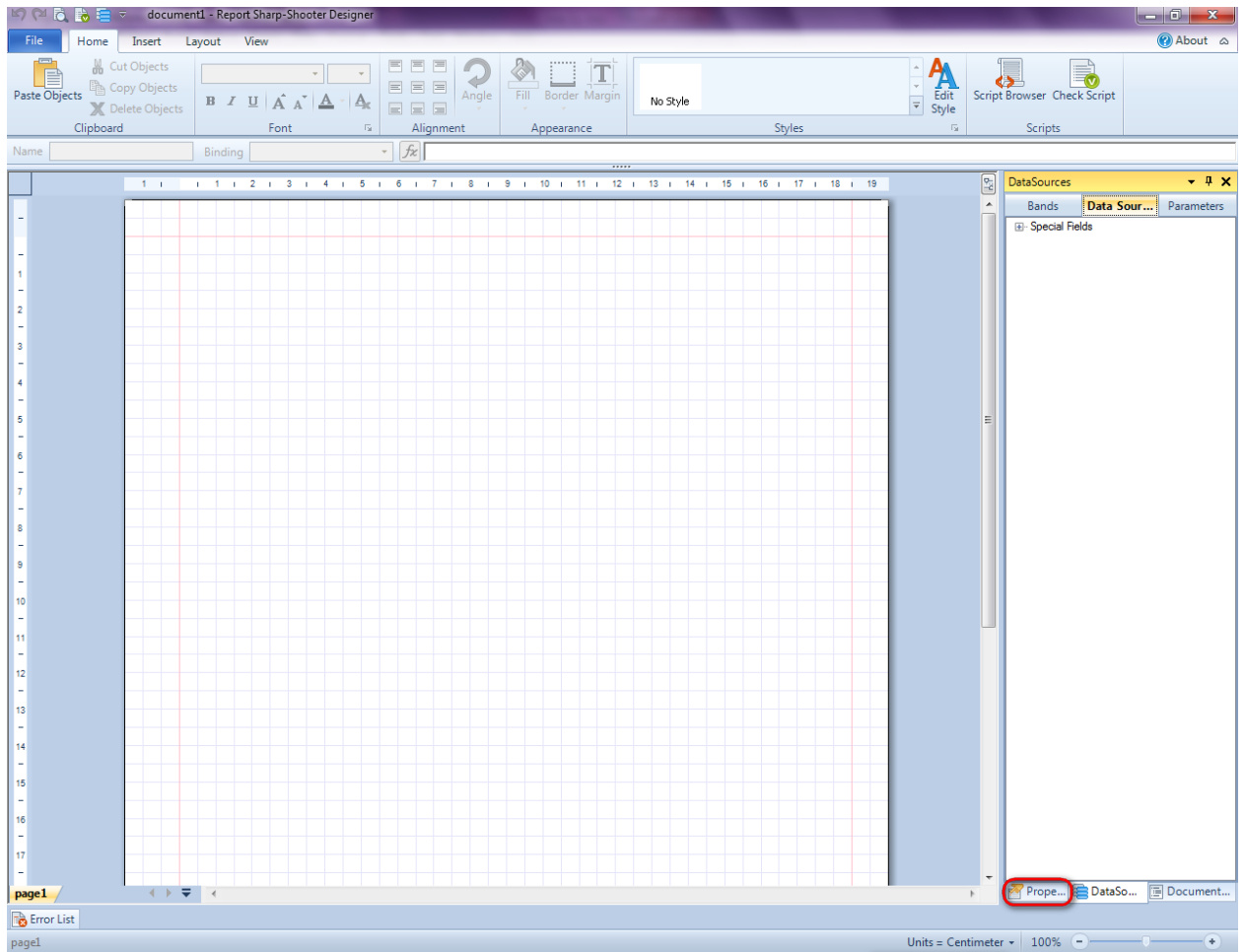


Select "Blank Report" in the Wizards Gallery and click "OK".



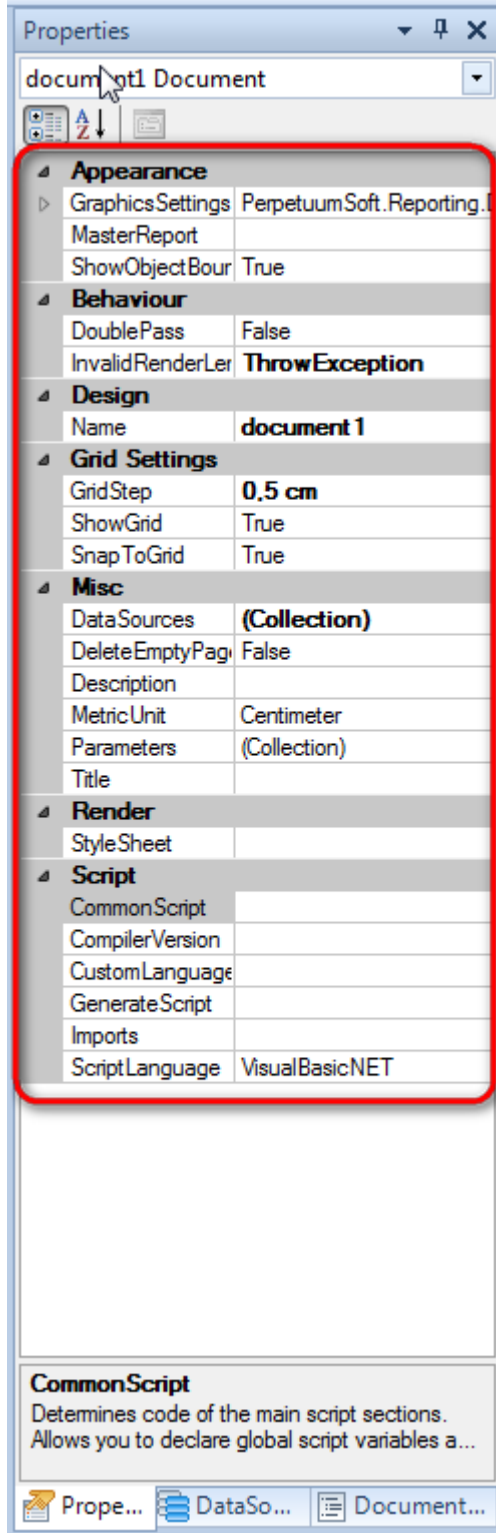
Step 13

Click the "Properties" tab of the tool window in the right part of the designer.

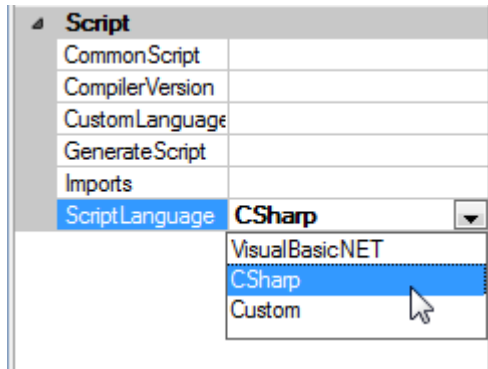




You will see properties of the edited template on the "Properties" tab

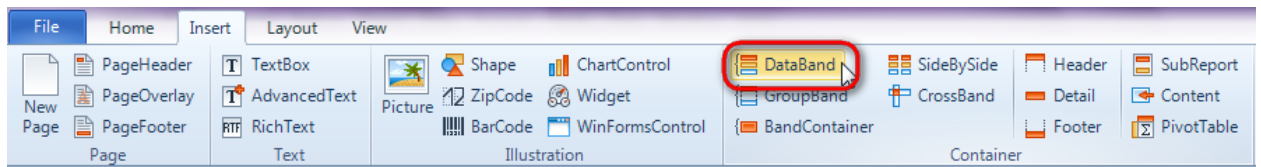


Set property ScriptLanguage = CSharp.



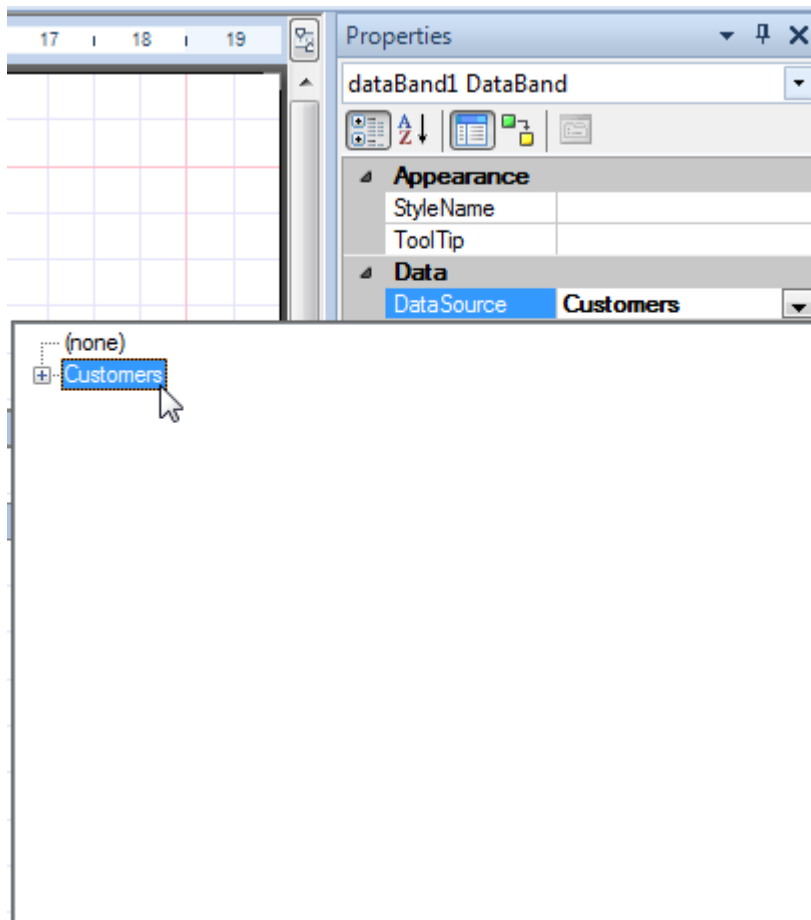
Step 14

Press "DataBand" button on the Insert tab in the group Container.



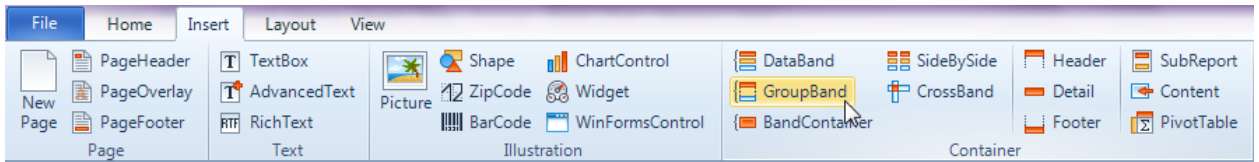
Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = Customers.




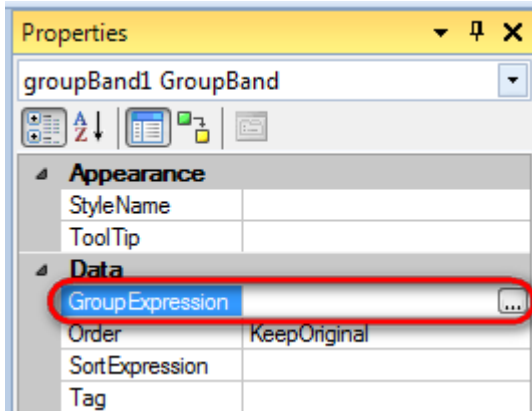
Step 15

Press "GroupBand" button on the Insert tab in the group Container.

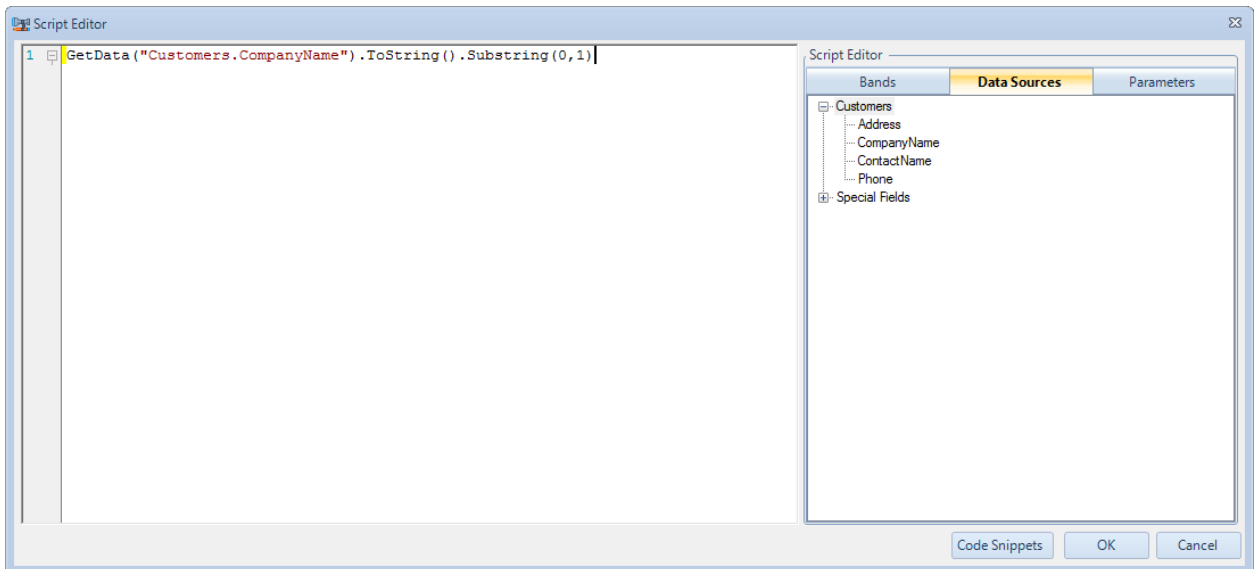


Click on the DataBand area to add GroupBand inside DataBand.

Select GroupExpression property on the "Properties" tab. Click button  to open property editor – Script Editor.



Enter code to group records by the first letter in the company name:
"GetData("Customers.CompanyName").ToString().Substring(0,1)".

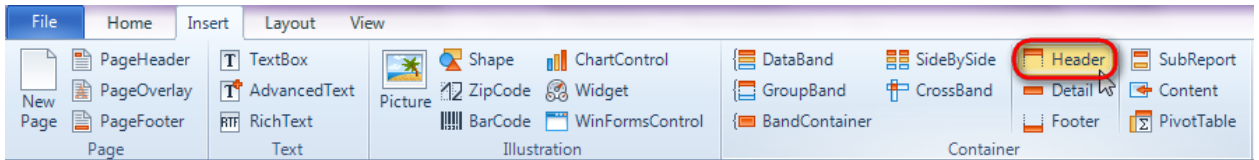


Template should look as follows on this stage:

```
dataBand1:DataBand DataSource = Customers
groupBand1:GroupBand Group = GetData("Customers.CompanyName").ToString().Substring(0,1)
end of groupBand1
end of dataBand1
```

Step 16

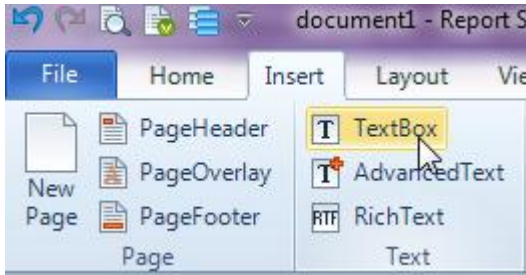
To display header for every group, create Header band – press "Header" button on the Insert tab in the group Container.



Click on the GroupBand area to add Header band inside GroupBand.

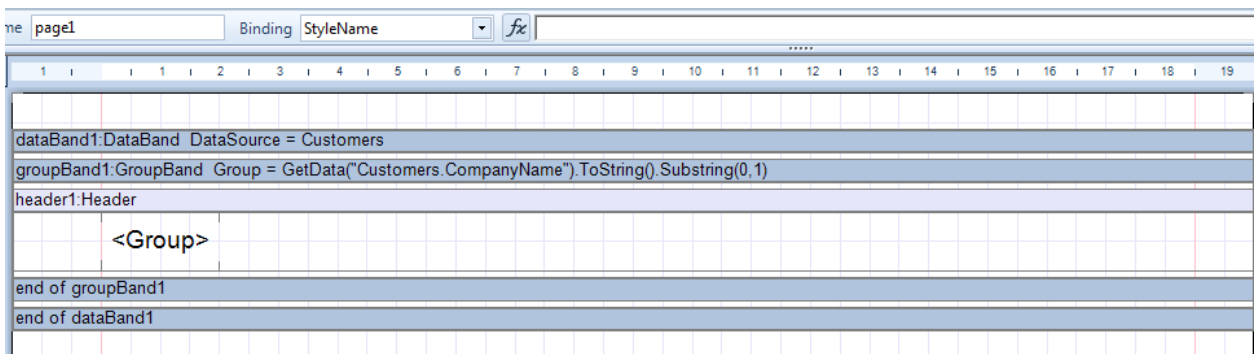
Step 17

Press button "TextBox" on the Insert tab in the group Text.



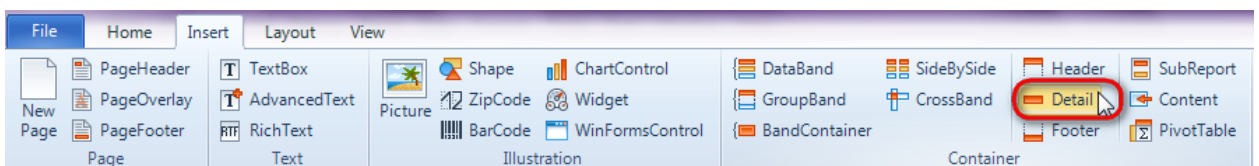
Click on the Header band to add TextBox element to Header.

Set Value property to "Group".

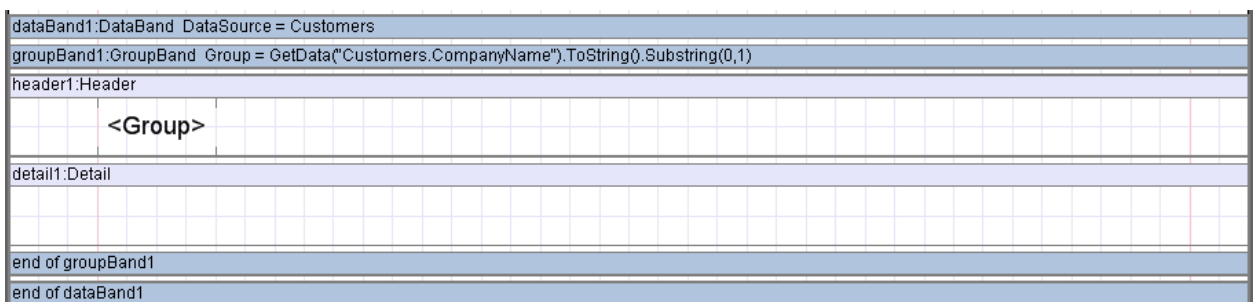


Step 18

Press "Detail" button on the Insert tab in the group Container.

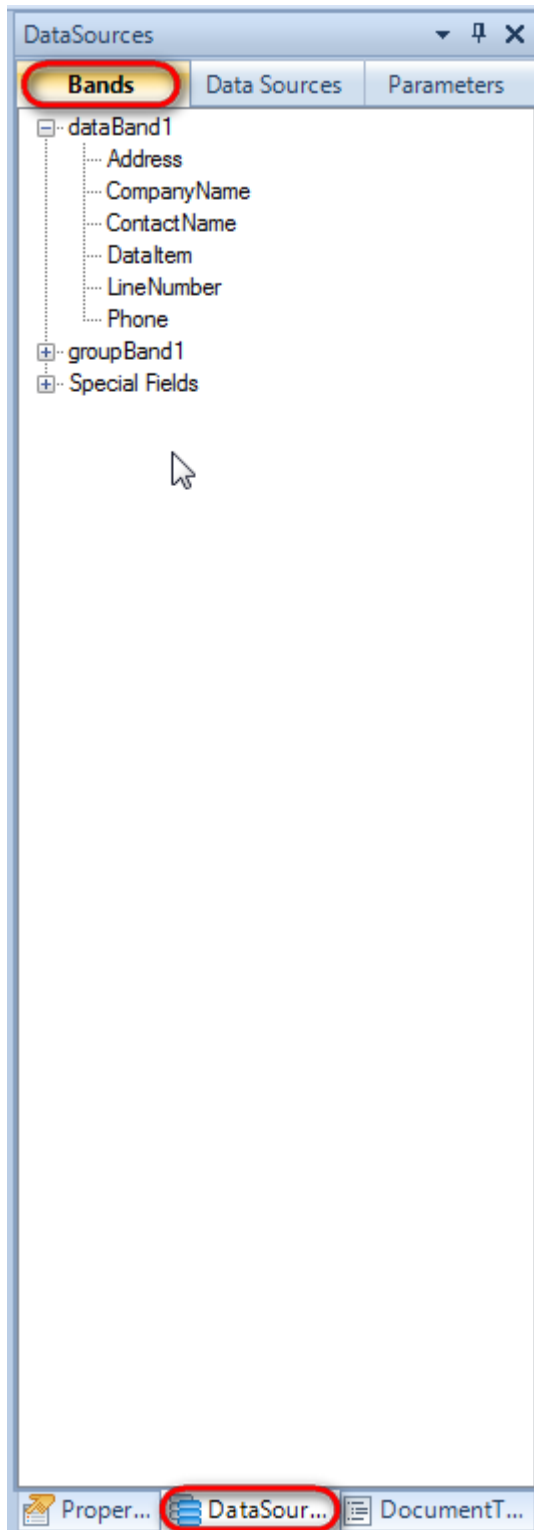


Click on the GroupBand area to add Detail band inside GroupBand.



Step 19

Go to "DataSources" tab.

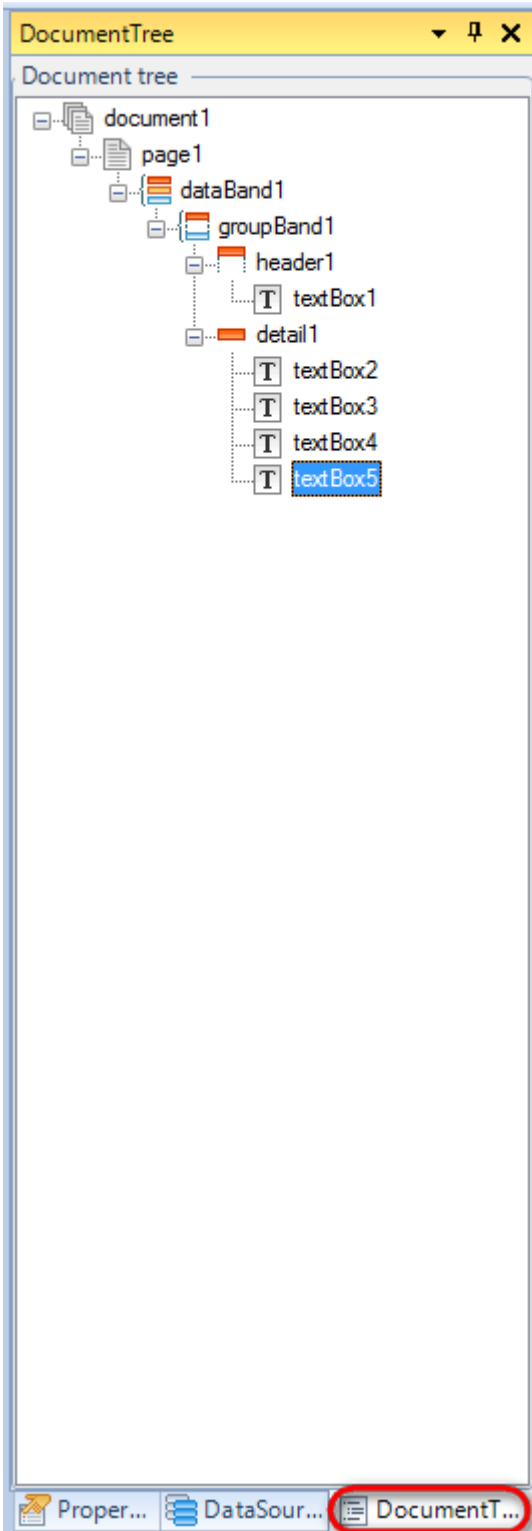


Drag and drop "CompanyName", "Address", "ContactName", "Phone" fields from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source. Change size of the elements and locate them in the way shown in the picture below.



```
dataBand1:DataBand DataSource = Customers
groupBand1:GroupBand Group = GetData("Customers.CompanyName").ToString().Substring(0,1)
header1:Header
<Group>
detail1:Detail
<dataBand1 ["CompanyName"]> <dataBand1 ["ContactName"]> <dataBand1 ["Address"]> <dataBand1["Phone"]>
end of groupBand1
end of dataBand1
```

In order to view template structure, go to "DocumentTree" tab.





Step 20

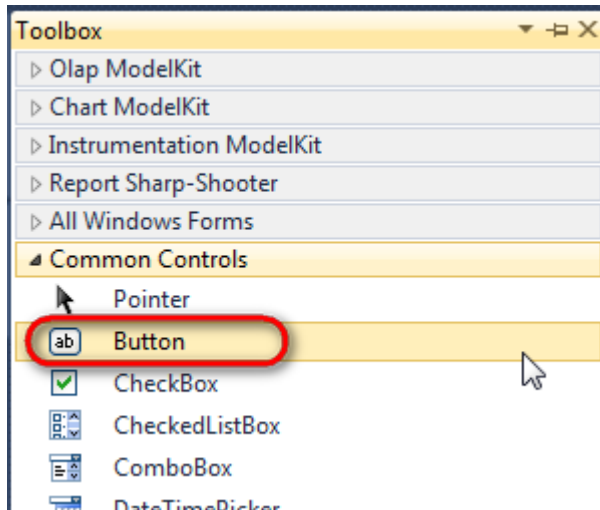
Save template, close Report Designer.

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object

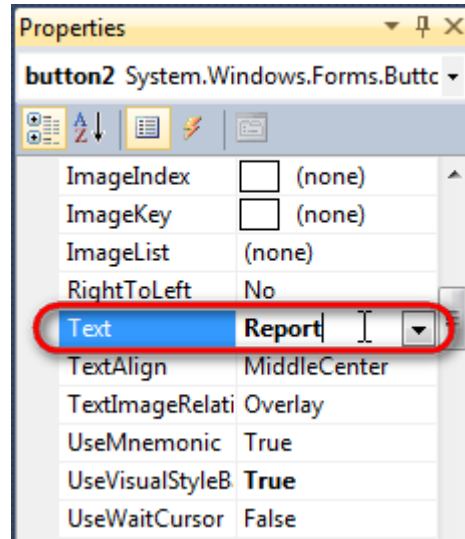
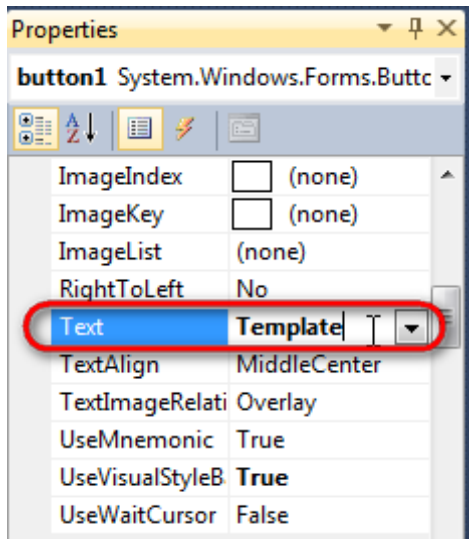
```
public Form1()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["Address"] = "Obere Str. 57";
    row["ContactName"] = "Maria Anders";
    row["Phone"] = "030-0074321";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["Address"] = "Avda. de la Constitución 2222";
    row["ContactName"] = "Ana Trujillo";
    row["Phone"] = "(5) 555-4729";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["Address"] = "Kirchgasse 6";
    row["ContactName"] = "Roland Mendel";
    row["Phone"] = "7675-3425";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["Address"] = "Luisenstr. 48";
    row["ContactName"] = "Karin Josephs";
    row["Phone"] = "0251-031259";
    row["Phone"] = "7675-3425";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["Address"] = "Luisenstr. 48";
    row["ContactName"] = "Karin Josephs";
    row["Phone"] = "0251-031259";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 21

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



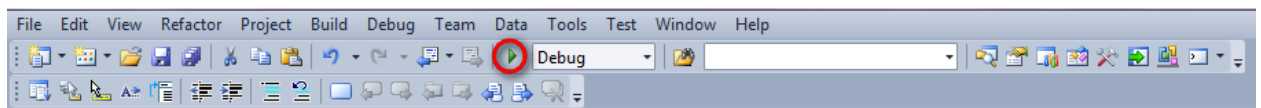
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

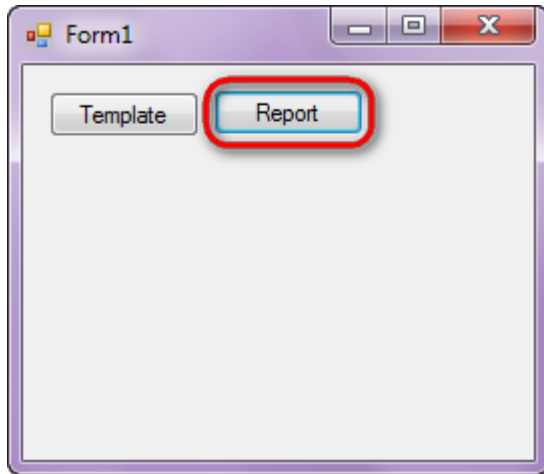
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

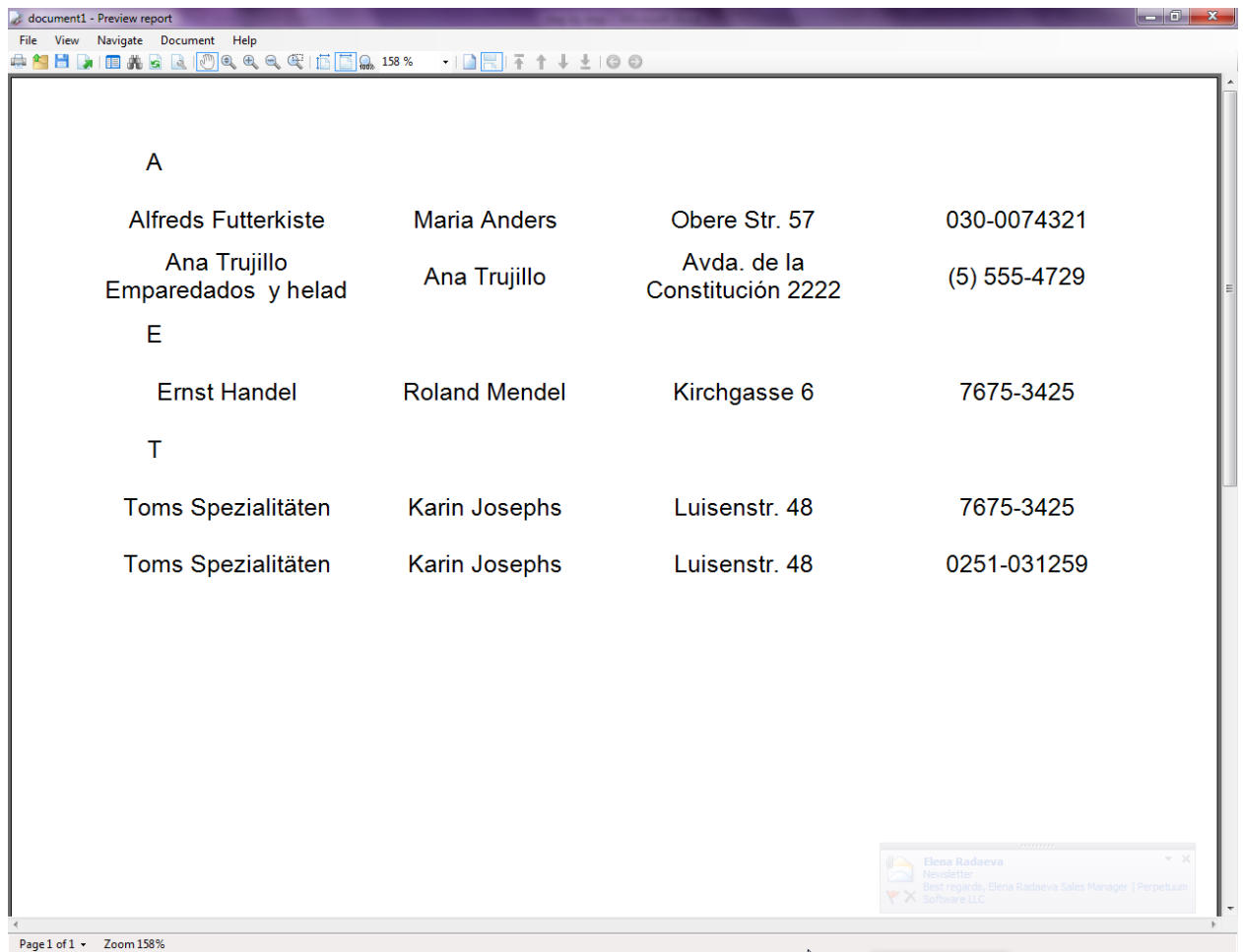
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



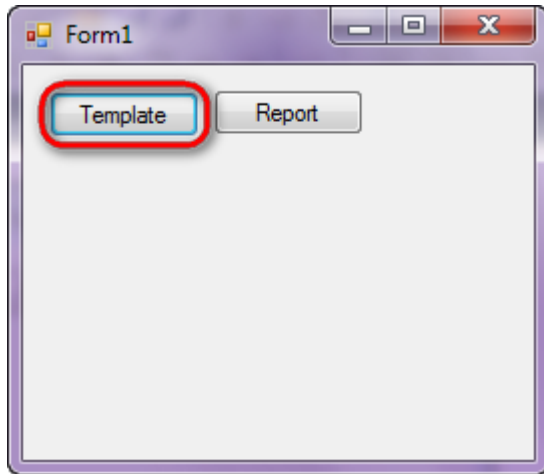
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.



In order to edit template, close Report Viewer and press "Template" on the application form.



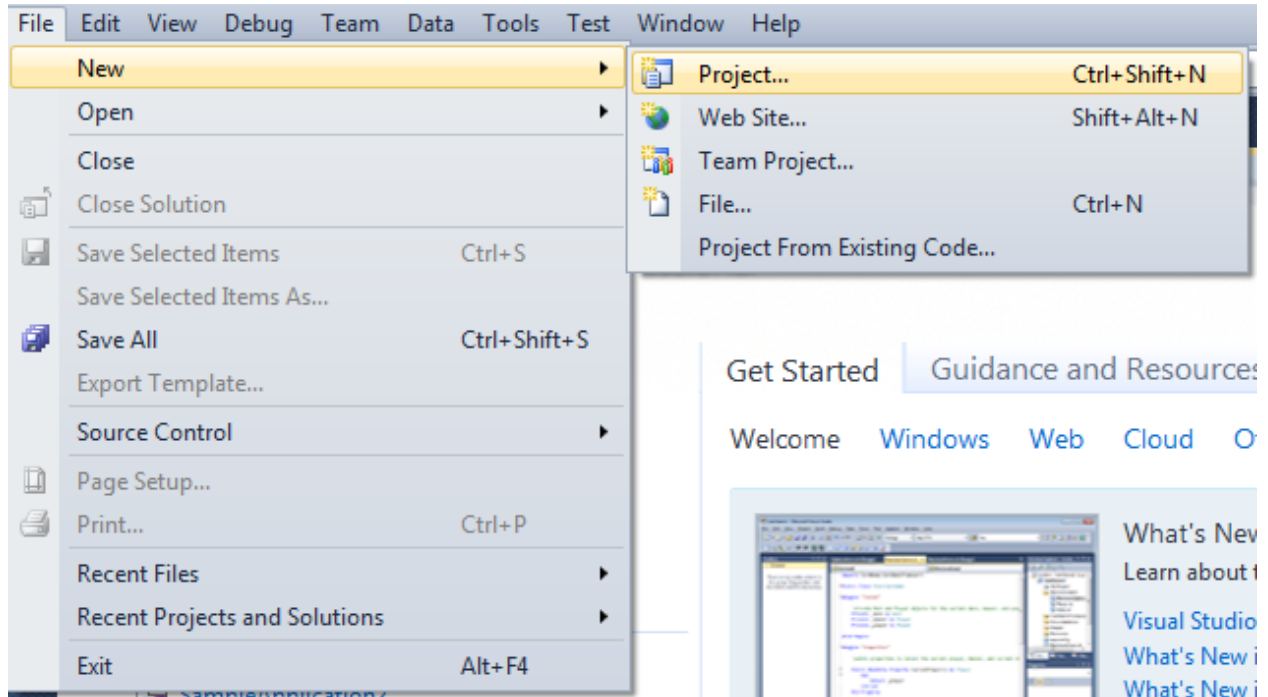
Similar sample in the Samples Center is Reports\Grouping\Groups.

Hyperlinks and Bookmarks

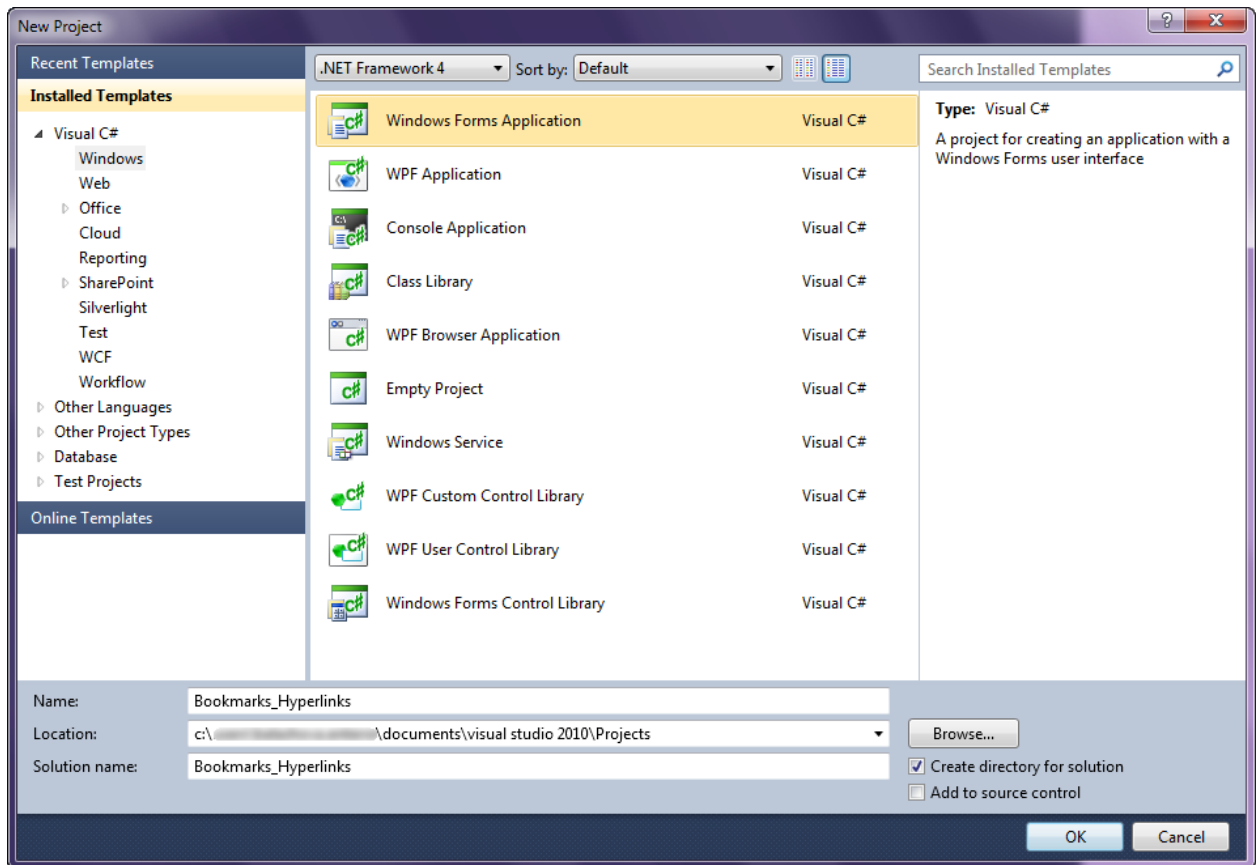
Template of a report containing a list of customers grouped by the first letter. Report contains a table of contents with hyperlinks for quick navigation and links to customer websites and emails of contact persons.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

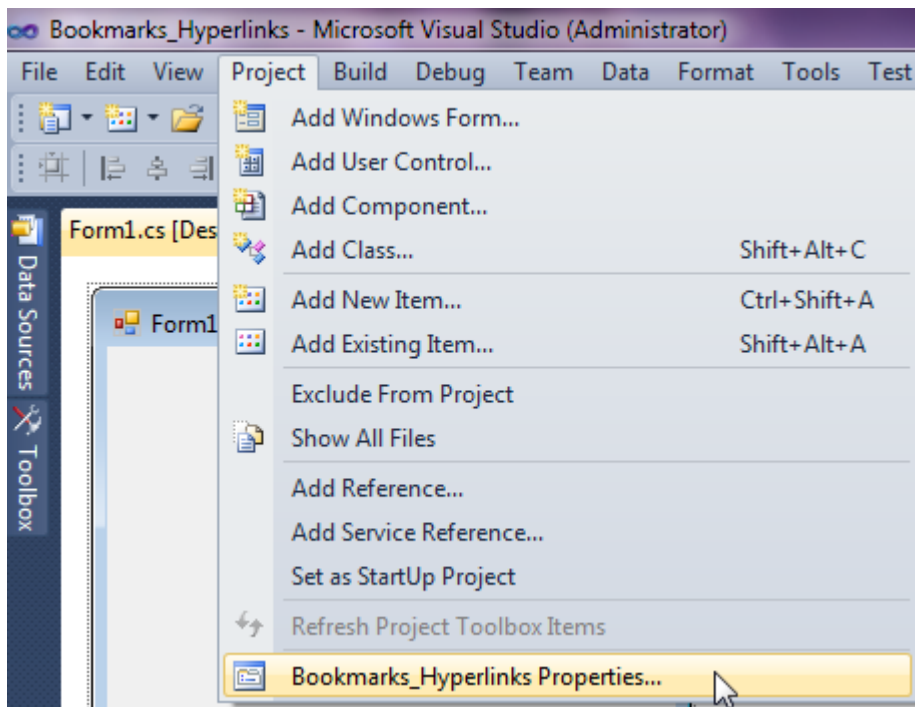


Select Windows Forms Application, set project name – “Bookmarks_Hyperlinks”, set directory to save the project to.

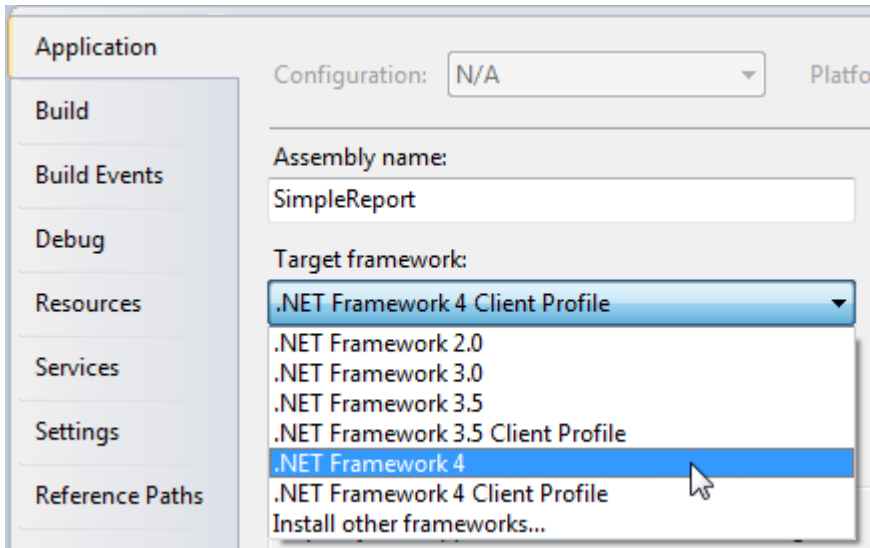


Step 2

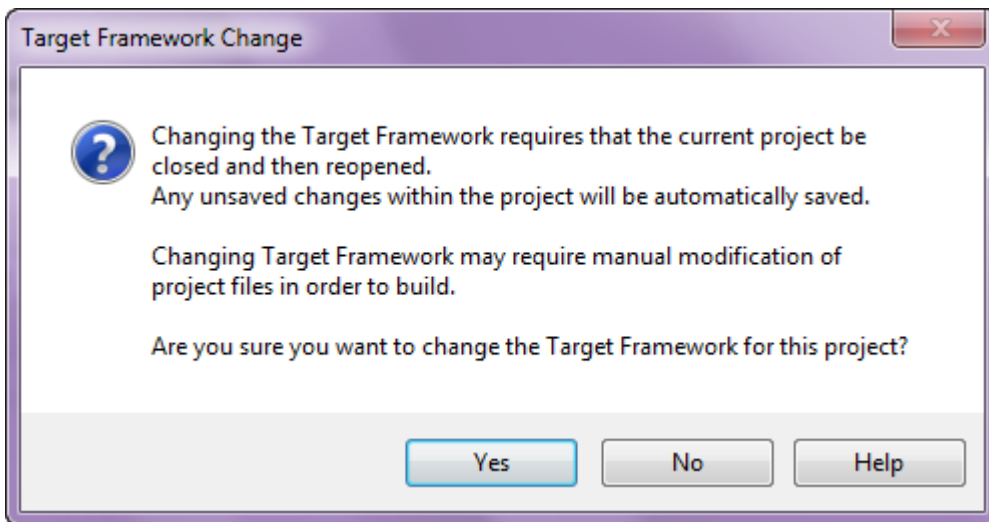
Change the project properties. Select the Project\Bookmarks_Hiberlinks Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

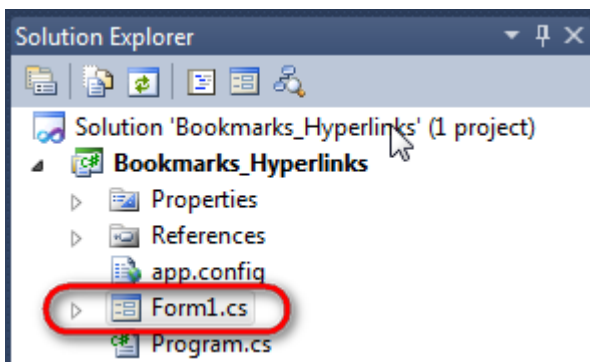


Press the "Yes" button in the opened window.

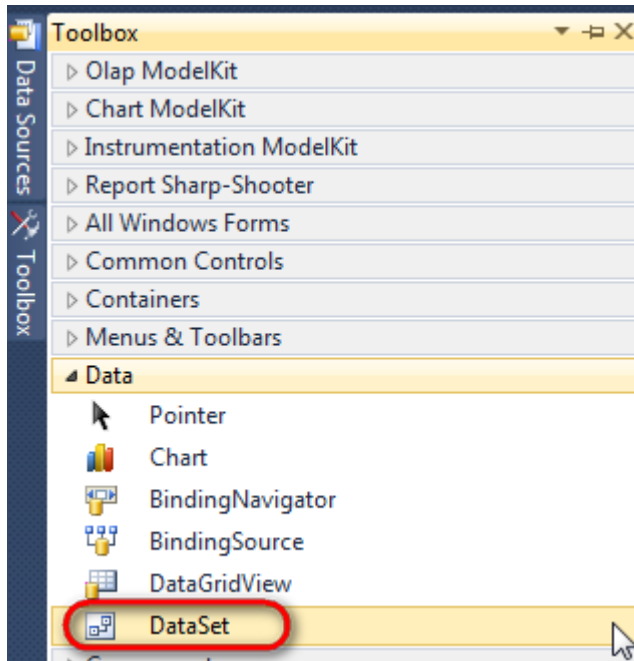


Step 3

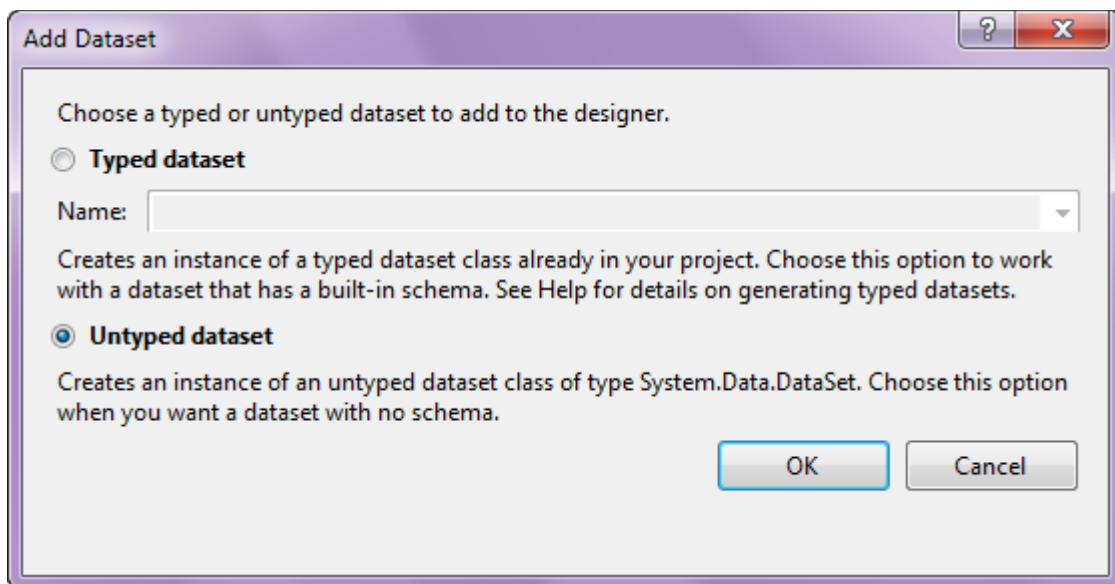
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



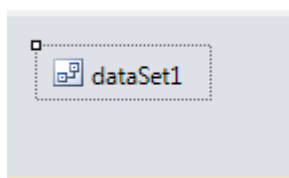
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

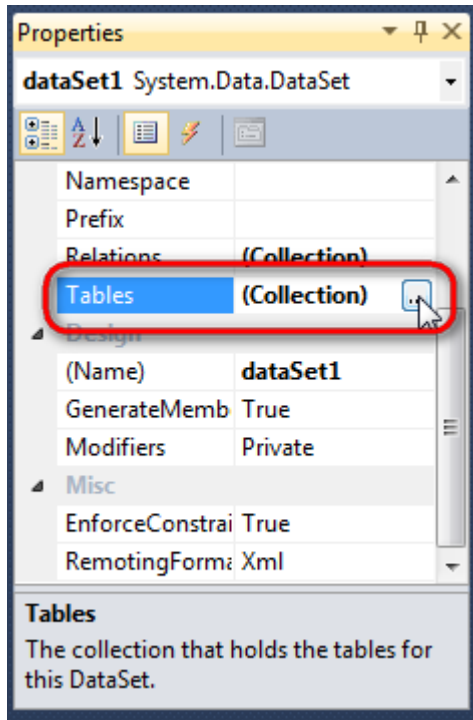


The component is available in the lower part of the window.

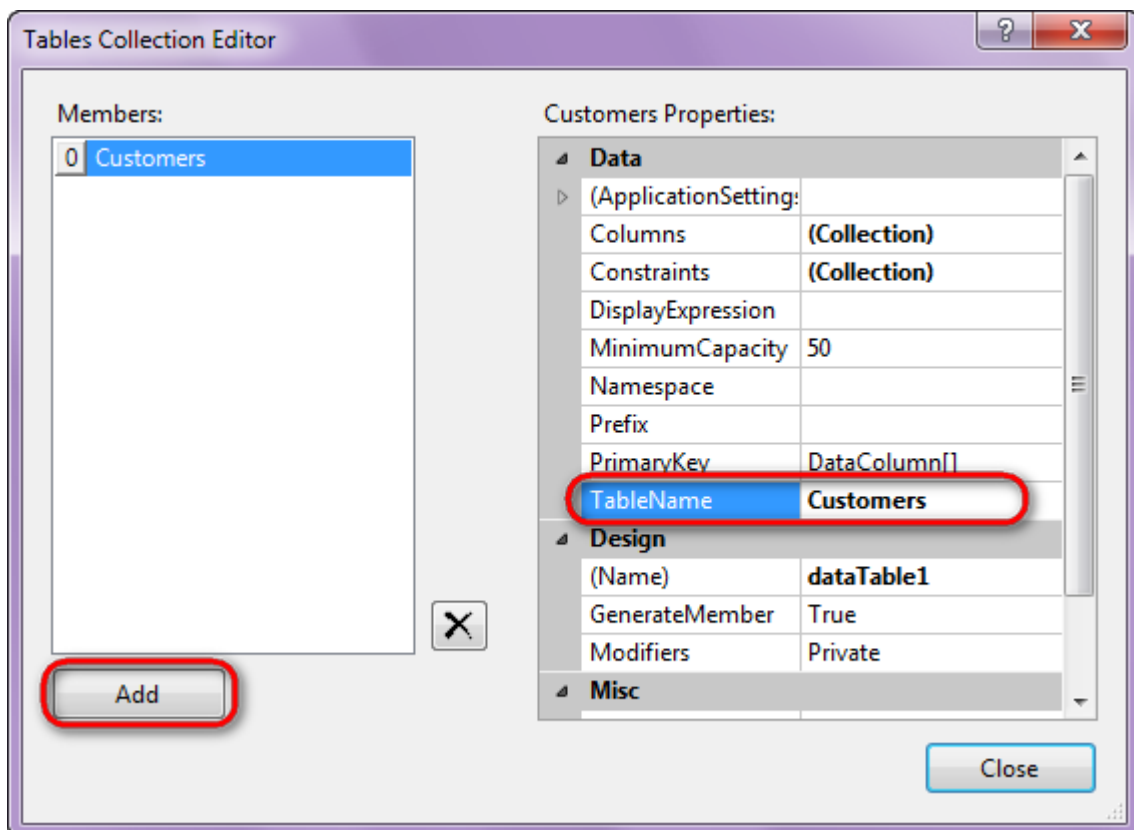


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

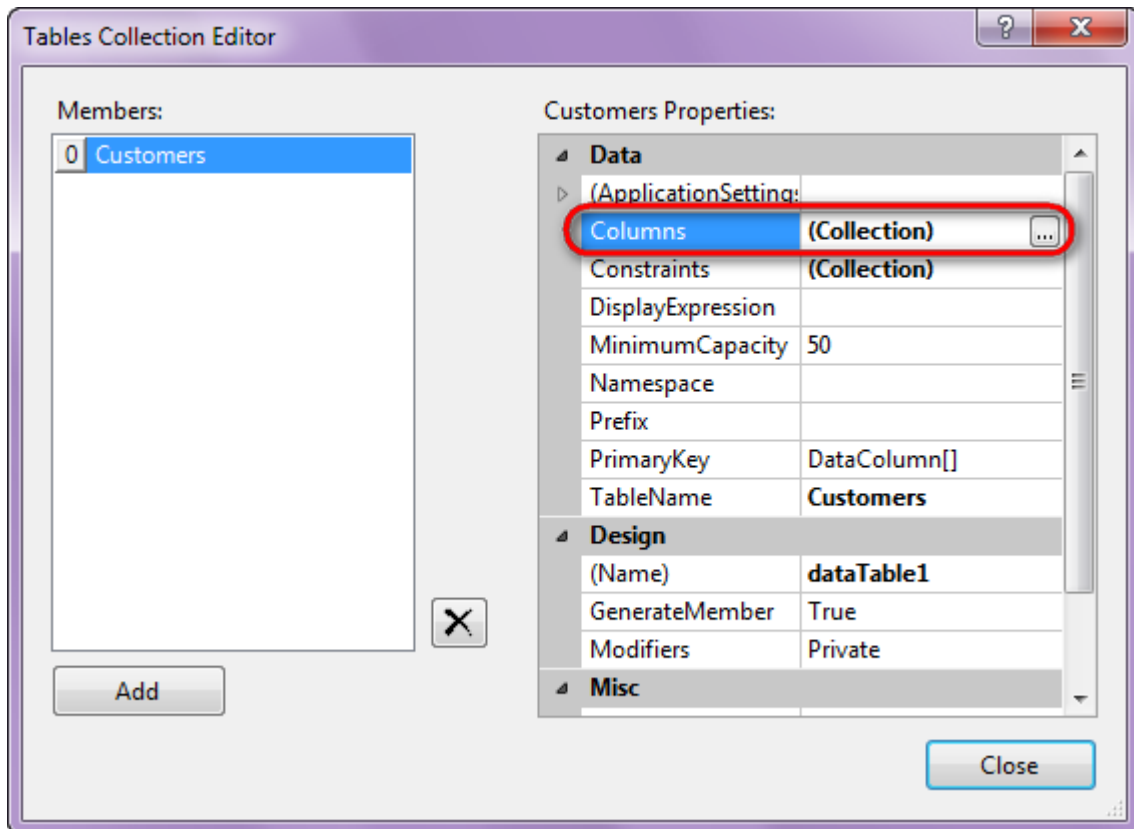


Click "Add" in order to add table. Set property TableName = Customers.

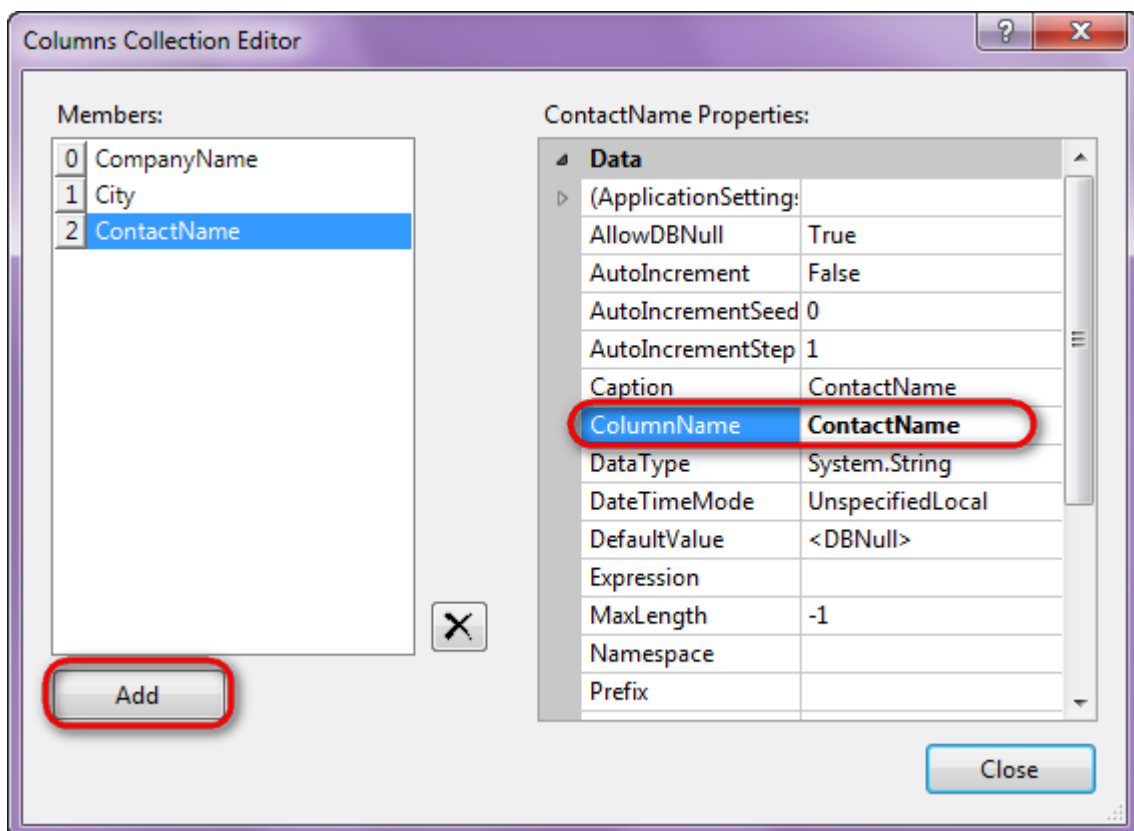


Step 5

Select Columns property, click button  in order to open property editor.

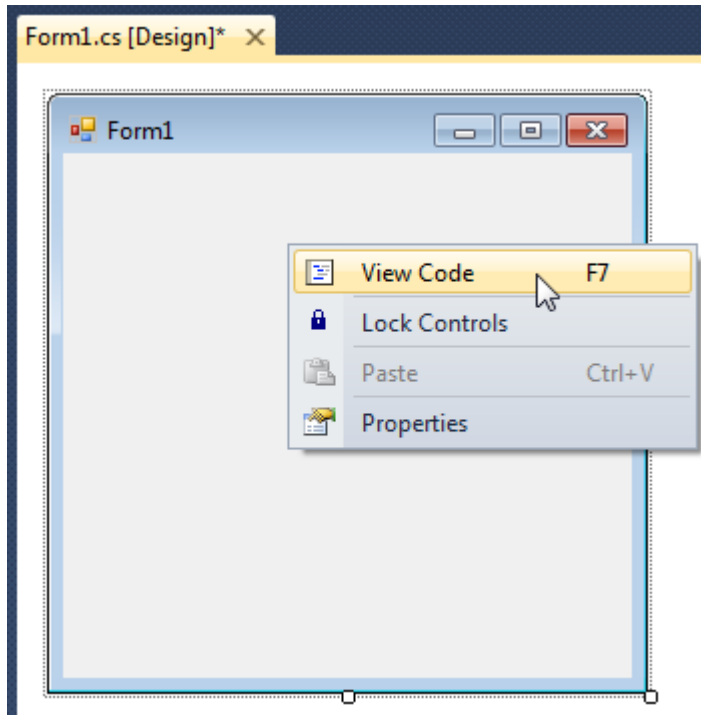


Click "Add" to add a new column. Add three columns. Set ColumnName property to "CompanyName", "City", and "ContactName".



Step 6

Right click on the form and select "View Code" in the context menu to view code.

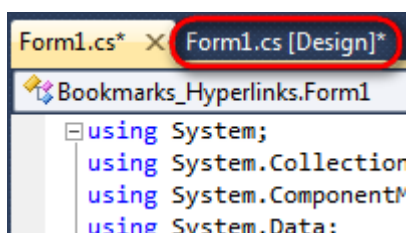


Add the following code to the class constructor in order to fill data source.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["City"] = "Boston";
    row["ContactName"] = "Maria Anders";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["City"] = "London";
    row["ContactName"] = "Ana Trujillo";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CompanyName"] = "Ernst Handel";
    row["City"] = "Paris";
    row["ContactName"] = "Roland Mendel";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CompanyName"] = "Toms Spezialitäten";
    row["City"] = "Moscow";
    row["ContactName"] = "Karin Josephs";
    dataTable1.Rows.Add (row);
}
```

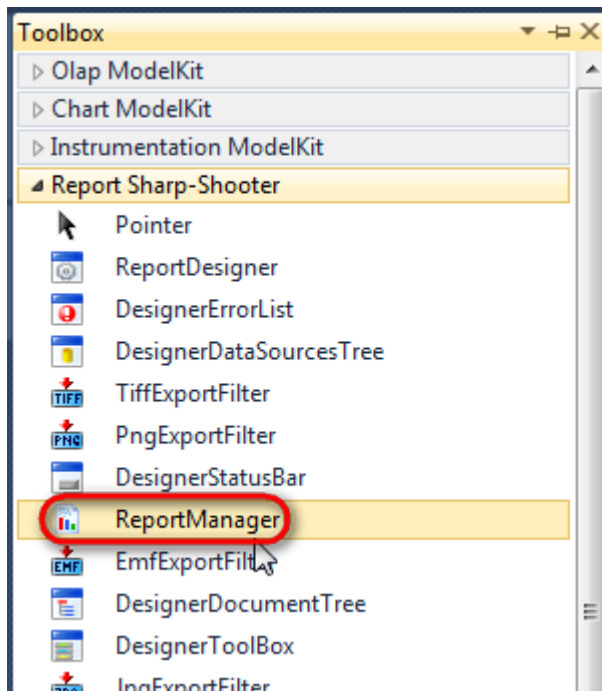
Step 7

Get back to the application form by clicking the "Form1.cs[Design]" tab.

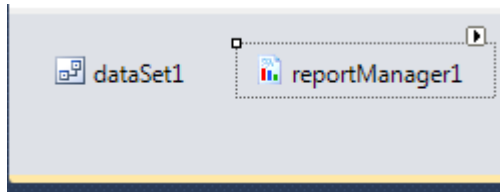




Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

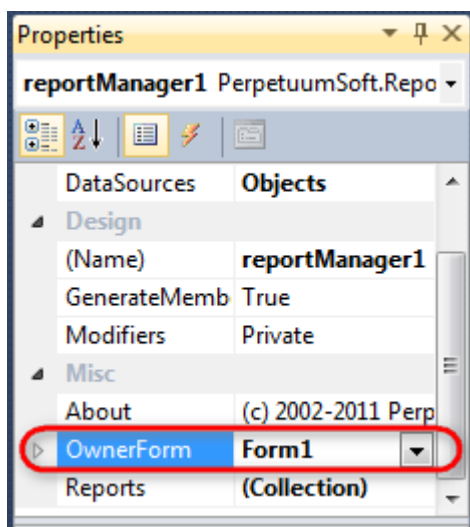


The component is available in the lower part of the window.



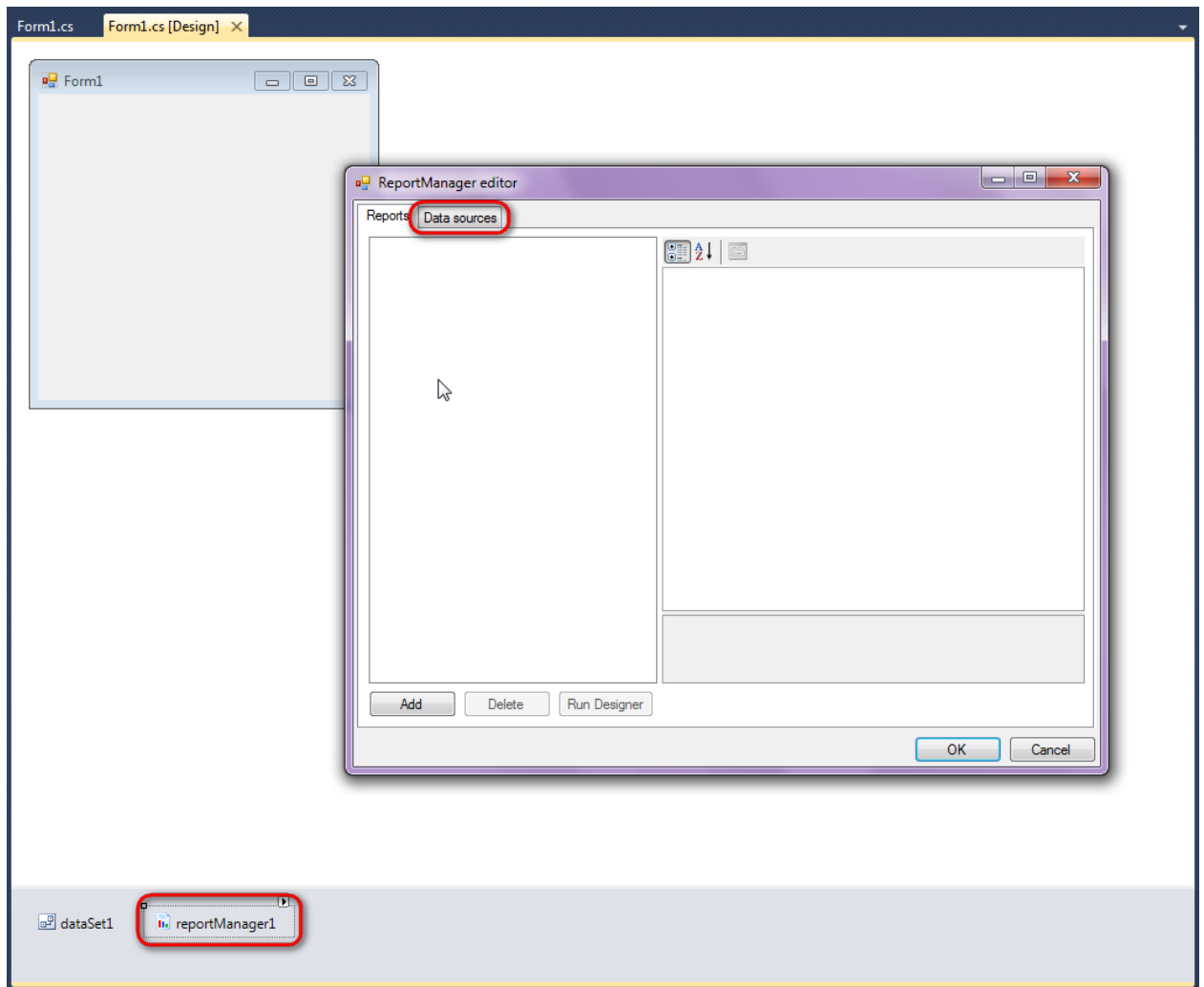
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

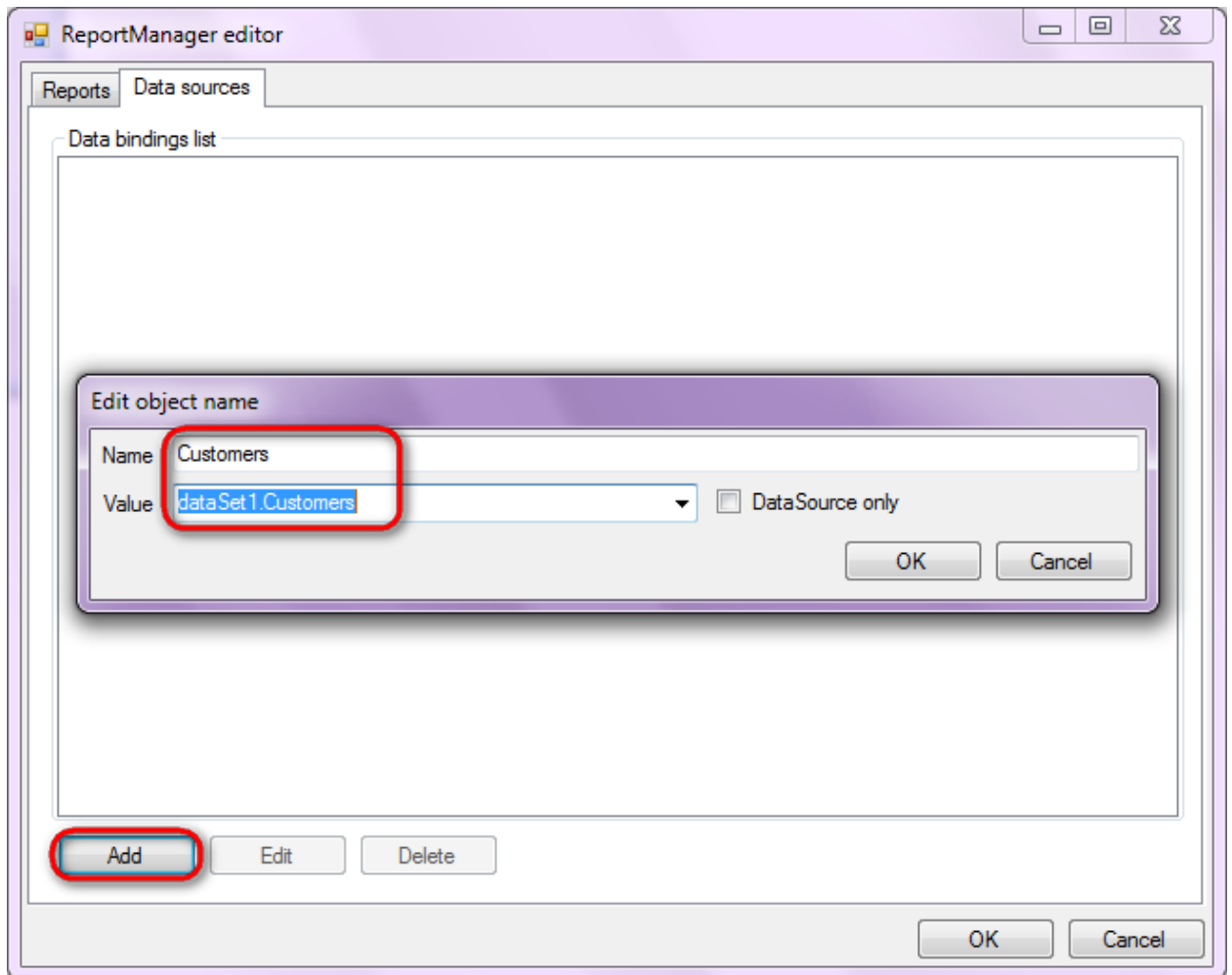


Step 9

Double click on ReportManager to open ReportManager editor.

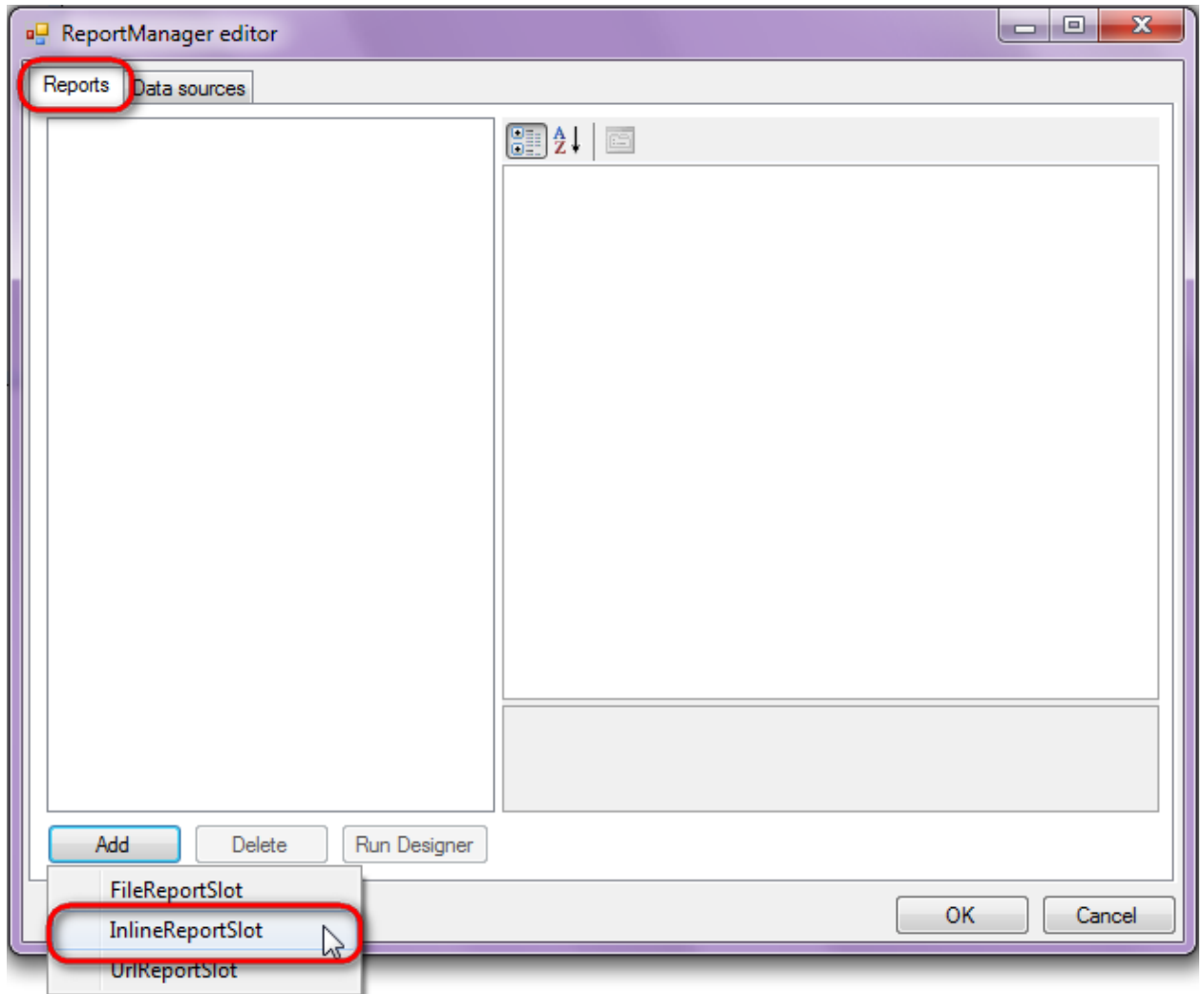


Go to "Data sources" tab, click "Add", set data source name – "Customers", select data source value – "dataSet1.Customers".



Step 10

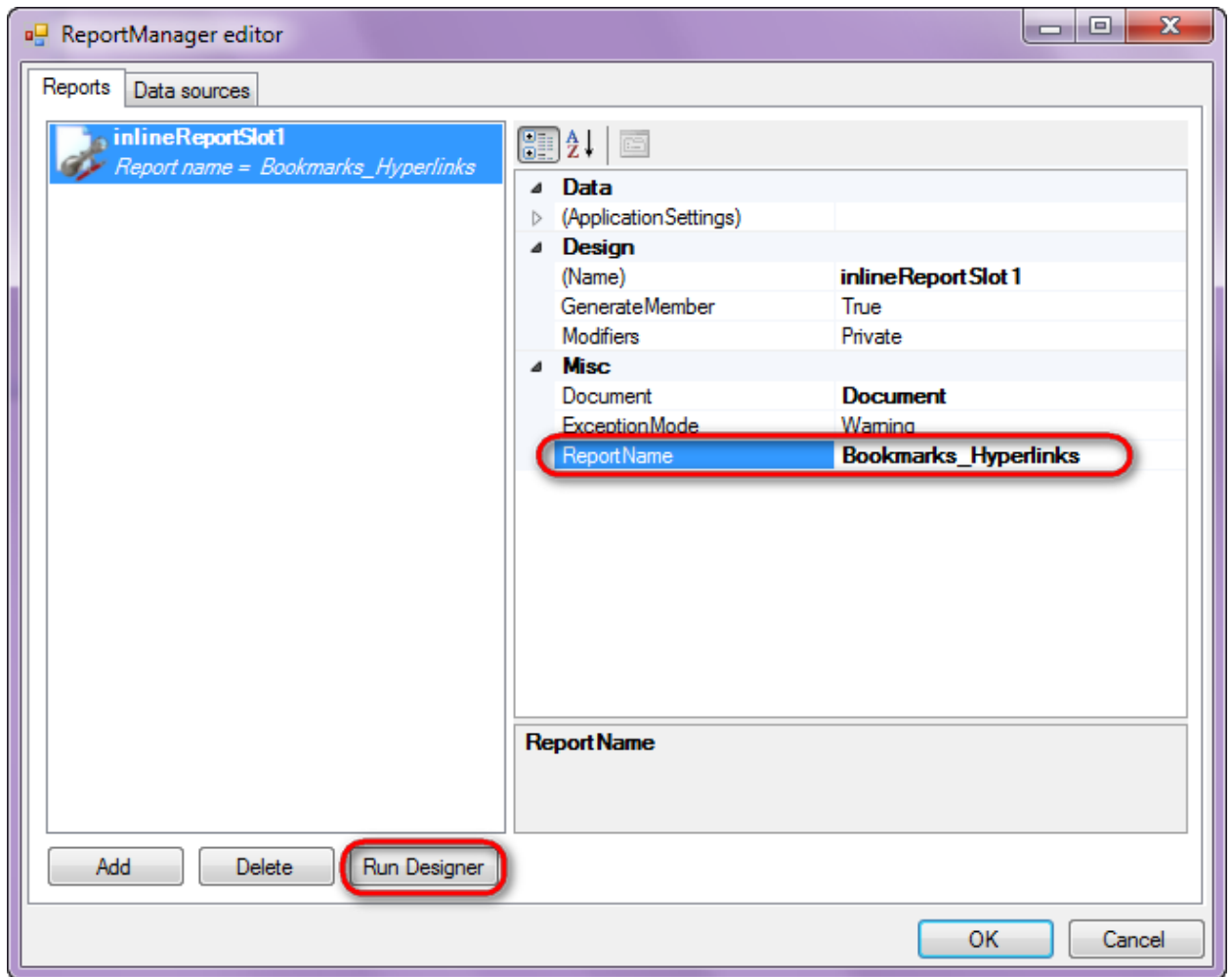
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

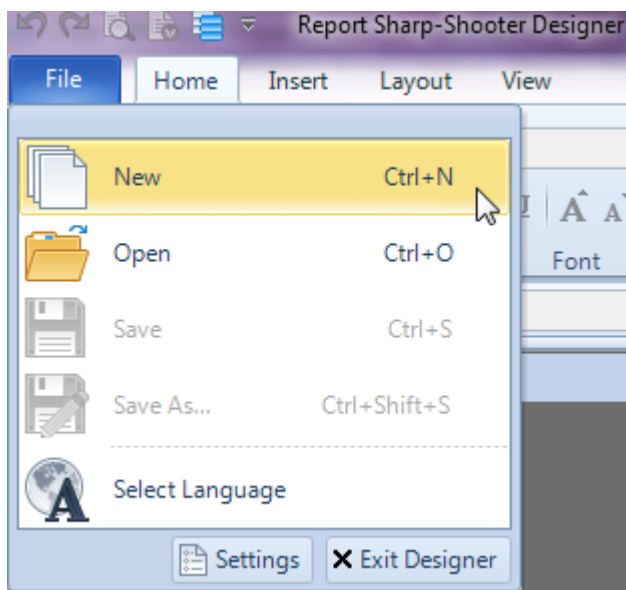
Set name of the report in the property ReportName – “Bookmarks_Hyperlinks”.

Click “Run Designer” in order to open template editor - Report Designer.

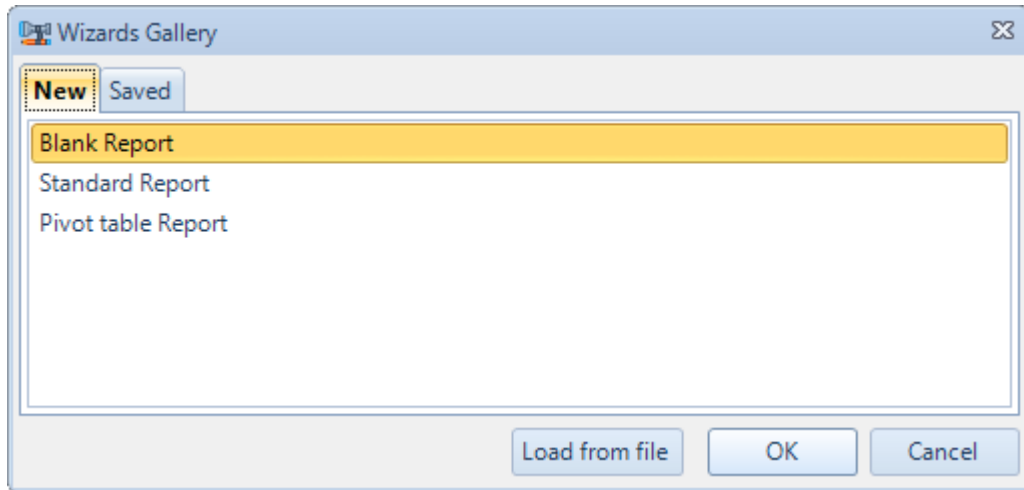


Step 12

Create new empty template – select File\New from the main menu.

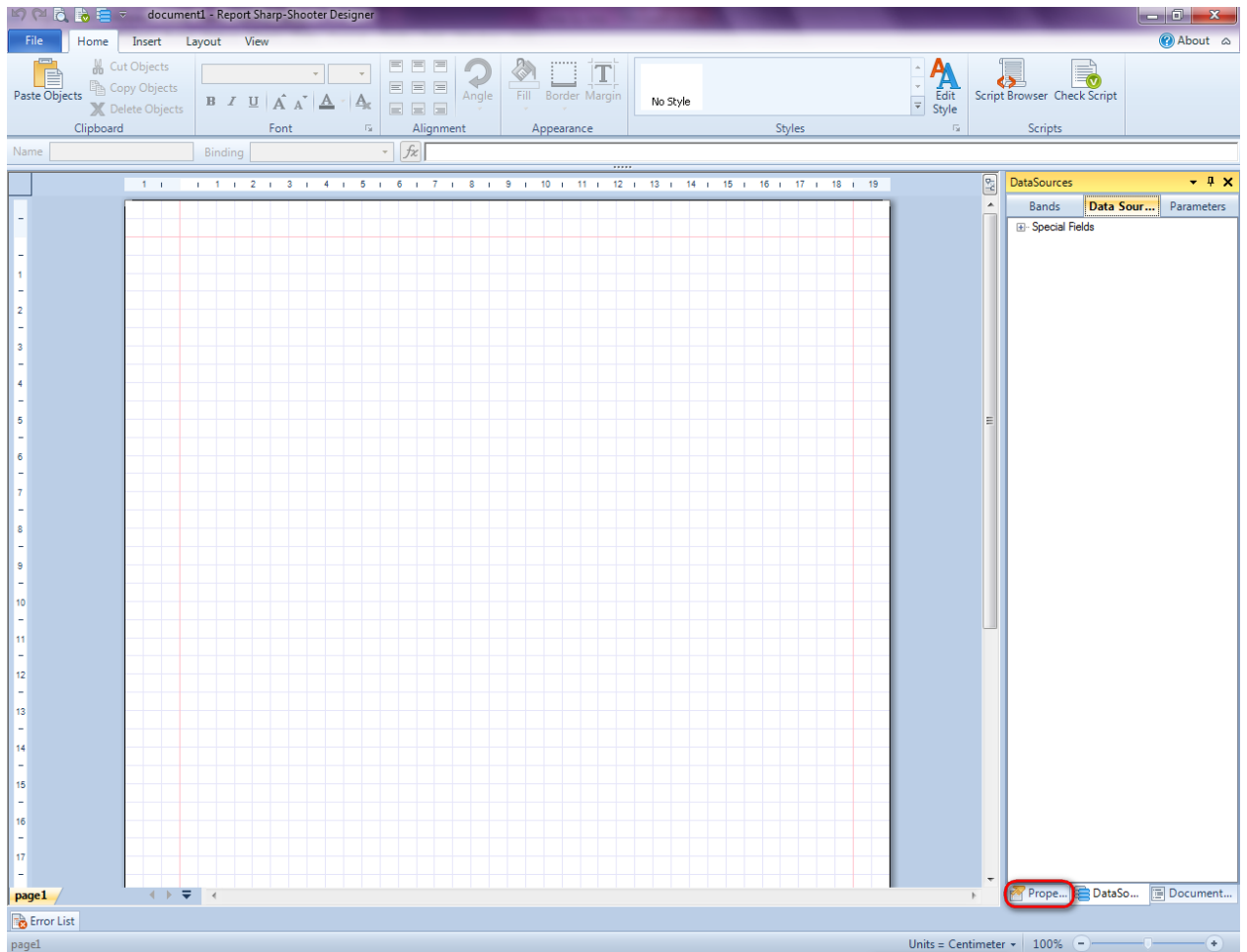


Select "Blank Report" in the Wizards Gallery and click "OK".

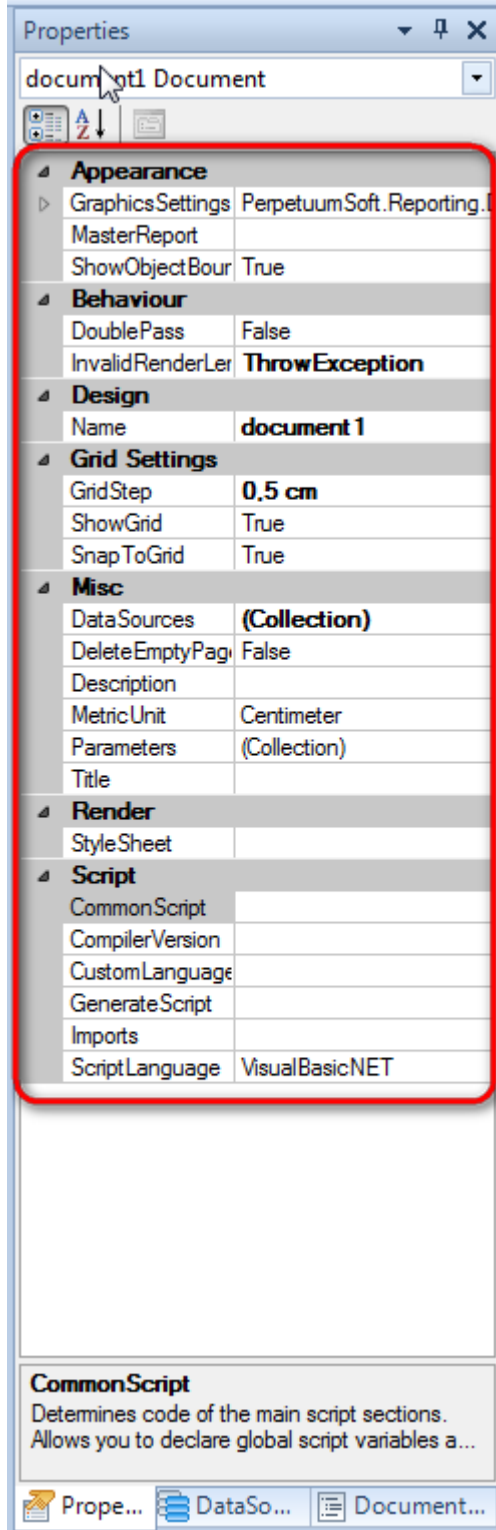


Step 13

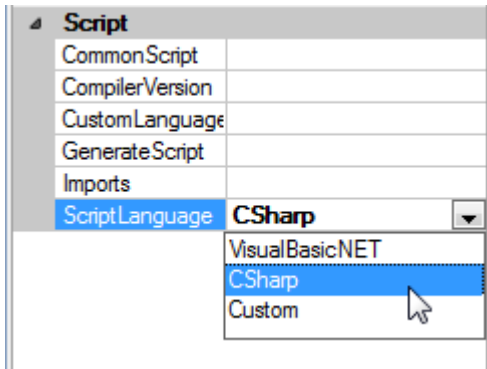
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

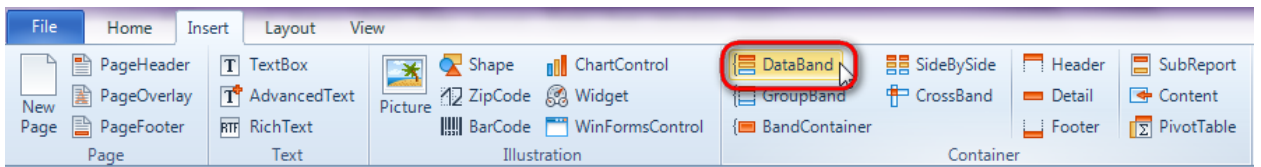


Set property ScriptLanguage = CSharp.



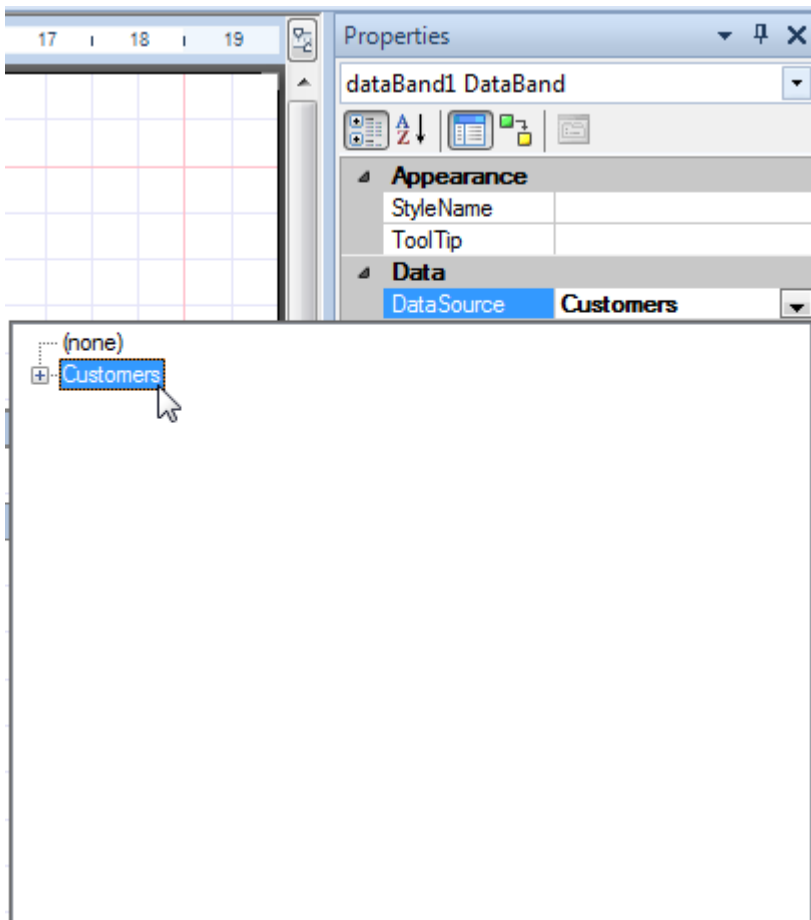
Step 14

Press "DataBand" button on the Insert tab in the group Container.



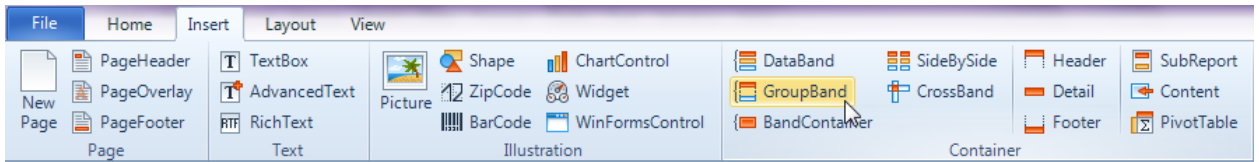
Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = Customers.



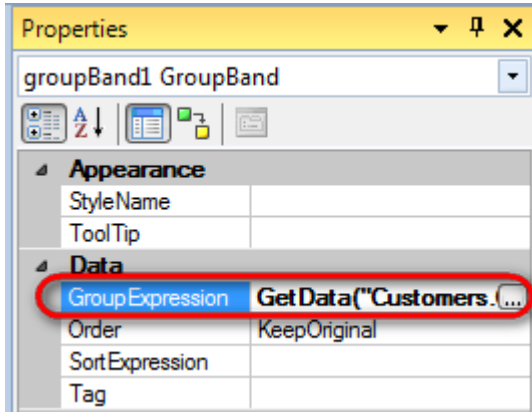
Step 15

Press "GroupBand" button on the Insert tab in the group Container.



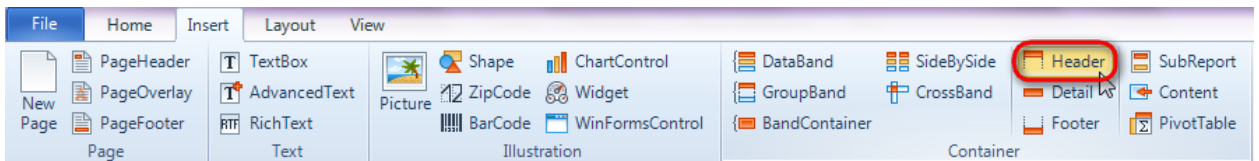
Click on the DataBand area to add GroupBand inside DataBand.

Set property GroupExpression =
`GetData("Customers.CompanyName").ToString().Substring(0,1)`.



Step 16

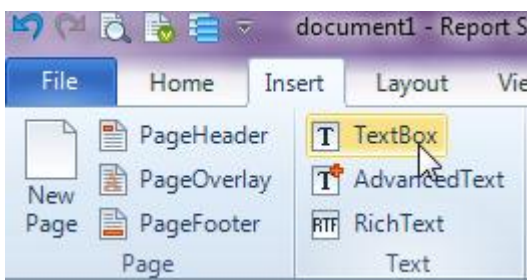
Press "Header" button on the Insert tab in the group Container.



Click on the GroupBand area to add Header band inside GroupBand.

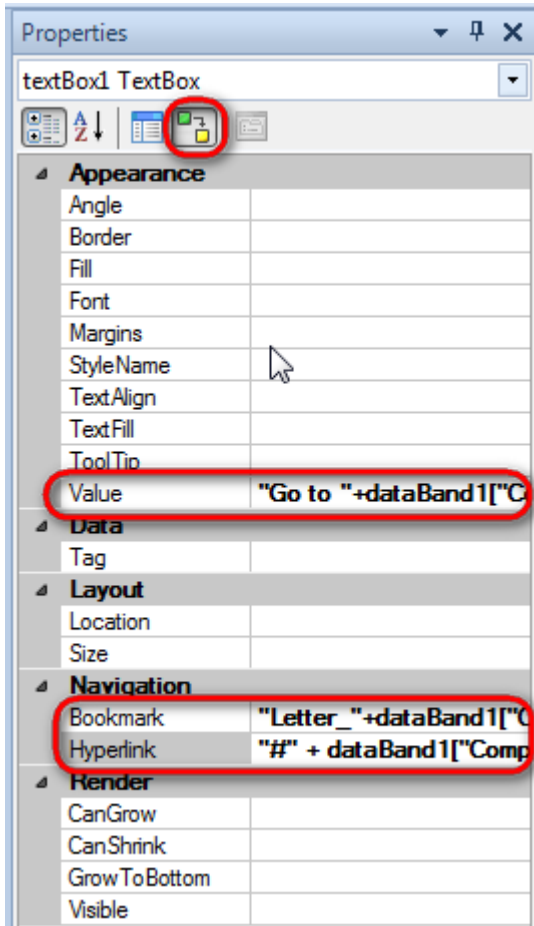
Step 17

Press button "TextBox" on the Insert tab in the group Text.

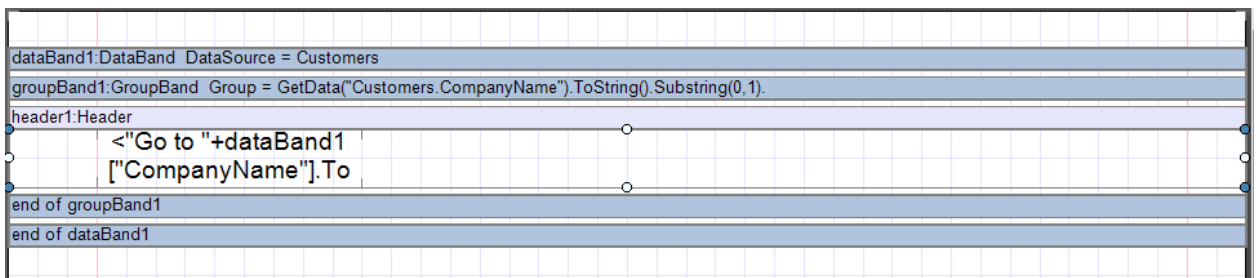


Click on the Header area to add TextBox element inside Header.

Click "Bindings" button on the "Properties" tab. Set properties Value = "`Go to "+dataBand1["CompanyName"].ToString().Substring(0,1)`", Bookmark = "`Letter_" + dataBand1["CompanyName"].ToString().Substring(0,1)`", Hyperlink = "`#" + dataBand1["CompanyName"].ToString().Substring(0,1)`".

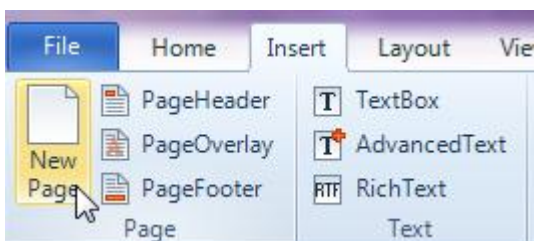


The first template page looks as follows:



Step 18

Press button "NewPage" on the Insert tab in the group Page to add a new template.



Step 19

Add DataBand to the second page. Set property DataSource = Customers.

Step 20

Add GroupBand to the dataBand2 band. Set property GroupExpression = dataBand2["CompanyName"].ToString().Substring(0,1).

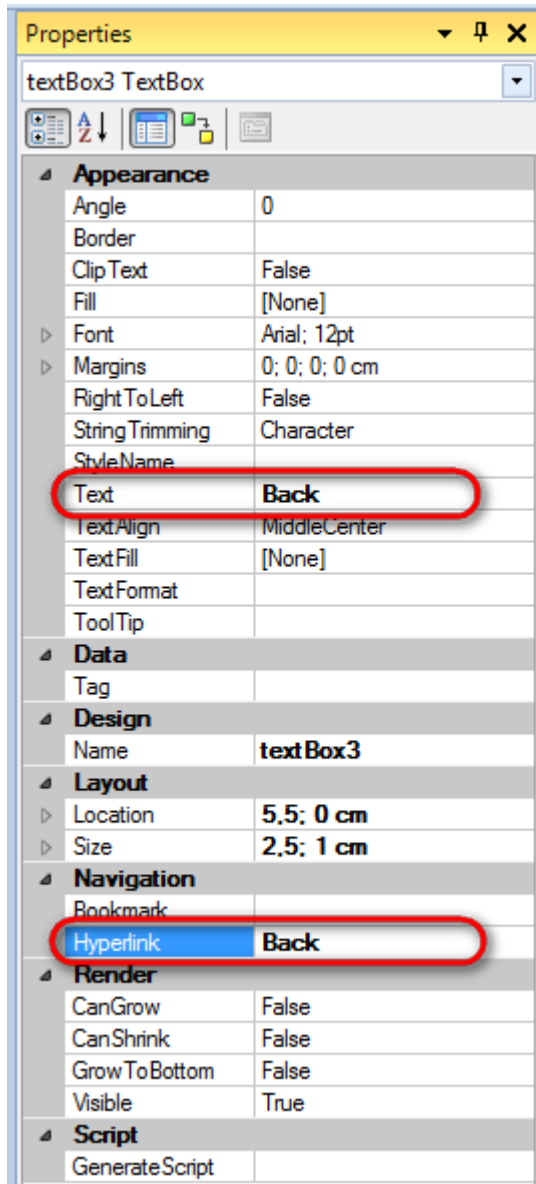


Step 21

Add Header band to the groupBand2 band. Add TextBox. Click "Bindings" button, set properties Value = dataBand2["CompanyName"].ToString().Substring(0,1), Bookmark = "#" + dataBand2["CompanyName"].ToString().Substring(0,1).

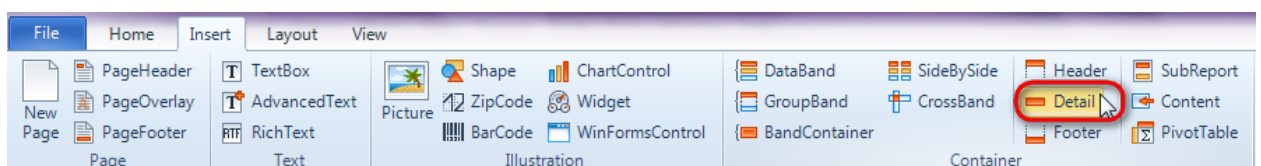
Step 22

Add one more TextBox element to the header2 band. Set properties Text = Back, Hyperlink = Back.



Step 23

Press "Detail" button on the Insert tab in the group Container.

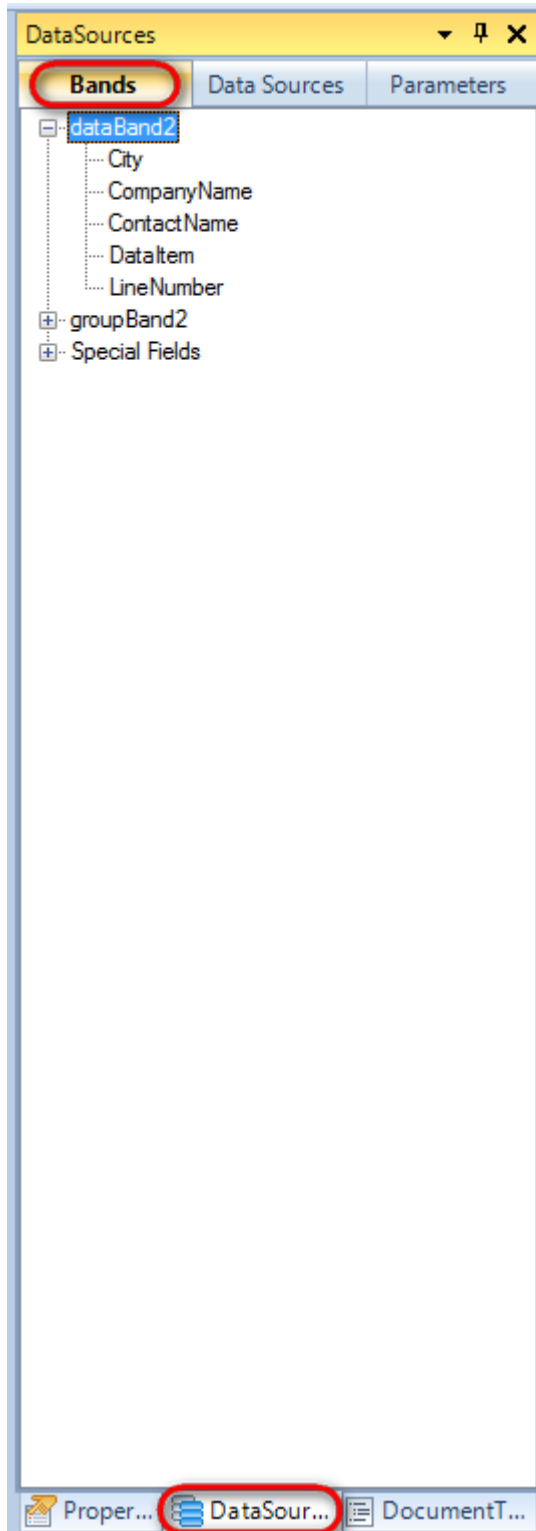


Click on the groupBand2 area to add Detail band inside groupBand2.



Step 24

Go to "DataSources" tab.



Drag and drop "CompanyName" and "ContactName" fields from the dataBand1 to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.

Step 25

Select the first TextBox element. Click "Bindings" button, set property Hyperlink = "http://www." + dataBand2["CompanyName"].ToString() + ".com". Select the second



TextBox element, set property Hyperlink = "mailto:" + dataBand2["ContactName"].ToString() + "@" + dataBand2["CompanyName"].ToString() + ".com".

The second template page should look as follows:

dataBand2:DataBand DataSource = Customers	
groupBand2:GroupBand Group = dataBand2["CompanyName"].ToString().Substring(0,1)	
header2:Header	
<dataBand2["CompanyName"].ToString().Subst	Back
detail1:Detail	
<dataBand2["CompanyName"]>	<dataBand2["ContactName"]>
end of groupBand2	
end of dataBand2	

Step 26

Save template, close Report Designer.

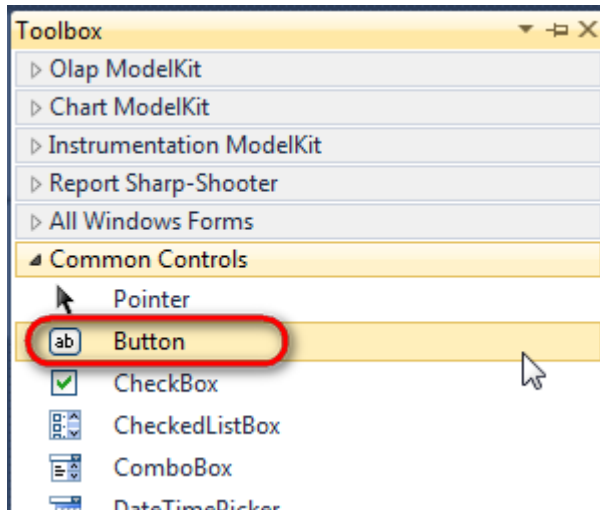
Step 27

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

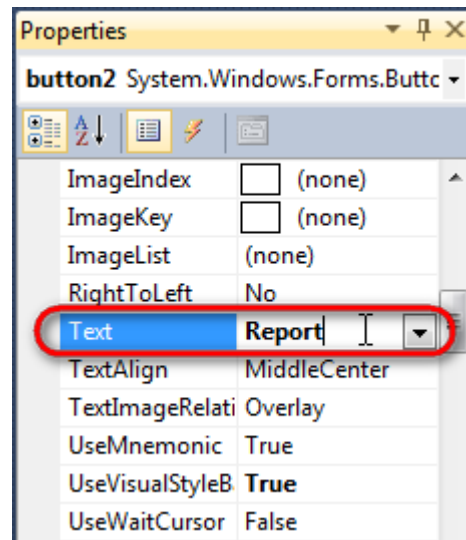
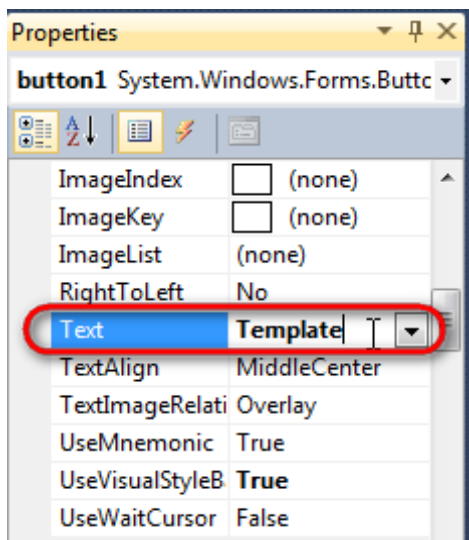
```
public Form1()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["City"] = "Boston";
    row["ContactName"] = "Maria Anders";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["City"] = "London";
    row["ContactName"] = "Ana Trujillo";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["City"] = "Paris";
    row["ContactName"] = "Roland Mendel";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["City"] = "Moscow";
    row["ContactName"] = "Karin Josephs";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
    EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
    PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 28

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



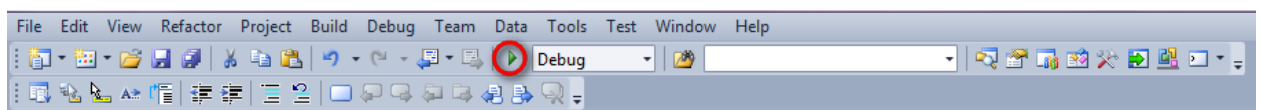
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

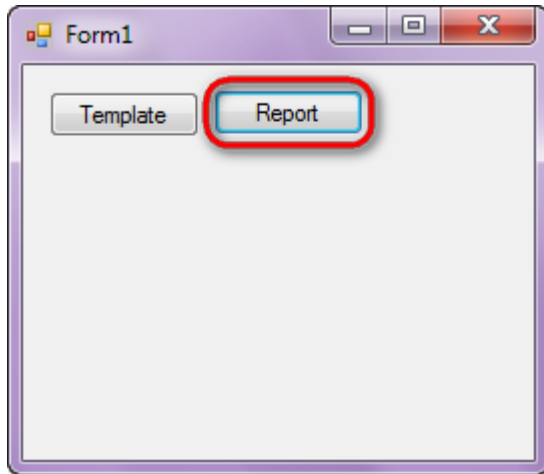
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 29

Click “Start Debugging” on the Visual Studio toolbar in order to start application.

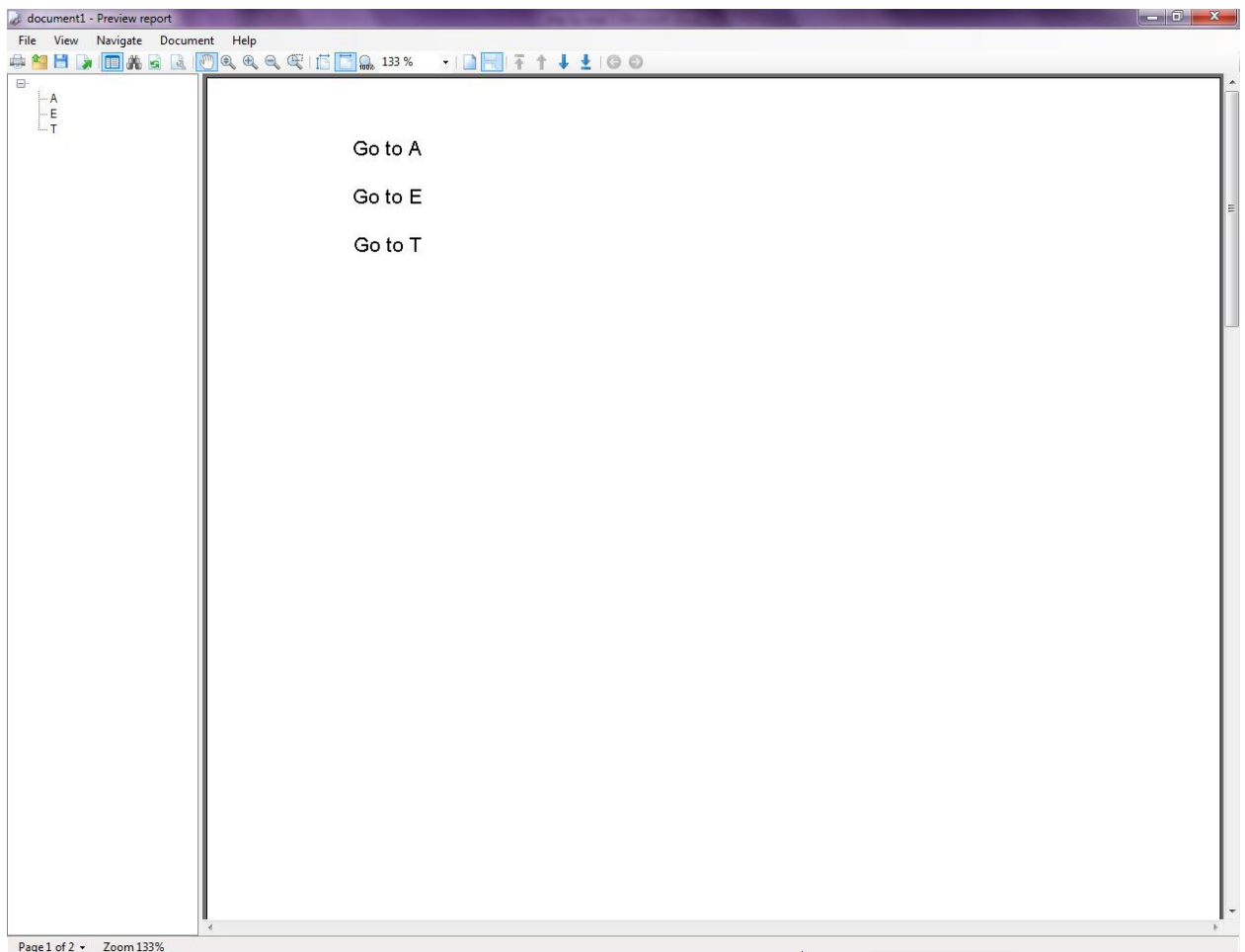


Click the “Report” button in the opened application window.



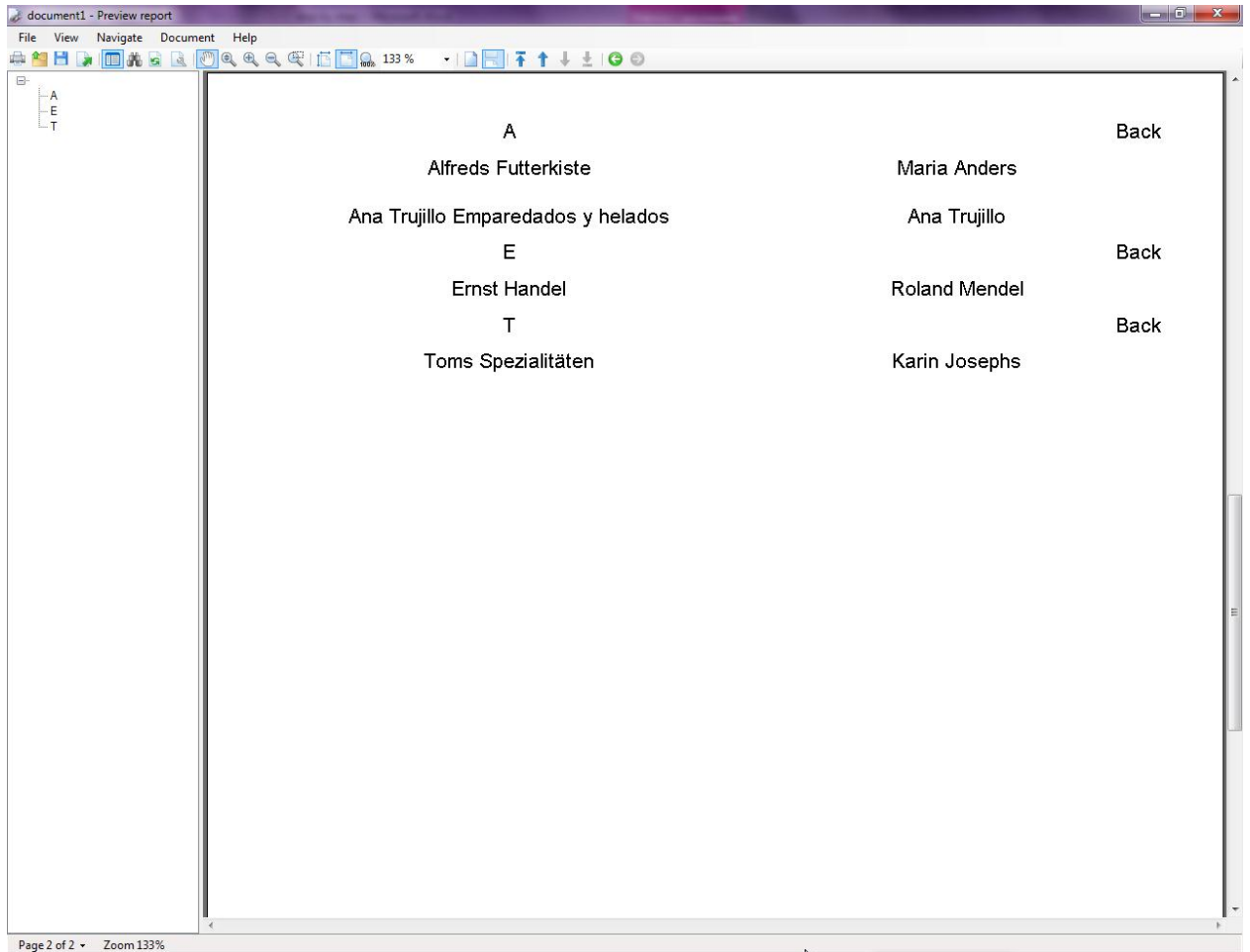
Generated report is viewed in the Report Viewer.

The first page:



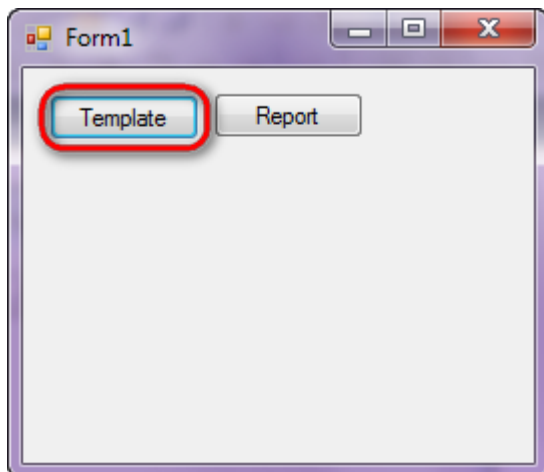
You can navigate through the report using hyperlinks – click on the hyperlink to open the corresponding group.

The second page:



Click on the company name and you will open company webpage in the browser. Click on the contact name and mail client will open to send email to this person.

To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Reports\Special features\Hyperlinks & Bookmarks.

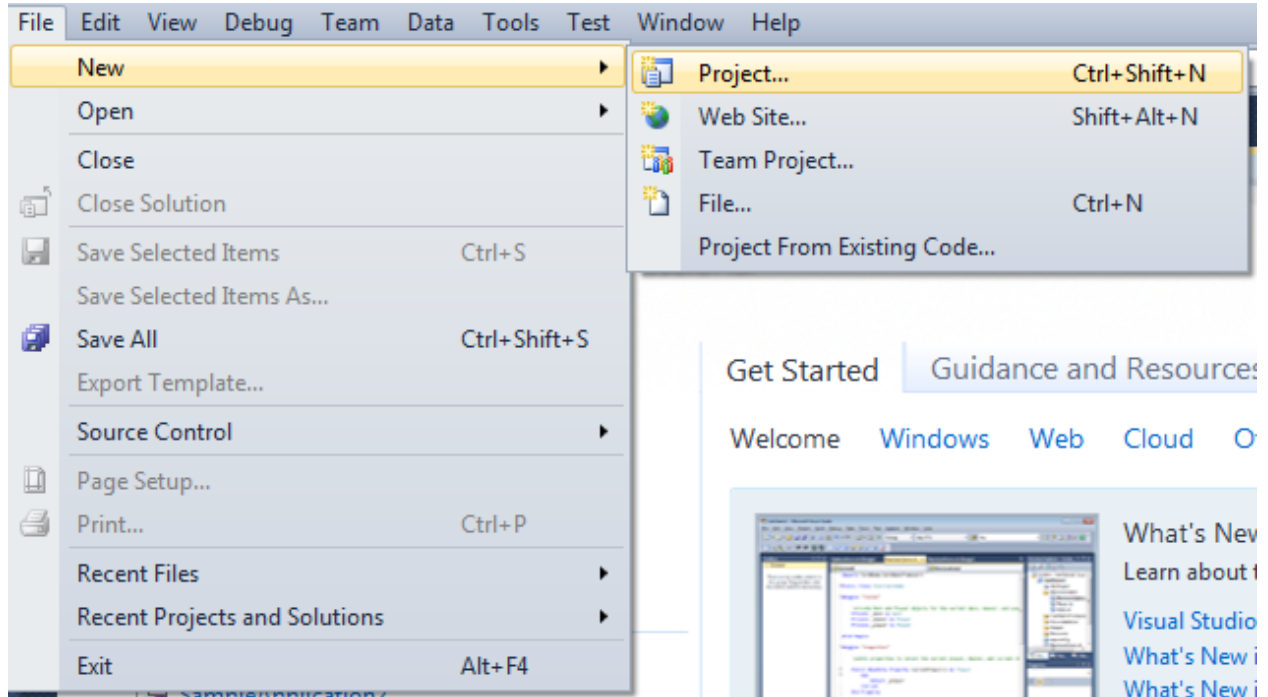


Totals

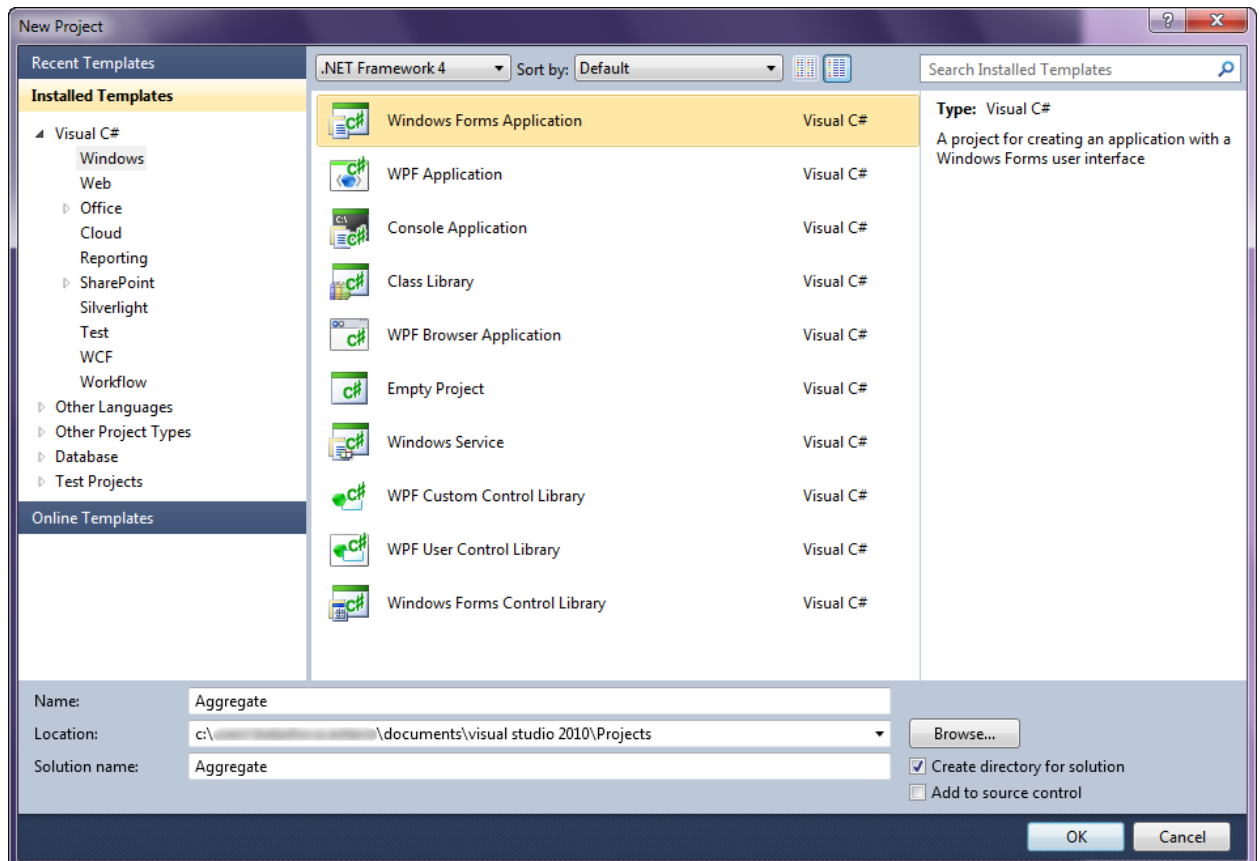
Template of a report containing a list of products, price of every product and total order price.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

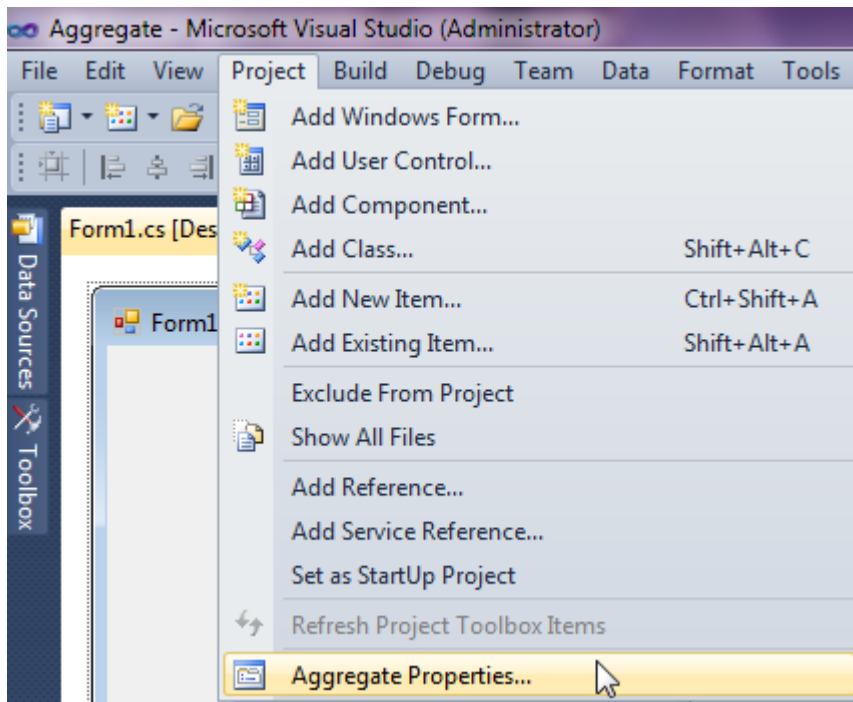


Select Windows Forms Application, set project name – “Aggregate”, set directory to save the project to.

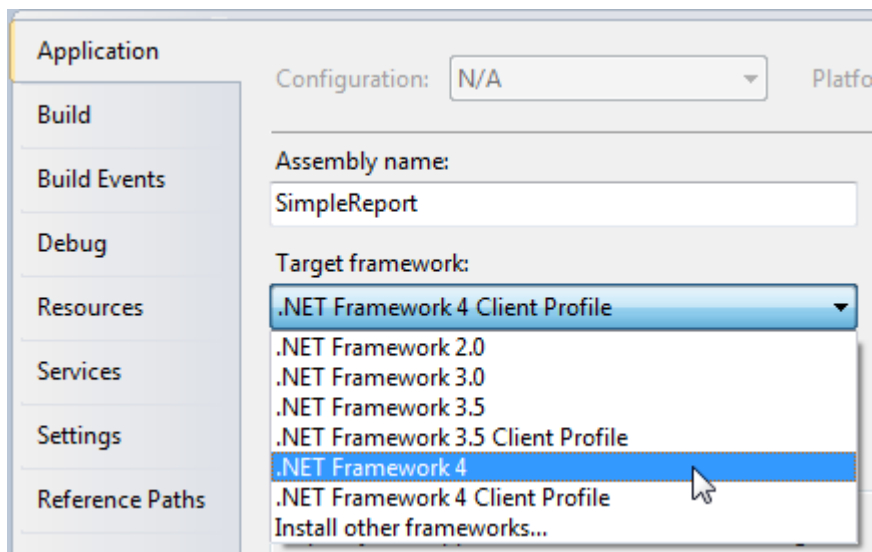


Step 2

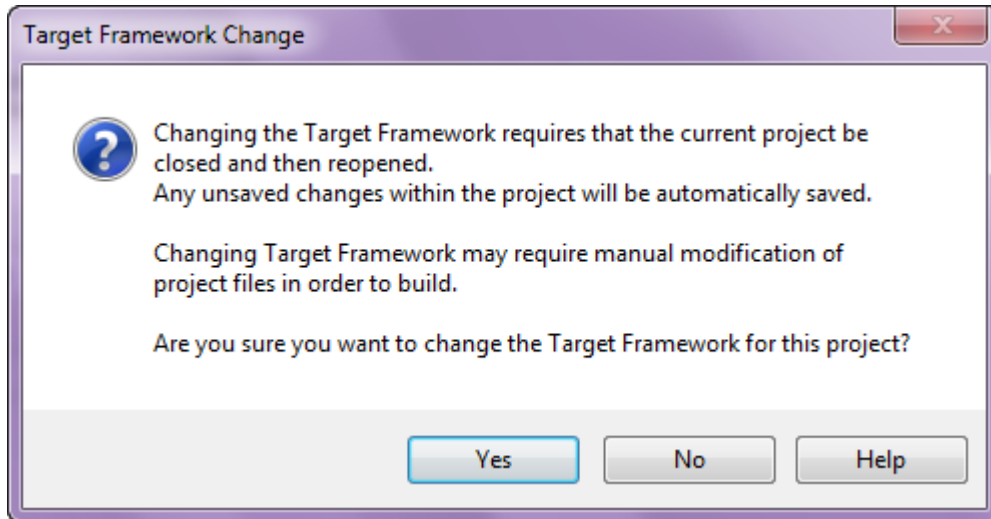
Change the project properties. Select the Project\Aggregate Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

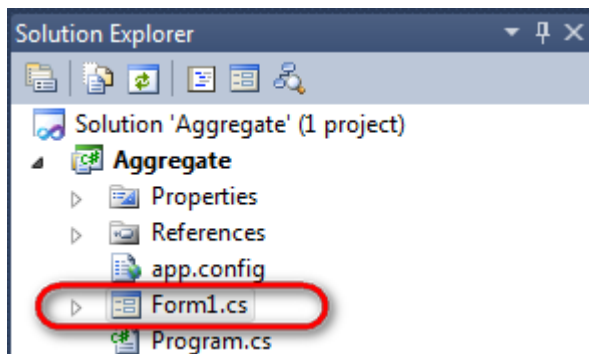


In the opened window press the "Yes" button.

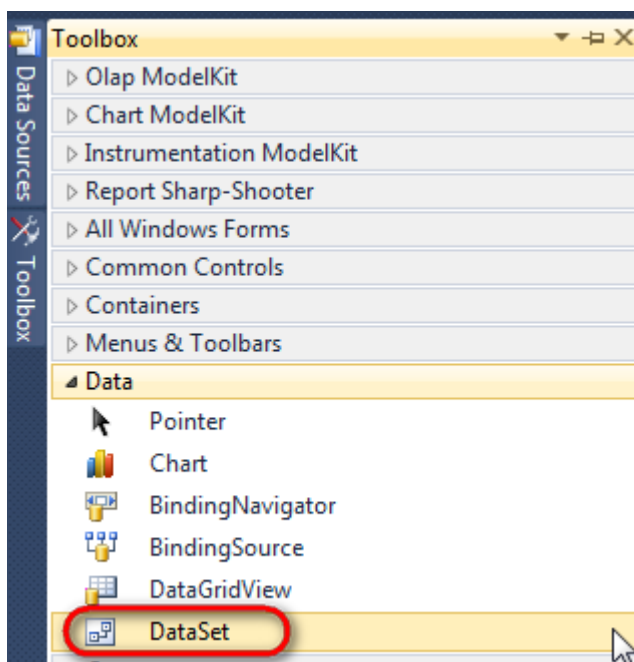


Step 3

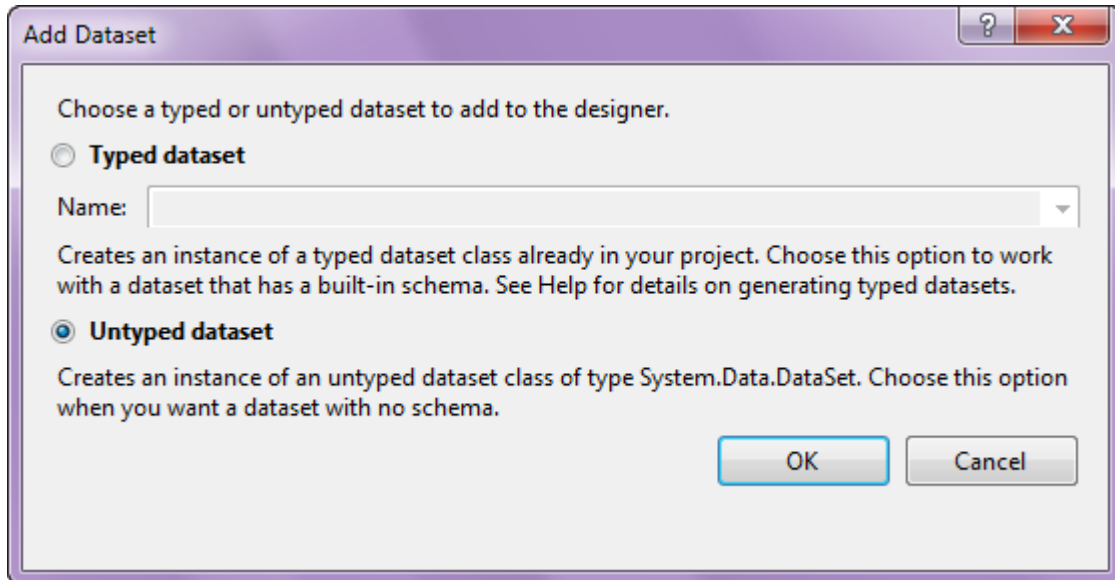
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



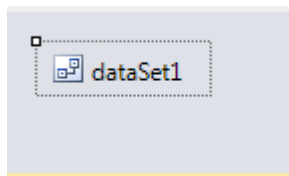
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

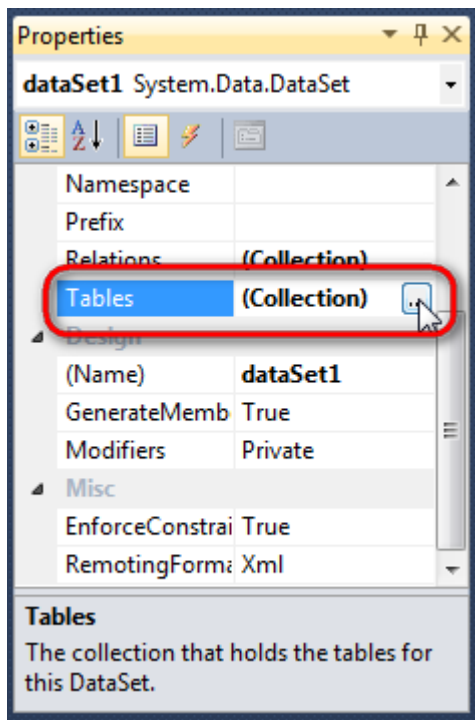


The component is available in the lower part of the window.

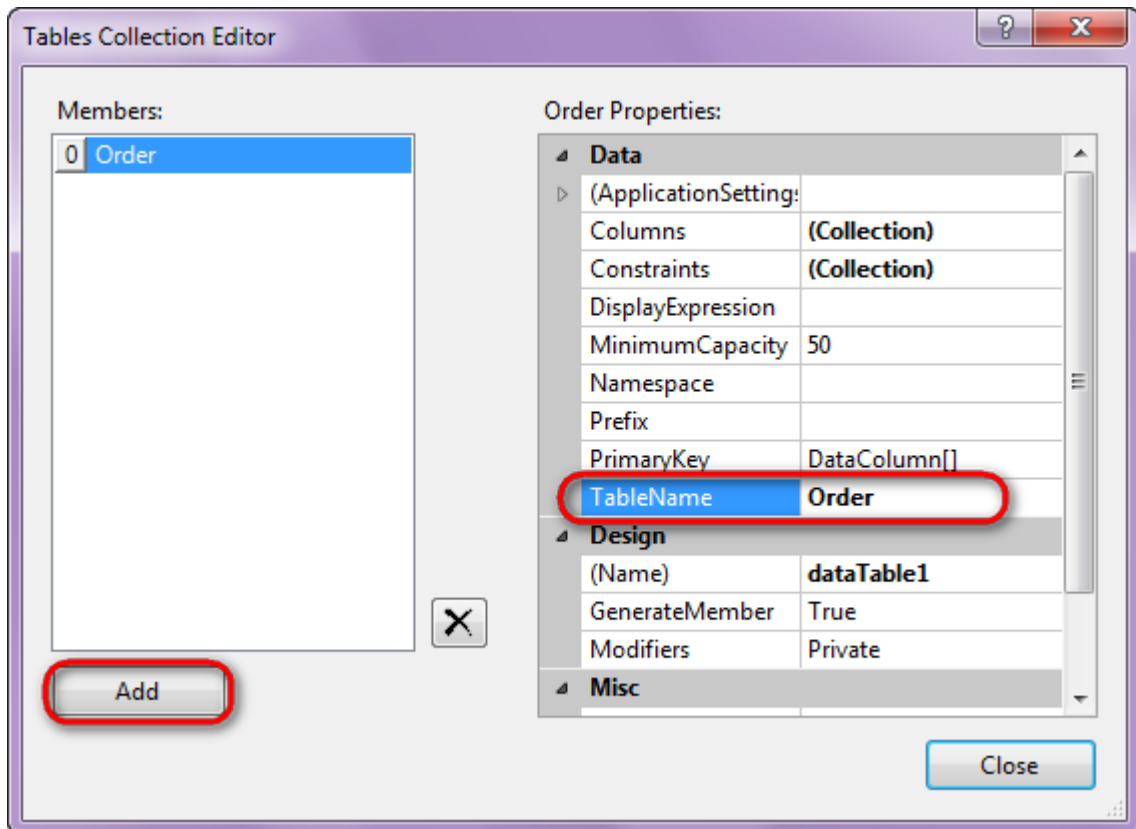


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

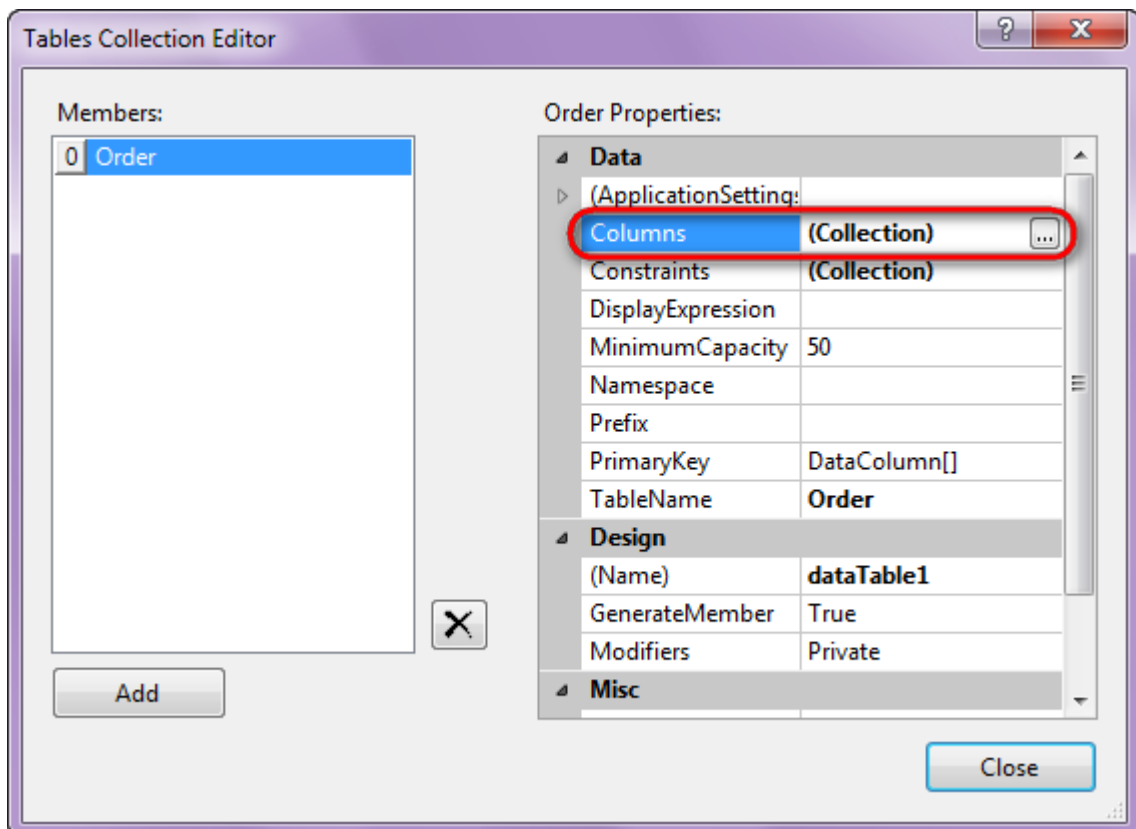


Click "Add" in order to add table. Set property TableName = Order.

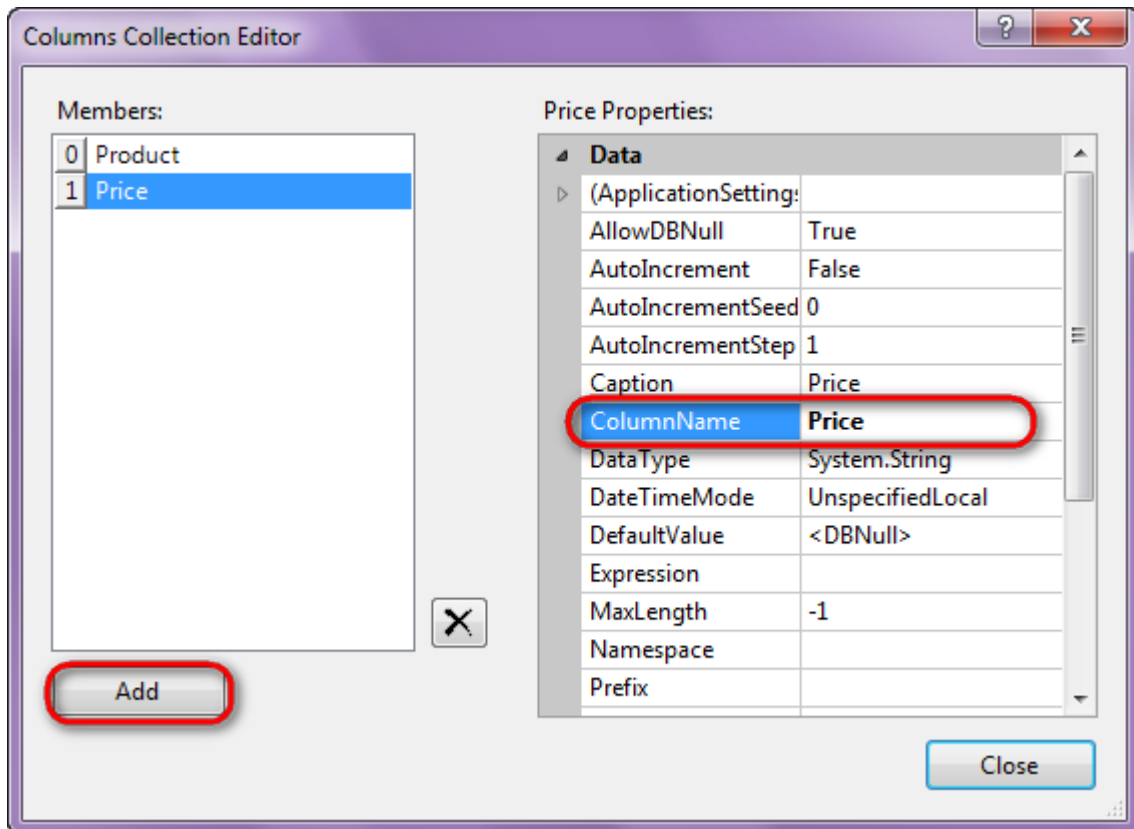


Step 5

Select Columns property, click button  in order to open property editor.

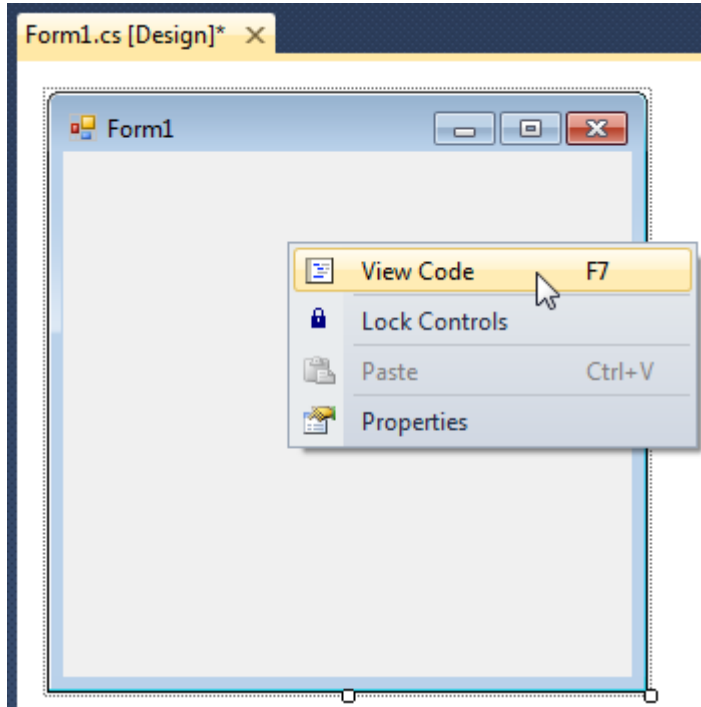


Click "Add" to add a new column. Add two columns. Set ColumnName property to "Product", "Price" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu to view code.



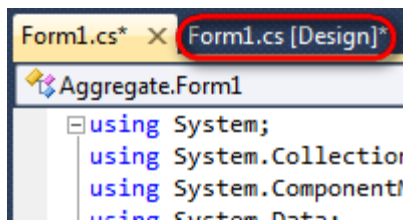
Add the following code to the class constructor in order to fill data source.

```
public Form1 ()  
{  
    InitializeComponent ();  
    DataRow row = dataTable1.NewRow ();  
    row["Product"] = "Chai";  
}
```

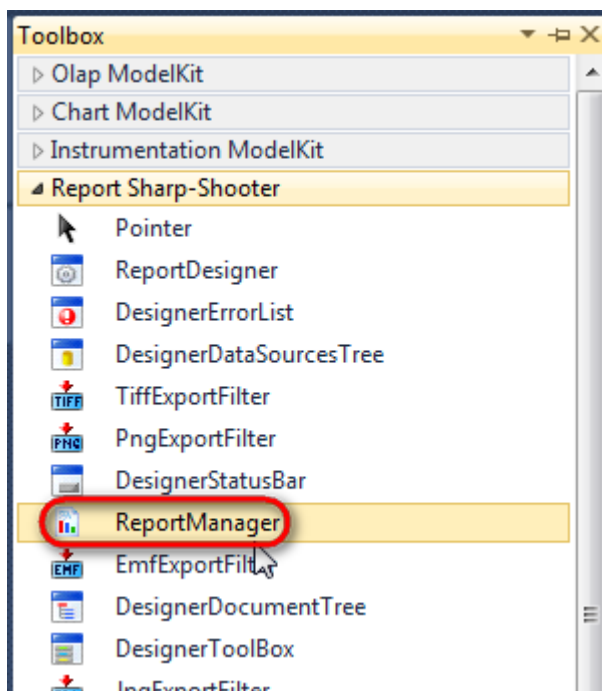
```
row["Price"]="14.40";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["Product"] = "Chang";  
row["Price"] = "15.20";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["Product"] = "Ipoh Coffee";  
row["Price"] = "46.00";  
dataTable1.Rows.Add(row);  
}
```

Step 7

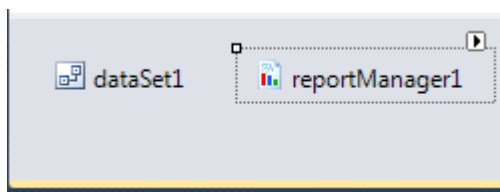
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

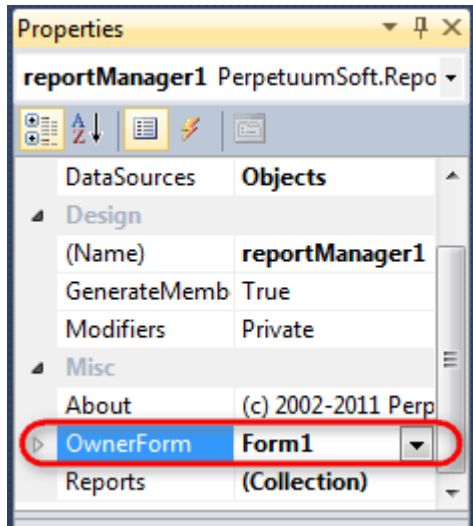


The component is available in the lower part of the window.



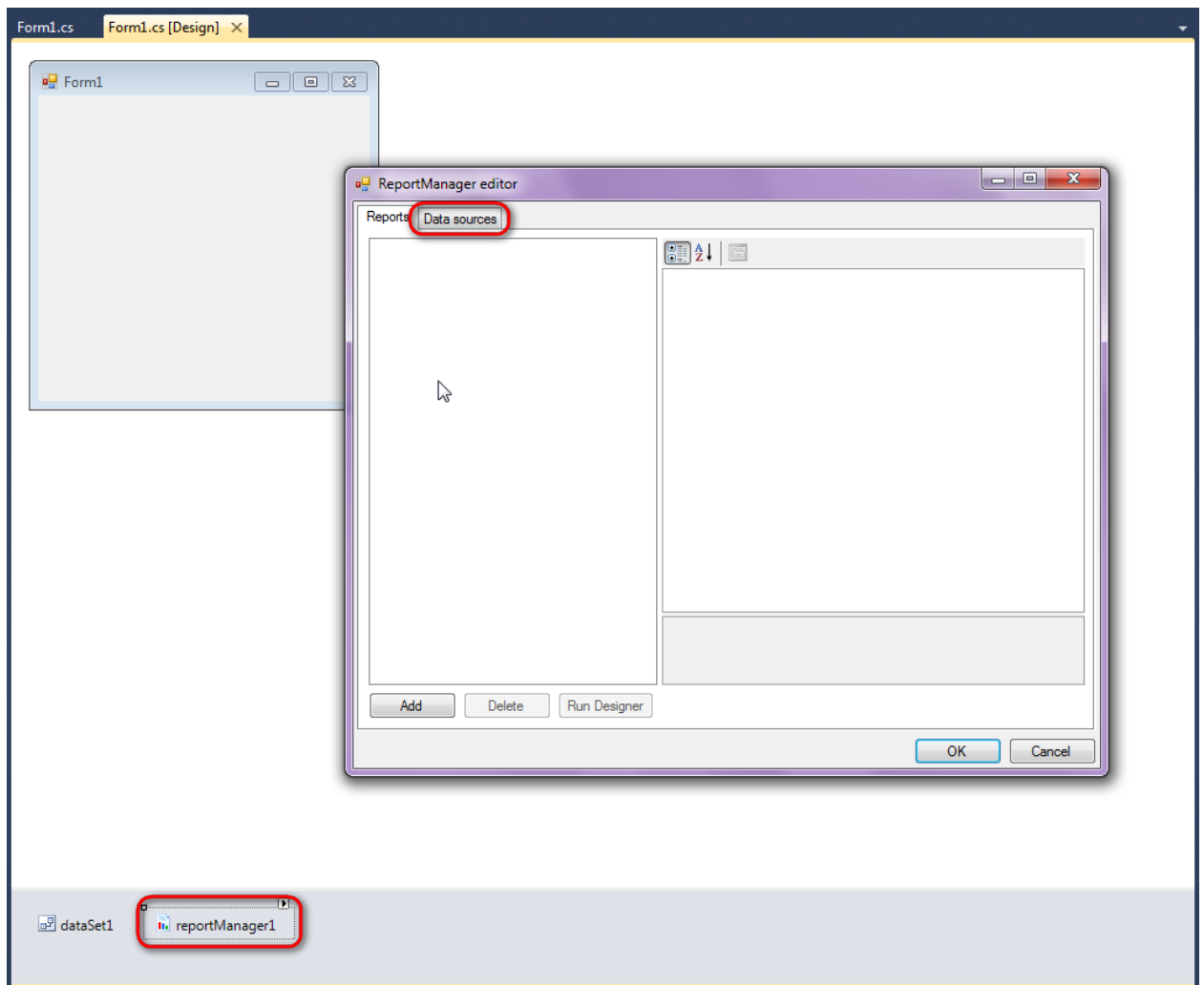
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

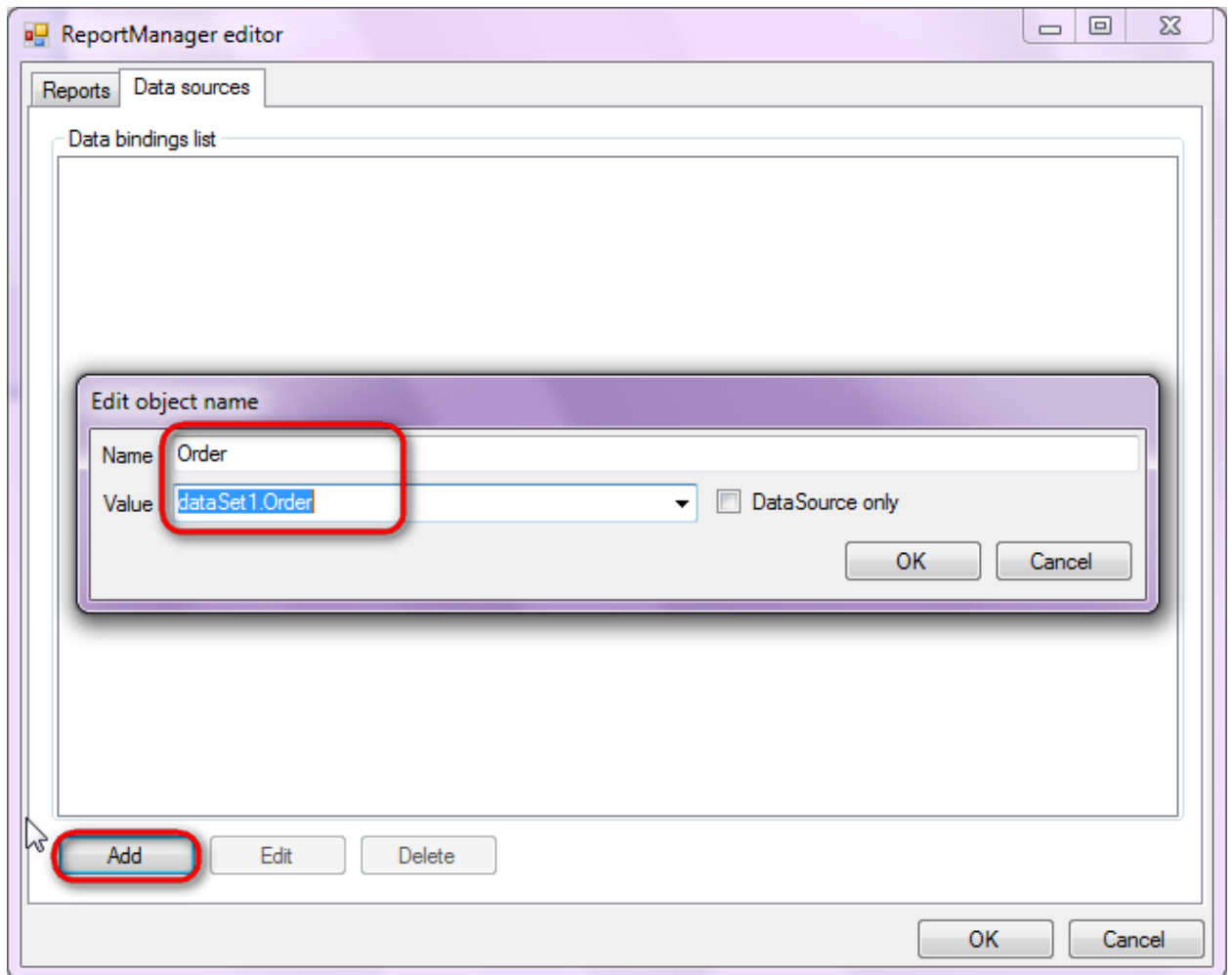


Step 9

Double click on ReportManager to open ReportManager editor.

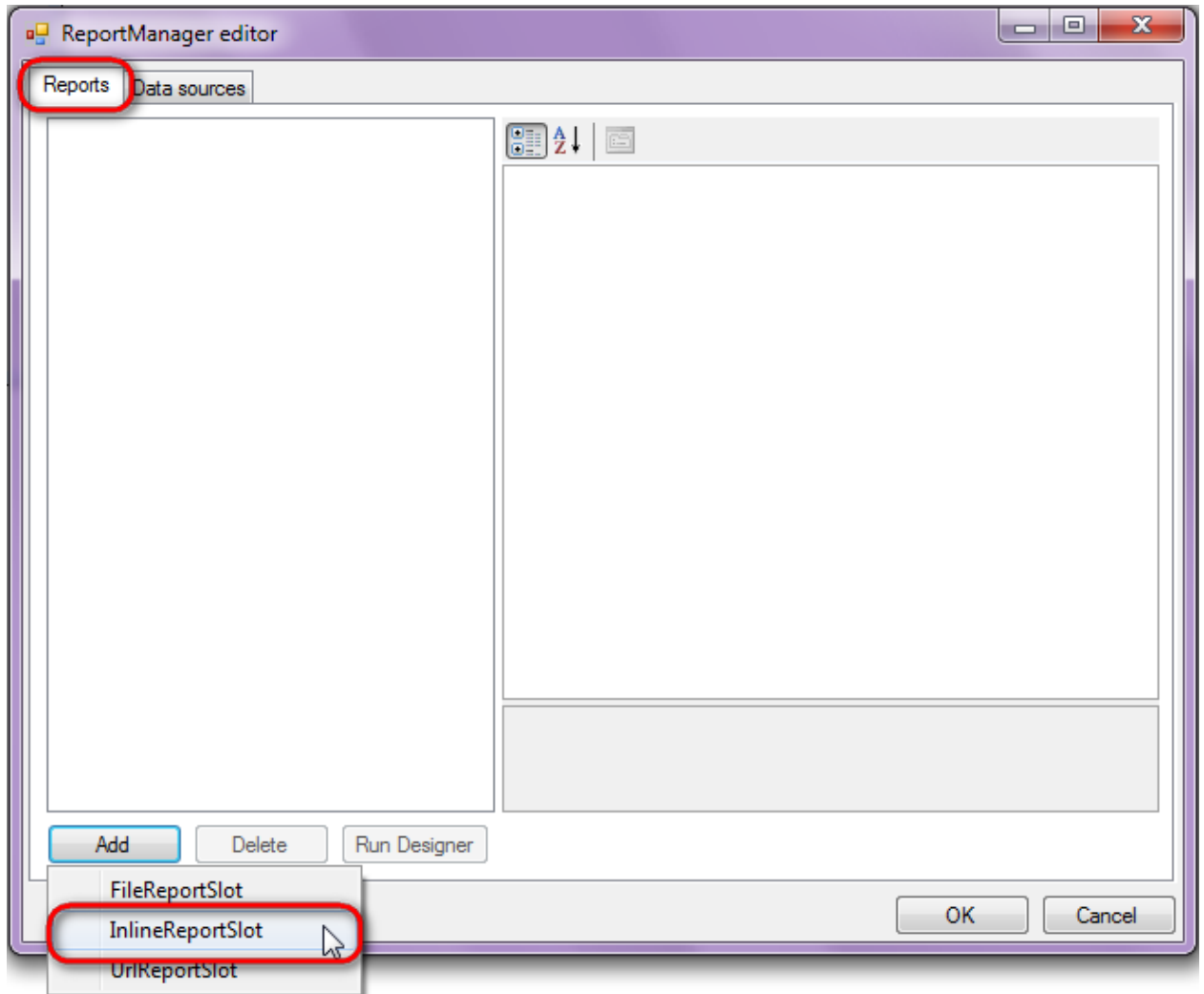


Go to "Data sources" tab, click "Add", set data source name - "Order", select data source value - "dataSet1.Order".



Step 10

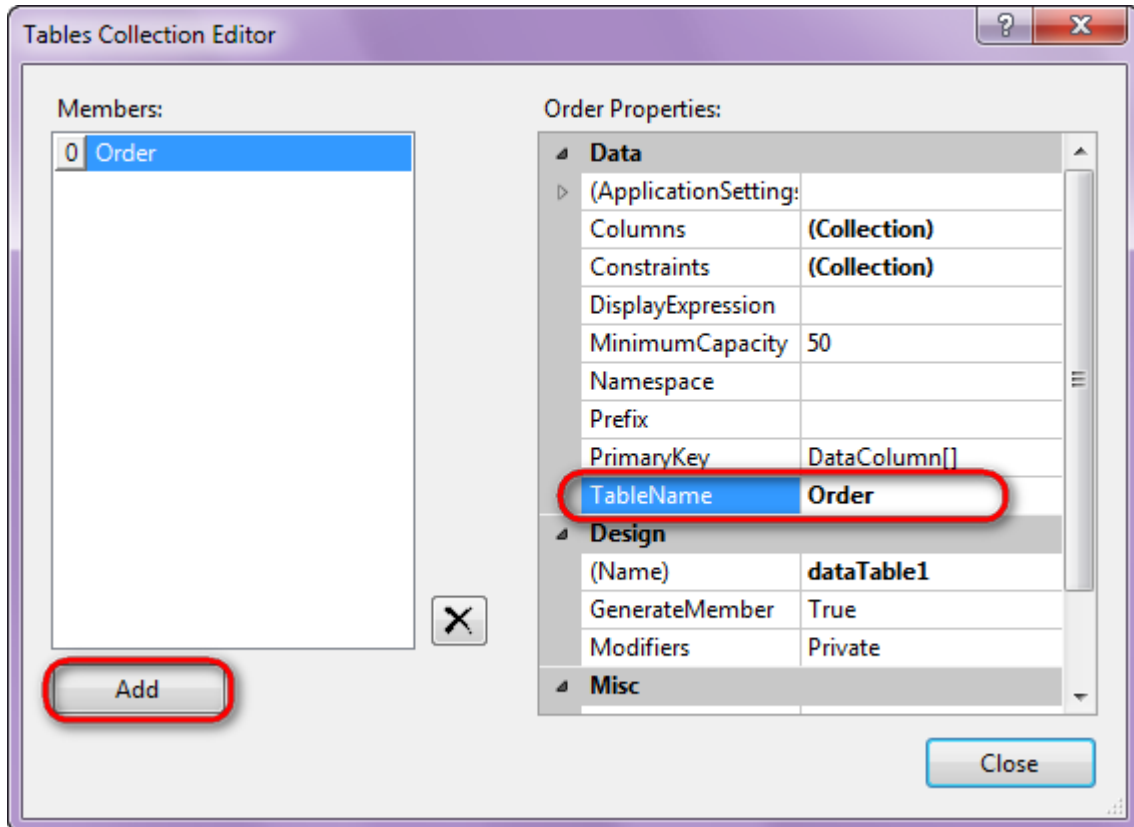
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

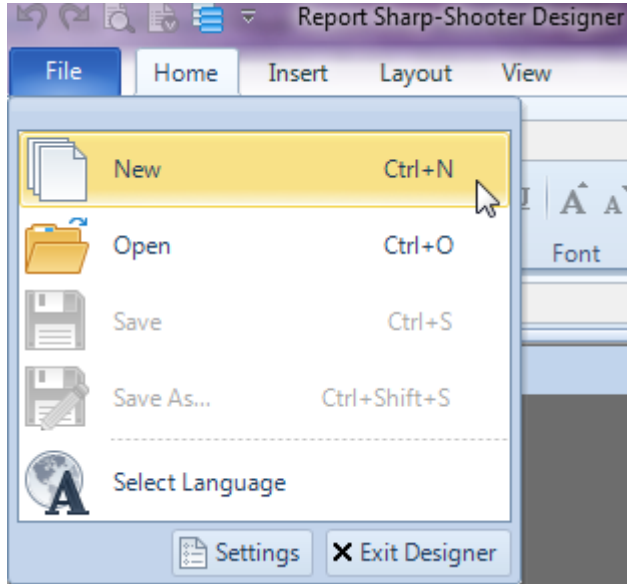
Set name of the report in the property ReportName – “Aggregate”.

Click “Run Designer” in order to open template editor - Report Designer.

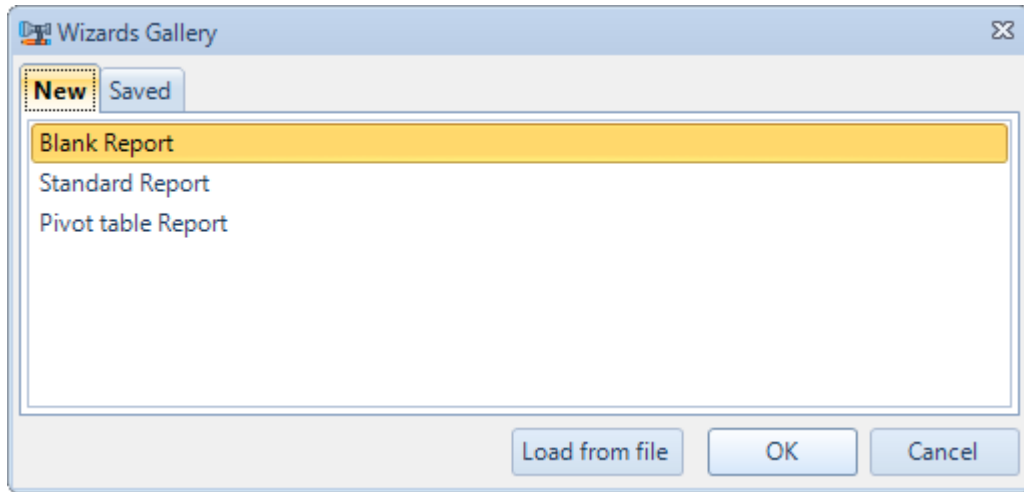


Step 12

Create new empty template – select File\New from the main menu.

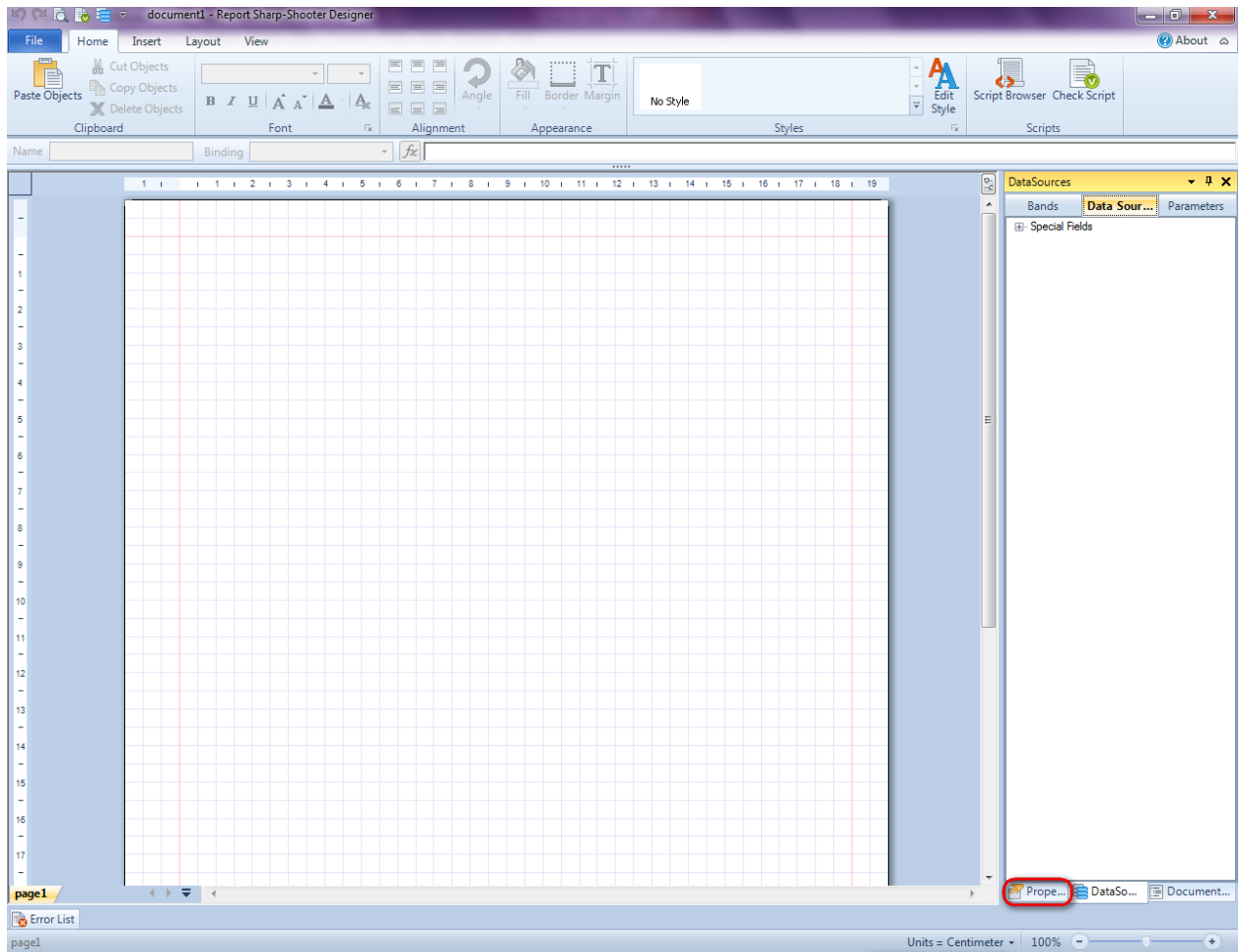


Select "Blank Report" in the Wizards Gallery and click "OK".

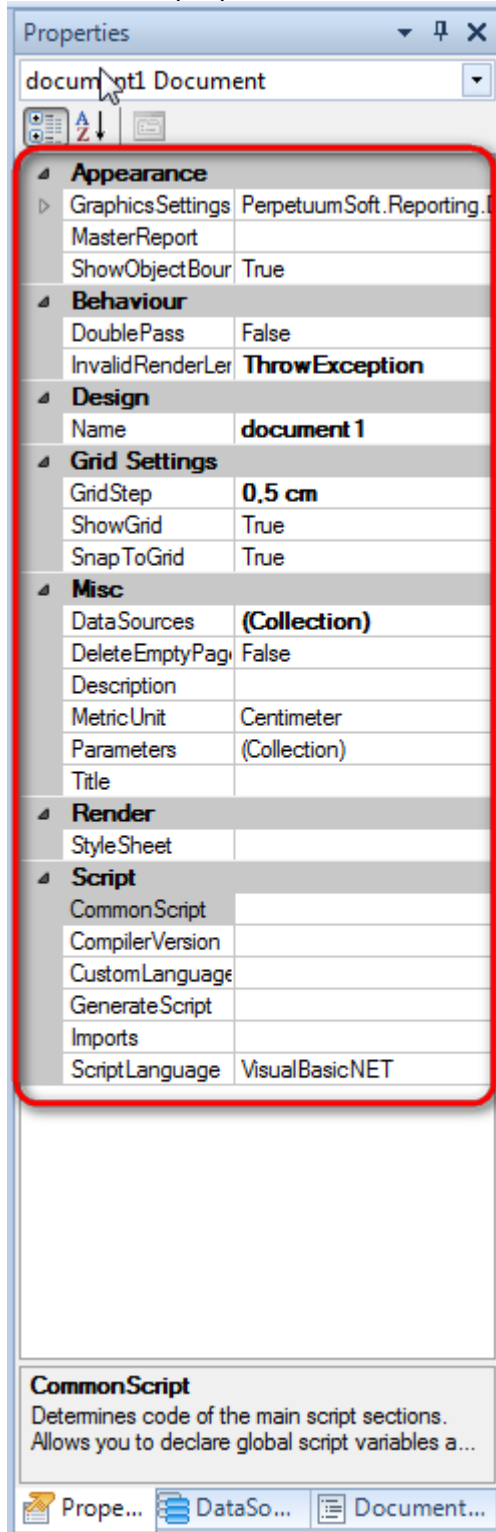


Step 13

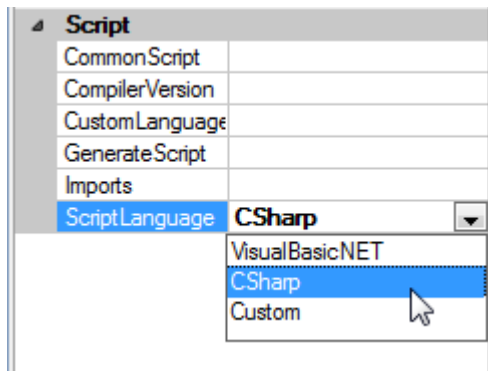
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

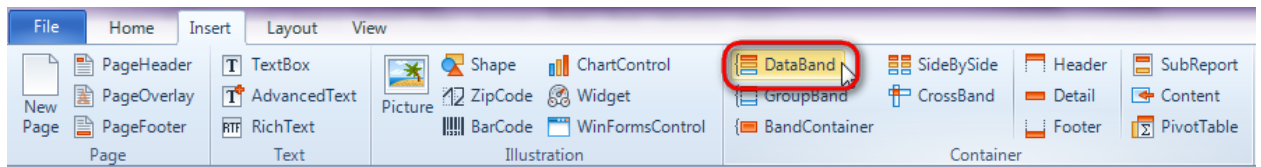


Set property ScriptLanguage = CSharp.



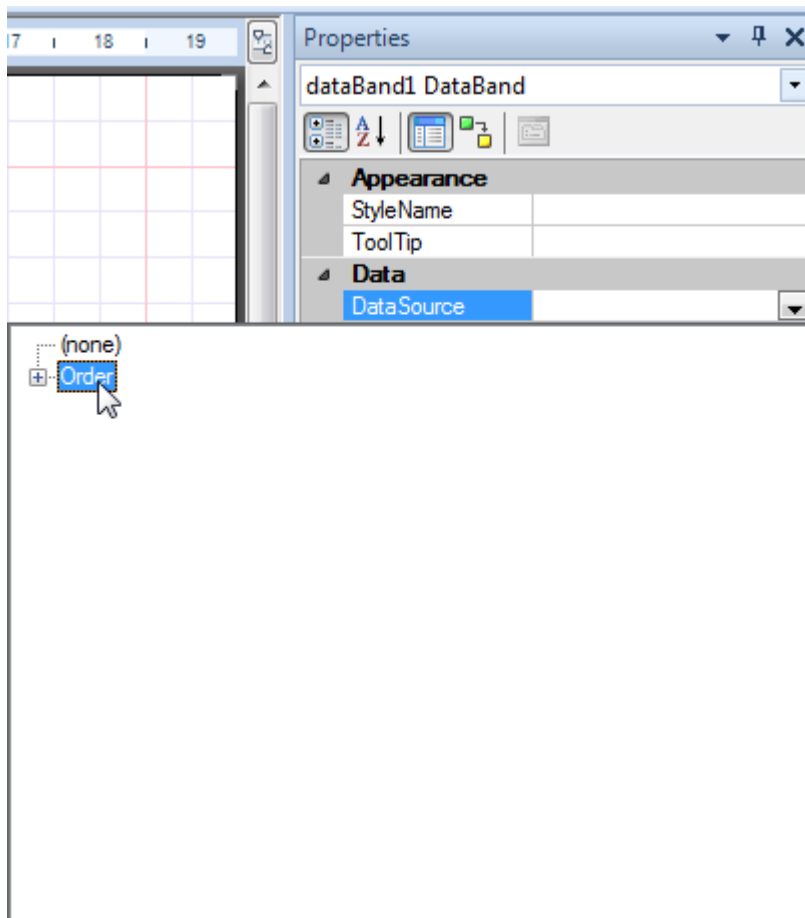
Step 14

Press "DataBand" button on the Insert tab in the group Container.




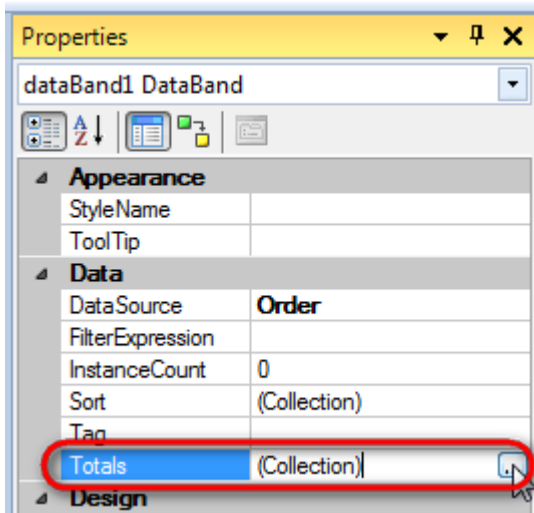
Click on the template area to add DataBand band to the template.


Set data source in the property DataSource = Order.

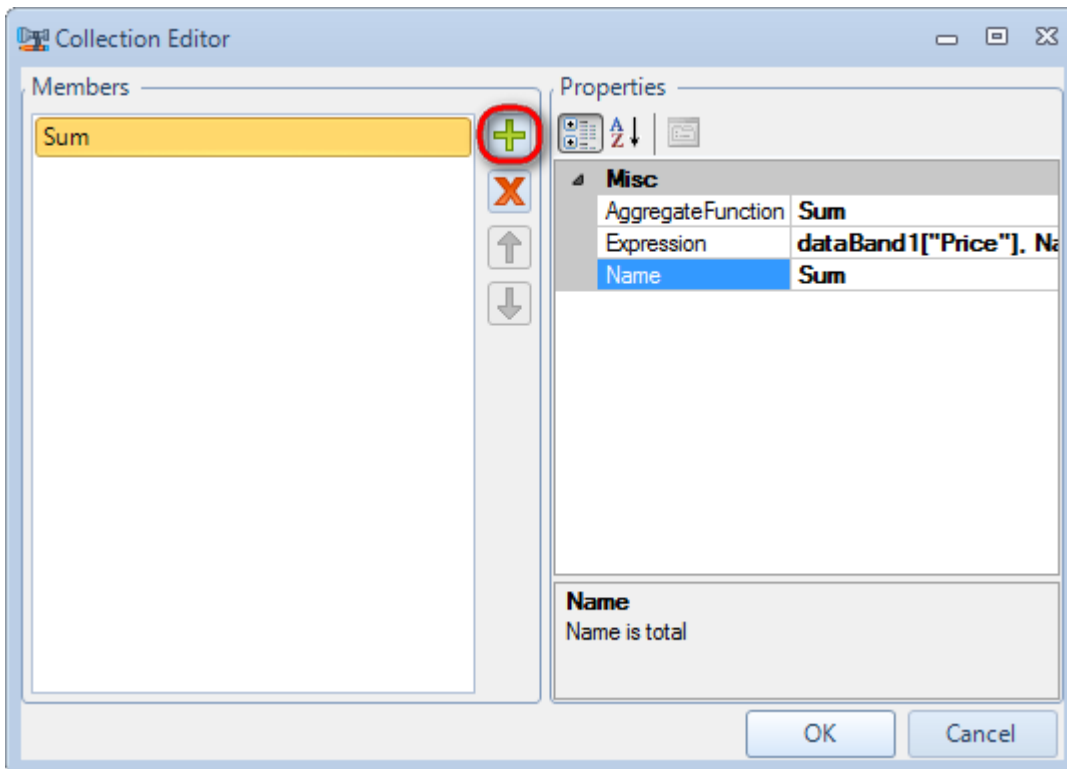


Step 15

Select Totals property, click button  to open editor of the Total property collection.

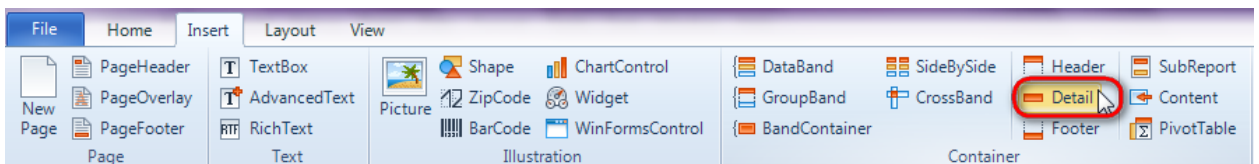


In the Collection editor, click  button to add aggregation function. Set the following properties: AggregateFunction = Sum, Expression = dataBand1["Price"], Name = Sum.



Step 16

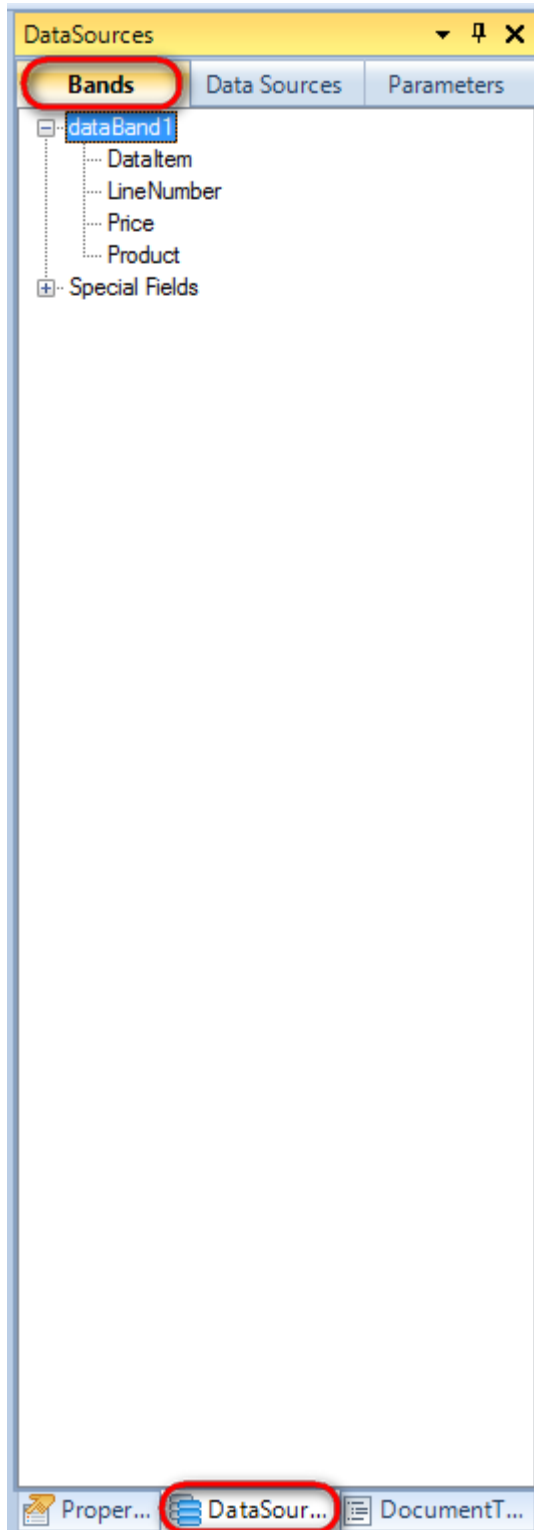
Press "Detail" button on the Insert tab in the group Container.



Click on the DataBand area to add Detail band inside DataBand.

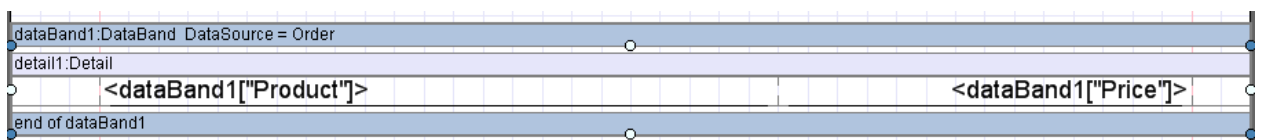
Step 17

Go to "DataSources" tab.



Drag and drop "Product" and "Price" fields from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.

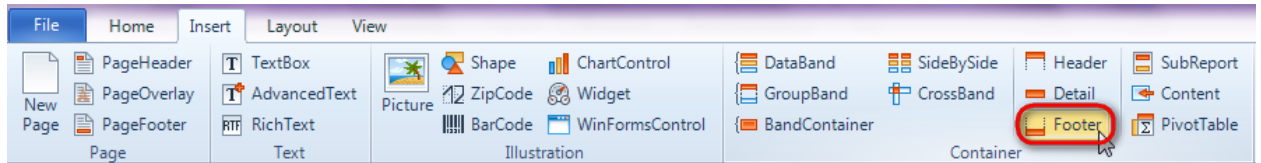
Change size of the elements and locate them in the way shown in the picture below.





Step 18

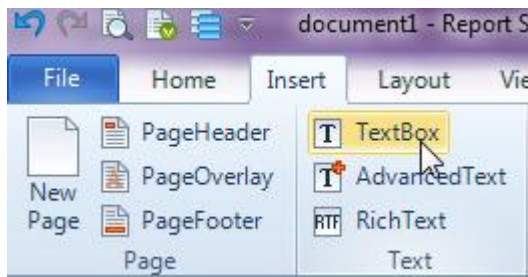
To display footer containing summarized value of the Price field, create Footer band – press “Footer” button on the Insert tab in the group Container.



Click on the DataBand area to add Footer inside DataBand.

Step 19

Press button “TextBox” on the Insert tab in the group Text.



Click on the Footer band area to add TextBox element inside Footer.

Set Value property to “GetTotal(“Sum”)”.



Report Template:

dataBand1:DataBand DataSource = Order	
detail1:Detail	
<dataBand1["Product"]>	<dataBand1["Price"]>
footer1:Footer	
	<GetTotal("Sum")>
end of dataBand1	

Step 20

Save template, close Report Designer.

Step 21

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

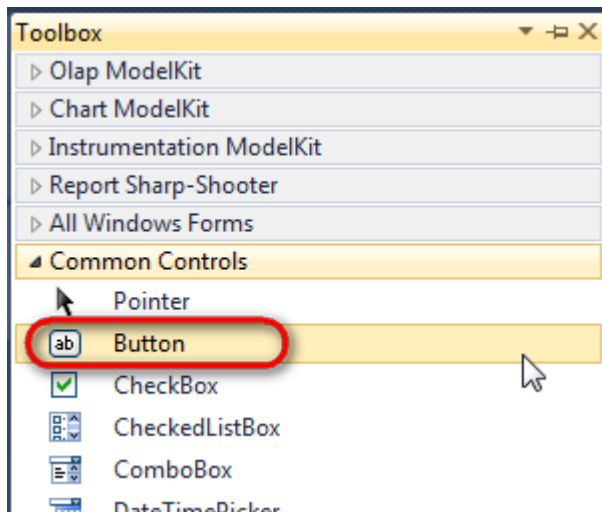
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row ["Product"] = "Chai";
    row ["Price"] = "14.40";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["Product"] = "Chang";
    row ["Price"] = "15.20";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
```



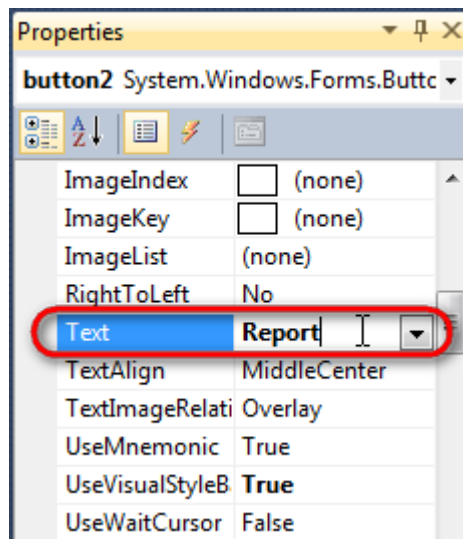
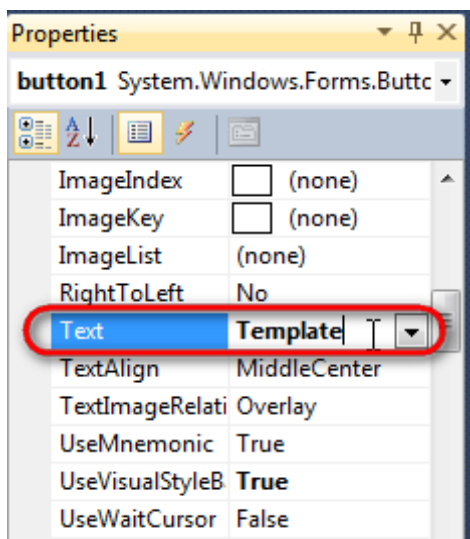
```
row["Product"] = "Ipoh Coffee";  
row["Price"] = "46.00";  
dataTable1.Rows.Add(row);  
        inlineReportSlot1.RenderCompleted += new  
EventHandler(reportSlot_RenderCompleted);  
    }  
    private void reportSlot_RenderCompleted(object sender, EventArgs e)  
    {  
        using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new  
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))  
        {  
            previewForm.WindowState = FormWindowState.Maximized;  
            previewForm.ShowDialog(this);  
        }  
    }  
}
```

Step 22

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



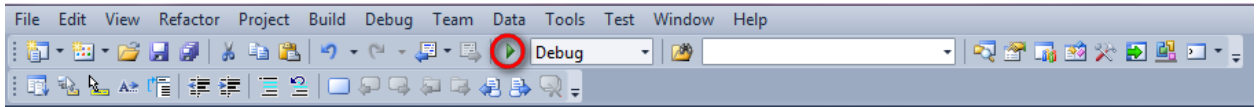
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

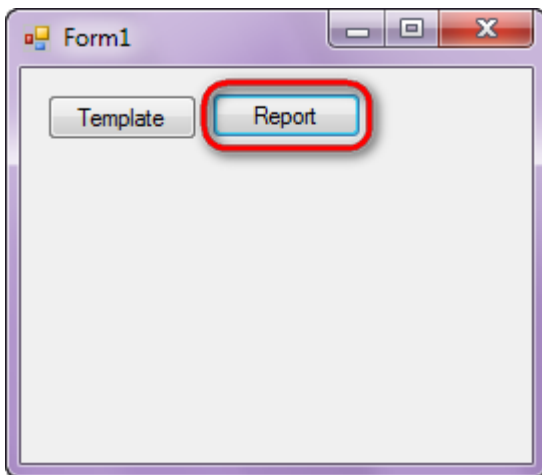
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 23

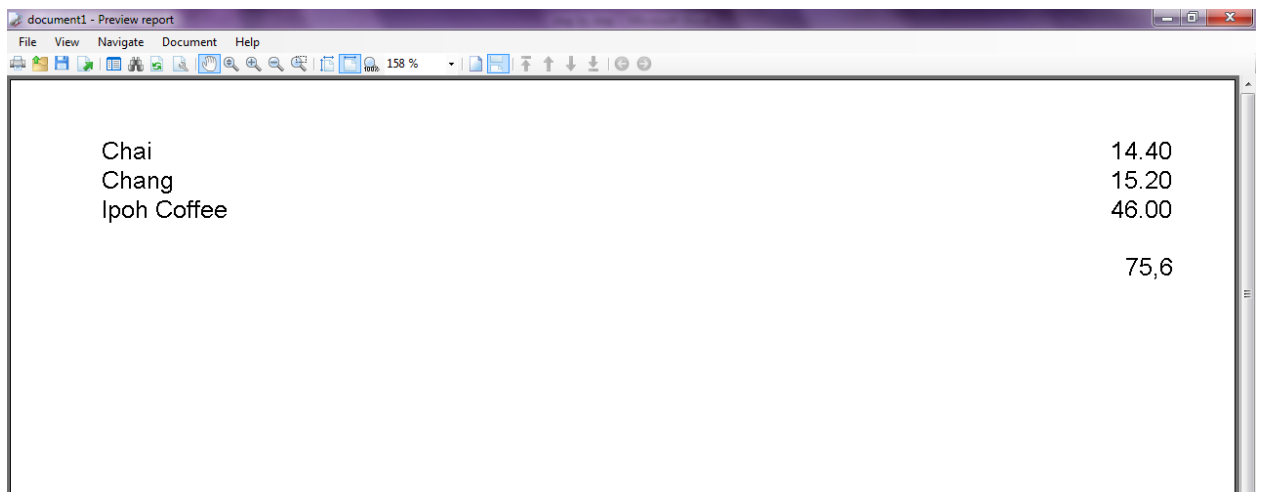
Click "Start Debugging" on the Visual Studio toolbar in order to start application.



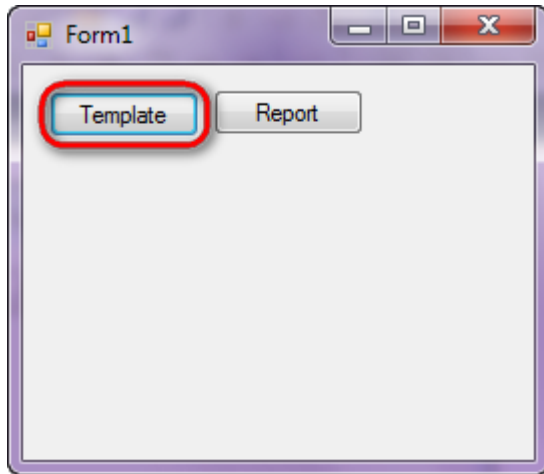
Click the "Report" button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Reports\Grouping\Group aggregates.

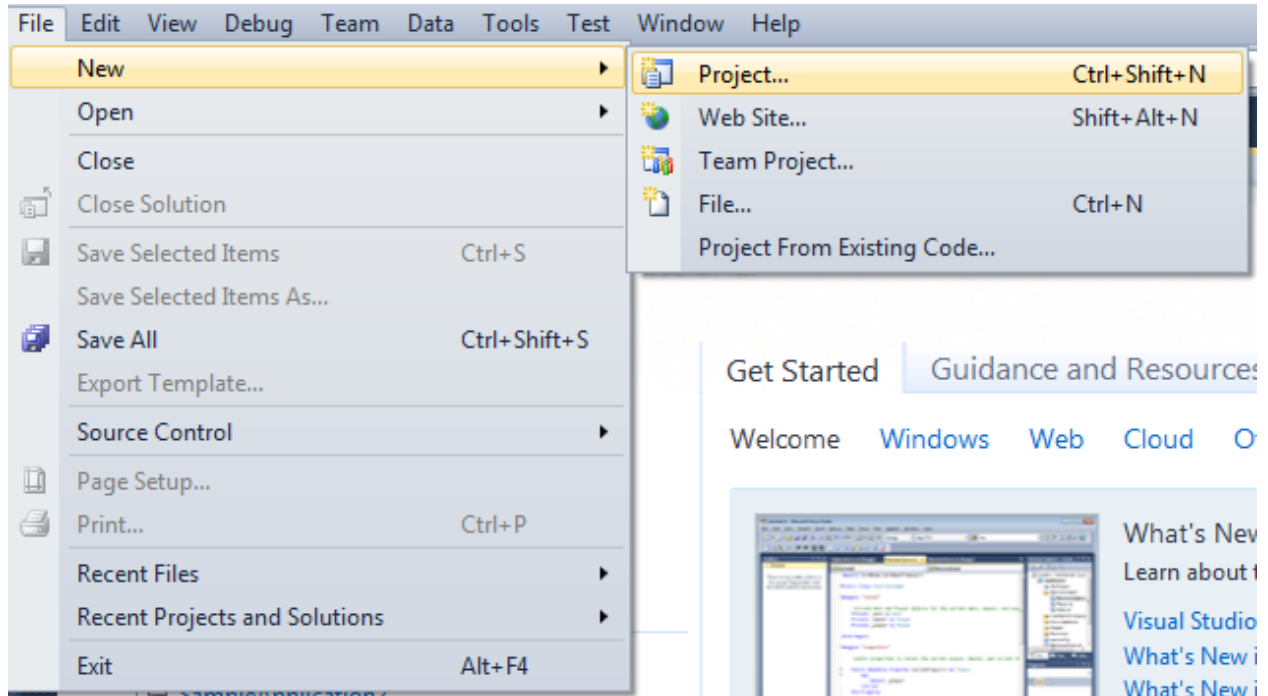


Nested Group

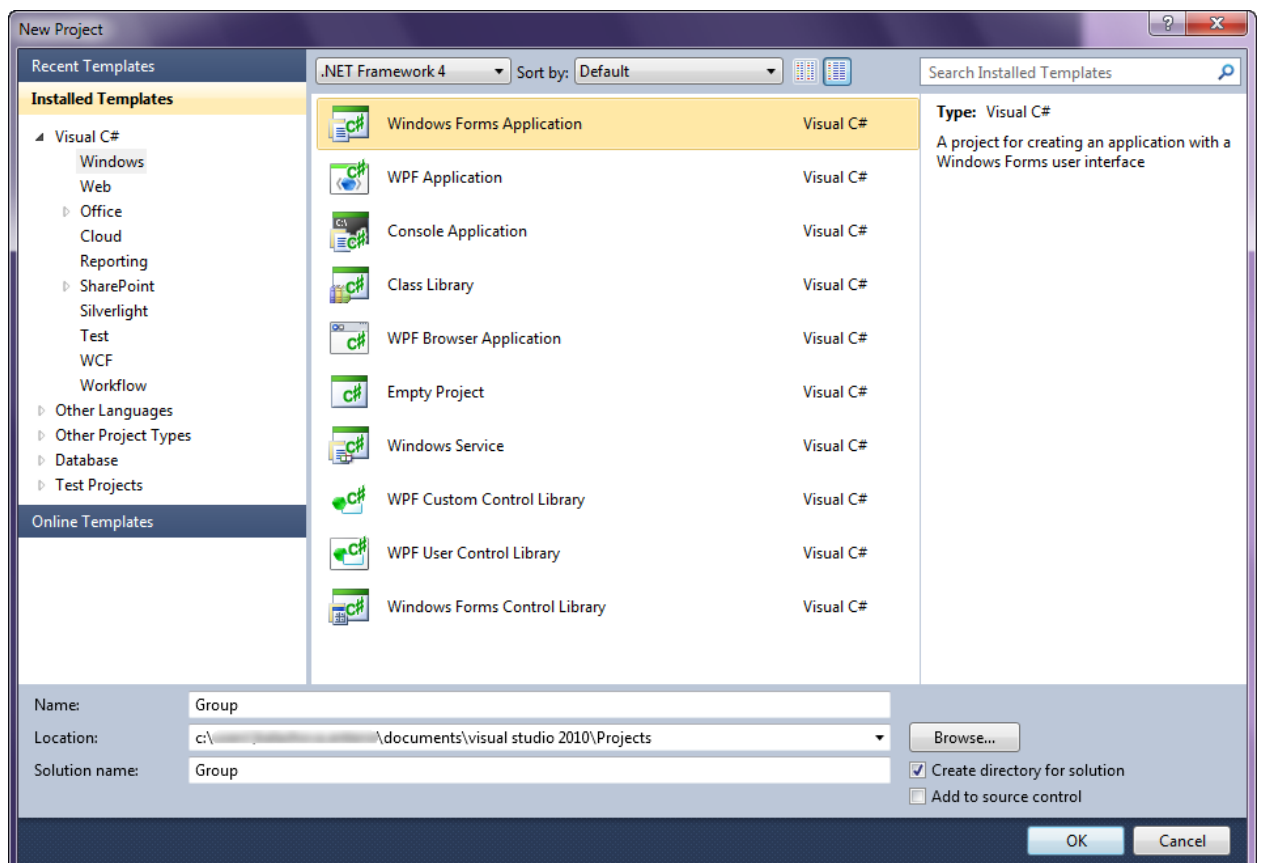
Template of a report containing a list of customers grouped by countries and cities.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

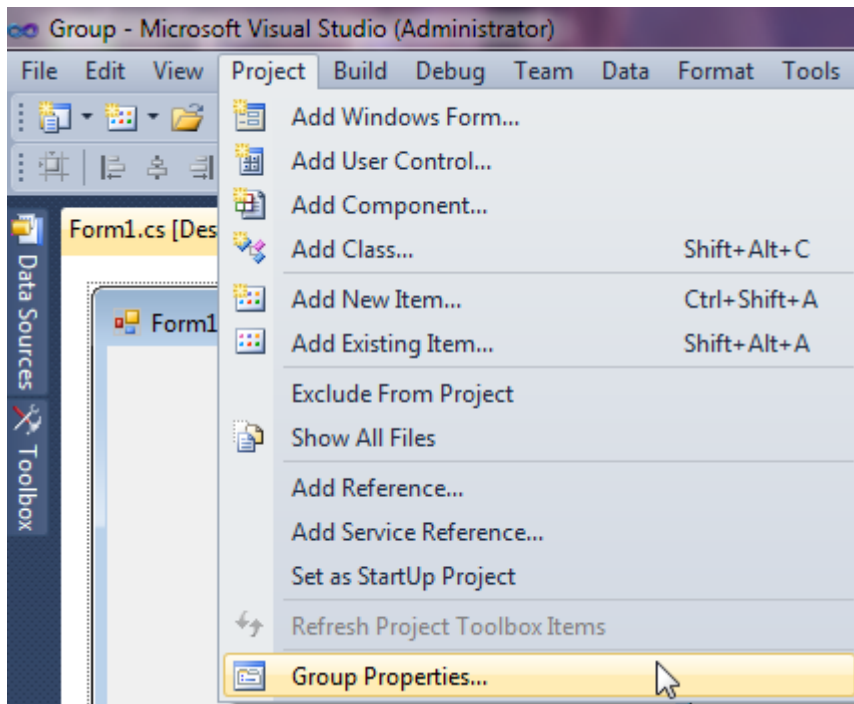


Select Windows Forms Application, set project name – “Group”, set directory to save the project to.

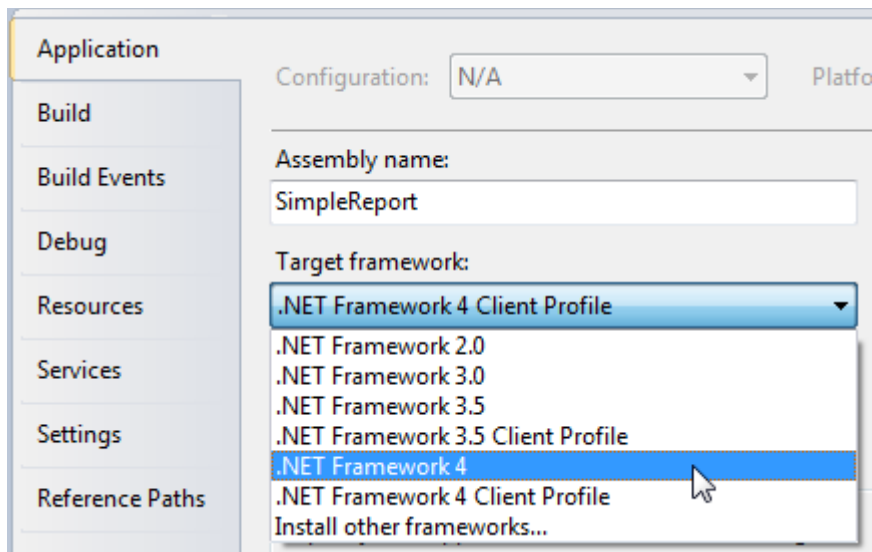


Step 2

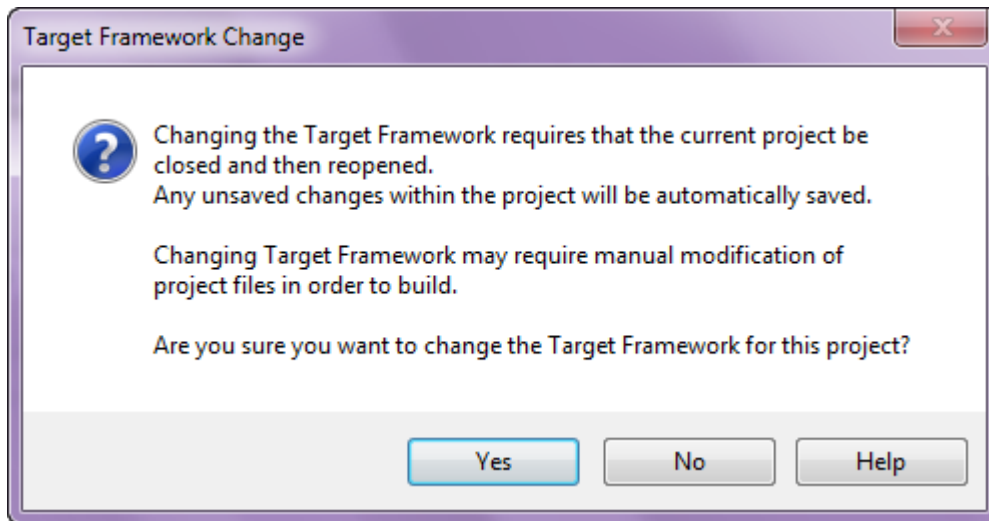
Change the project properties. Select the Project\Group Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

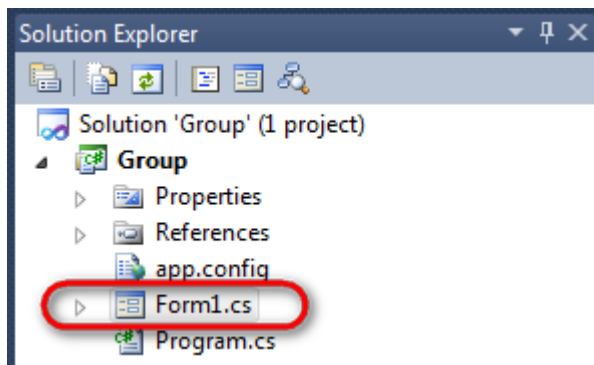


In the opened window press the "Yes" button.

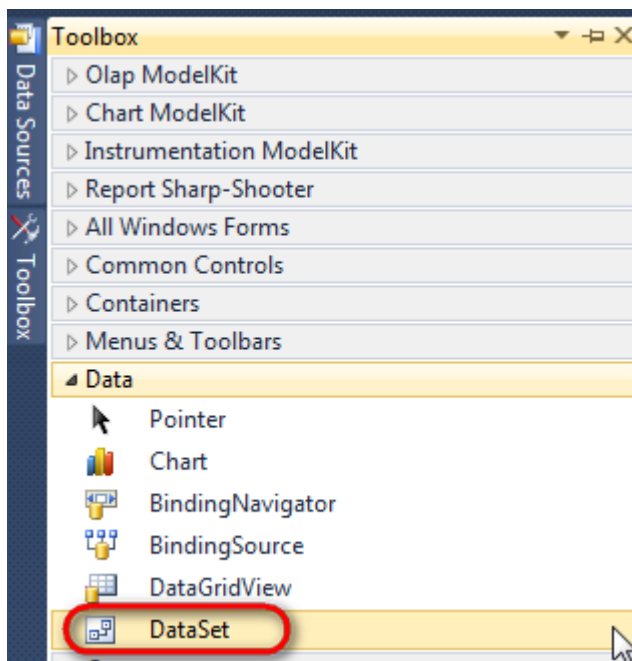


Step 3

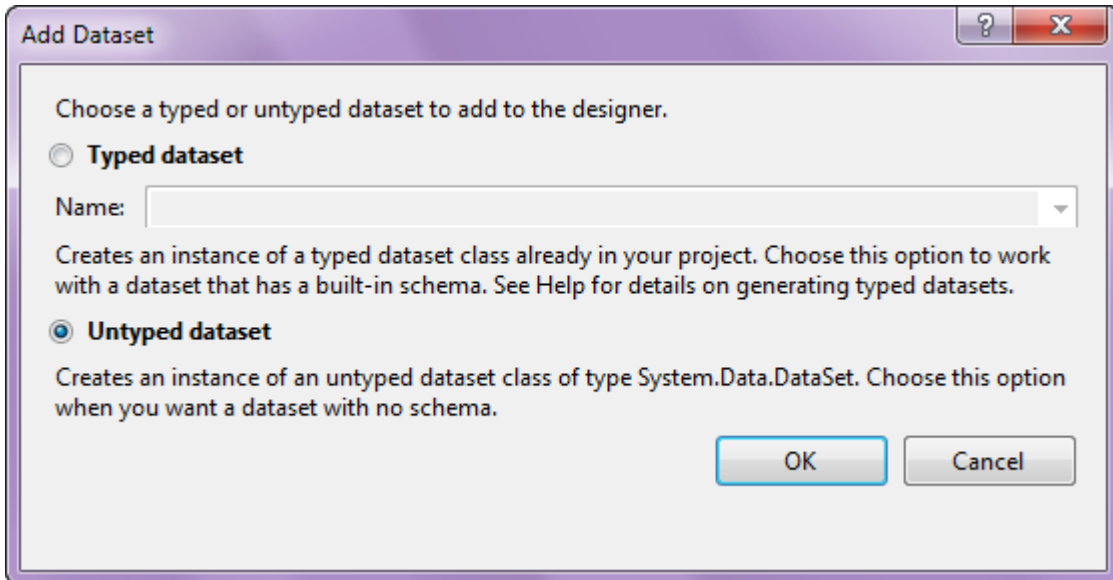
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



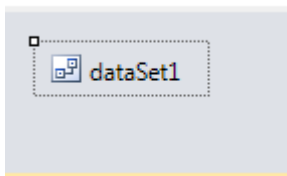
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

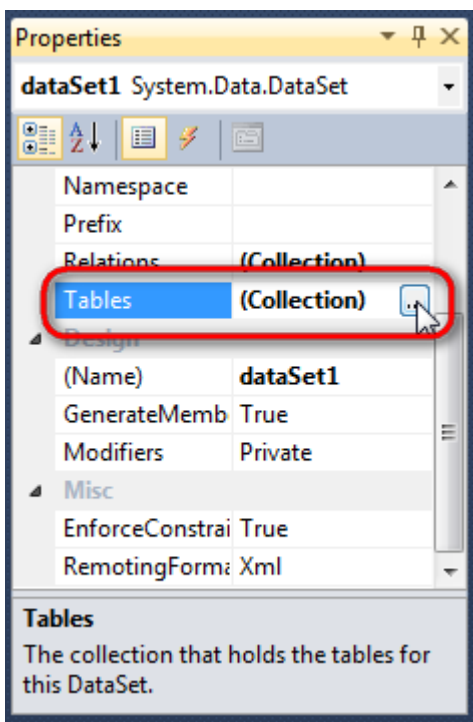


The component is available in the lower part of the window.

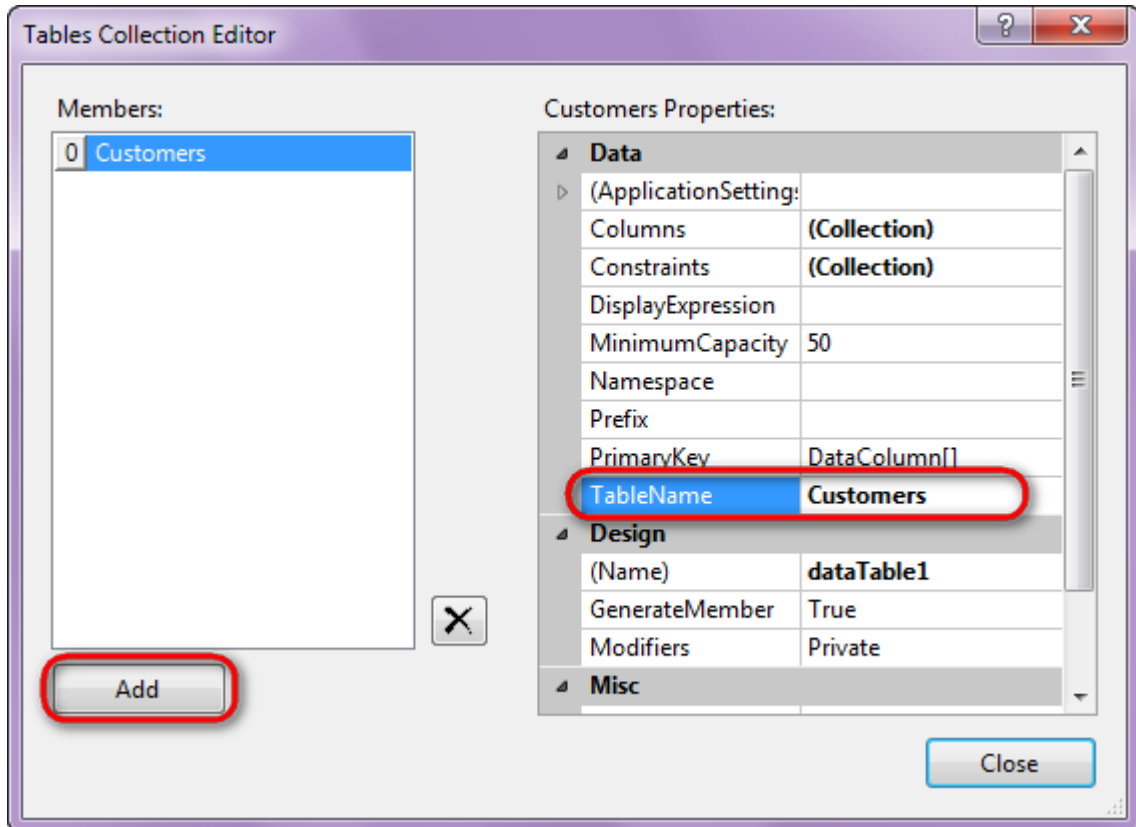


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

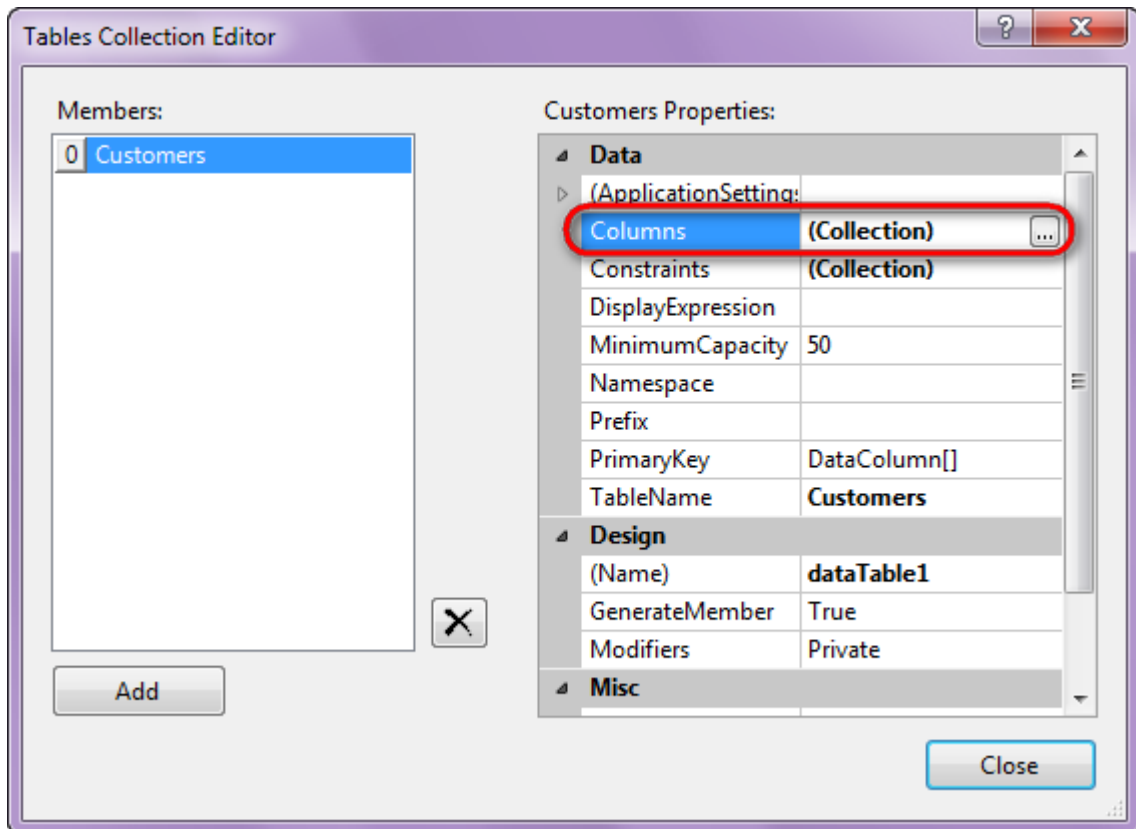


Click "Add" in order to add table. Set property TableName = Customers.

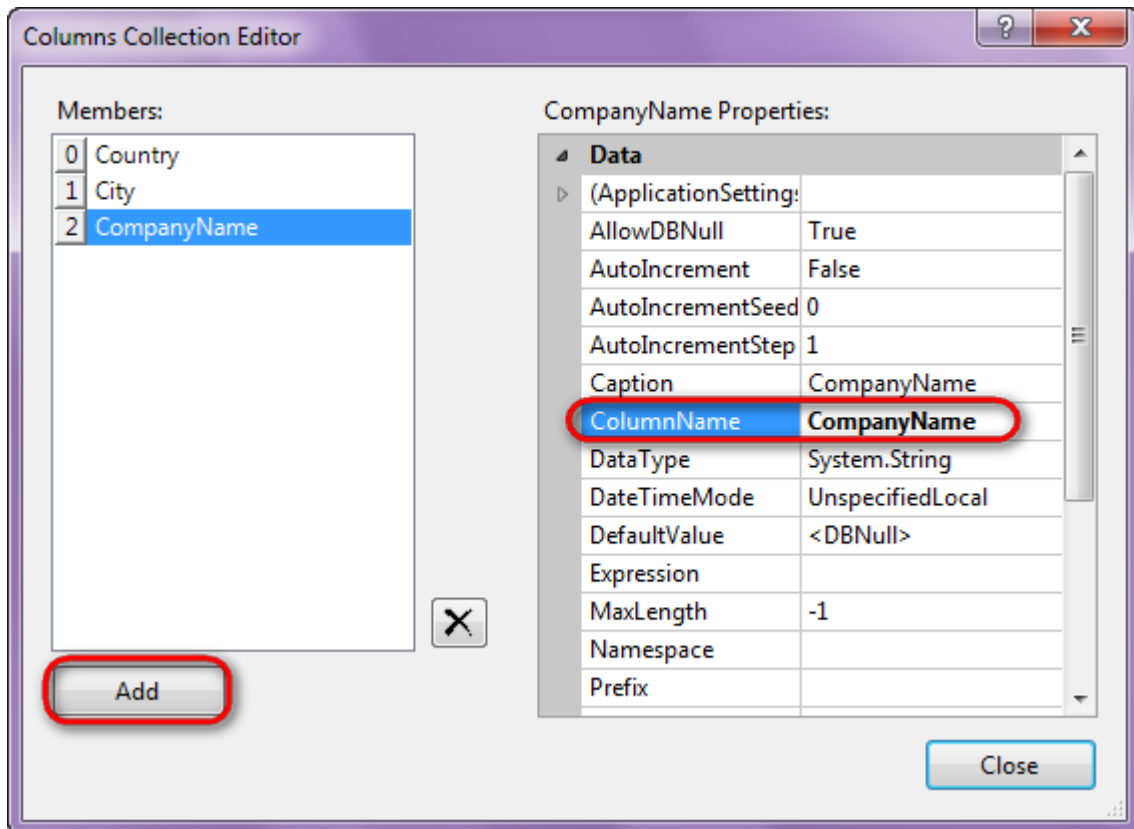


Step 5

Select Columns property, click button  in order to open property editor.

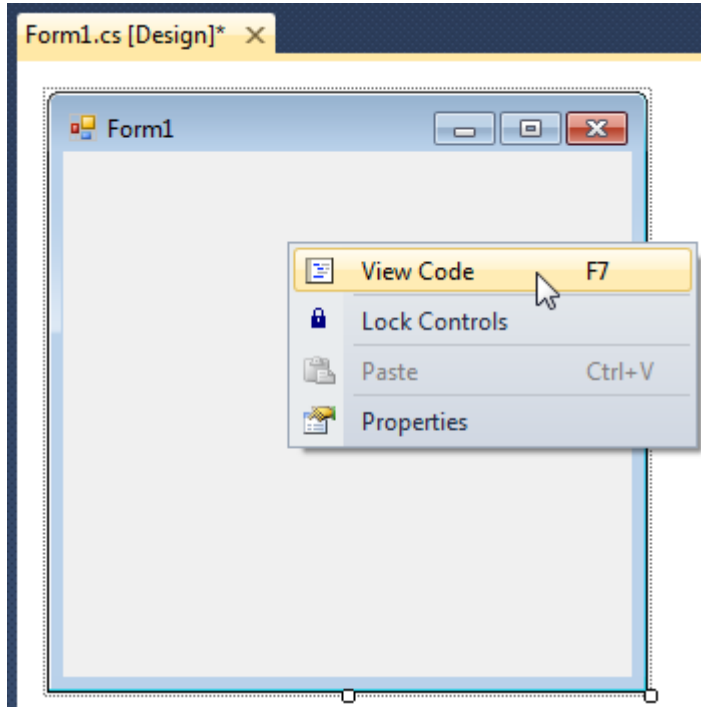


Click "Add" to add a new column. Add three columns. Set ColumnName property to "Country", "City", "CompanyName" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu to view code.



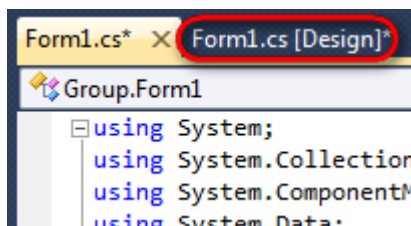
Add the following code to the class constructor in order to fill data source.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
}
```

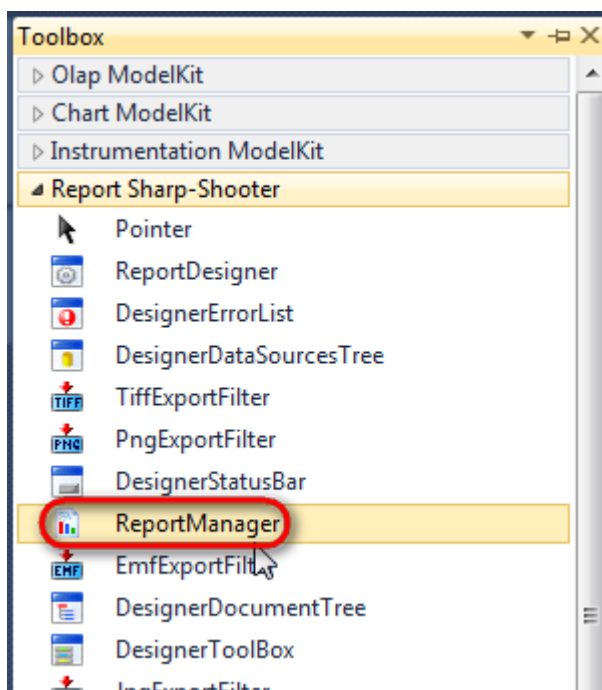
```
row["City"] = "London";  
row["Country"] = "England";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["City"] = "Paris";  
row["Country"] = "France";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["City"] = "Manchester";  
row["Country"] = "England";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Prod House";  
row["City"] = "Manchester";  
row["Country"] = "England";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["City"] = "New York";  
row["Country"] = "USA";  
dataTable1.Rows.Add(row);  
}
```

Step 7

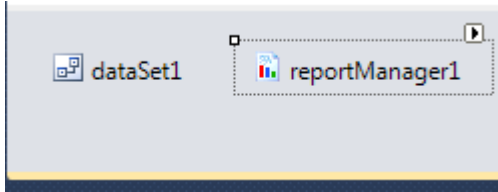
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

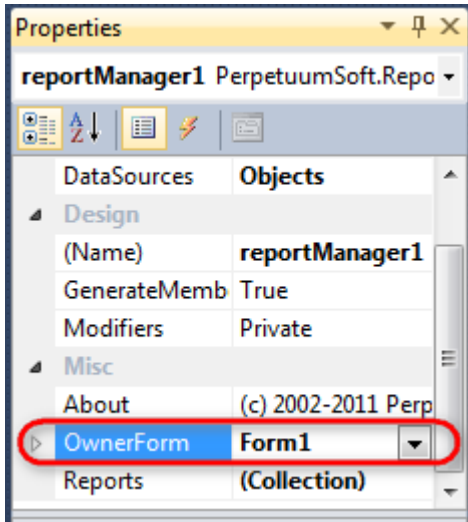


The component is available in the lower part of the window.



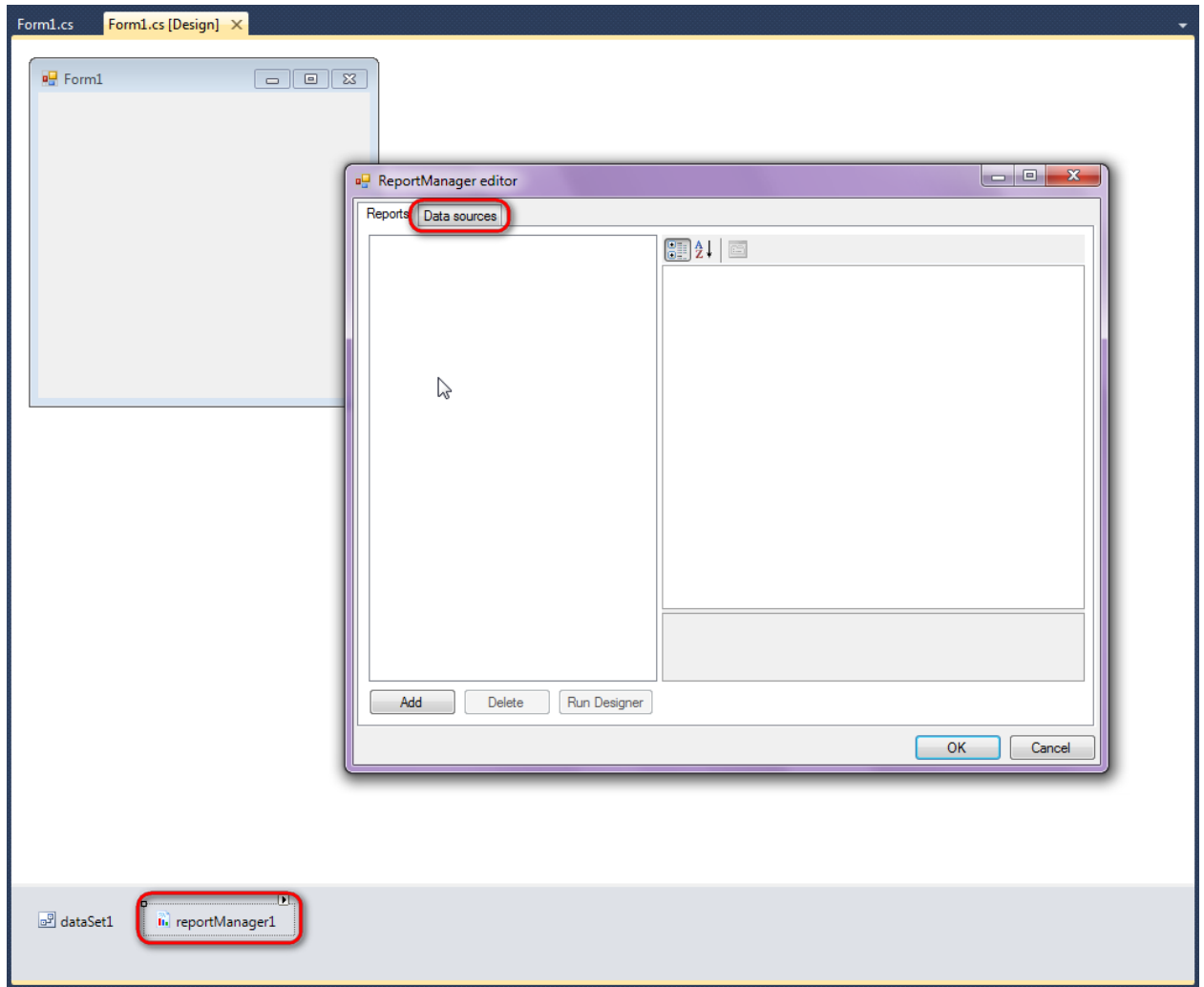
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

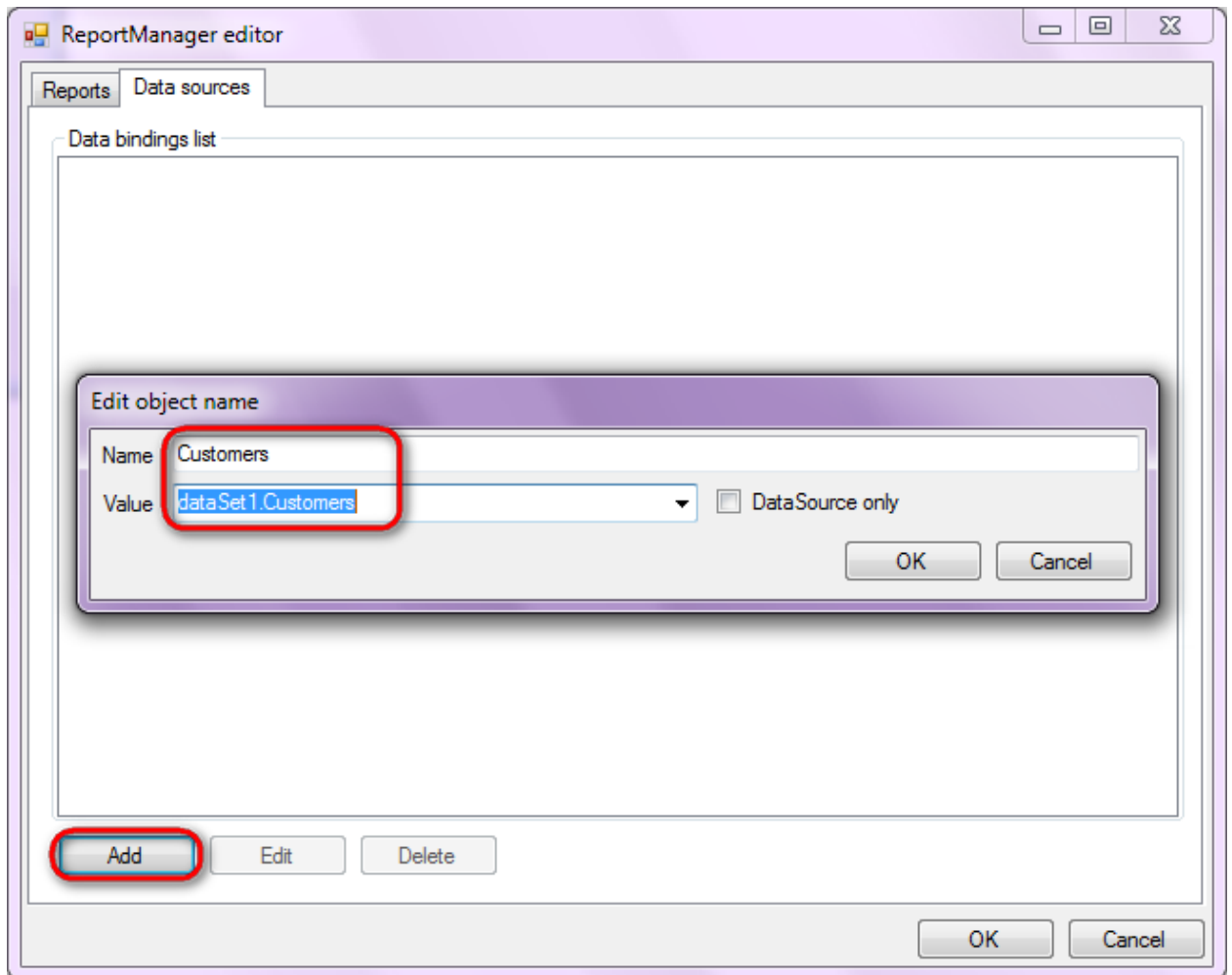


Step 9

Double click on ReportManager to open ReportManager editor.

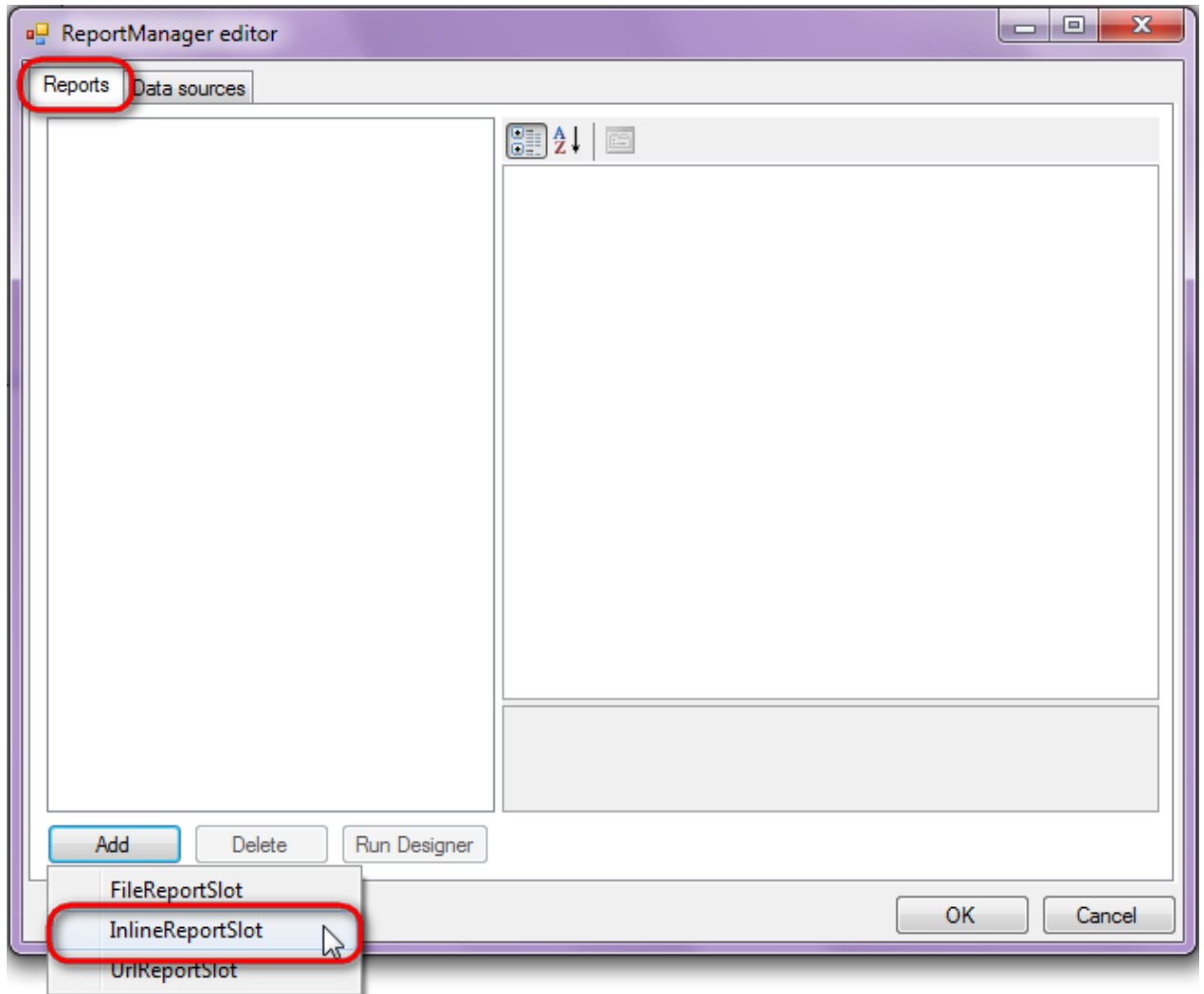


Go to "Data sources" tab, click "Add", set data source name – "Customers", select data source value – "dataSet1.Customers".



Step 10

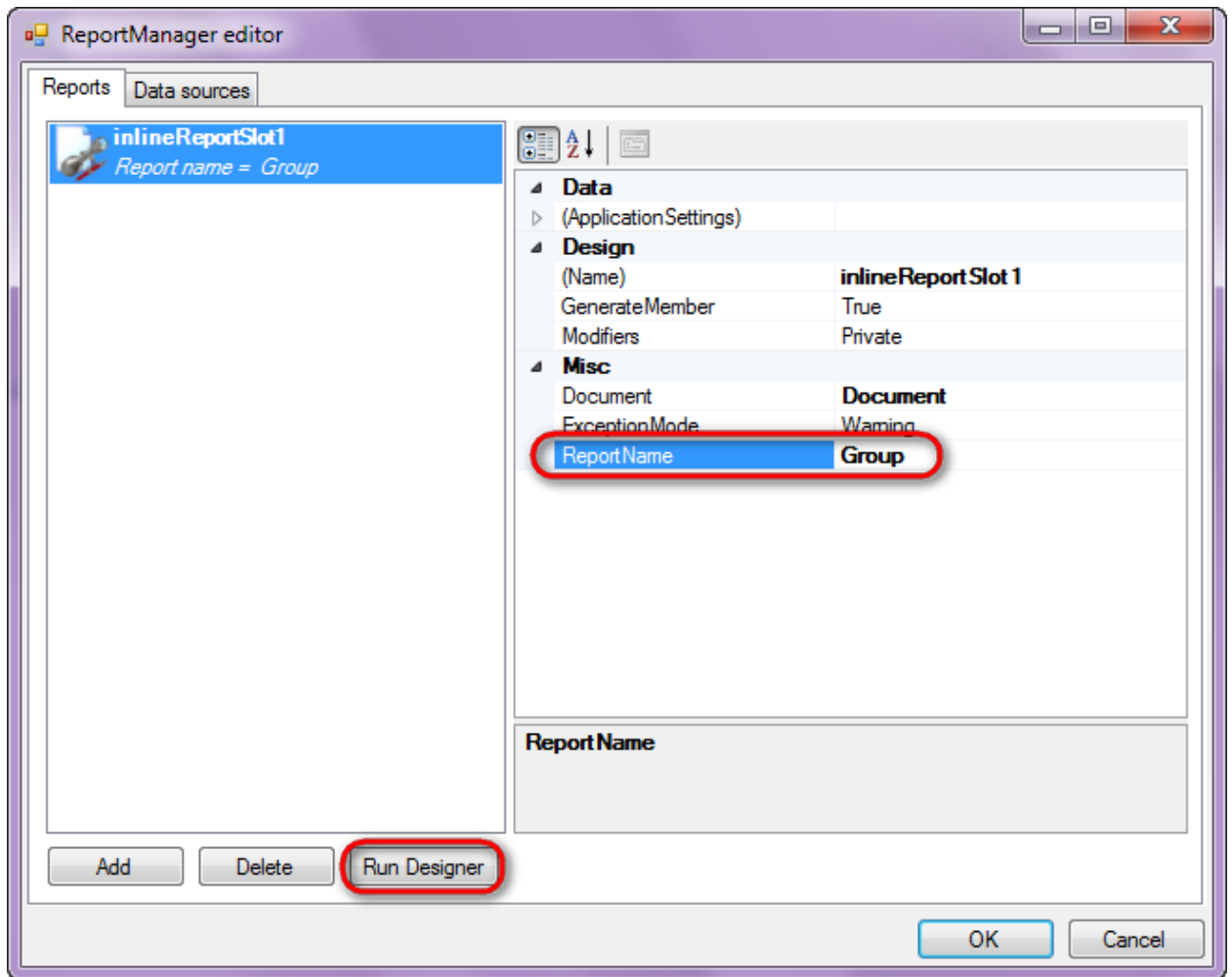
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

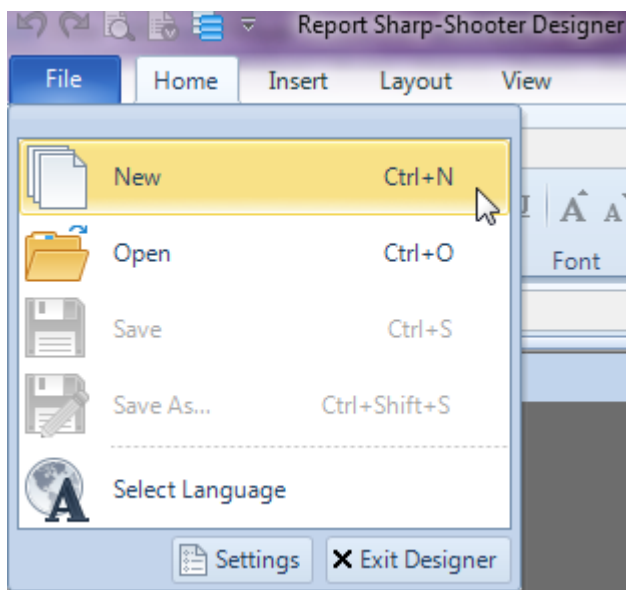
Set name of the report in the property ReportName - "Group".

Click "Run Designer" in order to open template editor - Report Designer.

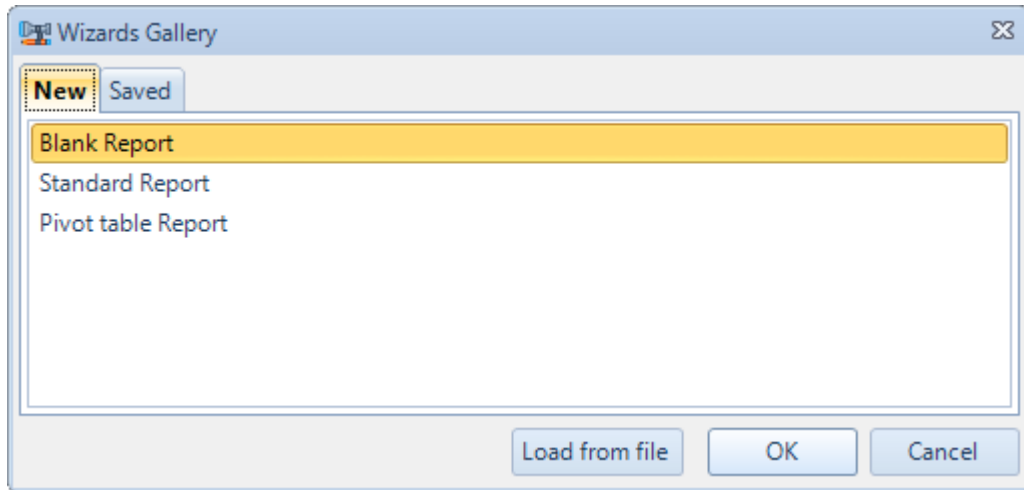


Step 12

Create new empty template – select File\New from the main menu.

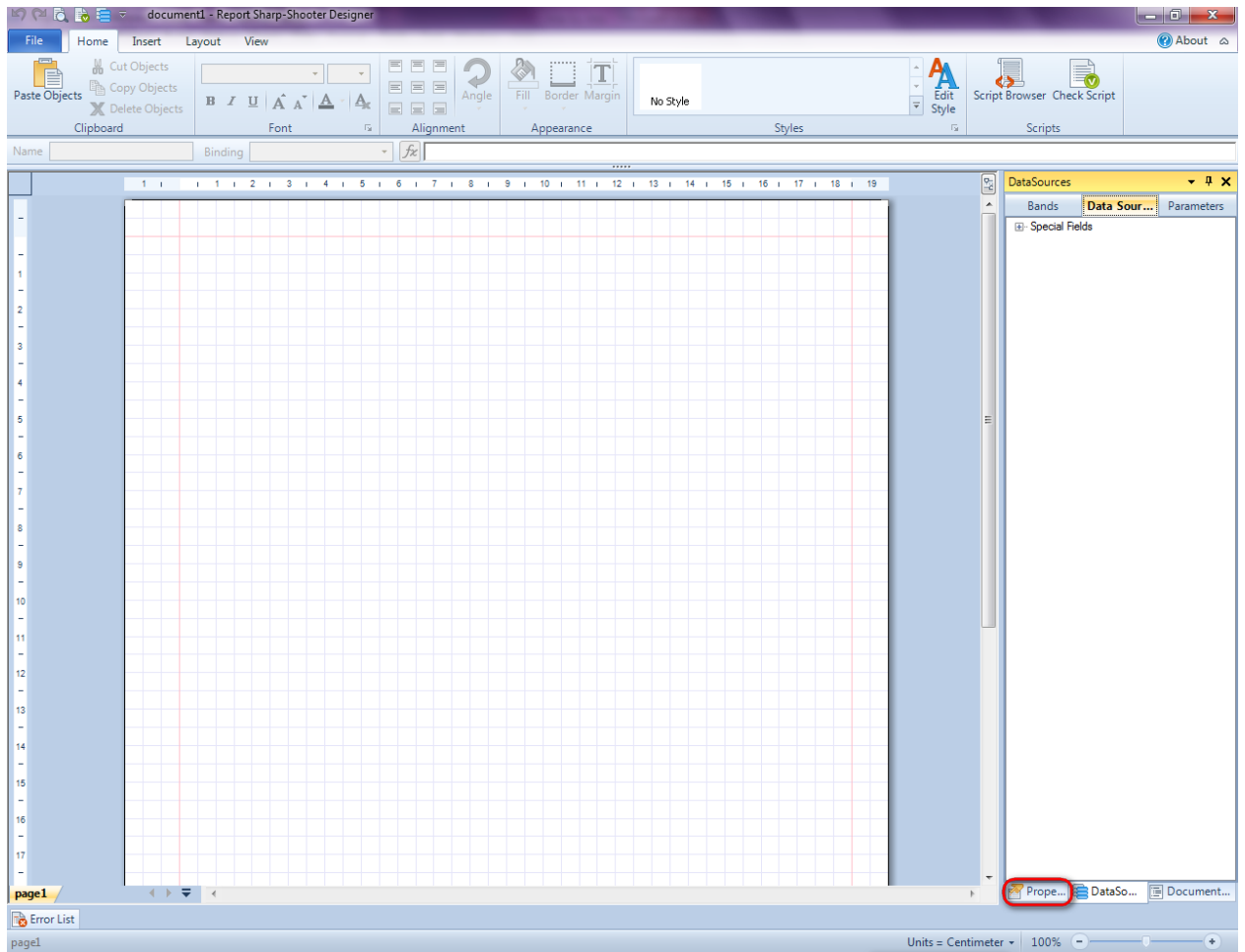


Select "Blank Report" in the Wizards Gallery and click "OK".

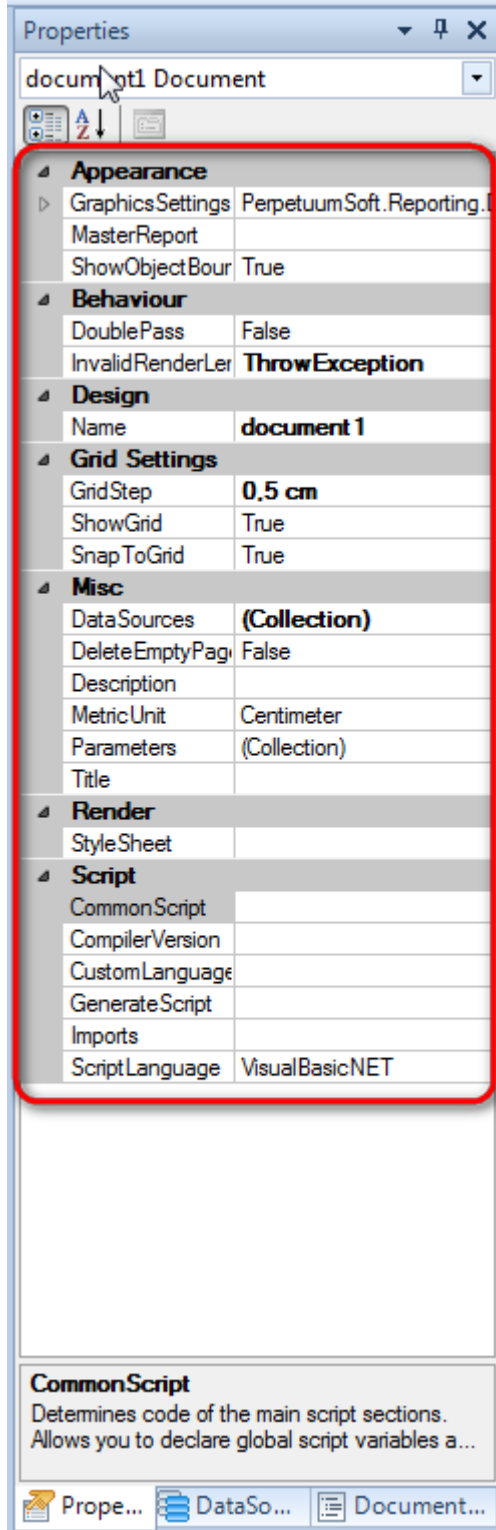


Step 13

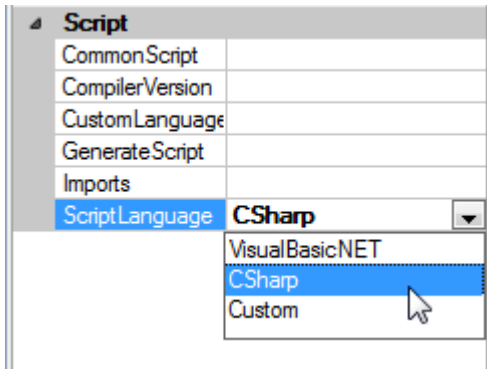
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

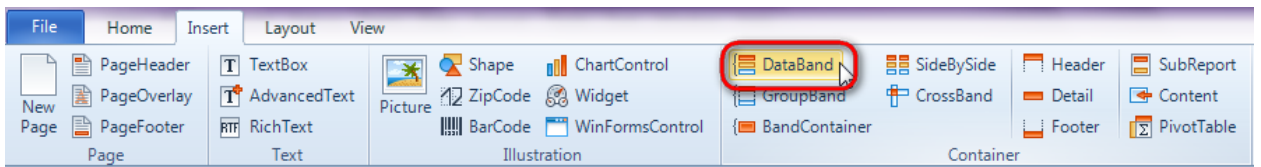


Set property ScriptLanguage = CSharp.



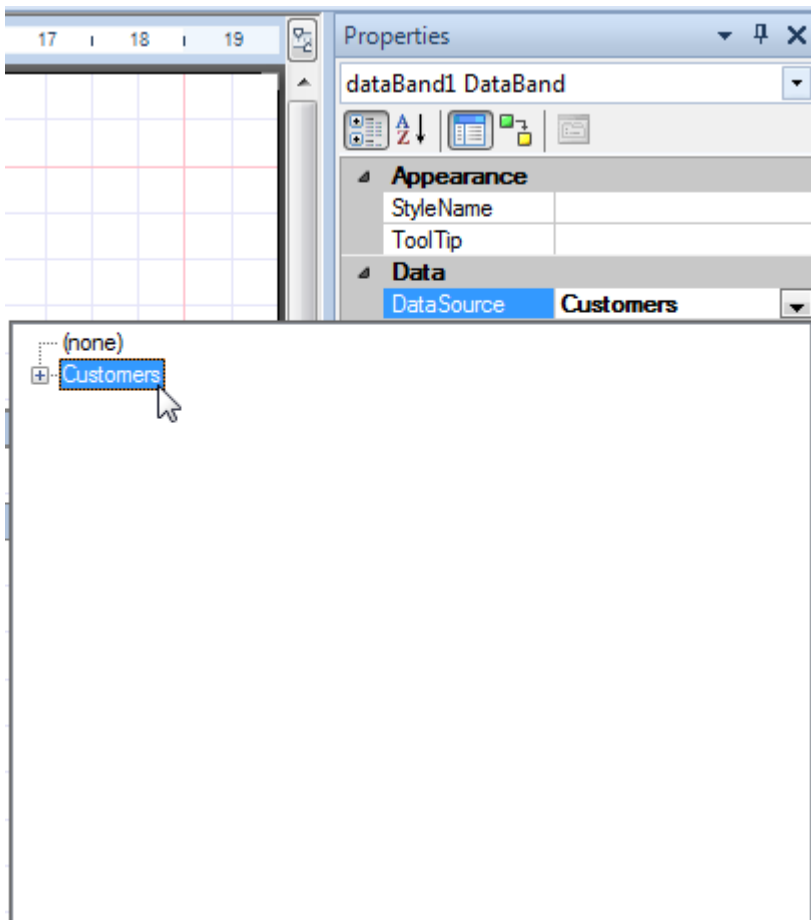
Step 14

Press "DataBand" button on the Insert tab in the group Container.



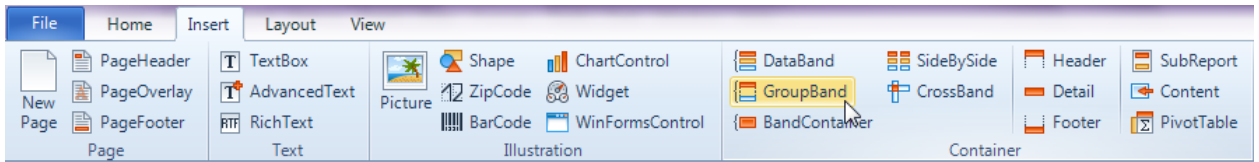
Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = Customers.



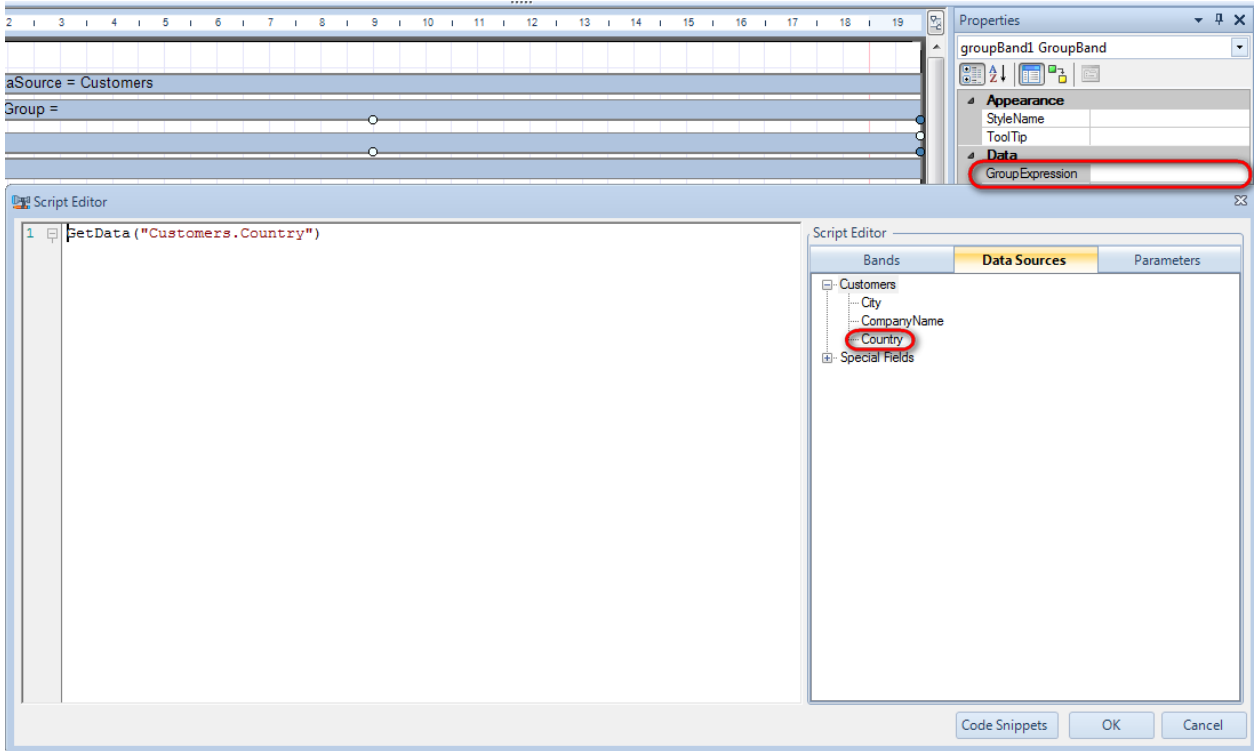
Step 15

Press "GroupBand" button on the Insert tab in the group Container.

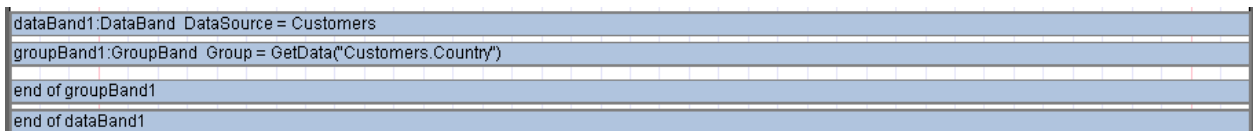


Click on the DataBand area to add GroupBand inside DataBand.

To group list by country set GroupExpression property on the "Properties" tab to "GetData("Customers.Country")".

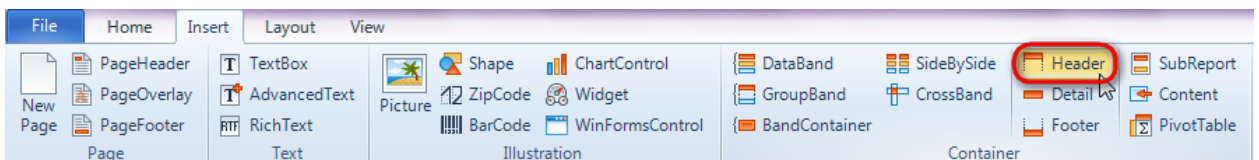


Report template on this stage should look as follows:



Step 16

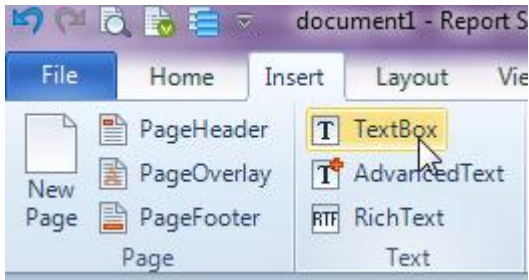
To display header of every group create Header band - press "Header" button on the Insert tab in the group Container.



Click on the GroupBand area to add Header band inside GroupBand.

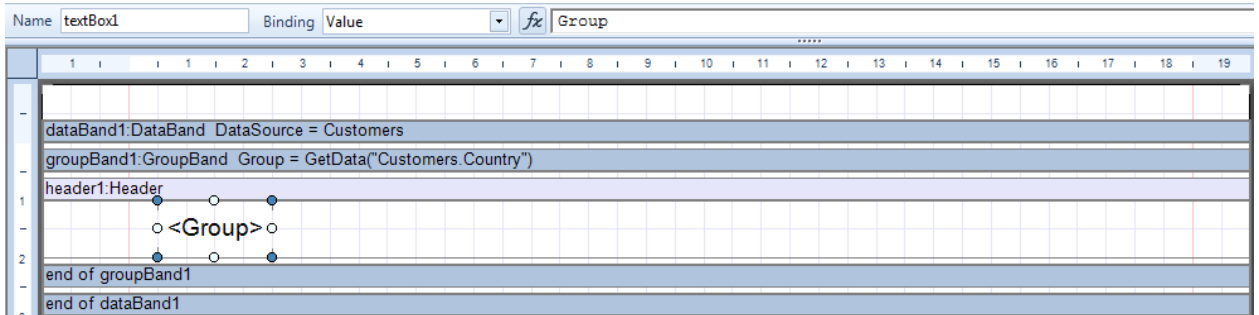
Step 17

Press button "TextBox" on the Insert tab in the group Text.



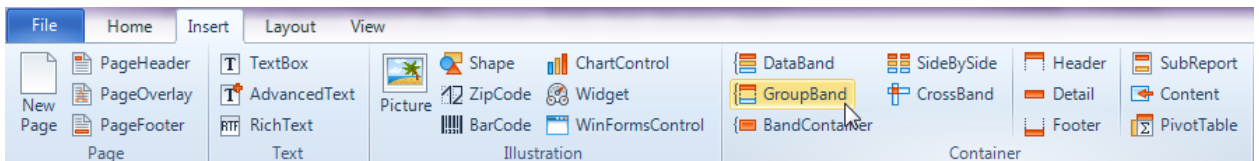
Click on the Header area to add TextBox element inside the Header.

Set Value property to "Group".



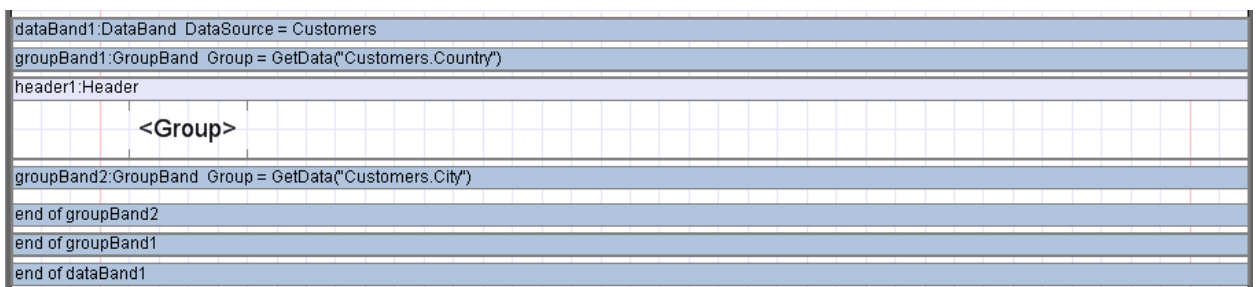
Step 18

Press "GroupBand" button on the Insert tab in the group Container.



Click on the GroupBand area to add the second GroupBand inside it.

To group list by city set GroupExpression property on the "Properties" tab to "GetData("Customers.City)".

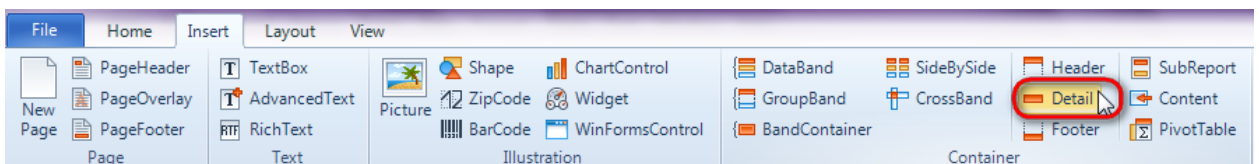


Step 19

To display group header add Header band to the groupBand2, add TextBox, set property Value = Group.

Step 20

Press "Detail" button on the Insert tab in the group Container.



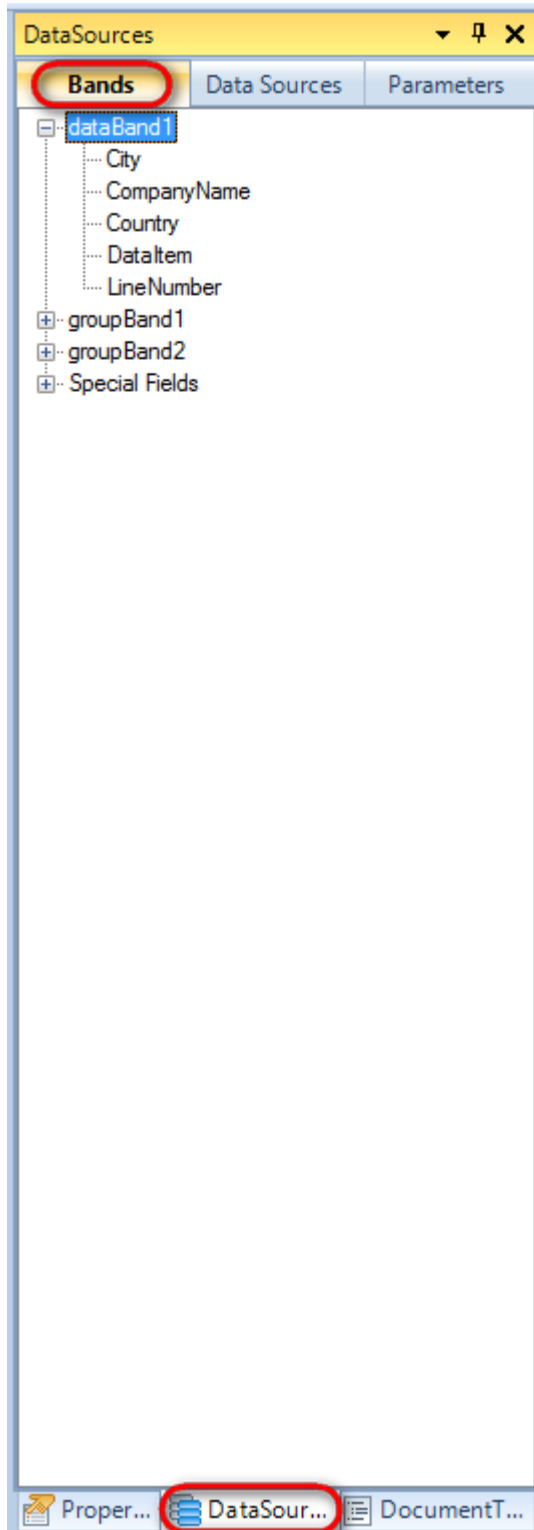


Click on the groupBand2 area to add Detail band inside groupBand2.

dataBand1:DataBand DataSource = Customers	
groupBand1:GroupBand Group = GetData("Customers.Country")	
header1:Header	
<Group>	
groupBand2:GroupBand Group = GetData("Customers.City")	
header2:Header	
<Group>	
detail1:Detail	
end of groupBand2	
end of groupBand1	
end of dataBand1	

Step 21

Go to "DataSources" tab.



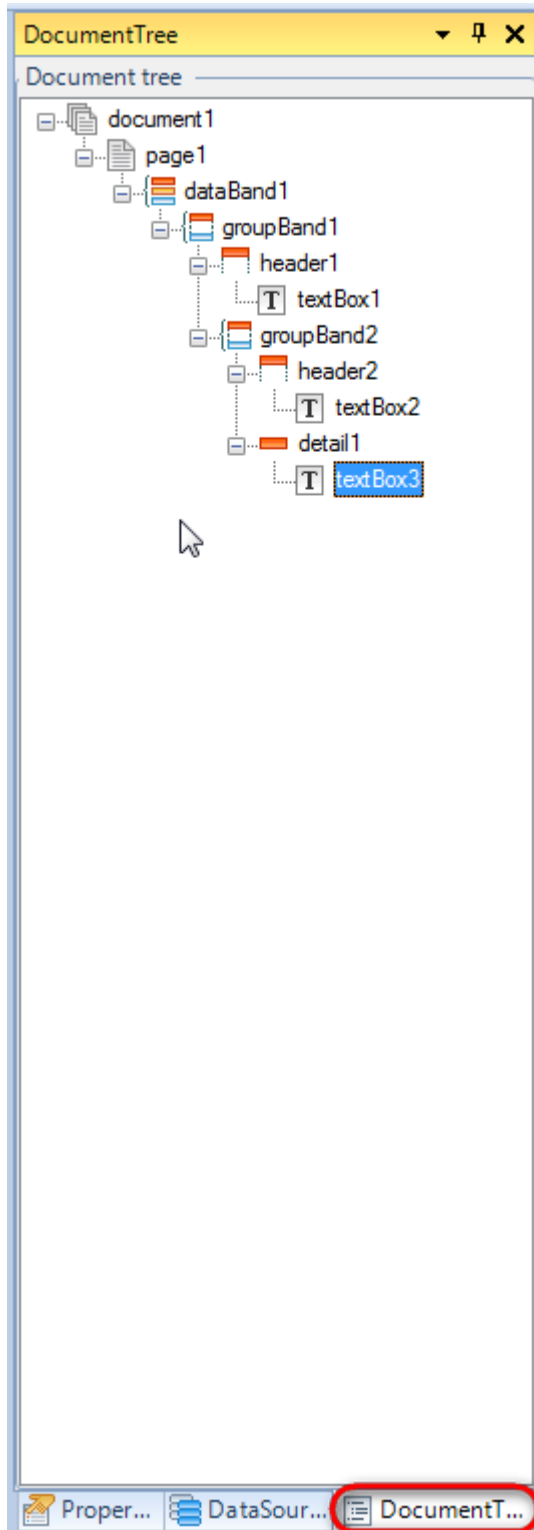
Drag and drop "CompanyName" field from the dataBand1 tree to the detail1 band. As a result TextBox is created. Value property is automatically filled with script loading data from the data source.

Change size of the elements and locate them in the way shown in the picture below.



```
dataBand1:DataBand DataSource = Customers
groupBand1:GroupBand Group = GetData("Customers.Country")
header1:Header
    <Group>
groupBand2:GroupBand Group = GetData("Customers.City")
header2:Header
    <Group>
detail1:Detail
    <dataBand1["CompanyName"]>
end of groupBand2
end of groupBand1
end of dataBand1
```

In order to view template structure, go to "DocumentTree" tab.



Step 22

Save template, close Report Designer.

Step 23

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

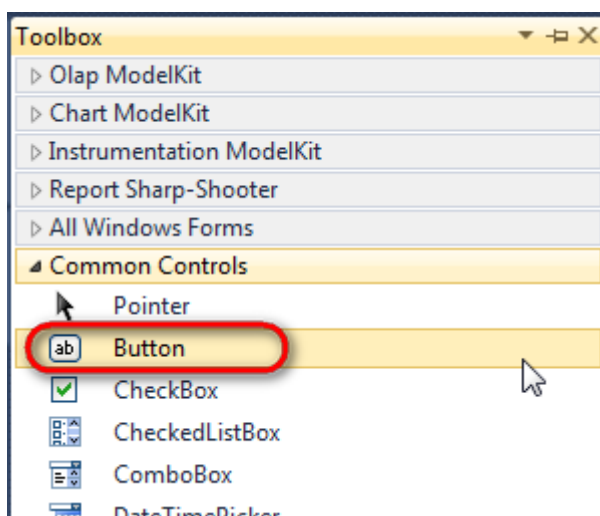
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
```



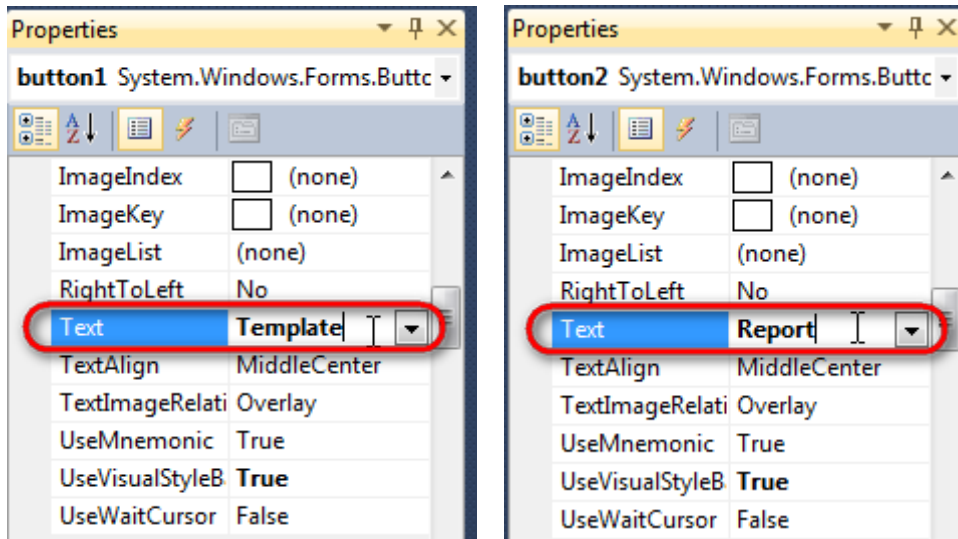
```
row["CompanyName"] = "Alfreds Futterkiste";
row["City"] = "London";
row["Country"] = "England";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Ana Trujillo Emparedados y helados";
row["City"] = "Paris";
row["Country"] = "France";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Ernst Handel";
row["City"] = "Manchester";
row["Country"] = "England";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Prod House";
row["City"] = "Manchester";
row["Country"] = "England";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Toms Spezialitäten";
row["City"] = "New York";
row["Country"] = "USA";
dataTable1.Rows.Add(row);
inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 24

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



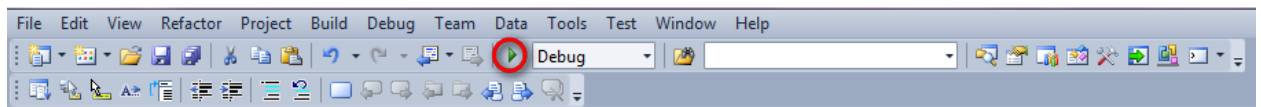
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

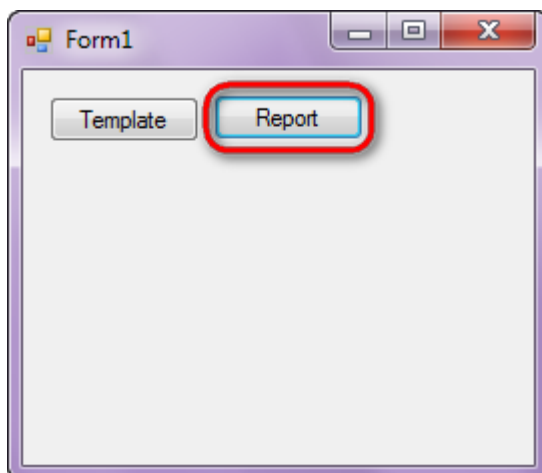
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 25

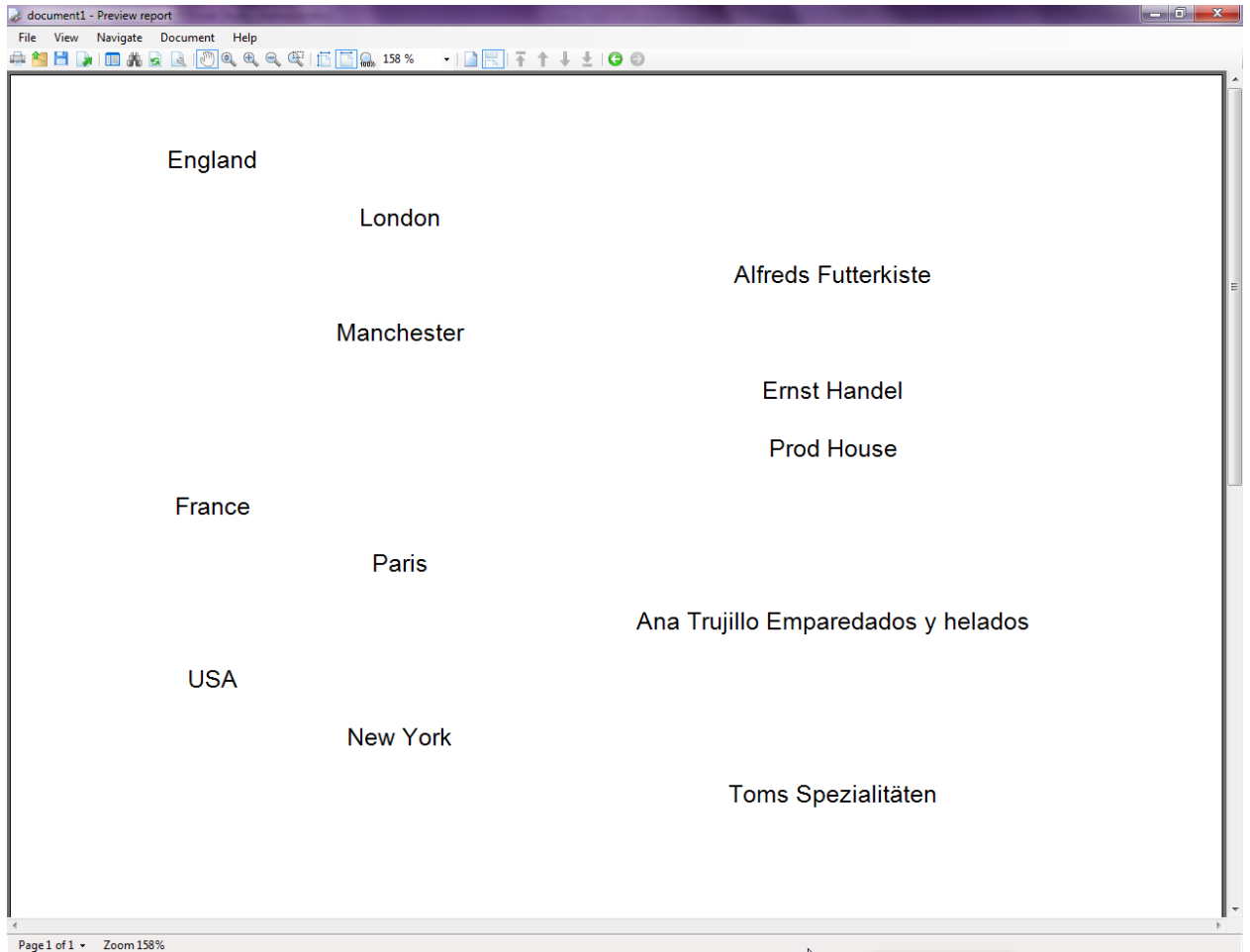
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



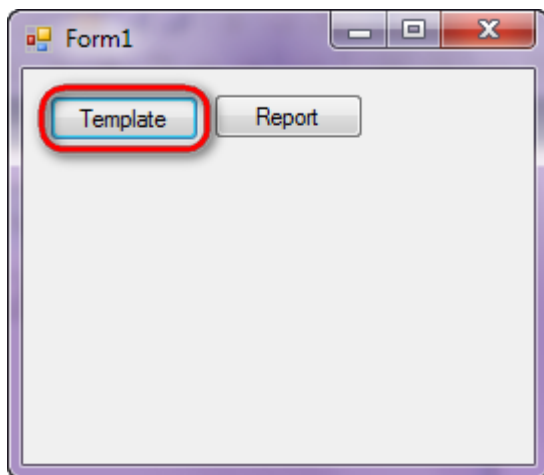
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Reports\Grouping\Nested Groups.

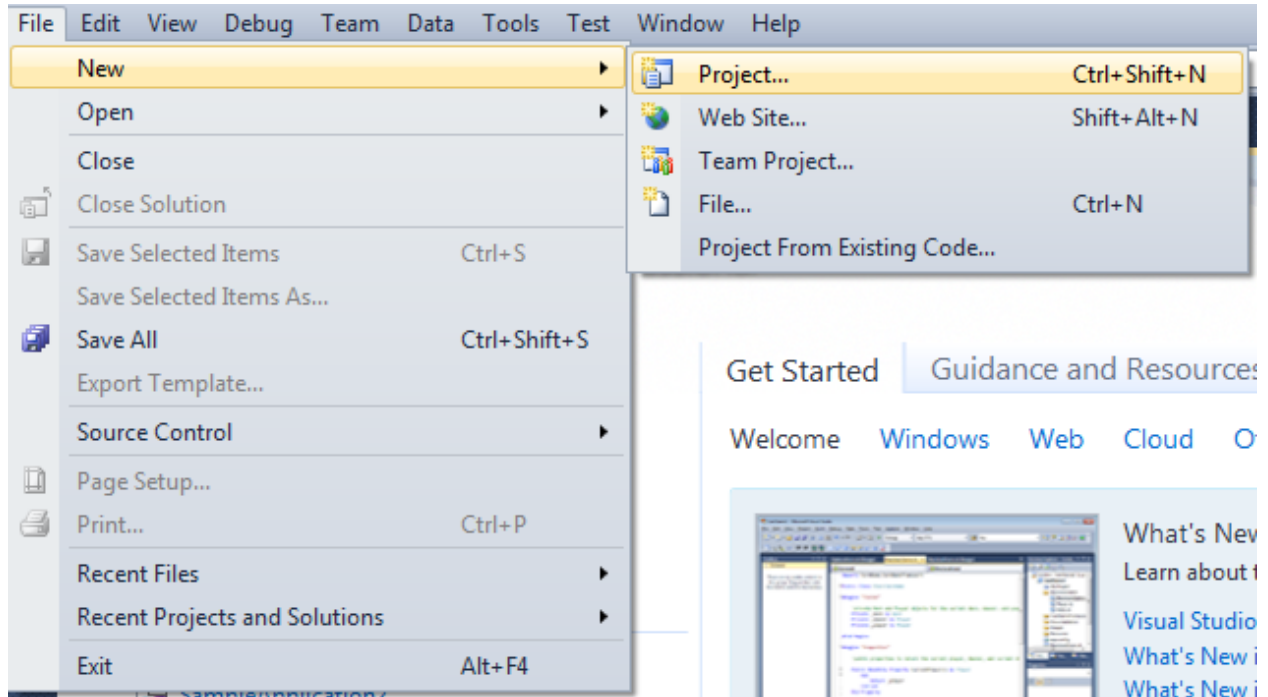


Joint Use of Grouping, Sorting, Filtering and Totals

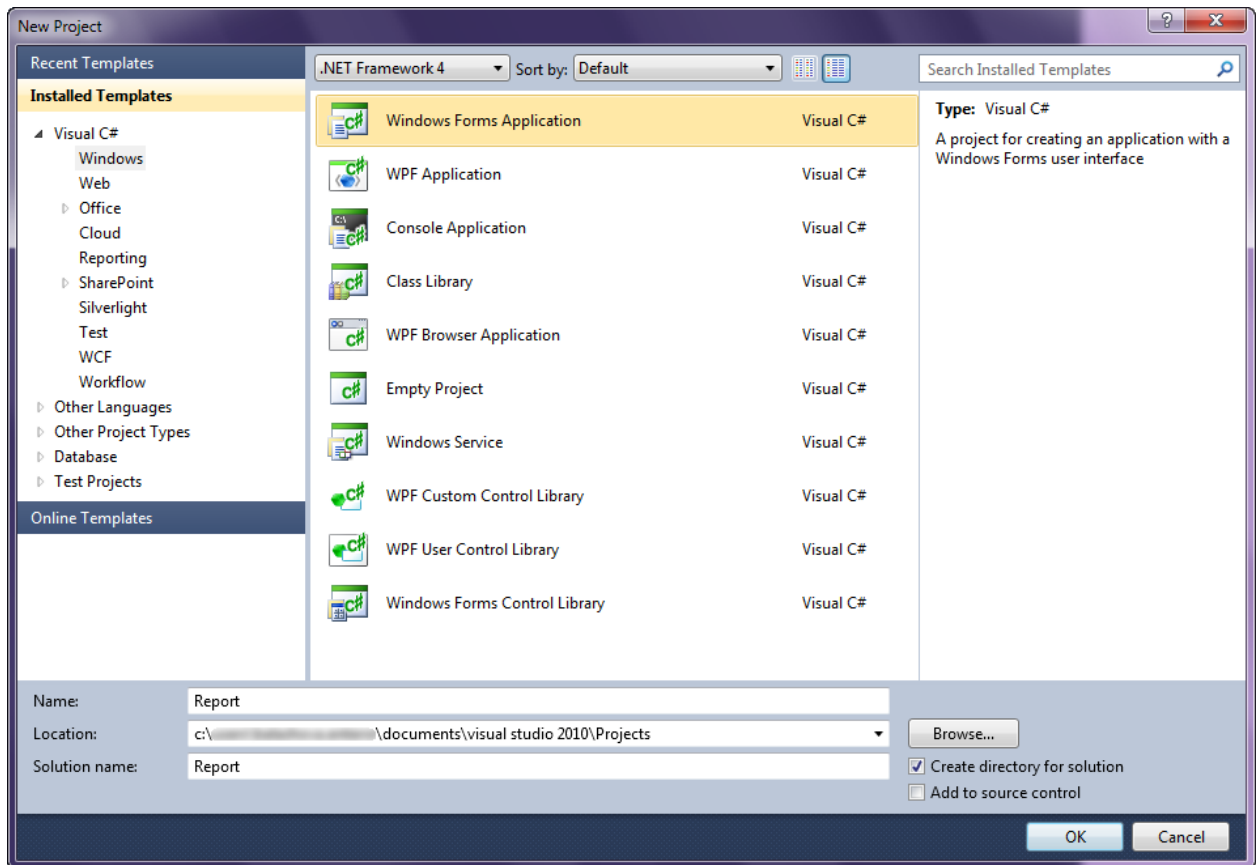
Template of a report containing list of customers grouped by city; for every customer – a list of orders sorted by date. The report will contain orders made after January 1, 2010. Aggregation function will calculate total orders price for every employee.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

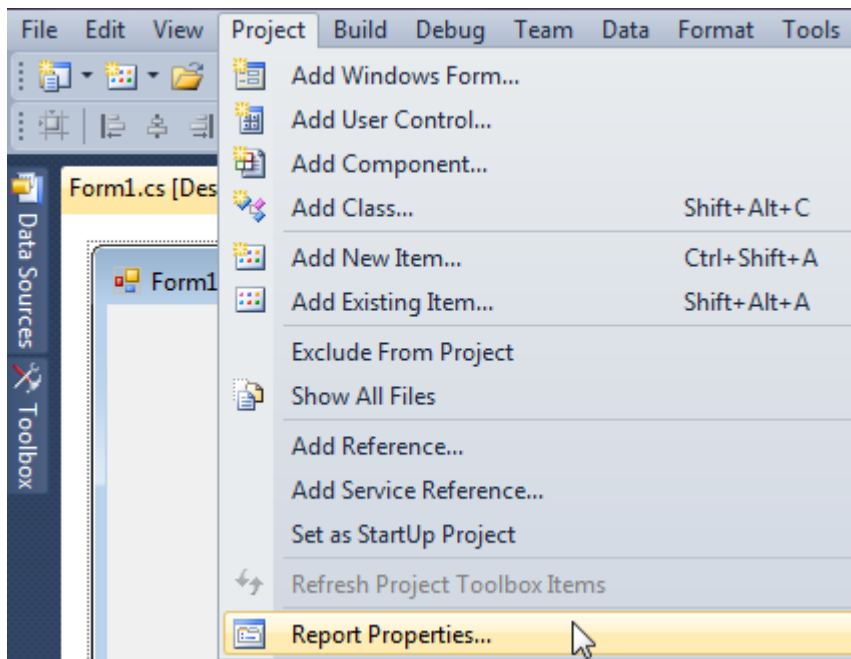


Select Windows Forms Application, set project name – “Report”, set directory to save the project to.

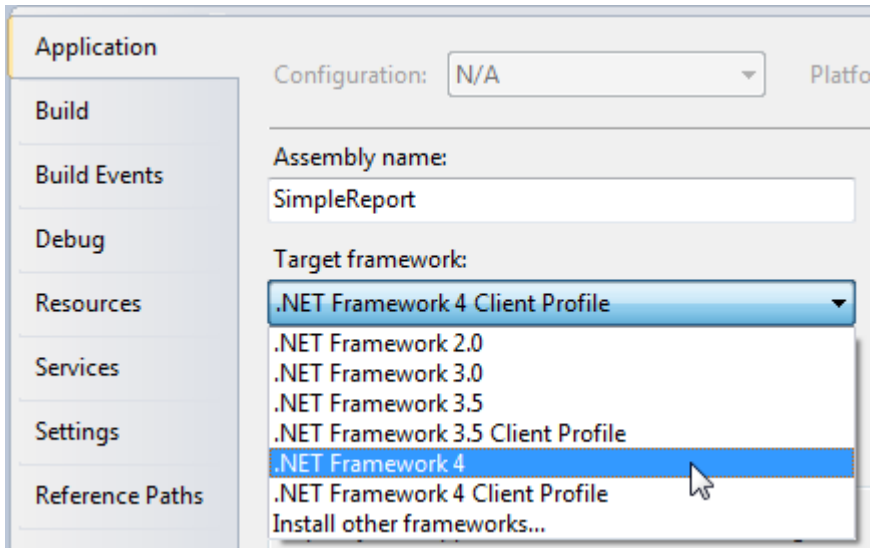


Step 2

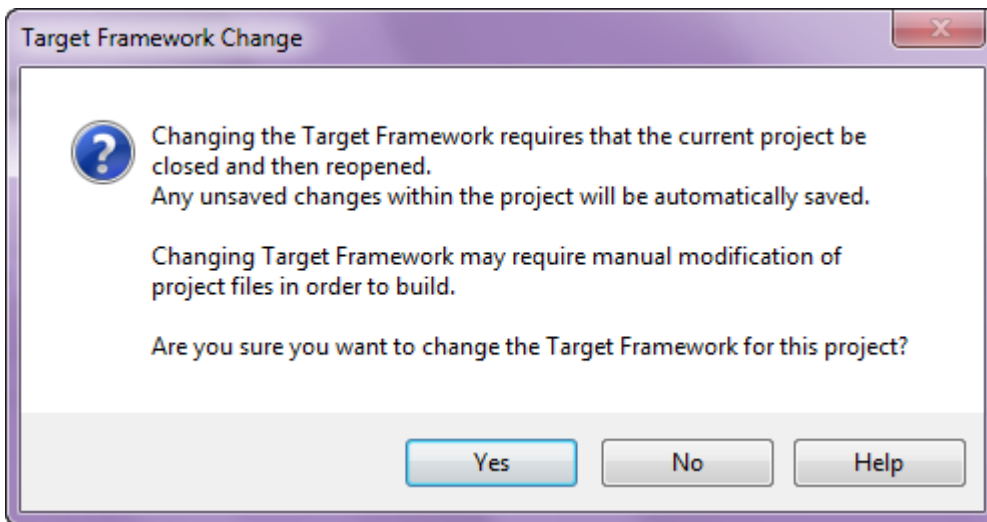
Change the project properties. Select the Project\Report Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

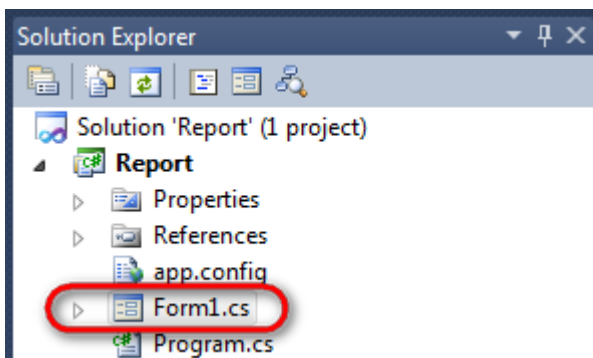


In the opened window press the “Yes” button.

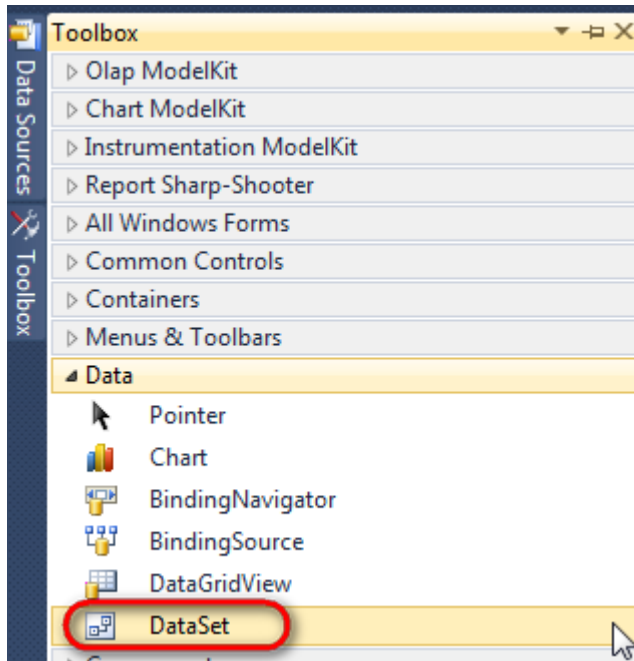


Step 3

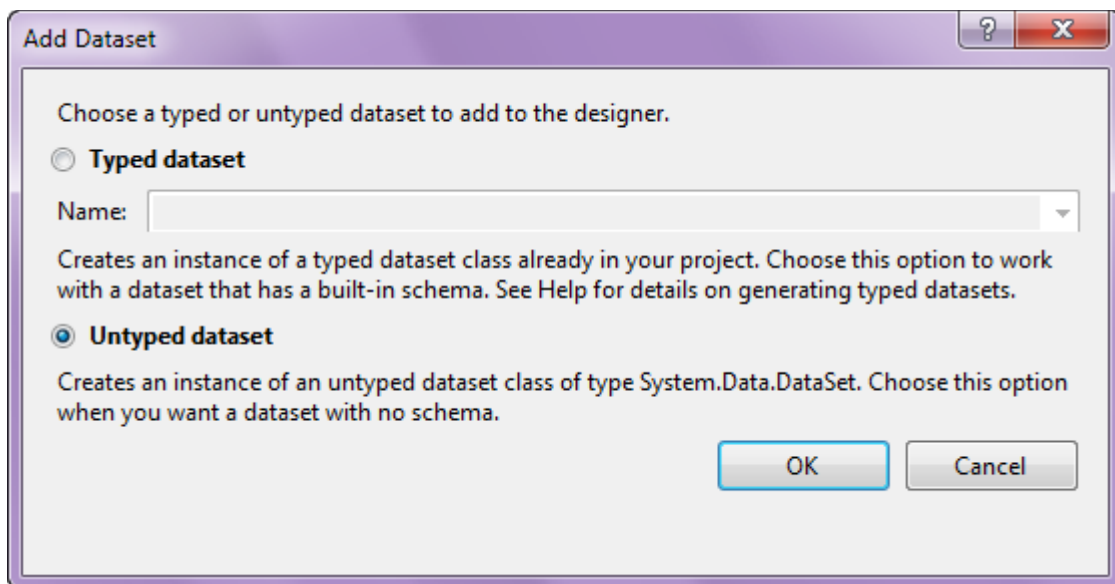
Open main form of the application by double click on the “Form1.cs” in the Solution Explorer.



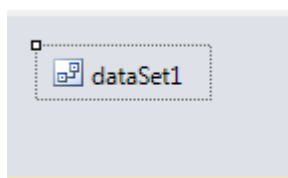
Click “DataSet” element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

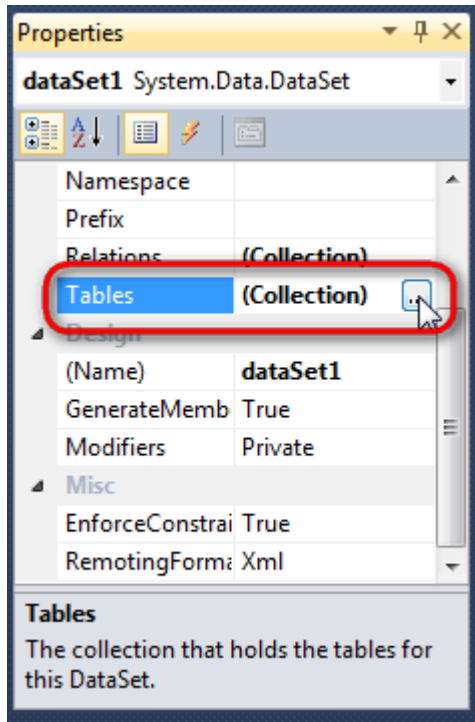


The component is available in the lower part of the window.

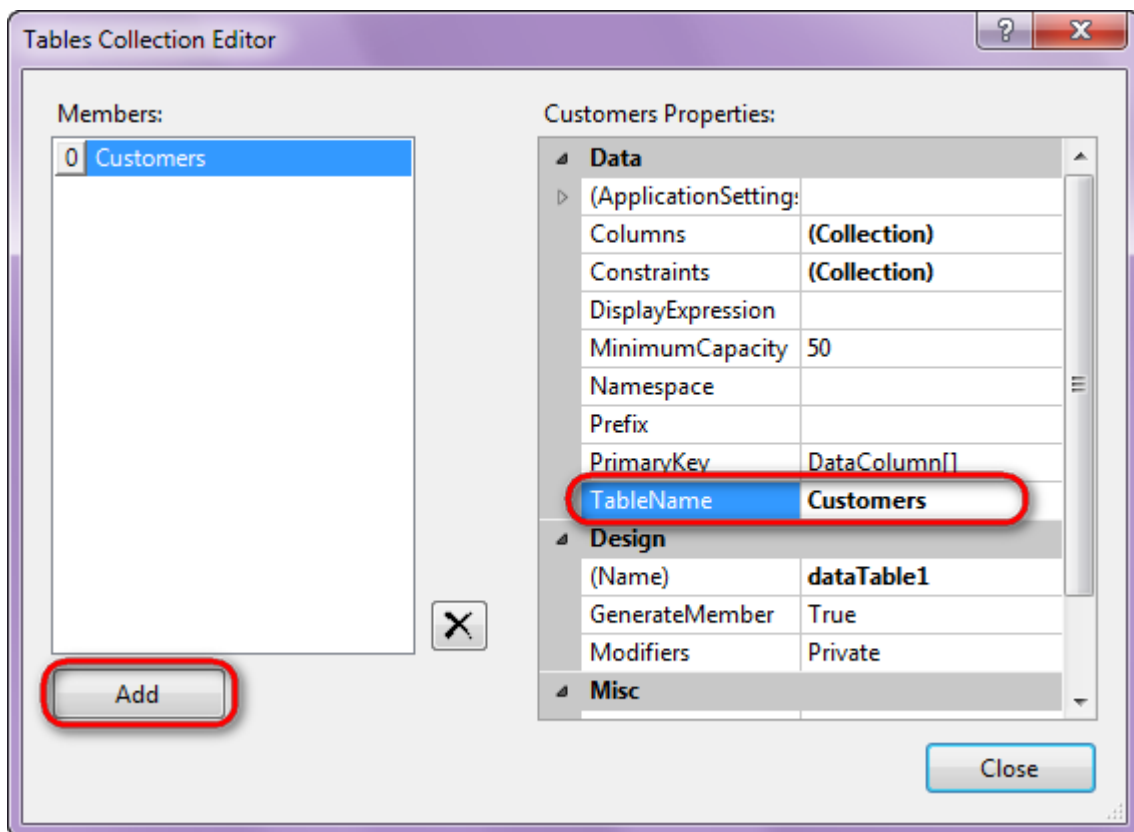


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

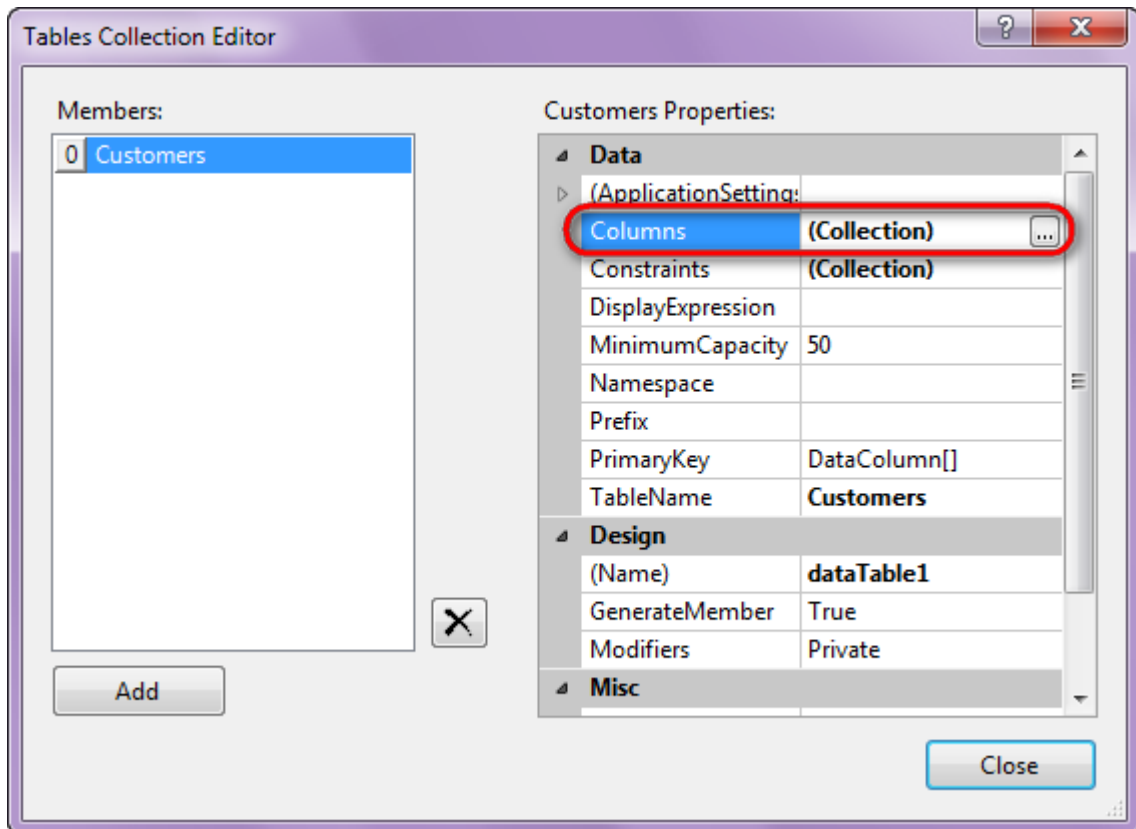


Click "Add" in order to add table. Set property TableName = Customers.

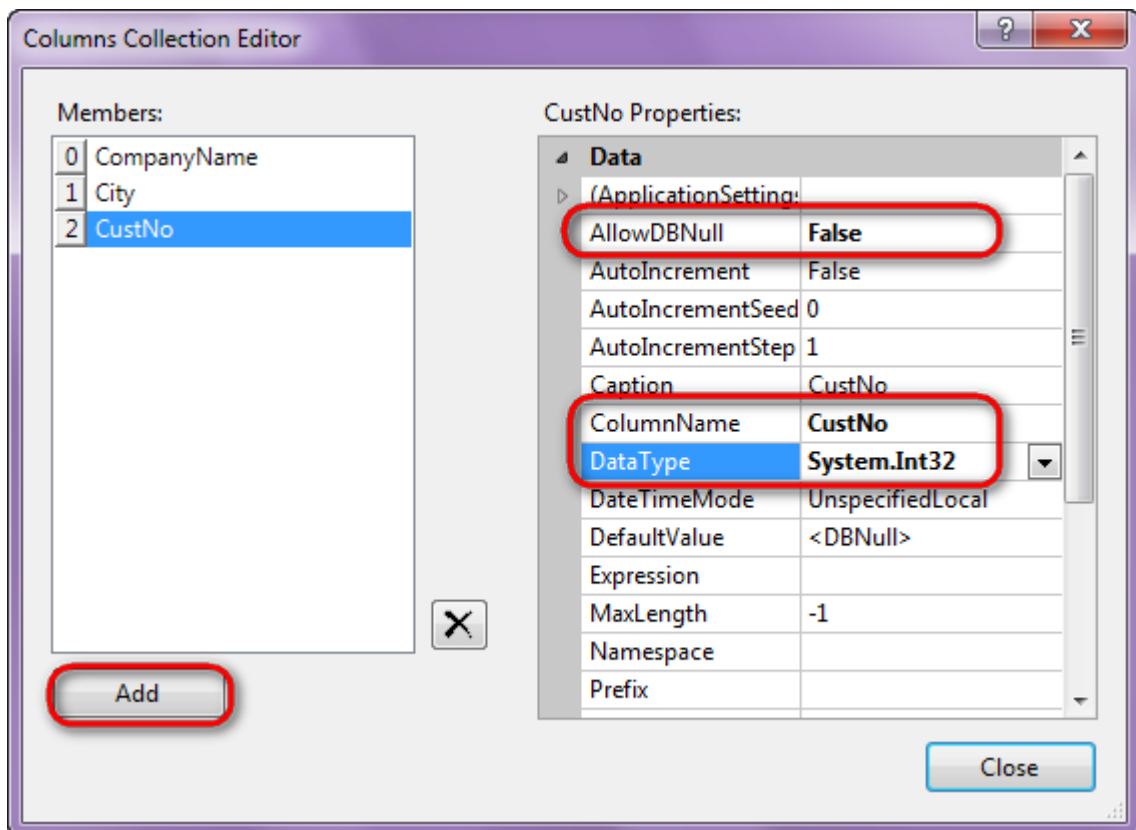


Step 5

Select Columns property, click button  in order to open property editor.



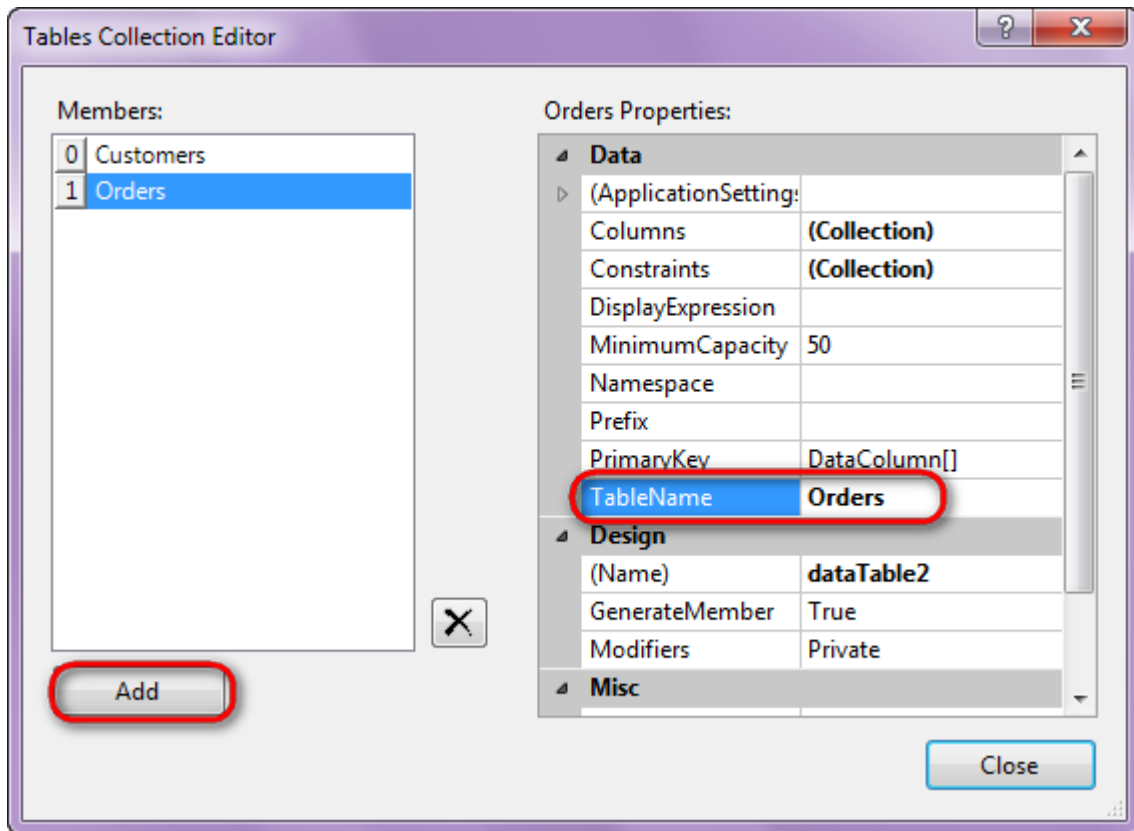
Click "Add" to add a new column. Add three columns. Set ColumnName property to "CompanyName", "City", "CustNo" correspondingly. For the "CustNo" column, set the following properties DataType = System.Int32, AllowDBNull = False.



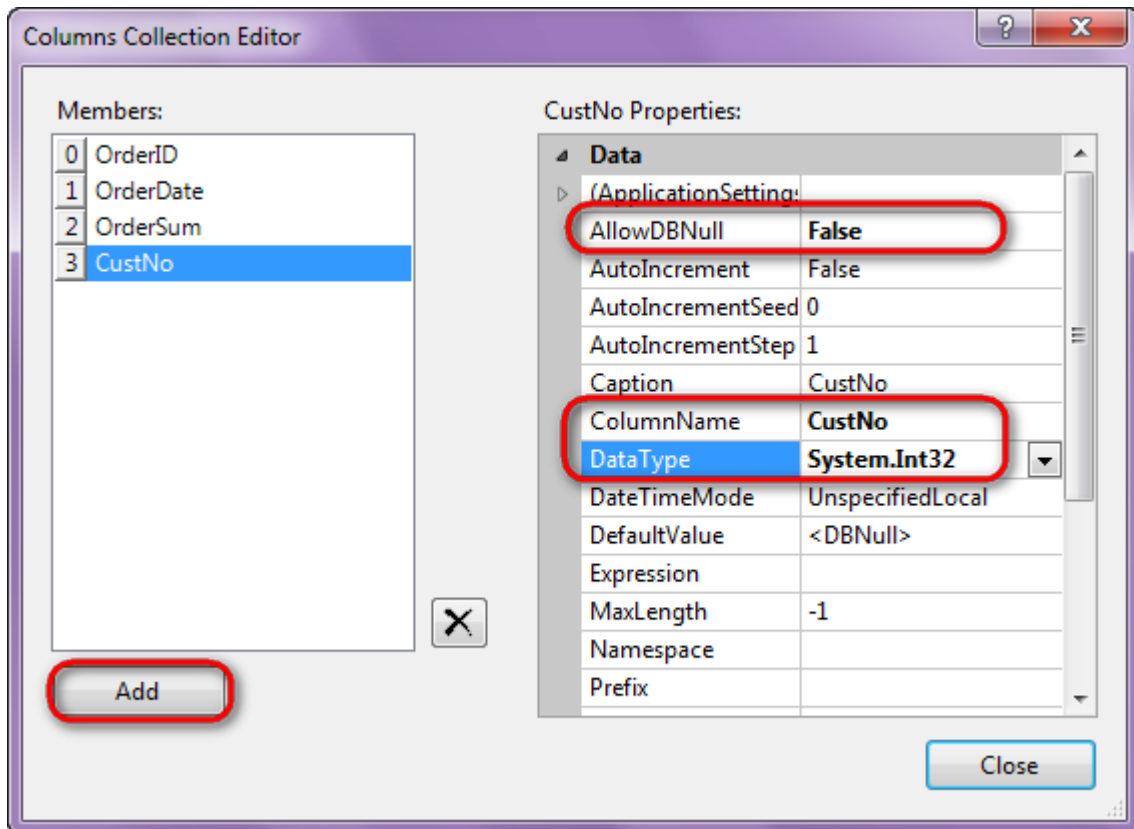


Step 6


In the Tables Collection Editor, click "Add" in order to add one more table. Set TableName to "Orders".

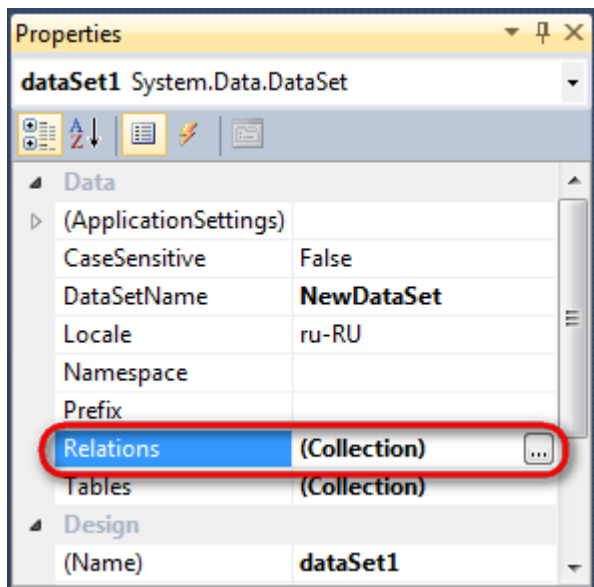


Open Columns property editor. Add four columns, set ColumnName property to "OrderID", "OrderDate", "and OrderSum ", " CustNo". For the "CustNo" column, set the following properties DataType = System.Int32, AllowDBNull = False. For the "OrderDate" column, set property DataType = System.DateTime.

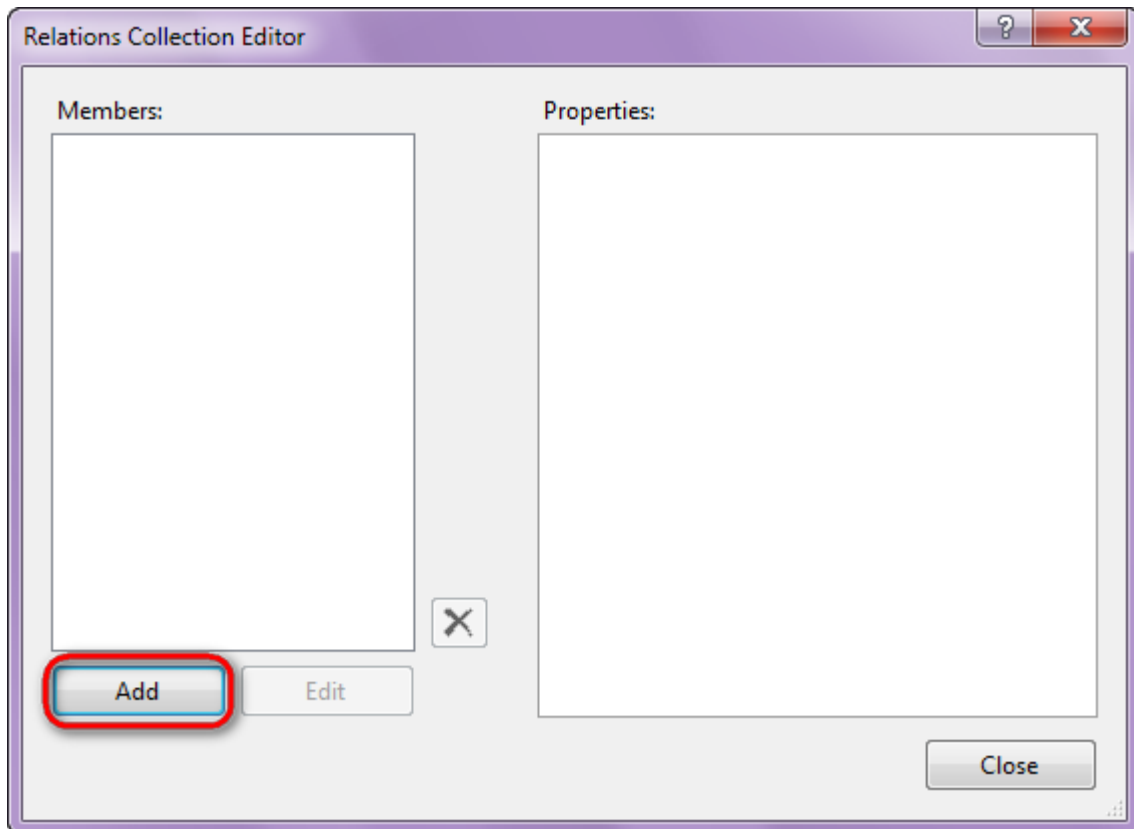


Step 7

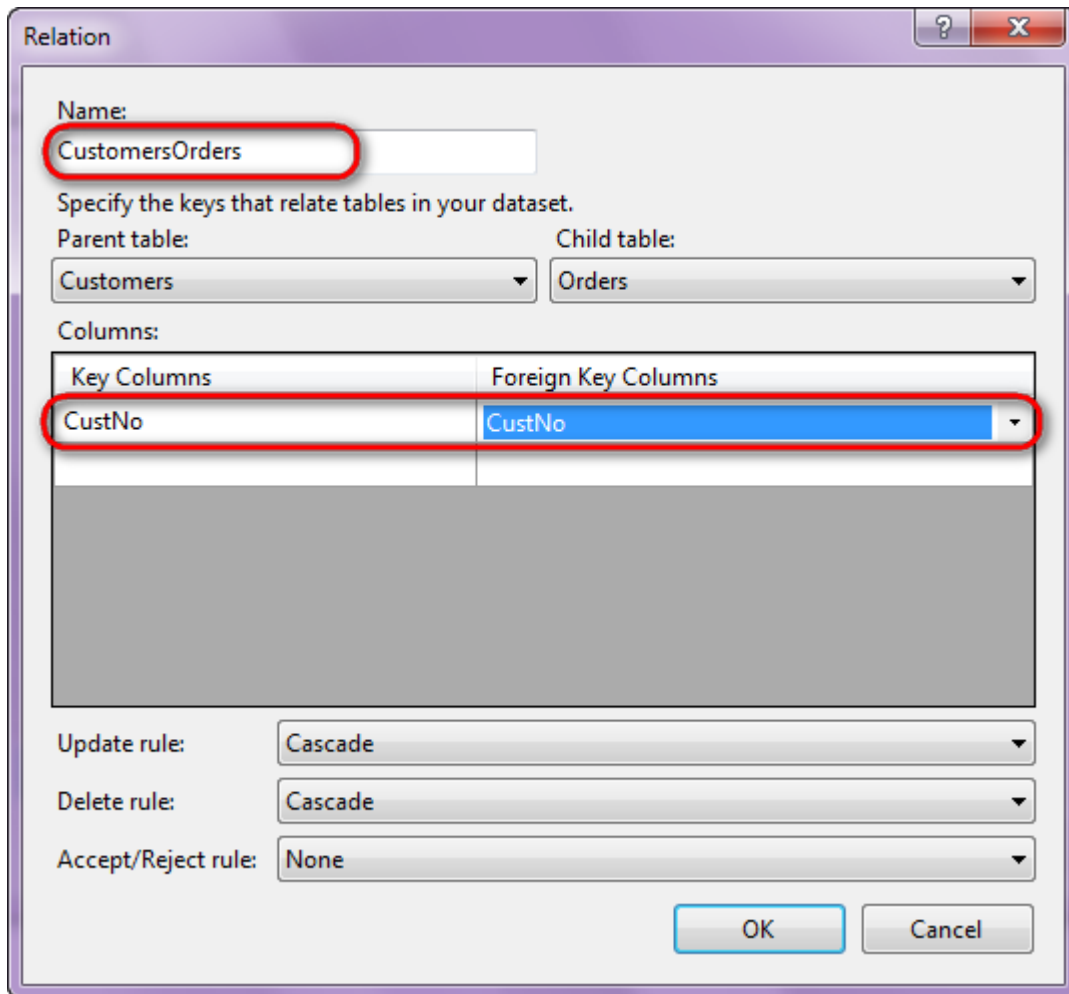
Select the Relations property in the property grid. Click the  button in order to open the property editor.



Press the "Add" button in order to add the binding between the tables.

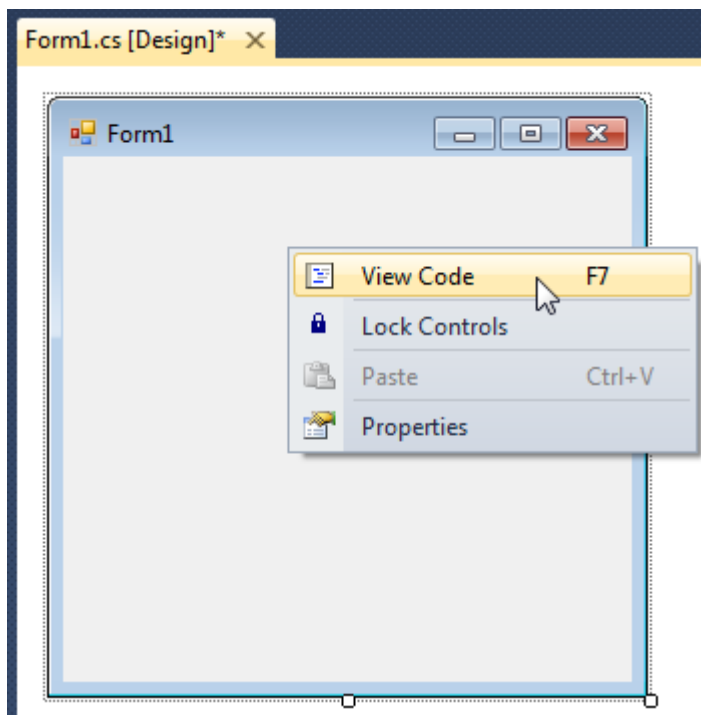


Set the Name = "CustomersOrders" properties in the Relation opened window. Select the Key Columns and Foreign Key Columns = "CustNo" in Columns. Click "Ok".



Step 8

Right click on the form and select "View Code" in the context menu to view code.



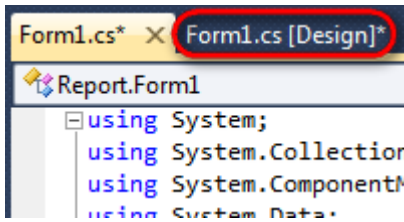
Add the following code to the class constructor in order to fill data source.



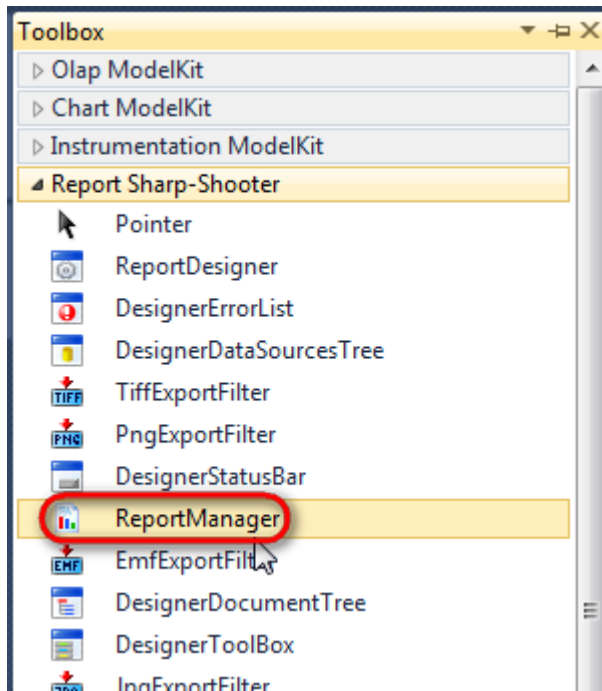
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CustNo"] = 1;
    row["CompanyName"] = "Bon App'";
    row["City"] = "Paris";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CustNo"] = 2;
    row["CompanyName"] = "Chop-suey Chinese";
    row["City"] = "London";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CustNo"] = 3;
    row["CompanyName"] = "Maison Dewey";
    row["City"] = "Paris";
    dataTable1.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 1;
    row["OrderID"] = "00001";
    row["OrderDate"] = "2010,03,21";
    row["OrderSum"] = "50.00";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 1;
    row["OrderID"] = "00002";
    row["OrderDate"] = "2010,02,15";
    row["OrderSum"] = "14.50";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00010";
    row["OrderDate"] = "2010,04,03";
    row["OrderSum"] = "134.00";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00011";
    row["OrderDate"] = "2010,01,15";
    row["OrderSum"] = "45.45";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00013";
    row["OrderDate"] = "2010,02,01";
    row["OrderSum"] = "500.00";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 2;
    row["OrderID"] = "00101";
    row["OrderDate"] = "2009,12,30";
    row["OrderSum"] = "6.03";
    dataTable2.Rows.Add (row);
    row = dataTable2.NewRow ();
    row["CustNo"] = 3;
    row["OrderID"] = "00666";
    row["OrderDate"] = "2010,06,06";
    row["OrderSum"] = "66.66";
    dataTable2.Rows.Add (row);
}
```


Step 9

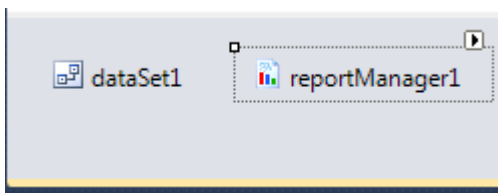
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

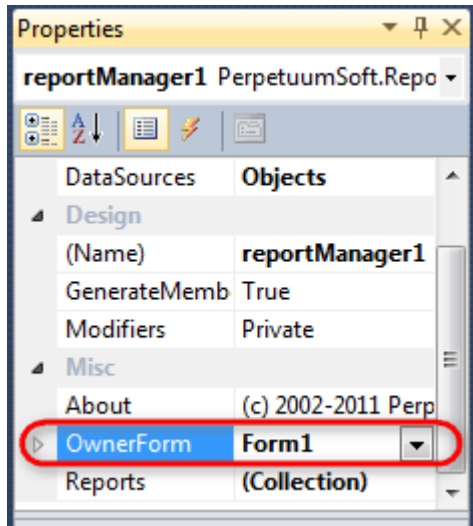


The component is available in the lower part of the window.



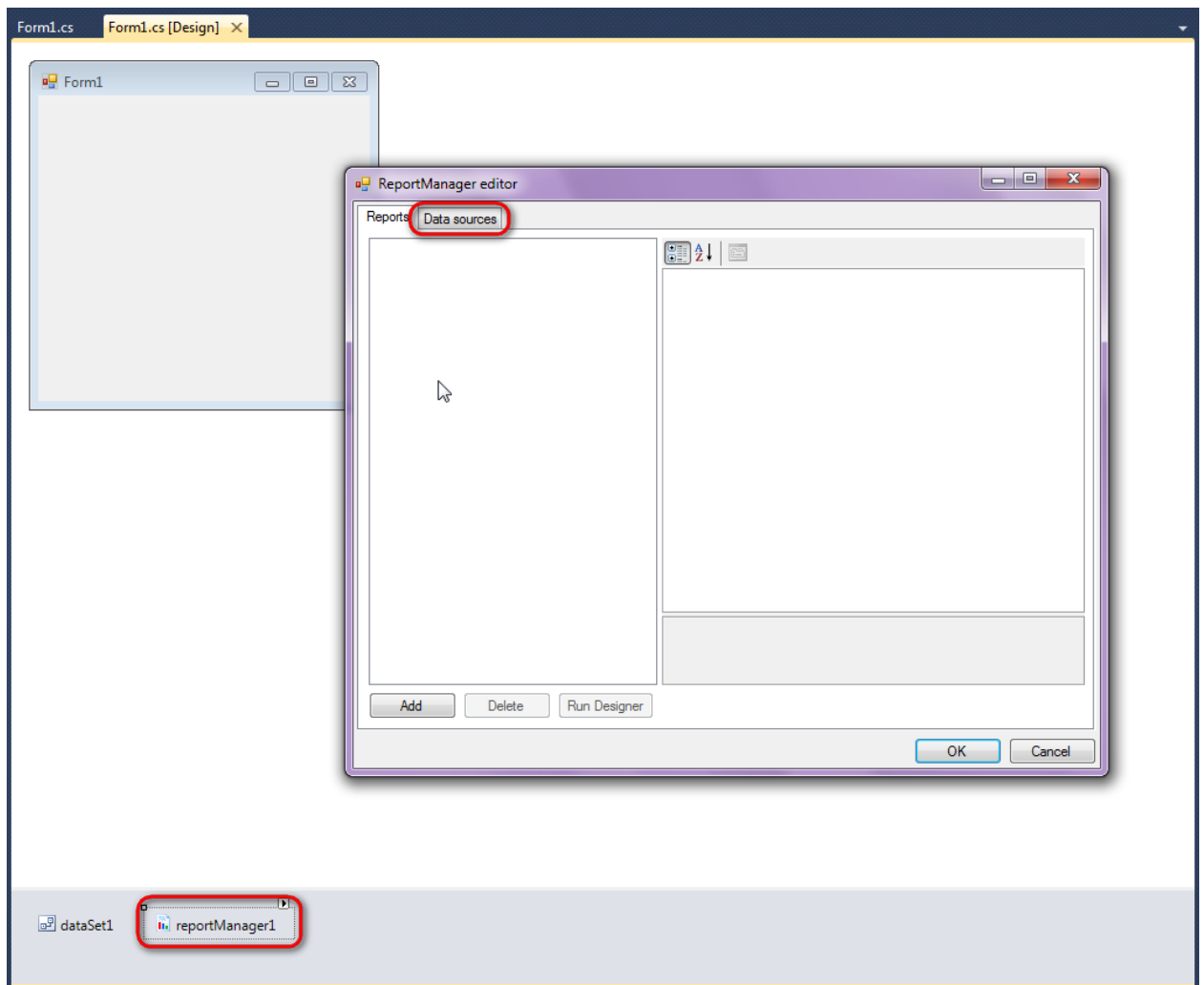
Step 10

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

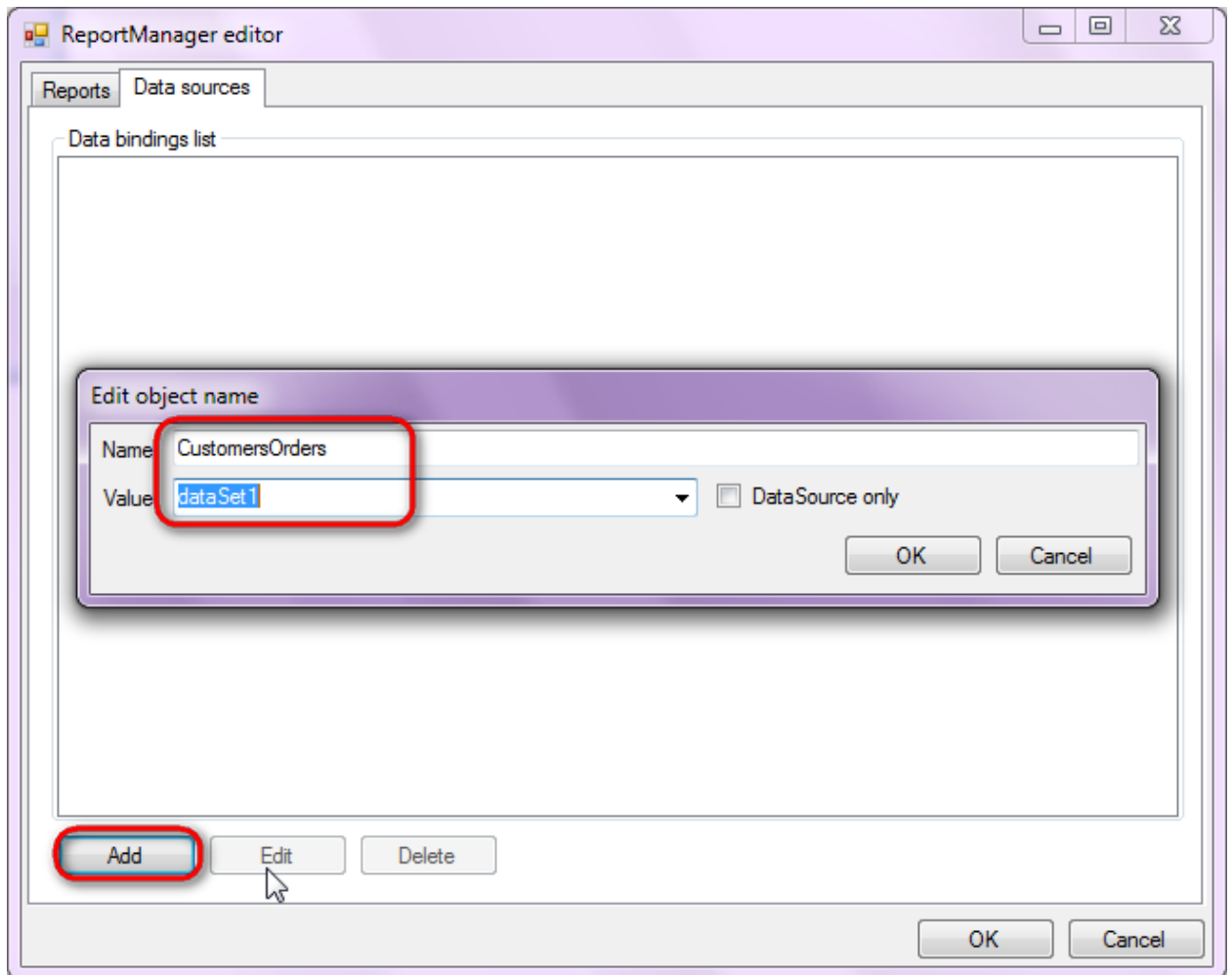


Step 11

Double click on ReportManager to open ReportManager editor.

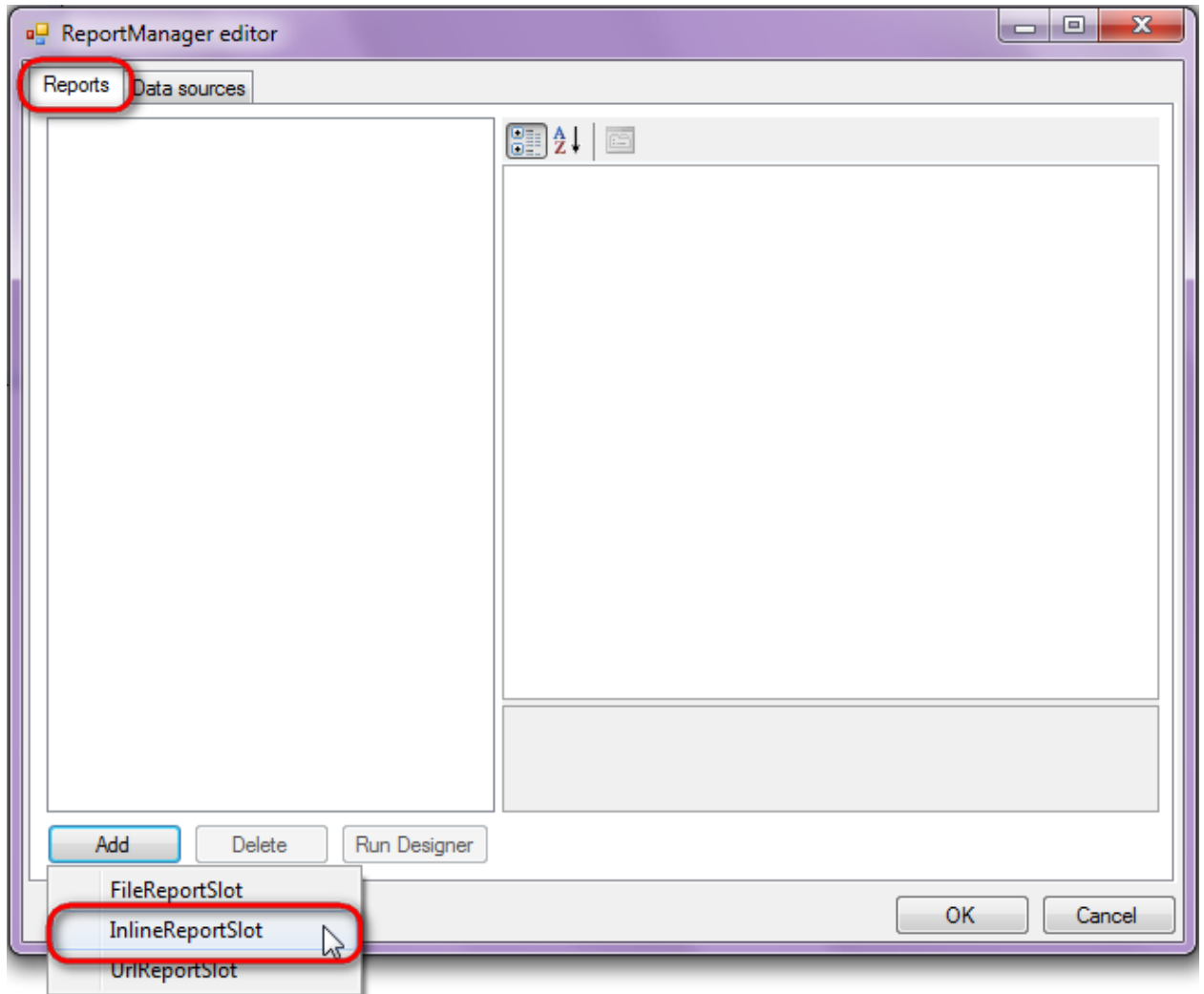


Go to "Data sources" tab, click "Add", set data source name - "CustomersOrders", select data source value - "dataSet1".



Step 12

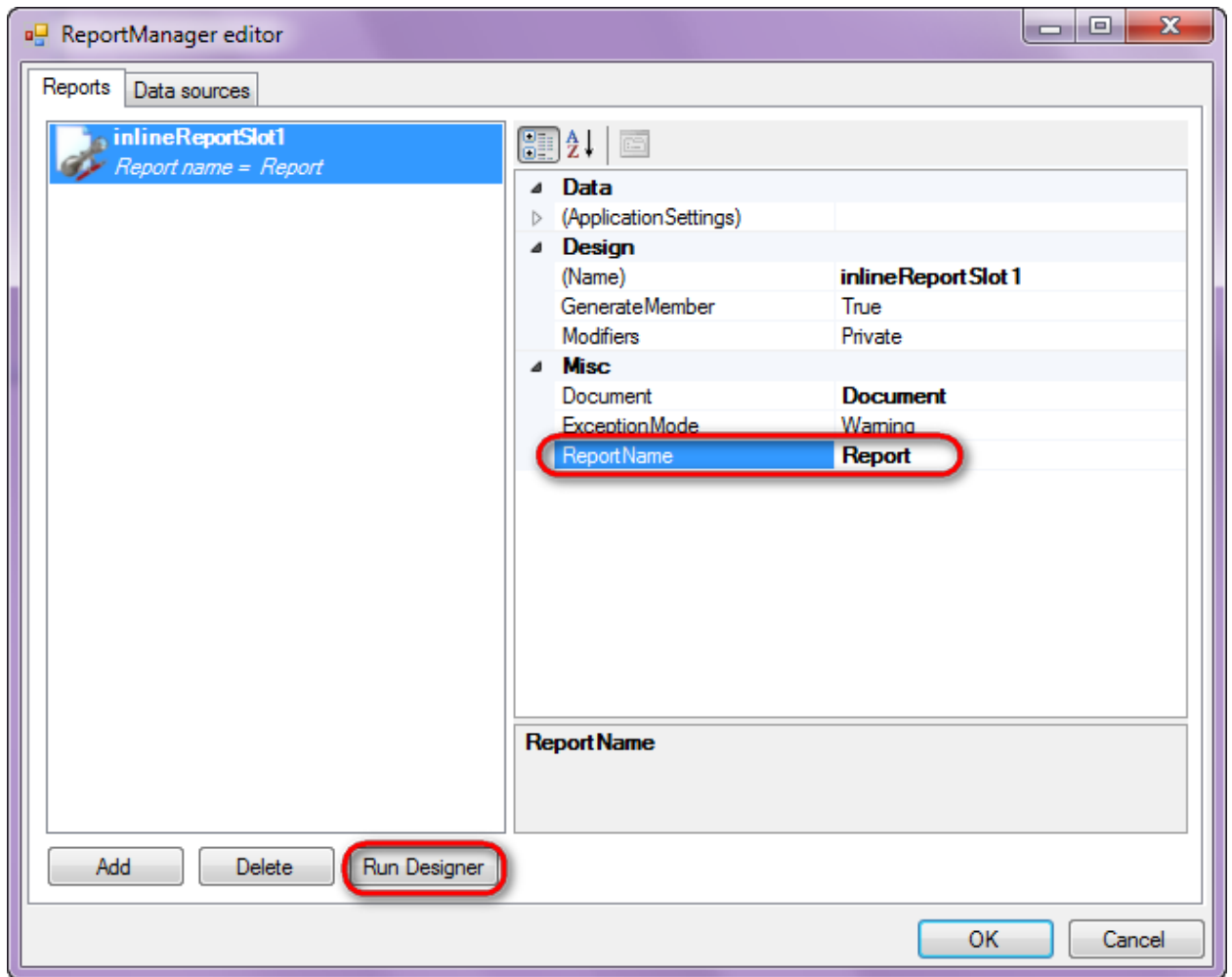
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 13

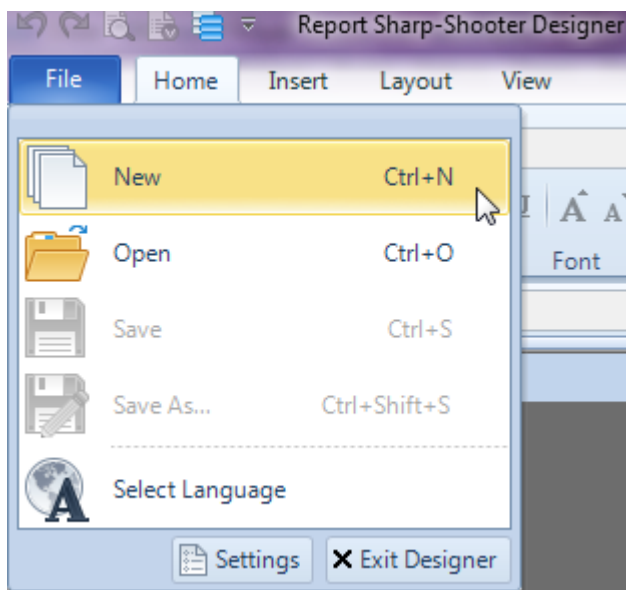
Set name of the report in the property ReportName – "Report".

Click "Run Designer" in order to open template editor - Report Designer.

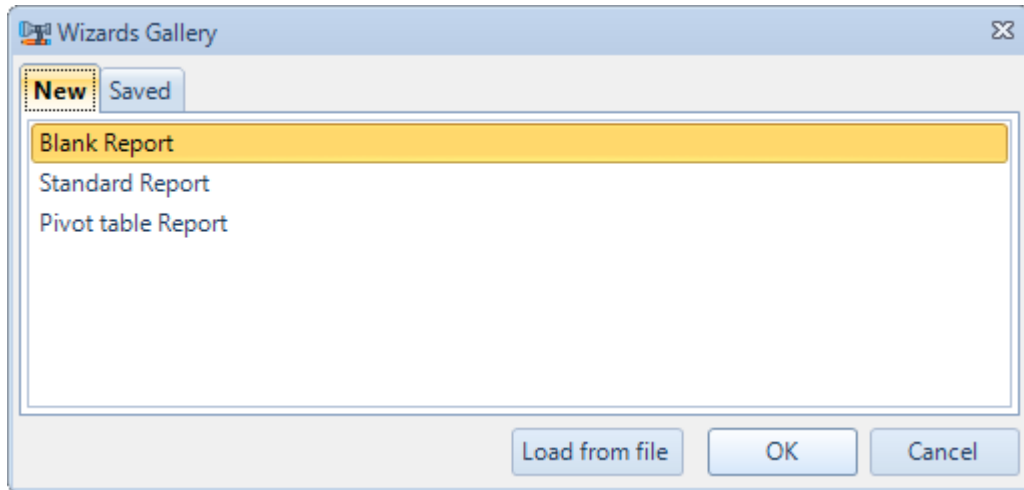


Step 14

Create new empty template – select File\New from the main menu.

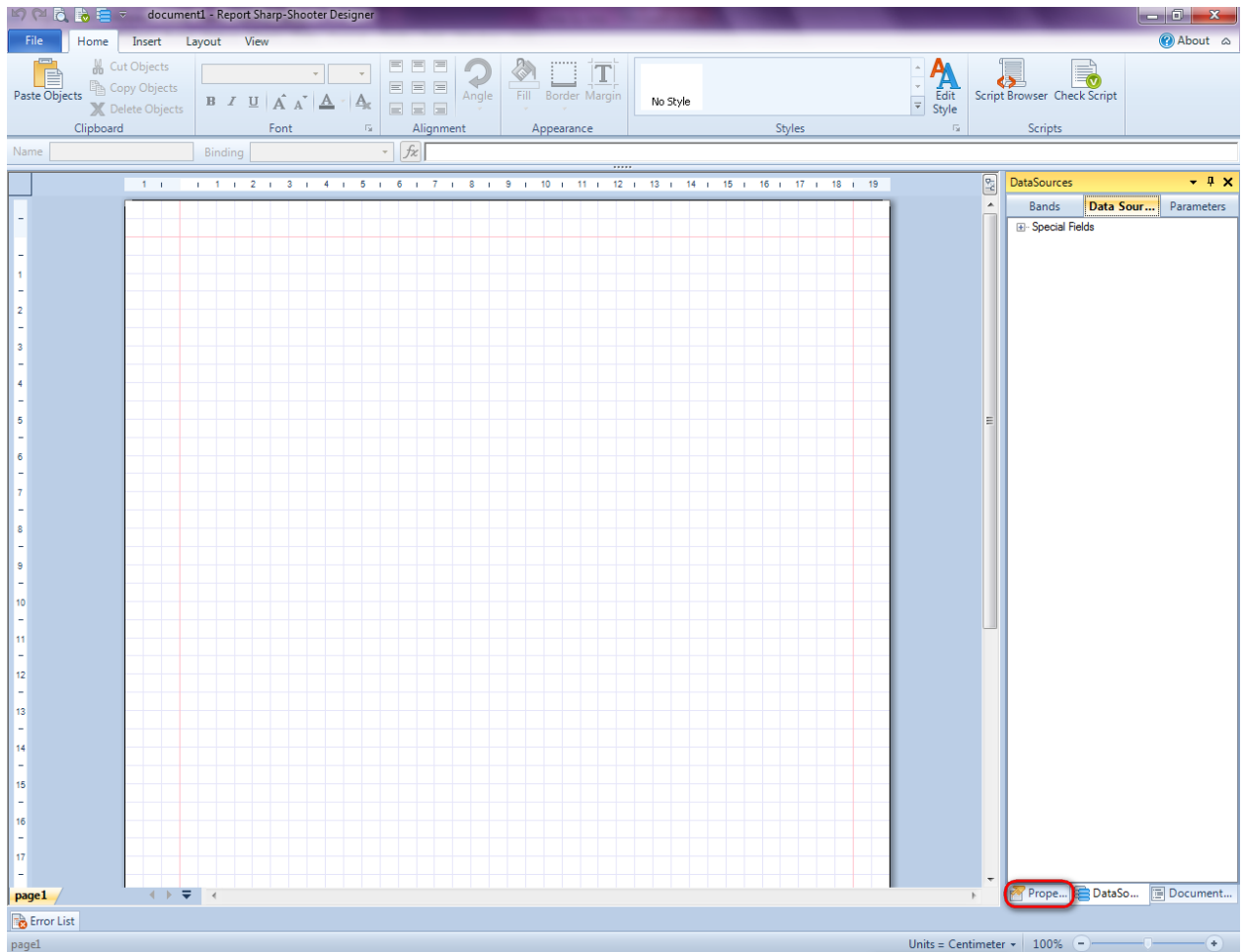


Select "Blank Report" in the Wizards Gallery and click "OK".

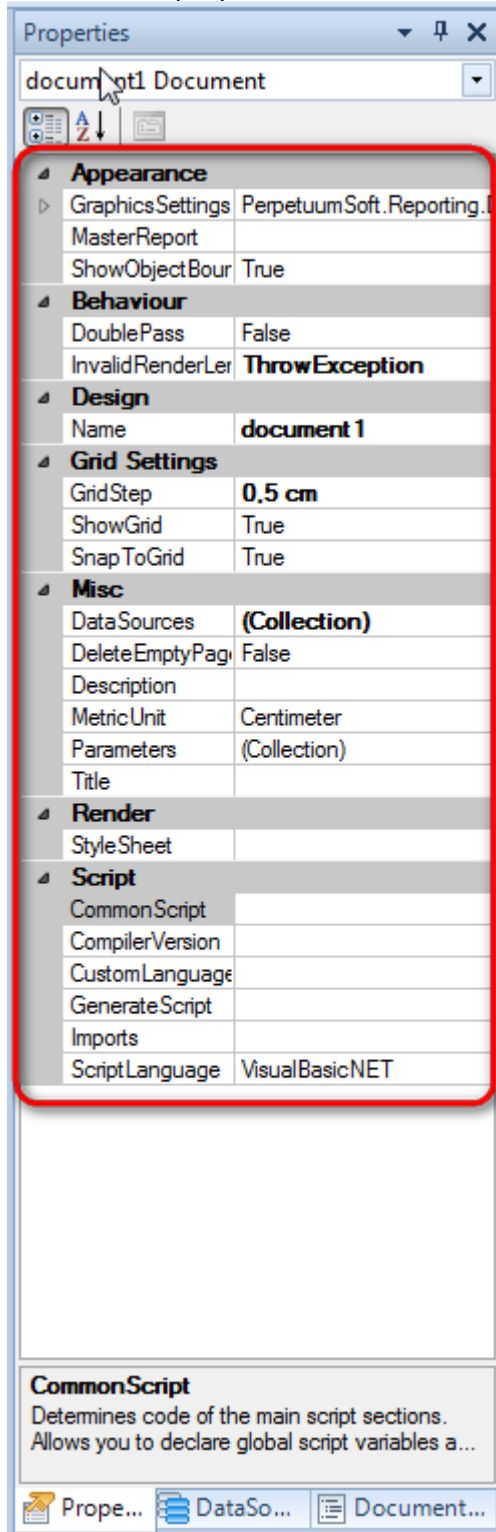


Step 15

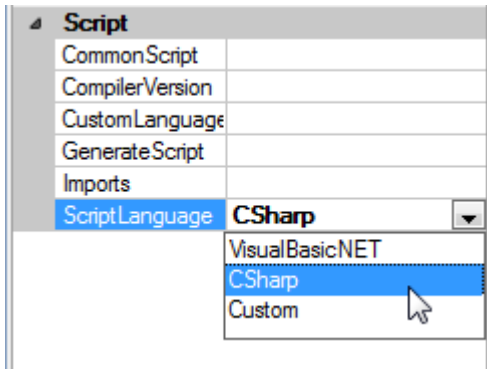
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

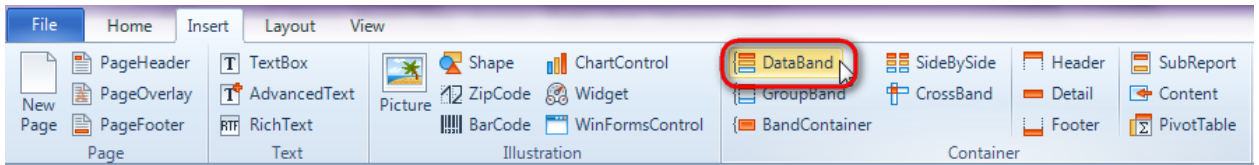


Set property ScriptLanguage = CSharp.



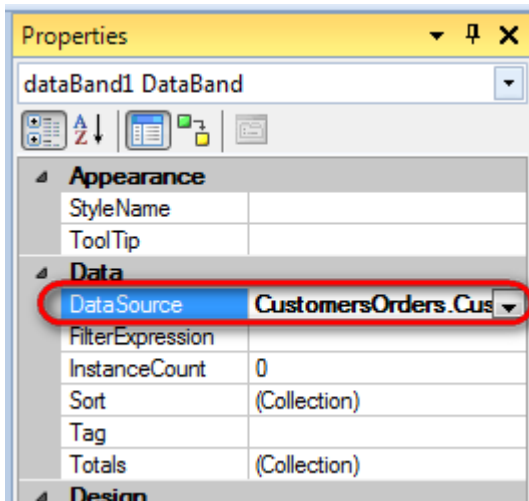
Step 16

Press "DataBand" button on the Insert tab in the group Container.



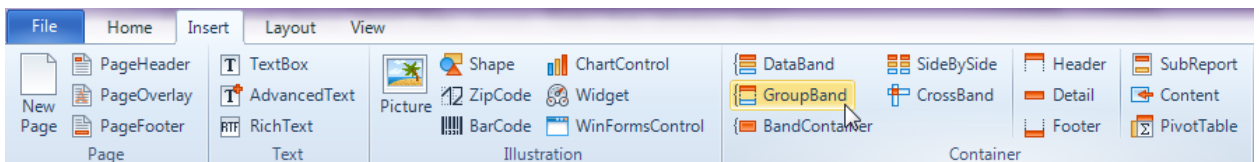
Click on the template area to add DataBand band to the template.

Set data source in the property DataSource = CustomersOrders.Customers.



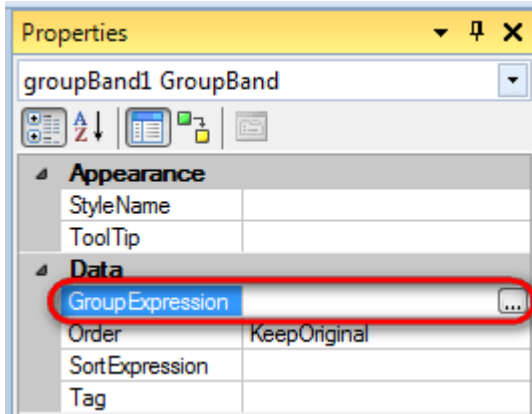
Step 17

Press "GroupBand" button on the Insert tab in the group Container.

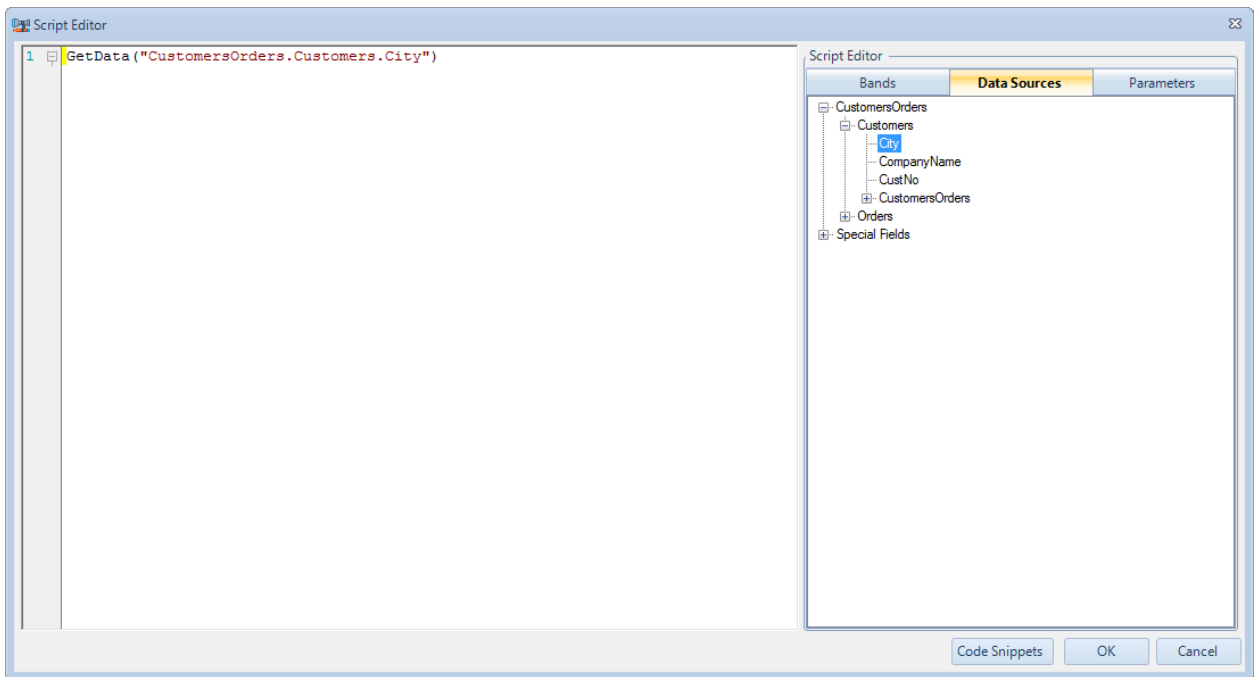


Click on the DataBand area to add GroupBand inside DataBand.

Select GroupExpression property on the "Properties" tab. Click button  to open property editor – Script Editor.



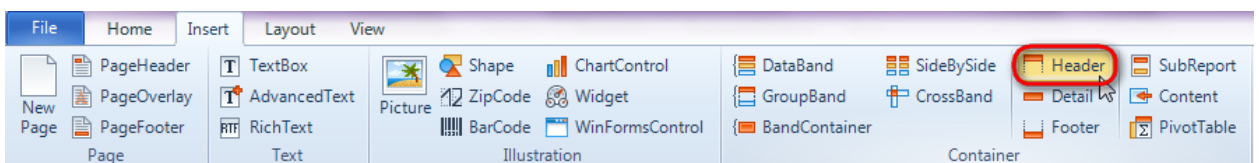
Set condition to group records by city "GetData("CustomersOrders.Customers.City")".



```
dataBand1:DataBand DataSource = CustomersOrders.Customers
groupBand1:GroupBand Group = GetData("CustomersOrders.Customers.City")
end of groupBand1
end of dataBand1
```

Step 18

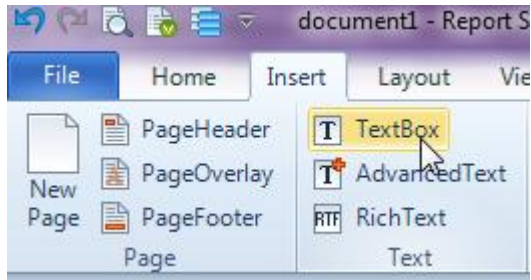
To display header of every page create Header band – press "Header" button on the Insert tab in the group Container.



Click on the GroupBand area to add Header band inside GroupBand.

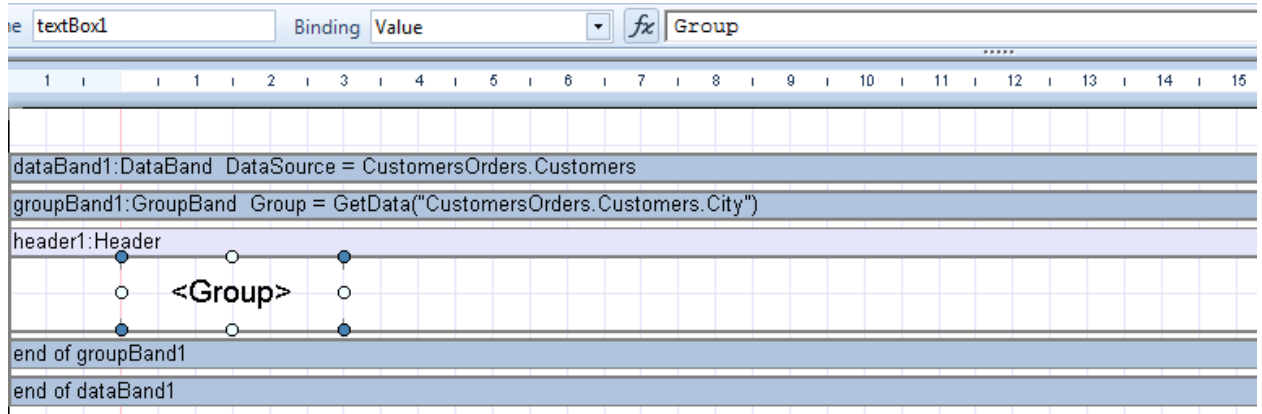
Step 19

Press button "TextBox" on the Insert tab in the group Text.



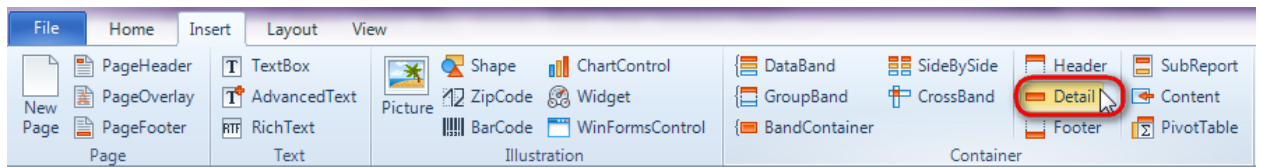
Click on the Header area to add TextBox element inside Header.

Set Value property to "Group".



Step 20

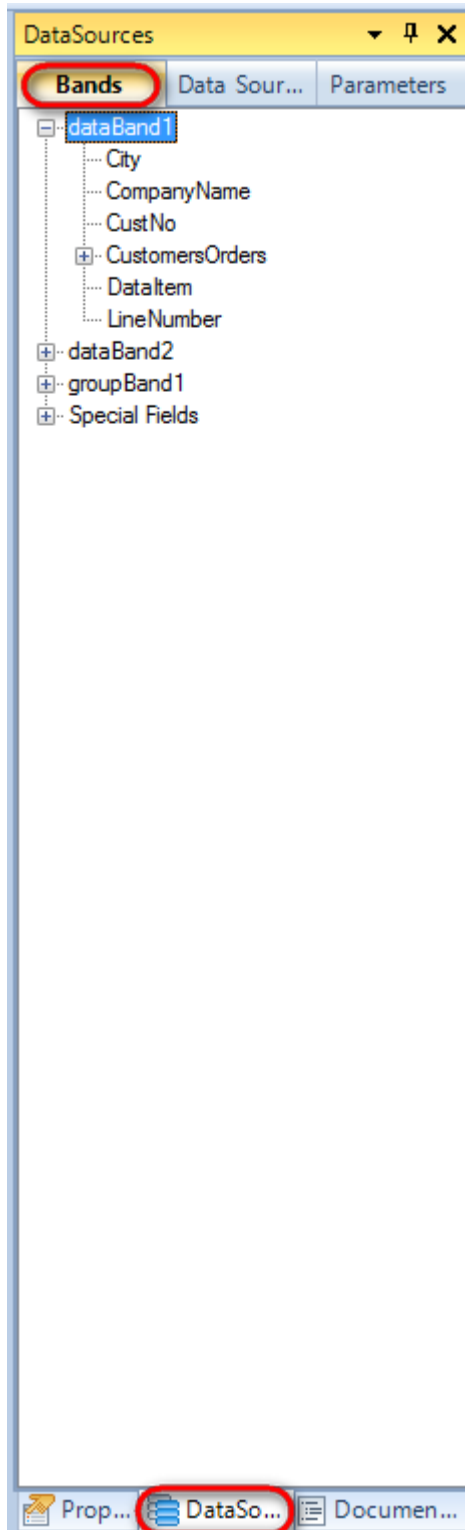
Press "Detail" button on the Insert tab in the group Container.



Click on the GroupBand area to add Detail band inside GroupBand.

Step 21

Go to "DataSources" tab.



Drag and drop "CompanyName" field from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.



```

dataBand1:DataBand DataSource = CustomersOrders.Customers
groupBand1:GroupBand Group = GetData("CustomersOrders.Customers.City")
header1:Header
<Group>
detail1:Detail
<dataBand1
["CompanyName"]>
end of groupBand1
end of dataBand1

```

Step 22


Add DataBand inside GroupBand. Set property DataSource = CustomersOrders.Customers.CustomresOrders

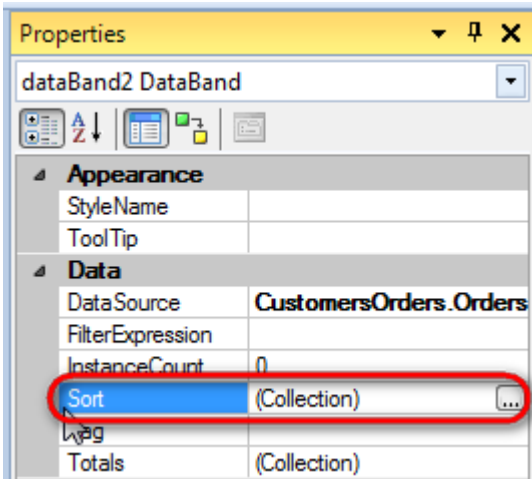
```


dataBand1:DataBand DataSource = CustomersOrders.Customers
groupBand1:GroupBand Group = GetData("CustomersOrders.Customers.City")
header1:Header
<Group>
detail1:Detail
<dataBand1
["CompanyName"]>
dataBand2:DataBand DataSource = CustomersOrders.Customers.CustomresOrders
end of dataBand2
end of groupBand1
end of dataBand1

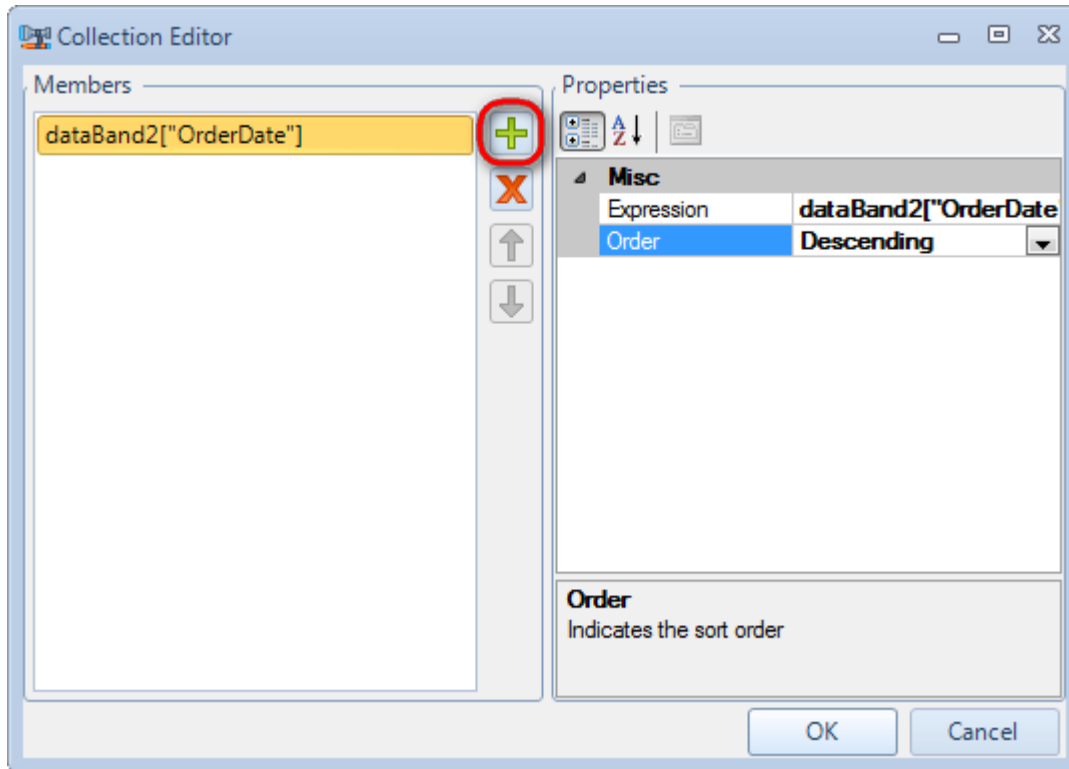
```

Step 23


Select dataBand2, select Sort property, click button  to open Collection Editor – Sort property editor.

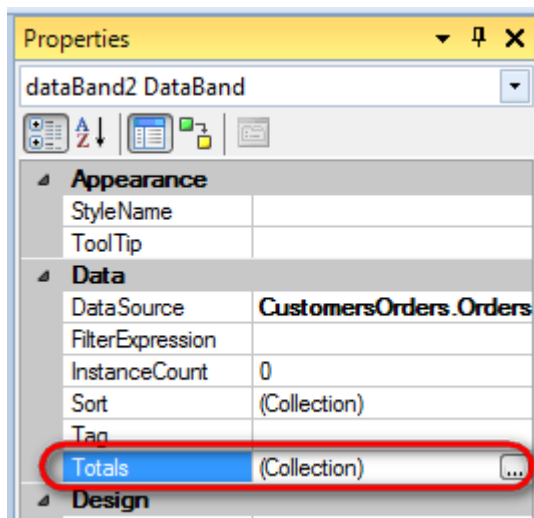



In the Collections Editor, click  button. Set the following properties: Expression = dataBand2["OrderDate"], Order = Descending to soft orders by date in descending order.

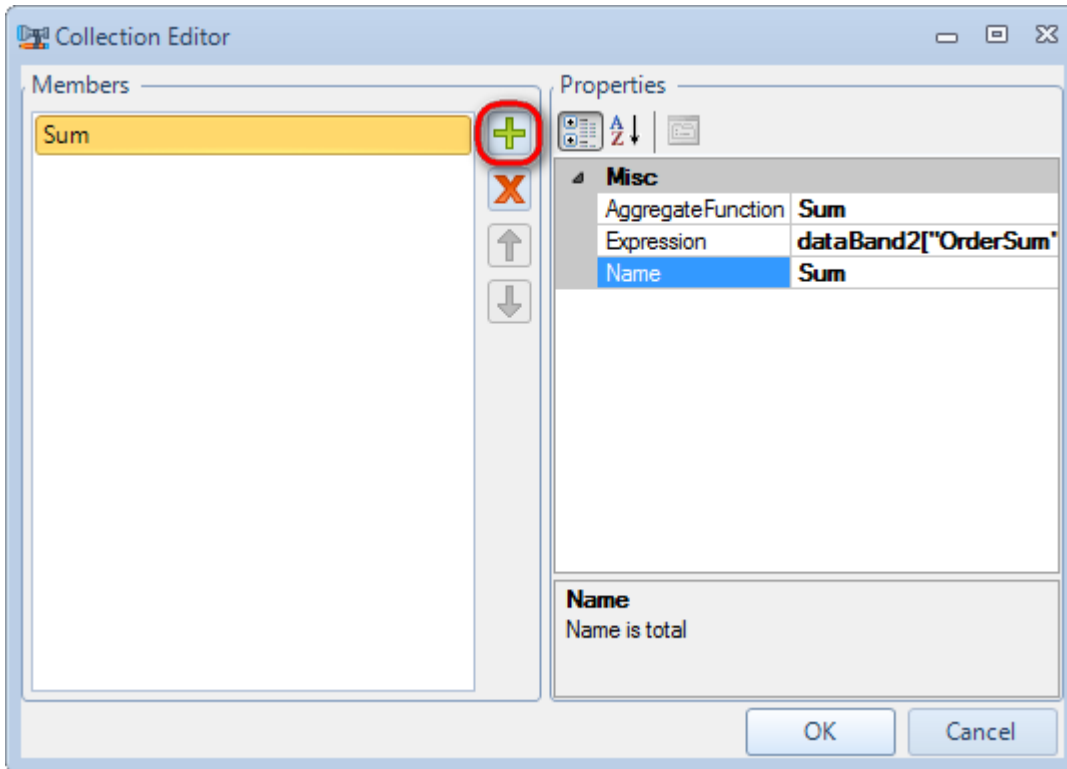


Step 24


Select dataBand2, select Totals property, click  to open Collections Editor – Totals property editor.

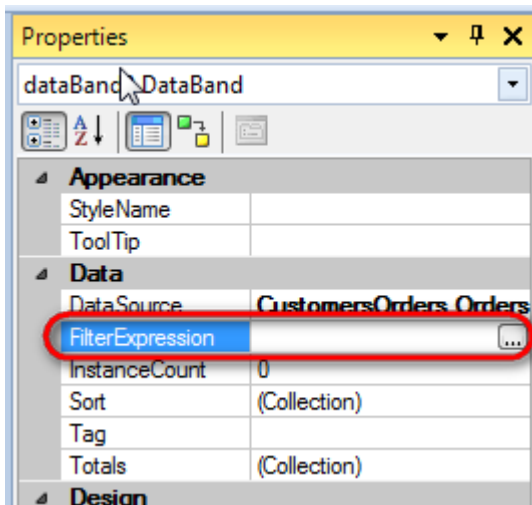


In the Collections Editor click  button. Set the following properties: AggregateFunction = Sum, Expression = dataBand2["OrderSum"], Name = Sum.



Step 25

Select dataBand2, select FilterExpression property, click button  to open Script Editor – FilterExpression property editor.



Enter the following code:

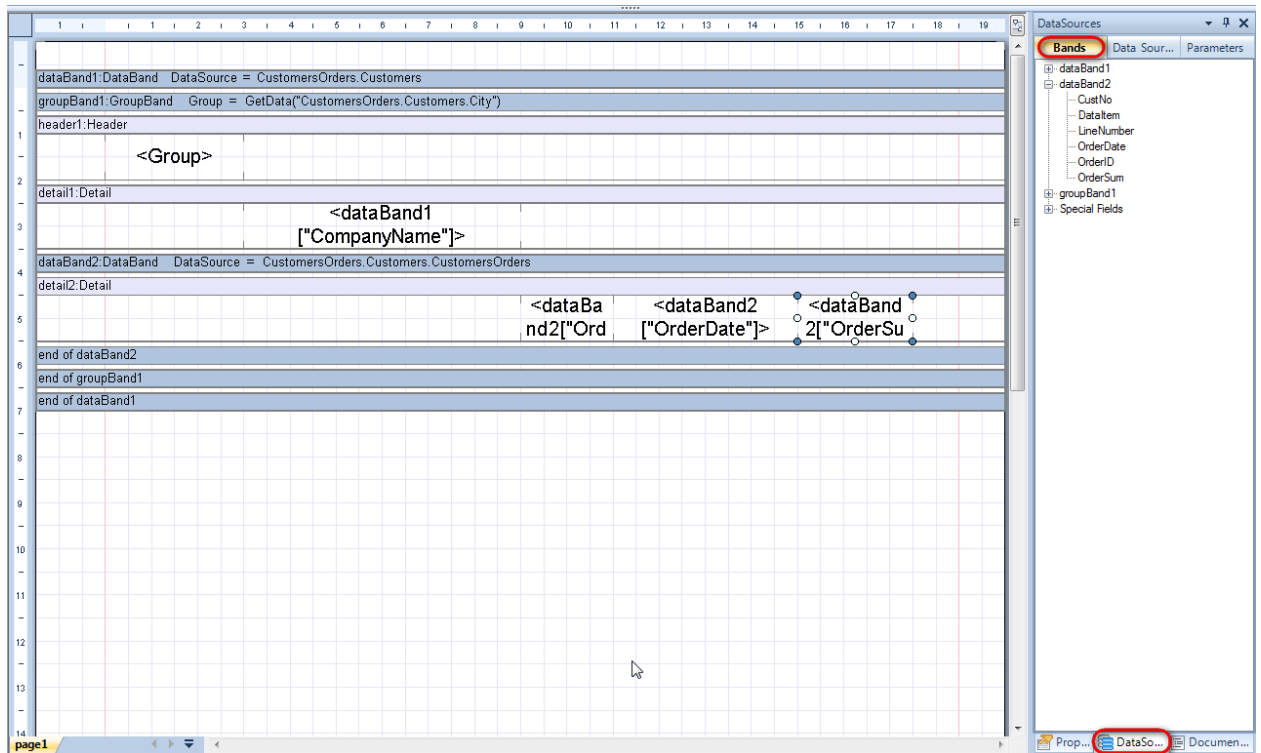
```
"((DateTime)dataBand2["OrderDate"]) > new DateTime(2010,01,01)"
```

Step 26


Add Detail band to the dataBand2.

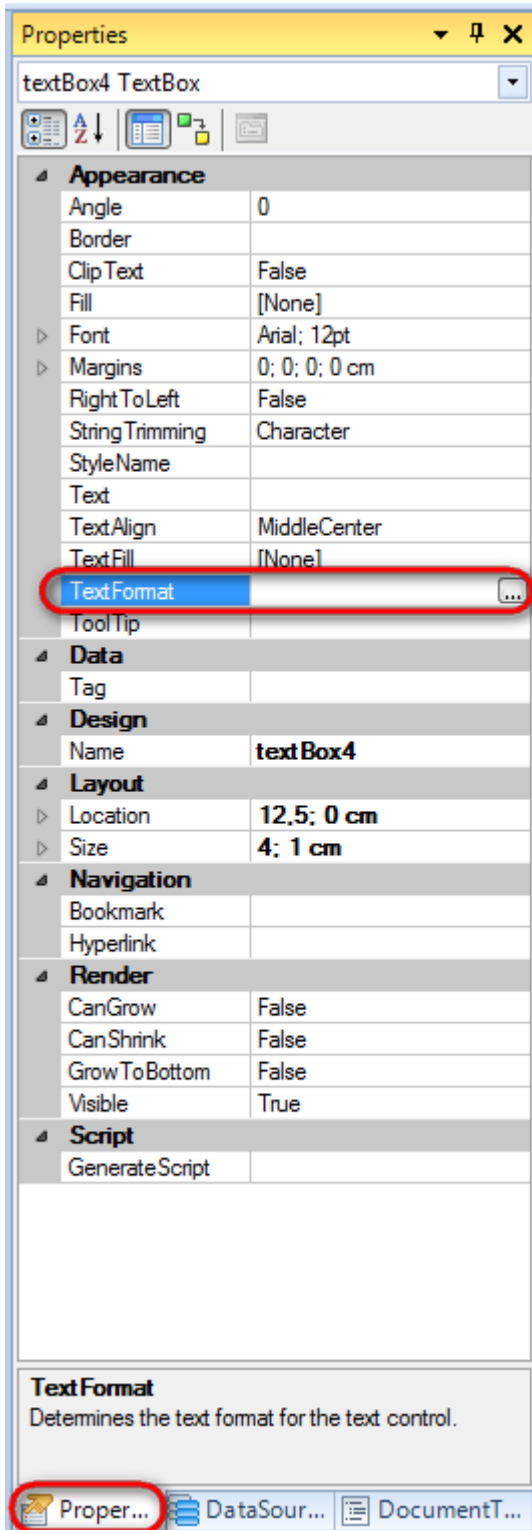
Step 27

Go to DataSources tab and drag and drop the following fields: "OrderID", "OrderDate", and "OrderSum" from the dataBand2 tree to the detail2 band.

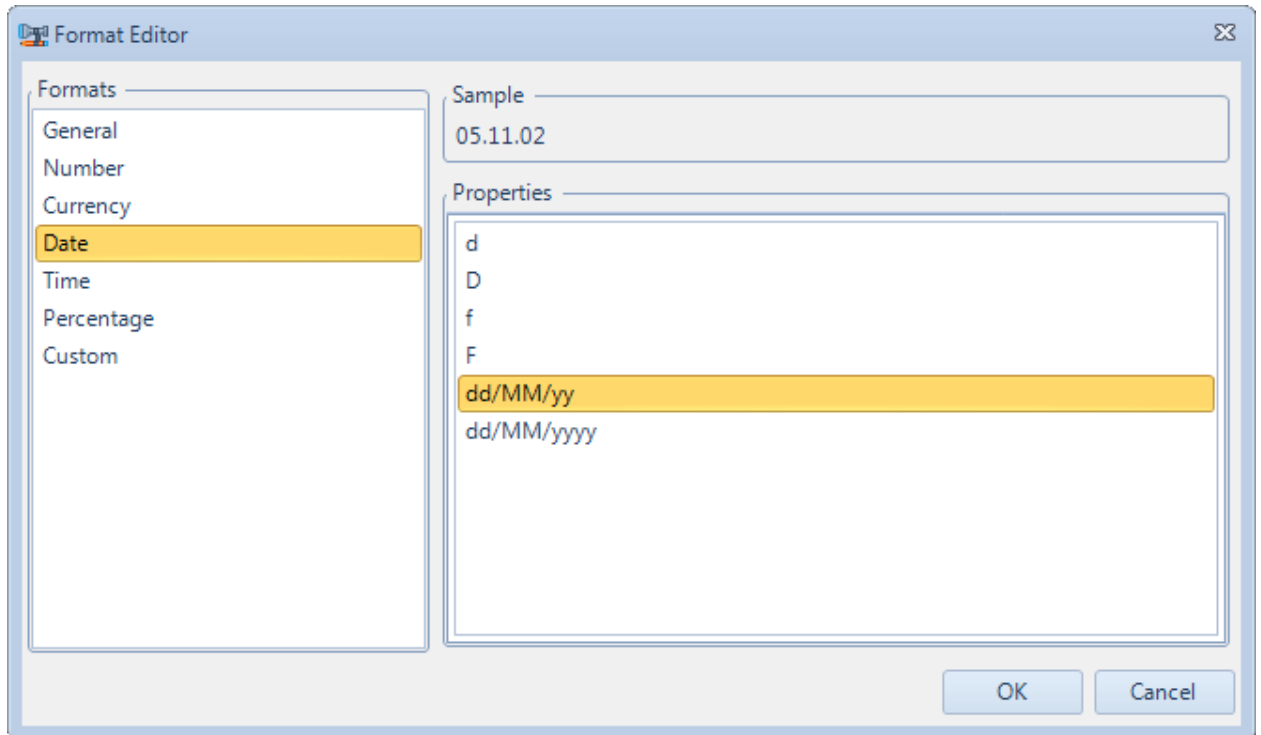


Step 28


Select TextBox element with the Value = dataBand2["OrderDate"]. Select TextFormat property, click button  to open form of selecting text format.



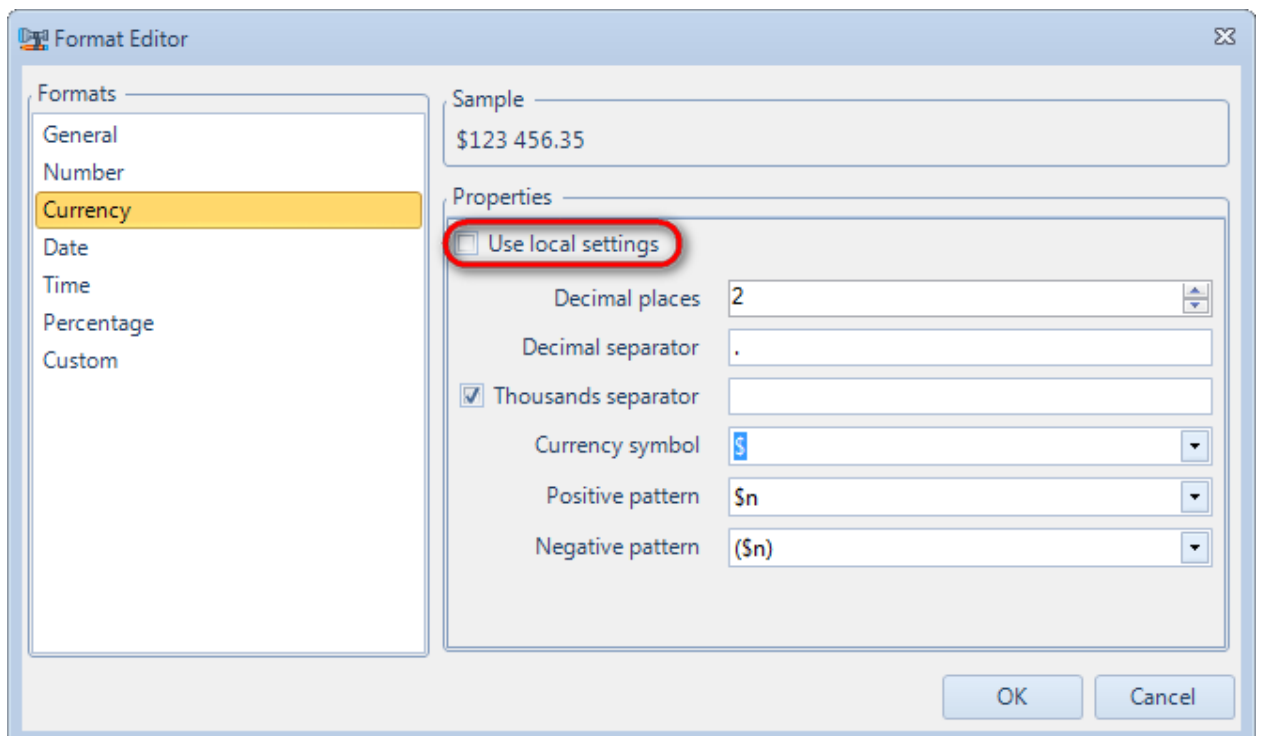
Select "Date" in the "Formats" list; item "dd/MM/yy" in the "Properties" list.



Step 29

In the same way, the select TextBox element with the Value = dataBand2["OrderSum"]. Select the TextFormat property, click the  button to open form of the selected text format.

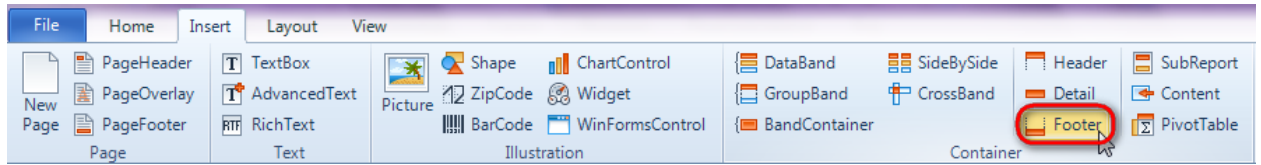
Select "Currency" in the "Formats" list. Uncheck "Use local settings" in the "Properties" tab. Set the other properties as you need.





Step 30

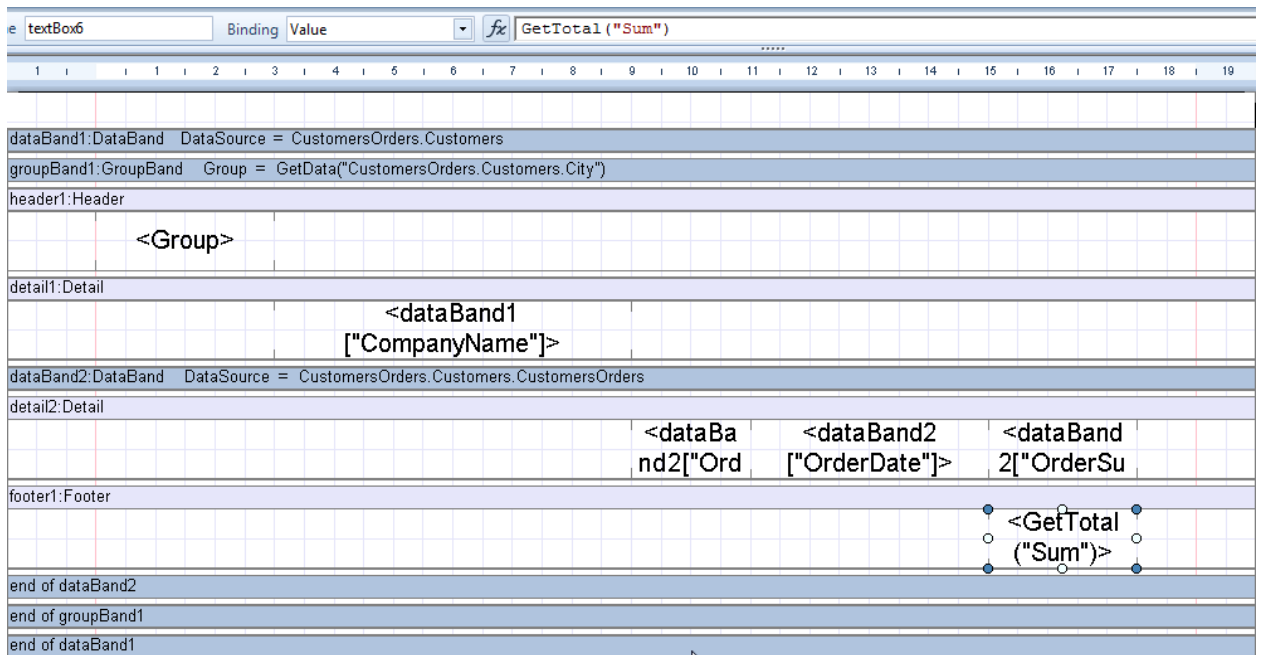
To display footer containing summarized value of the OrderSum field create Footer band – press “Footer” button on the Insert tab in the group Container.



Click on the dataBand2 area to add Footer band inside dataBand2.

Step 31

Add the TextBox element inside Footer band, set the Value = GetTotal("Sum") property. Set the element’s properties as you did it in step 29.



Step 32

Save template, close Report Designer.

Step 33

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

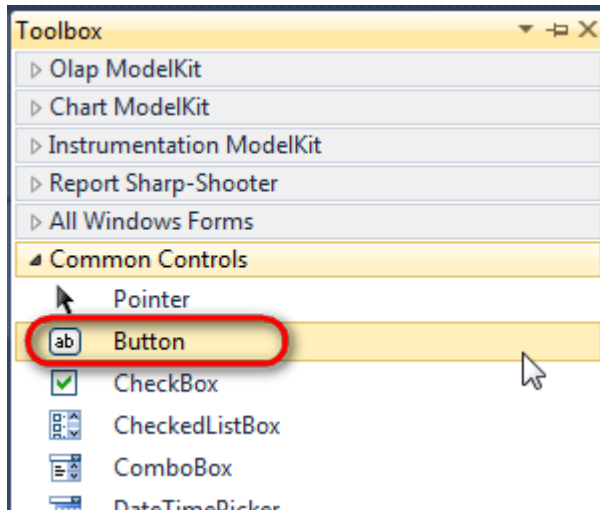
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row ["CustNo"] = 1;
    row ["CompanyName"] = "Bon App'";
    row ["City"] = "Paris";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["CustNo"] = 2;
    row ["CompanyName"] = "Chop-suey Chinese";
    row ["City"] = "London";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["CustNo"] = 3;
    row ["CompanyName"] = "Maison Dewey";
}
```



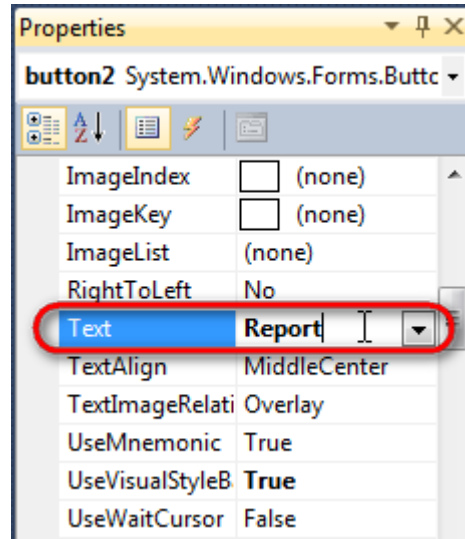
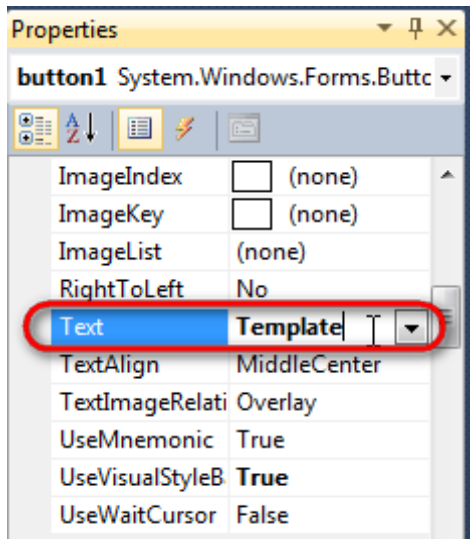
```
row["City"] = "Paris";
dataTable1.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 1;
row["OrderID"] = "00001";
row["OrderDate"] = "2010,03,21";
row["OrderSum"] = "50.00";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 1;
row["OrderID"] = "00002";
row["OrderDate"] = "2010,02,15";
row["OrderSum"] = "14.50";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00010";
row["OrderDate"] = "2010,04,03";
row["OrderSum"] = "134.00";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00011";
row["OrderDate"] = "2010,01,15";
row["OrderSum"] = "45.45";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00013";
row["OrderDate"] = "2010,02,01";
row["OrderSum"] = "500.00";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 2;
row["OrderID"] = "00101";
row["OrderDate"] = "2009,12,30";
row["OrderSum"] = "6.03";
dataTable2.Rows.Add(row);
row = dataTable2.NewRow();
row["CustNo"] = 3;
row["OrderID"] = "00666";
row["OrderDate"] = "2010,06,06";
row["OrderSum"] = "66.66";
dataTable2.Rows.Add(row);
inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 34

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



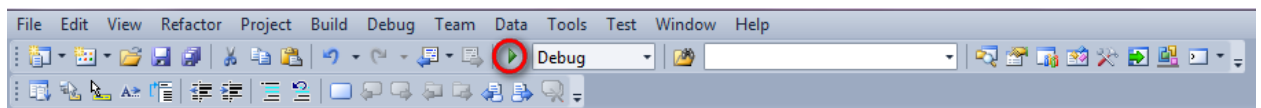
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

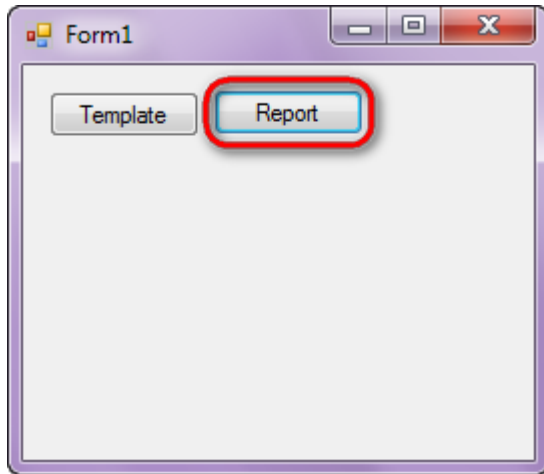
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 35

Click “Start Debugging” on the Visual Studio toolbar in order to start application.



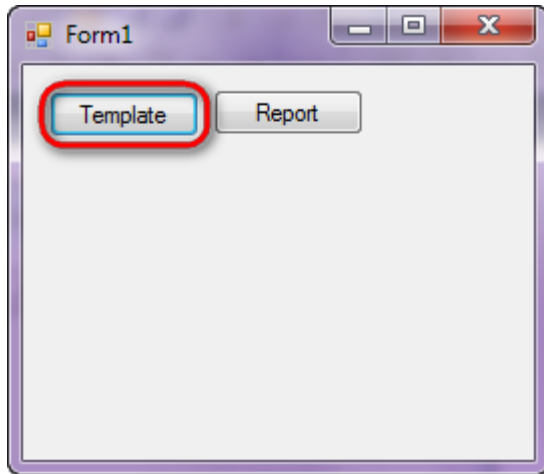
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.

Location	Item	ID	Date	Amount
Paris	Bon App'	00001	21.03.10	\$50.00
		00002	15.02.10	\$14.50
				\$64.50
Maison Dewey		00666	06.06.10	\$66.66
				\$66.66
				\$66.66
London	Chop-suey Chinese	00010	03.04.10	\$134.00
		00013	01.02.10	\$500.00
		00011	15.01.10	\$45.45
				\$679.45

To edit report template, close Report Viewer and click "Template" on the application form.



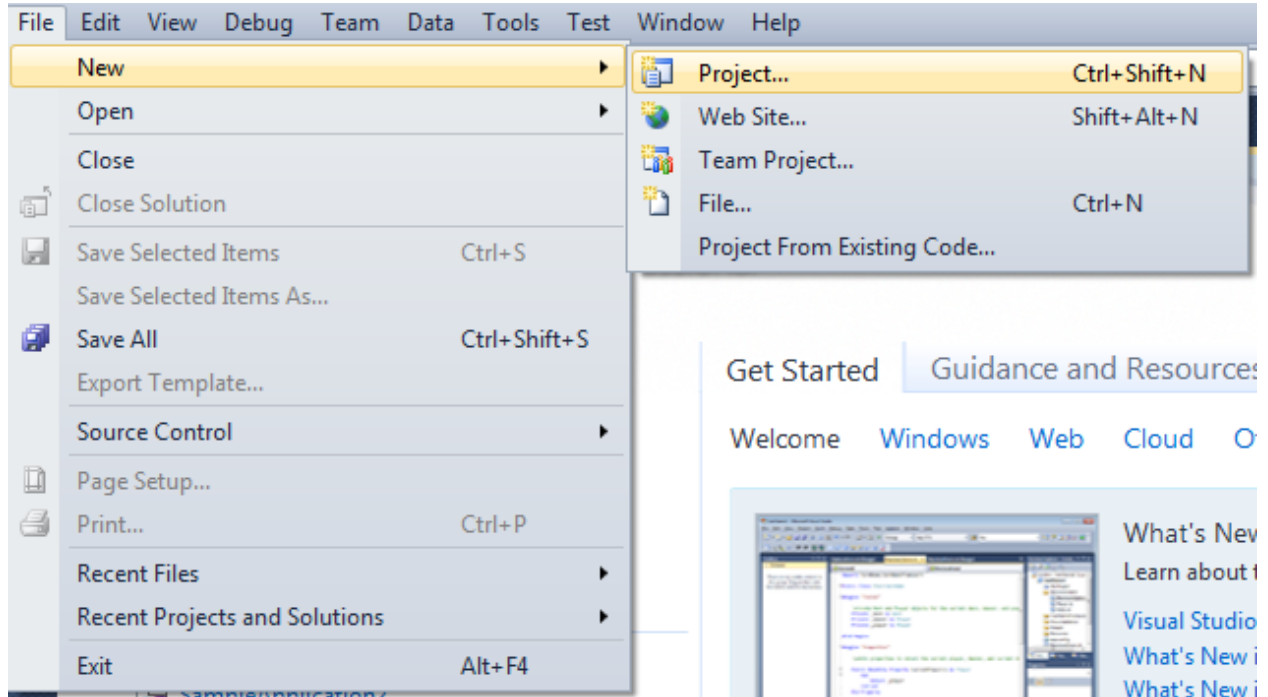


Managing Size

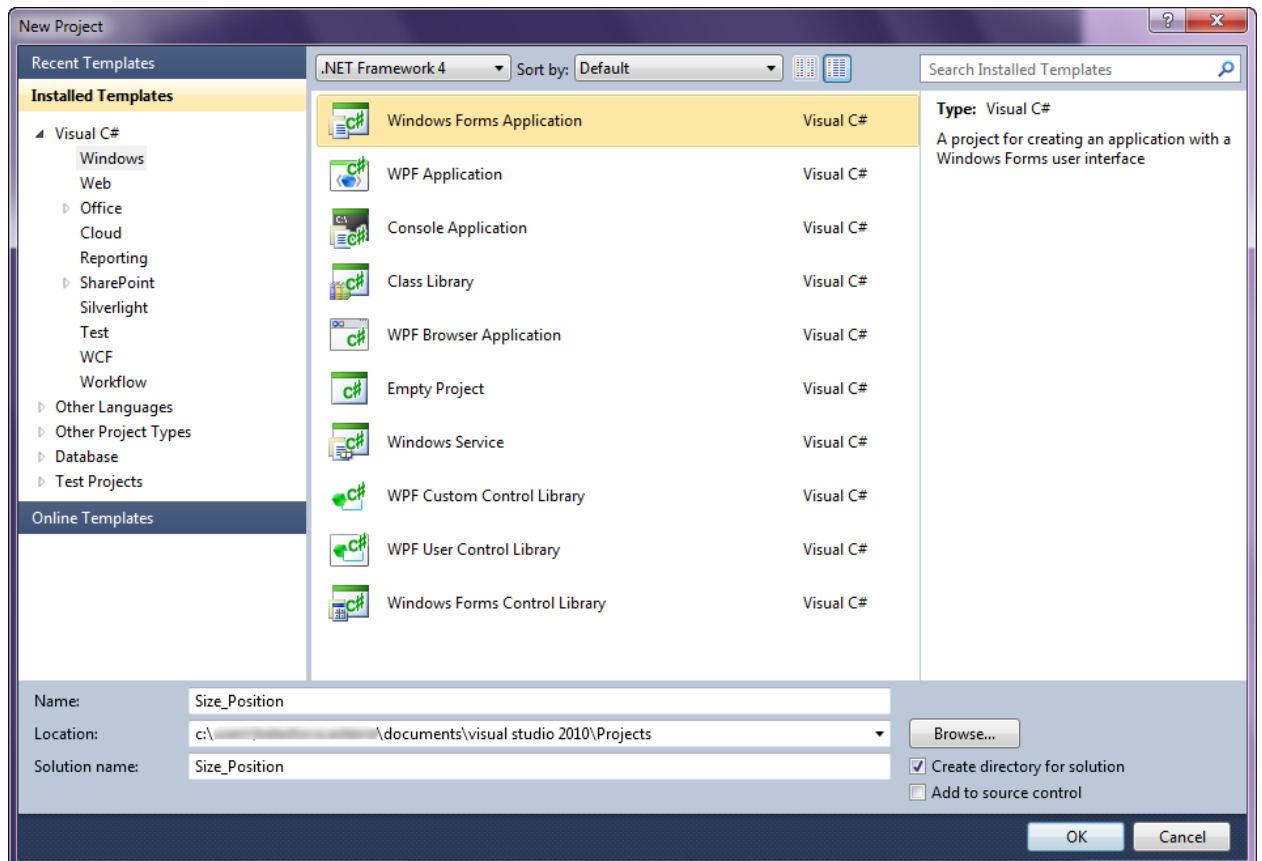
Template of a report containing information on employees – name, phone, note. Size of the TextBox containing a note is changed depending on data.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

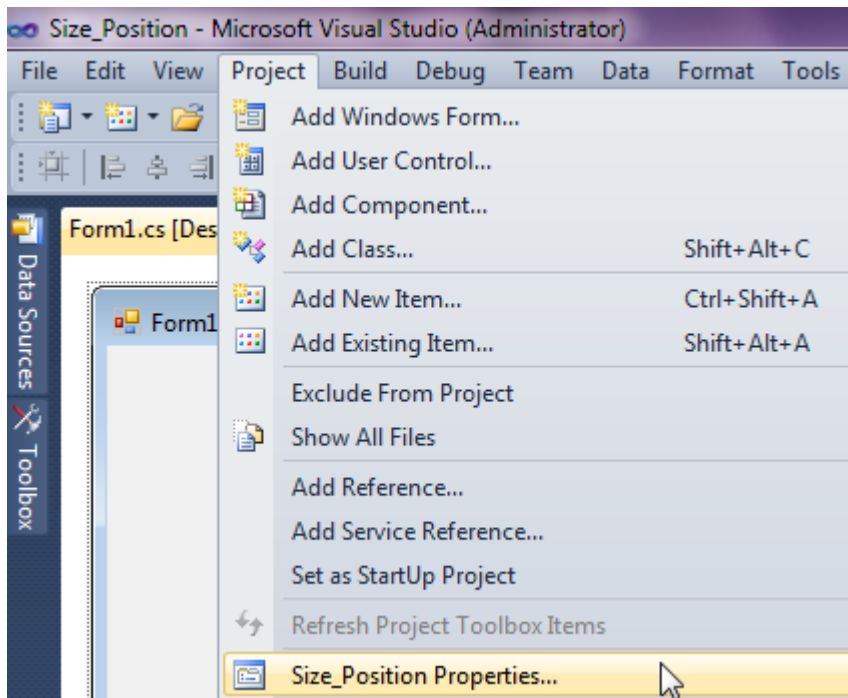


Select Windows Forms Application, set project name – “Size_Position”, set directory to save the project to.

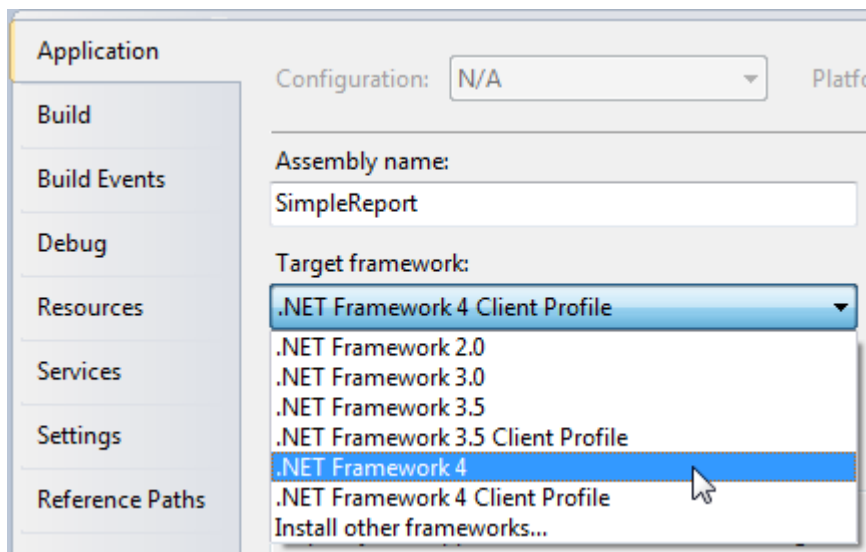


Step 2

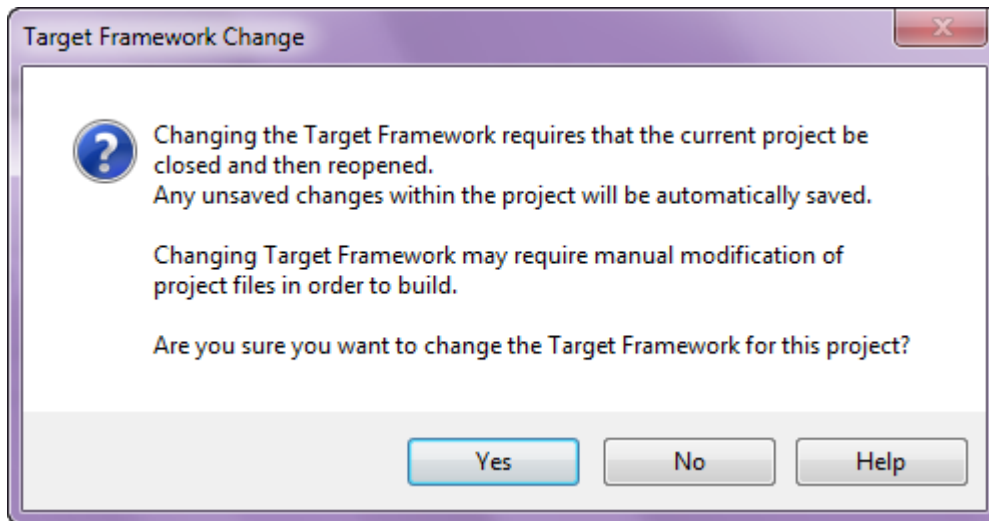
Change the project properties. Select the Project\Size_Position Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

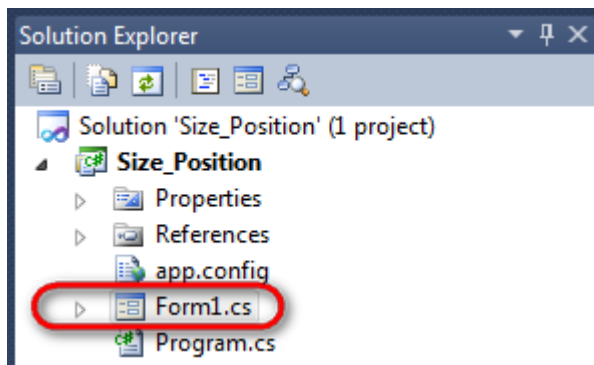


In the opened window press the "Yes" button.

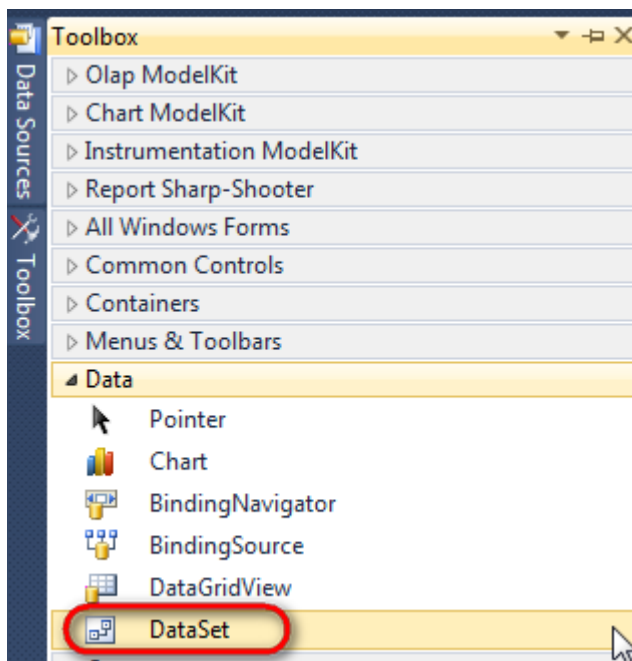


Step 3

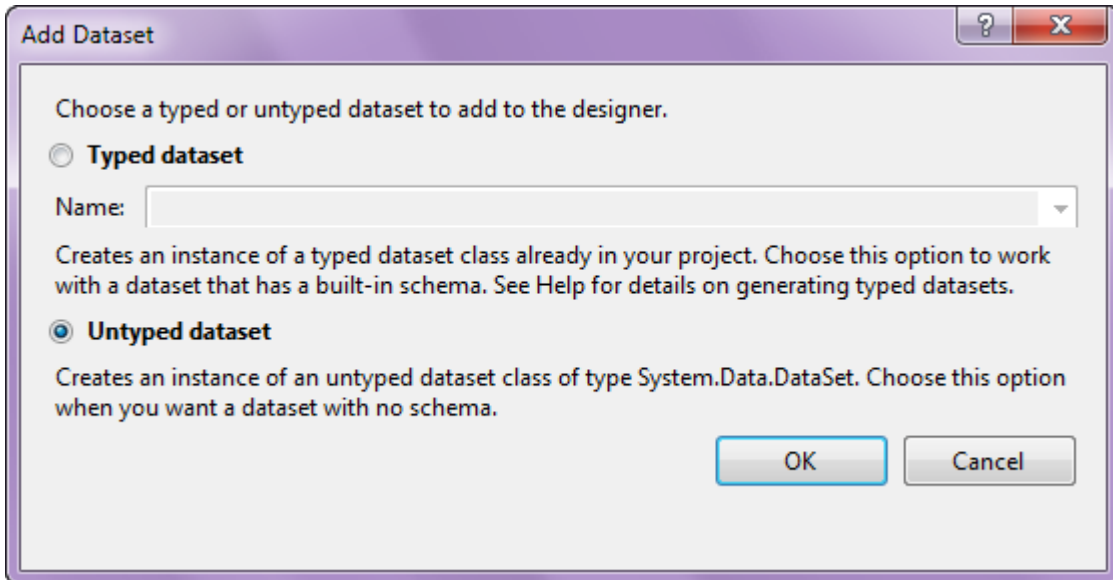
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



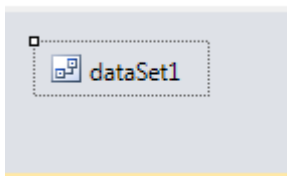
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

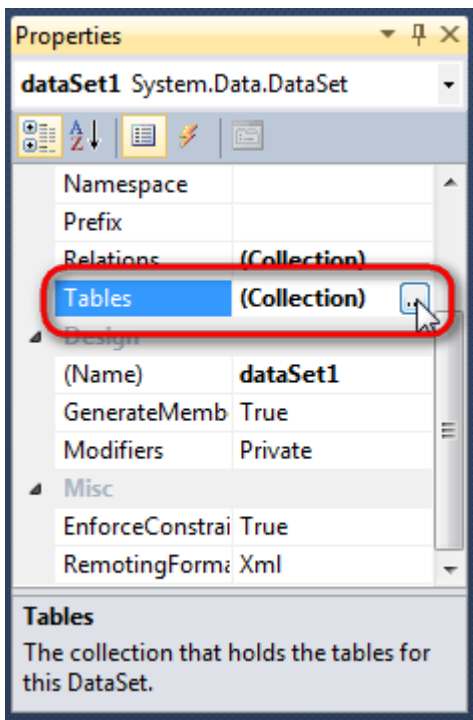


The component is available in the lower part of the window.

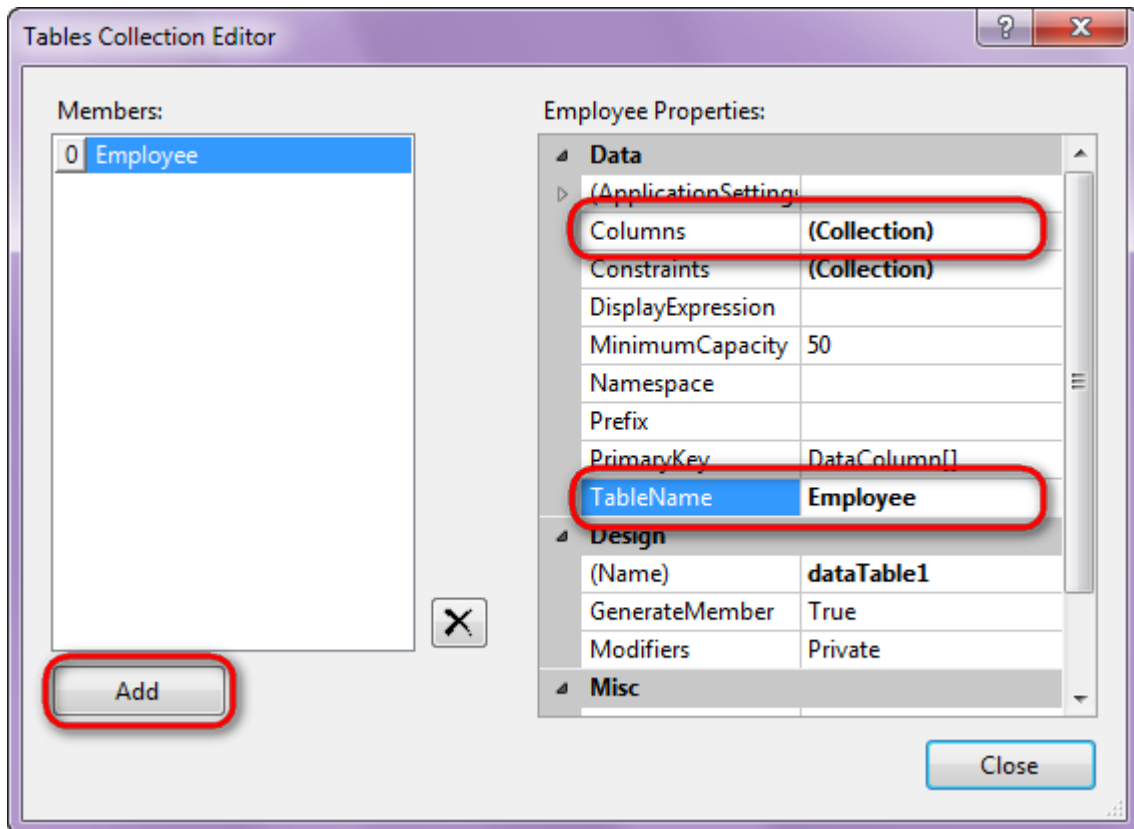


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.



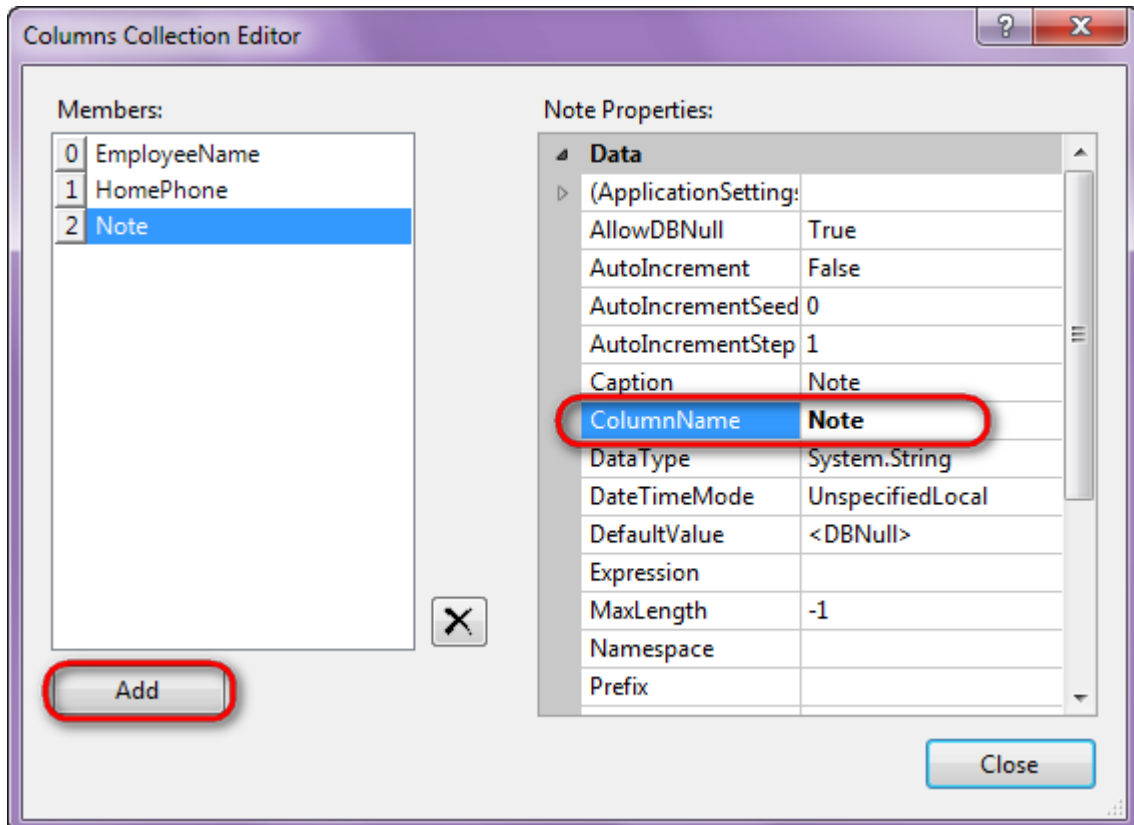
Click "Add" in order to add table. Set property TableName = Employee.



Step 5

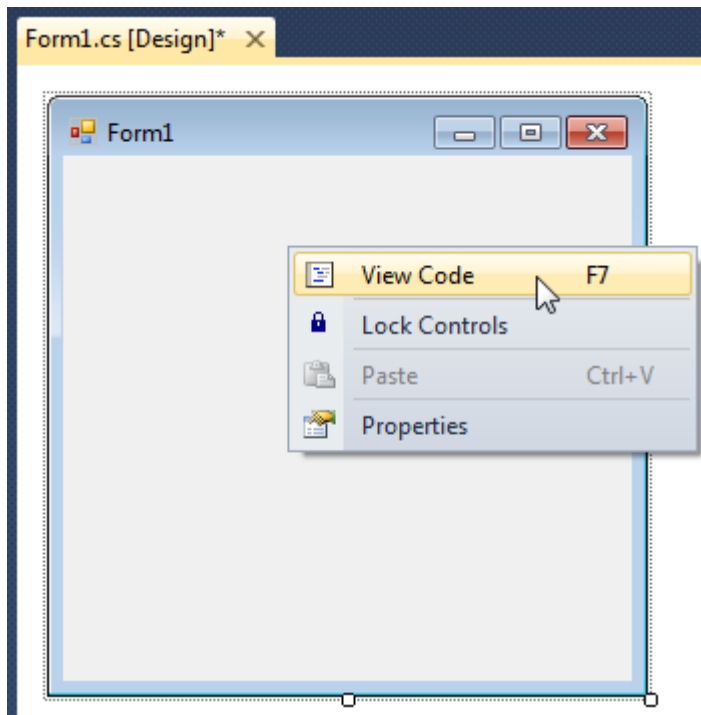
Select Columns property, click button  in order to open property editor.

Click "Add" to add a new column. Add three columns. Set ColumnName property to "EmployeeName", "HomePhone", and "Note".



Step 6

Right click on the form and select "View Code" in the context menu to view code.

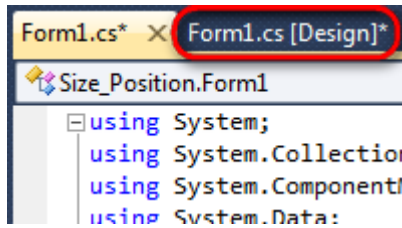


Add the following code to the class constructor in order to fill data source.

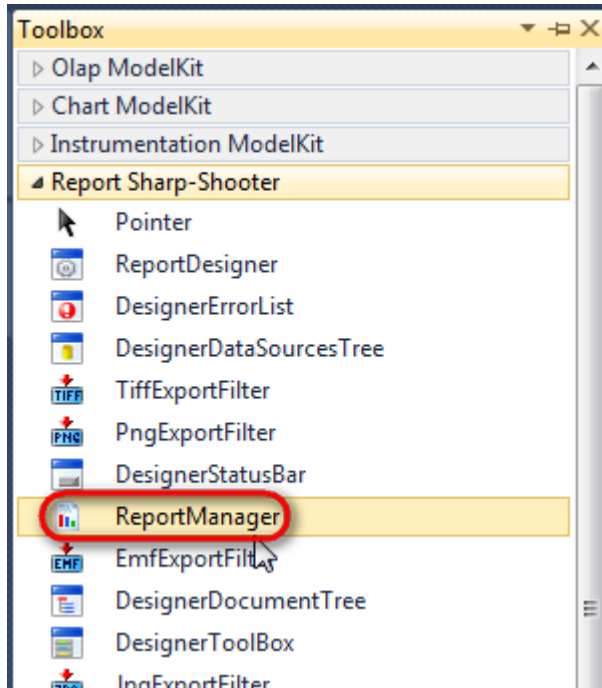
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["EmployeeName"] = "Nancy Davolio";
    row["HomePhone"] = "(206) 555-9857";
    row["Note"] = "Education includes a BA in psychology from Colorado State University in 1970. She also completed The Art of the Cold Call. Nancy is a member of Toastmasters International.";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["EmployeeName"] = "Andrew Fuller";
    row["HomePhone"] = "(206) 555-9482";
    row["Note"] = "Andrew received his BTS commercial in 1974 and a Ph.D. in international marketing from the University of Dallas in 1981. He is fluent in French and Italian and reads German. He joined the company as a sales representative, was promoted to sales manager in January 1992 and to vice president of sales in March 1993. Andrew is a member of the Sales Management Roundtable, the Seattle Chamber of Commerce, and the Pacific Rim Importers Association.";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["EmployeeName"] = "Anne Dodsworth";
    row["HomePhone"] = "(71) 555-4444";
    row["Note"] = "Anne has a BA degree in English from St. Lawrence College. She is fluent in French and German.";
    dataTable1.Rows.Add(row);
}
```

Step 7

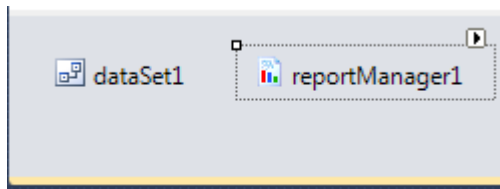
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

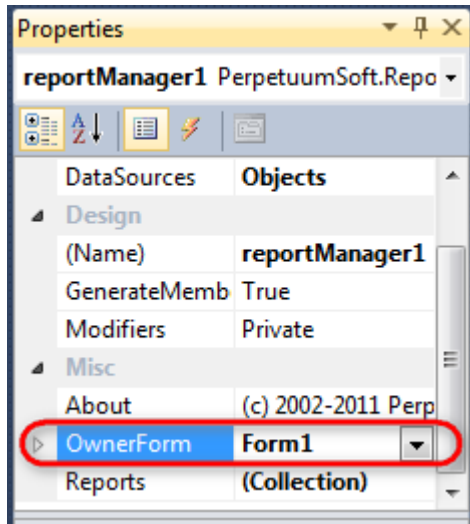


The component is available in the lower part of the window.



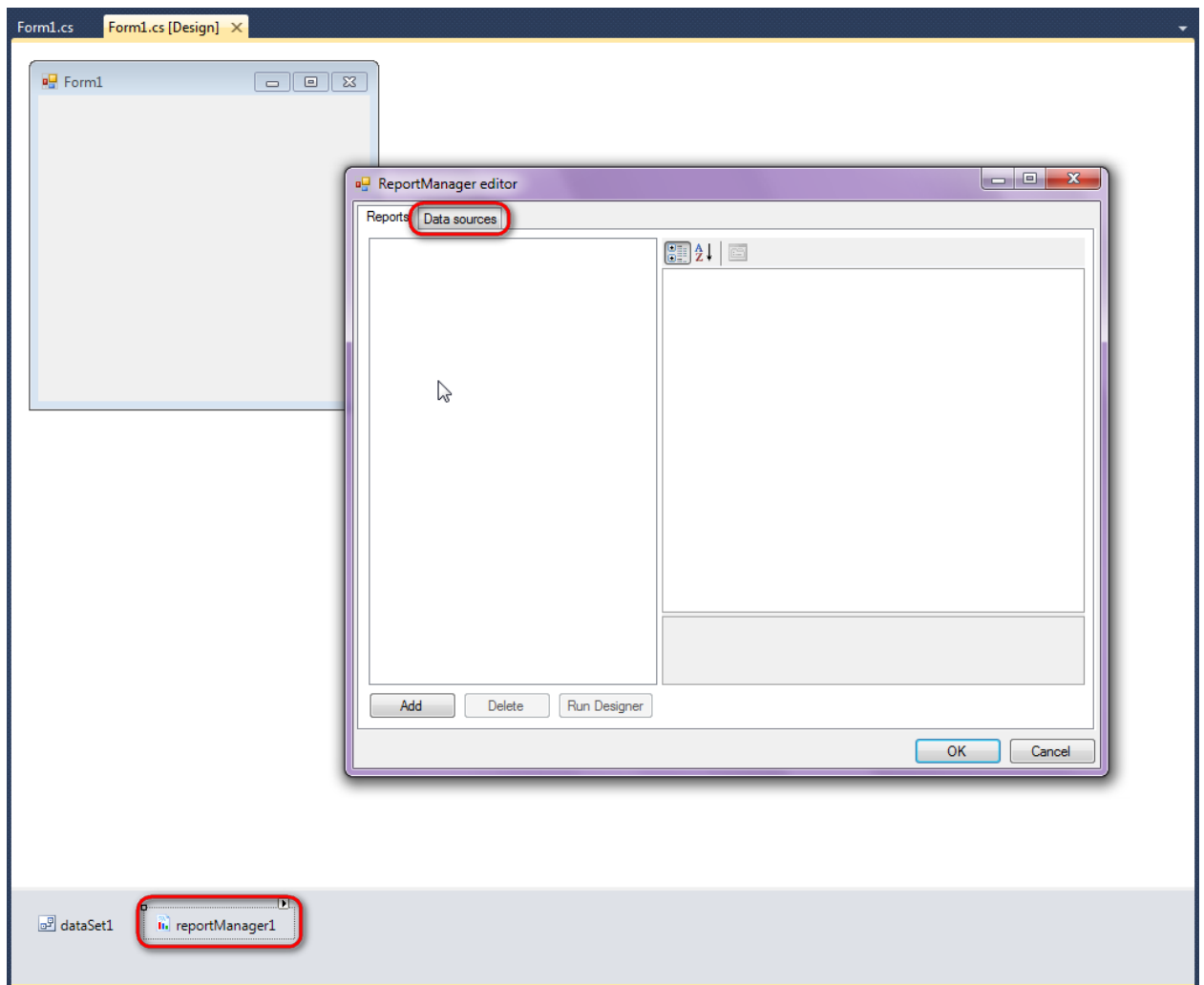
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

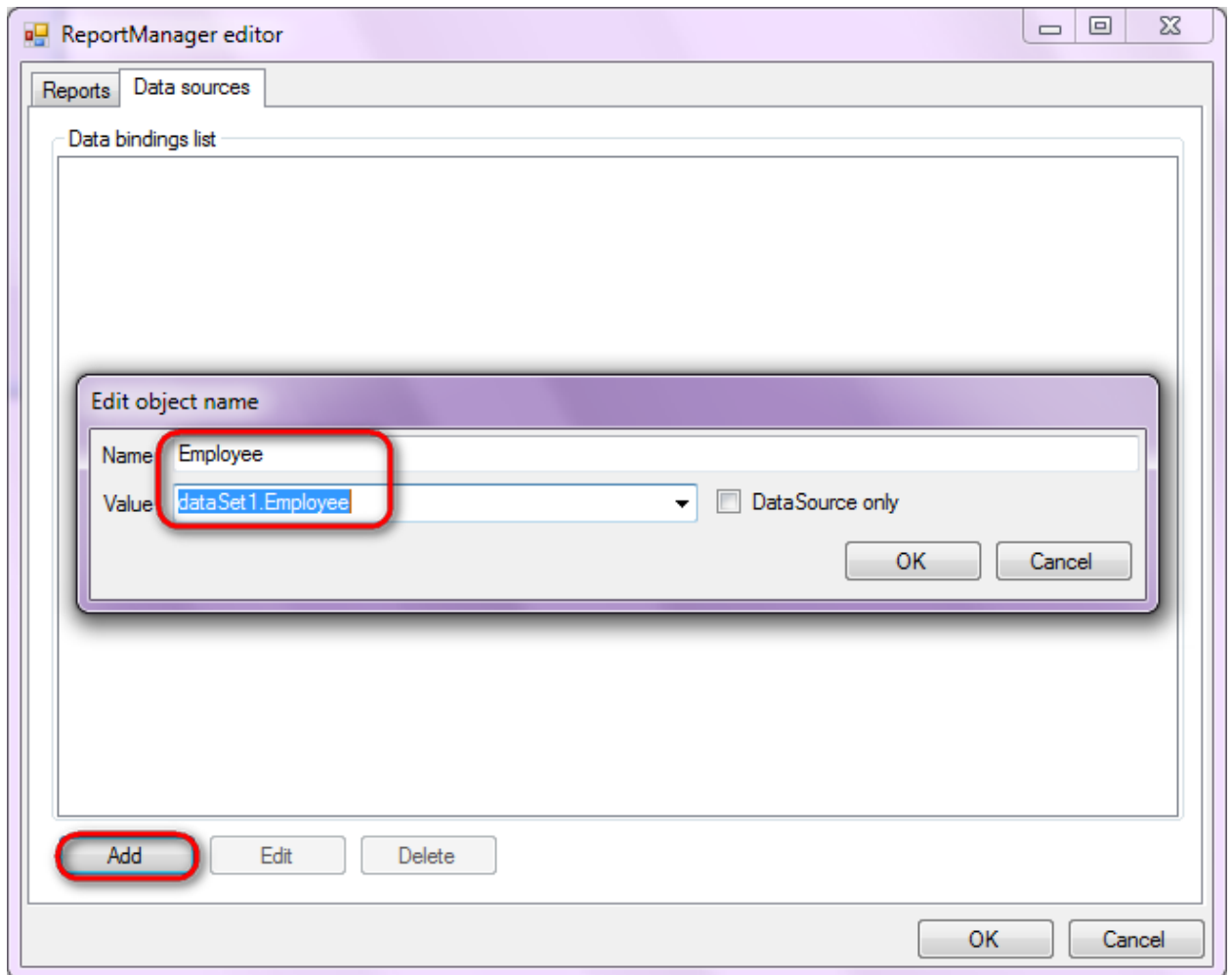


Step 9

Double click on ReportManager to open ReportManager editor.

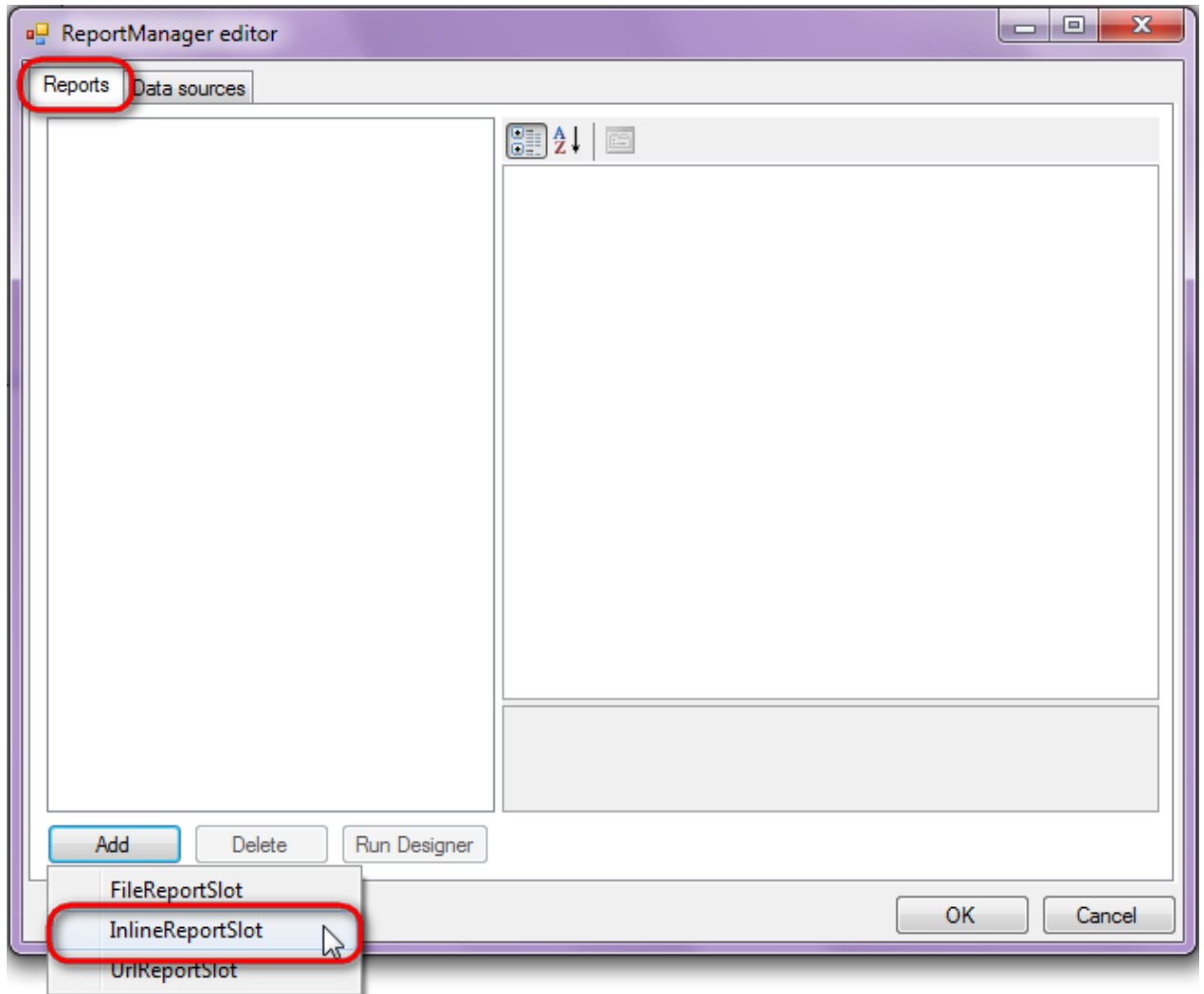


Go to "Data sources" tab, click "Add", set data source name - "Employee", select data source value - "dataSet1.Employee".



Step 10

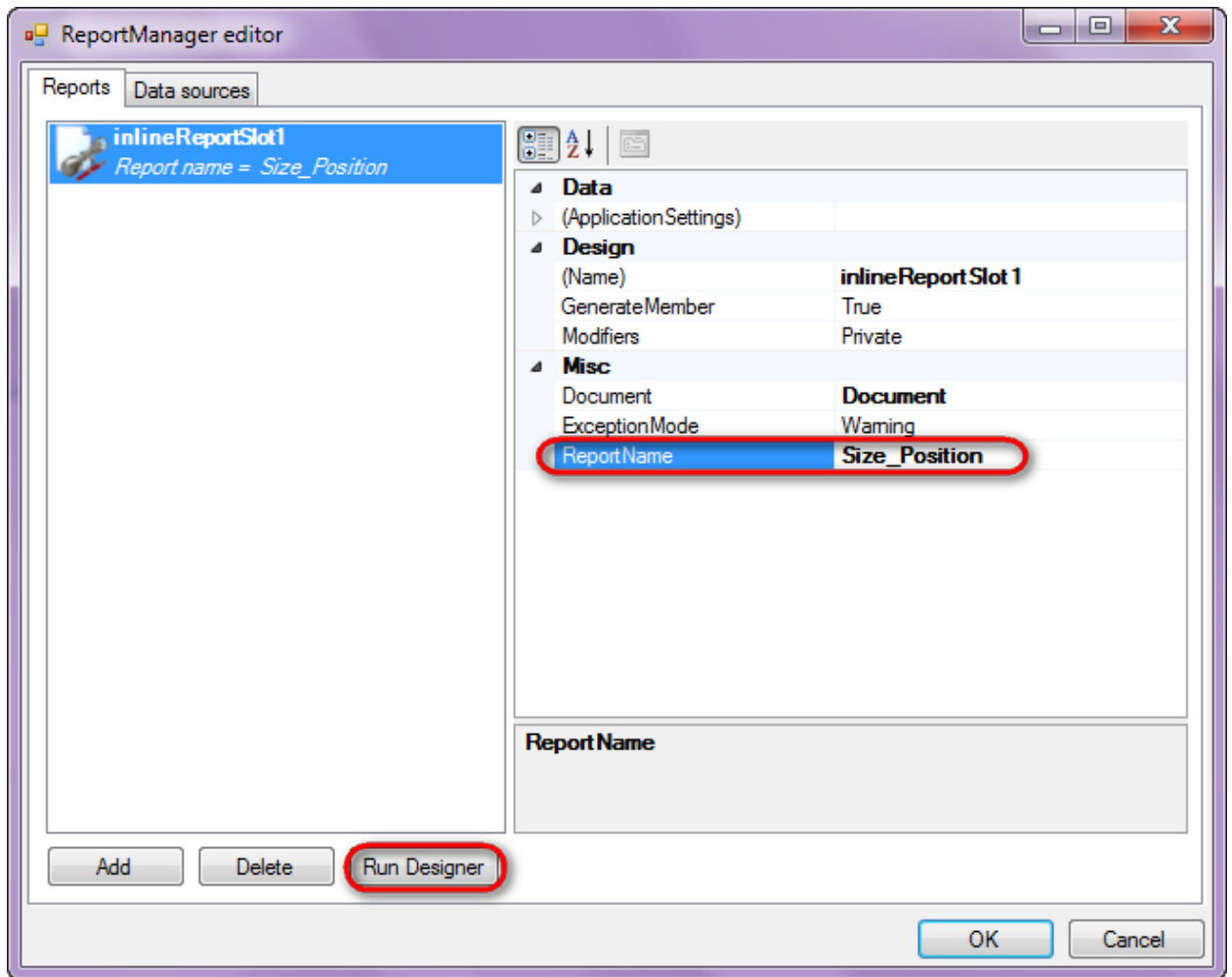
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

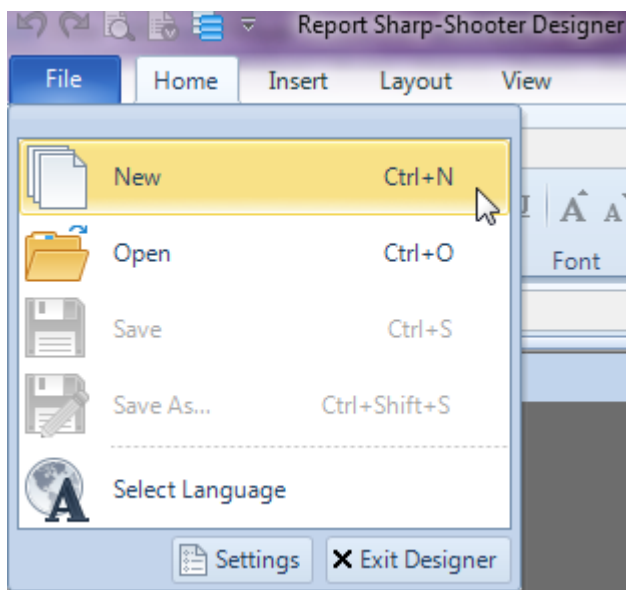
Set name of the report in the property ReportName – "Size_Position".

Click "Run Designer" in order to open template editor - Report Designer.

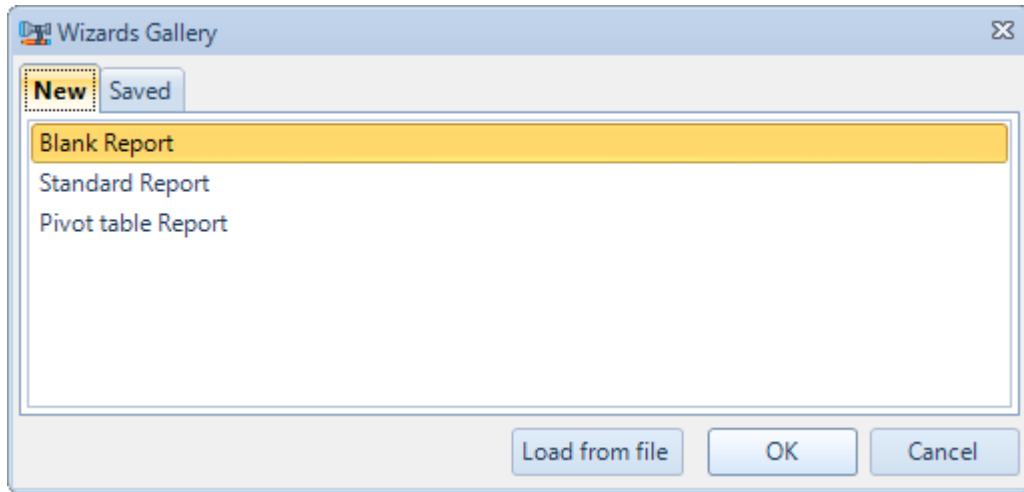


Step 12

Create new empty template – select File\New from the main menu.

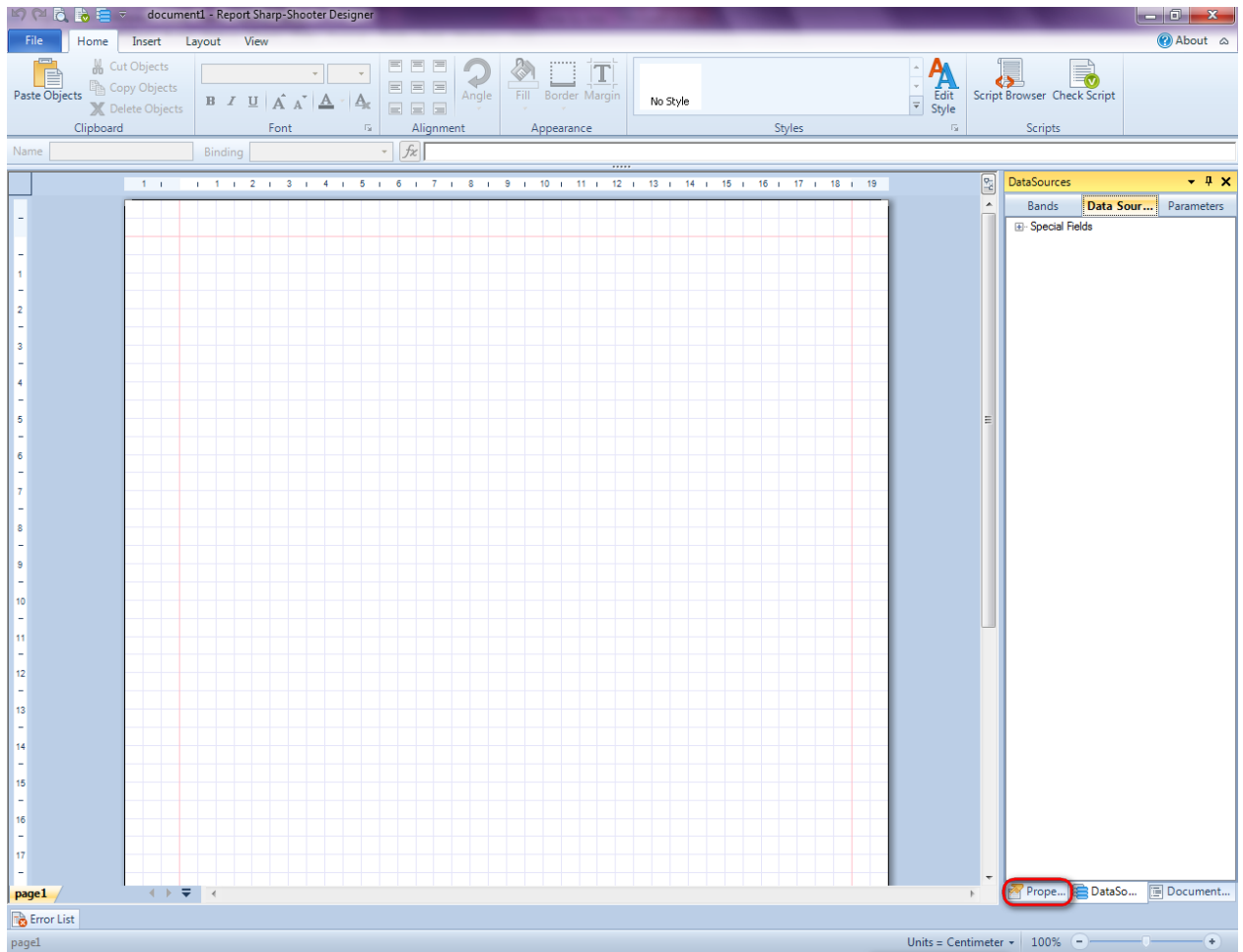


Select "Blank Report" in the Wizards Gallery and click "OK".

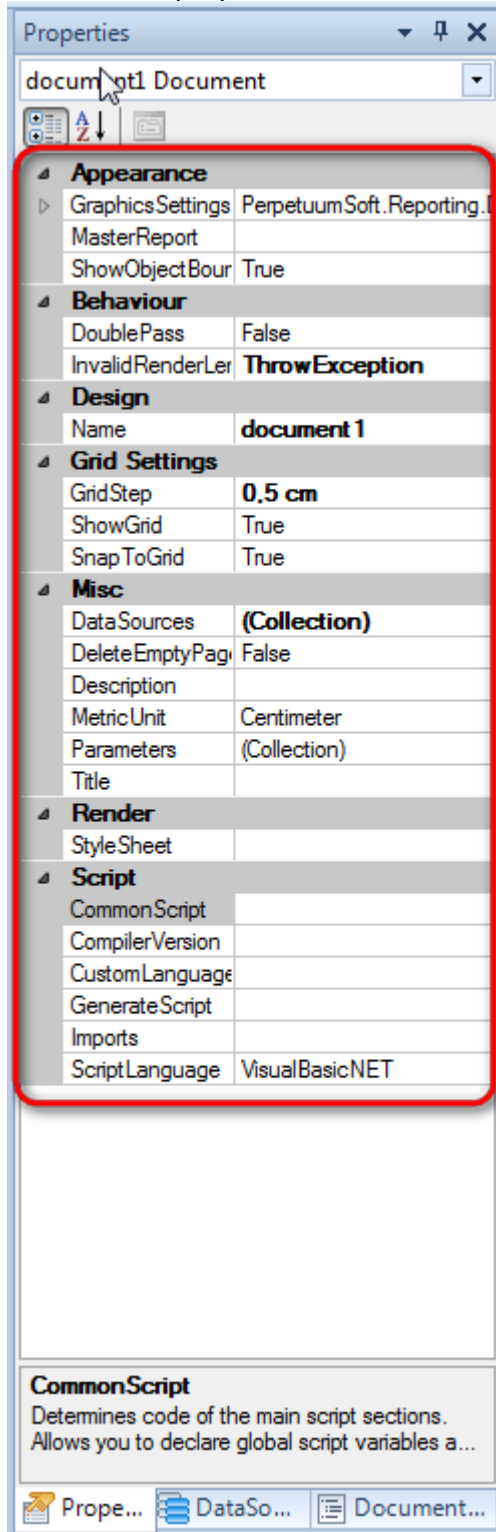


Step 13

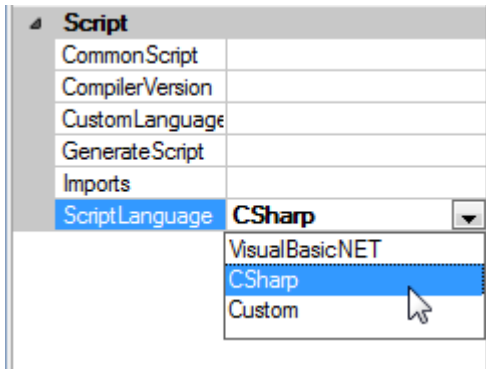
Click the "Properties" tab of the tool window in the right part of the designer.



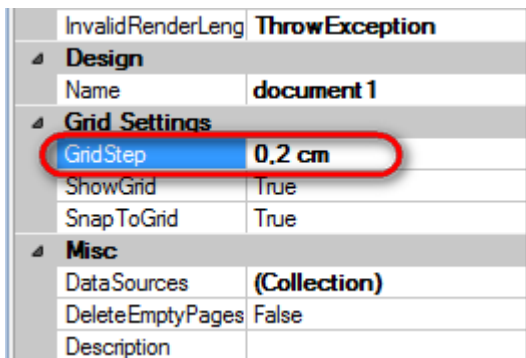
You will see properties of the edited template on the “Properties” tab



Set property ScriptLanguage = CSharp.

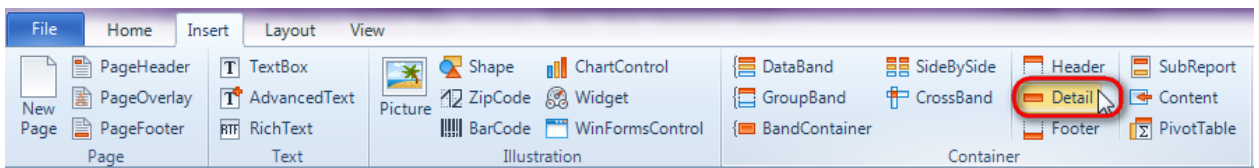


Set property **GridStep = 0,2 cm.**



Step 14

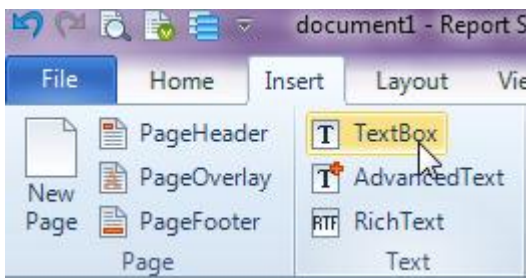
Press "Detail" button on the Insert tab in the group Container.



Click on the template area to add a band.

Step 15

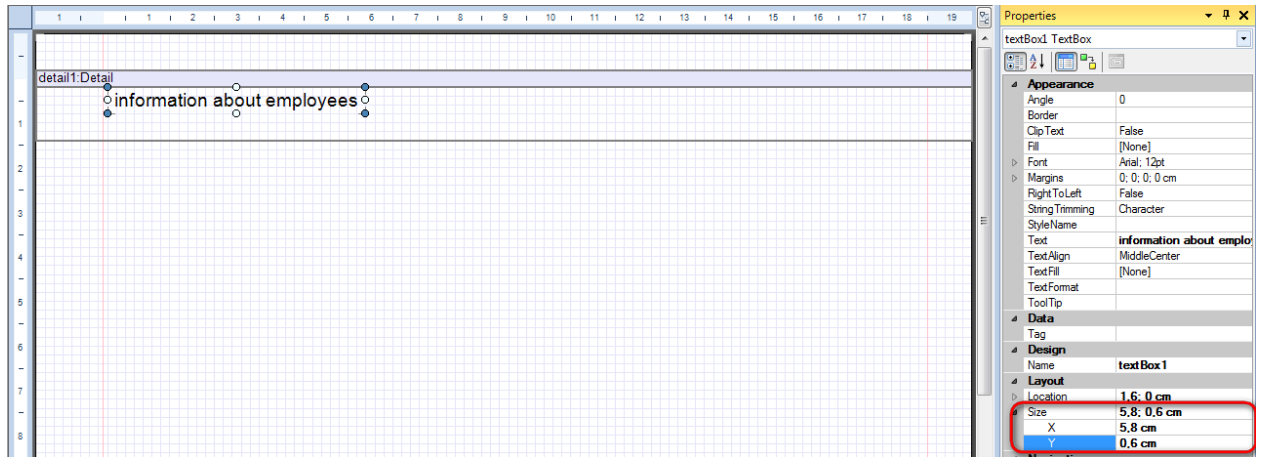
Press button "TextBox" on the Insert tab in the group Text.



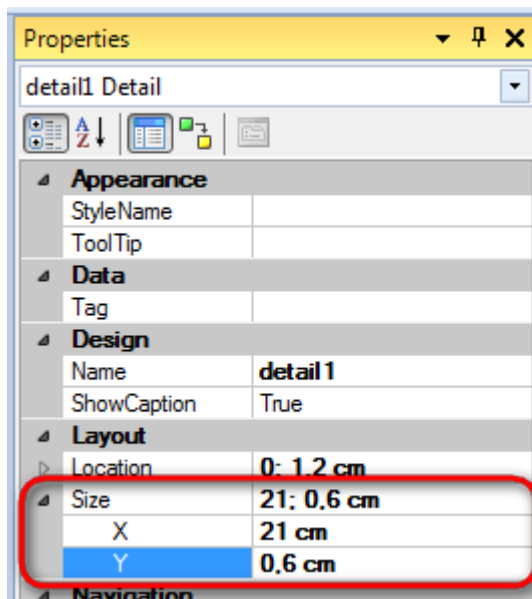
Click on the Detail band area to add TextBox element inside Detail.

Double click on the TextBox area to open Text editor. Enter the following text: "information about employees".

Change size of the TextBox element by dragging its corner. Set its size to "5,8; 0,6 cm". Changing of the size is visible on the property grid.

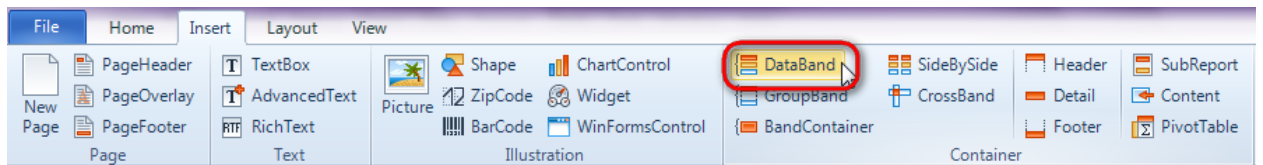


Select Detail band. Set Size property to "21; 0,6 cm".



Step 16

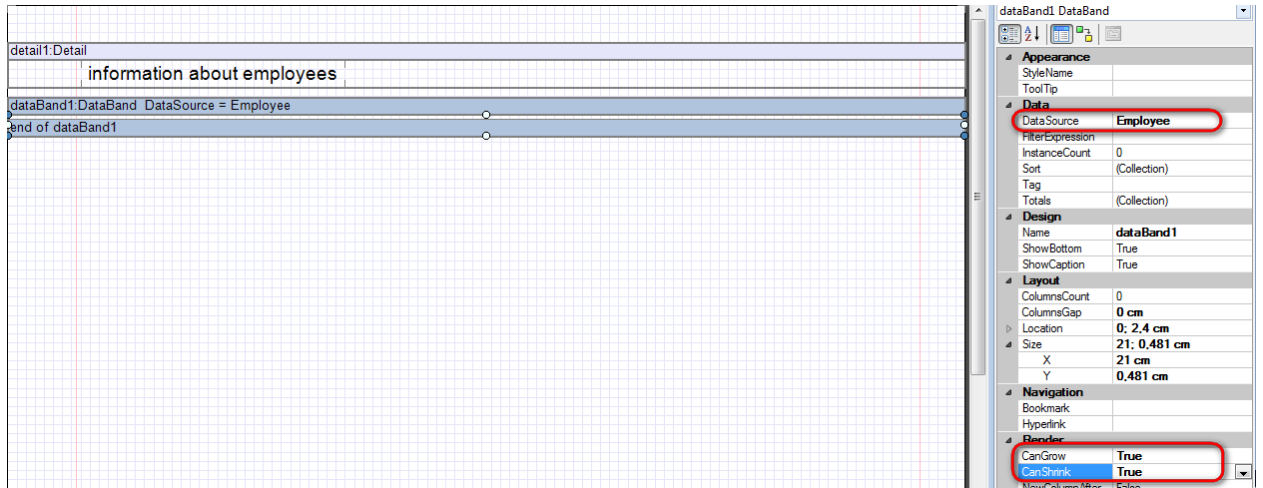
Press "DataBand" button on the Insert tab in the group Container.



Click on the template area to add DataBand to the template.

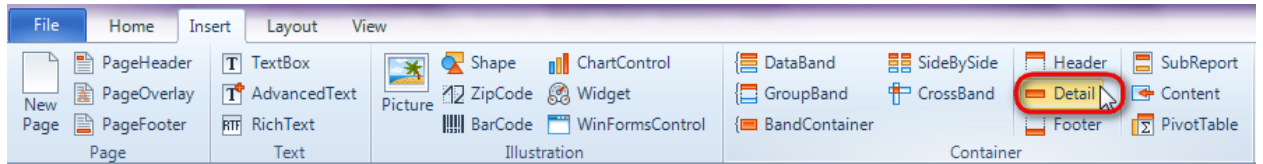
Set data source in the property DataSource = Employee.

Set CanGrow and CanShrink properties to "True".



Step 17

Press "Detail" button on the Insert tab in the group Container.

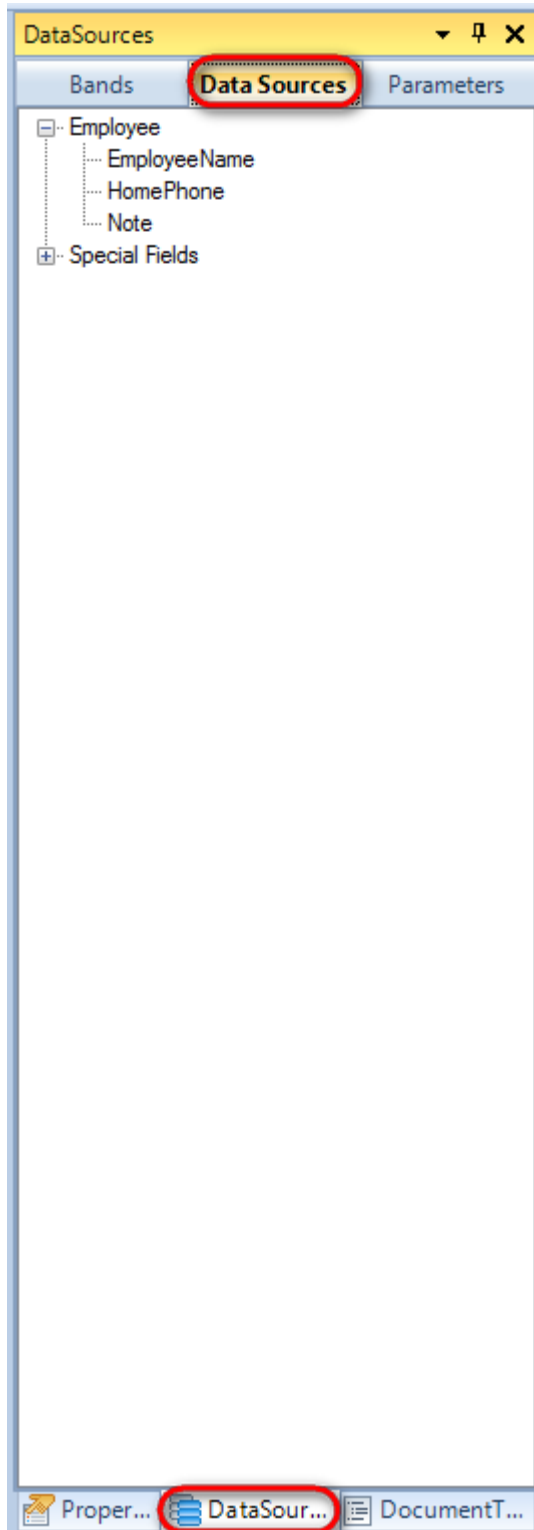


Click on the DataBand area to add Detail band inside DataBand.

Set CanGrow and CanShrink properties of the Detail band to "True".

Step 18

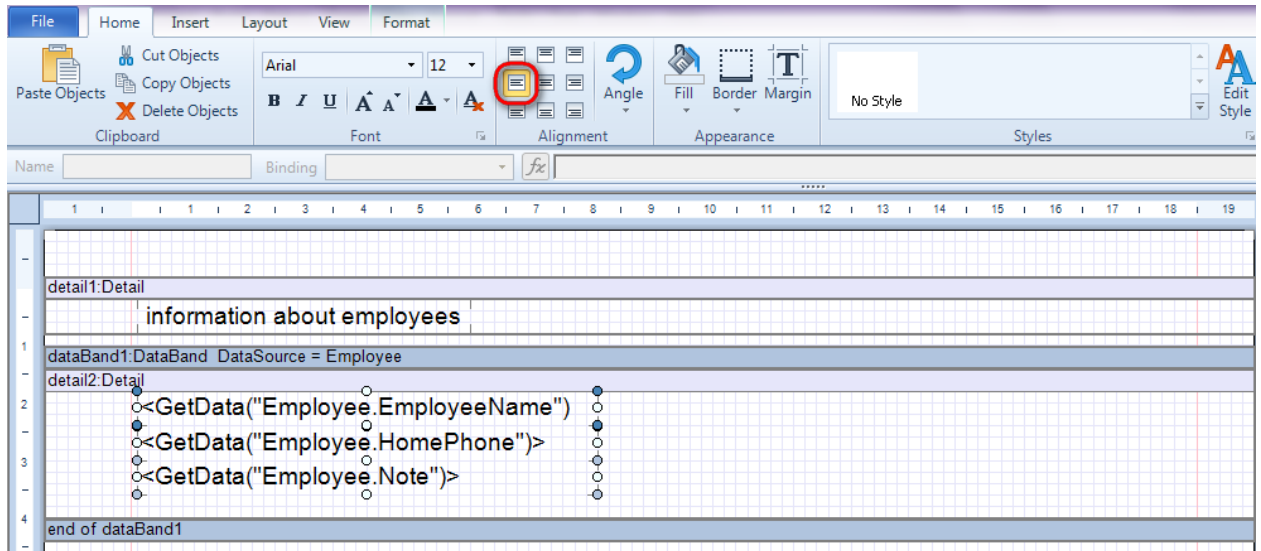
Go to "DataSources" tab.



Drag and drop "EmployeeName", "HomePhone", "Note" fields from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.

Change size and position of the DataBand and Detail bands.


Select all TextBox elements, click "Left Alignment" button on the Home tab in the group Alignment. Set Size property of TextBoxes to "8; 0,6 cm".

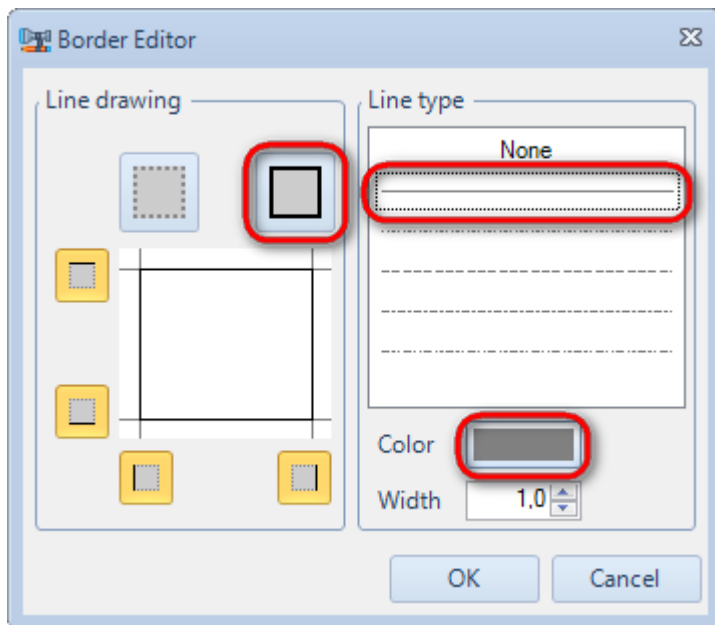


Step 19

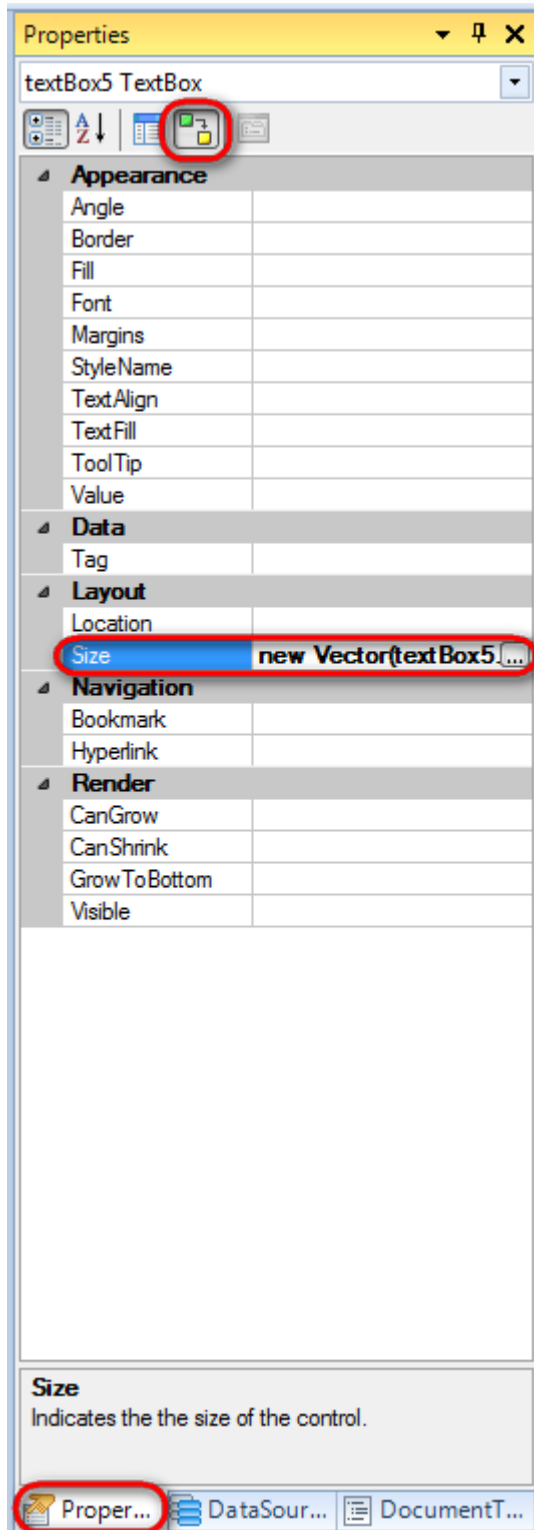
Select TextBox with dataBand1["Note"]. Set the CanShrink and CanGrow properties to "True" in property grid.

Step 20

Add one more TextBox to the detail2 band. Change its size and position so that the other three TextBox elements were inside the added element. Select the Border property, click the  button to open Border Editor. Set border for the element.



Select the Size property in Binding ToolBar and set the value to "new Vector(textBox5.Size.X, Unit.Convert(1.2,Unit.Centimeter, Unit.InternalUnit) + (double)textBox4.RenderHeight)".



Step 21

Save template, close Report Designer.

Step 22

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

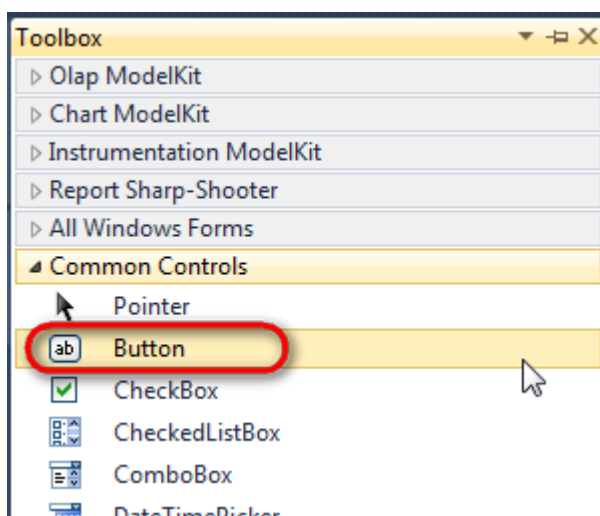
```
public Form1 ()  
{  
    InitializeComponent();  
    DataRow row = dataTable1.NewRow();
```



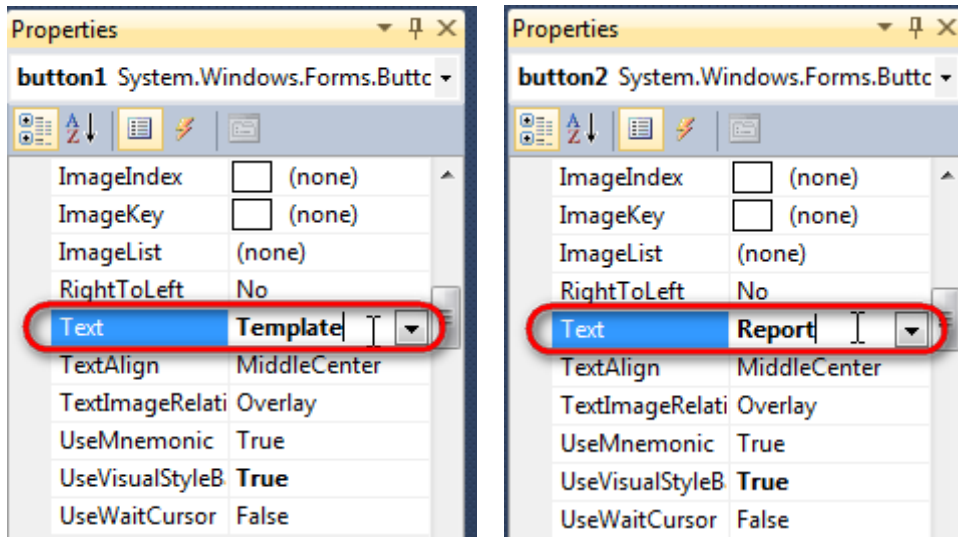
```
row["EmployeeName"] = "Nancy Davolio";
row["HomePhone"] = "(206)555-9857";
row["Note"] = "Education includes a BA in psychology from
Colorado State University in 1970. She also completed The Art of the Cold
Call. Nancy is a member of Toastmasters International.";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["EmployeeName"] = "Andrew Fuller";
row["HomePhone"] = "(206) 555-9482";
row["Note"] = "Andrew received his BTS commercial in 1974 and a
Ph.D. in international marketing from the University of Dallas in 1981. He
is fluent in French and Italian and reads German. He joined the company as a
sales representative, was promoted to sales manager in January 1992 and to
vice president of sales in March 1993. Andrew is a member of the Sales
Management Roundtable, the Seattle Chamber of Commerce, and the Pacific Rim
Importers Association.";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["EmployeeName"] = "Anne Dodsworth";
row["HomePhone"] = "(71) 555-4444";
row["Note"] = "Anne has a BA degree in English from St. Lawrence
College. She is fluent in French and German.";
dataTable1.Rows.Add(row);
inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 23

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



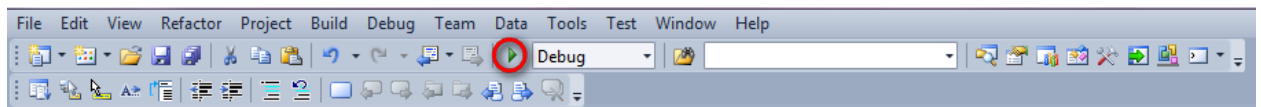
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

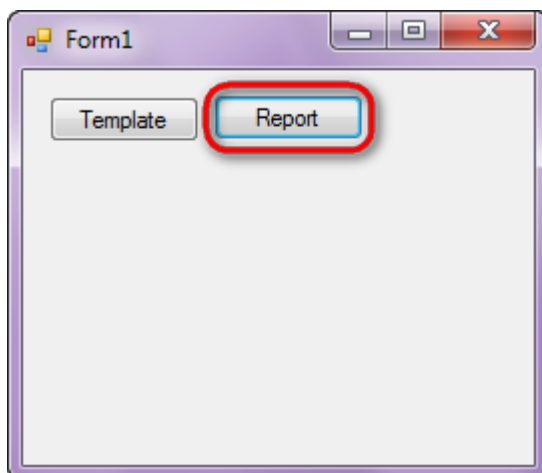
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 24

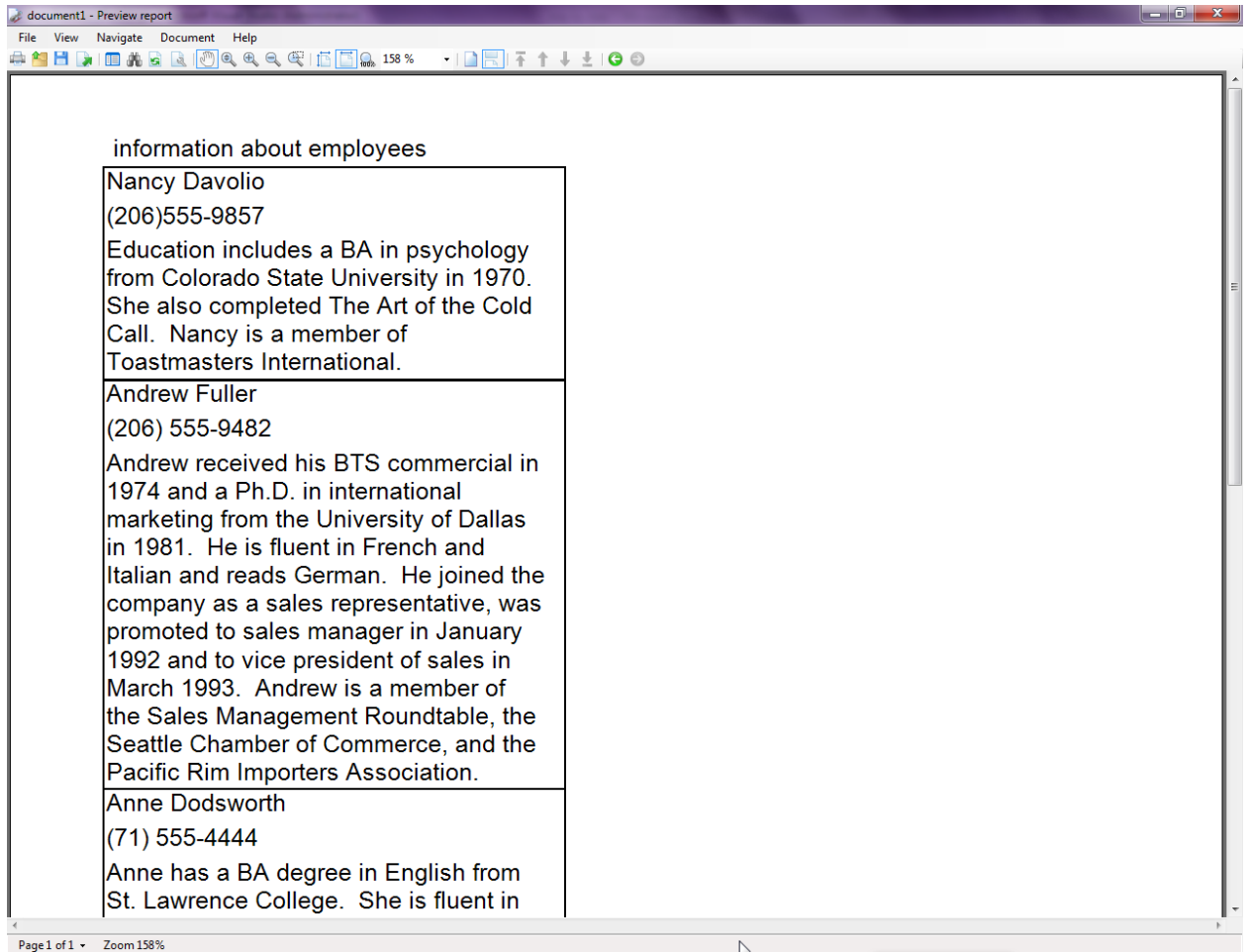
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



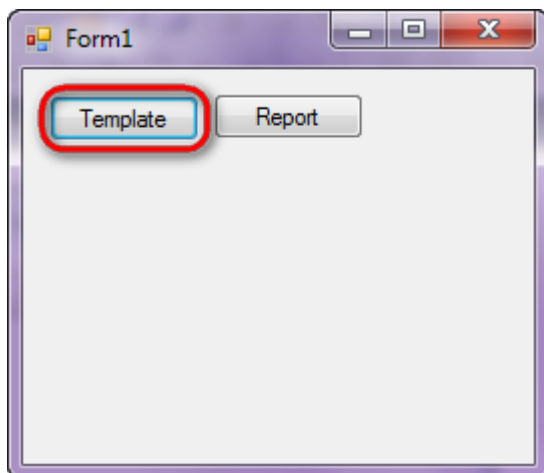
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



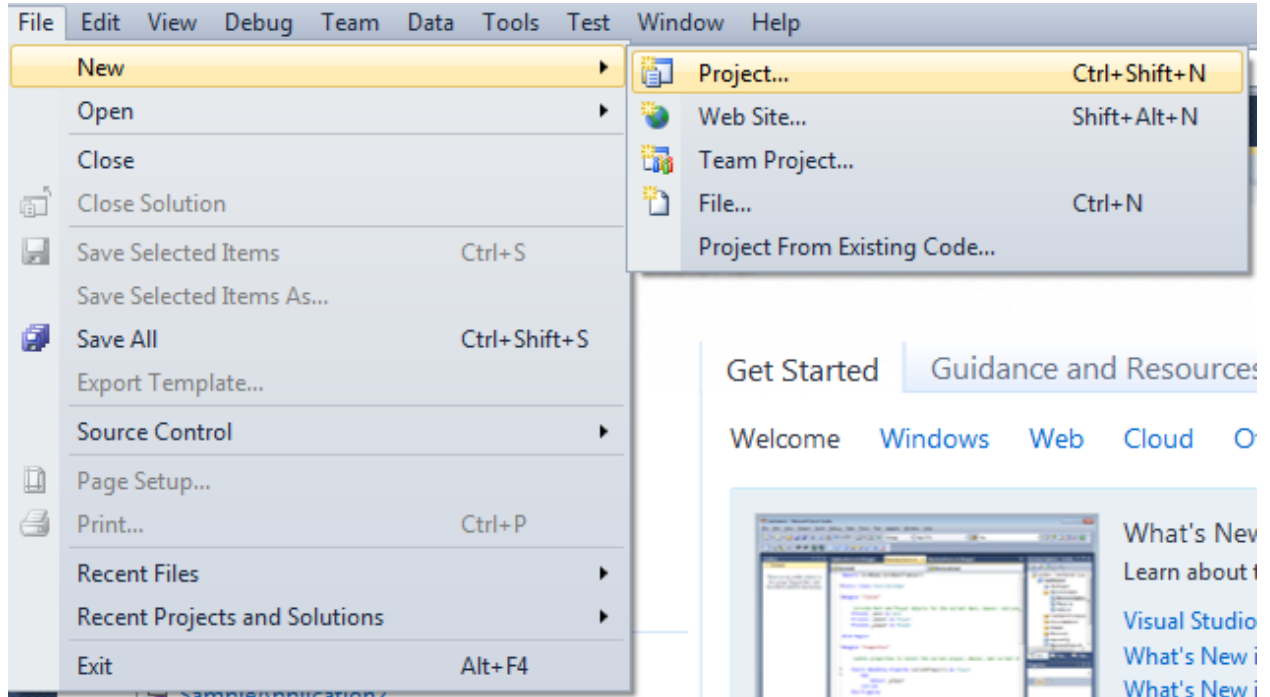


Report without Bands

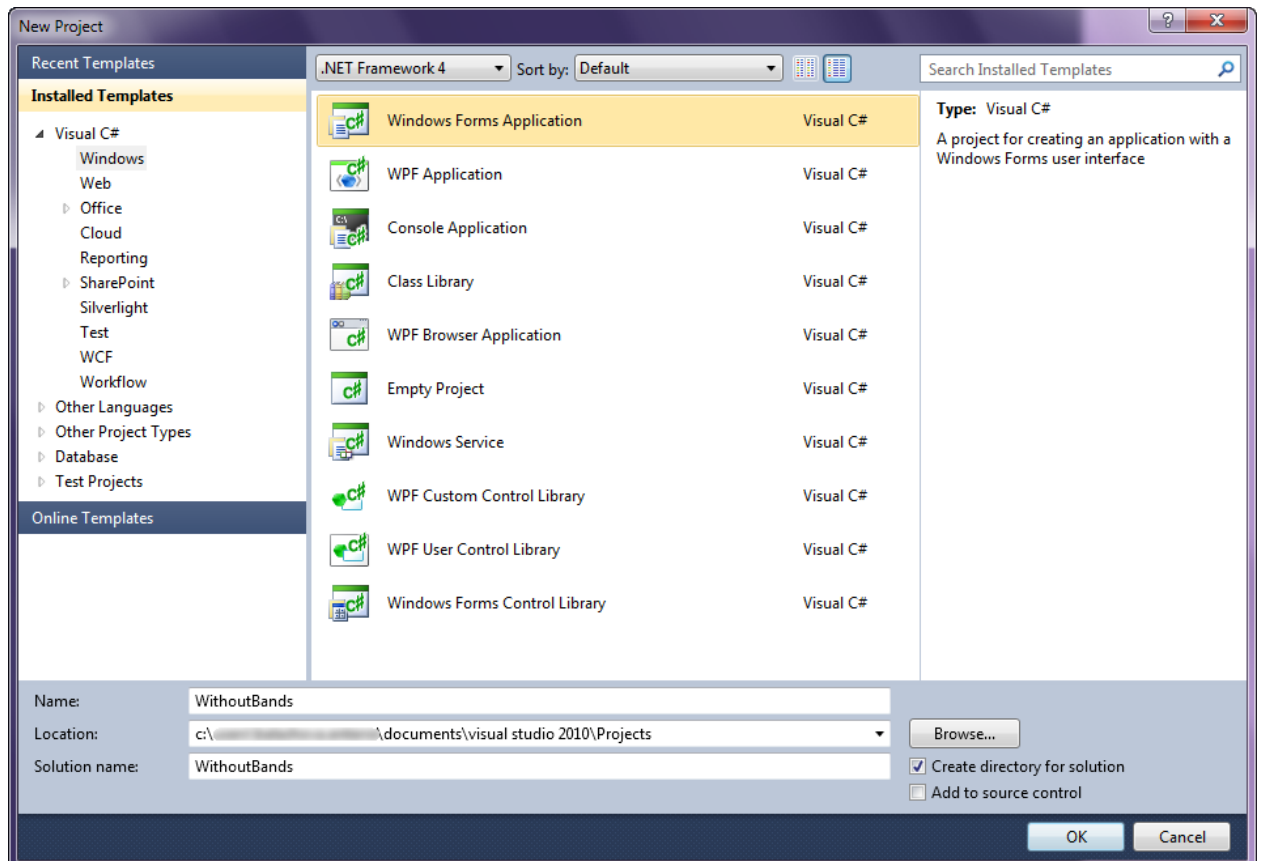
Template of a card storing information on the customer. The card contains data on company name, address, phone, and name of contact person.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

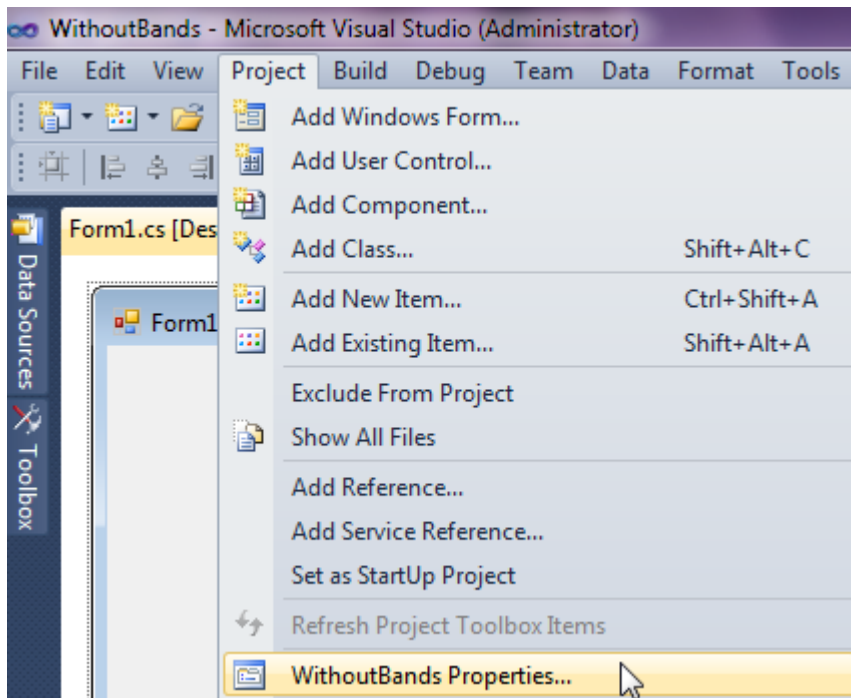


Select Windows Forms Application, set project name – “WithoutBands”, set directory to save the project to.

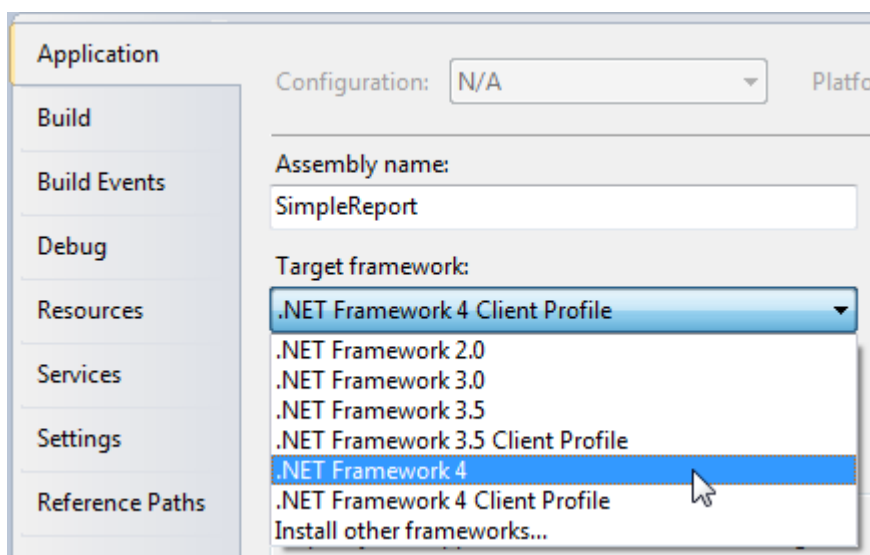


Step 2

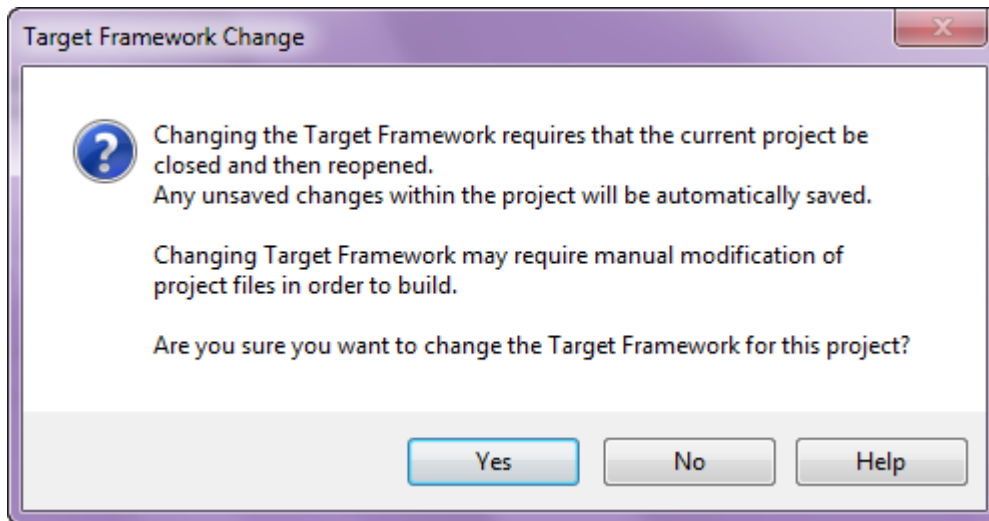
Change the project properties. Select the Project\WithoutBands Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

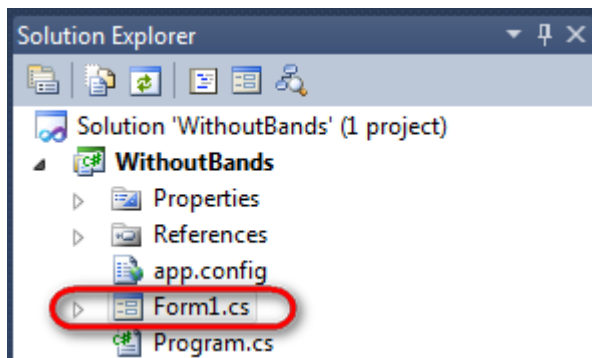


In the opened window press the "Yes" button.

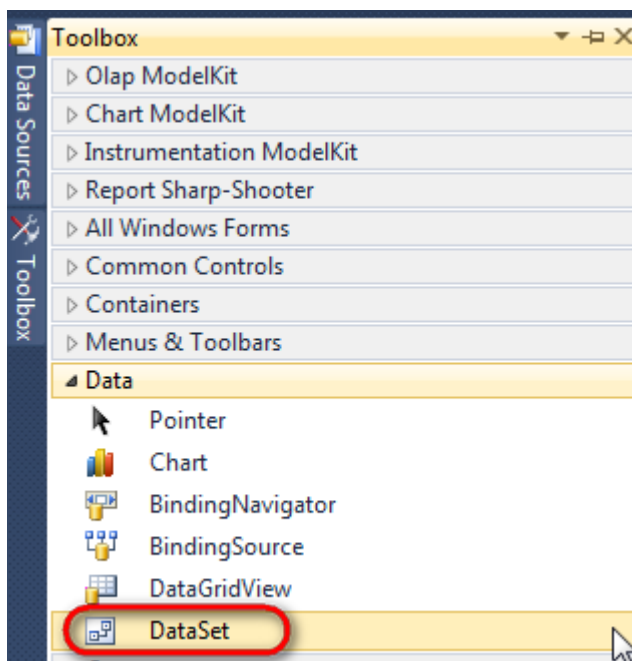


Step 3

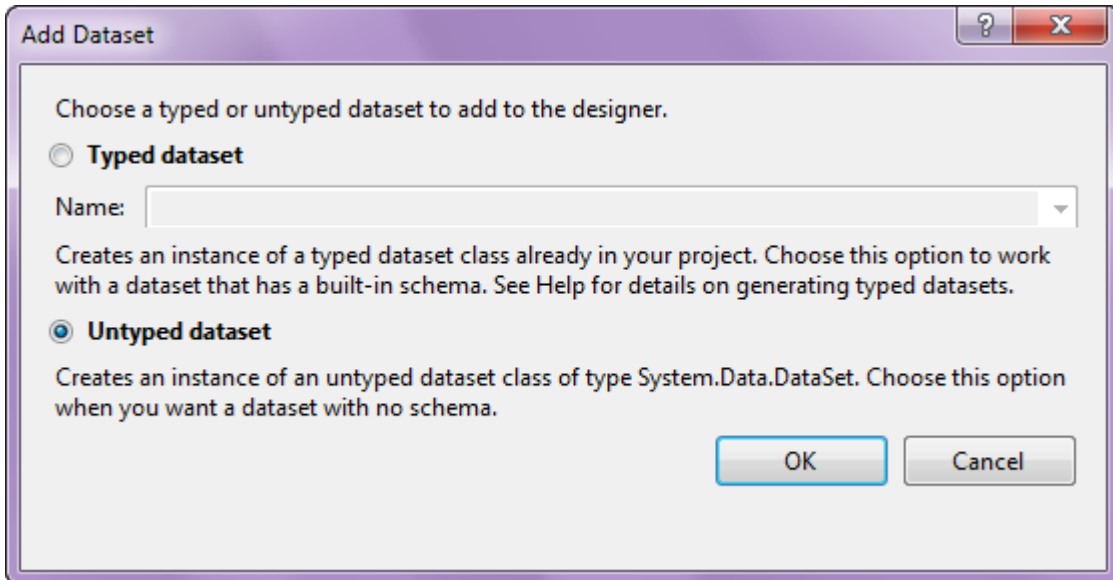
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



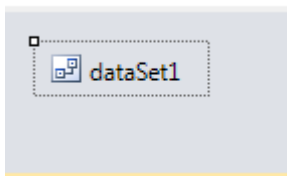
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK"

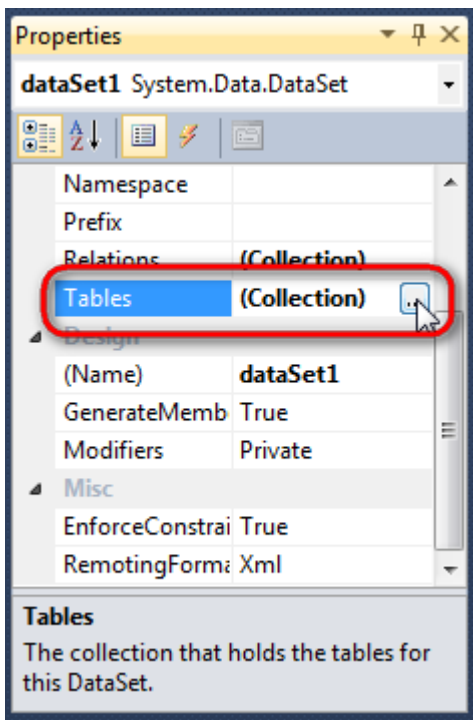


The component is available in the lower part of the window.

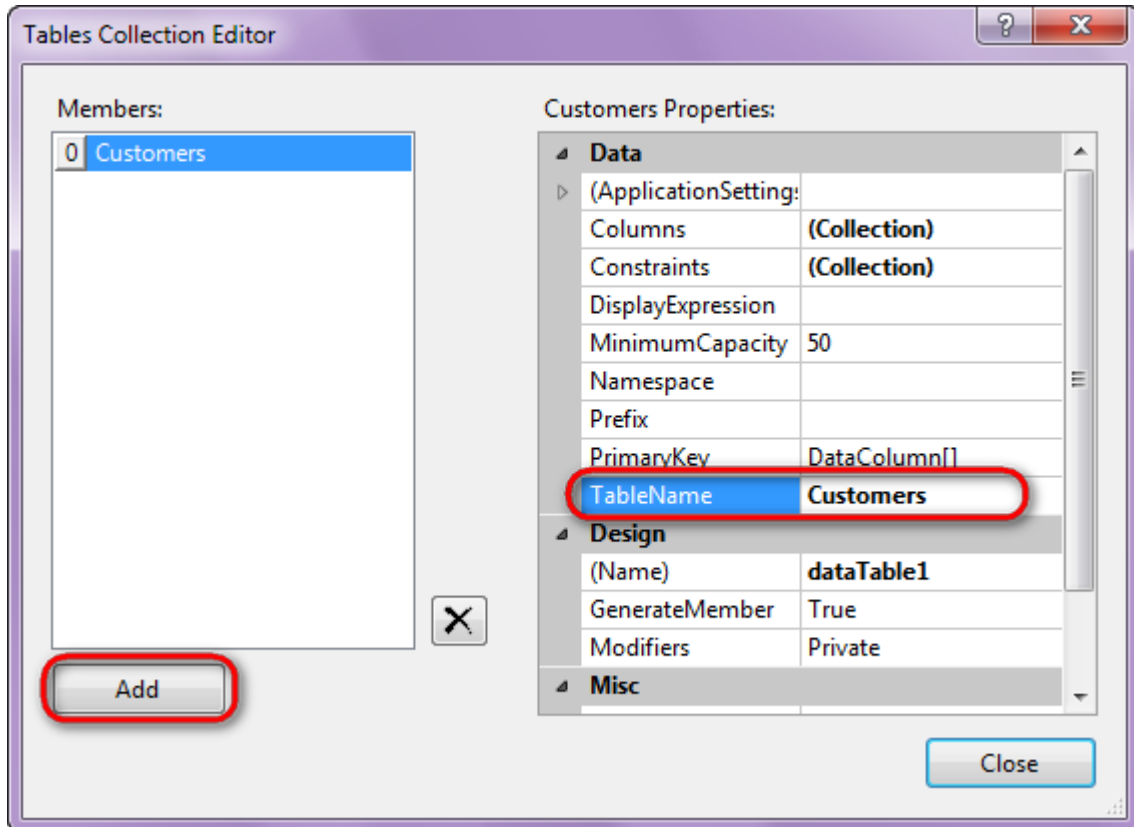


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

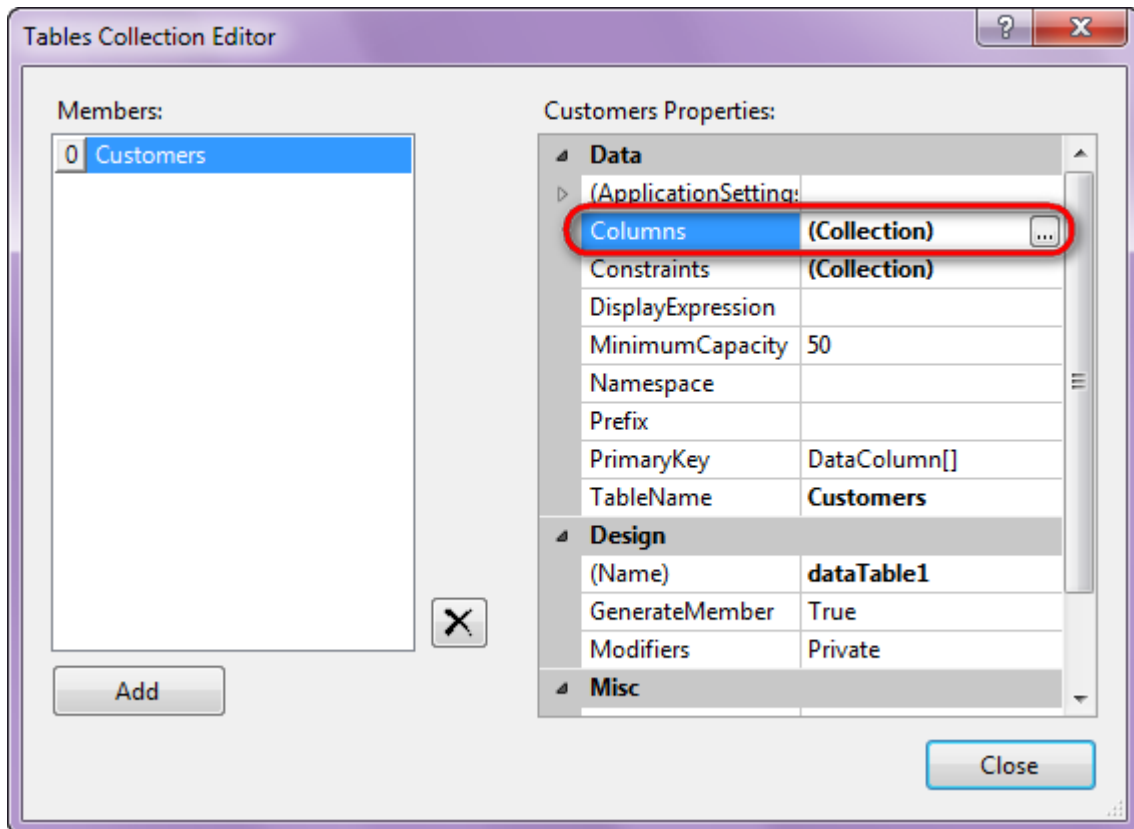


Click "Add" in order to add table. Set property TableName = Customers.

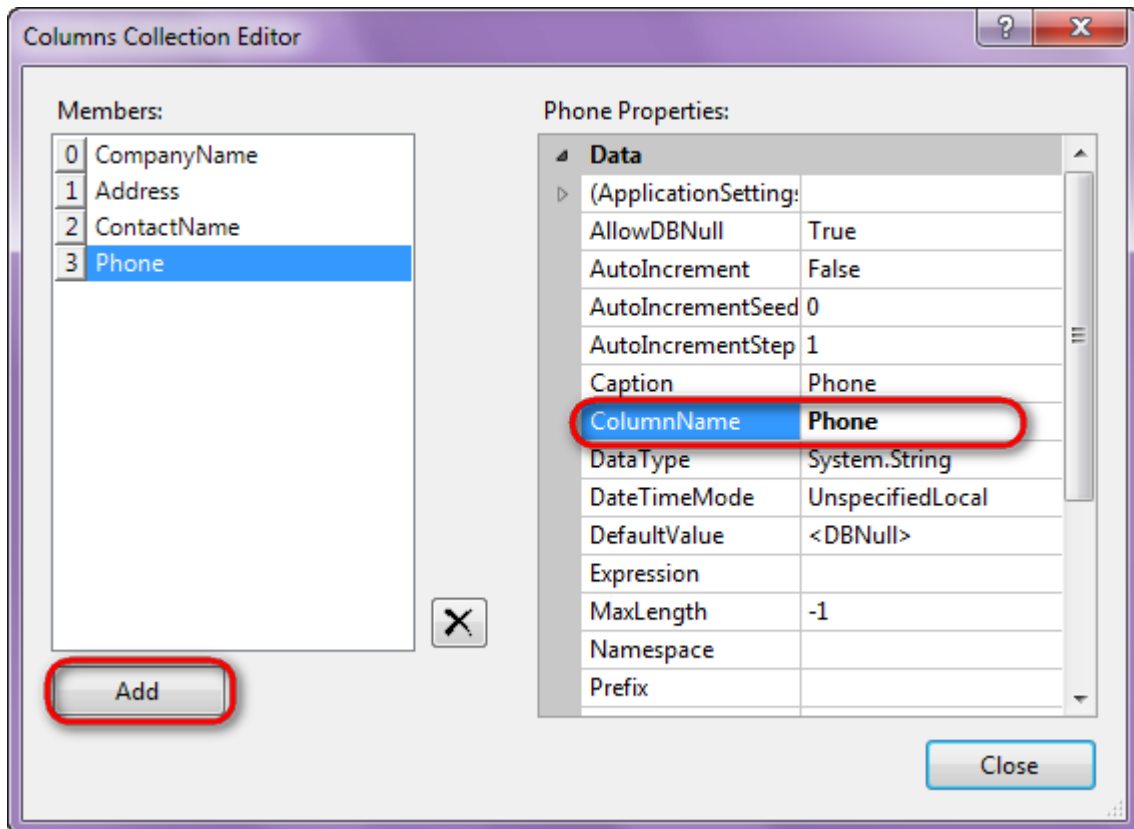


Step 5

Select Columns property, click button  in order to open property editor.

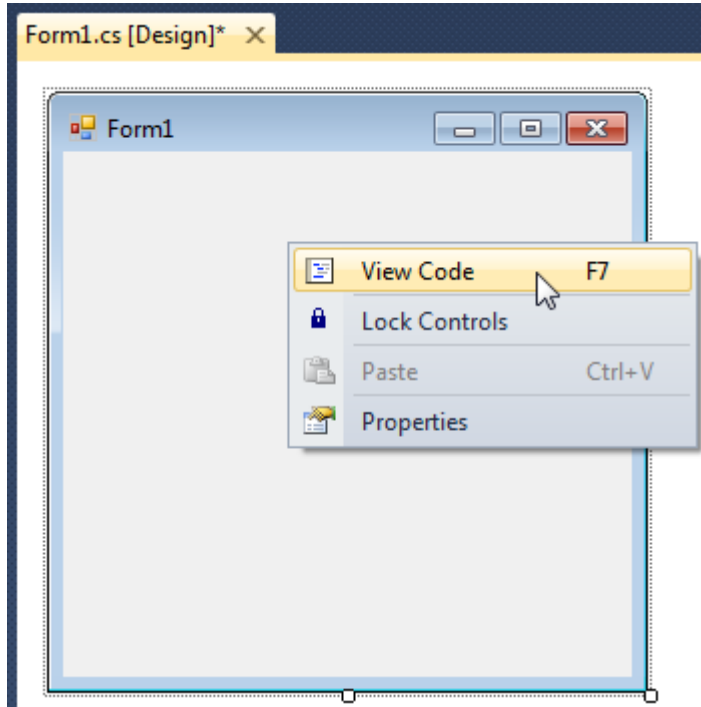


Click "Add" to add a new column. Add four columns. Set ColumnName property to "CompanyName", "Address", "ContactName", "Phone" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu to view code.



Add the following code to the class constructor in order to fill data source.

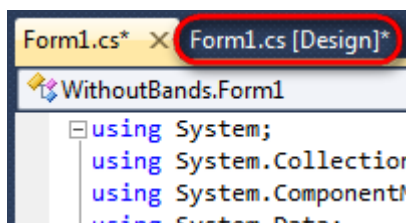
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
}
```



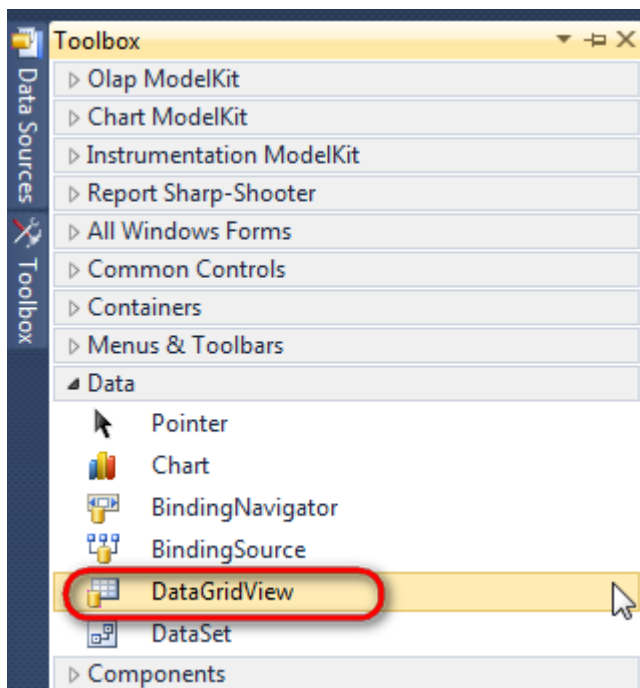
```
row["Address"] = "Obere Str. 57";  
row["ContactName"] = "Maria Anders";  
row["Phone"] = "030-0074321";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["Address"] = "Avda. de la Constitución 2222";  
row["ContactName"] = "Ana Trujillo";  
row["Phone"] = "(5) 555-4729";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["Address"] = "Kirchgasse 6";  
row["ContactName"] = "Roland Mendel";  
row["Phone"] = "7675-3425";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["Address"] = "Luisenstr. 48";  
row["ContactName"] = "Karin Josephs";  
row["Phone"] = "0251-031259";  
dataTable1.Rows.Add(row);  
}
```

Step 7


Get back to the application form by clicking the "Form1.cs[Design]" tab.

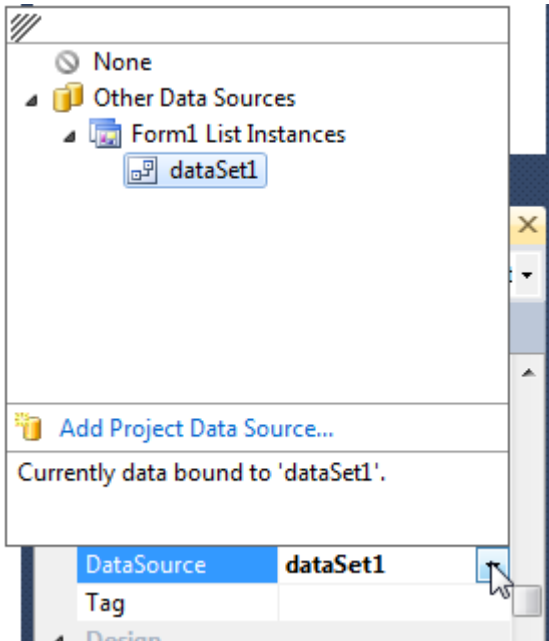


Click on the "DataGridView" on the Toolbox and place this component onto the form.

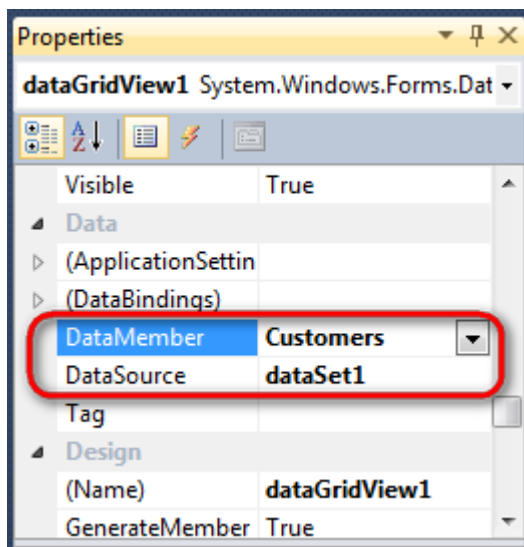


Step 8

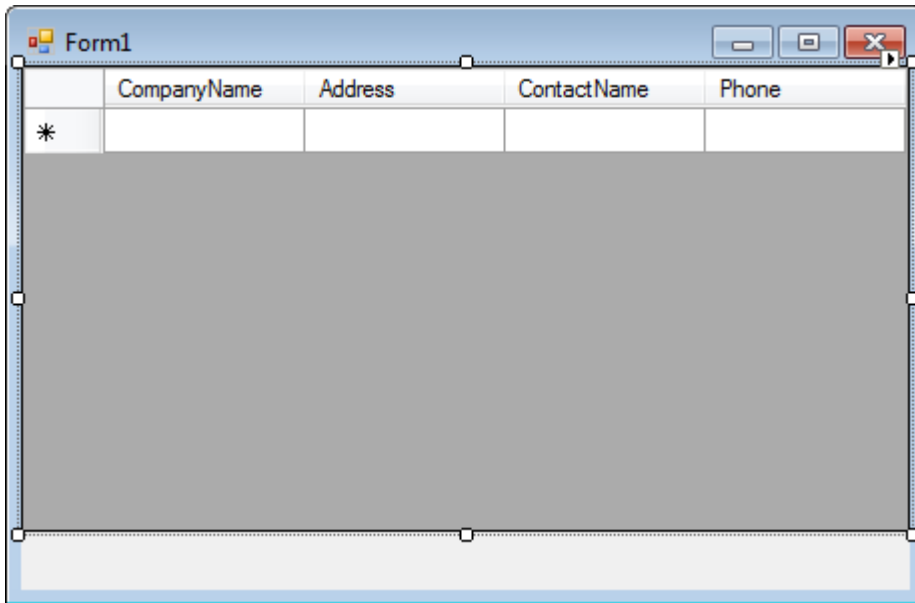
Select DataGridView element. On the property grid, select DataSource property, click button , select "dataSet1" as a data source.



Select "Customers" table in the DataMember property.

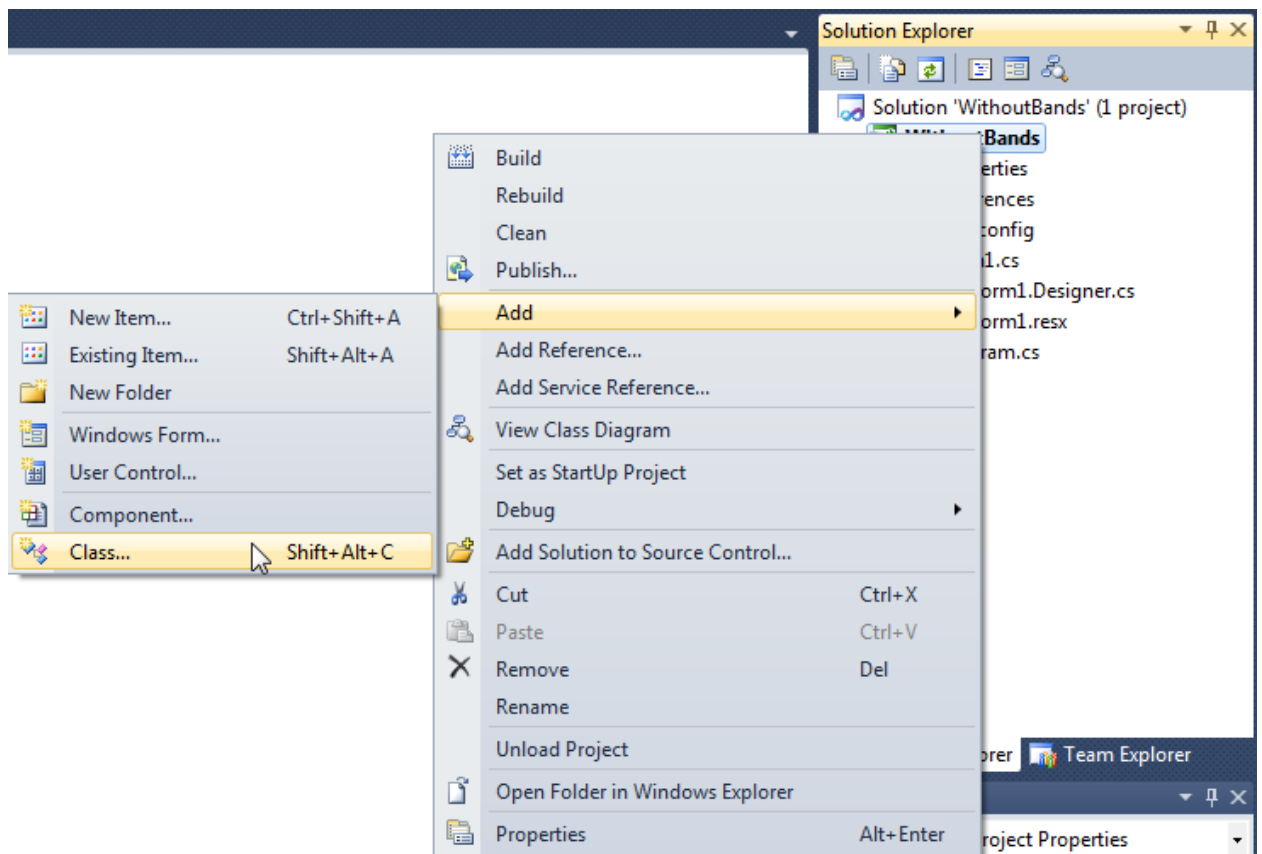


Change size of the form and DataGridView element so that all the columns are visible on the form.

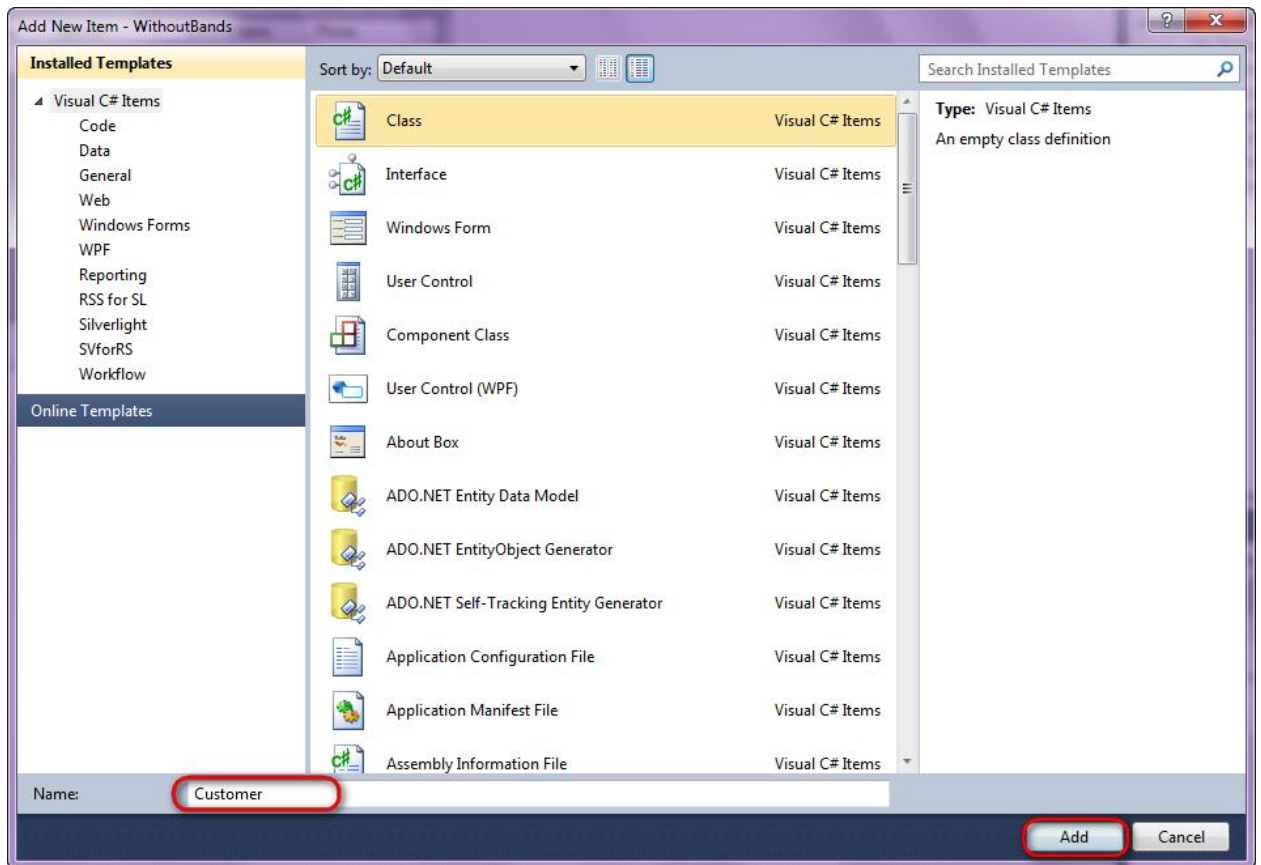


Step 9

Create a new class. Right click on "WithoutBands" in the Solution Explorer, select Add\Class... in the context menu.



Set class name – "Customer", click "Add" button.



Class code:

```
public class Customer
{
    public Customer()
    {
    }

    private string company = String.Empty;
    public string CompanyName
    {
        get
        {
            return company;
        }
        set
        {
            company = value;
        }
    }

    private string address = String.Empty;
    public string Address
    {
        get
        {
            return address;
        }
        set
        {
            address = value;
        }
    }

    private string phone = String.Empty;
    public string Phone

```



```
{
    get
    {
        return phone;
    }
    set
    {
        phone = value;
    }
}
private string contact = String.Empty;
public string ContactName
{
    get
    {
        return contact;
    }
    set
    {
        contact = value;
    }
}
}
```

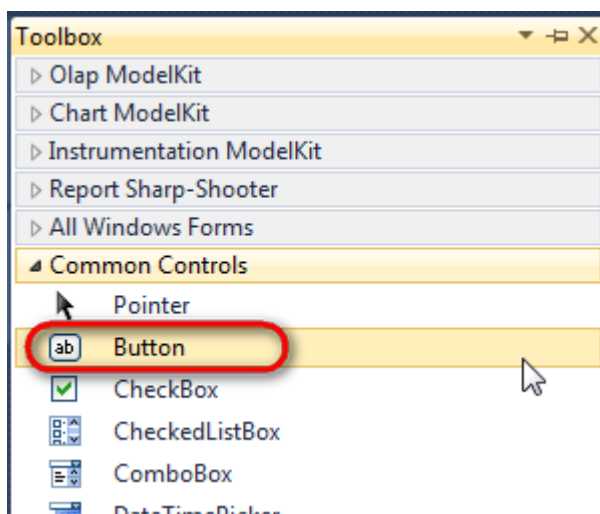
Step 10

Go to the "Form1.cs" tab. Create instance of the Customer class. Write function to fill data source with the values from the DataGridView table.

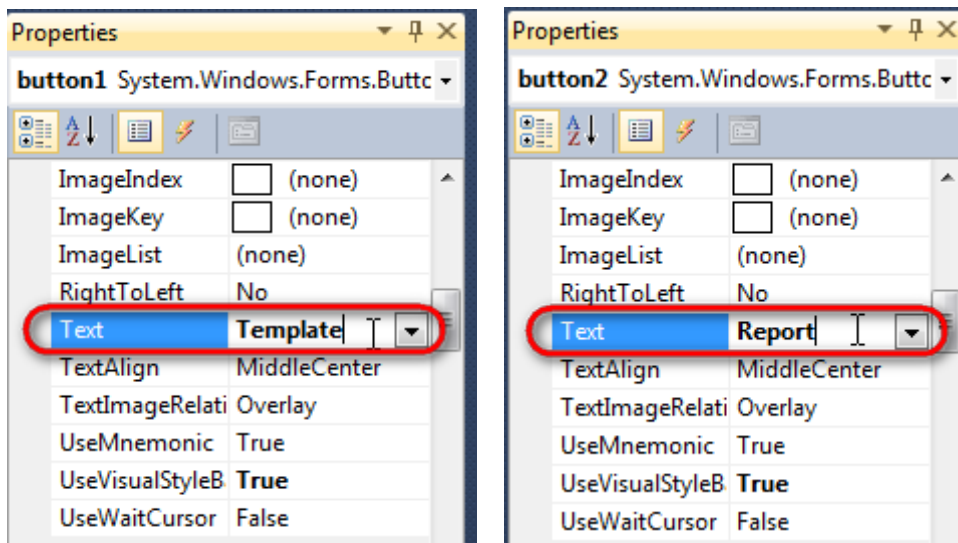
```
private Customer customer = new Customer();
private void SetData()
{
    customer.Address = dataGridView1[3,
dataGridView1.CurrentRow.Index].Value.ToString();
    customer.CompanyName = dataGridView1[0,
dataGridView1.CurrentRow.Index].Value.ToString();
    customer.ContactName = dataGridView1[1,
dataGridView1.CurrentRow.Index].Value.ToString();
    customer.Phone = dataGridView1[2,
dataGridView1.CurrentRow.Index].Value.ToString();
    reportManager1.DataSources.Add("Customer", customer );
}
}
```

Step 11

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



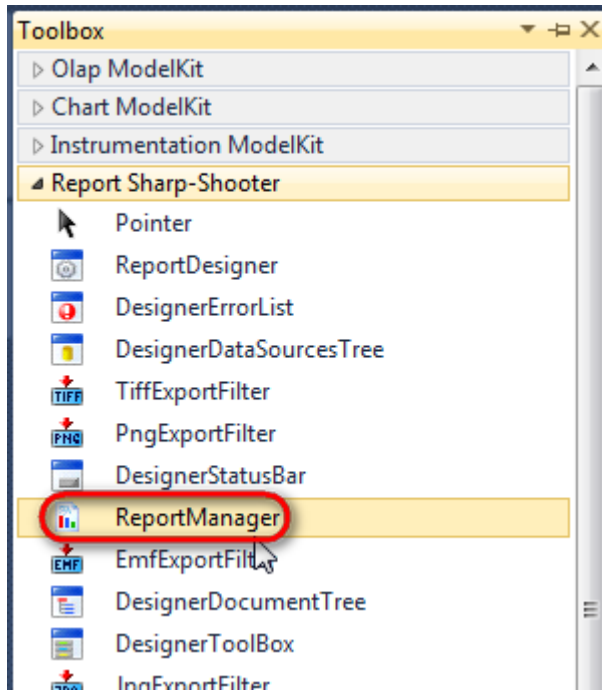
Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    SetData();
    inlineReportSlot1.DesignTemplate();
}

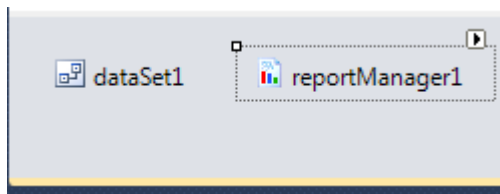
private void button2_Click(object sender, EventArgs e)
{
    SetData();
    inlineReportSlot1.Prepare();
}
```

Step 12

Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

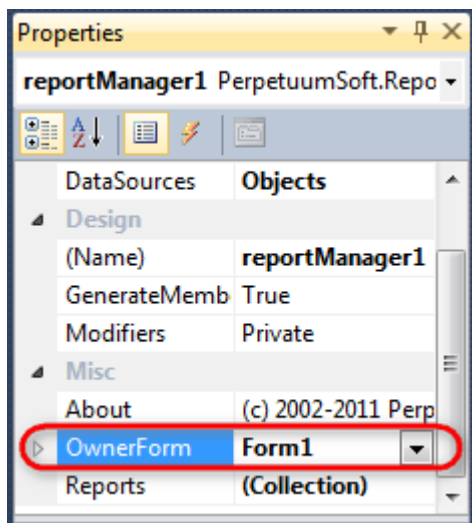


The component is available in the lower part of the window.



Step 13

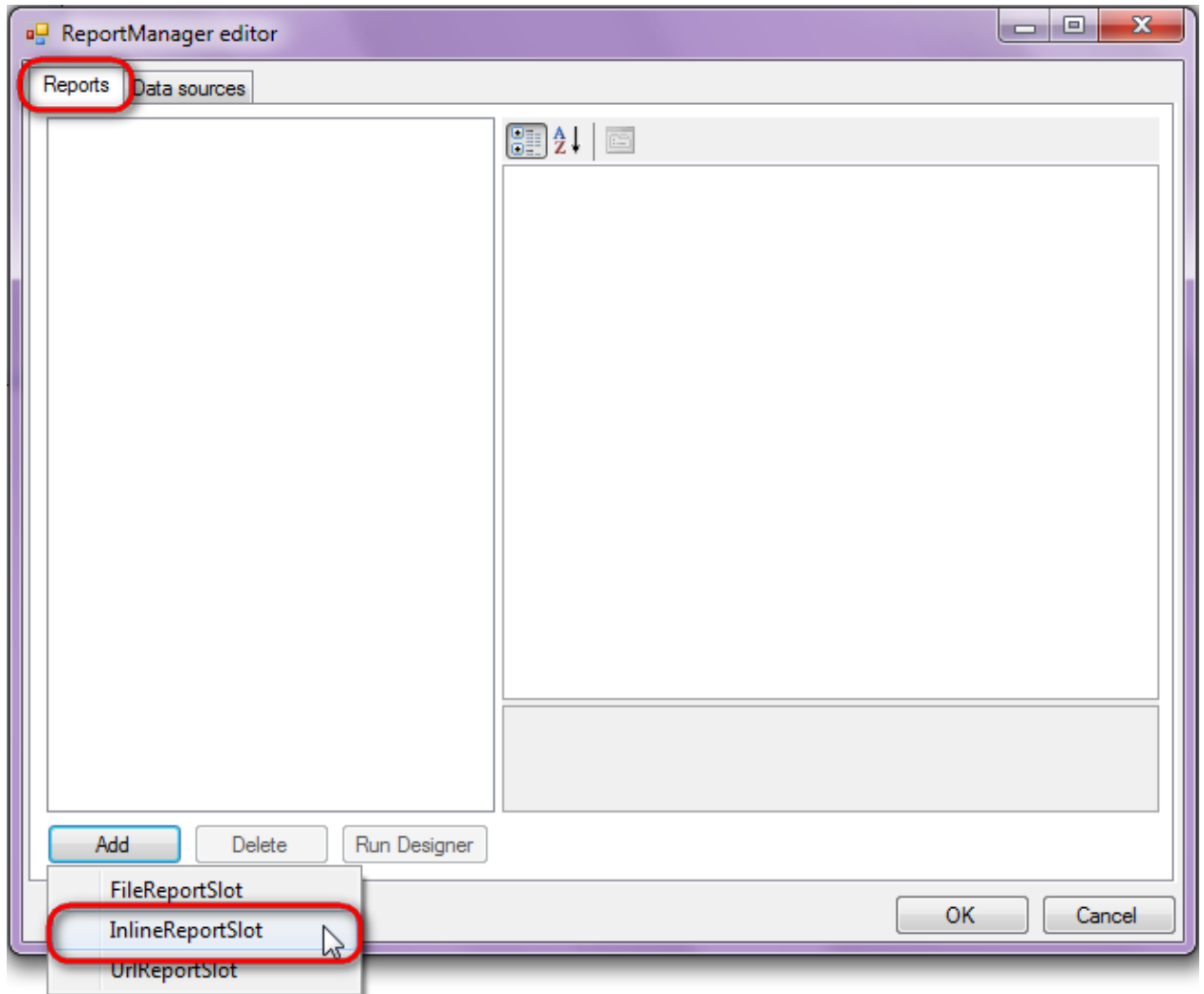
On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 14

Double click on ReportManager to open ReportManager editor.

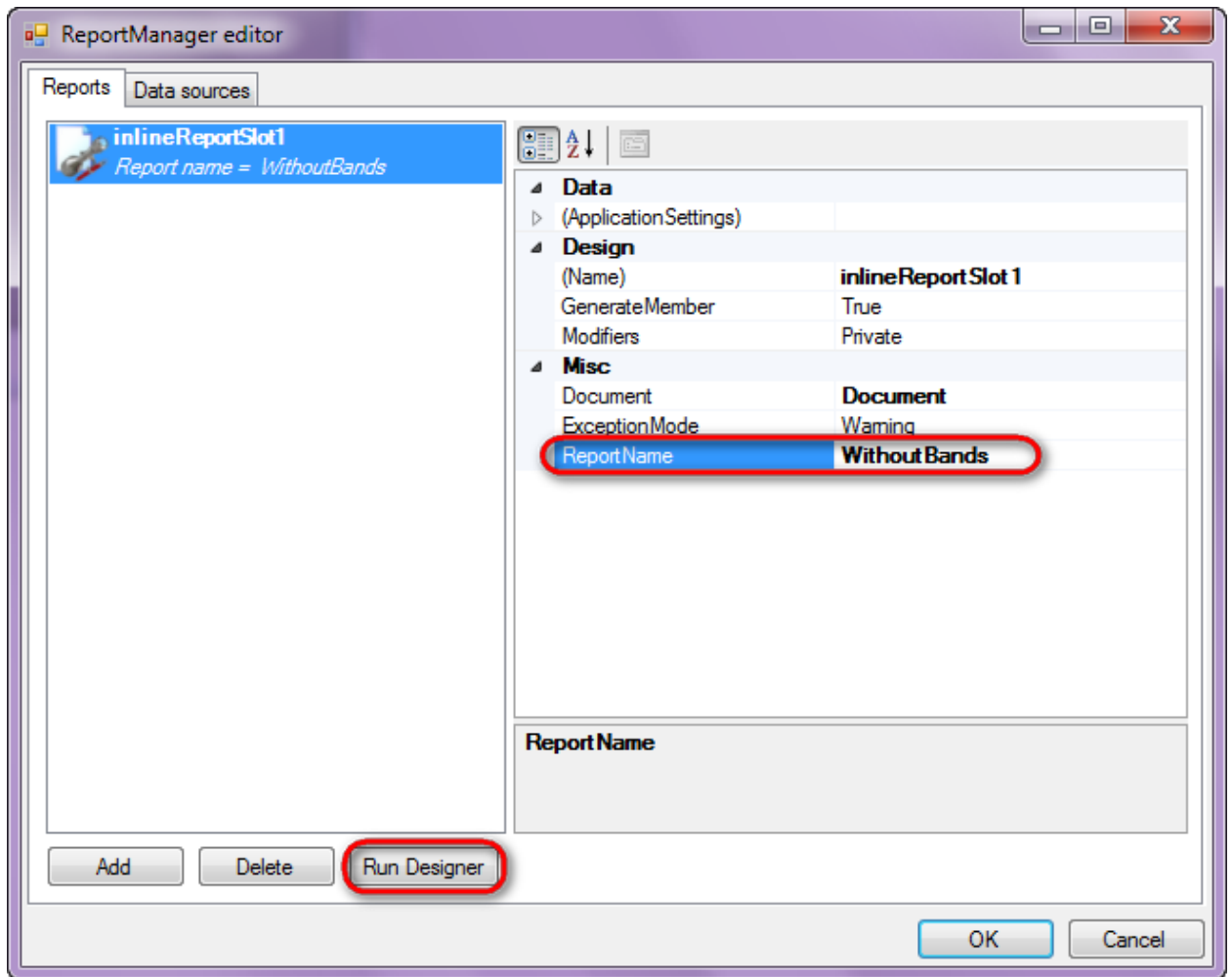
On the "Reports" tab, click "Add" and select "InlineReportSlot".



Step 15

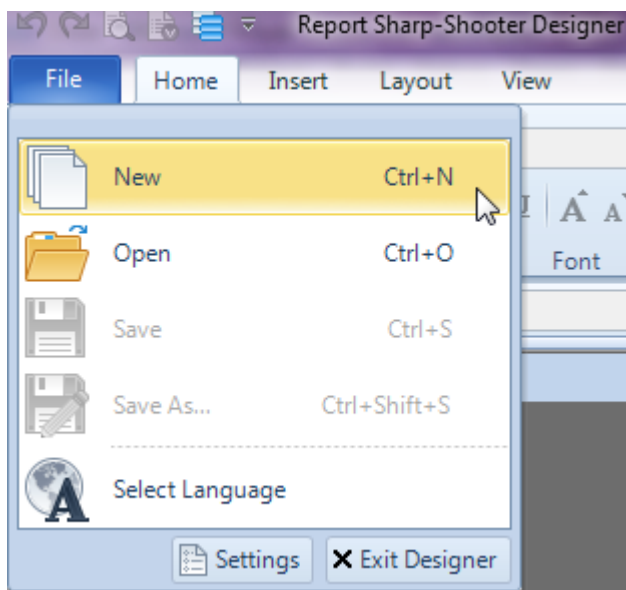
Set name of the report in the property ReportName – “WithoutBands”.

Click “Run Designer” in order to open template editor – “Report Designer”.

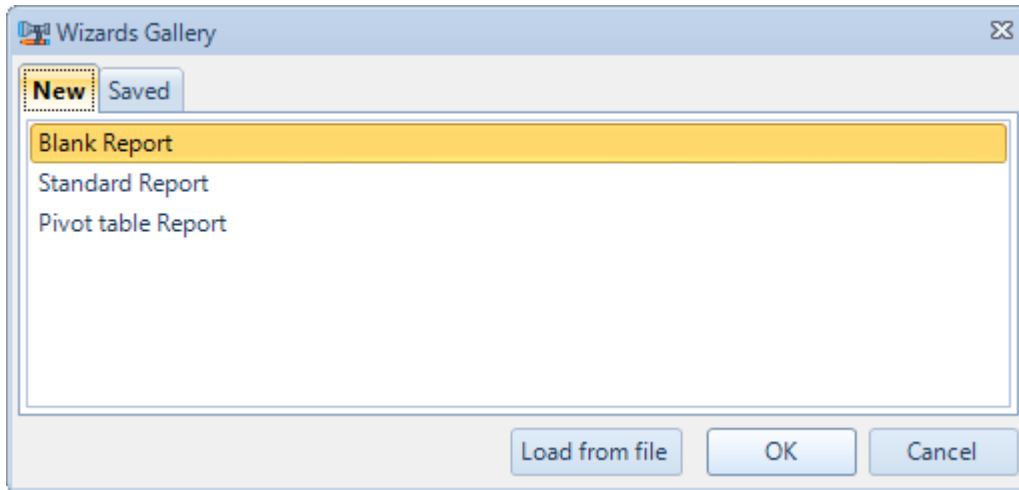


Step 16

Create new empty template – select File\New from the main menu.

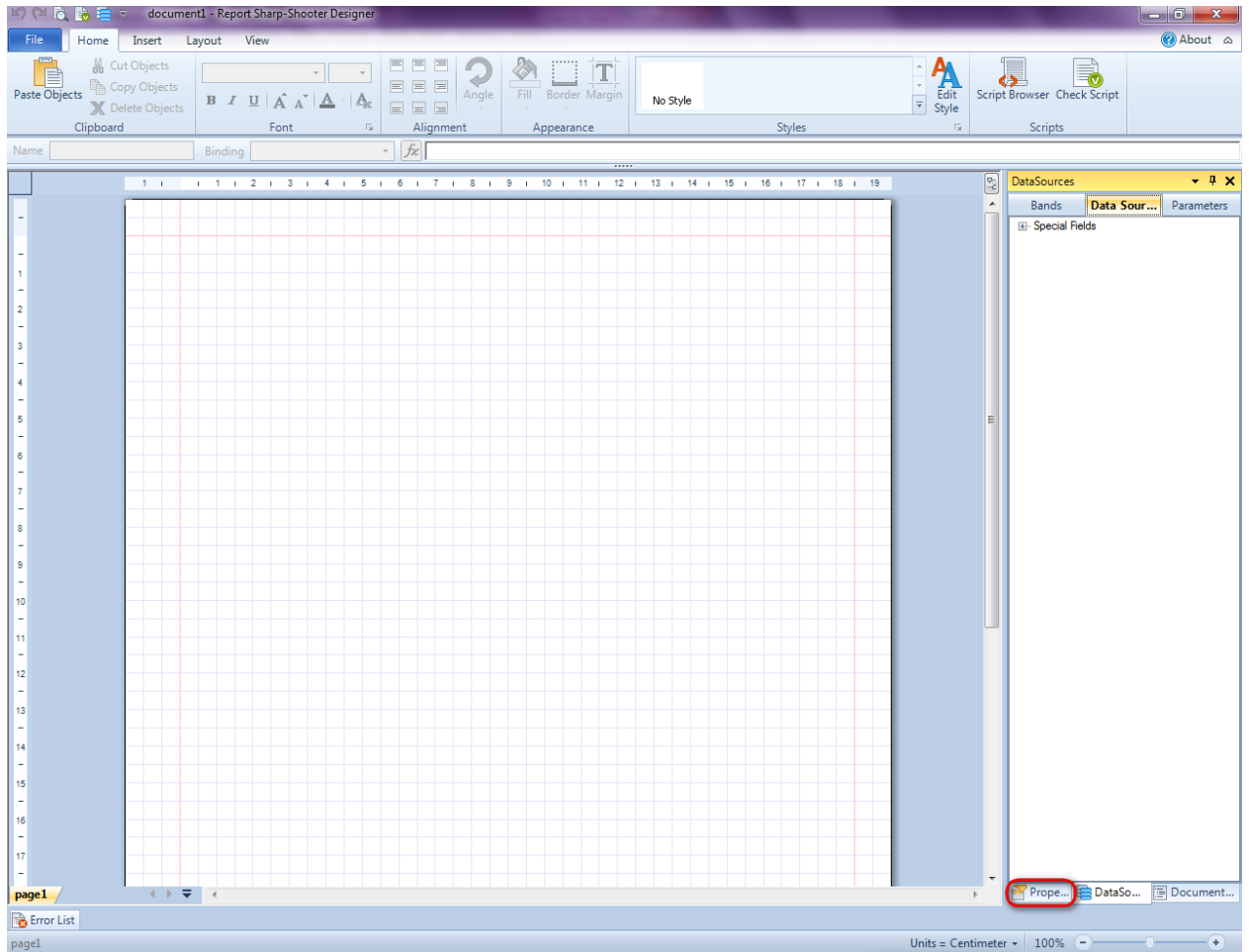


Select "Blank Report" in the Wizards Gallery and click "OK".



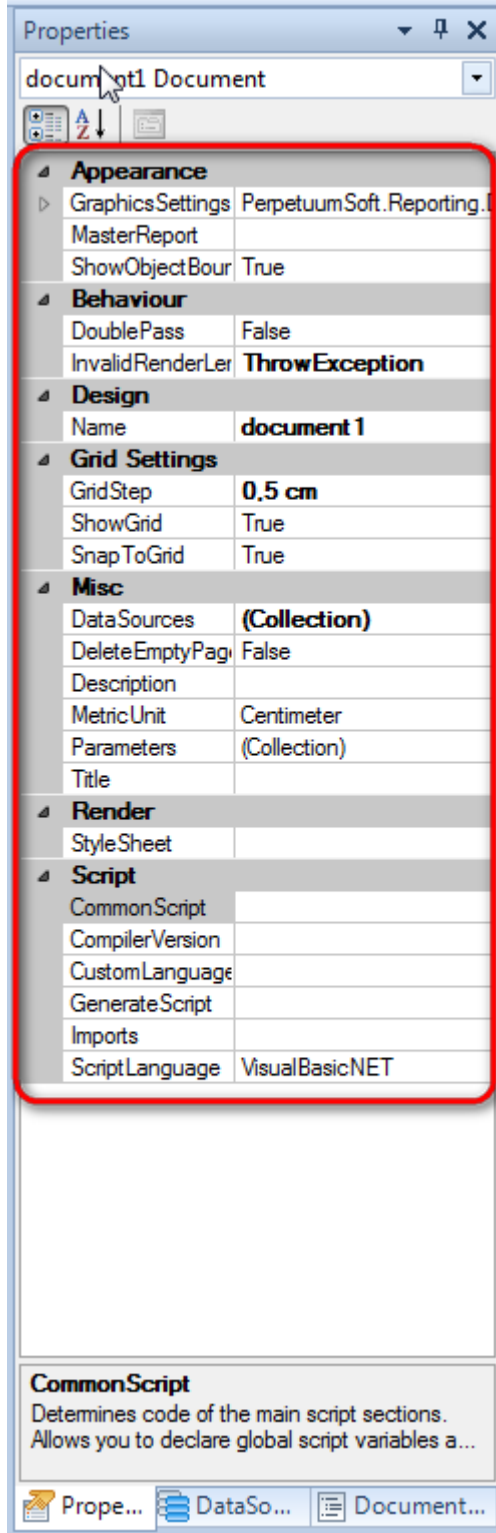
Step 17

Click the "Properties" tab of the tool window in the right part of the designer.

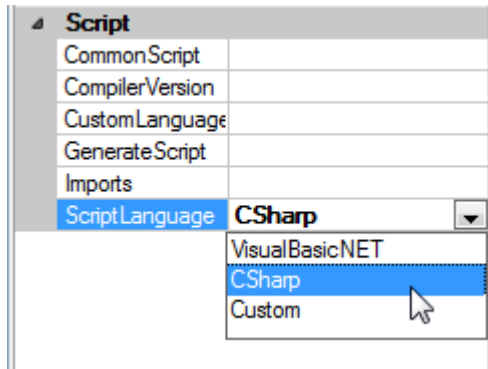




You will see properties of the edited template on the "Properties" tab

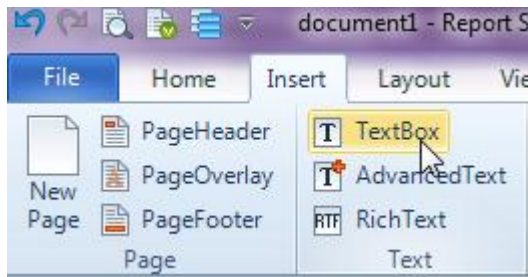


Set property ScriptLanguage = CSharp.



Step 18

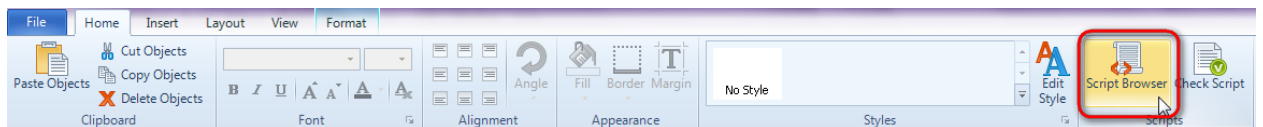
Press button "TextBox" on the Insert tab in the group Text.



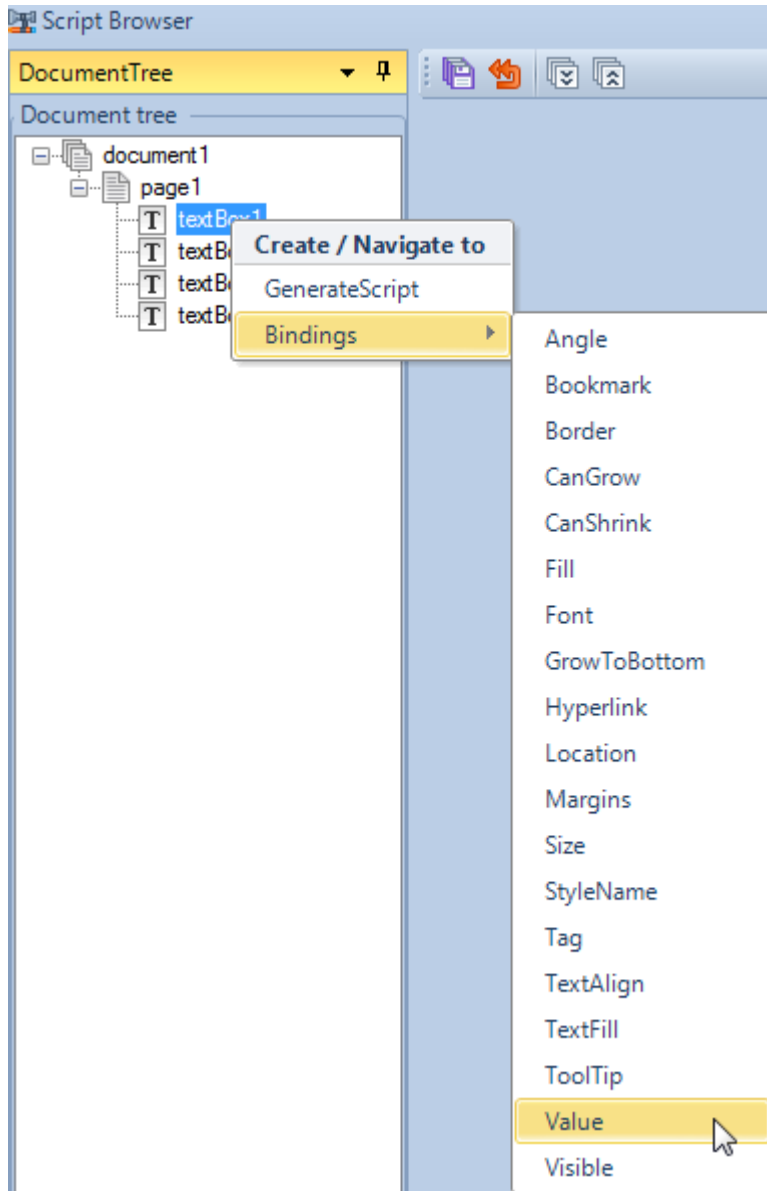
Click on the template area to add TextBox element to the template. In the same way add three more TextBox elements.

Step 19

Open Script Browser - press the "Script Browser" button in the Home tab in the Scripts group.



Right click on textBox1 in DocumentTree. Select Bindings\Value in the context menu.



Write the "GetData("Customer.CompanyName")" script in the appeared window.



Step 20

Set the Value property to "GetData("Customer.Address")", "GetData("Customer.ContactName")", "GetData("Customer.Phone")" for other TextBoxes in the same way.

Press the "Save all" button.

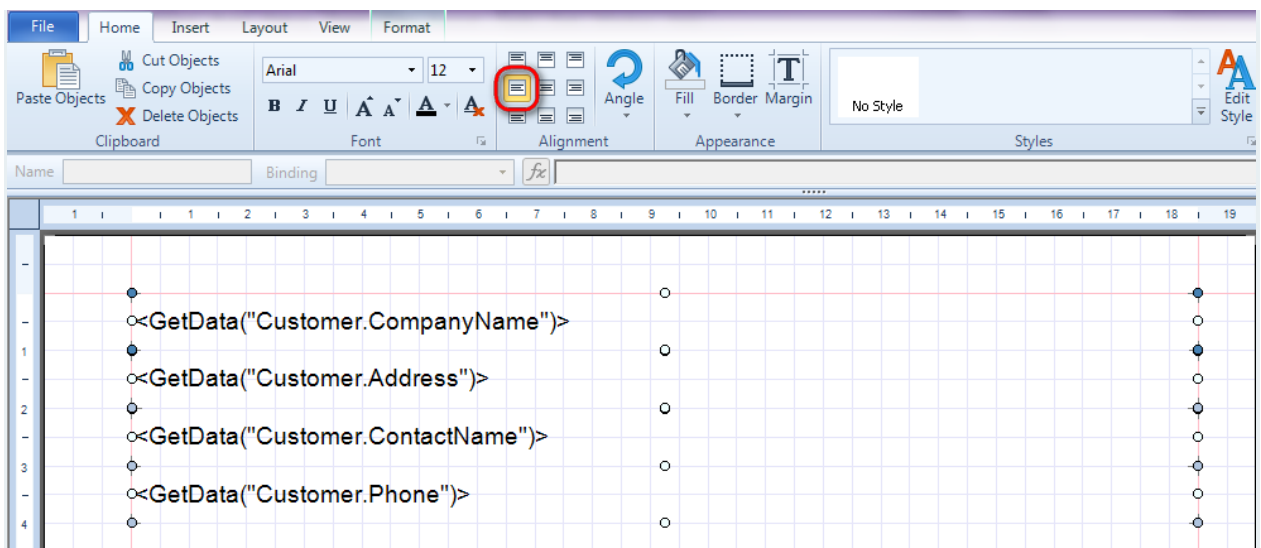


Close Script Browser.

Step 21

Press Shift and select all the elements by the left mouse button. Click "Left Alignment" button on the Home tab in the group Alignment.

Change size of the elements and set positions in the following way:



Step 22

Save template, close Report Designer.

Step 23

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

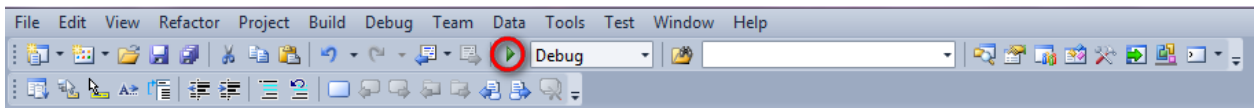
```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["Address"] = "Obere Str. 57";
    row["ContactName"] = "Maria Anders";
    row["Phone"] = "030-0074321";
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
```




```
row["Address"] = "Avda. de la Constitución 2222";
row["ContactName"] = "Ana Trujillo";
row["Phone"] = "(5) 555-4729";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Ernst Handel";
row["Address"] = "Kirchgasse 6";
row["ContactName"] = "Roland Mendel";
row["Phone"] = "7675-3425";
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Toms Spezialitäten";
row["Address"] = "Luisenstr. 48";
row["ContactName"] = "Karin Josephs";
row["Phone"] = "0251-031259";
dataTable1.Rows.Add(row);
inlineReportSlot1.RenderCompleted += new
EventHandler(reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    PerpetuumSoft.Reporting.Components.IReportSource rs = sender as
PerpetuumSoft.Reporting.Components.IReportSource;
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(rs))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 24

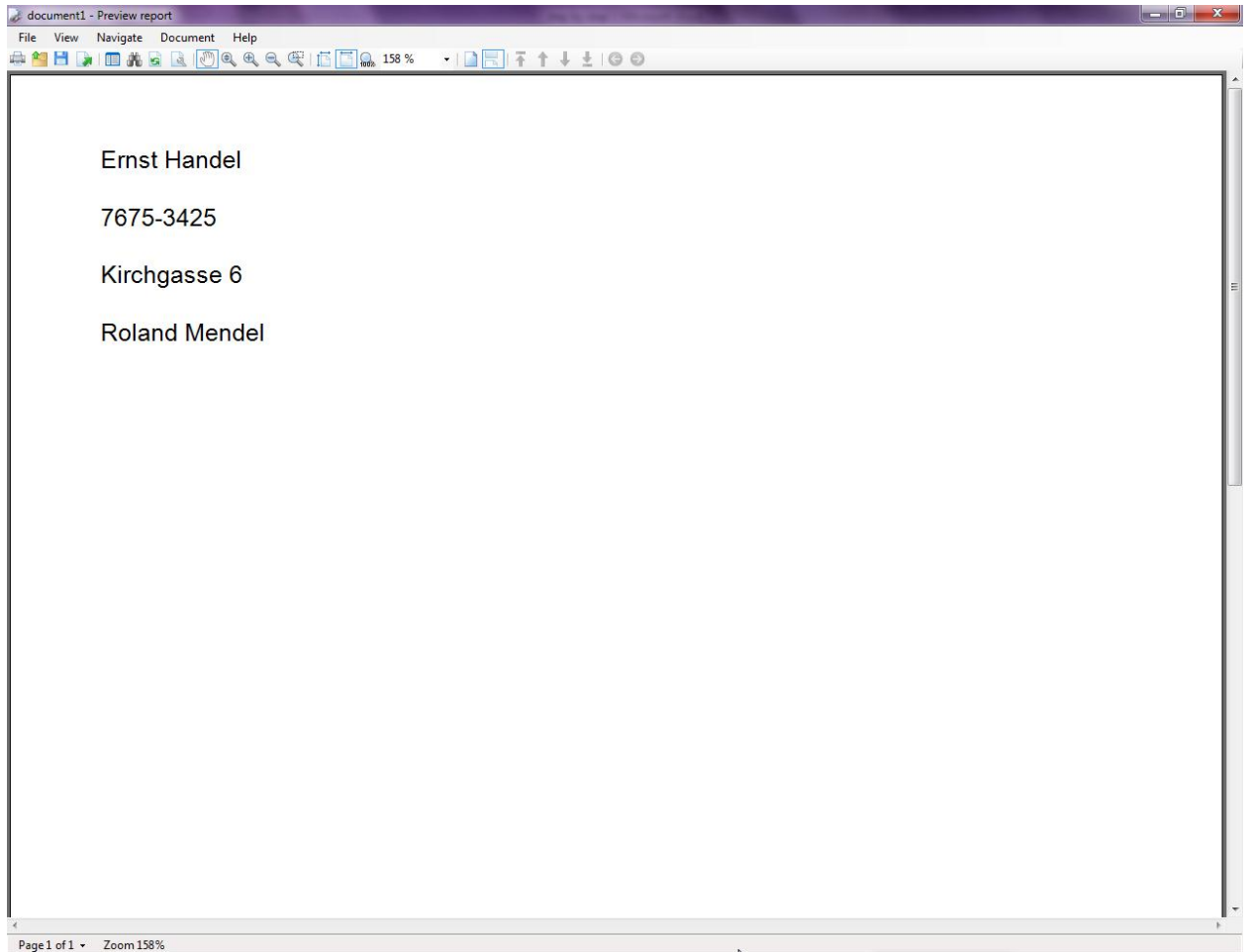
Click "Start Debugging" on the Visual Studio toolbar in order to start application.



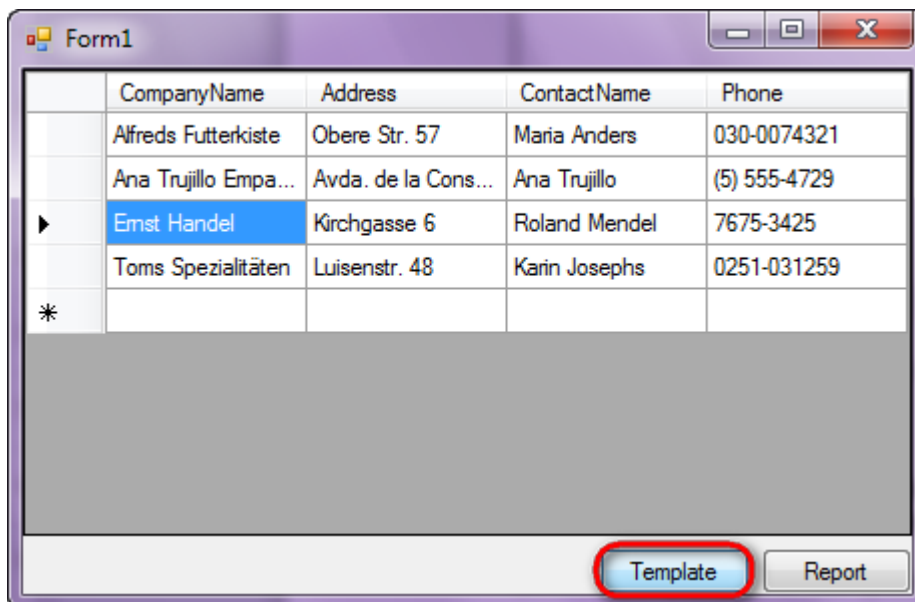
On the application form, select any record and click "Report" button to open card with the information on the selected company.

	CompanyName	Address	ContactName	Phone
	Alfreds Futterkiste	Obere Str. 57	Maria Anders	030-0074321
	Ana Trujillo Empa...	Avda. de la Cons...	Ana Trujillo	(5) 555-4729
▶	Ernst Handel	Kirchgasse 6	Roland Mendel	7675-3425
	Toms Spezialitäten	Luisenstr. 48	Karin Josephs	0251-031259
*				

Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar application sample is located in the following folder "\\Perpetuum Software\Net ModelKit Suite\ Samples\Report Sharp-Shooter\CSharp\WithoutBands".

Similar sample in the Samples Center is Reports\Simple Reports\Without Bands.

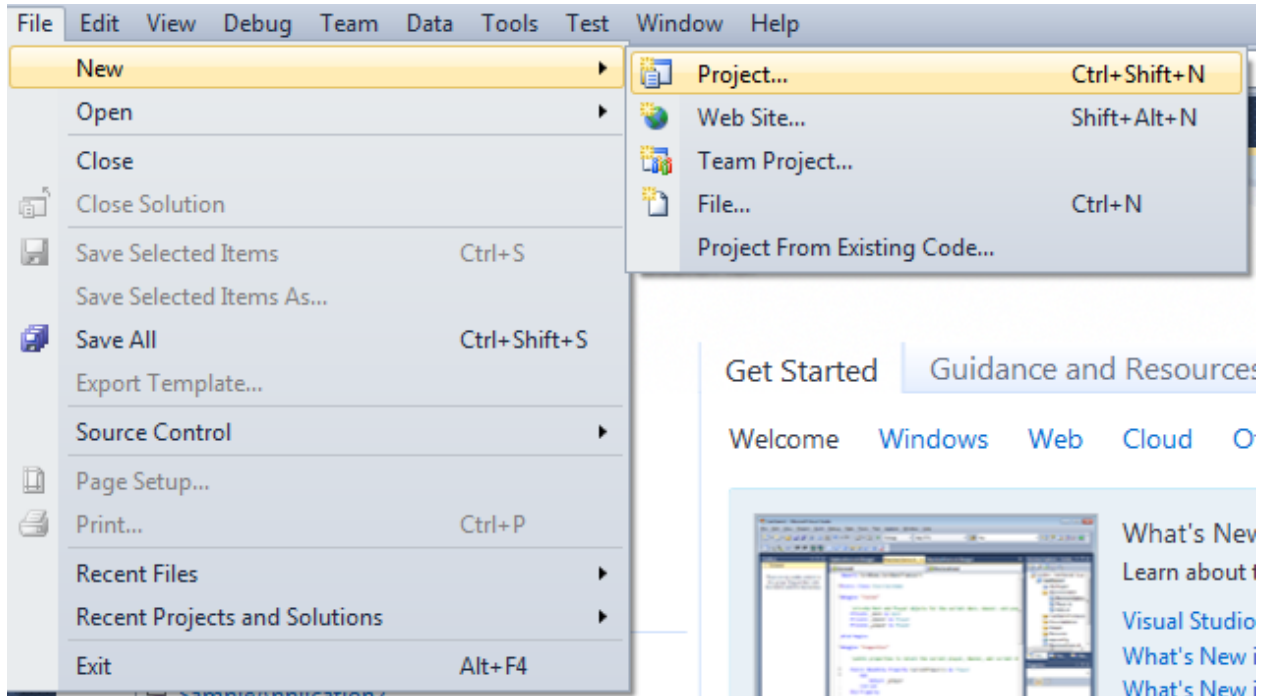


Parameterized Report

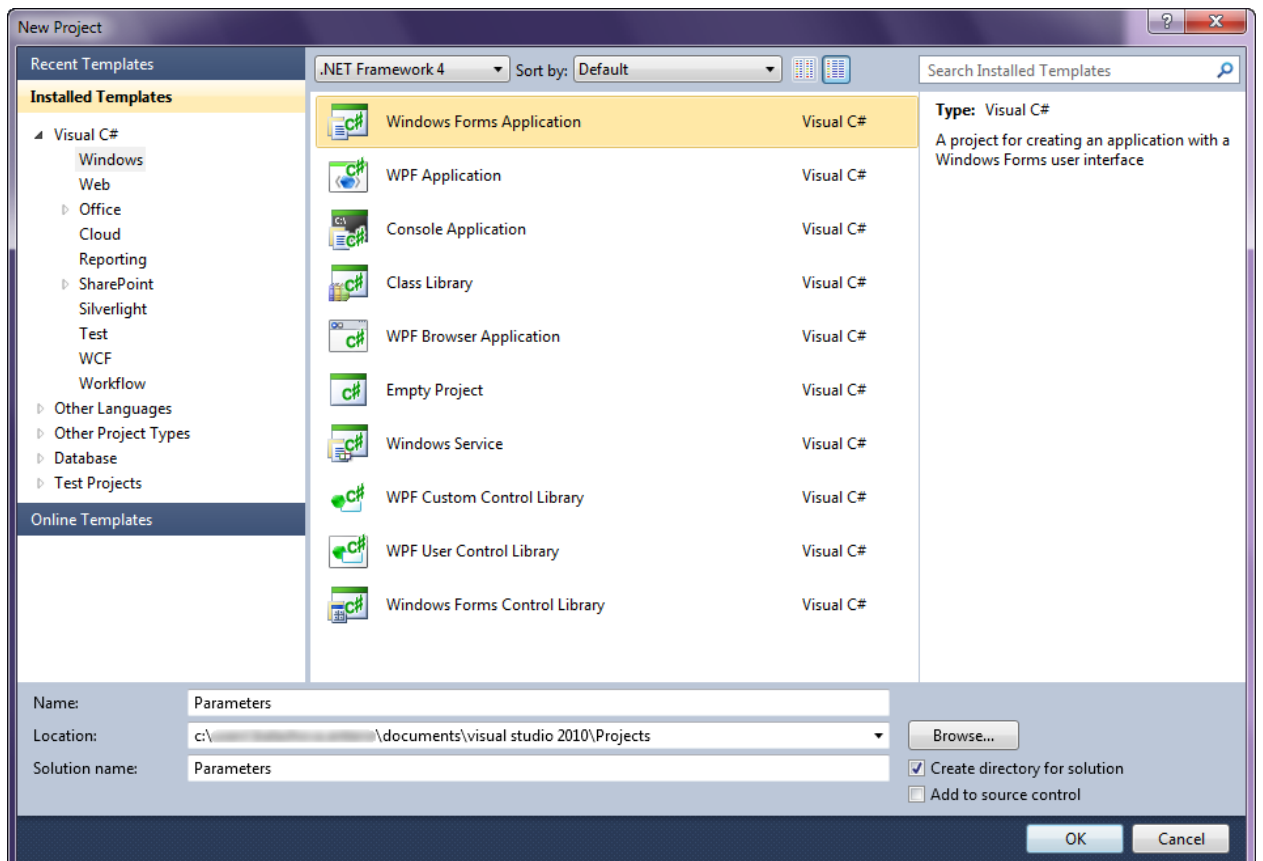
Template of a report containing date and time set by the user.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

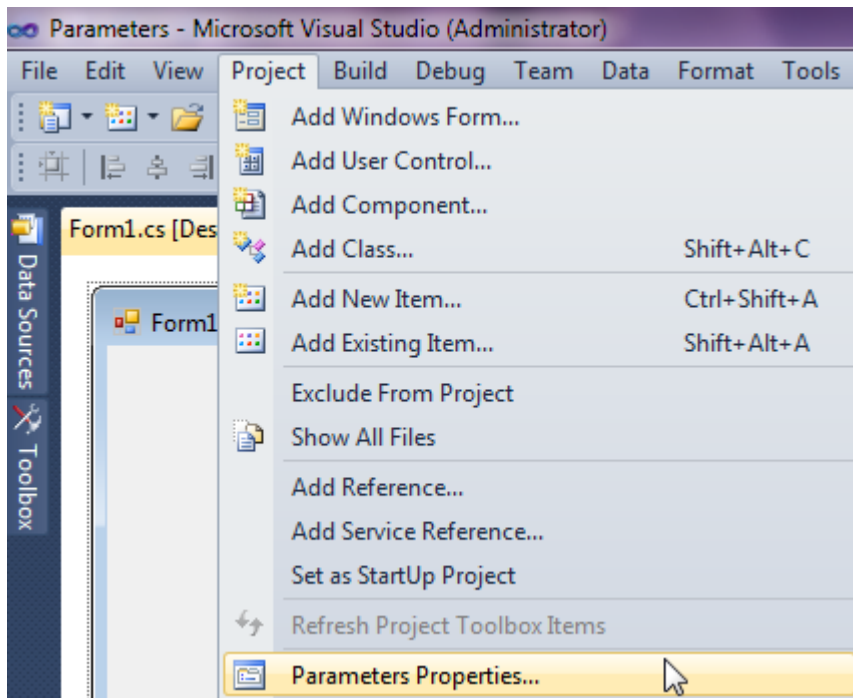


Select Windows Forms Application, set project name – “Parameters”, set directory to save the project to.

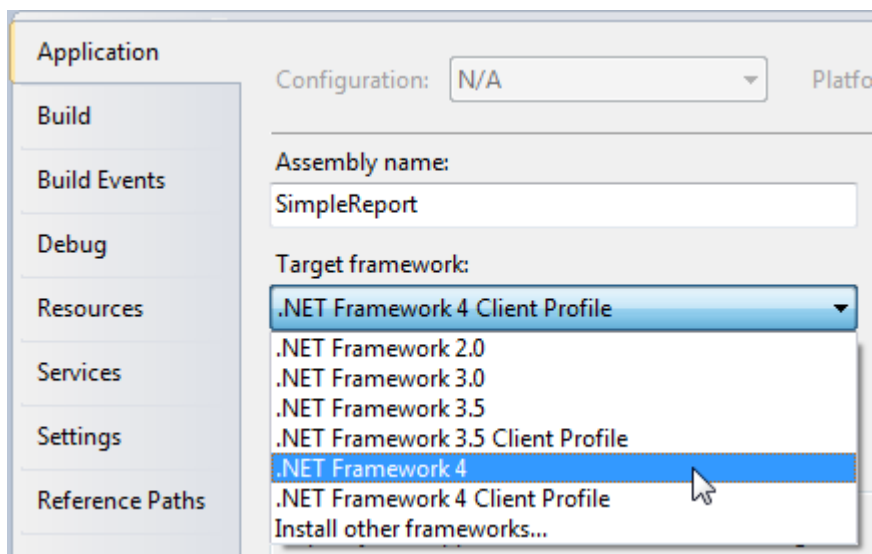


Step 2

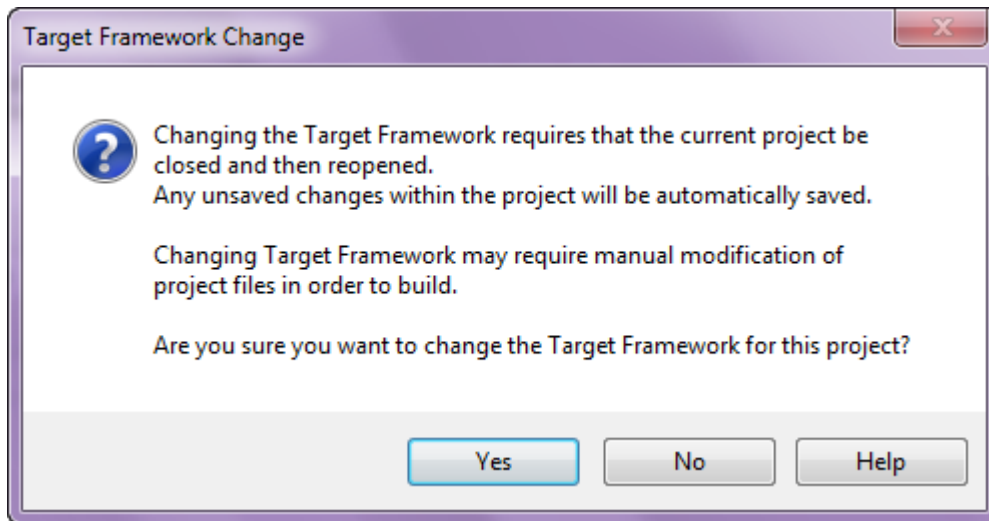
Change the project properties. Select the Project\Parameters Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

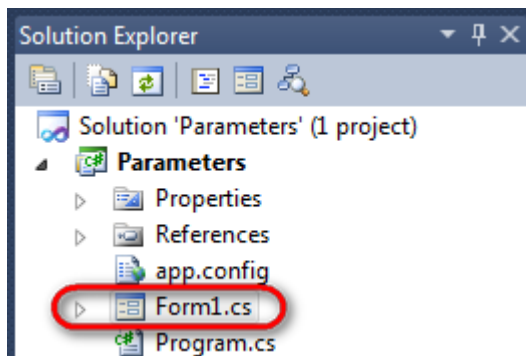


In the opened window press the "Yes" button.

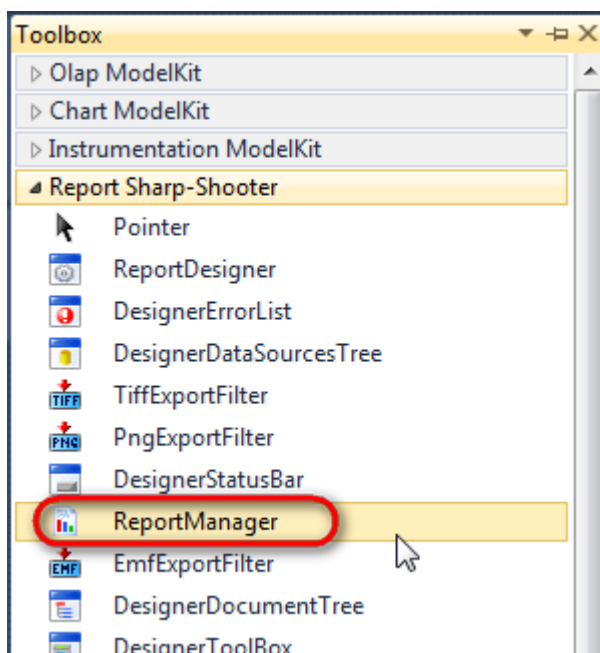


Step 3

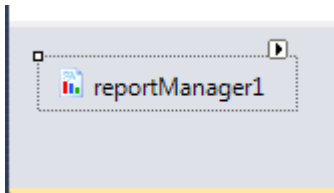
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

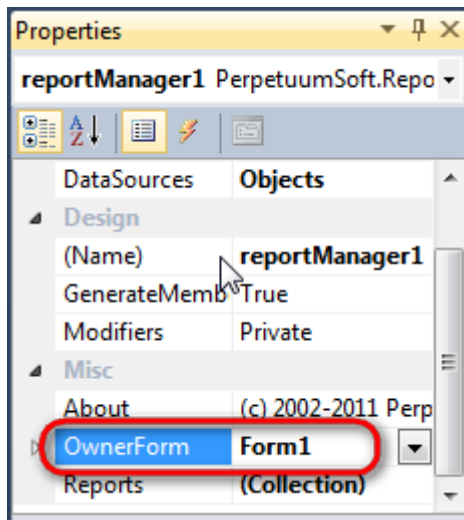


The component is available in the lower part of the window.



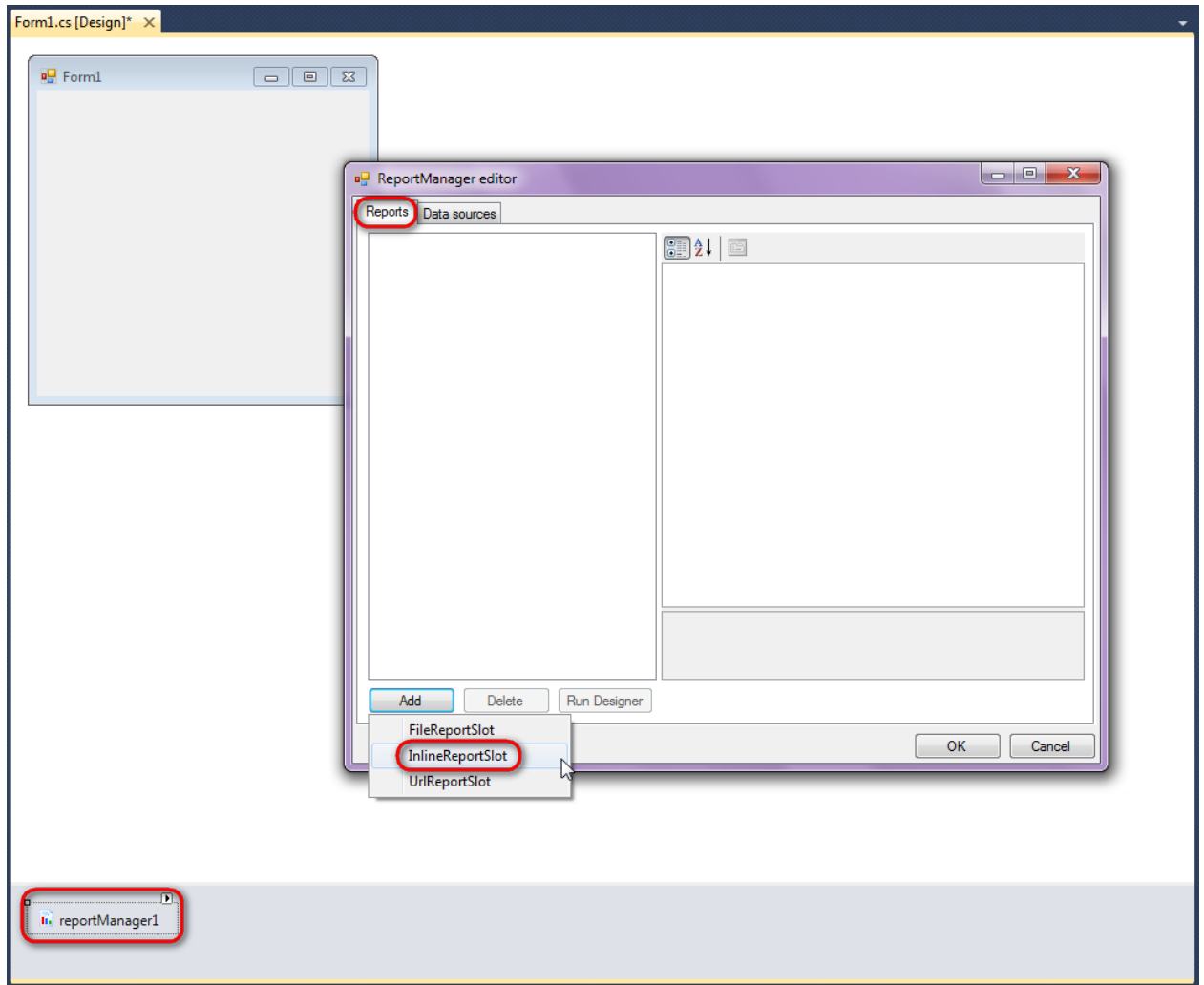
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

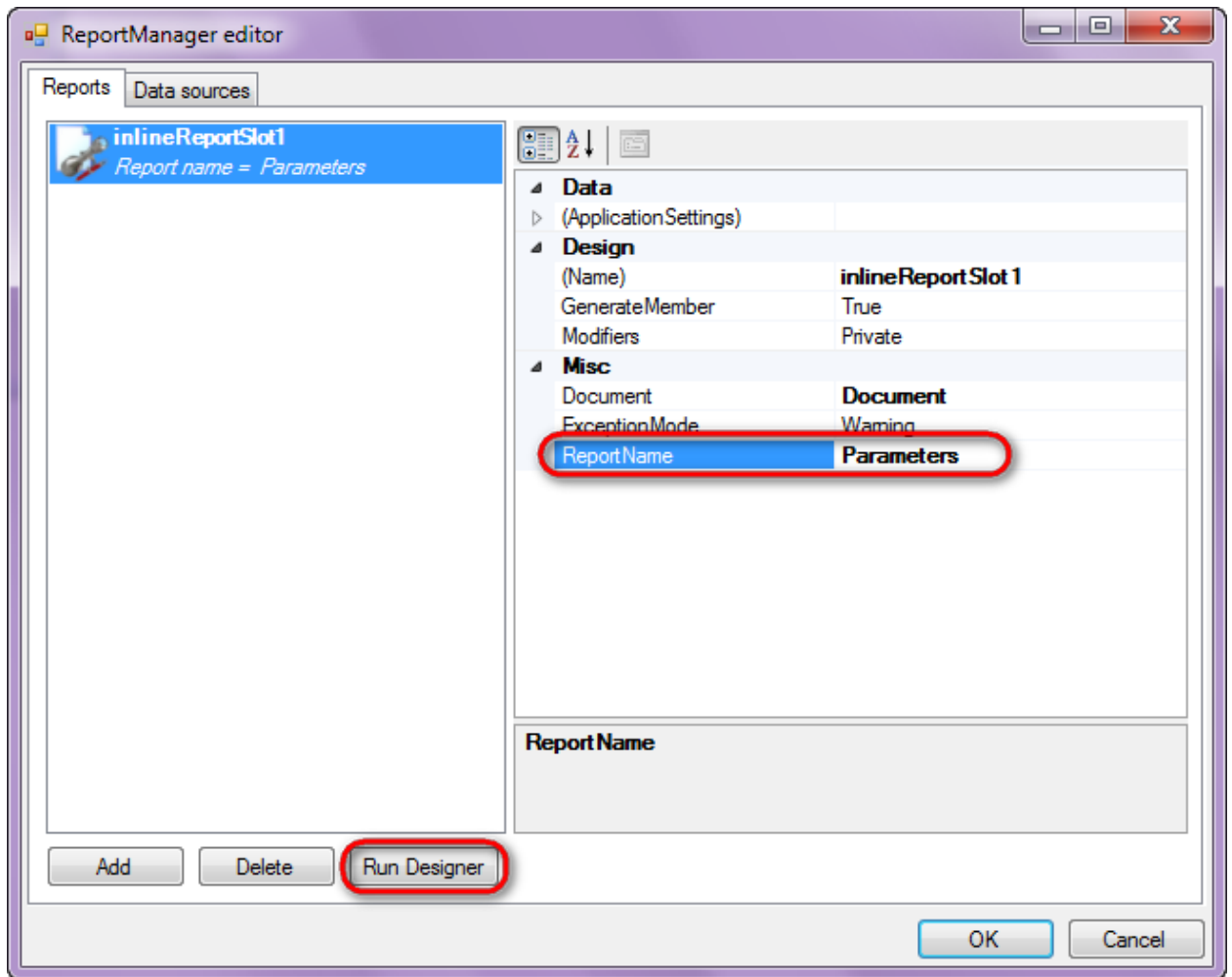


On the "Reports" tab, click "Add" and select "InlineReportSlot".

Step 6

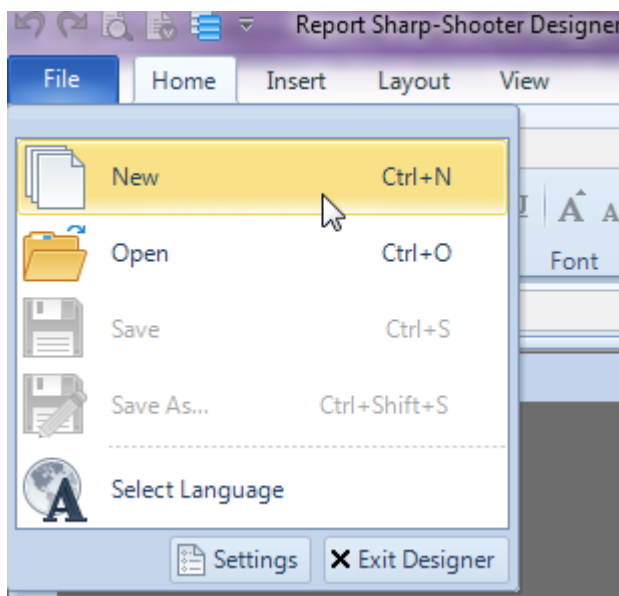
Set name of the report in the property ReportName – "Parameters".

Click "Run Designer" in order to open template editor – Report Designer.

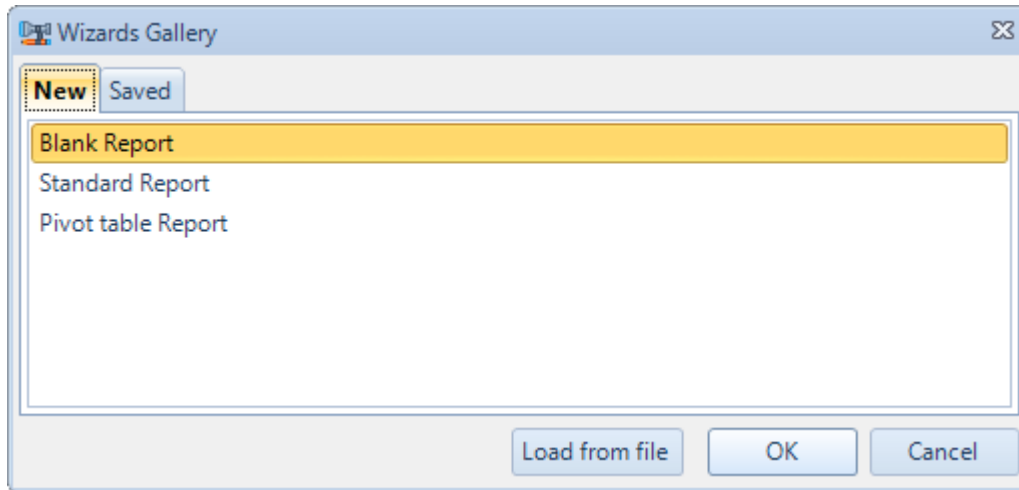


Step 7

Create new empty template – select item File\New from the main menu.

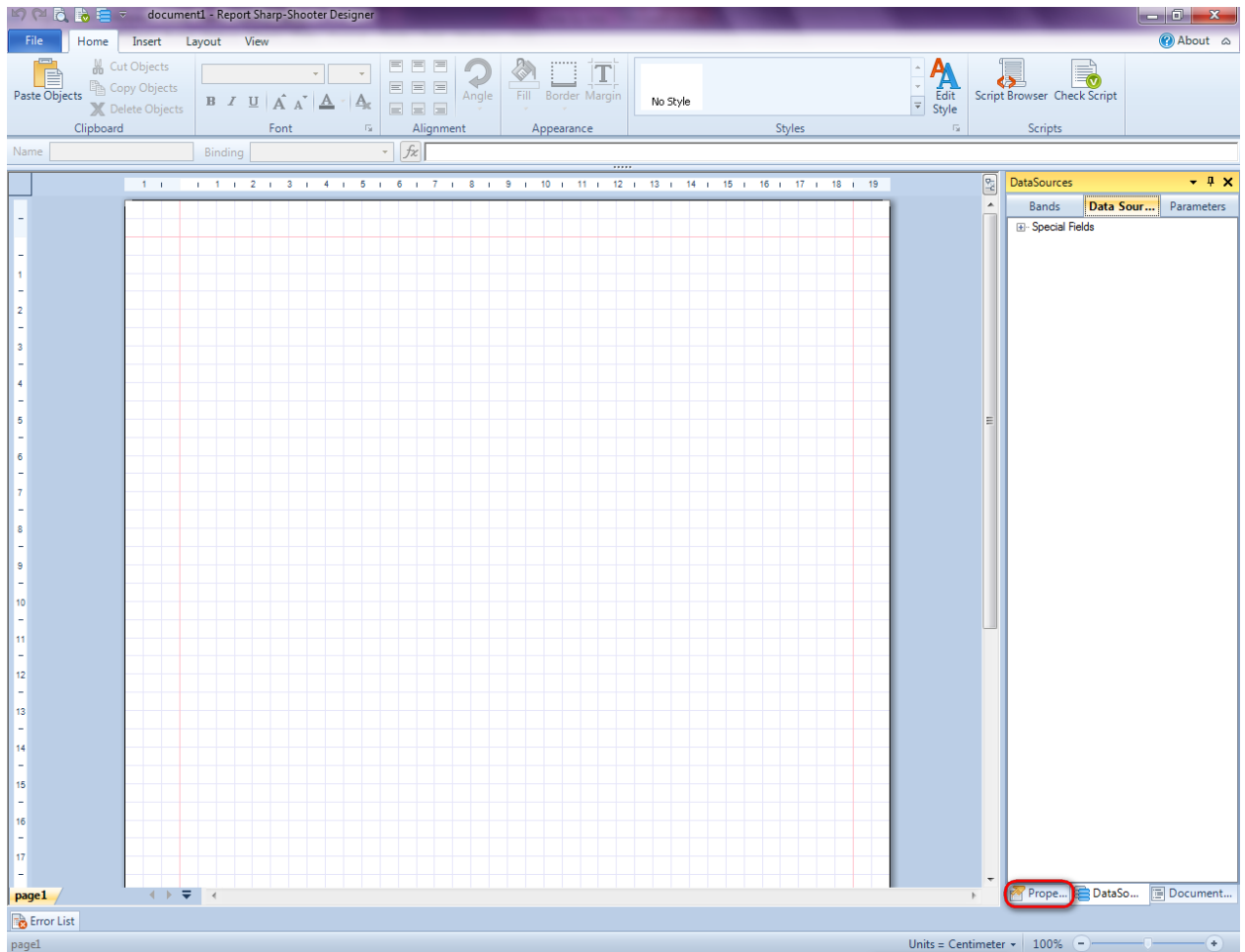


Select "Blank Report" in the Wizards Gallery and click "OK".

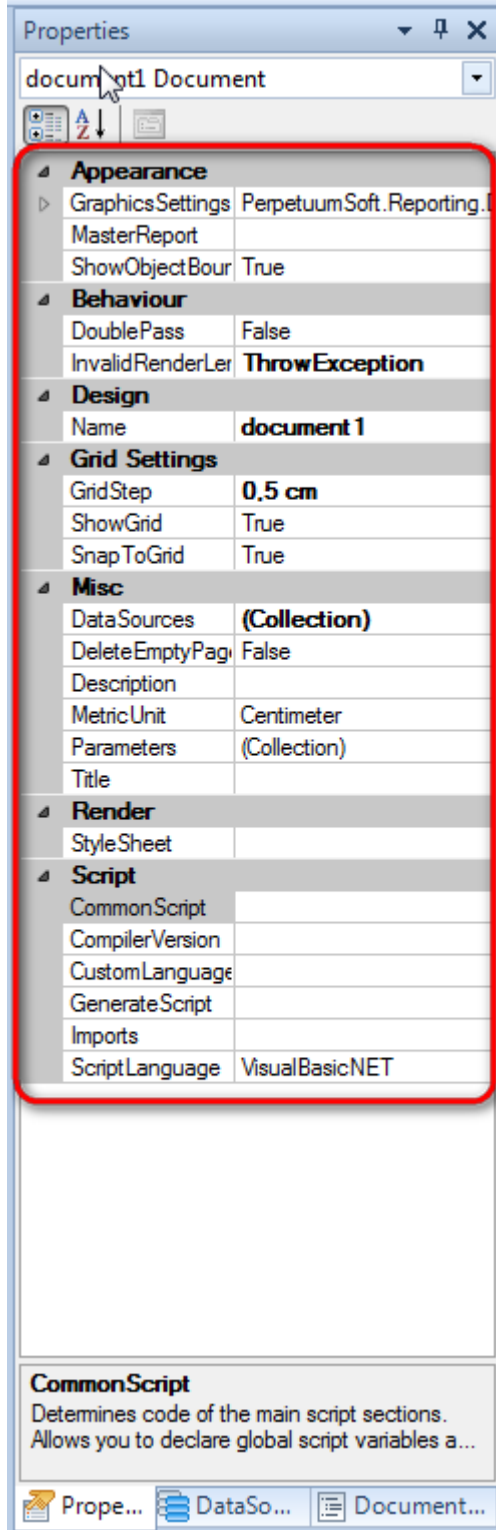


Step 8

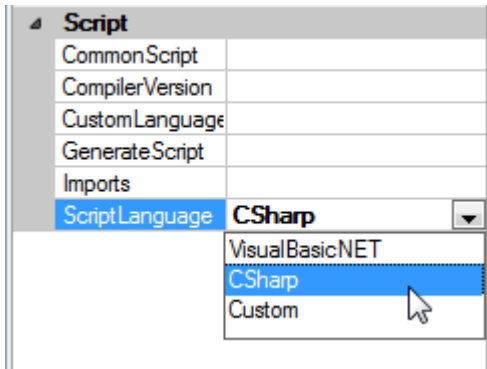
Click the "Properties" tab of the tool window in the right part of the designer.




You will see properties of the edited template on the “Properties” tab

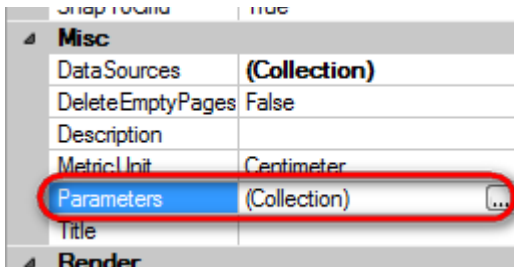



Set property ScriptLanguage = CSharp.



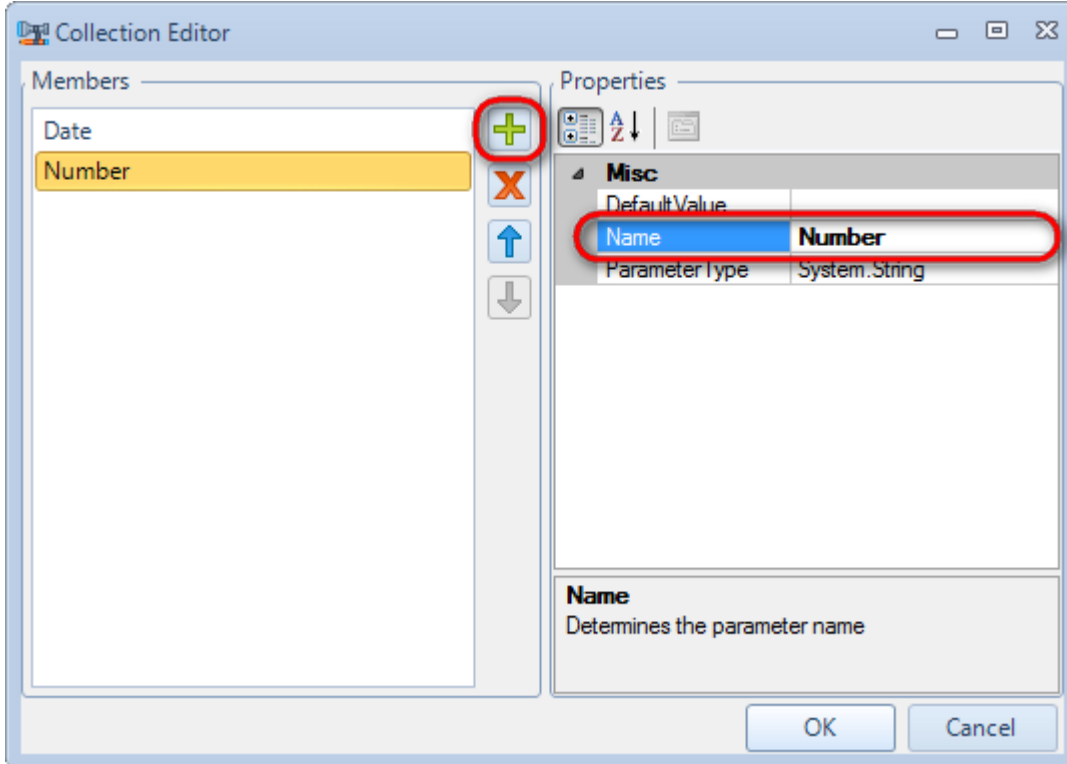
Step 9

Select Parameters property, click button  to open collection editor.



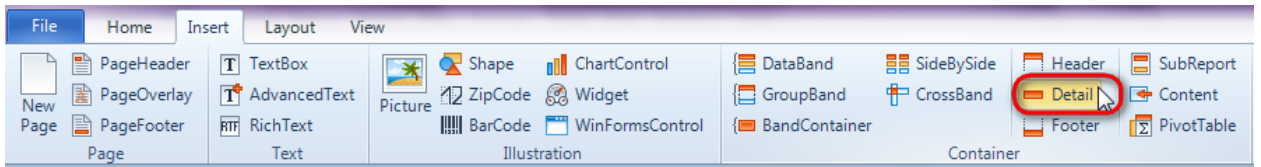
In the Collection Editor, click  button to add a new parameter. Set property Name = Date.

In the same way add one more parameter and set property Name = Number.



Step 10

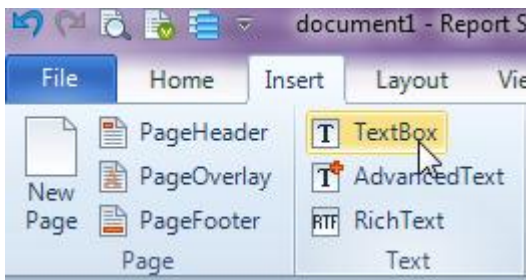
Press "Detail" button on the Insert tab in the group Container.



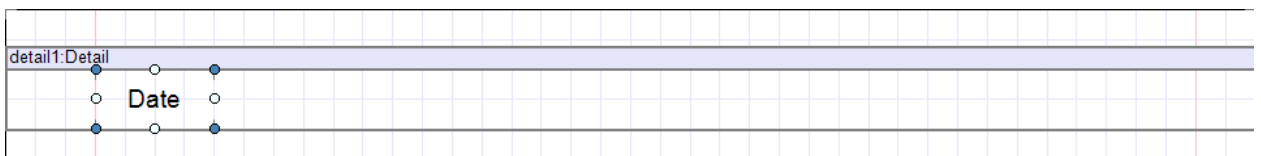
Click on the template area to add Detail band to the report template.

Step 11

Press button "TextBox" on the Insert tab in the group Text.

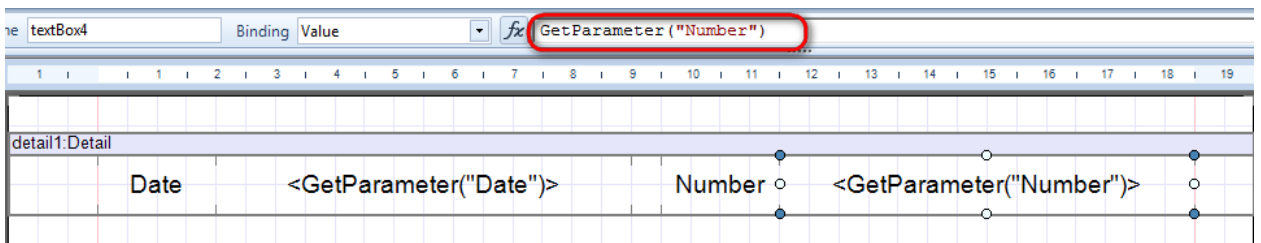


Click on the Detail band area to add TextBox element inside Detail. Set Text property to "Date".



Step 12

Add three more TextBox elements to the Detail band. Set the following properties for the first one: set Value = GetParameter("Date").



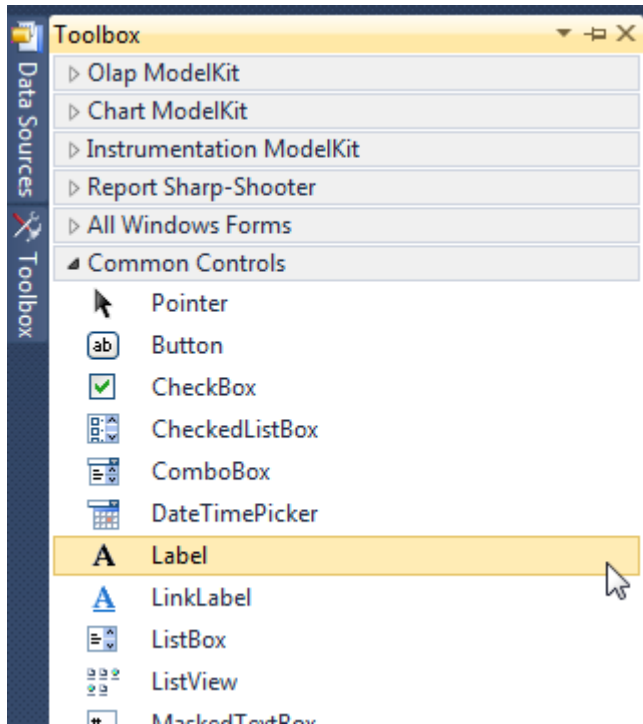
For the second one, set property Text = Number. For the third one, reset Text property and set Value to "GetParameter("Number")".

Step 13

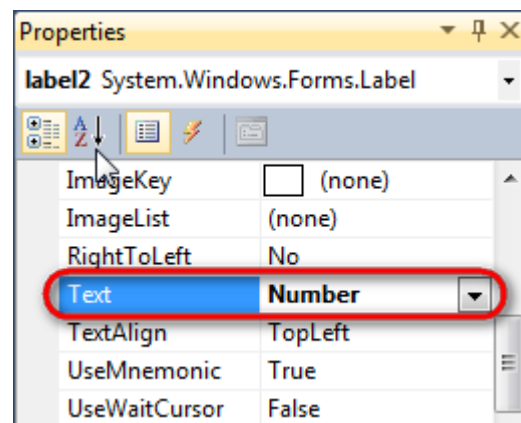
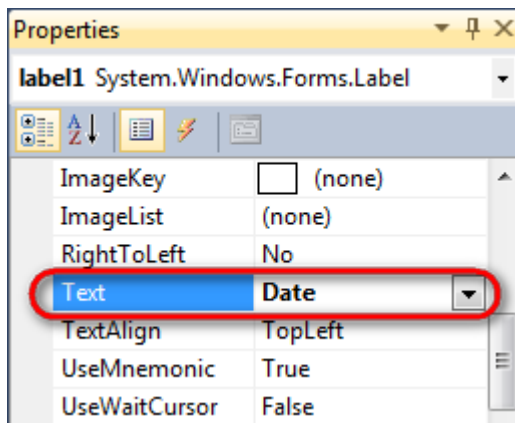
Save template, close Report Designer.

Step 14

Place two Label elements onto the form (drag and drop "Label" element from the Toolbox to the form).

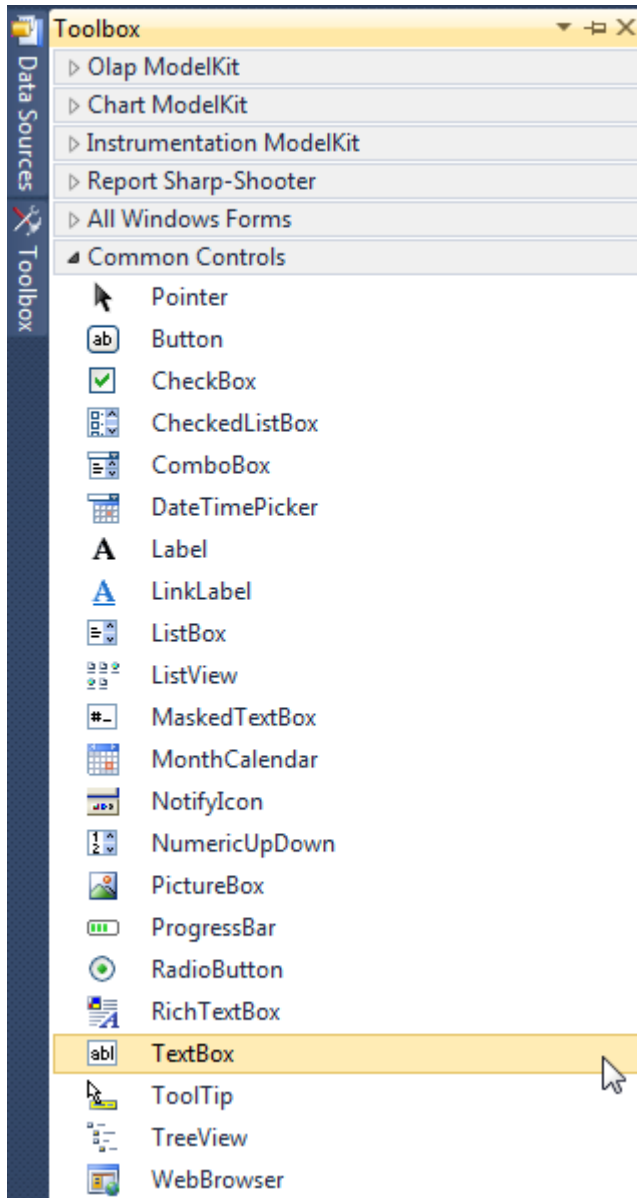


Select Label element on the form, select Text property. Set Text = Date for the first one, set Text = Number for the second one.

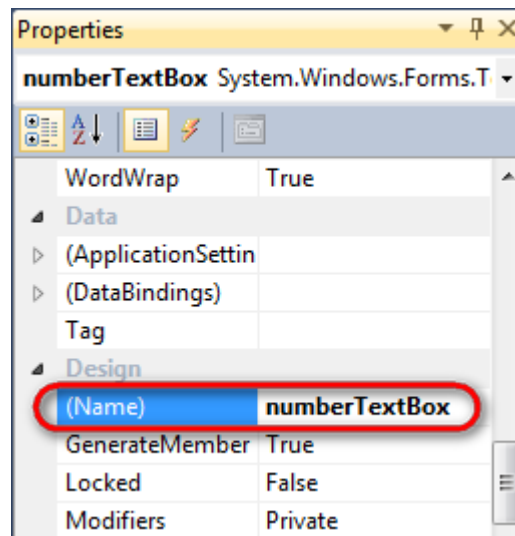
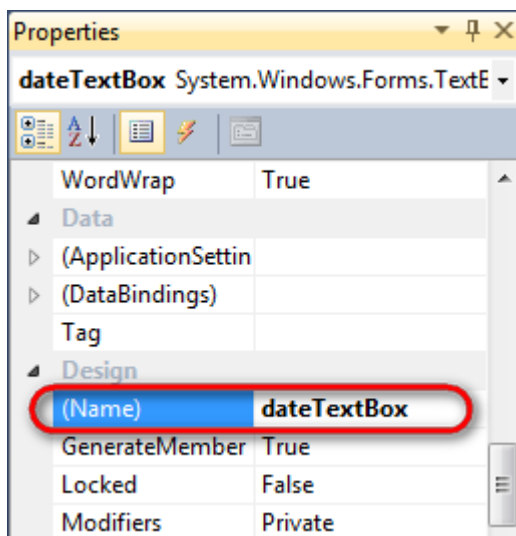


Step 15

Add two TextBox elements onto the form (drag and drop "TextBox" element from the Toolbox onto the form).

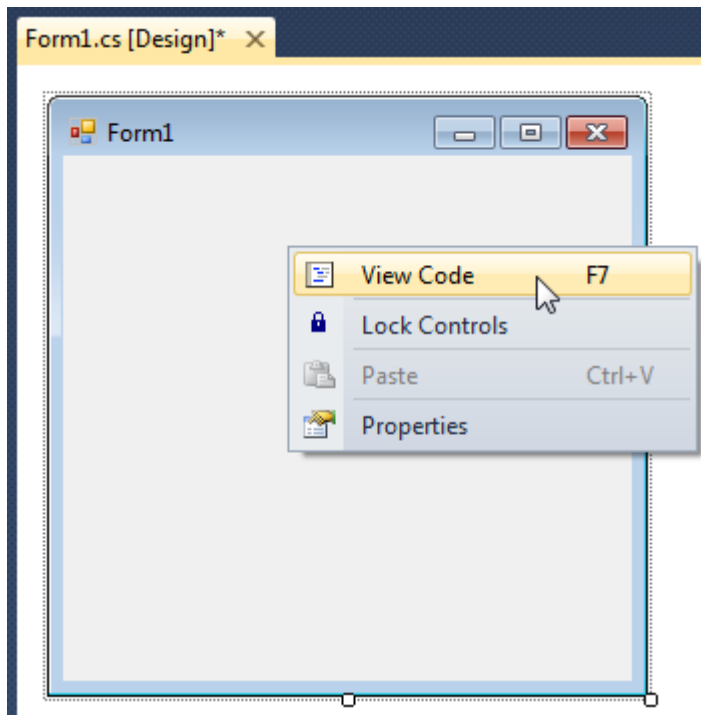


Select TextBox element on the form, edit Name property on the property grid. Set Name = dateTextBox for the first one, set Name = numberTextBox for the second one.



Step 16

Right click on the form and select "View Code" in the context menu to view code.



Add code to fill form fields and display report to the class constructor. Create RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent ();
    dateTextBox.Text = DateTime.Now.ToString ();
    numberTextBox.Text = Environment.TickCount.ToString ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted (object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 17

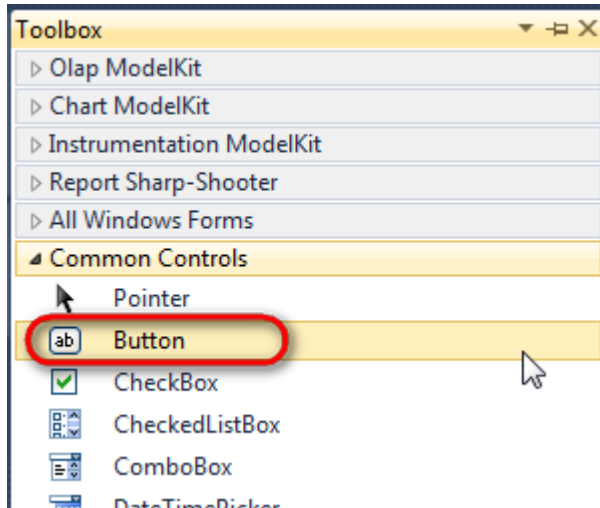
Add code to pass parameters to the report to the class constructor, create GetReportParameter event handler:

```
public Form1 ()
{
    InitializeComponent ();
    dateTextBox.Text = DateTime.Now.ToString ();
    numberTextBox.Text = Environment.TickCount.ToString ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
    inlineReportSlot1.GetReportParameter += new
PerpetuumSoft.Reporting.Components.GetReportParameterEventHandler (inlineRepor
tSlot1_GetReportParameter);
}
```

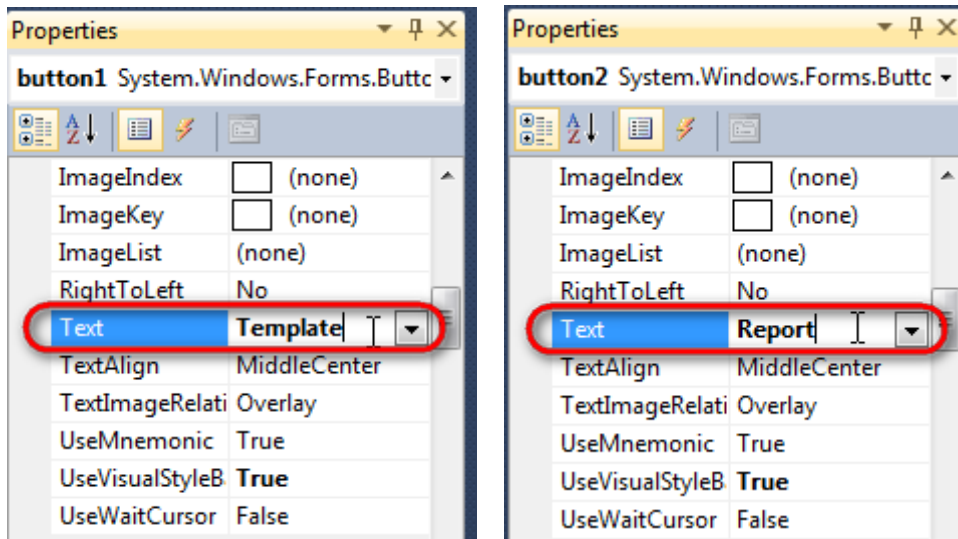
```
    }  
    private void inlineReportSlot1_GetReportParameter(object sender,  
    PerpetuumSoft.Reporting.Components.GetReportParameterEventArgs e)  
    {  
        e.Parameters["Date"].Value = dateTextBox.Text;  
        e.Parameters["Number"].Value = numberTextBox.Text;  
    }  
}
```

Step 18

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



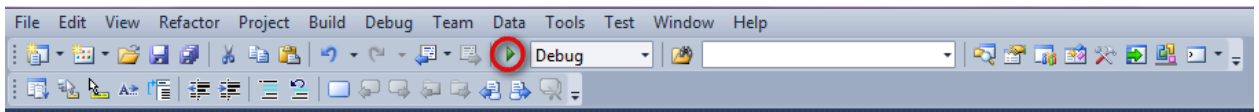
Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)  
{  
    inlineReportSlot1.DesignTemplate();  
}  
  
private void button2_Click(object sender, EventArgs e)  
{  
    inlineReportSlot1.Prepare();  
}
```

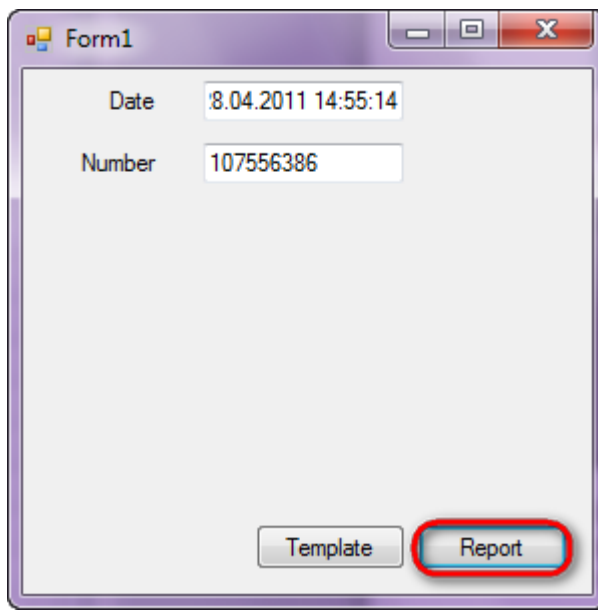

}

Step 19

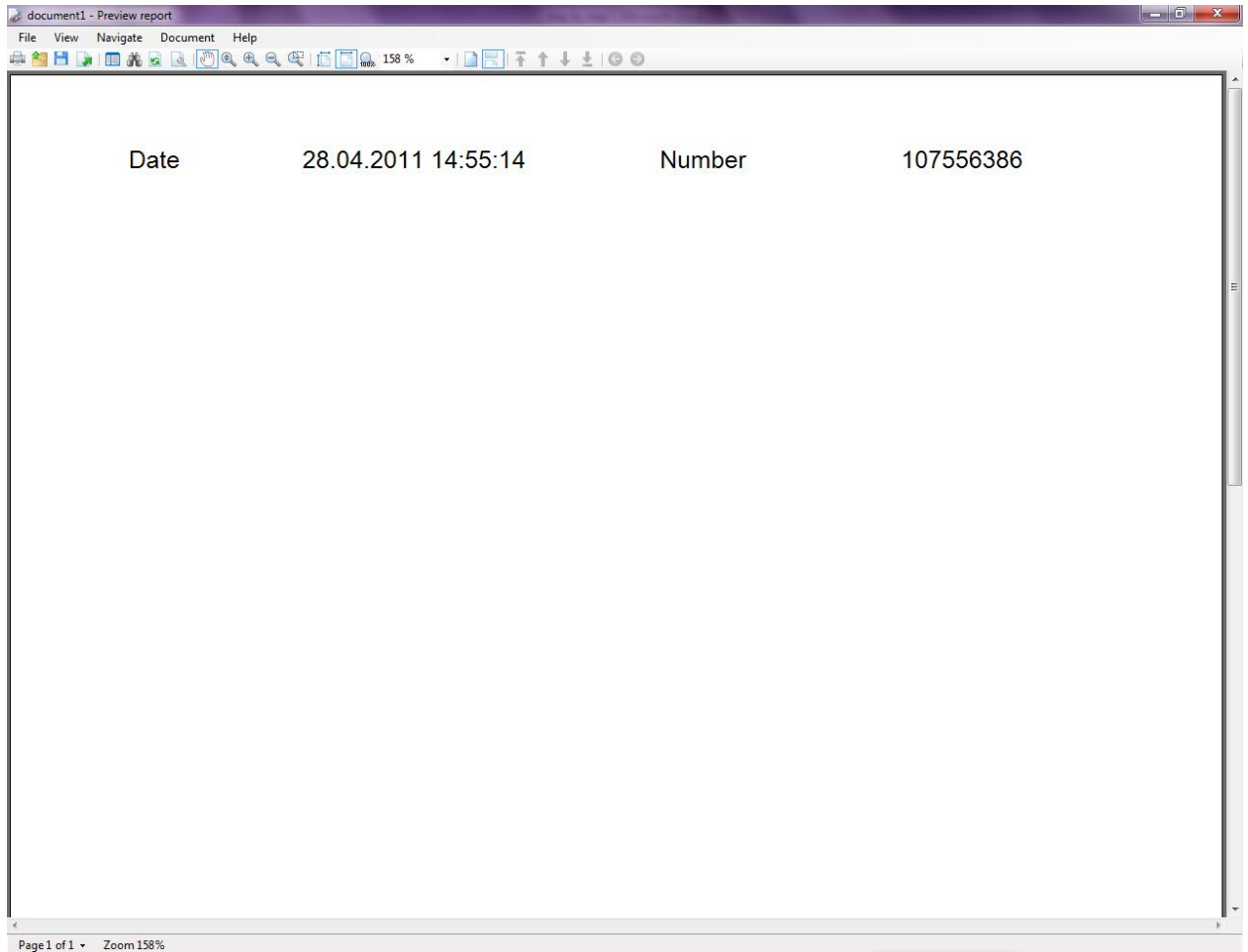
Click "Start Debugging" on the Visual Studio toolbar in order to start application.



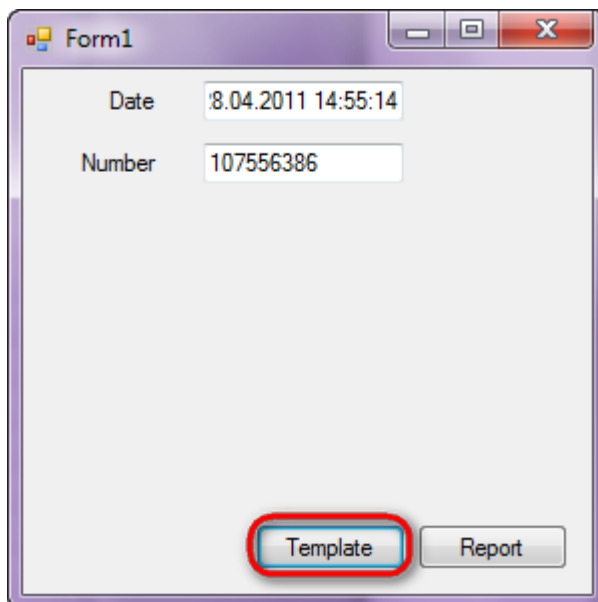
Application form appears.

A screenshot of a Windows application window titled 'Form1'. The window contains two text input fields: 'Date' with the value '8.04.2011 14:55:14' and 'Number' with the value '107556386'. At the bottom of the form, there are two buttons: 'Template' and 'Report'. The 'Report' button is circled in red.

Date and Number fields can be edited. When you click "Report" button, the report is generated and can be viewed in Report Viewer. Values from the fields are passed to the report as parameters.



To edit report template click "Template" on the application form.



Similar application sample is located in the following folder "\\Perpetuum Software\\Net ModelKit Suite\\ Samples\\Report Sharp-Shooter\\CSharp\\DocumentParametersUsing".

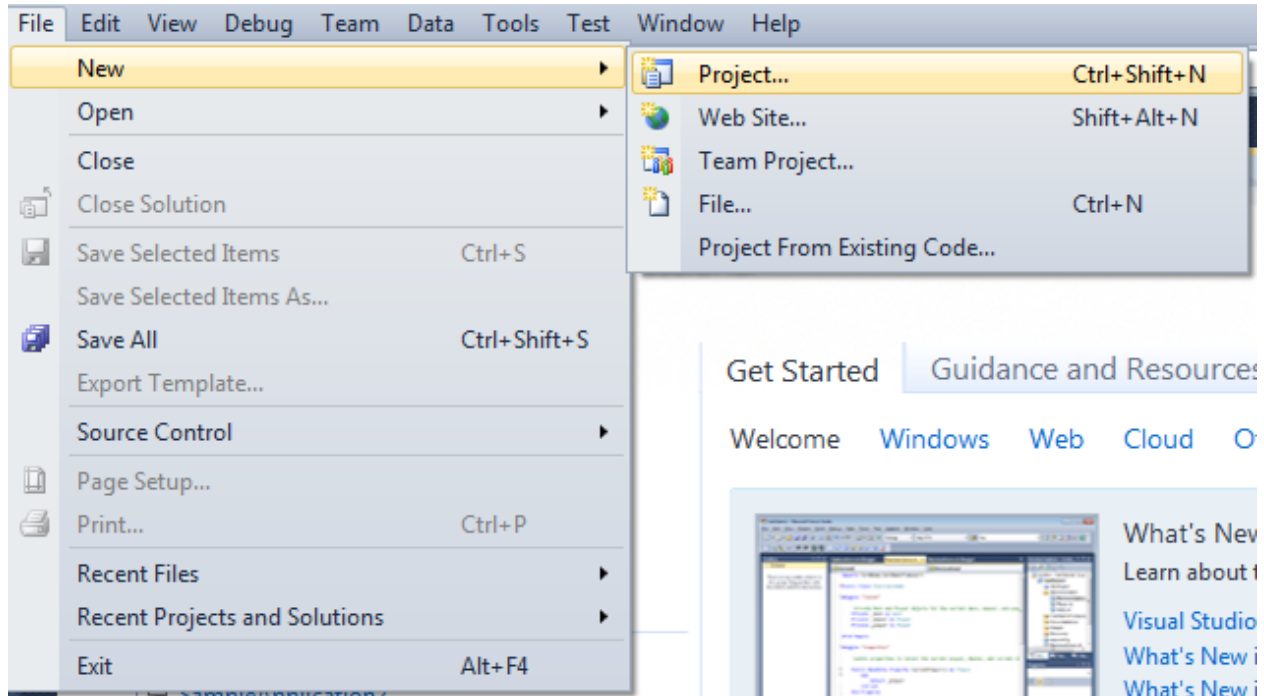


Multicolumn Report

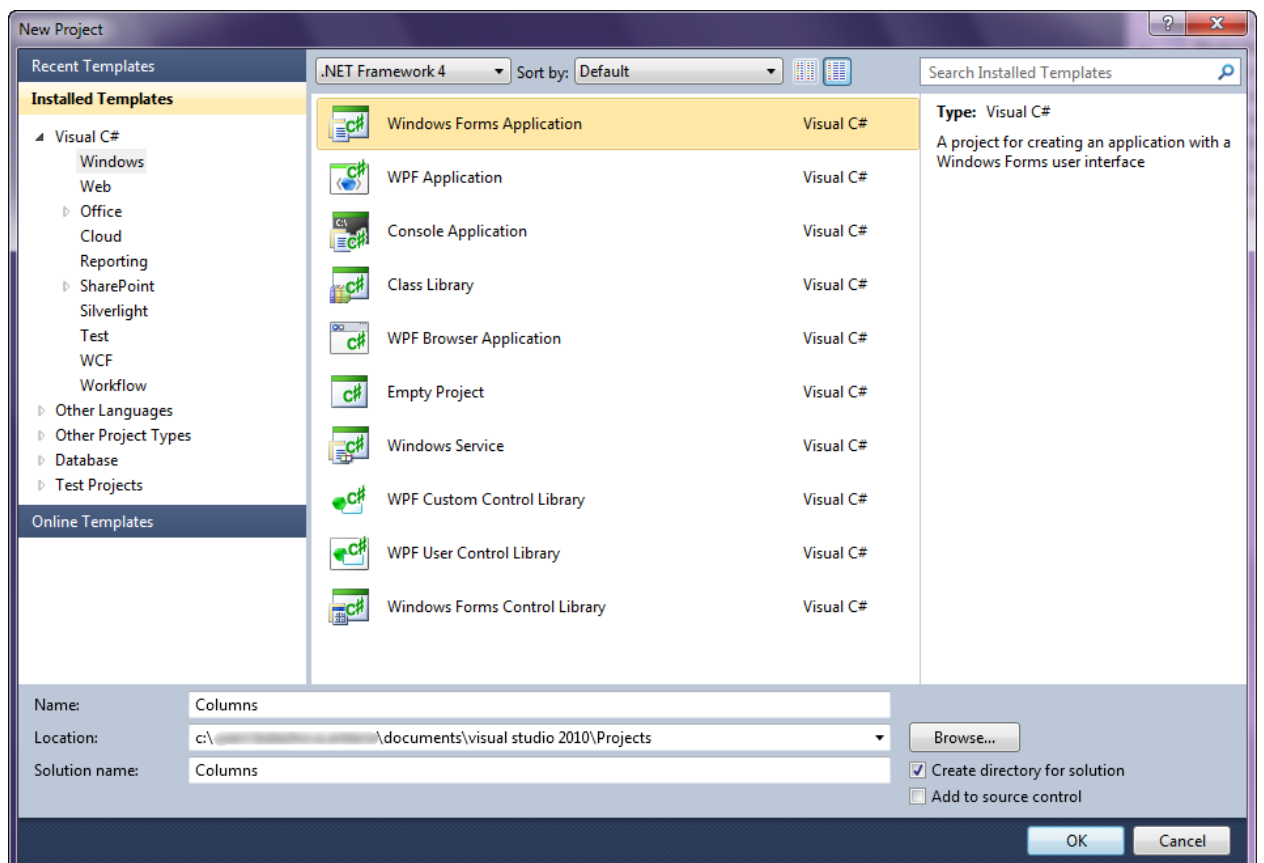
Template of a report containing a list of clients (company name and phone) in two columns.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

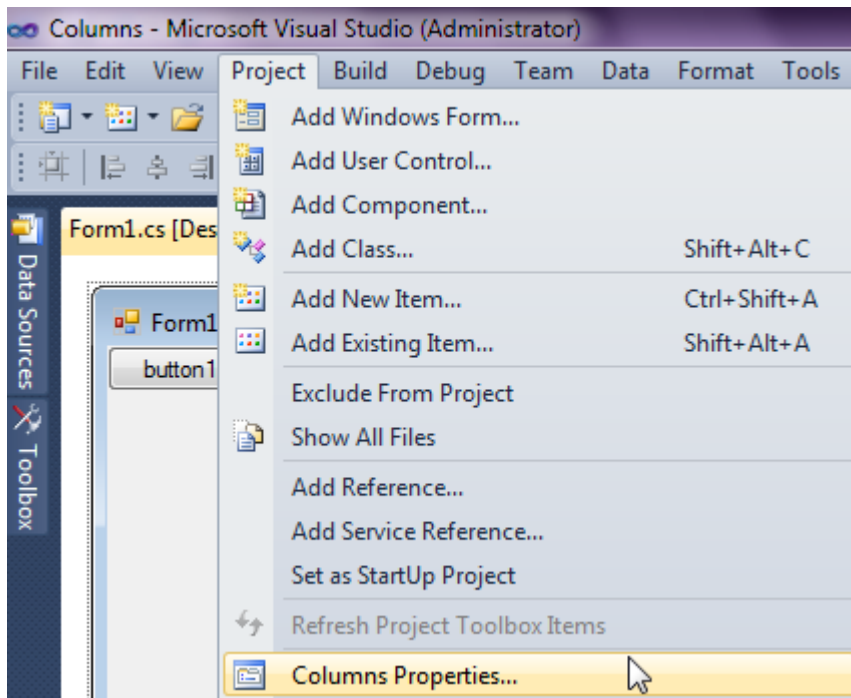


Select Windows Forms Application, set project name – “Columns”, set directory to save the project to.

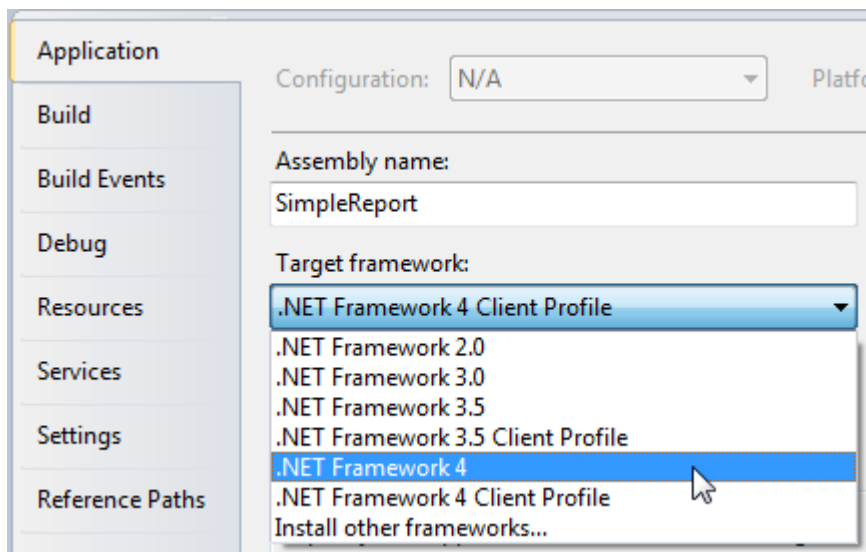


Step 2

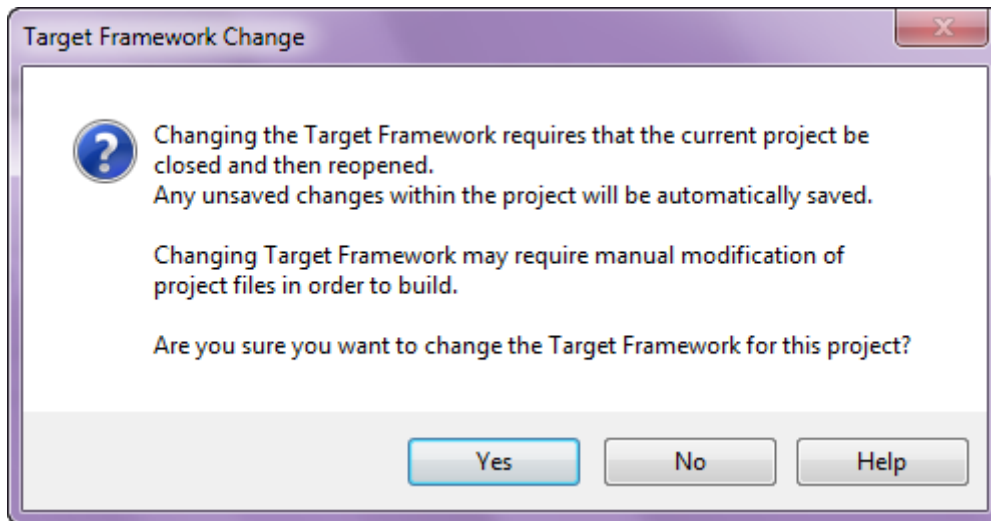
Change the project properties. Select the Project\Columns Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

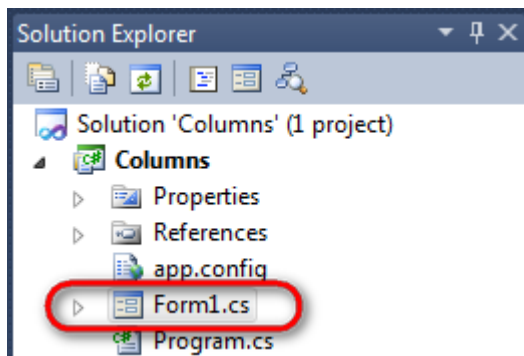


In the opened window press the "Yes" button.

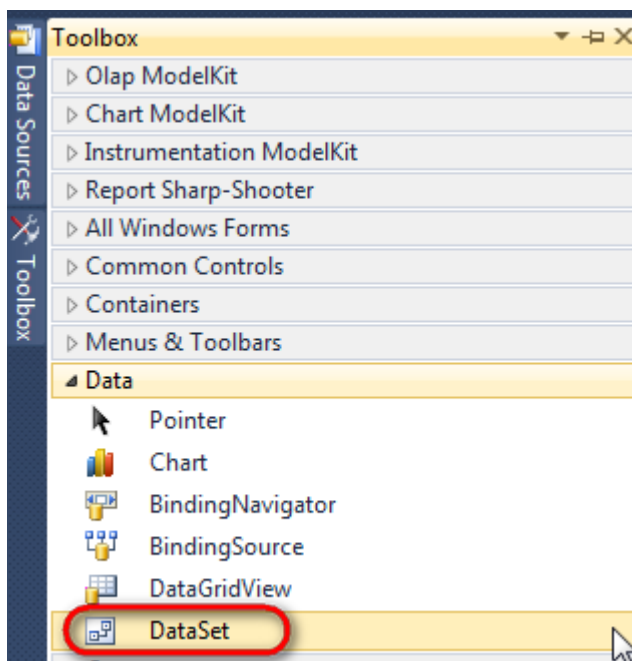


Step 3

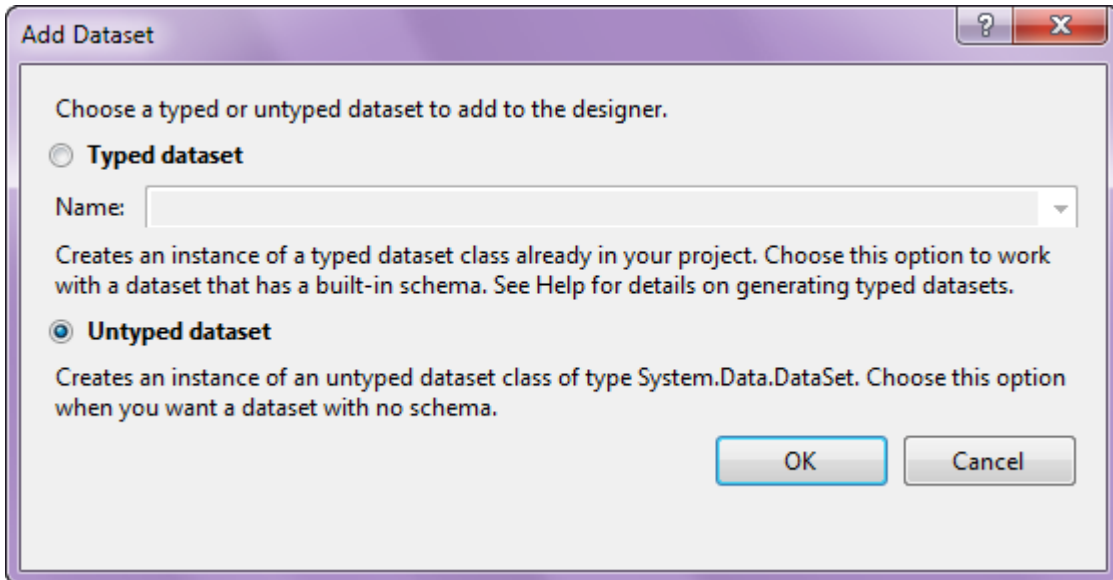
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



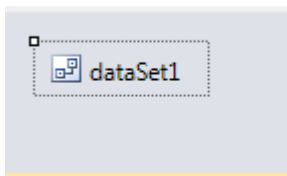
Click "DataSet" element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click «OK»

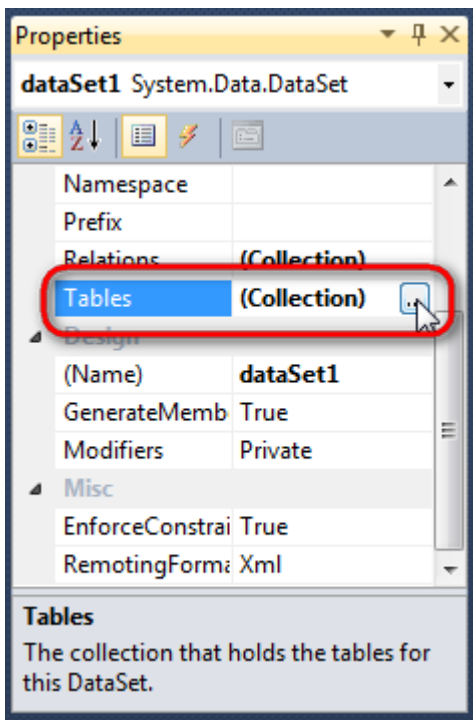


The component is available in the lower part of the window.

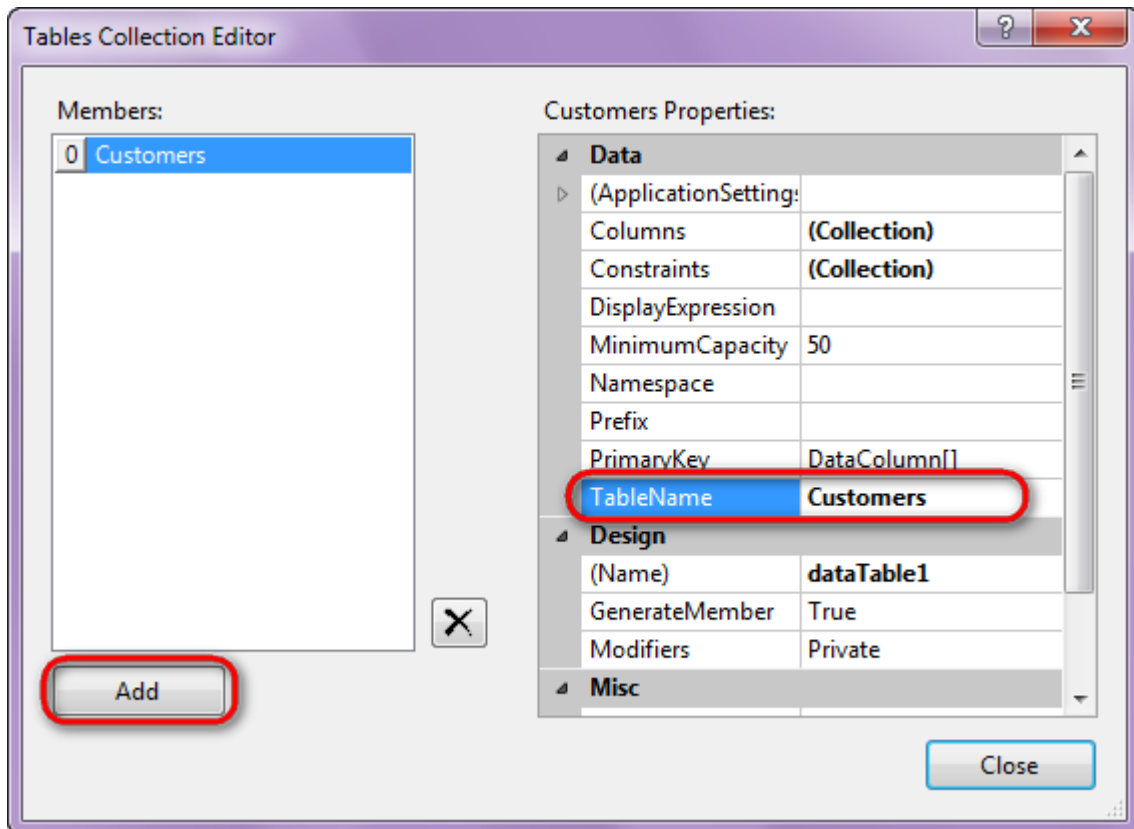


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.

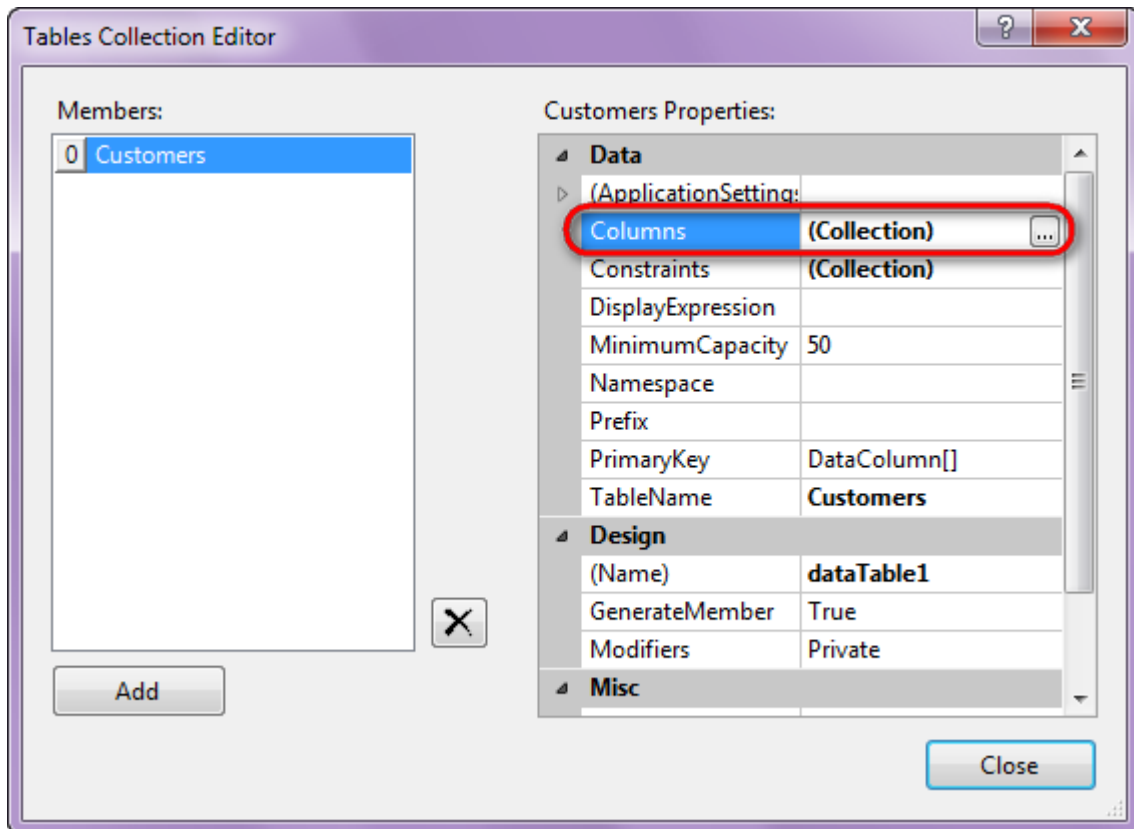


Click "Add" in order to add table. Set property TableName = Customers.

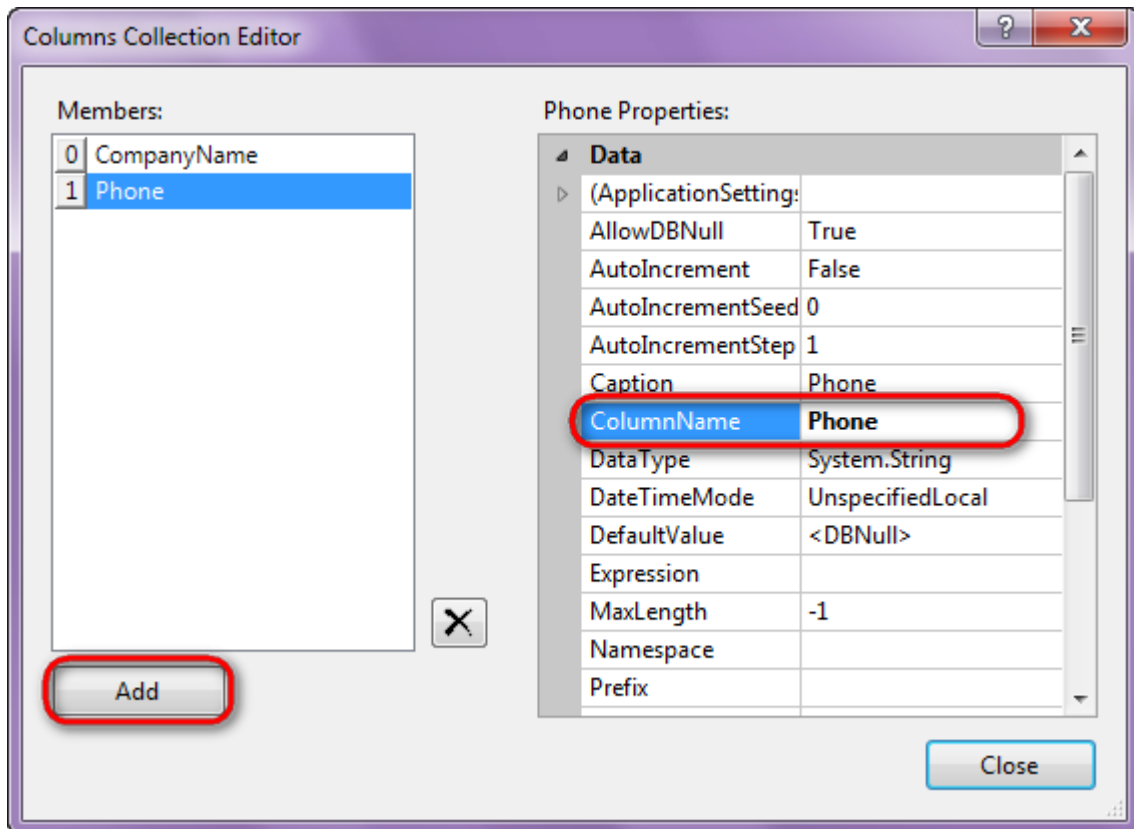


Step 5

Select Columns property, click button  in order to open property editor.

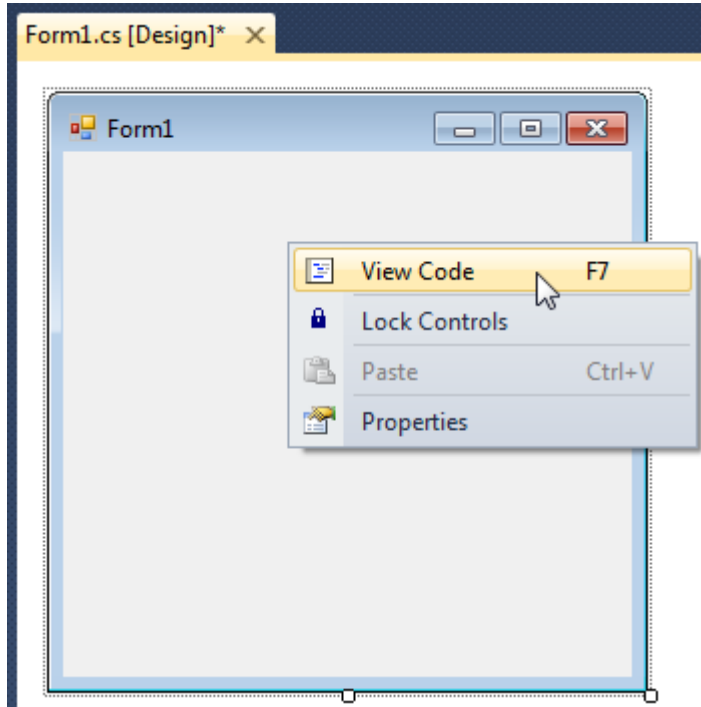


Click "Add" to add a new column. Add two columns. Set ColumnName property to "CompanyName", "Phone" correspondingly.



Step 6

Right click on the form and select "View Code" in the context menu to view code.



Add the following code to the class constructor in order to fill data source.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
}
```

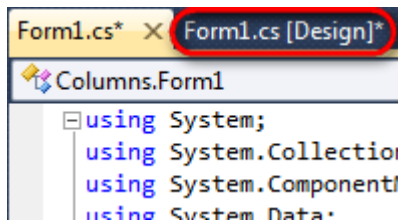


```
row["Phone"] = "030-0074321";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ana Trujillo Emparedados y helados";  
row["Phone"] = "(5) 555-4729";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Ernst Handel";  
row["Phone"] = "7675-3425";  
dataTable1.Rows.Add(row);  
row = dataTable1.NewRow();  
row["CompanyName"] = "Toms Spezialitäten";  
row["Phone"] = "0251-031259";  
dataTable1.Rows.Add(row);  
}
```

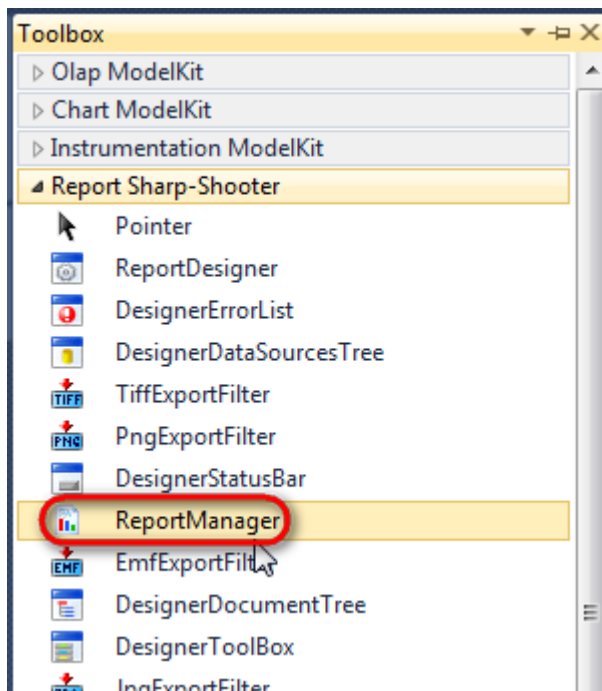
To see how the list is divided into two columns add more data.

Step 7

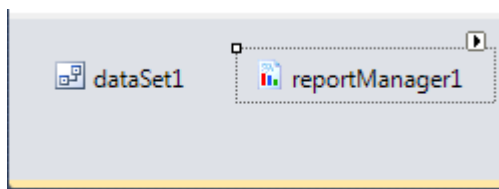
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

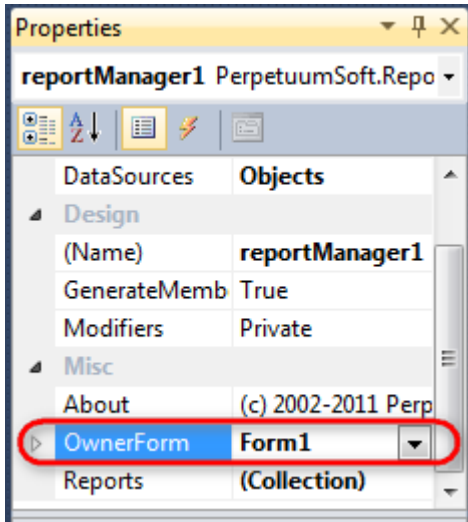


The component is available in the lower part of the window.



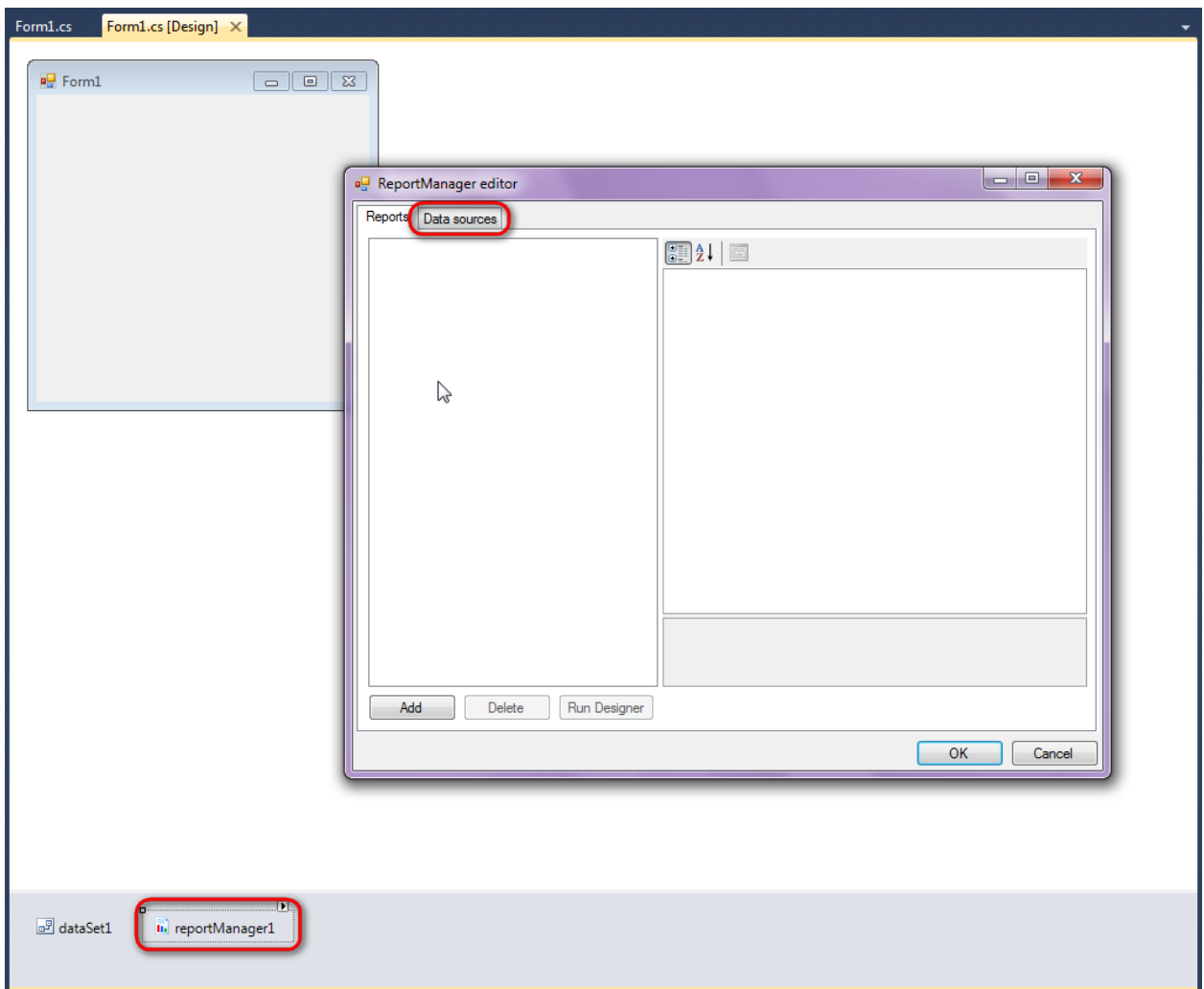
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

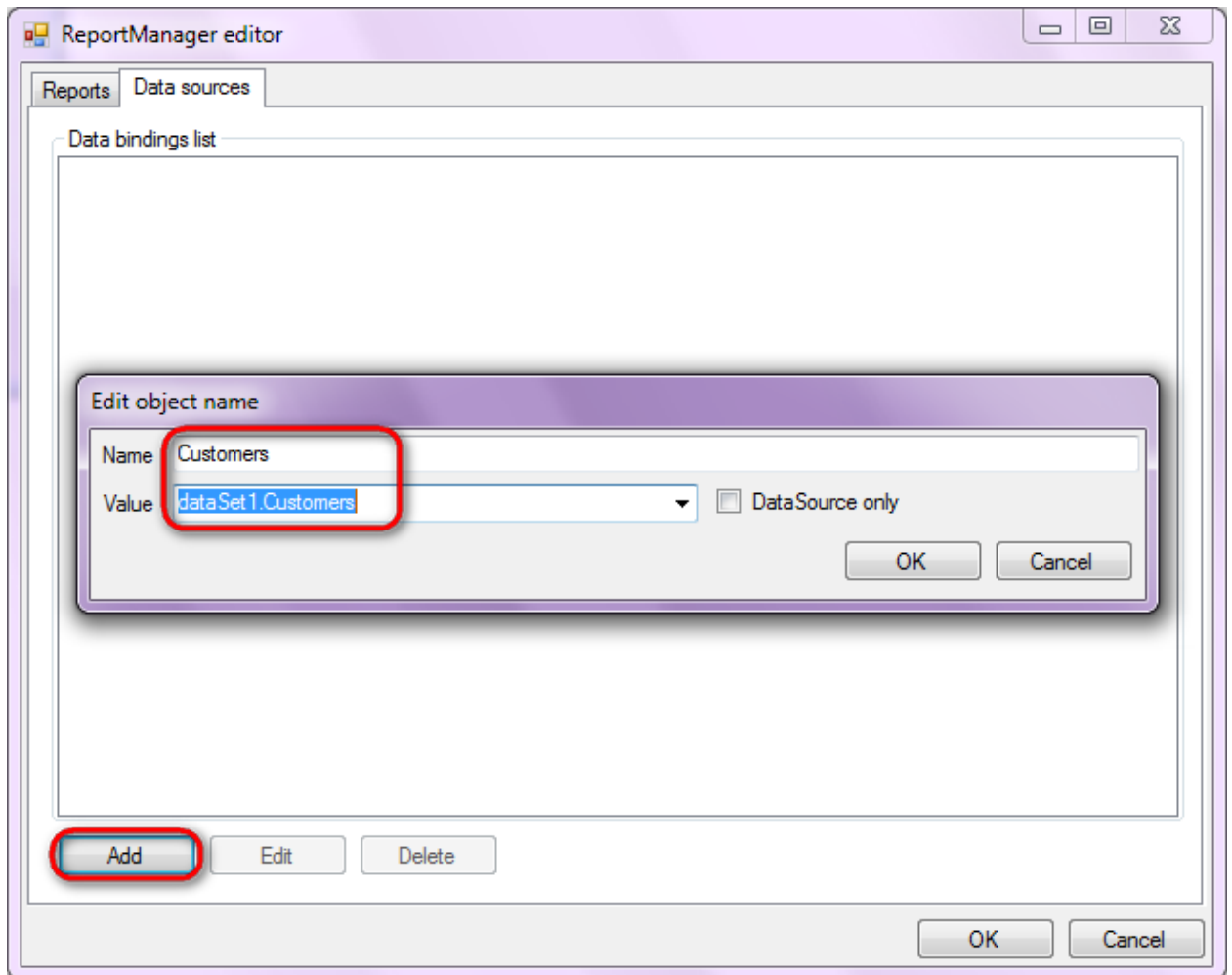


Step 9

Double click on ReportManager to open ReportManager editor.

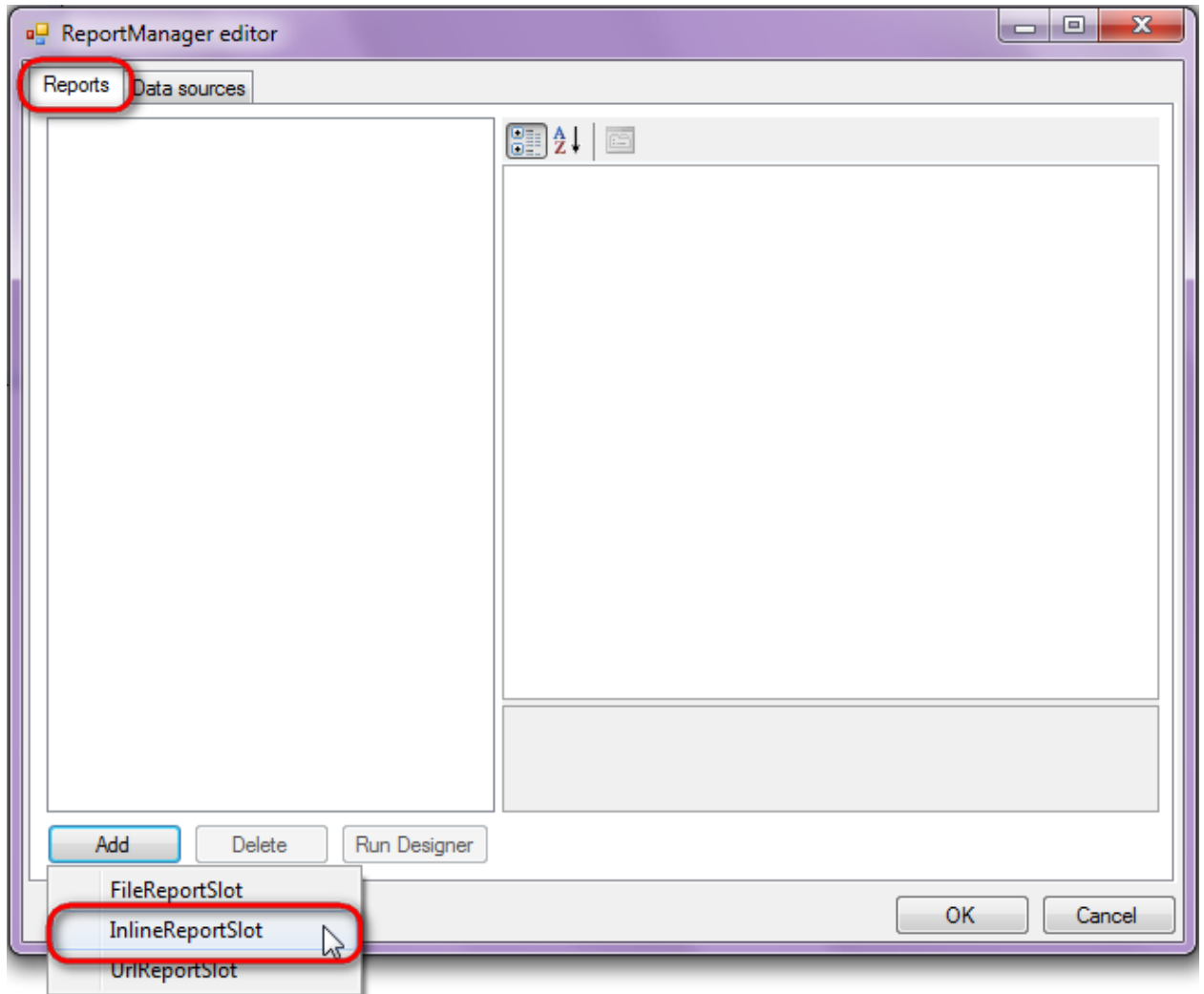


Go to "Data sources" tab, click "Add", set Data source name – "Customers", select data source Value – "dataSet1.Customers".



Step 10

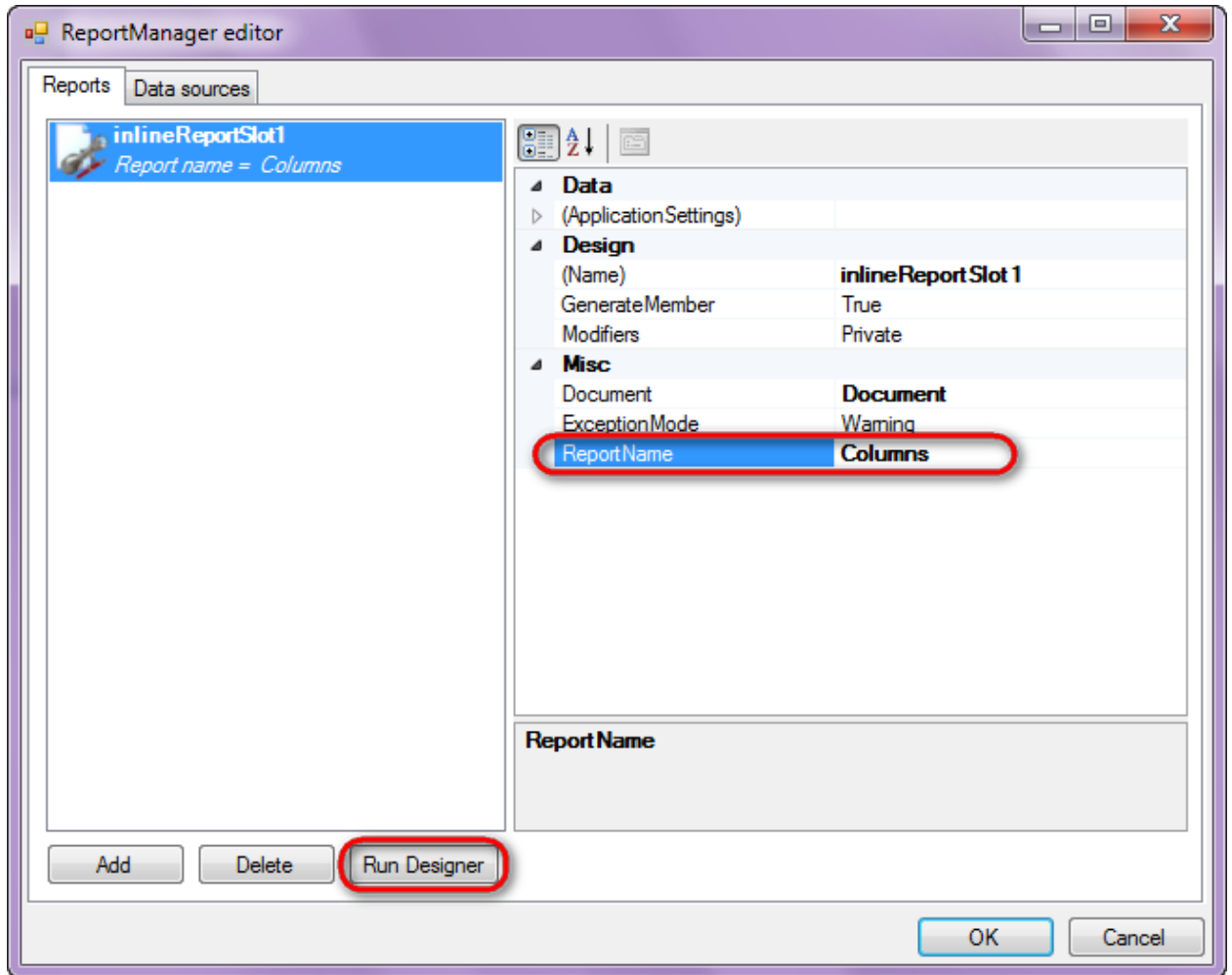
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

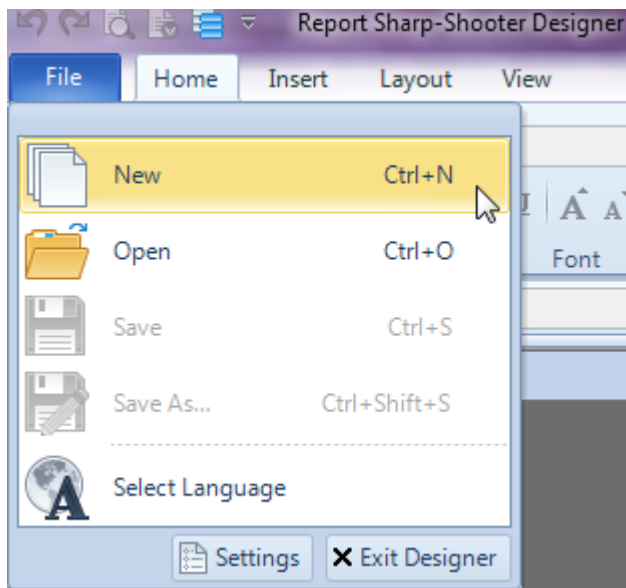
Set name of the report in the property ReportName – “Columns”.

Click “Run Designer” in order to open template editor – Report Designer.

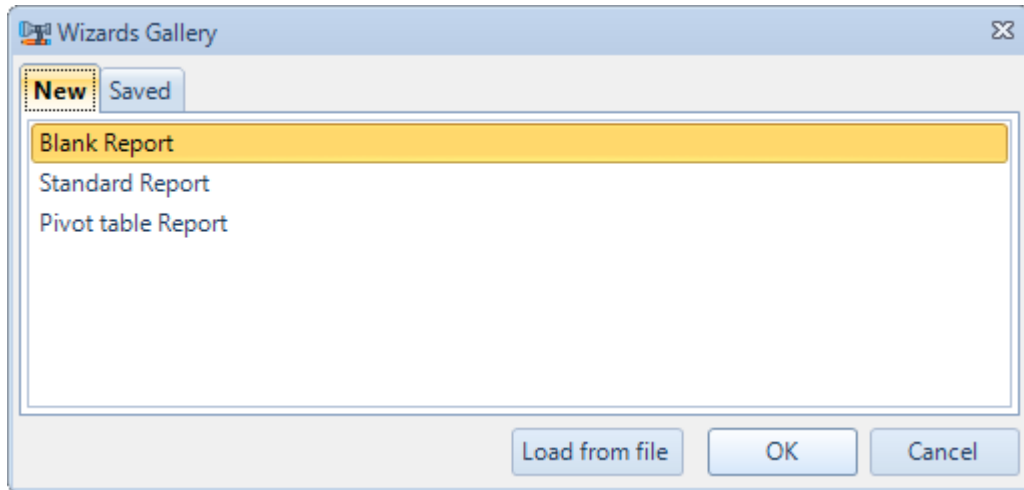


Step 12

Create new empty template – select File\New from the main menu.

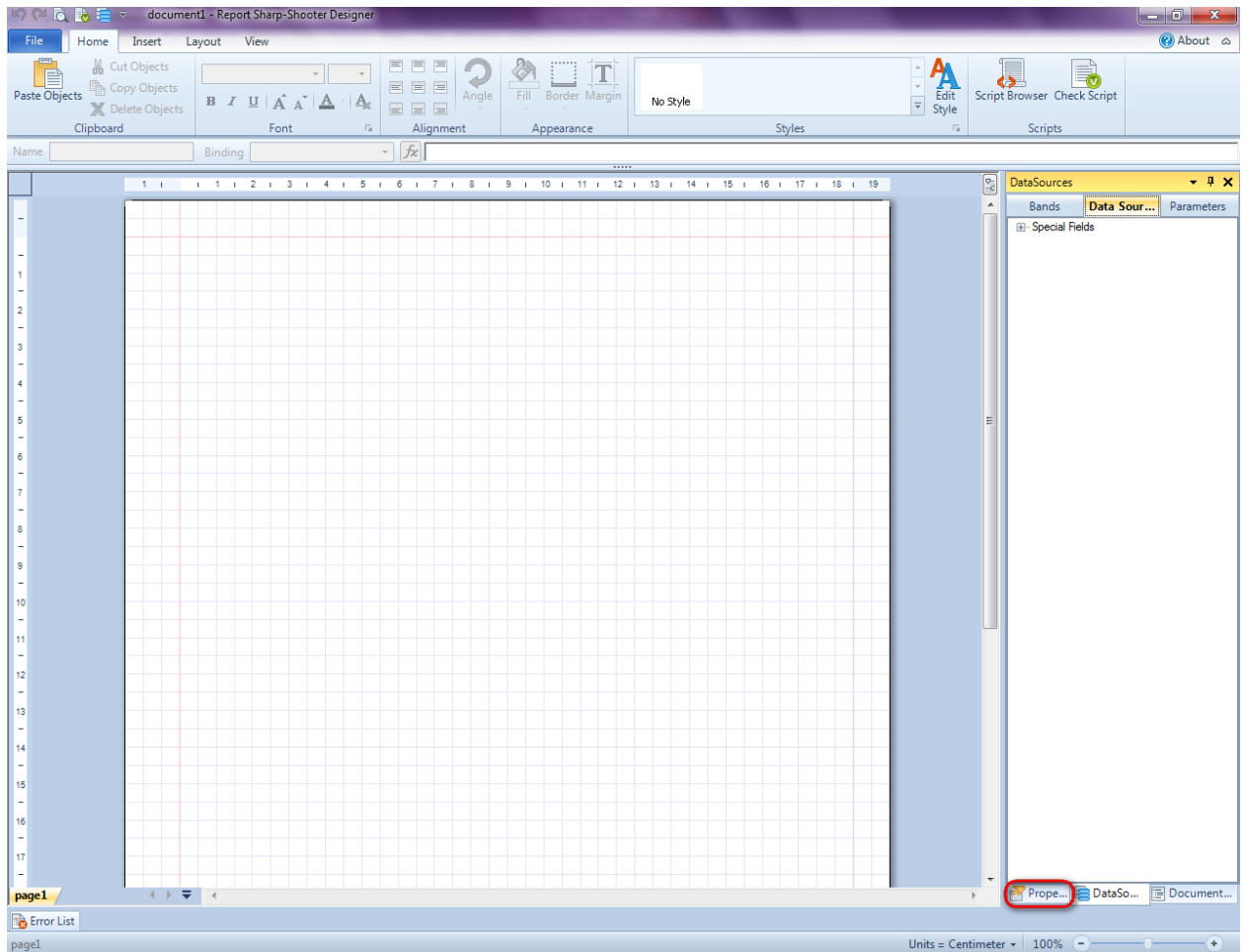


Select "Blank Report" in the Wizards Gallery and click "OK".

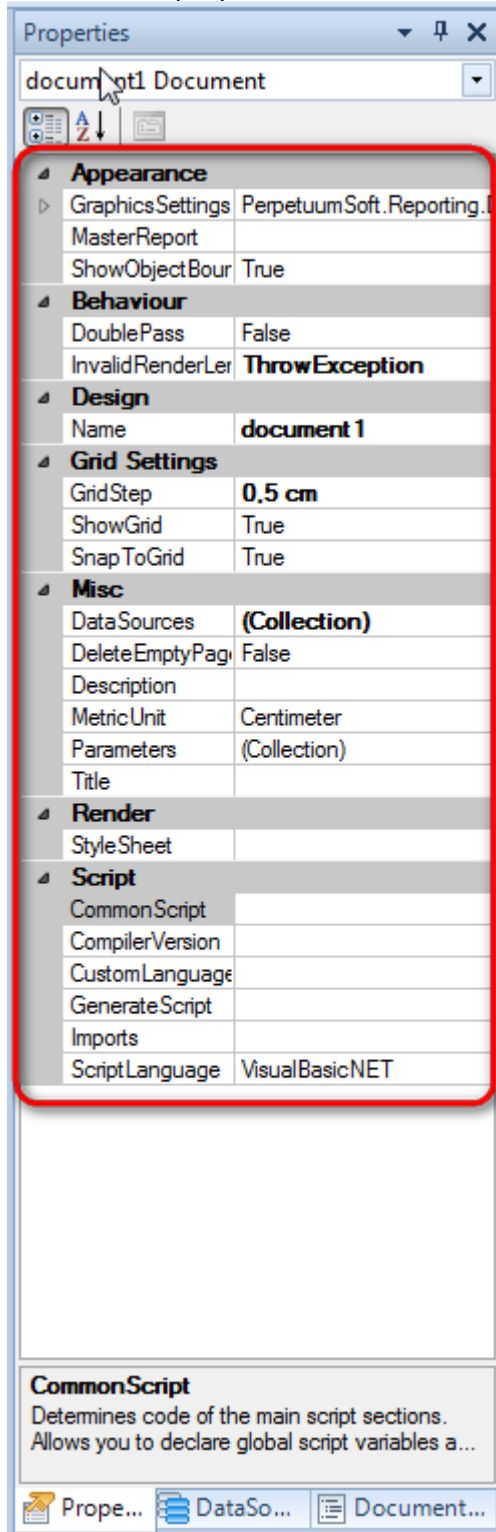


Step 13

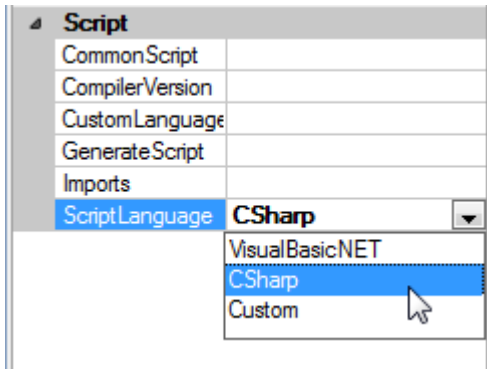
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

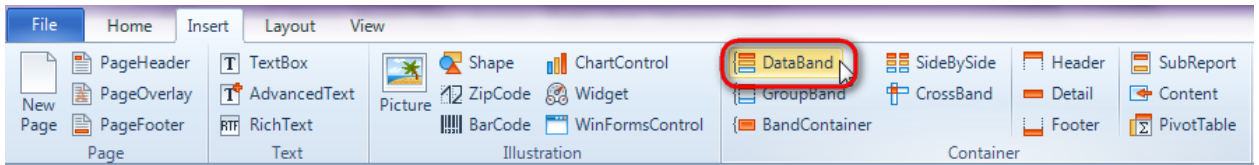


Set property ScriptLanguage = CSharp.



Step 14

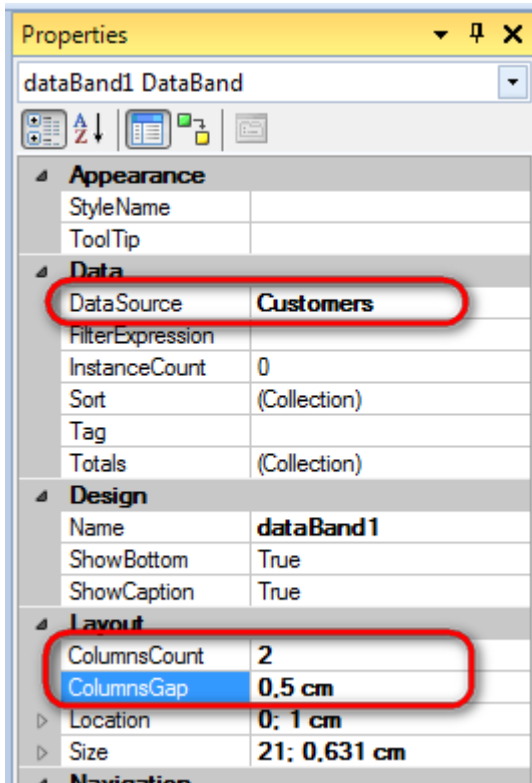
Press "DataBand" button on the Insert tab in the group Container.



Click on the template area to add DataBand section to the template.

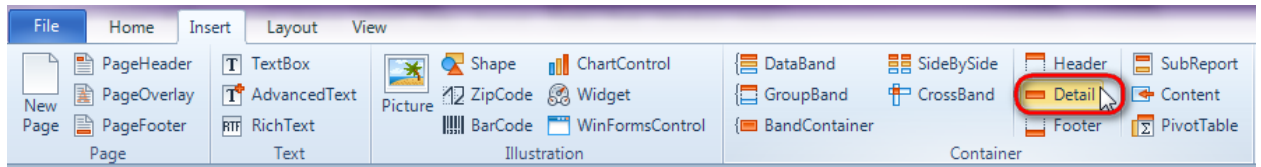
Set data source in the property DataSource = Customers.

Set the following properties: ColumnsCount = 2, ColumnsGap = 0,5 cm.



Step 15

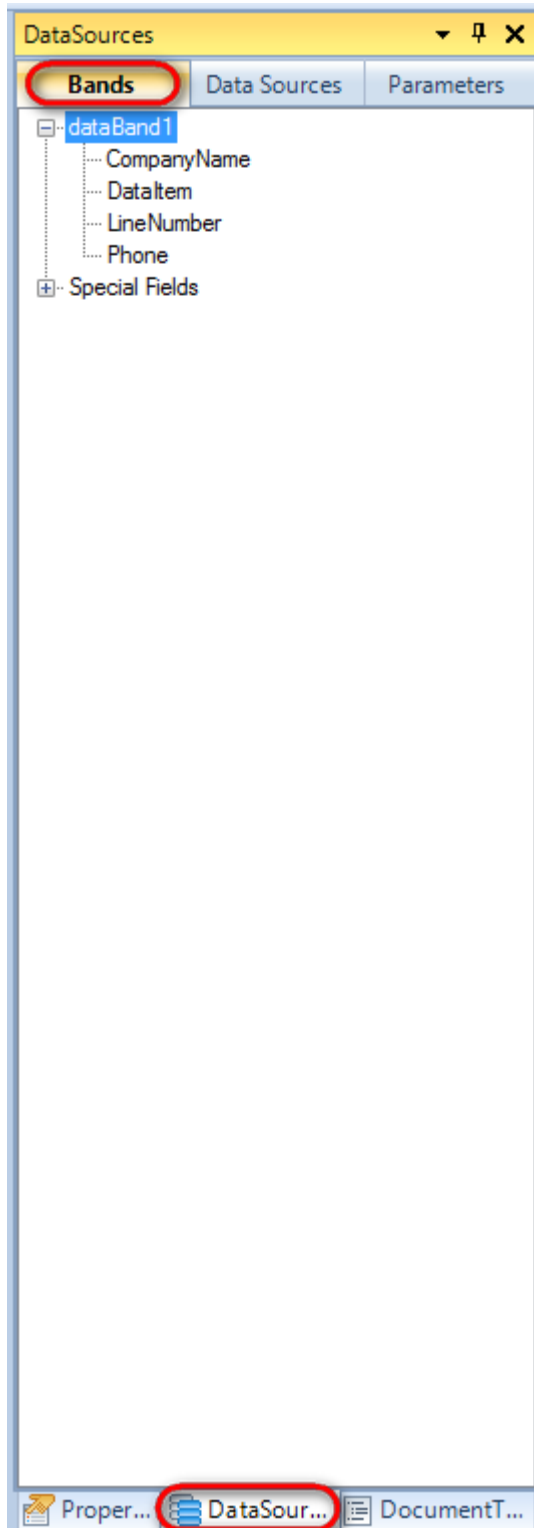
Press "Detail" button on the Insert tab in the group Container.



Click on the DataBand area to add Detail band inside DataBand.

Step 16

Go to "DataSources" tab.





Drag and drop "CompanyName", "Phone" fields from the dataBand1 tree to the detail1 band. As a result TextBoxes are created. Value property is automatically filled with script loading data from the data source.

Change size of the elements and locate them so that they don't exceed the red line determining column border.

dataBand1:DataBand DataSource = Customers	
detail1:Detail	
<dataBand1 ["CompanyName"]>	<dataBand1 ["Phone"]>
end of dataBand1	

Step 17

Save template, close Report Designer.

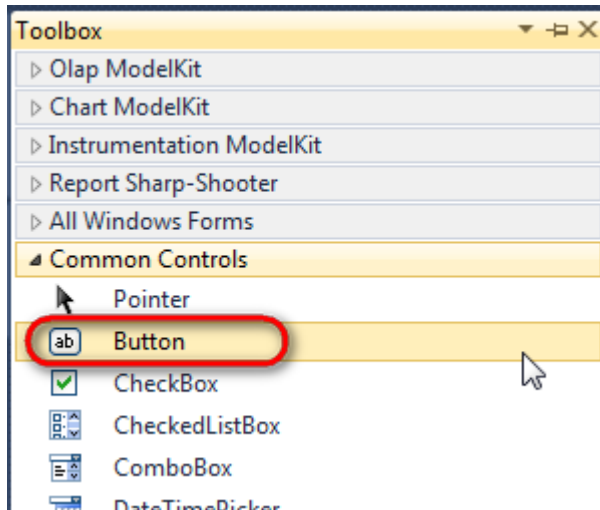
Step 18

Add code to display report to the class constructor. Write RenderCompleted event handler of the InlineReportSlot object.

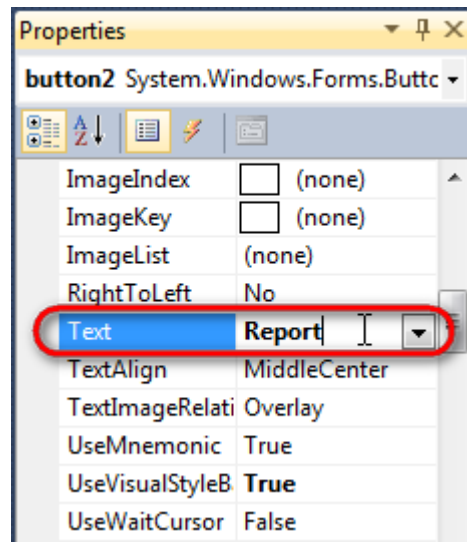
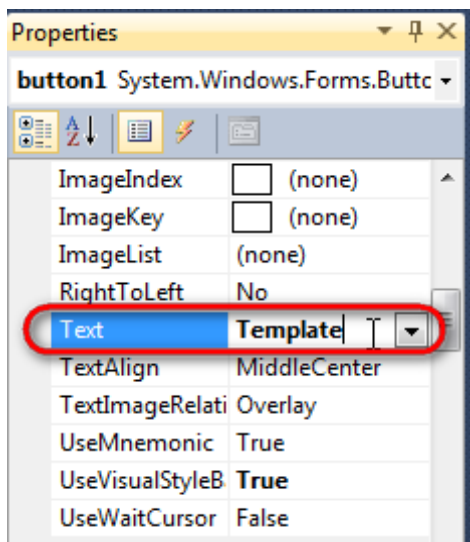
```
public Form1 ()
{
    InitializeComponent();
    DataRow row = dataTable1.NewRow();
    row["CompanyName"] = "Alfreds Futterkiste";
    row["Phone"] = "030-0074321";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ana Trujillo Emparedados y helados";
    row["Phone"] = "(5) 555-4729";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Ernst Handel";
    row["Phone"] = "7675-3425";
    dataTable1.Rows.Add(row);
    row = dataTable1.NewRow();
    row["CompanyName"] = "Toms Spezialitäten";
    row["Phone"] = "0251-031259";
    dataTable1.Rows.Add(row);
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog(this);
    }
}
```

Step 19

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



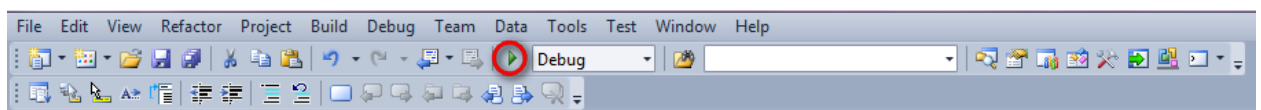
Create Click event handlers for the buttons – double click on the Button element on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

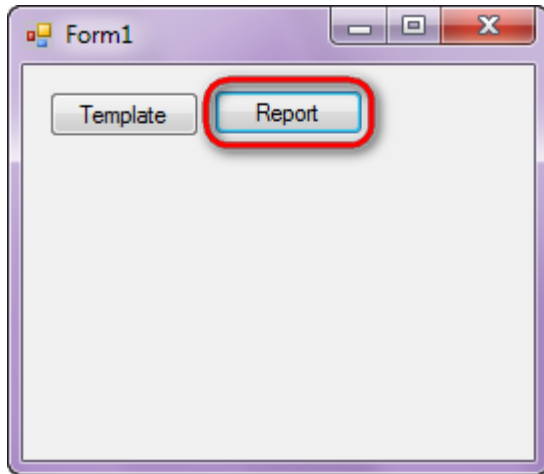
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 20

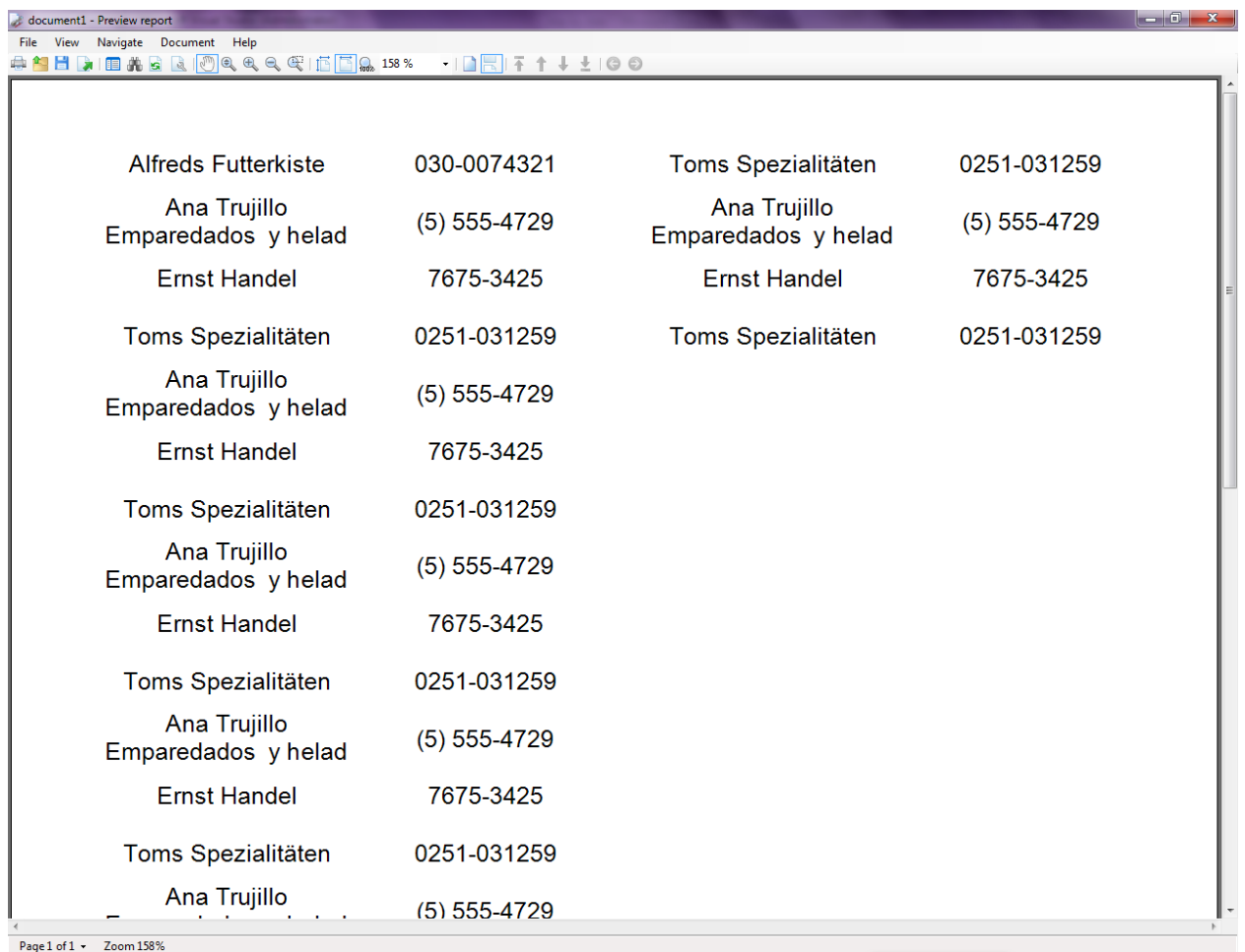
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



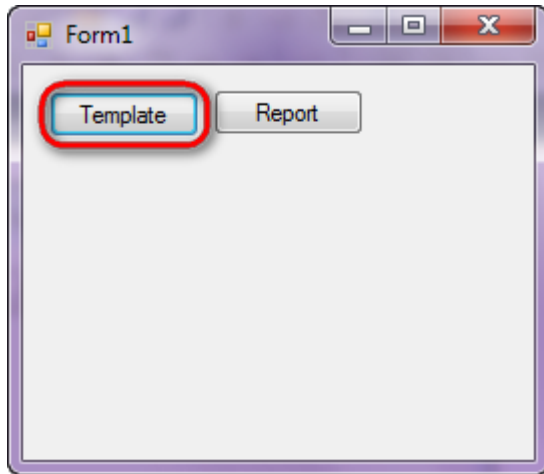
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Reports\Simple Reports\MultiColumn.

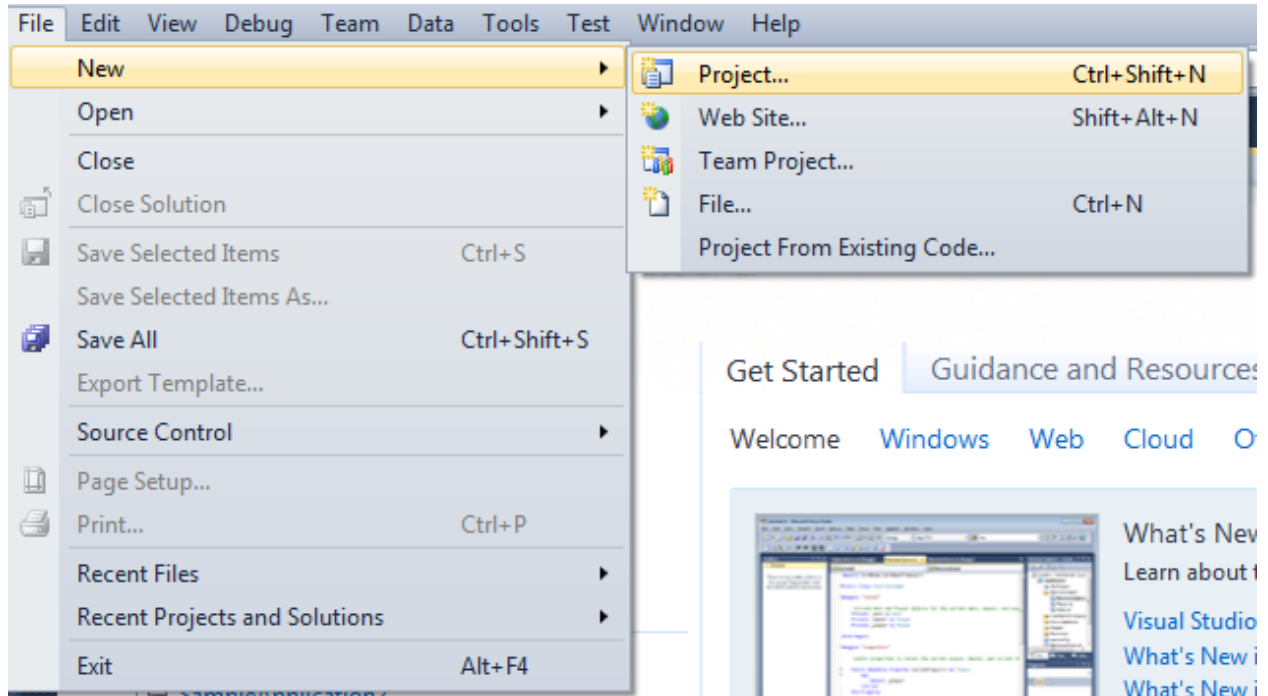


Horizontal List

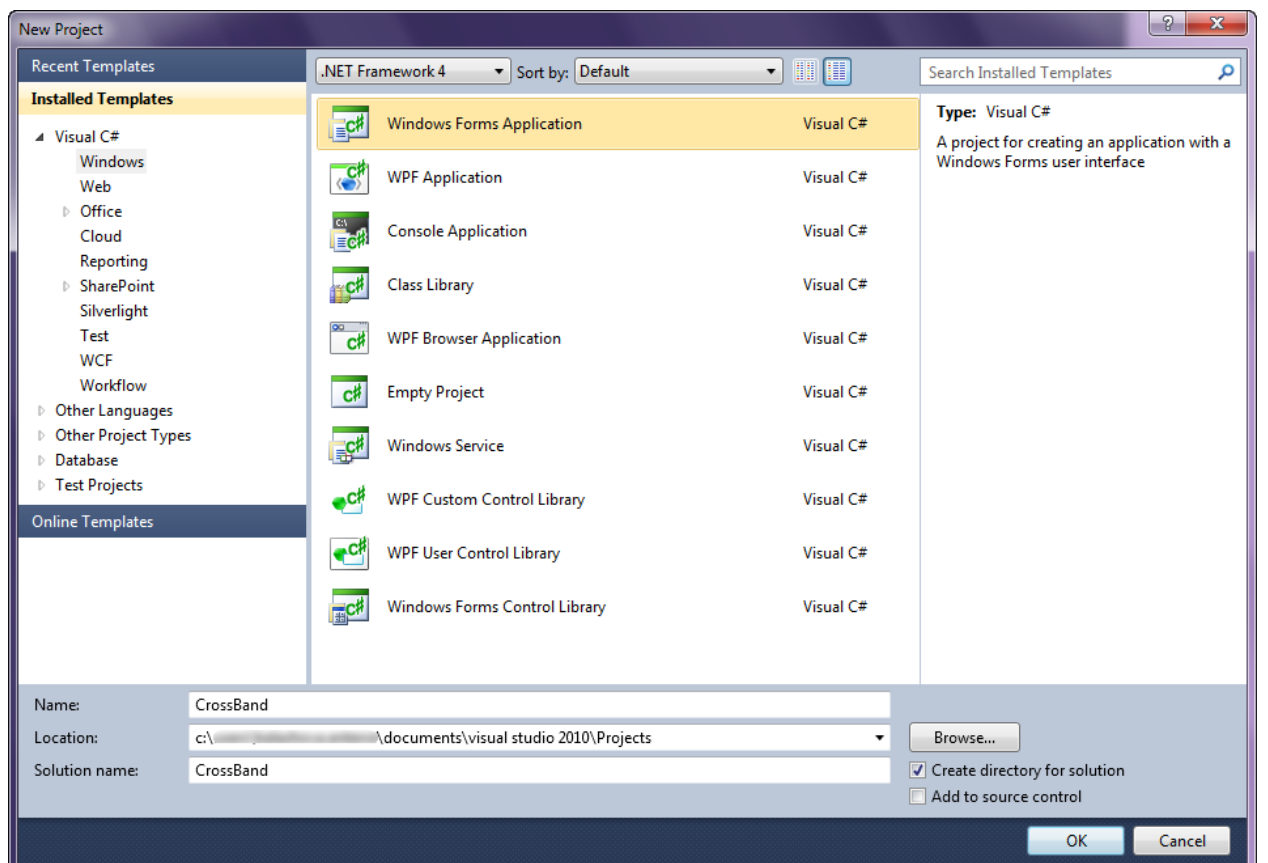
Template of a report containing positive whole numbers in a line.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

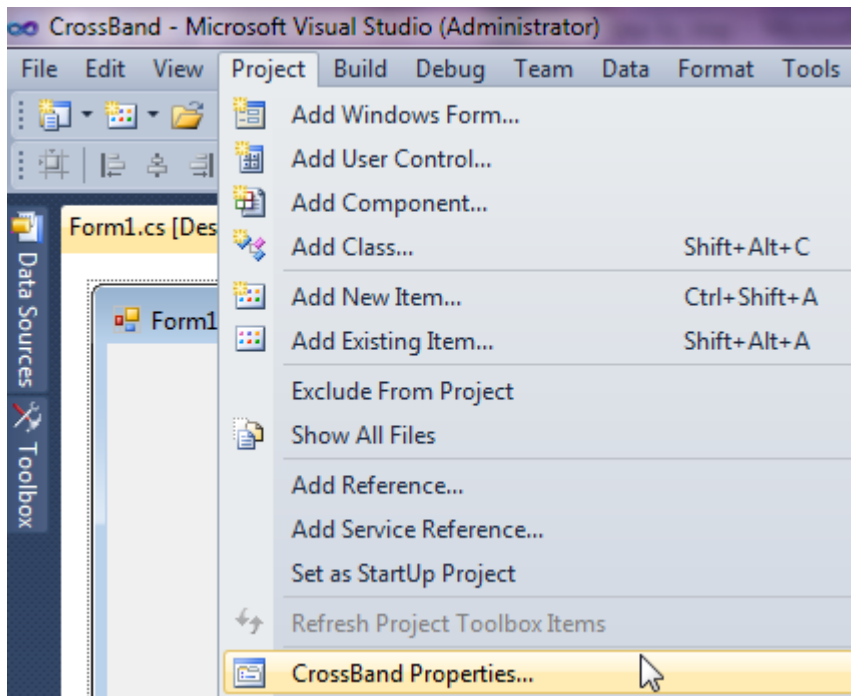


Select Windows Forms Application, set project name – “CrossBand”, set directory to save the project to.

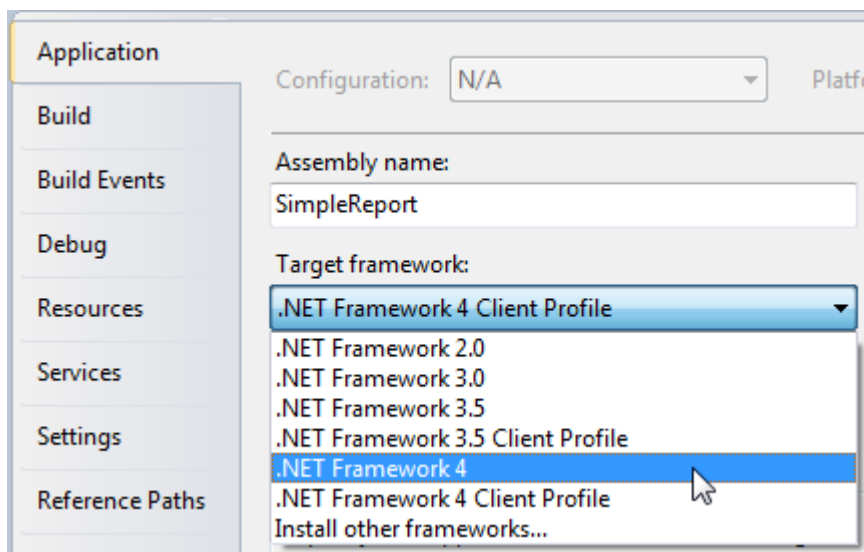


Step 2

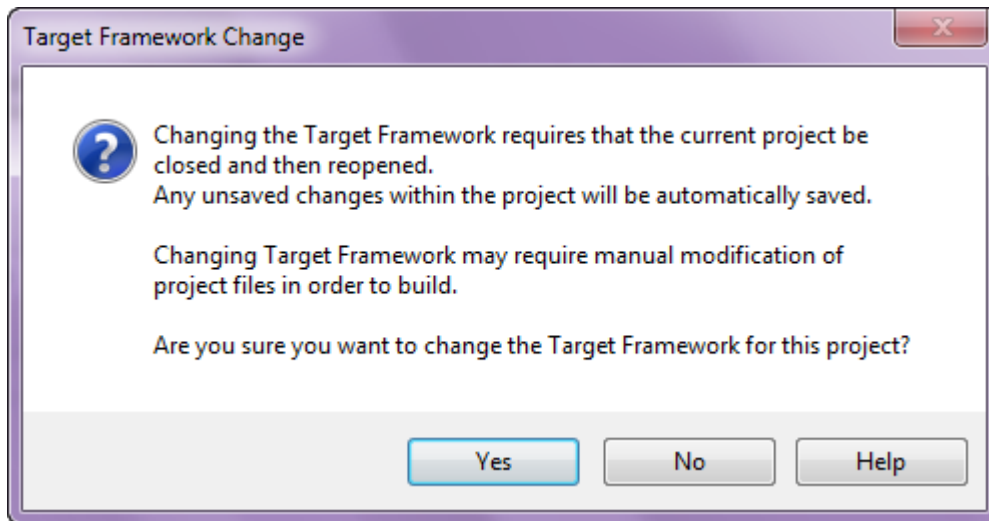
Change the project properties. Select the Project\CrossBand Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

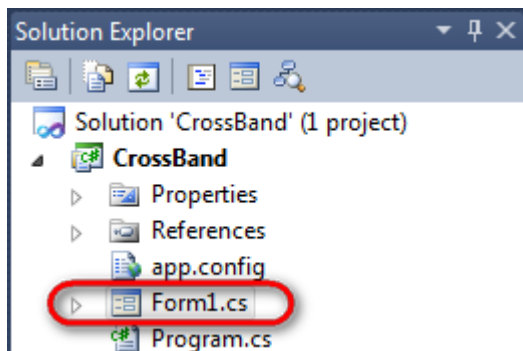


In the opened window press the "Yes" button.

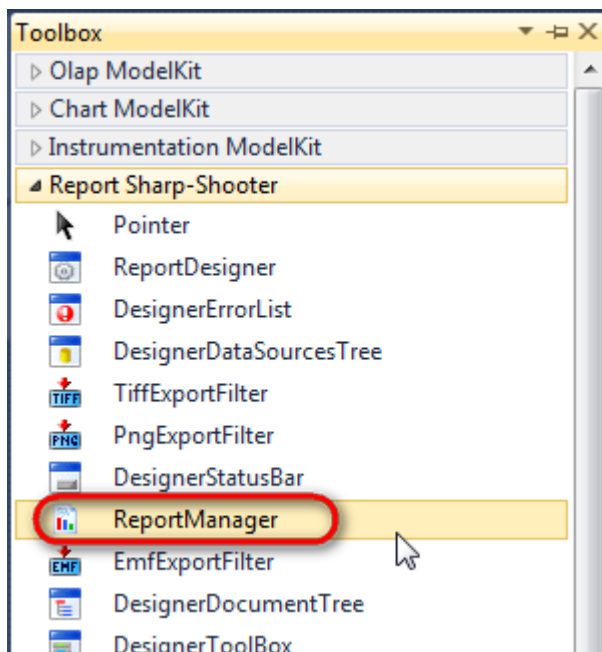


Step 3

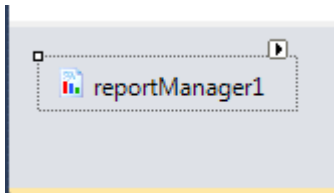
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

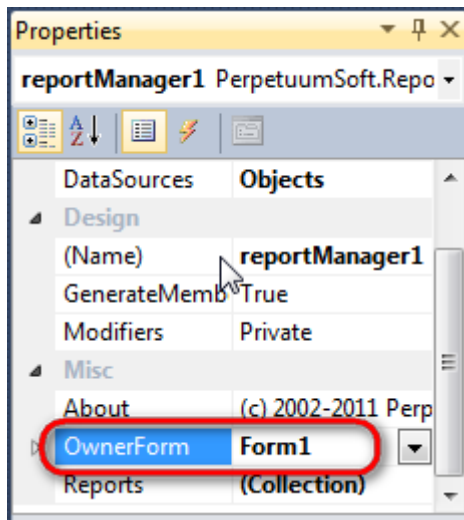


The component is available in the lower part of the window.



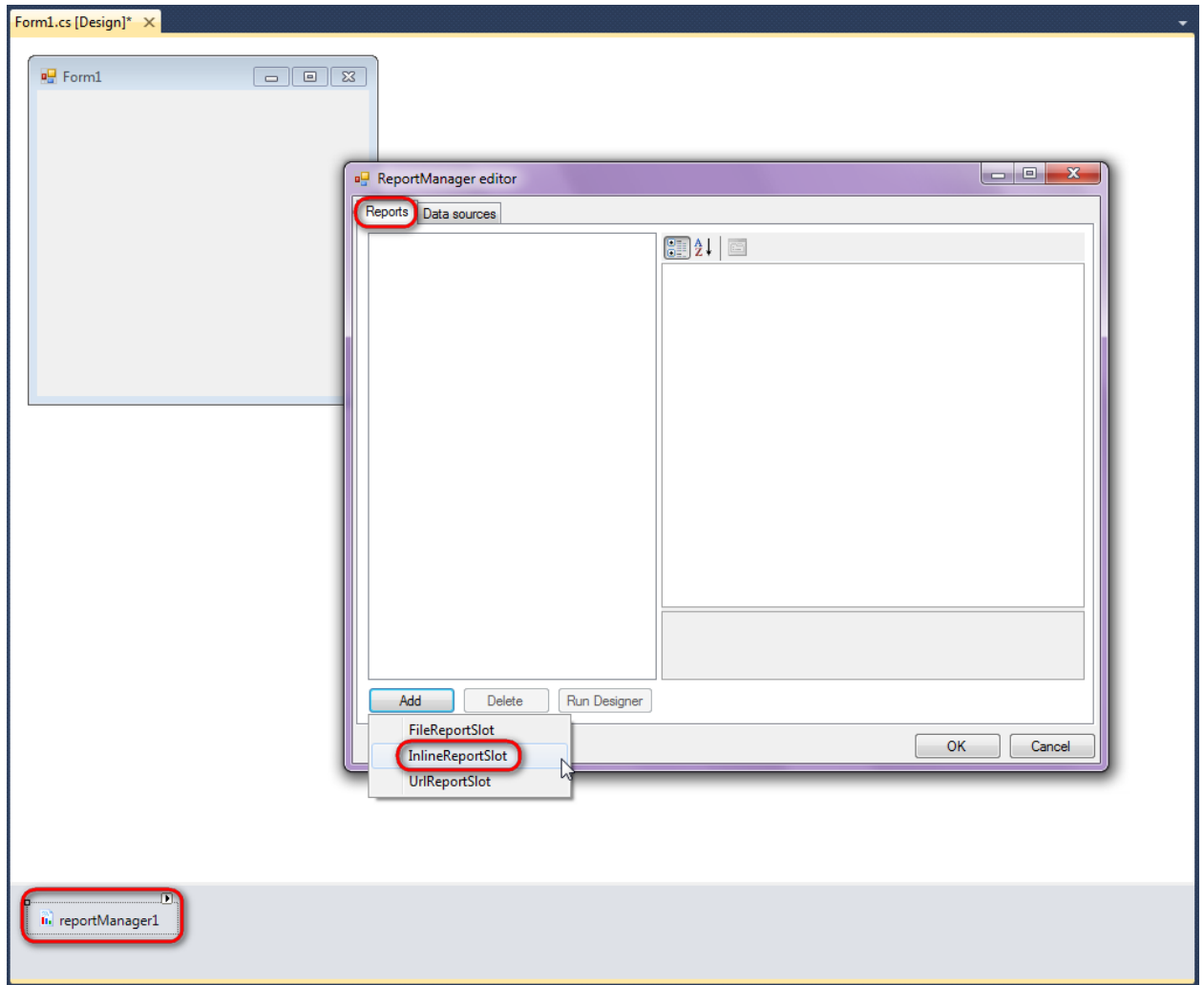
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

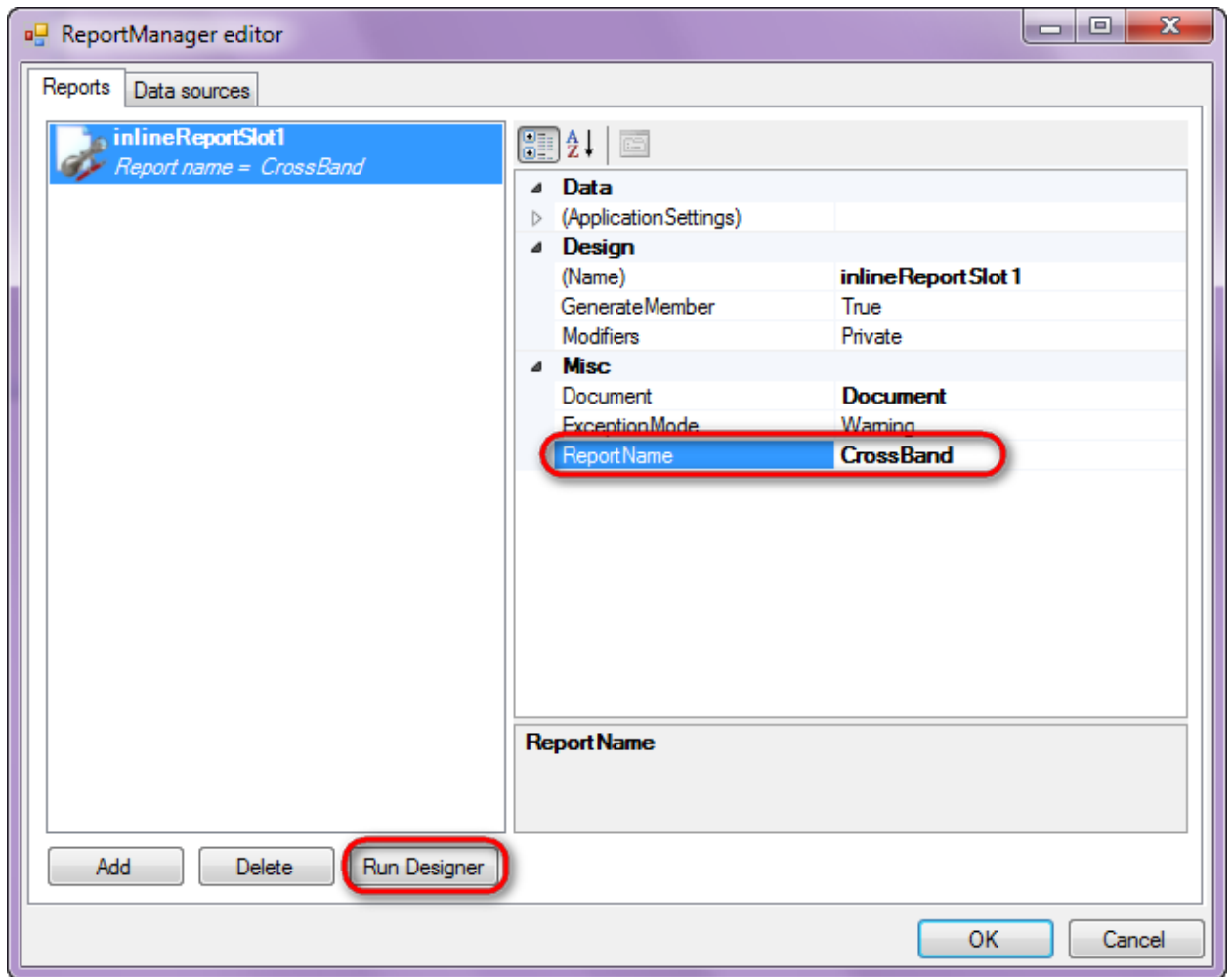


On the "Reports tab", click "Add" and select "InlineReportSlot".

Step 6

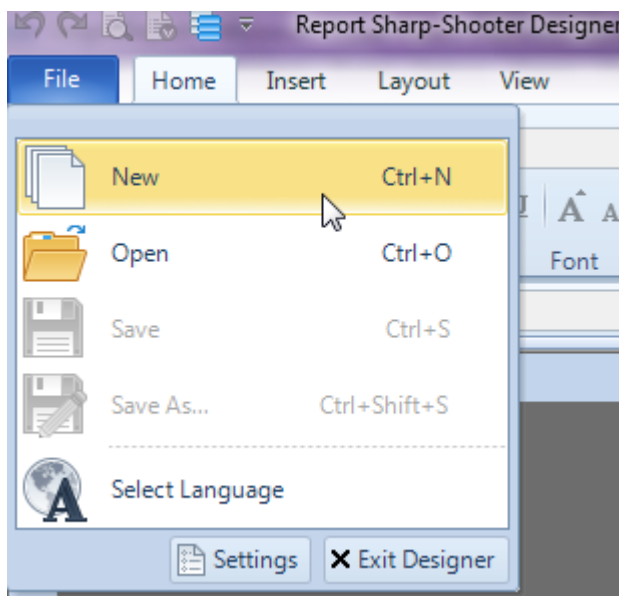
Set name of the report in the property ReportName – "CrossBand".

Click "Run Designer" in order to open template editor - Report Designer.

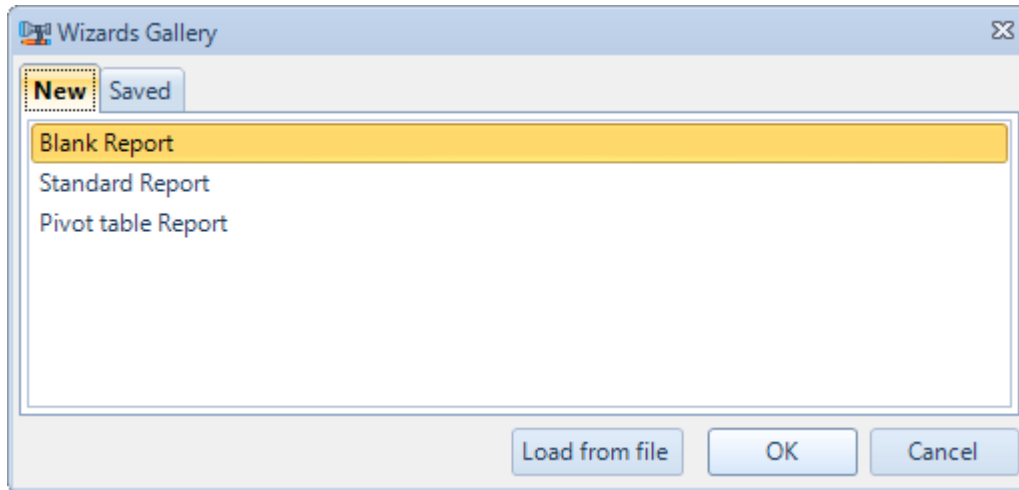


Step 7

Create new empty template – select item File\New from the main menu.

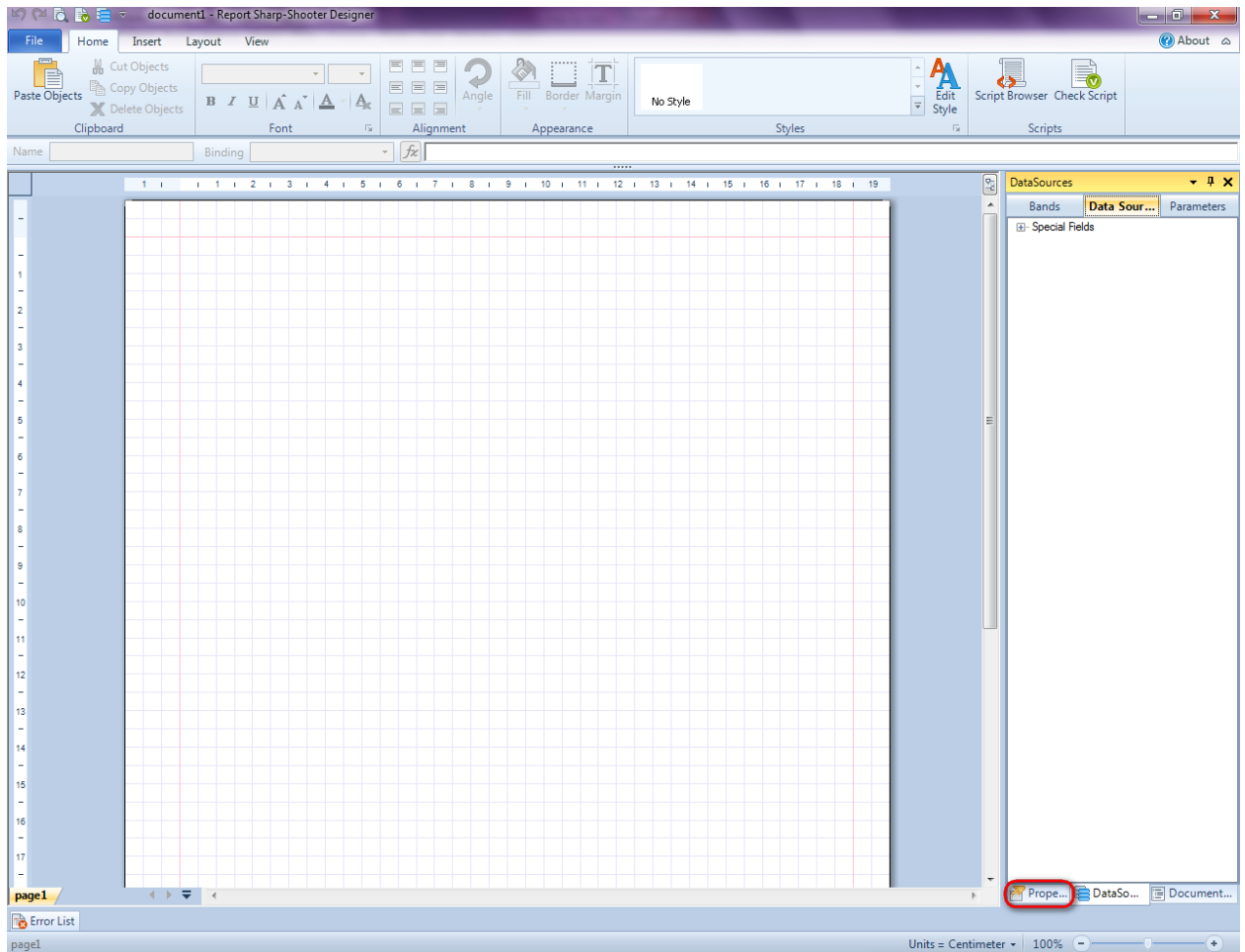


Select "Blank Report" in the Wizards Gallery and click "OK".



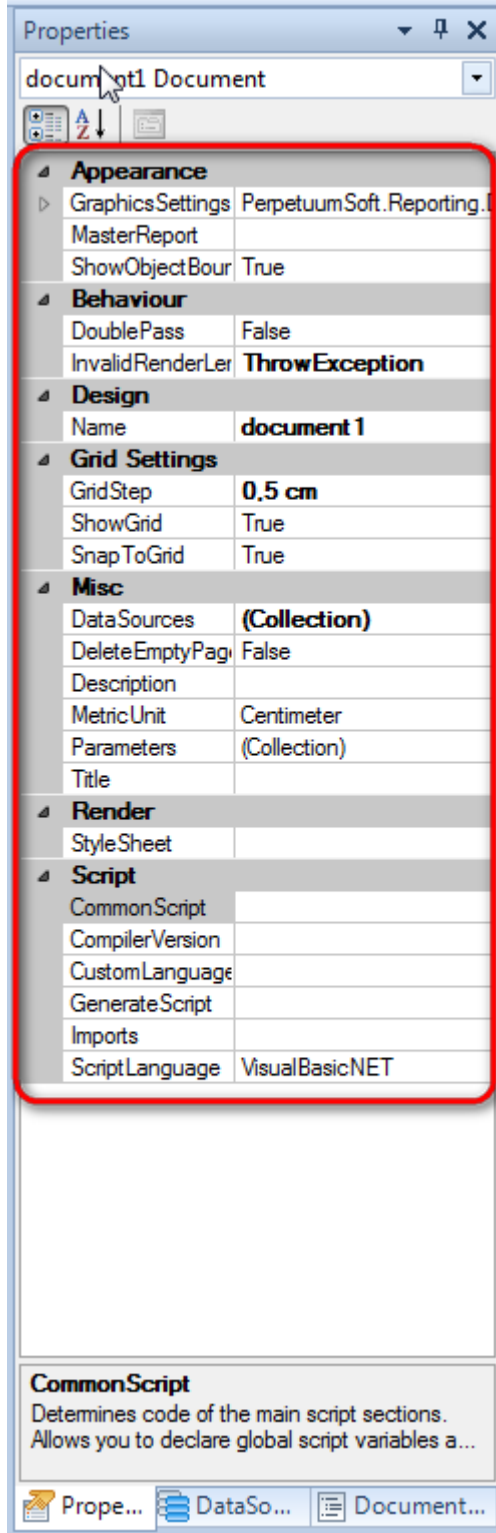
Step 8

Click the "Properties" tab of the tool window in the right part of the designer.

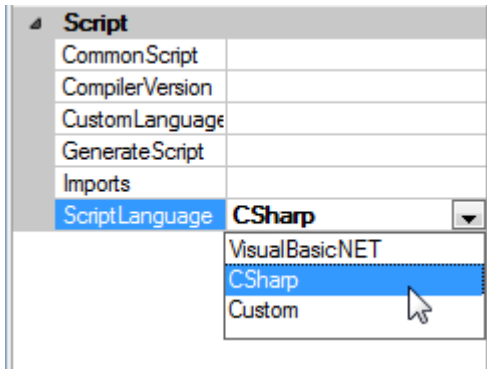




You will see properties of the edited template on the “Properties” tab

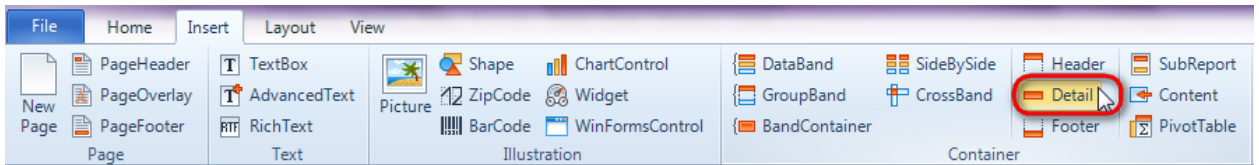


Set property ScriptLanguage = CSharp.



Step 9

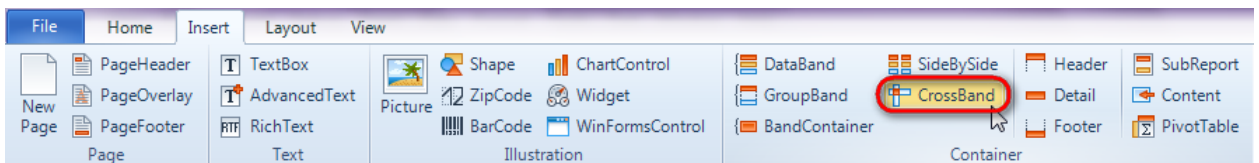
Press "Detail" button on the Insert tab in the group Container.



Click on the template area to add Detail band to the template.

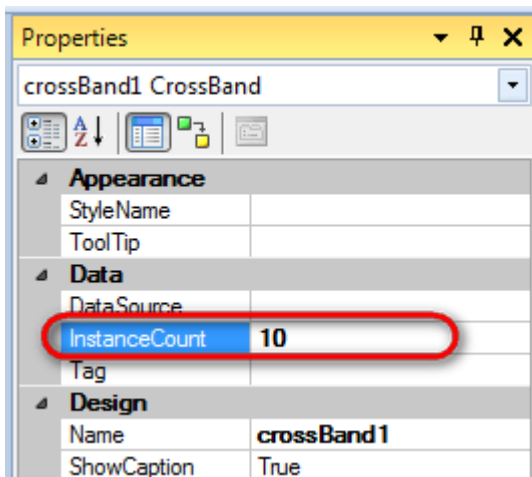
Step 10

Press "CrossBand" button on the Insert tab in the group Container.



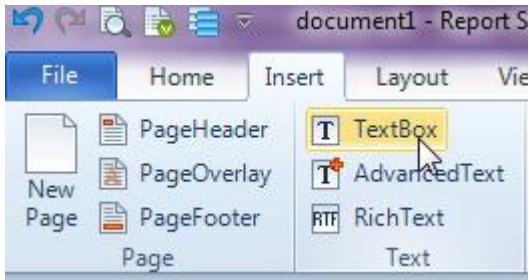
Click on the Detail band area to add CrossBand inside Detail.

Set property InstanceCount = 10



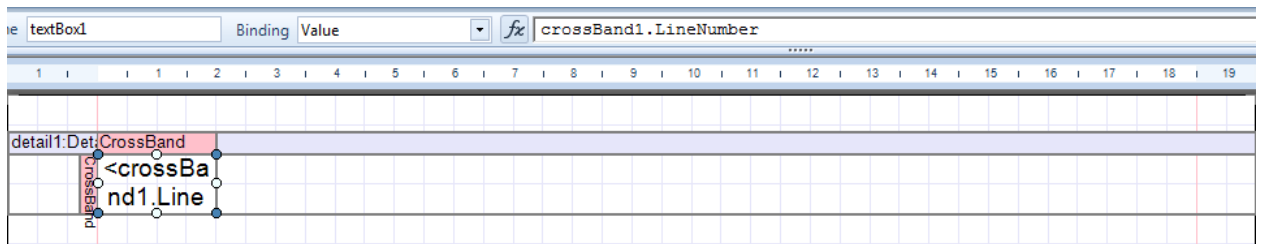
Step 11

Press button "TextBox" on the Insert tab in the group Text.



Click on the CrossBand area to add TextBox element inside CrossBand. Set property Value = crossBand1.LineNumber.

Report template should look as follows:

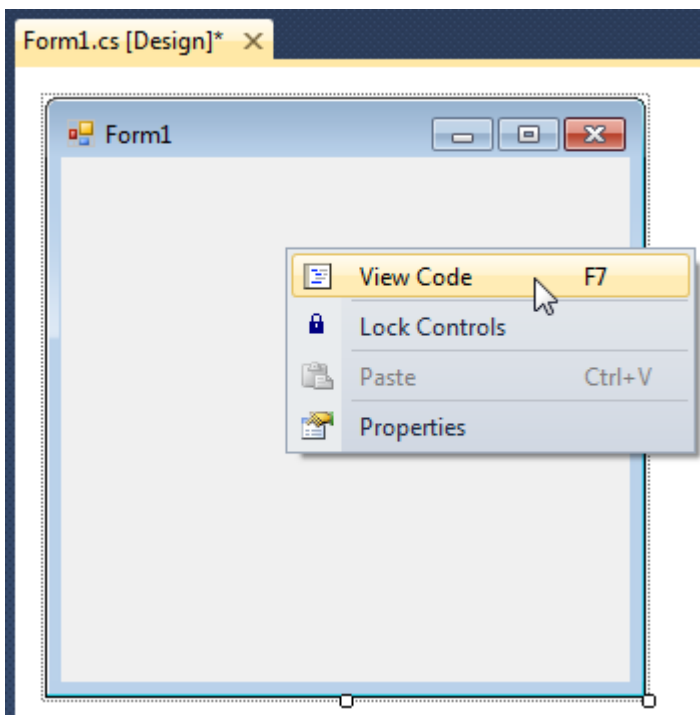


Step 12

Save template, close Report Designer.

Step 13

Right click on the application form and select "View Code" in the context menu to view code.



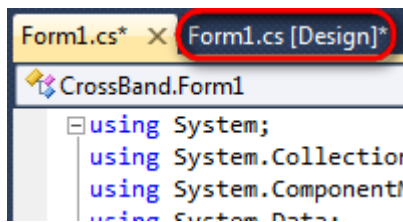
Add code to display report to the class constructor. Create RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
    EventHandler (reportSlot_RenderCompleted);
}
```

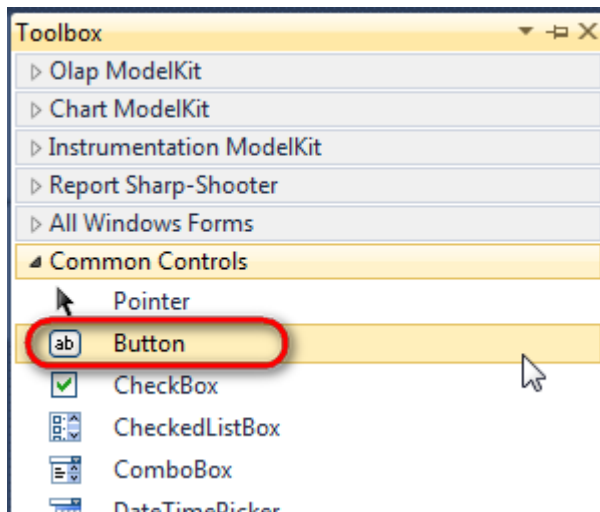
```
    }  
    private void reportSlot_RenderCompleted(object sender, EventArgs e)  
    {  
        using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new  
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))  
        {  
            previewForm.WindowState = FormWindowState.Maximized;  
            previewForm.ShowDialog(this);  
        }  
    }  
}
```

Step 14

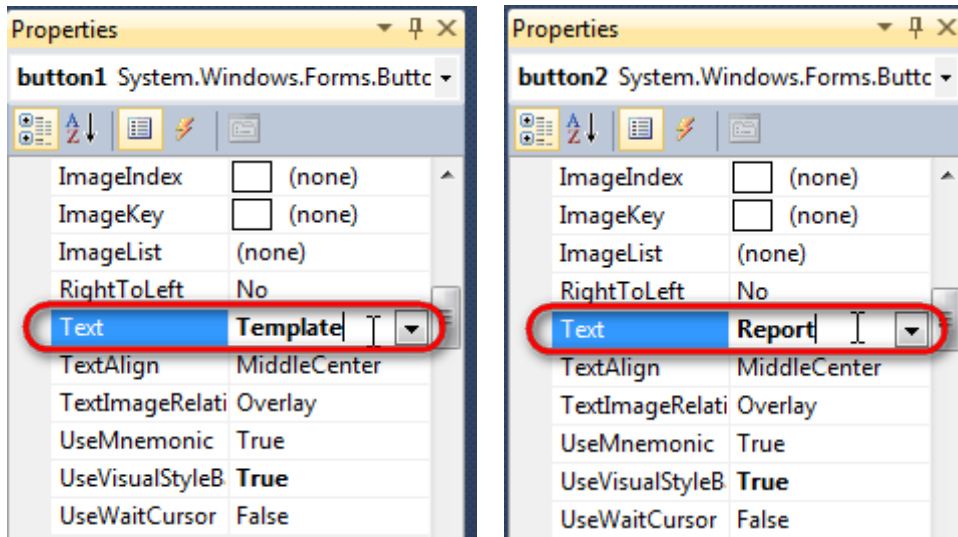
Get back to the application from by clicking "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



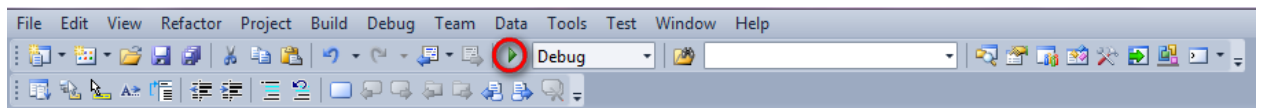
Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

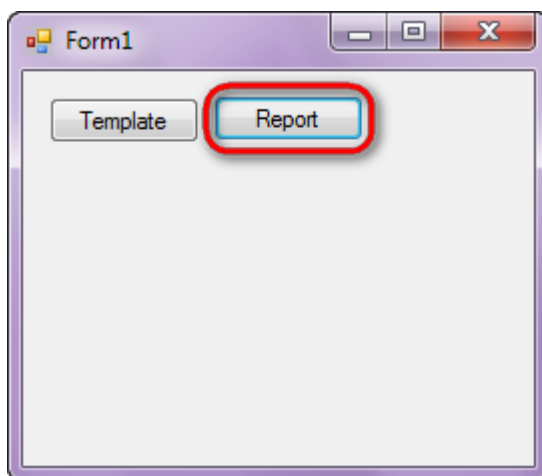
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 15

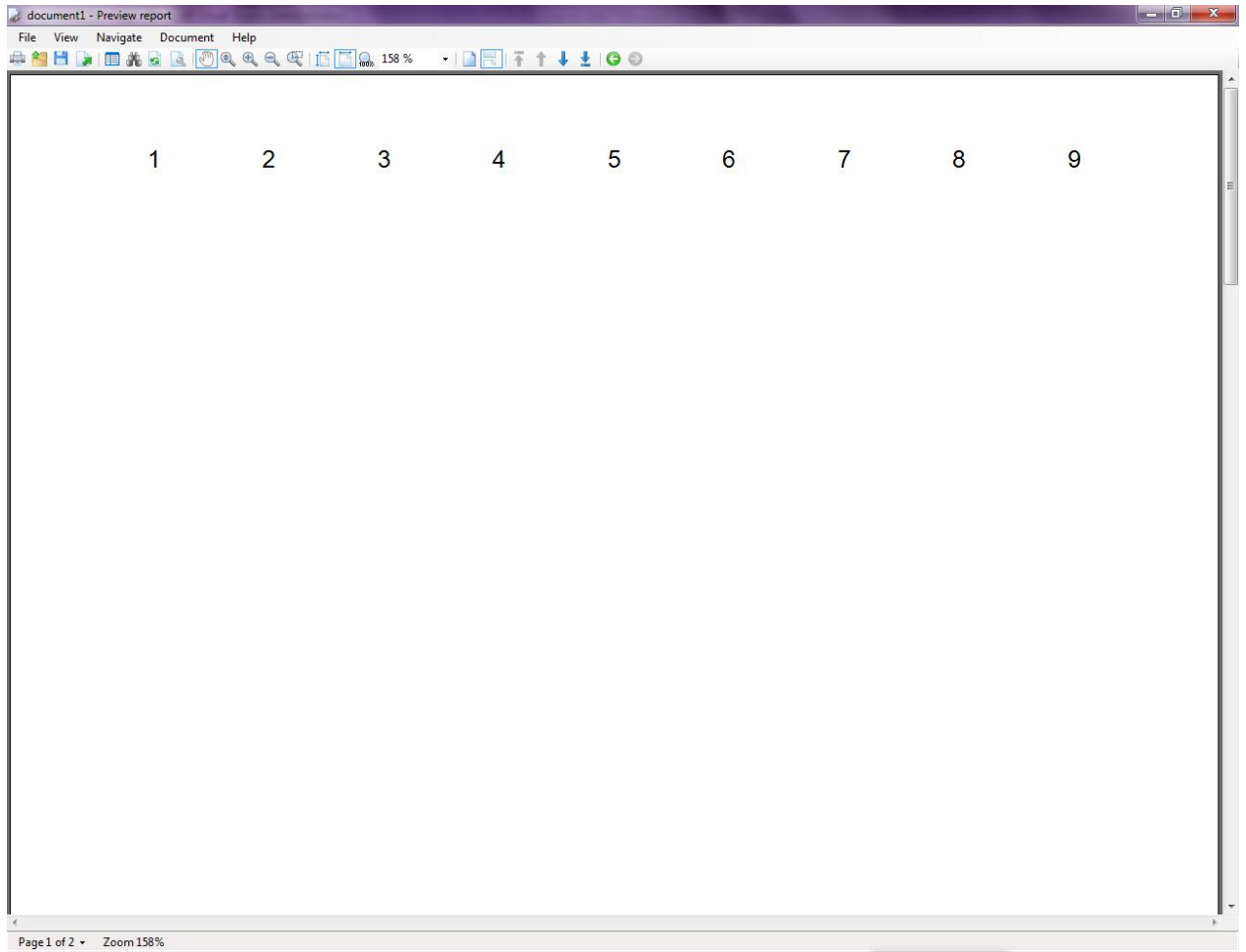
Click “Start Debugging” on the Visual Studio toolbar in order to start application.



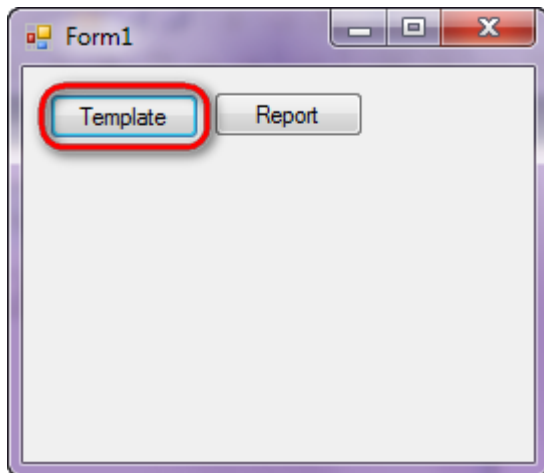
Click the “Report” button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



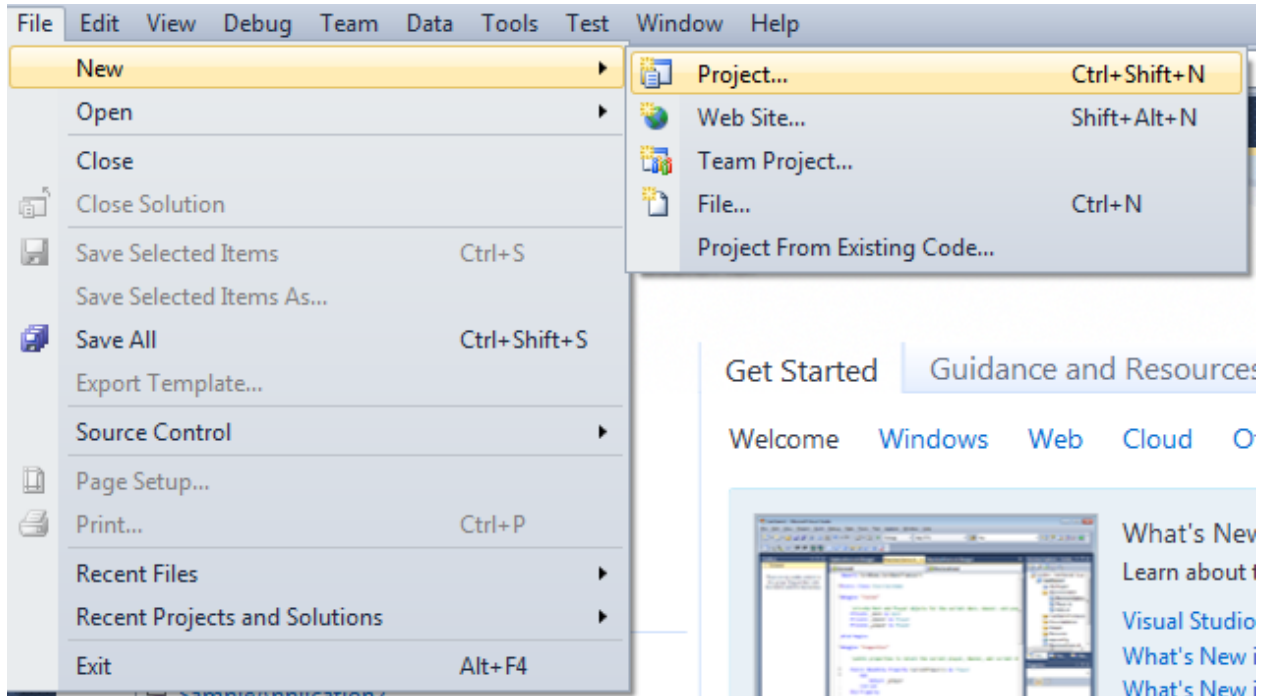


Matrix

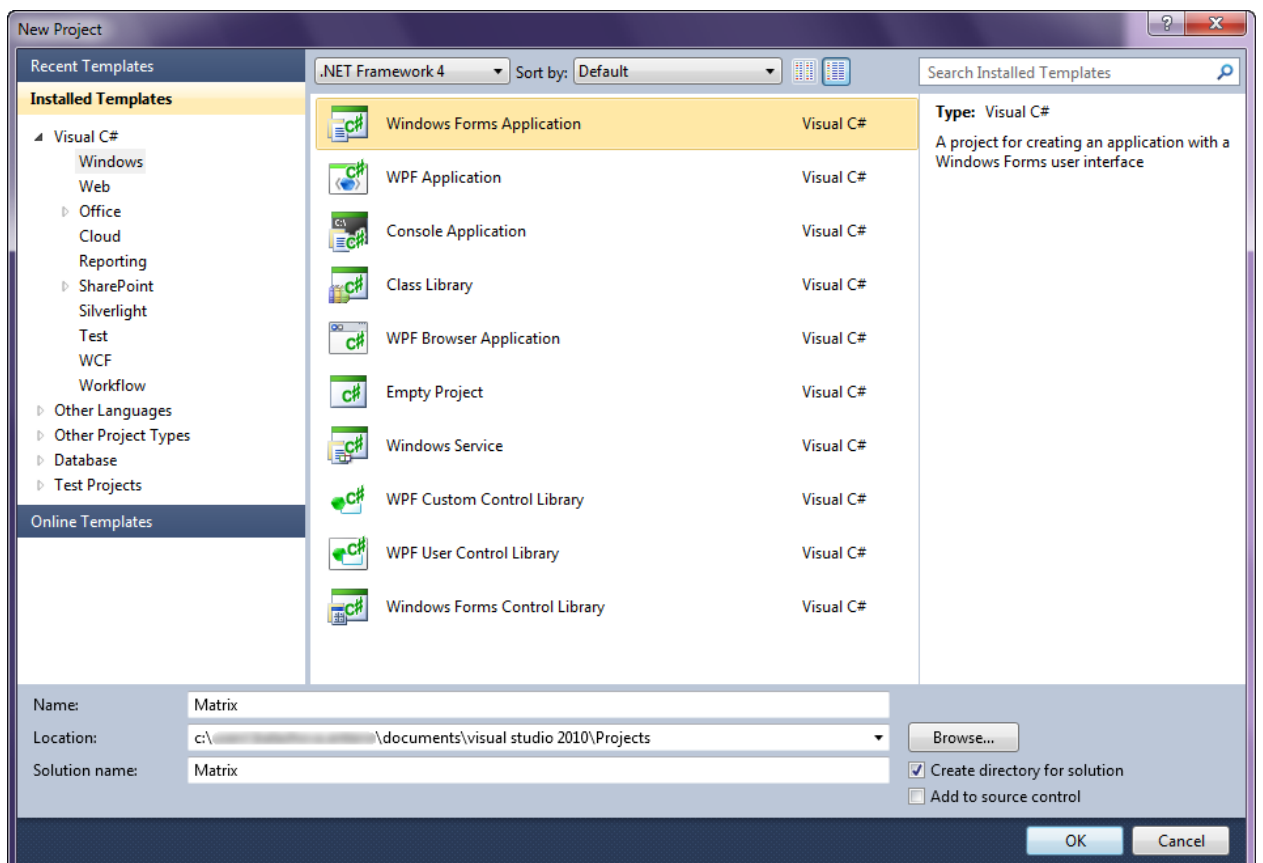
Template of a report containing multiplication table as matrix.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

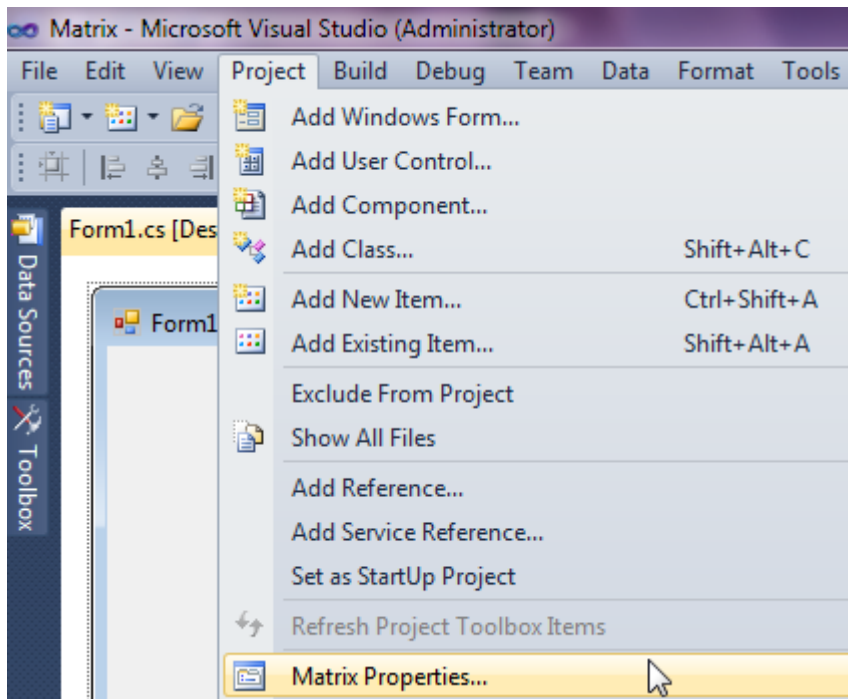


Select Windows Forms Application, set project name – “Matrix”, set directory to save the project to.

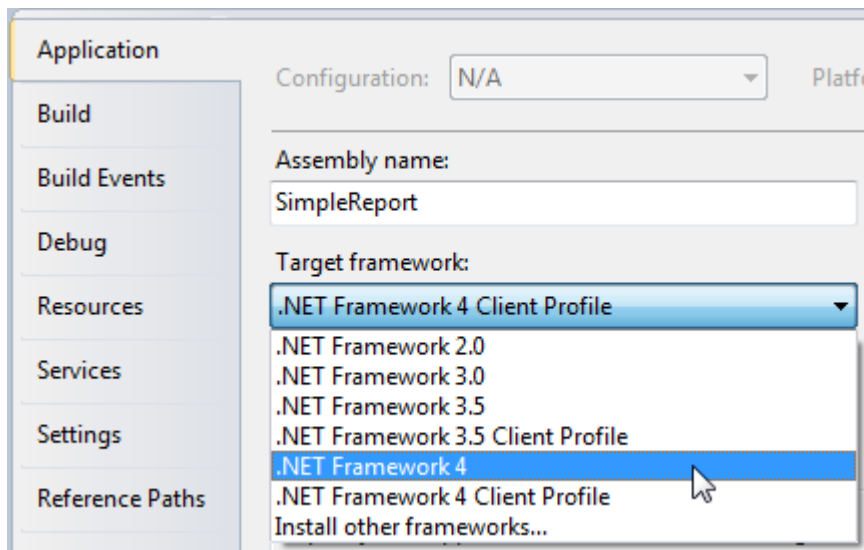


Step 2

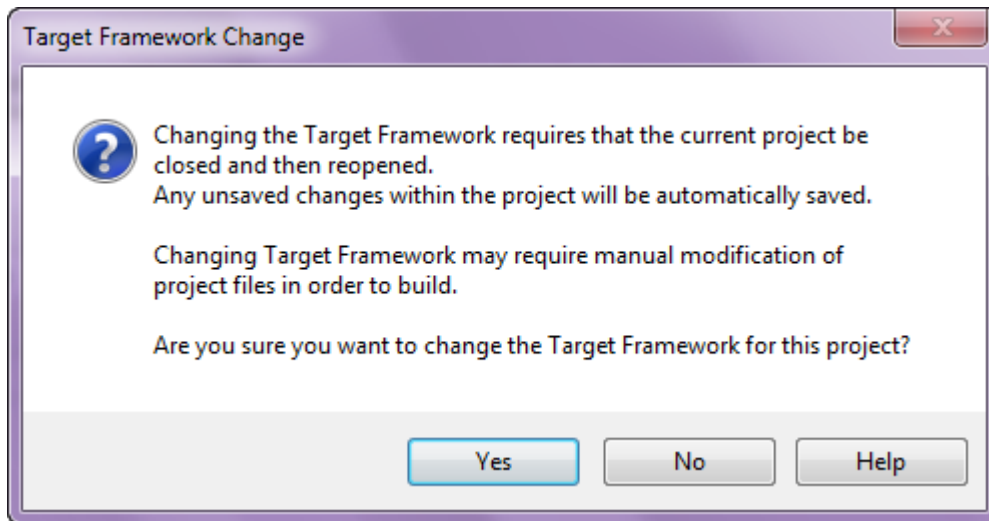
Change the project properties. Select the Project\Matrix Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

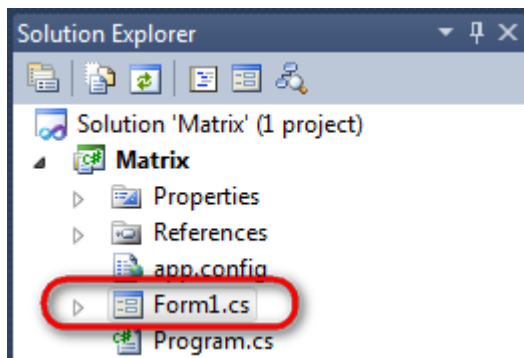


In the opened window press the "Yes" button.

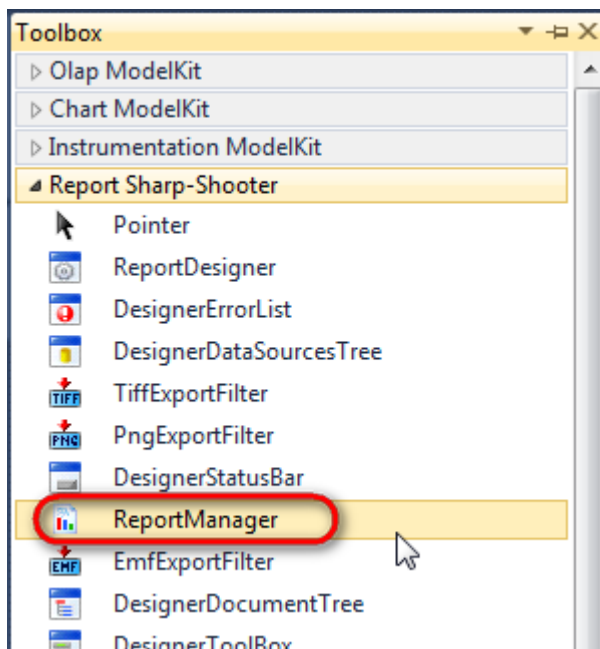


Step 3

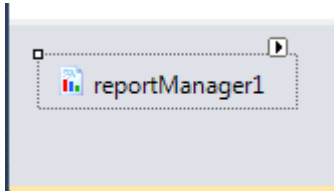
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

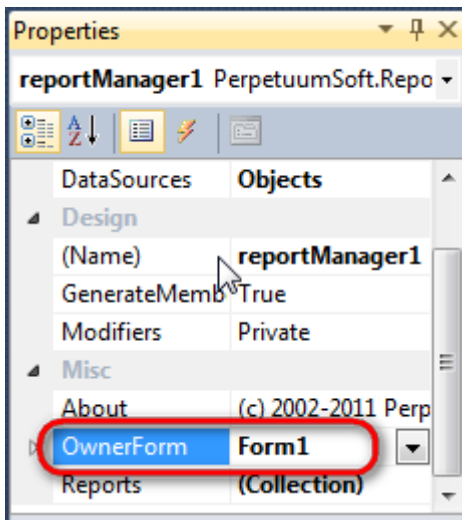


The component is available in the lower part of the window.



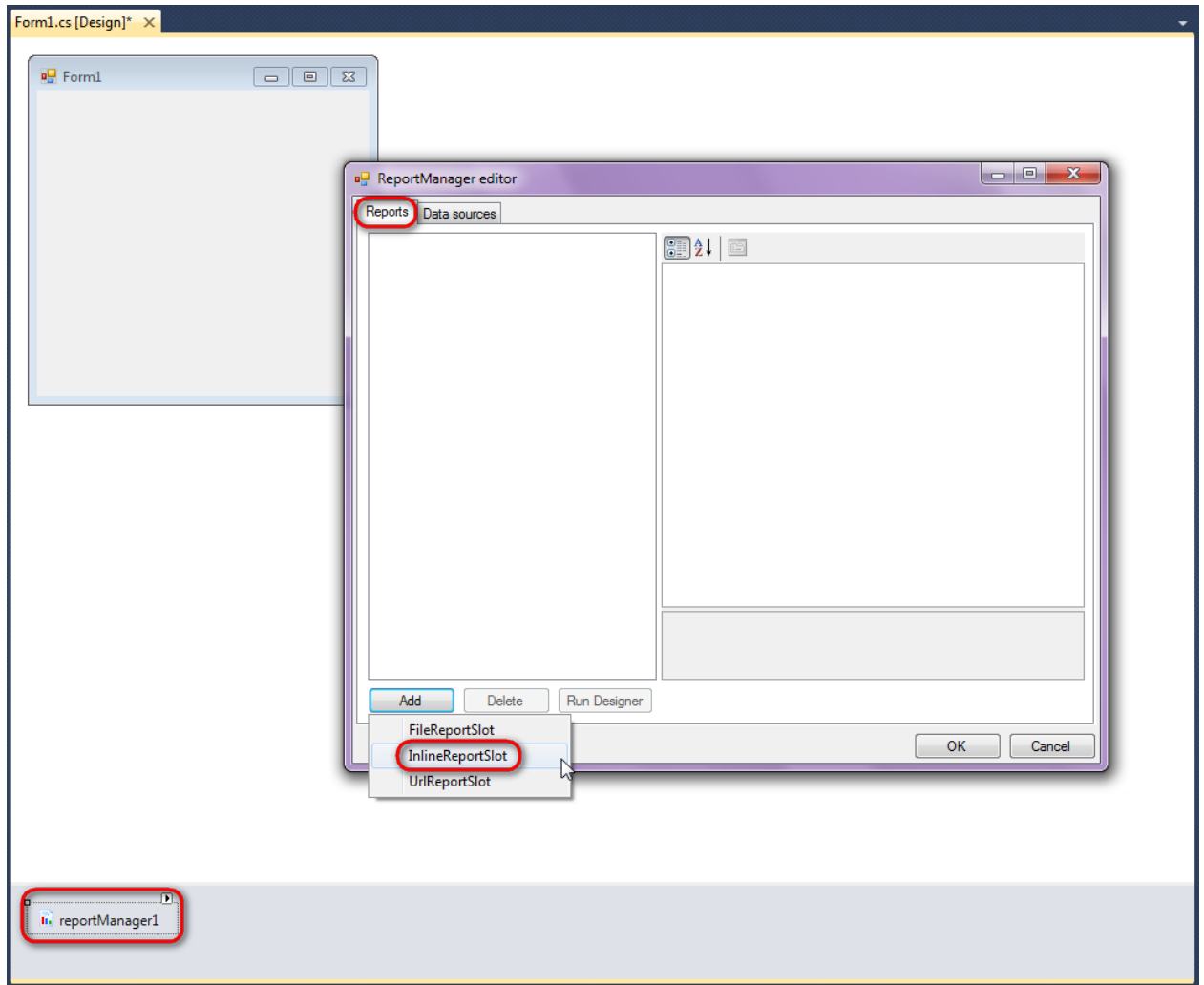
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

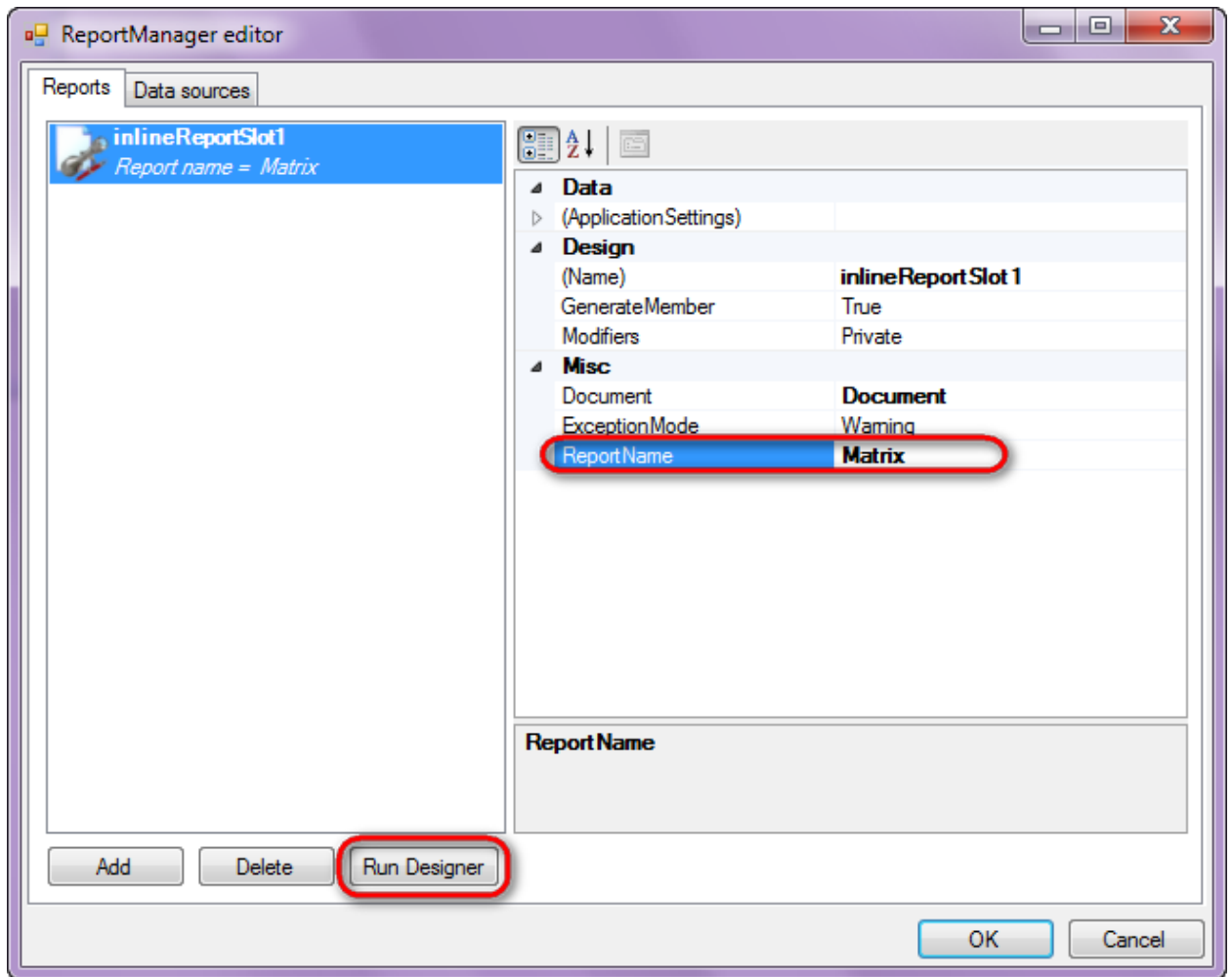


On the "Reports tab", click "Add" and select "InlineReportSlot".

Step 6

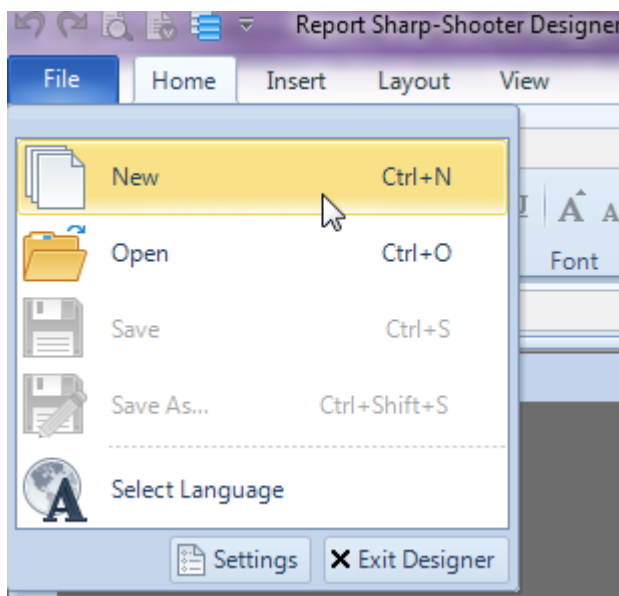
Set name of the report in the property ReportName - "Matrix".

Click "Run Designer" in order to open template editor - Report Designer.

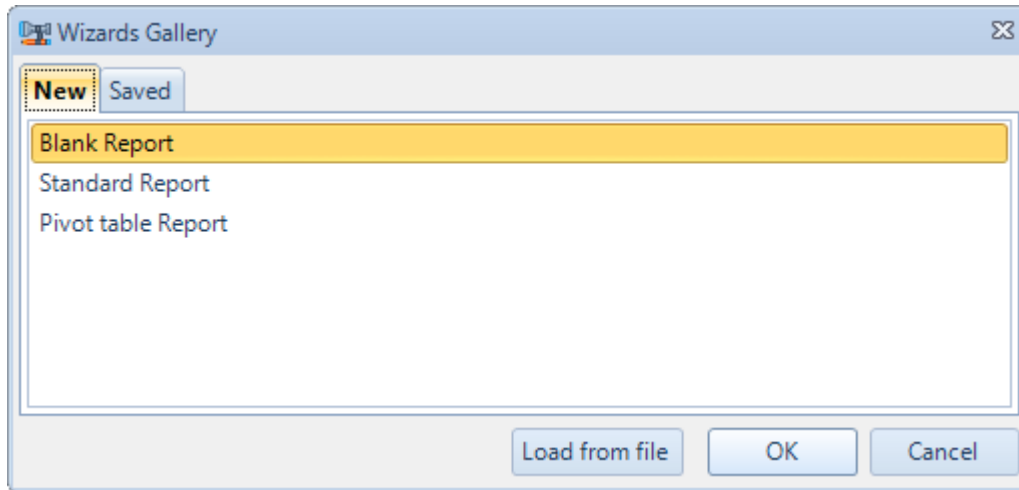


Step 7

Create new empty template – select item File\New from the main menu.

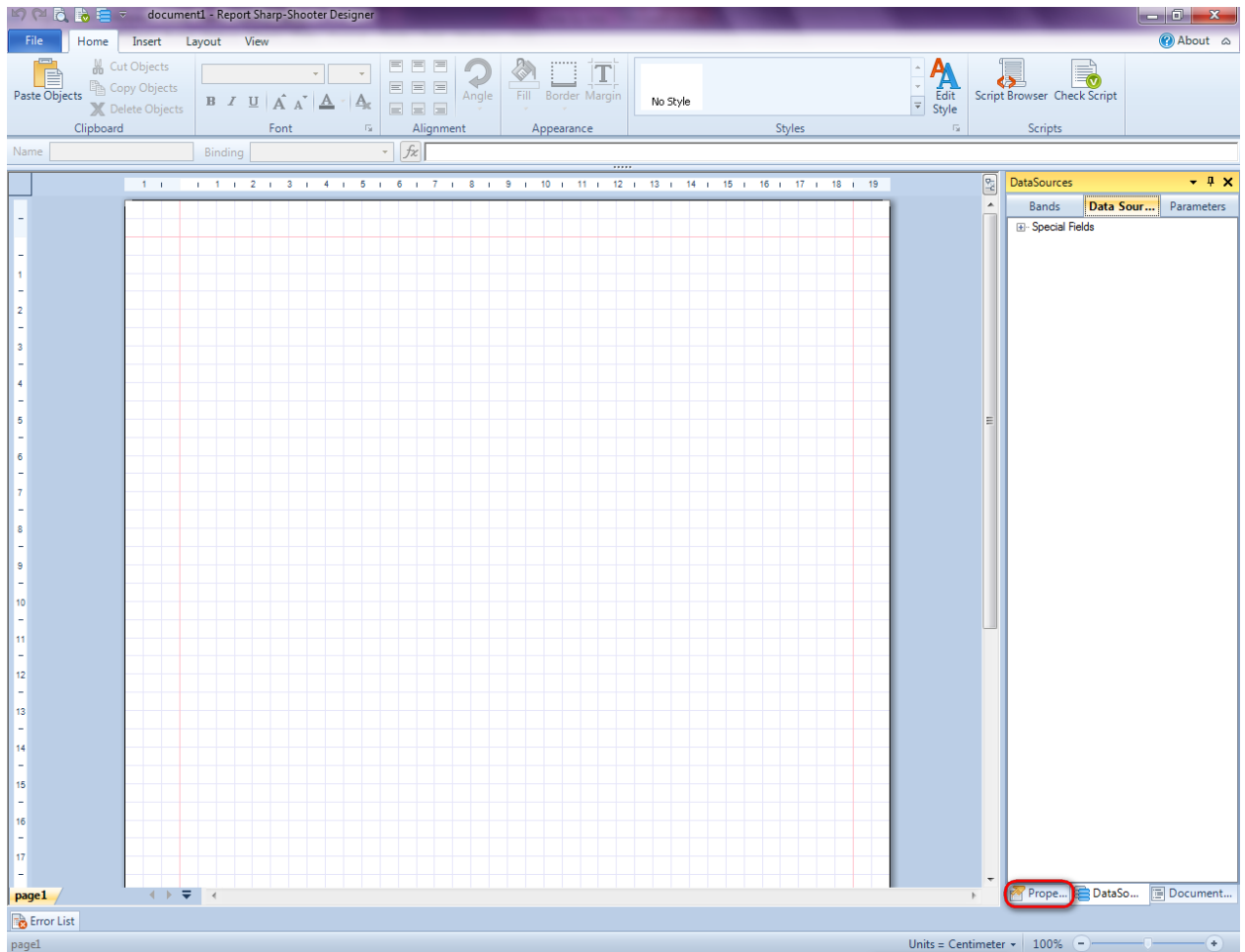


Select "Blank Report" in the Wizards Gallery and click "OK".

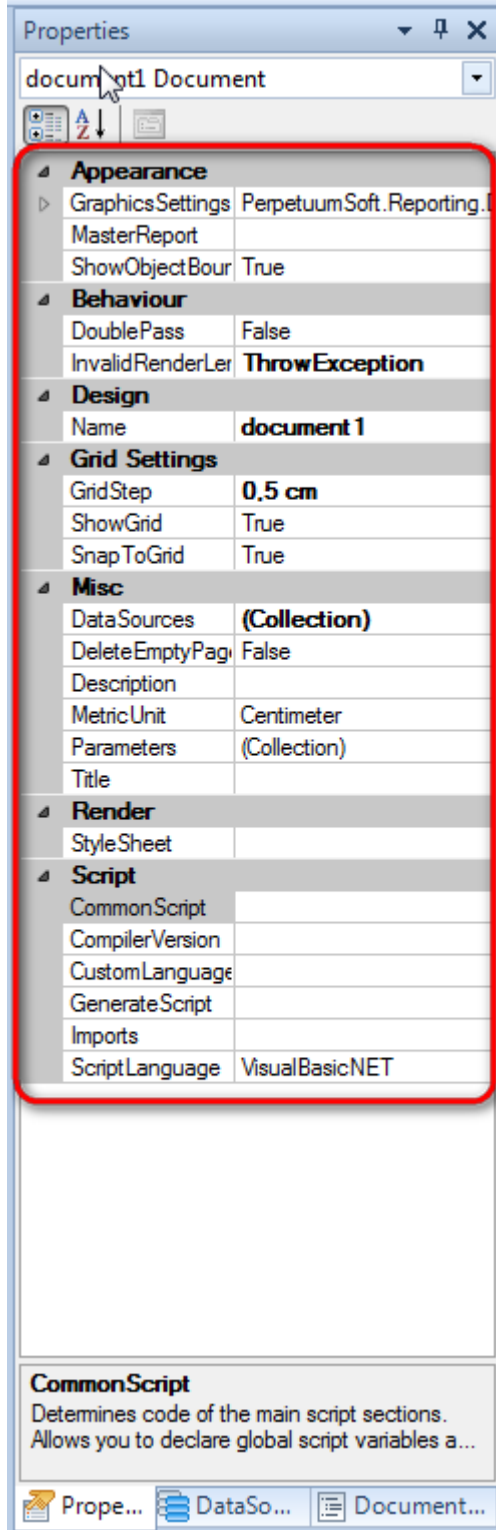


Step 8

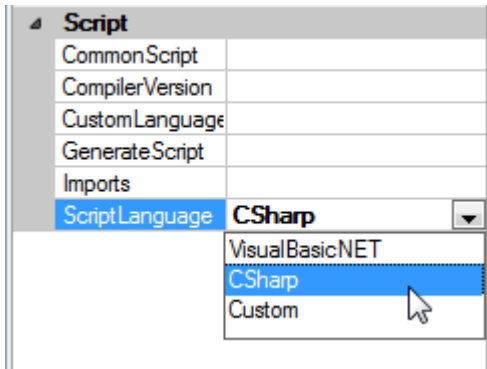
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the "Properties" tab

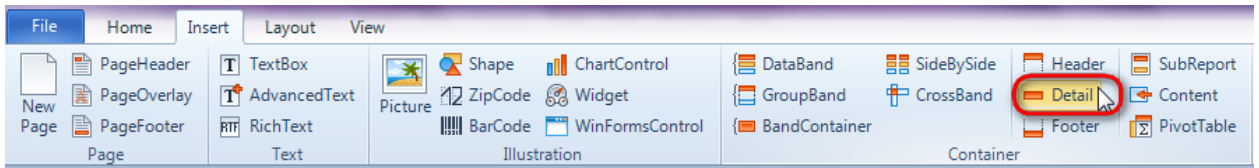


Set property ScriptLanguage = CSharp.



Step 9

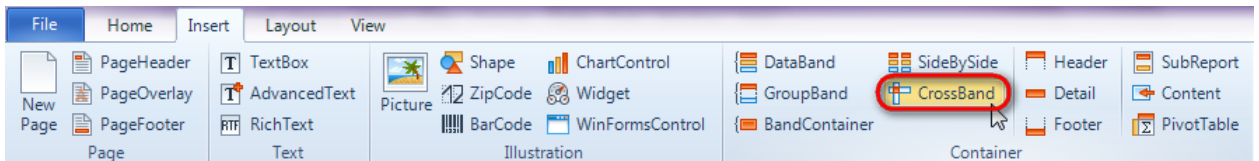
Press "Detail" button on the Insert tab in the group Container.



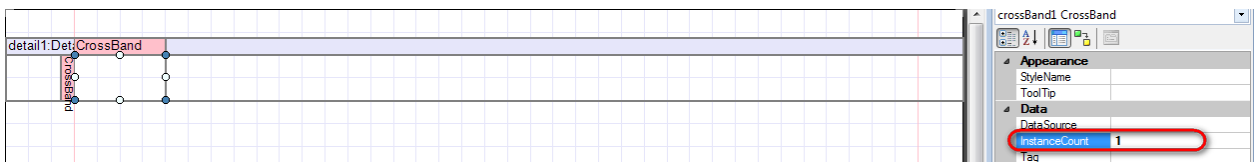
Click on the template area to add Detail band to the template.

Step 10

Press "CrossBand" button on the Insert tab in the group Container.

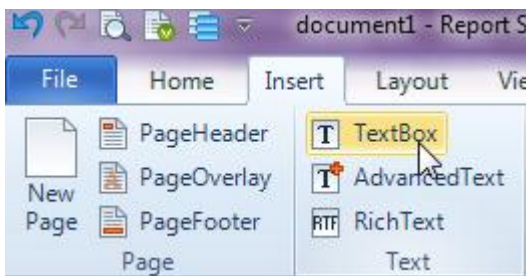


Click on the Detail band area to add CrossBand inside Detail. Set InstanceCount property to "1".



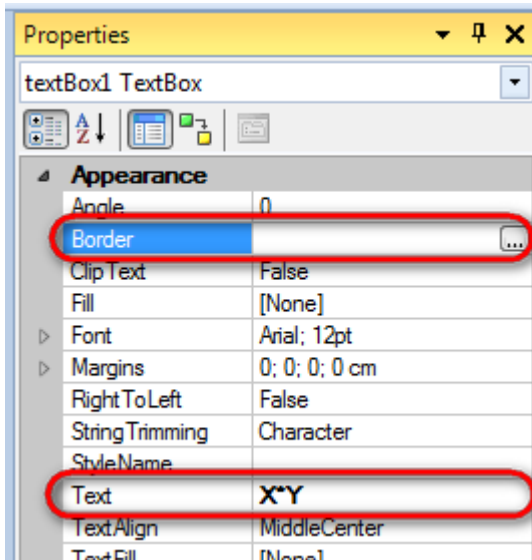
Step 11


Press button "TextBox" on the Insert tab in the group Text.

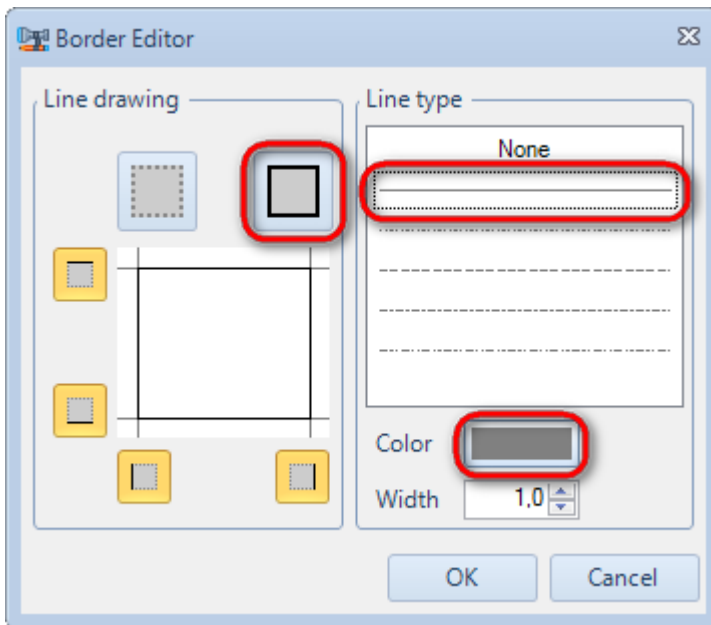


Click on the CrossBand area to add TextBox element inside CrossBand.

Set property Text = X * Y.



Select Border property. Click button  to open Border Editor. Set element border.

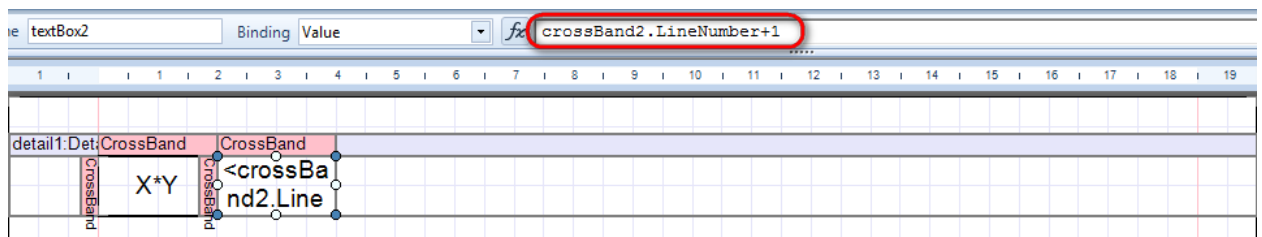


Step 12

Add another CrossBand inside detail1 band on the right of crossBand1. Set InstanceCount property to "8".

Step 13

Add TextBox element inside crossBand2. Set property Value = crossBand2.LineNumber+1.

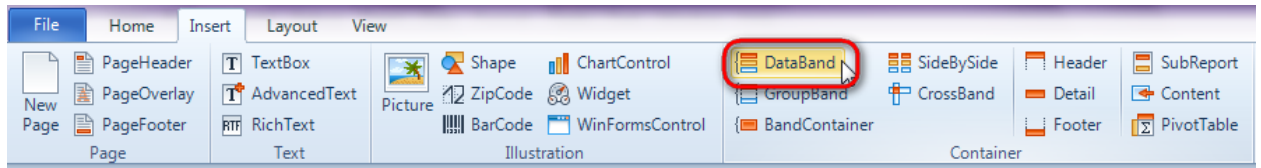


Set Border property as shown in step 10.

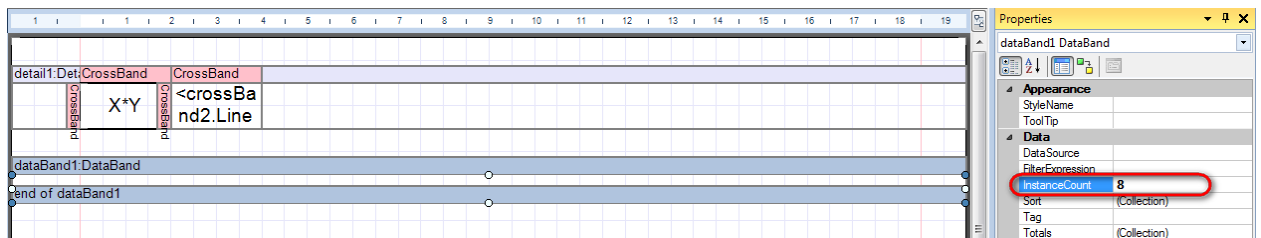


Step 14

Press "DataBand" button on the Insert tab in the group Container.

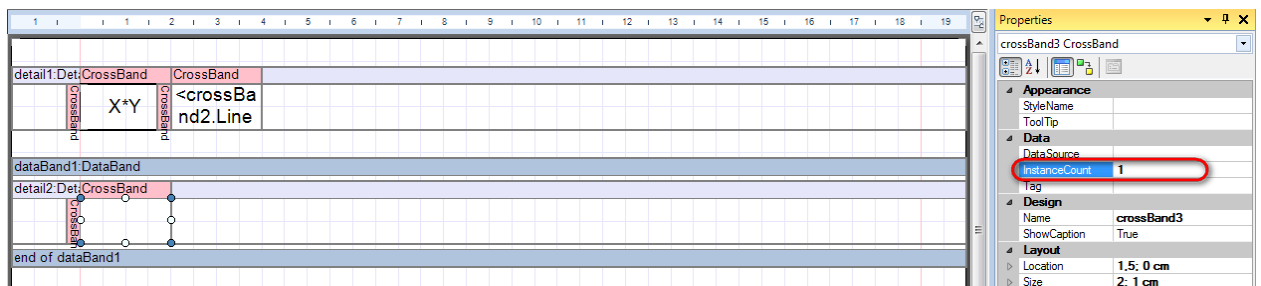


Click template area to add DataBand to the template. Set property InstanceCount = 8.



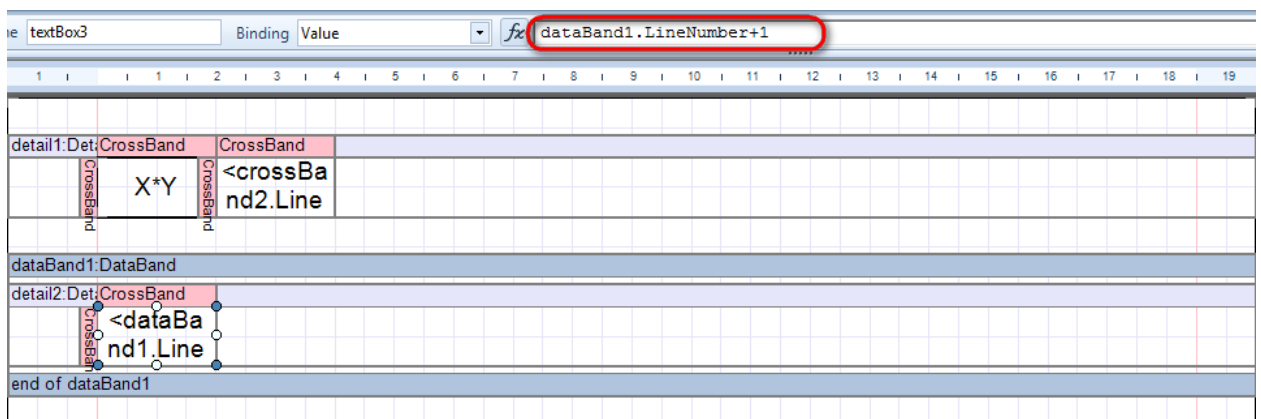
Step 15

Add Detail band inside dataBand1. In detail2, add CrossBand so that it is positioned left below crossBand1. Set property InstanceCount = 1.



Step 16

Add TextBox element to the crossBand3. Set Value property to dataBand1.LineNumber+1. Set Border property as shown in step 10.



Step 17

Add another CrossBand inside detail2 so that it is located right below crossBand2. Set property InstanceCount = 8.

Step 18

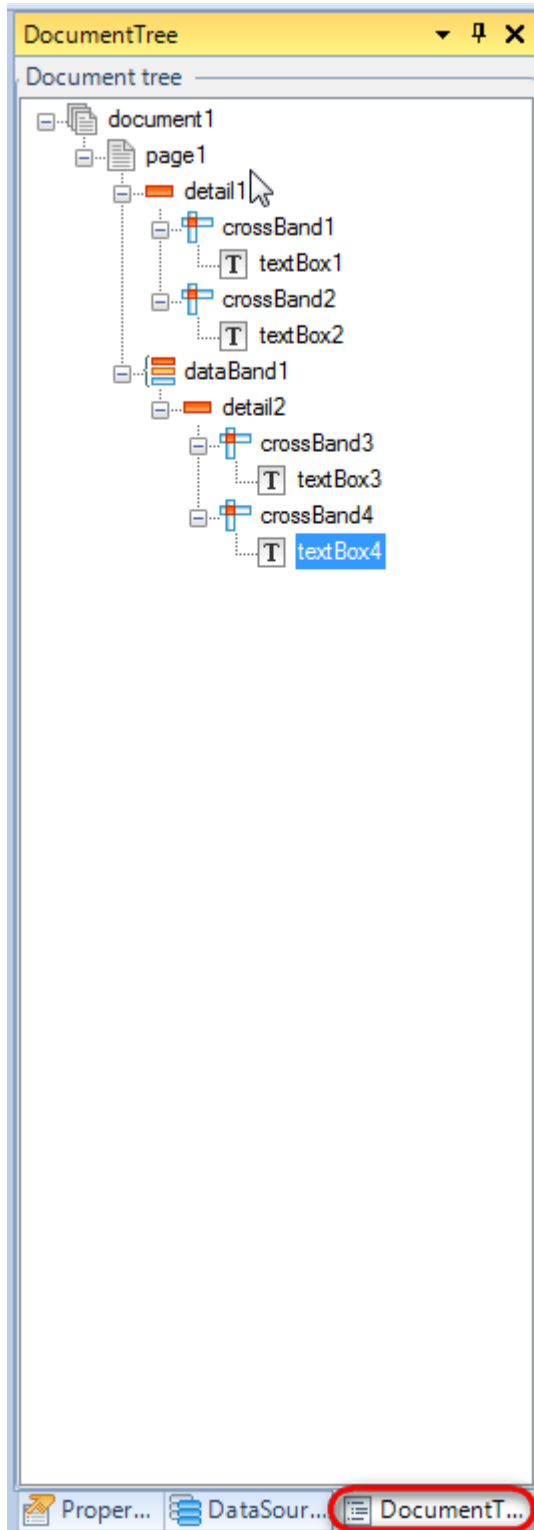
Add TextBox to the crossBand4. Set Value property to (dataBand1.LineNumber+1)*(crossBand4.LineNumber+1). Set Border property as shown in the Step 11.



Report template should look as follows:

detail1:Det:	CrossBand	CrossBand	
	CrossBand	CrossBand	
	X * Y	<crossBa nd2.Line	
	CrossBand	CrossBand	
dataBand1:DataBand			
detail2:Det:	CrossBand	CrossBand	
	CrossBand	CrossBand	
	<dataBa nd1.Line	< (dataBan	
	CrossBand	CrossBand	
end of dataBand1			

To view template structure go to "DocumentTree" tab.

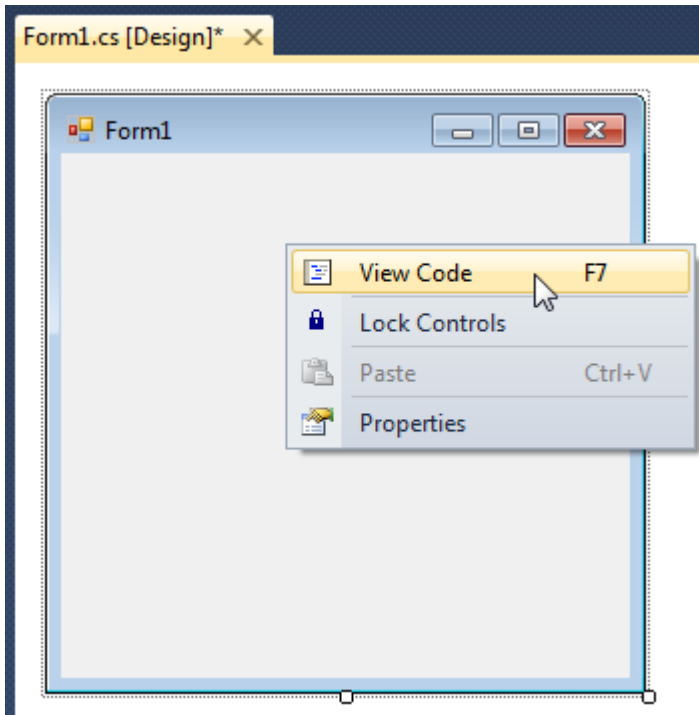


Step 19

Save template, close Report Designer.

Step 20

Right click on the application form and select "View Code" in the context menu to view code.

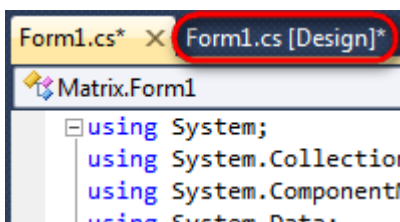


Add code to display report to the class constructor. Create `RenderCompleted` event handler of the `InlineReportSlot` object.

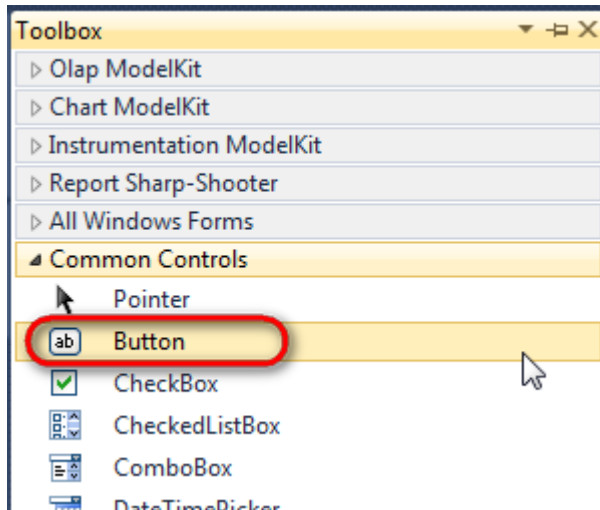
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 21

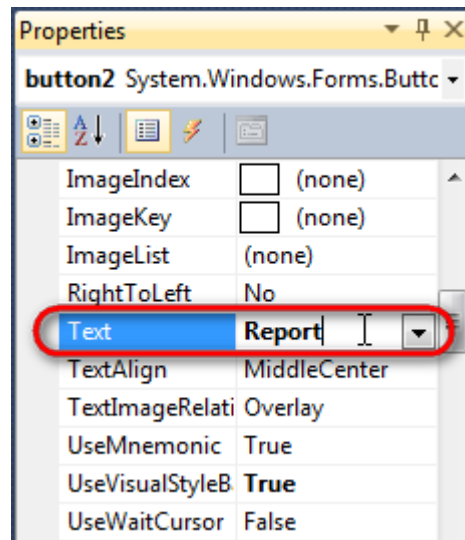
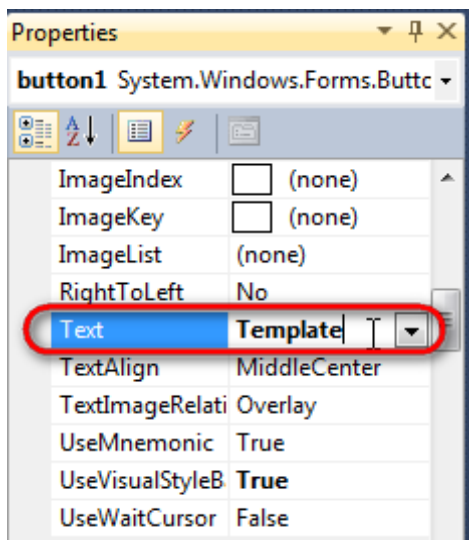
Get back to the application form by clicking "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



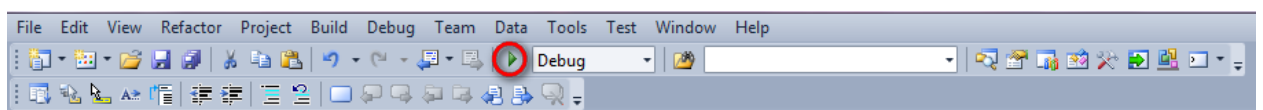
Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

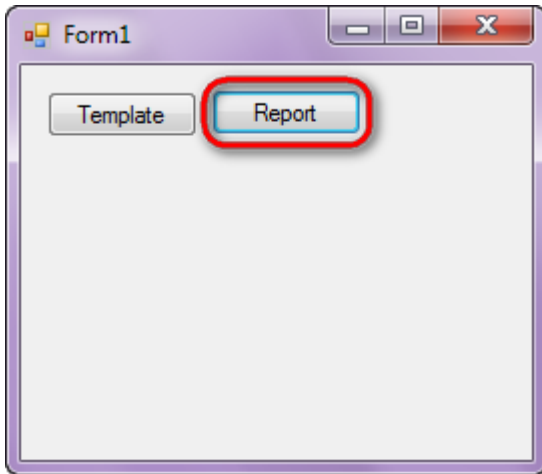
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

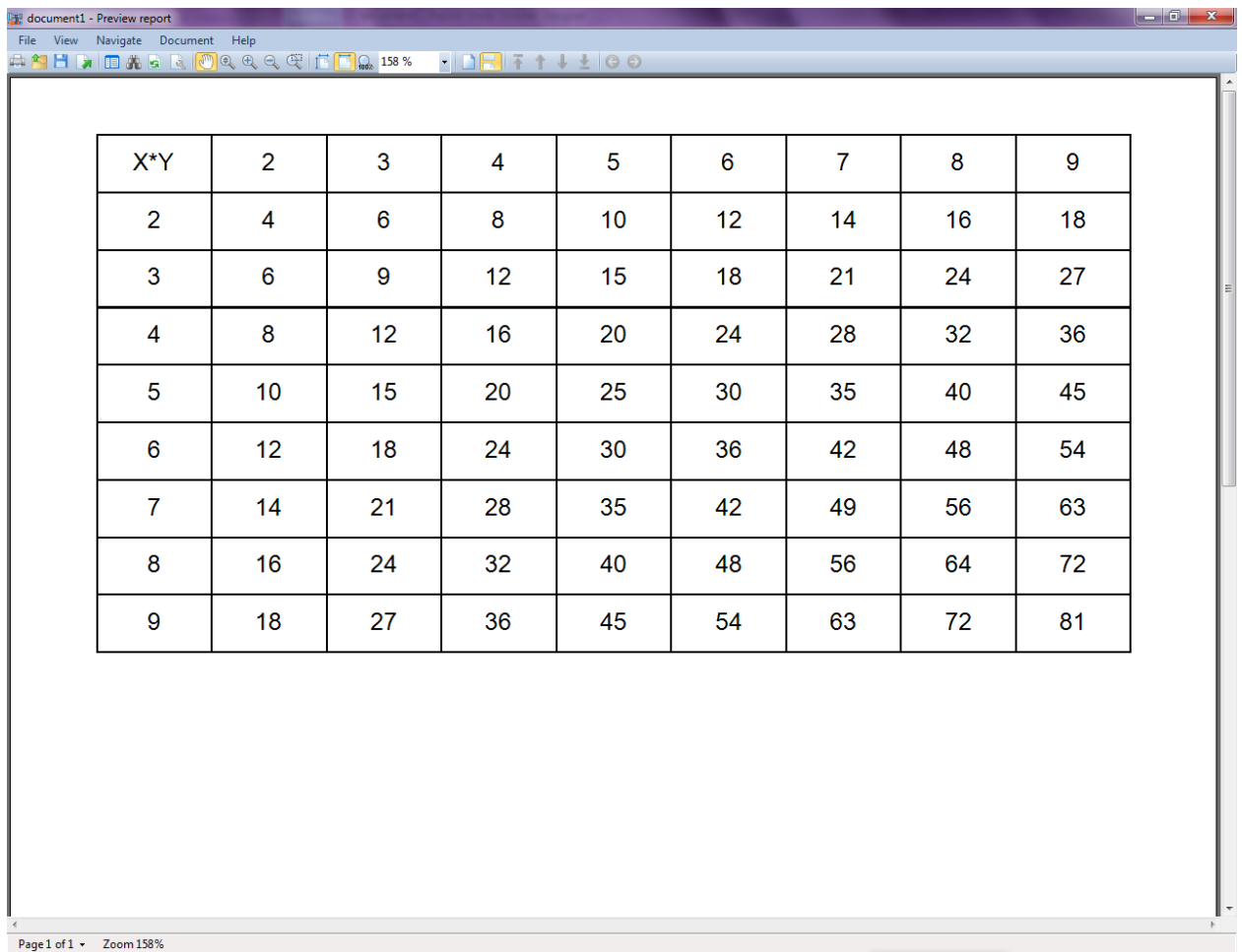
Click "Start Debugging" on the Visual Studio toolbar in order to start application.



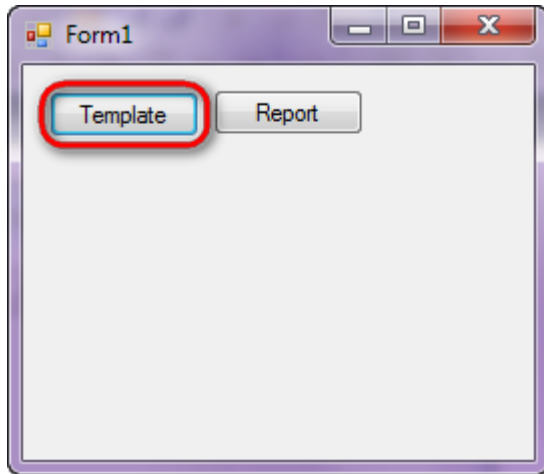
Click the "Report" button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



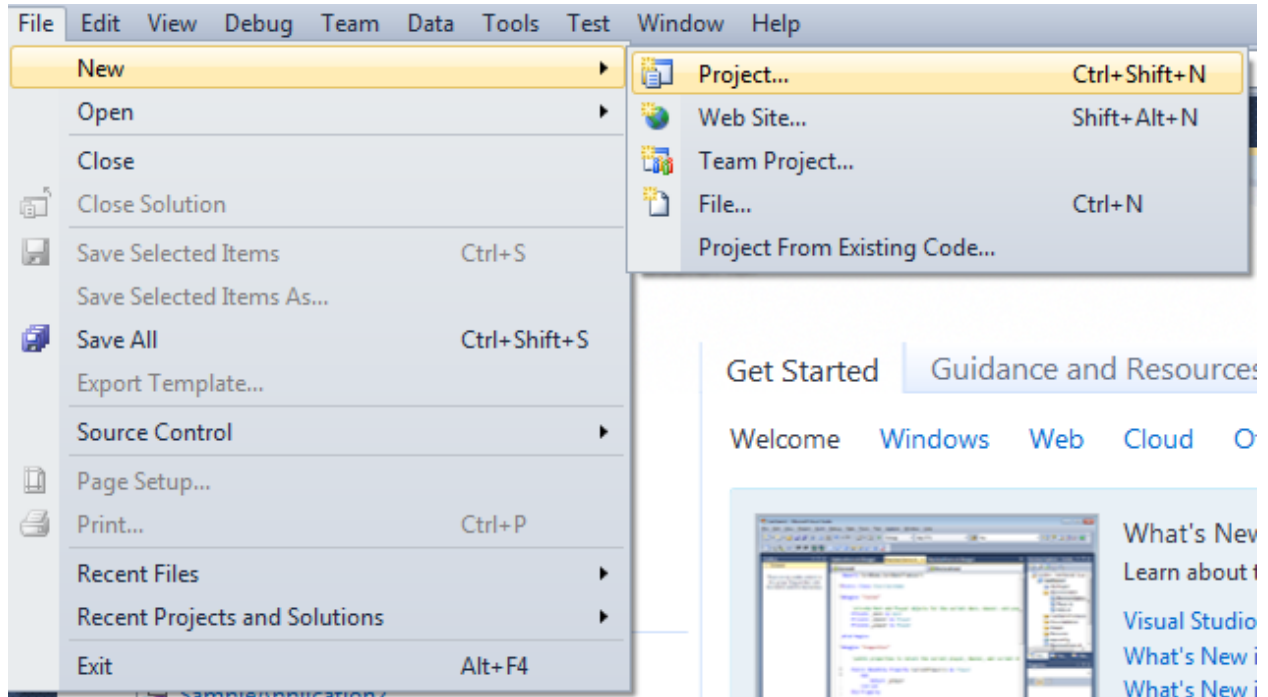
Similar sample in the Samples Center is Reports\Special Features\Cross-tab report.

Pivot Table

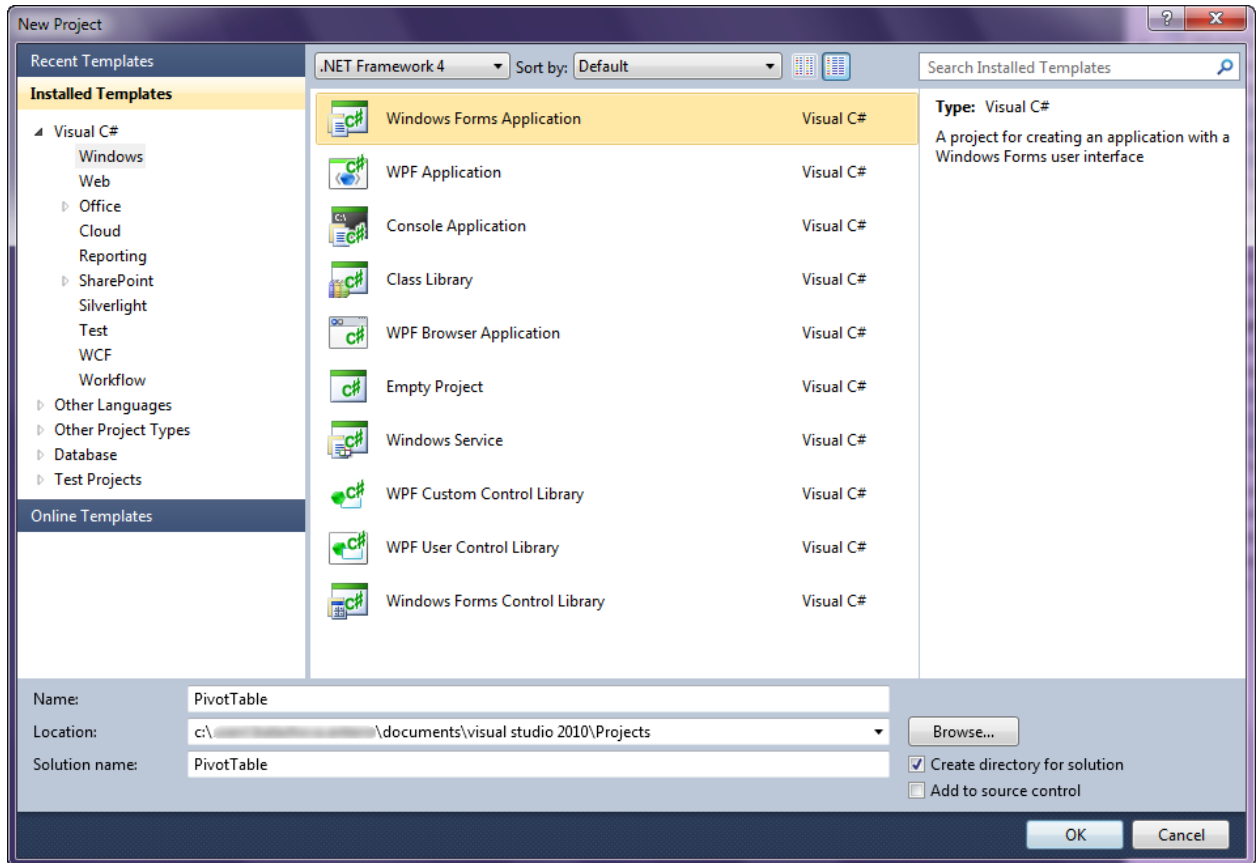
Template of a report containing pivot table with sales data. Vertical header contains information on categories and goods, horizontal header – customer names, table data – goods price considering amount and discount.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

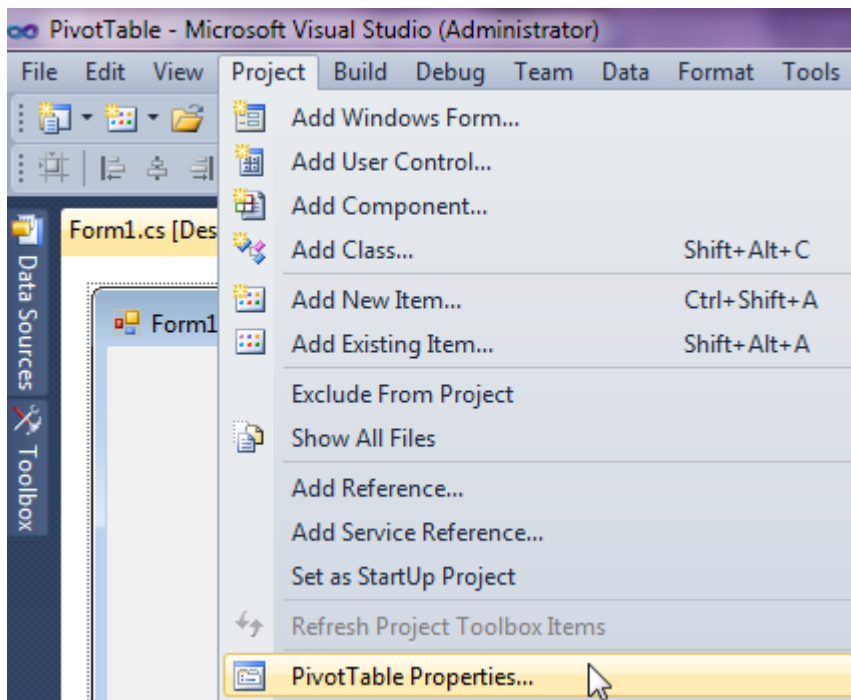


Select Windows Forms Application, set project name – “PivotTable”, set directory to save the project to.

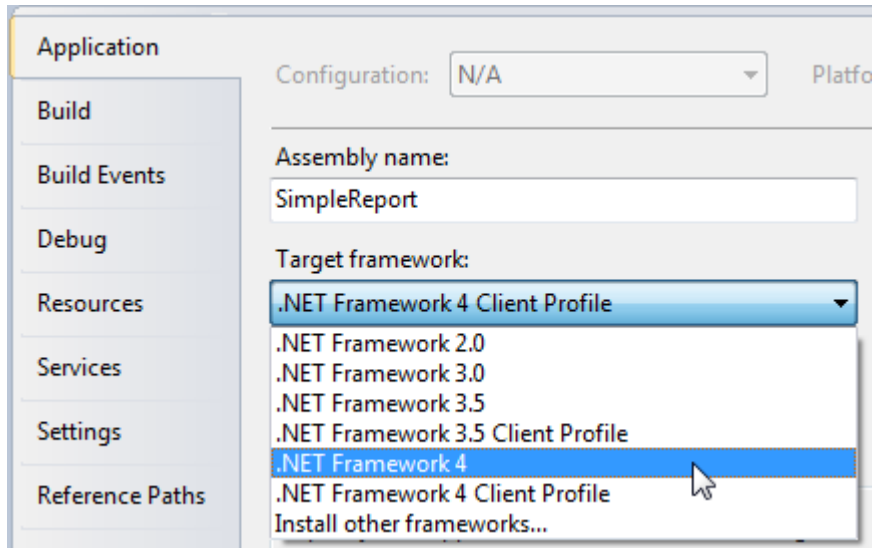


Step 2

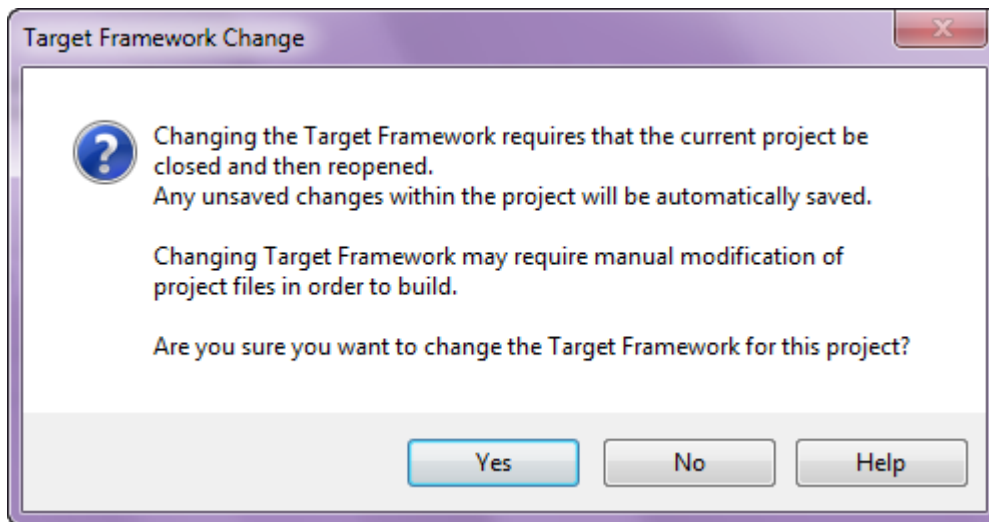
Change the project properties. Select the Project\PivotTable Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

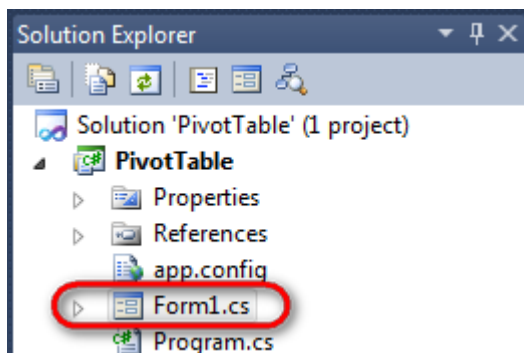


In the opened window press the “Yes” button.

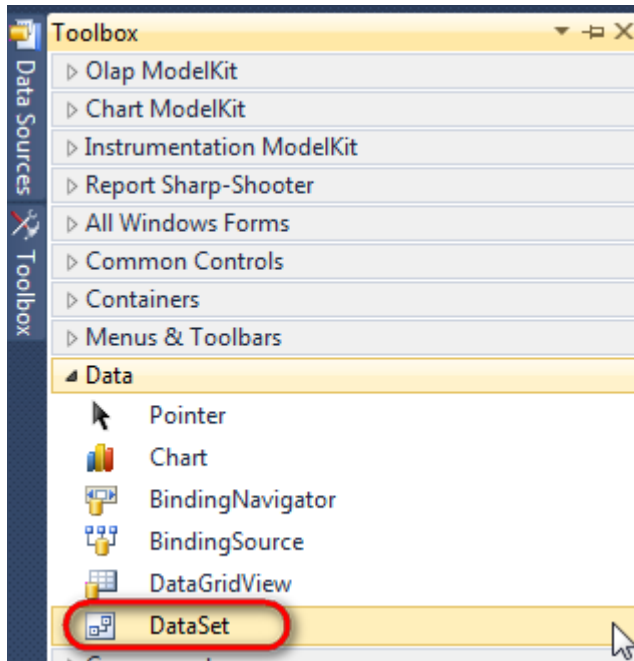


Step 3

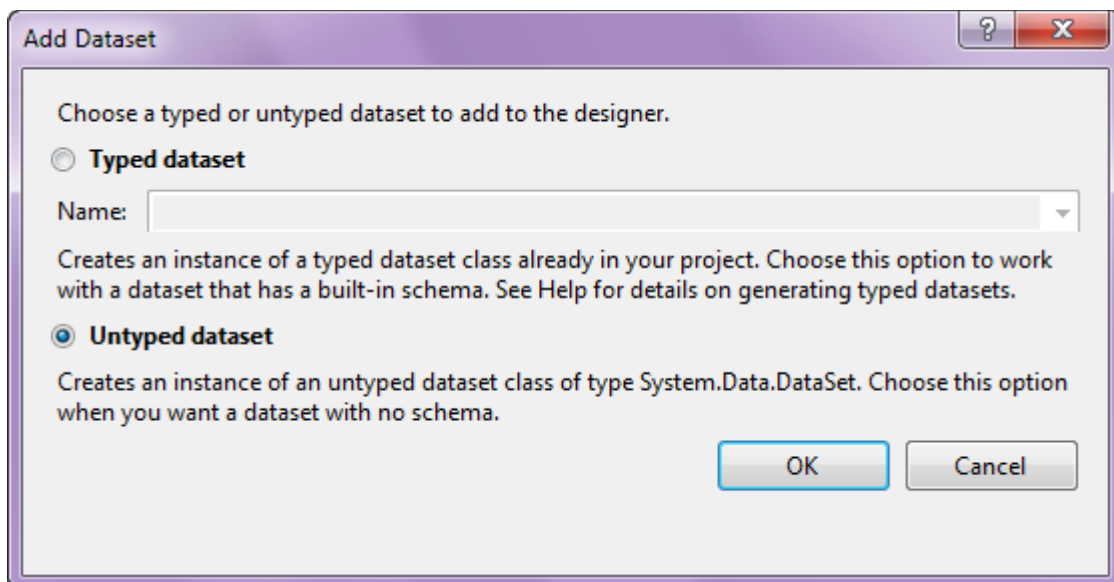
Open main form of the application by double click on the “Form1.cs” in the Solution Explorer.



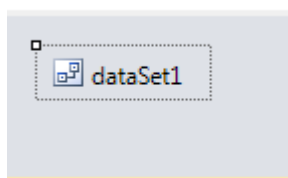
Click “DataSet” element on the Toolbox and place DataSet onto the form.




Select "Untyped dataset", click "OK".

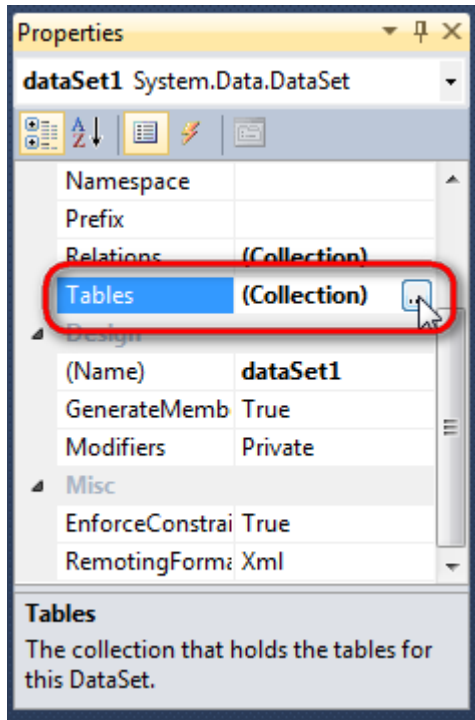


The component is available in the lower part of the window.

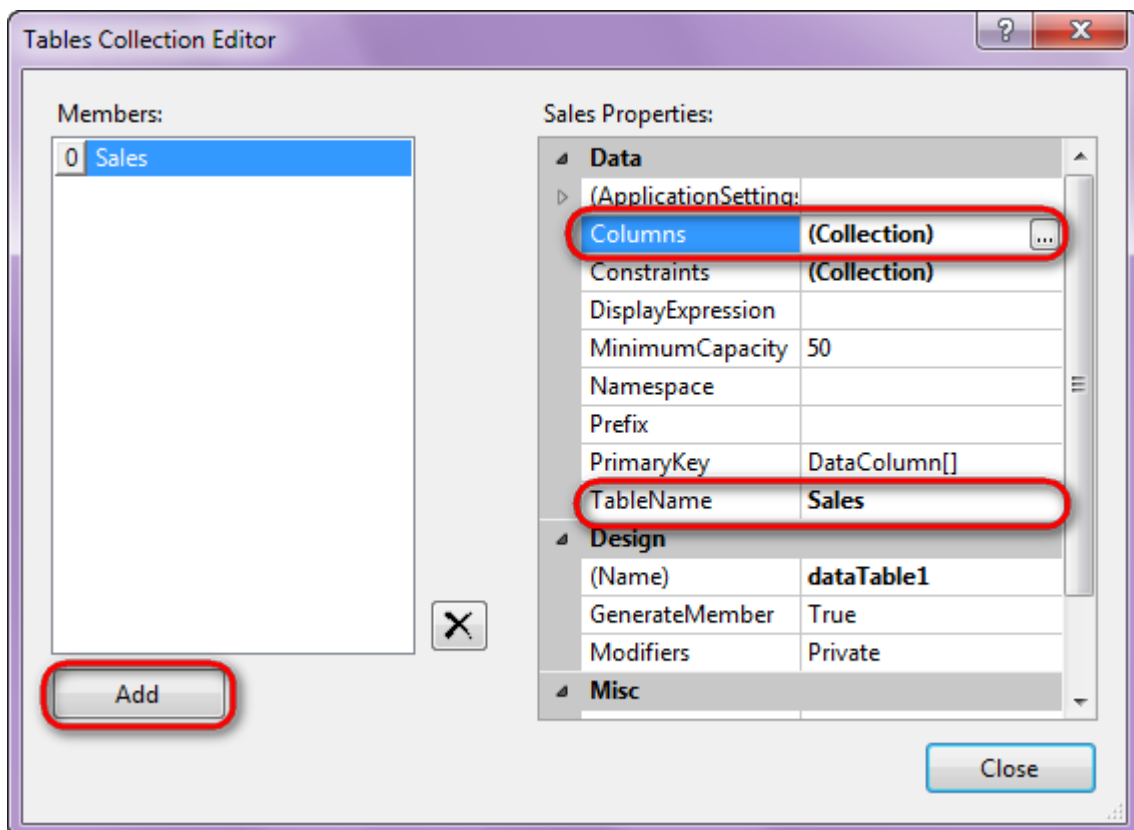


Step 4

Select dataSet1 in the form editor. On the property grid, select Tables property, click button  in order to open property editor.



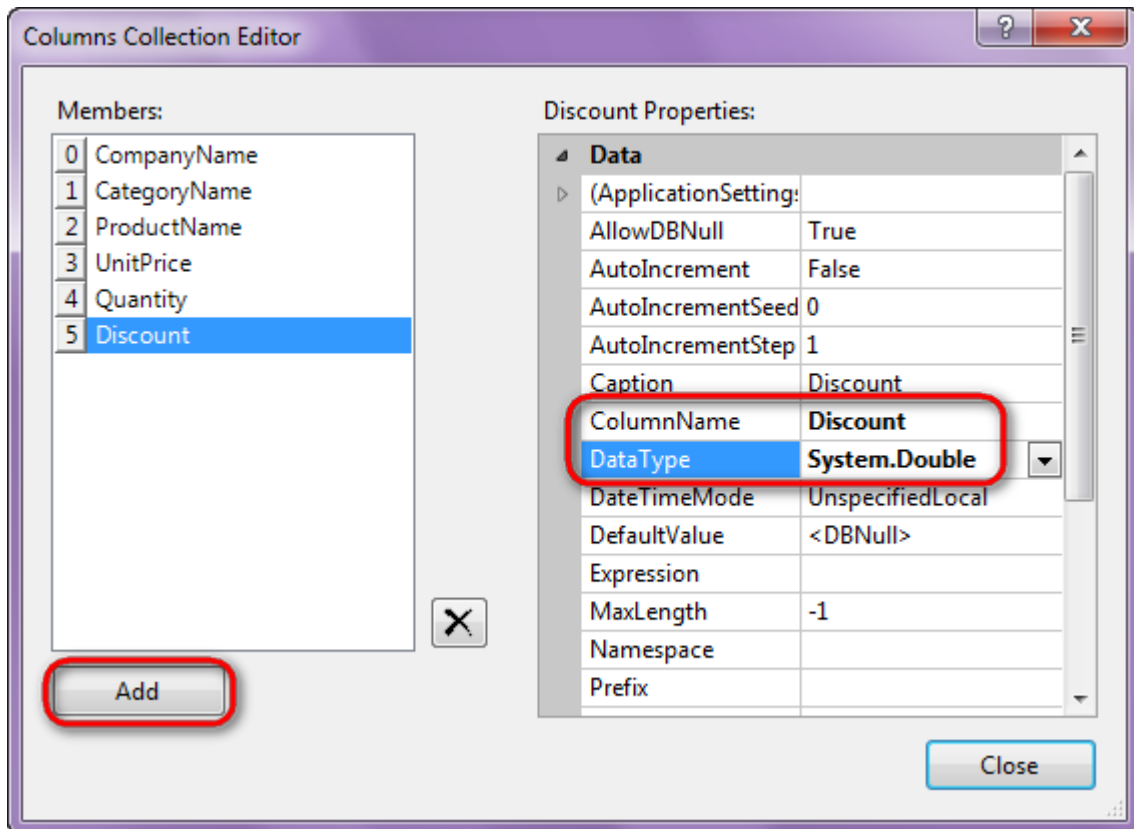
Click "Add" in order to add table. Set property TableName = Sales.



Step 5

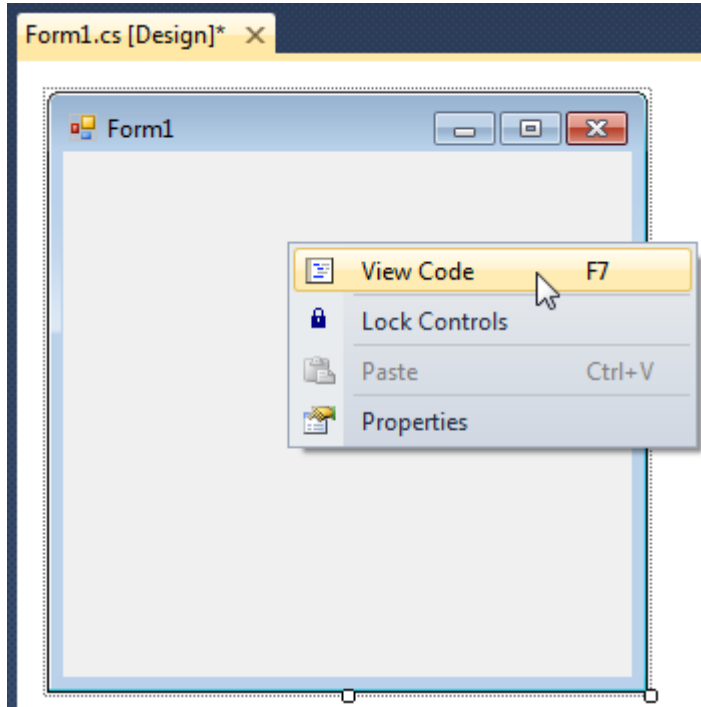
Select Columns property, click button  in order to open property editor.

Click "Add" to add a new column. Add six columns. Set ColumnName property to "CompanyName", "CategoryName", "ProductName", "UnitPrice", "Quantity", "Discount" correspondingly. For the "UnitPrice", "Quantity", "Discount" columns set DataType property to "System.Double".



Step 6

Right click on the form and select "View Code" in the context menu to view code.



Add the following code to the class constructor in order to fill data source.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row["CompanyName"] = "Alfreds Futterkiste";
}
```

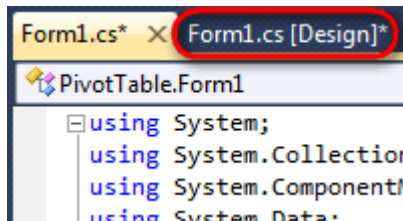


```
row["CategoryName"] = "Beverages";
row["ProductName"] = "Chai";
row["UnitPrice"] = 35.5;
row["Quantity"] = 15;
row["Discount"] = 0.05;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["CategoryName"] = "Beverages";
row["ProductName"] = "Steeleye Stout";
row["UnitPrice"] = 105;
row["Quantity"] = 5;
row["Discount"] = 0;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["CategoryName"] = "Confections";
row["ProductName"] = "Maxilaku";
row["UnitPrice"] = 2.6;
row["Quantity"] = 50;
row["Discount"] = 0;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["CategoryName"] = "Dairy Products";
row["ProductName"] = "Geitost";
row["UnitPrice"] = 15.8;
row["Quantity"] = 10;
row["Discount"] = 0.02;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["CategoryName"] = "Dairy Products";
row["ProductName"] = "Flotemysost";
row["UnitPrice"] = 240;
row["Quantity"] = 10;
row["Discount"] = 0.06;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["CategoryName"] = "Dairy Products";
row["ProductName"] = "Raclette Courdavault";
row["UnitPrice"] = 62.5;
row["Quantity"] = 15;
row["Discount"] = 0;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Karkki Oy";
row["CategoryName"] = "Beverages";
row["ProductName"] = "Chai";
row["UnitPrice"] = 35.5;
row["Quantity"] = 15;
row["Discount"] = 0.04;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Karkki Oy";
row["CategoryName"] = "Beverages";
row["ProductName"] = "Ipoh Coffee";
row["UnitPrice"] = 50.5;
row["Quantity"] = 20;
row["Discount"] = 0;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
```

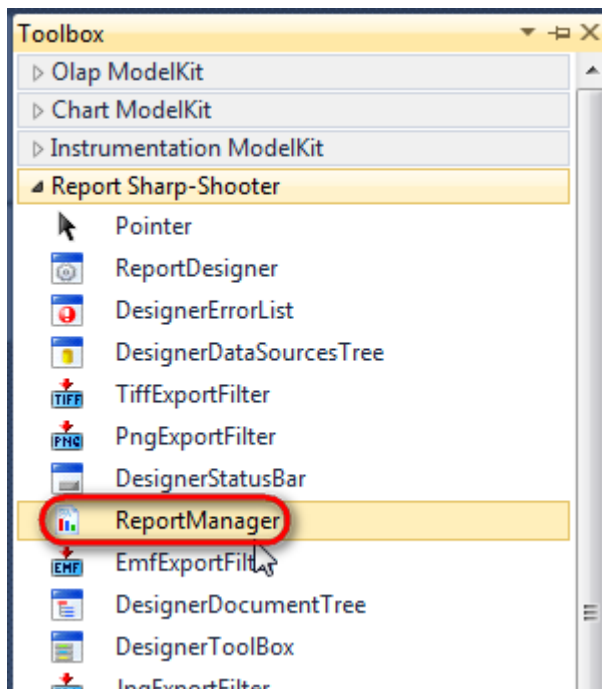
```
row["CompanyName"] = "Karkki Oy";  
row["CategoryName"] = "Grains/Cereals";  
row["ProductName"] = "Filo Mix";  
row["UnitPrice"] = 14;  
row["Quantity"] = 25;  
row["Discount"] = 0.05;  
dataTable1.Rows.Add(row);  
}
```

Step 7

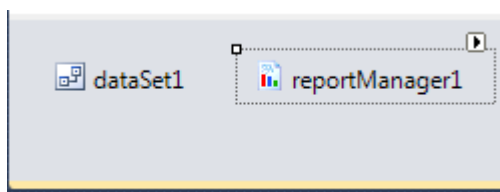
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

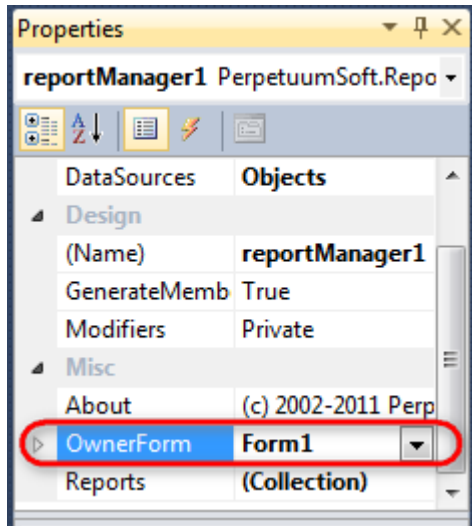


The component is available in the lower part of the window.



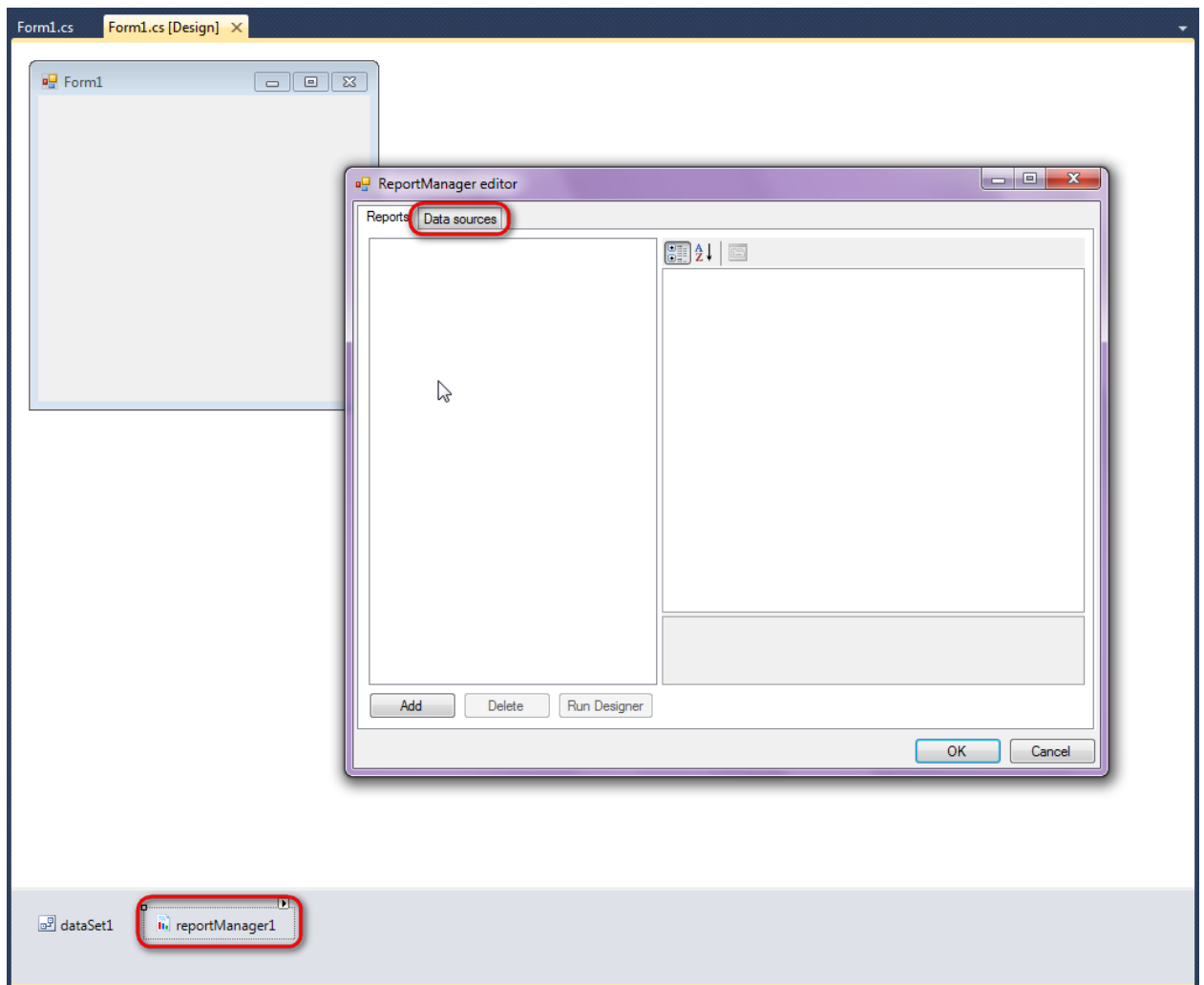
Step 8

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.

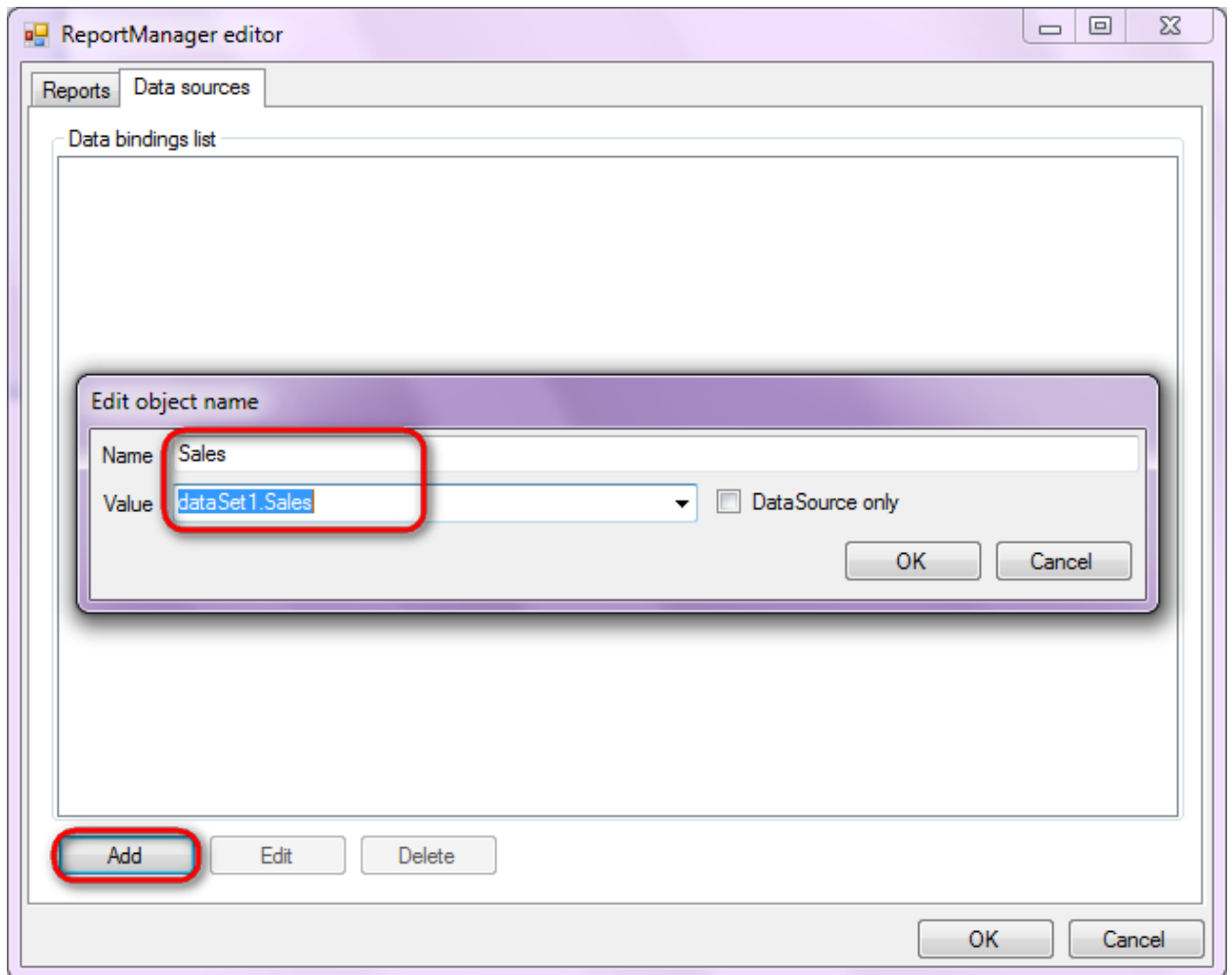


Step 9

Double click on ReportManager to open ReportManager editor.

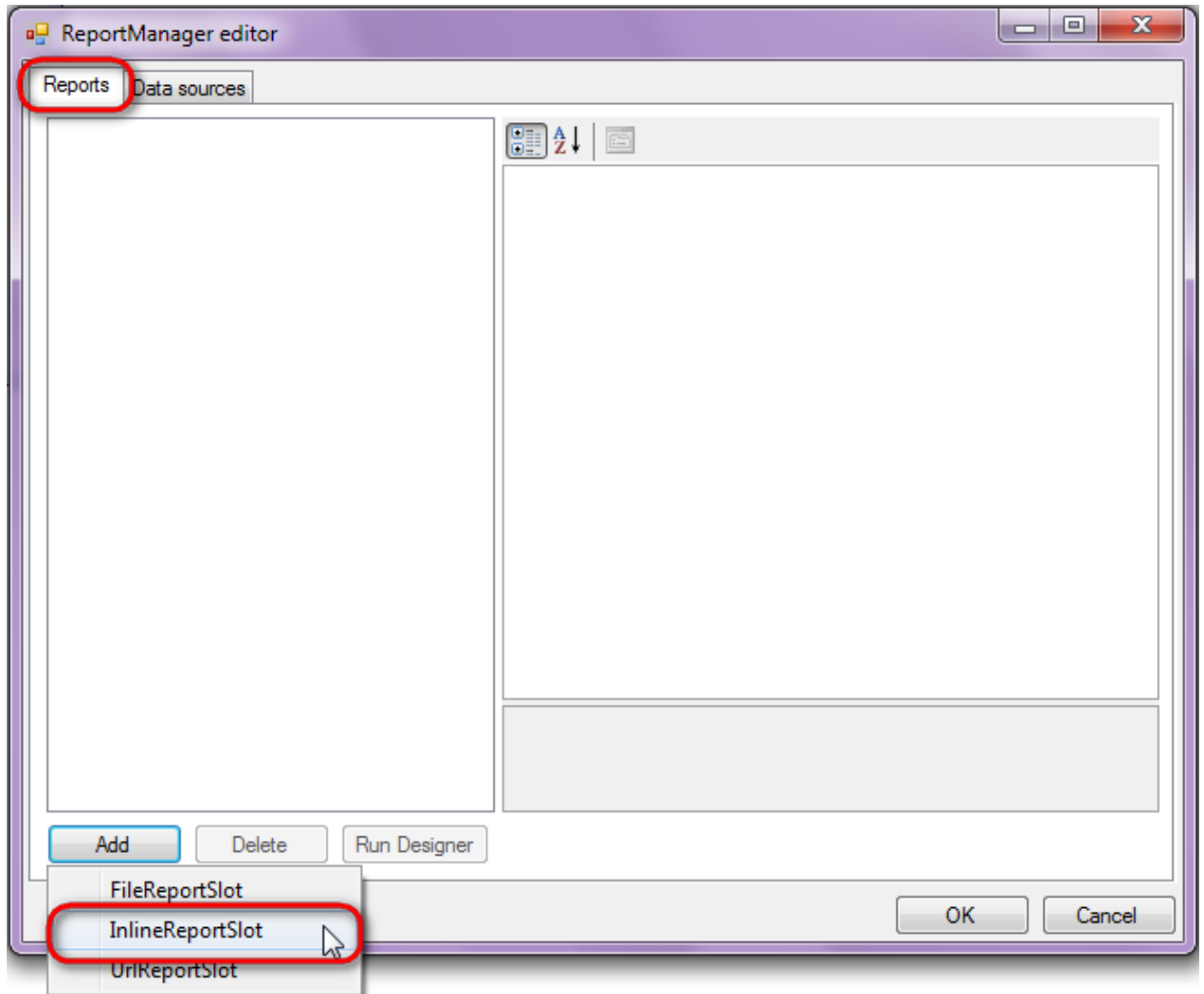


Go to "Data sources" tab, click "Add", and set data source name - "Sales", select data source value - "dataSet1.Sales".



Step 10

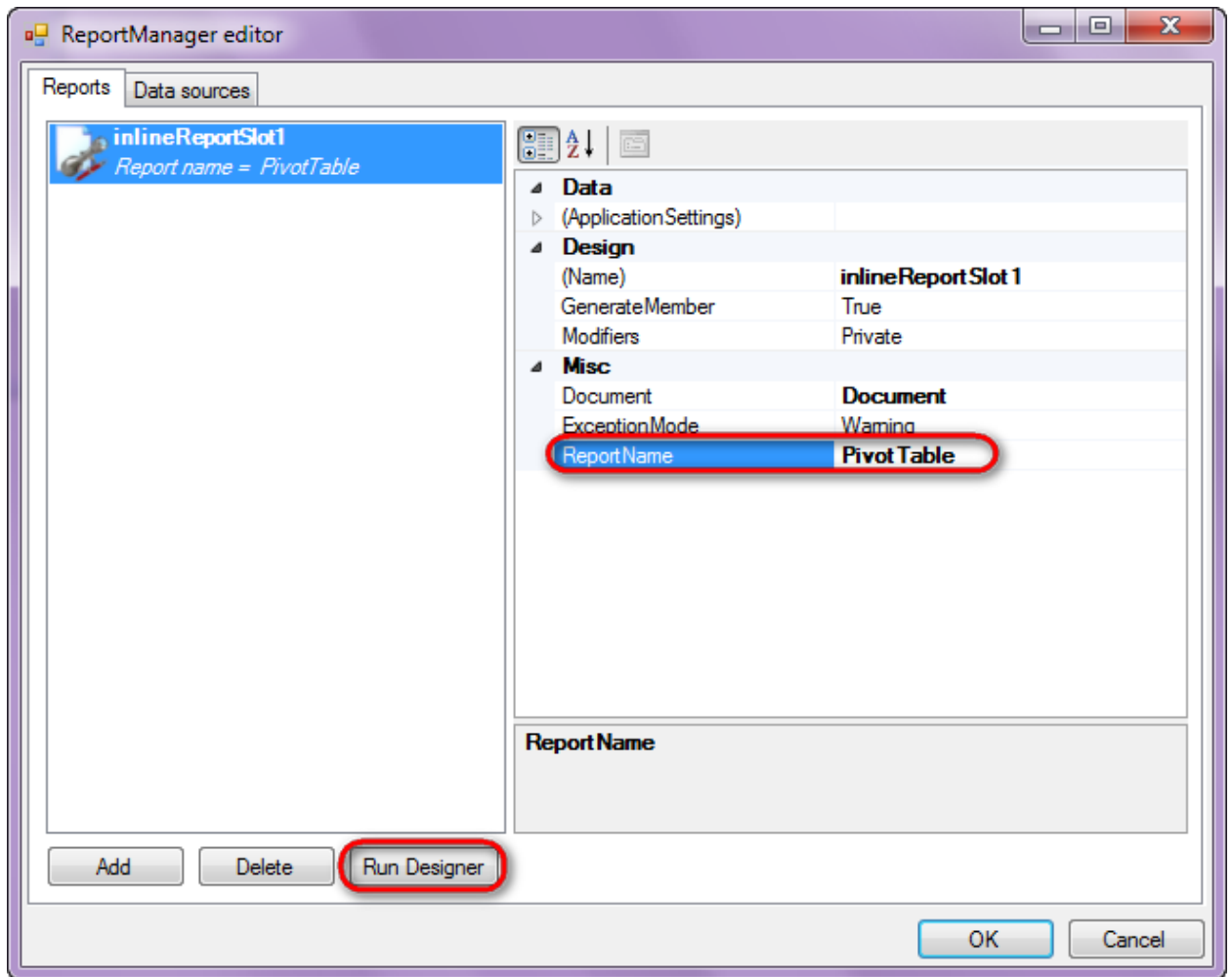
Go to "Reports" tab, click "Add" and select "InlineReportSlot".



Step 11

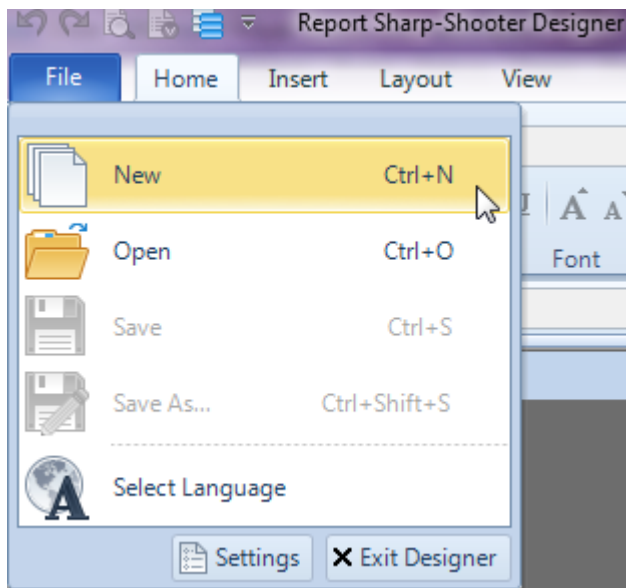
Set name of the report in the property ReportName – “PivotTable”.

Click “Run Designer” in order to open template editor – Report Designer.

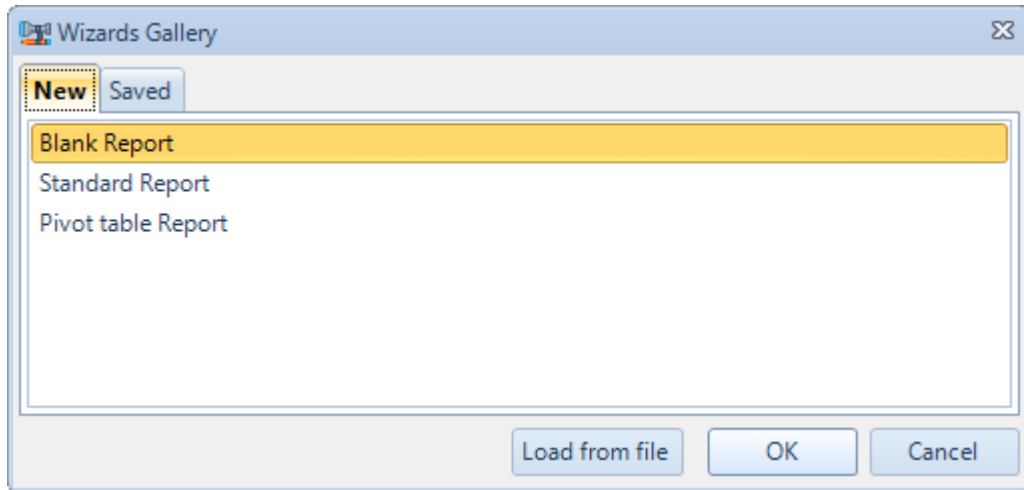


Step 12

Create new empty template – select File\New from the main menu.

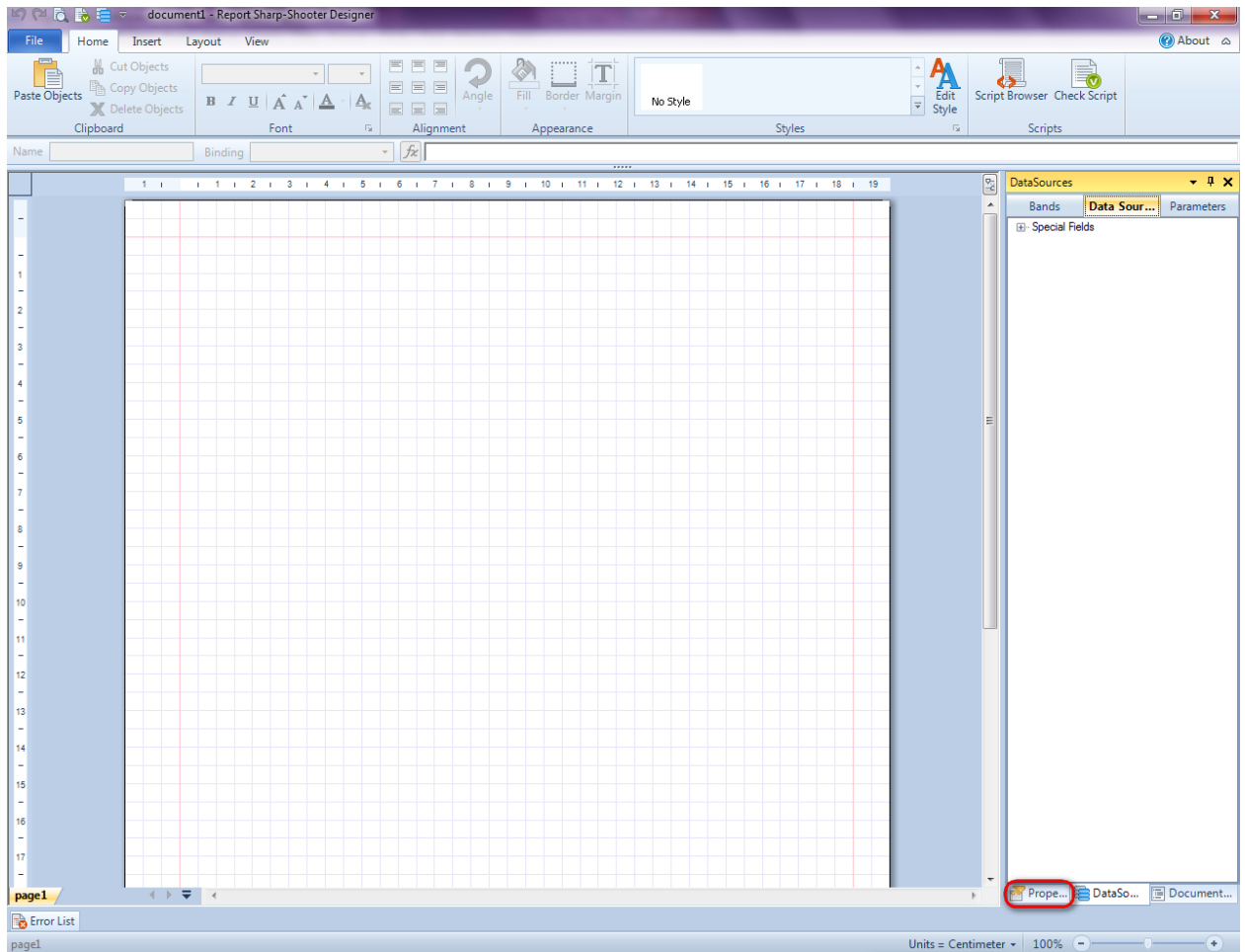


Select "Blank Report" in the Wizards Gallery and click "OK".

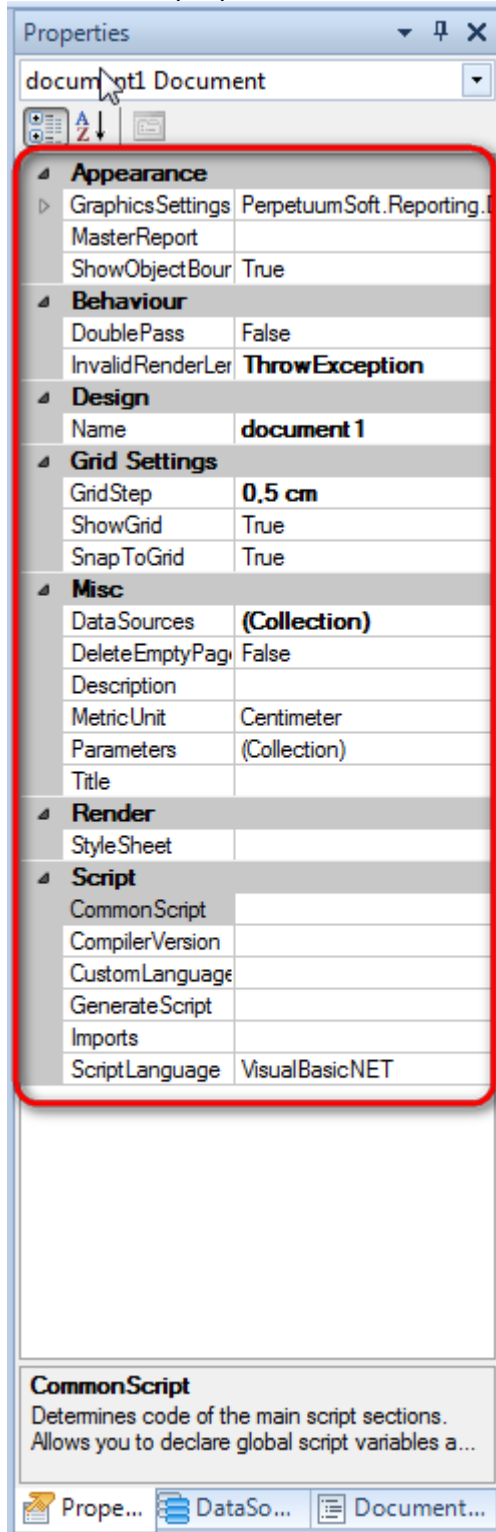


Step 13

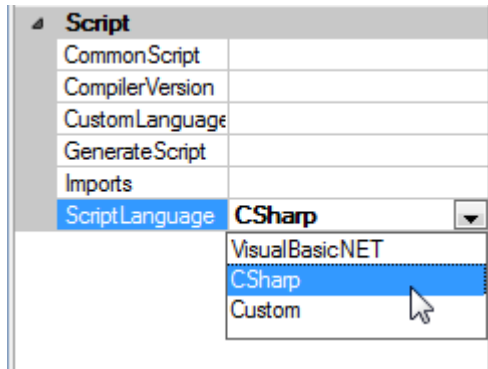
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab



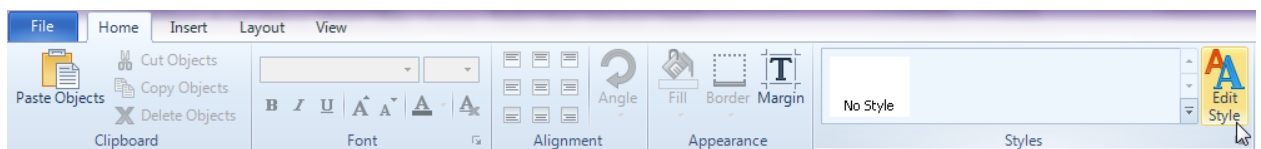
Set property ScriptLanguage = CSharp.



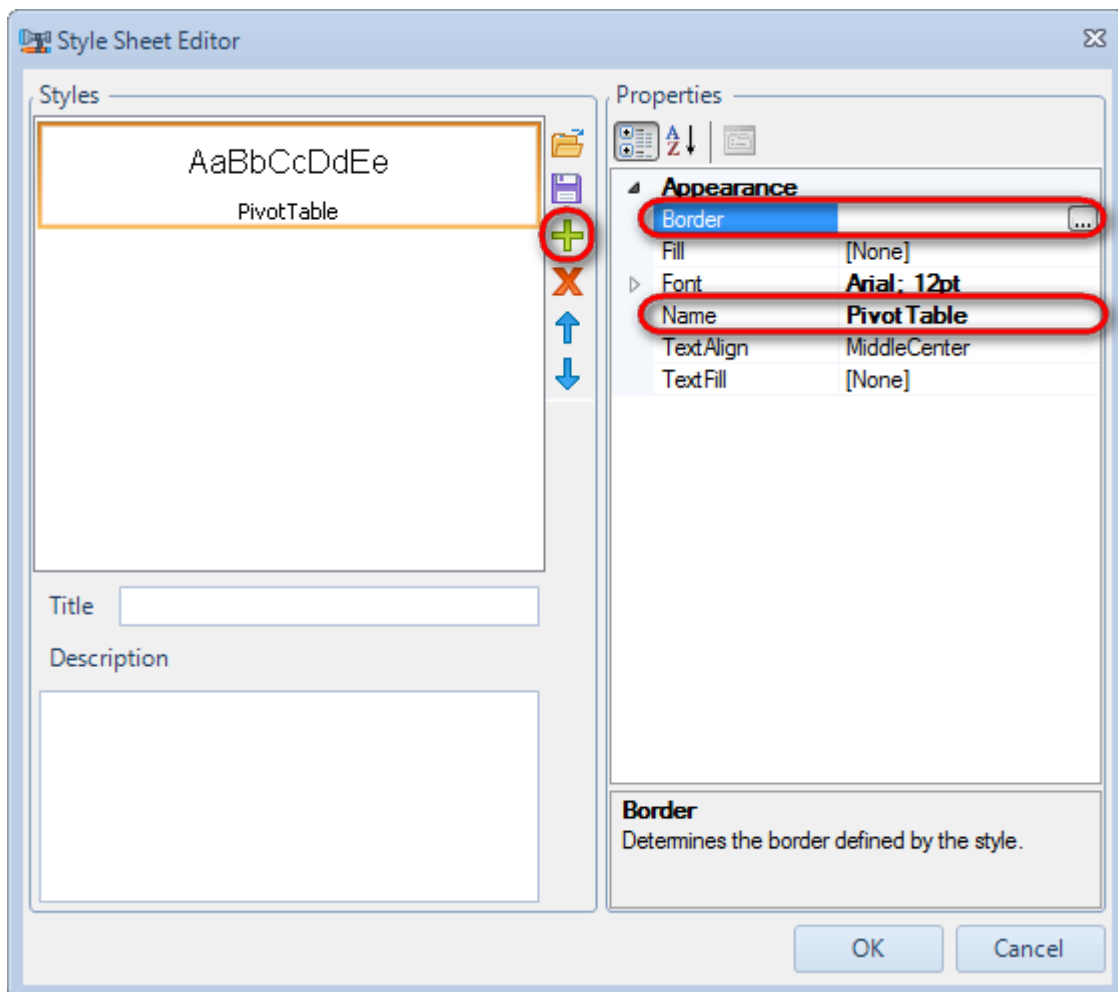
Step 14


Create style for the pivot table.

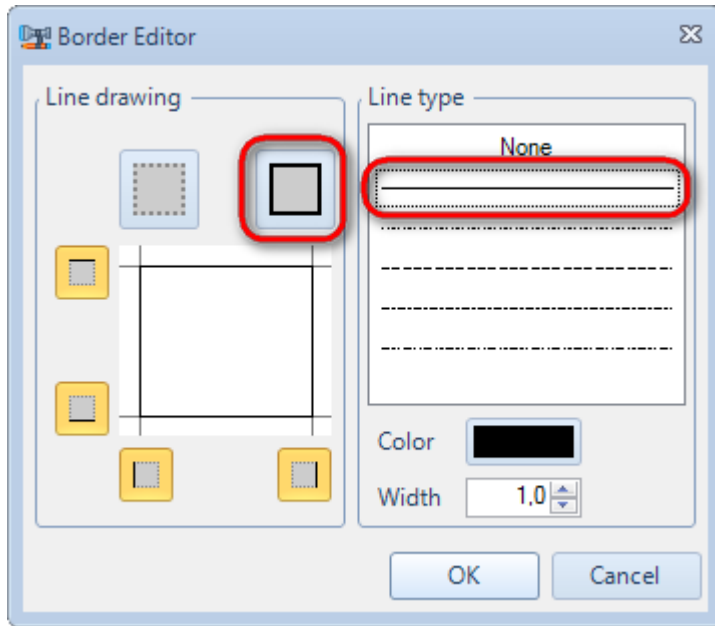
Press the "Edit Style" button on the Home tab in the group Styles.



In the Style Sheet Editor, click  button to add a new style.

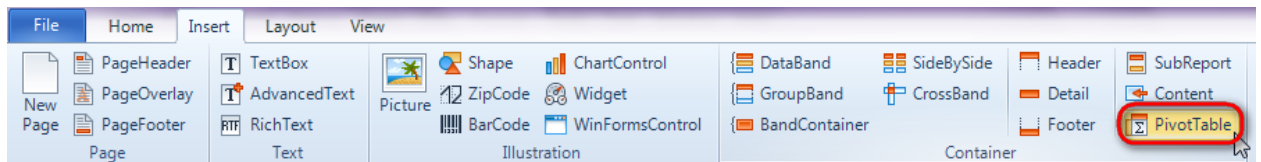


Set property Name = PivotTable. Select Border property, click button  to open property editor. Set borders.



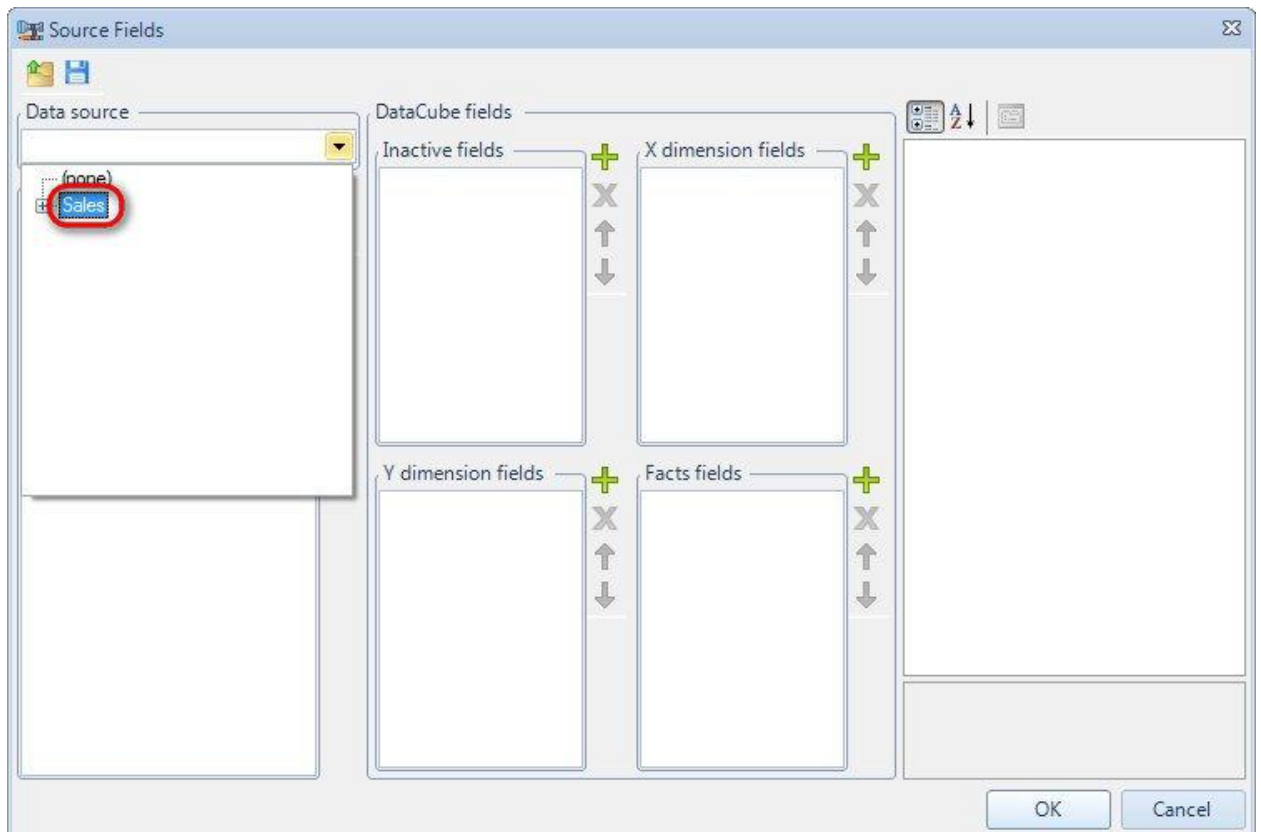
Step 15

Press "PivotTable" button on the Insert tab in the group Container.



Click on the template area to add PivotTable to the template.

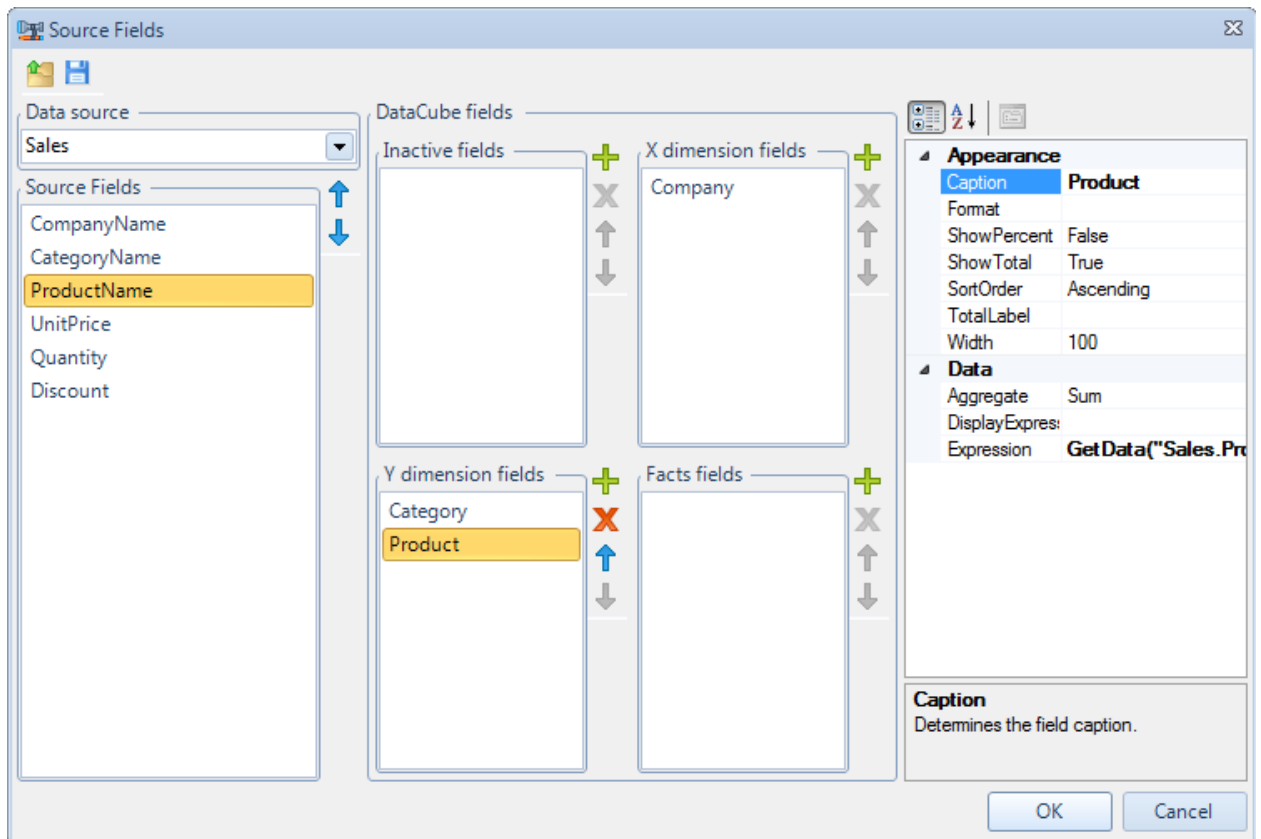
Set data source in the property DataSource = Sales.






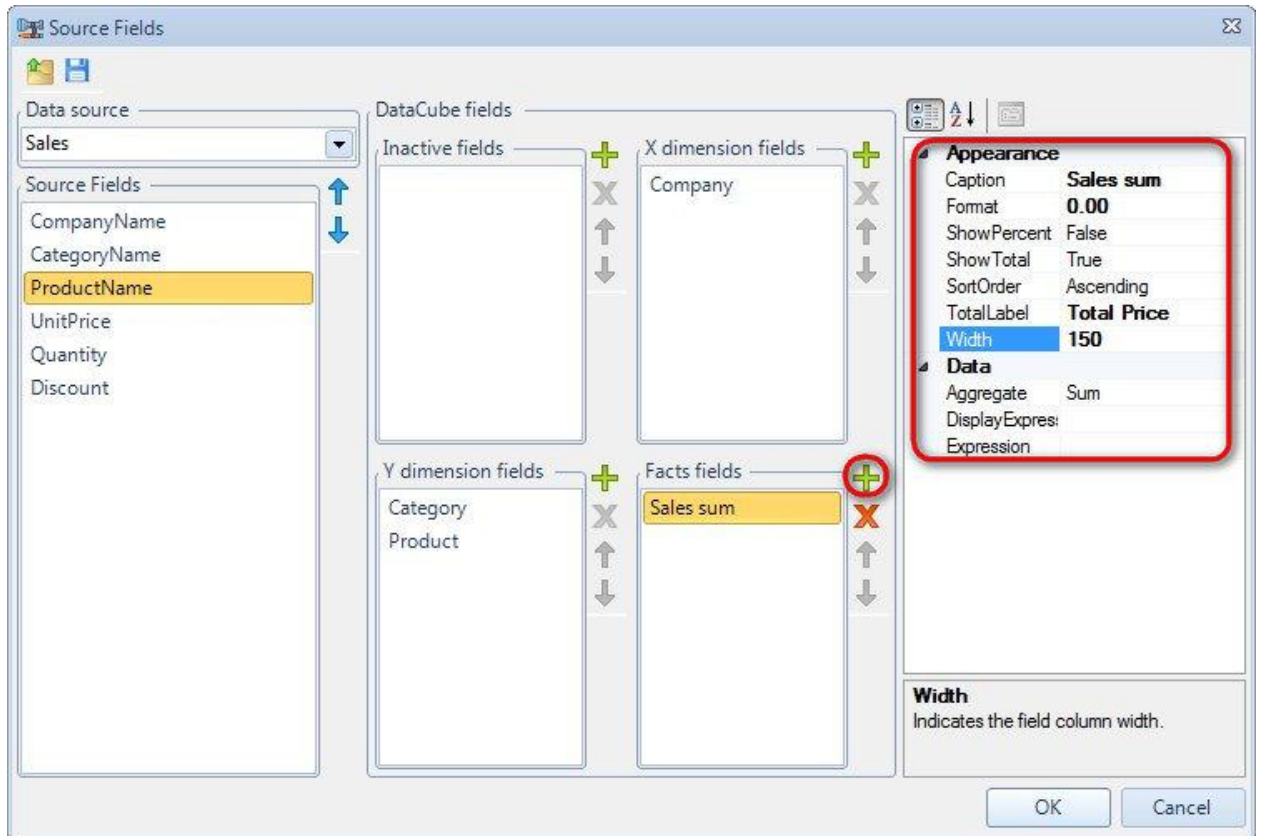
Step 16

Drag and drop "CompanyName" field from the "Source fields" list to "X dimension fields". Drag and drop "CategoryName" and "ProductName" to "Y dimension fields". Edit Caption property and set only "Company", "Category", "Product".



Step 17

In the Source fields editor, click  button for the "Facts fields". A new field will be added and its properties will be visible in the right part of the window.




Set the following properties:

Caption = Sales sum

Format = 0.00

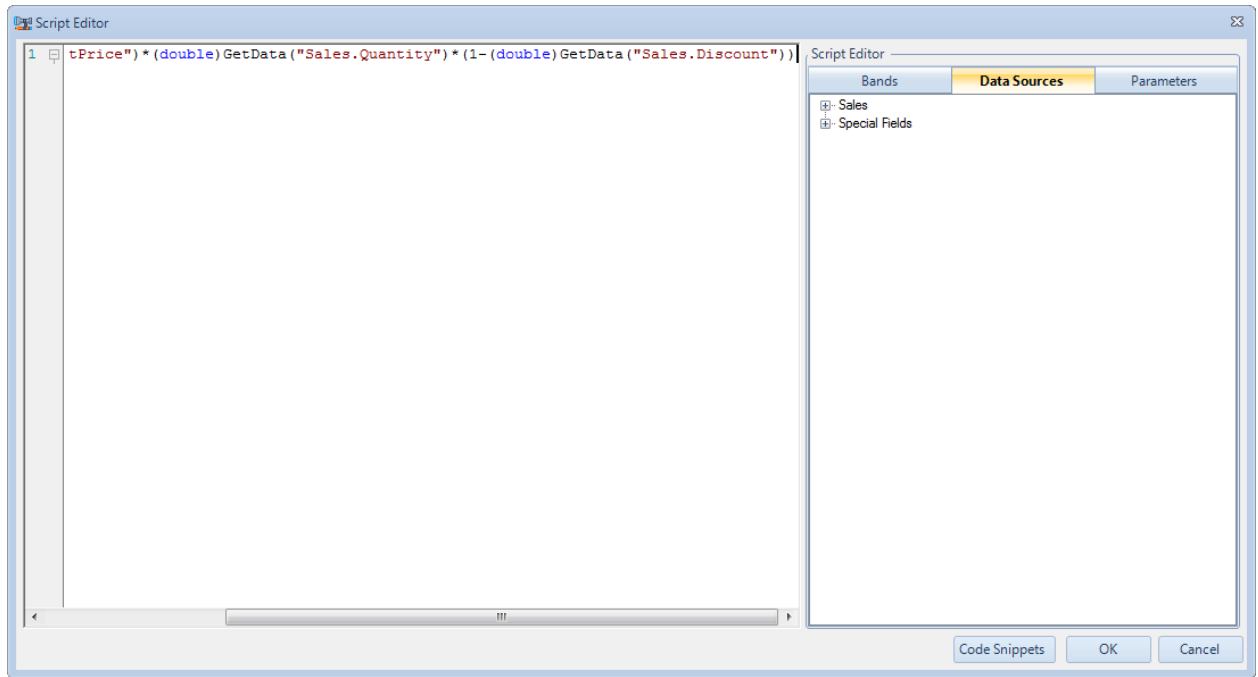
TotalLabel = Total price

Width = 150.

Select Expression property, click button  to open Script Editor.

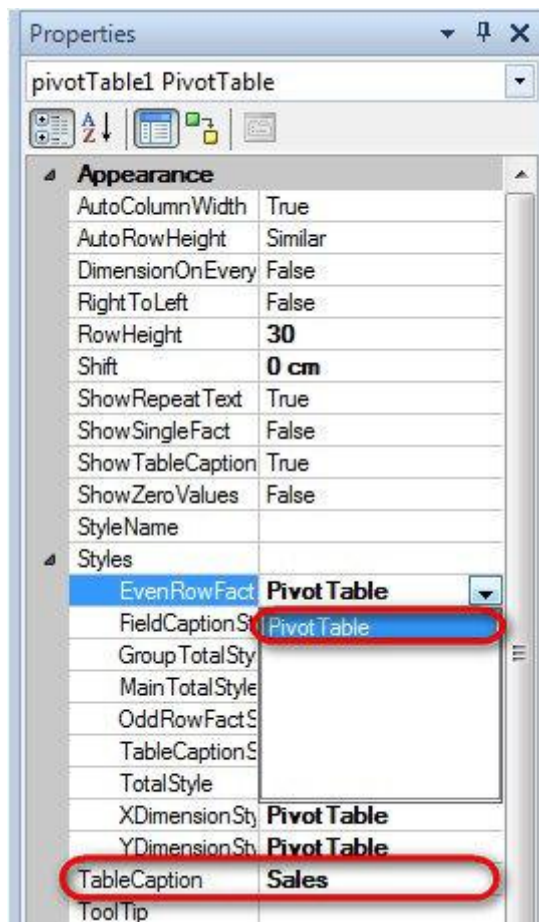
Set the following expression in the editor:

```
"(double)GetData("Sales.UnitPrice")*(double)GetData("Sales.Quantity")*(1-(double)GetData("Sales.Discount"))"
```



Step 18

Select PivotTable element. Expand Style list on the property grid. Select PivotTable style for all types of cells. Set the TableCaption property = Sales.



Report template should look as follows:



		Sales	
		Company	
		Item	Total
Category	Product	Sales sum	Sales sum
Group	Item 1	0.0	0.0
	Item 2	0.0	0.0
	Total	0.0	0.0
Total		0.0	0.0

Step 19

Save template, close Report Designer.

Step 20

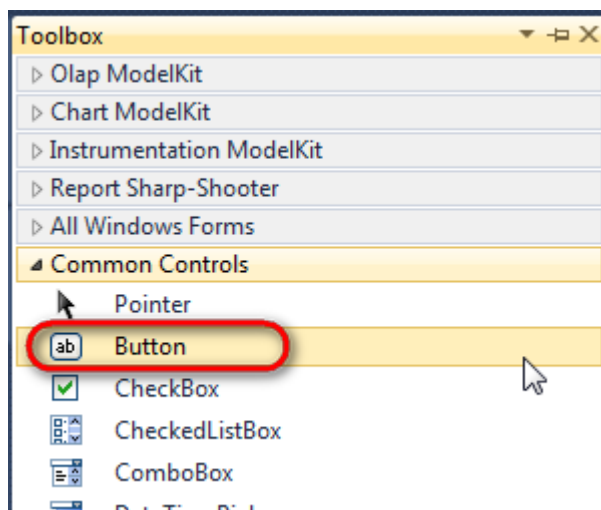
Add code to display report to the class constructor.

```
public Form1 ()
{
    InitializeComponent ();
    DataRow row = dataTable1.NewRow ();
    row ["CompanyName"] = "Alfreds Futterkiste";
    row ["CategoryName"] = "Beverages";
    row ["ProductName"] = "Chai";
    row ["UnitPrice"] = 35.5;
    row ["Quantity"] = 15;
    row ["Discount"] = 0.05;
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["CompanyName"] = "Alfreds Futterkiste";
    row ["CategoryName"] = "Beverages";
    row ["ProductName"] = "Steeleye Stout";
    row ["UnitPrice"] = 105;
    row ["Quantity"] = 5;
    row ["Discount"] = 0;
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["CompanyName"] = "Alfreds Futterkiste";
    row ["CategoryName"] = "Confections";
    row ["ProductName"] = "Maxilaku";
    row ["UnitPrice"] = 2.6;
    row ["Quantity"] = 50;
    row ["Discount"] = 0;
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["CompanyName"] = "Alfreds Futterkiste";
    row ["CategoryName"] = "Dairy Products";
    row ["ProductName"] = "Geitost";
    row ["UnitPrice"] = 15.8;
    row ["Quantity"] = 10;
    row ["Discount"] = 0.02;
    dataTable1.Rows.Add (row);
    row = dataTable1.NewRow ();
    row ["CompanyName"] = "Alfreds Futterkiste";
    row ["CategoryName"] = "Dairy Products";
    row ["ProductName"] = "Flotemysost";
    row ["UnitPrice"] = 240;
    row ["Quantity"] = 10;
    row ["Discount"] = 0.06;
    dataTable1.Rows.Add (row);
}
```

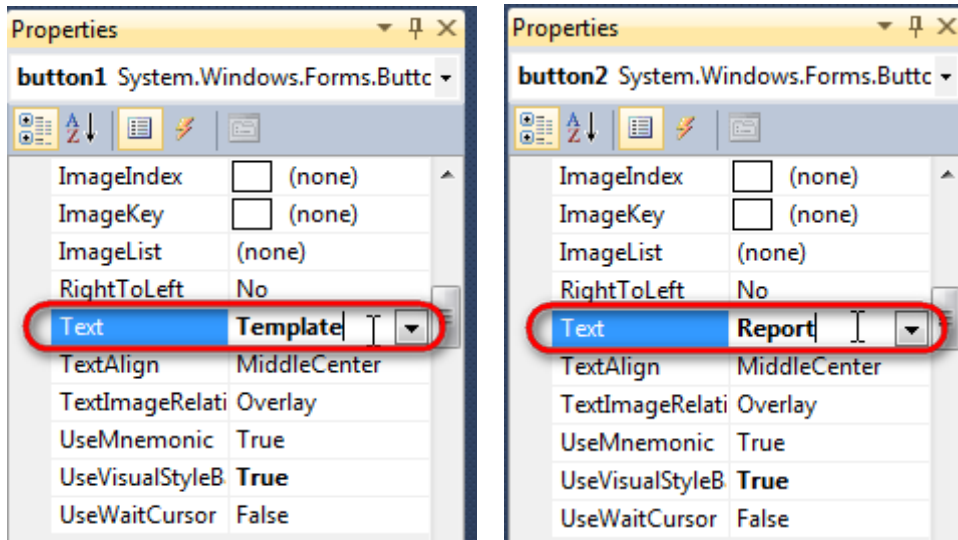
```
row = dataTable1.NewRow();
row["CompanyName"] = "Alfreds Futterkiste";
row["CategoryName"] = "Dairy Products";
row["ProductName"] = "Raclette Courdavault";
row["UnitPrice"] = 62.5;
row["Quantity"] = 15;
row["Discount"] = 0;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Karkki Oy";
row["CategoryName"] = "Beverages";
row["ProductName"] = "Chai";
row["UnitPrice"] = 35.5;
row["Quantity"] = 15;
row["Discount"] = 0.04;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Karkki Oy";
row["CategoryName"] = "Beverages";
row["ProductName"] = "Ipoh Coffee";
row["UnitPrice"] = 50.5;
row["Quantity"] = 20;
row["Discount"] = 0;
dataTable1.Rows.Add(row);
row = dataTable1.NewRow();
row["CompanyName"] = "Karkki Oy";
row["CategoryName"] = "Grains/Cereals";
row["ProductName"] = "Filo Mix";
row["UnitPrice"] = 14;
row["Quantity"] = 25;
row["Discount"] = 0.05;
dataTable1.Rows.Add(row);
using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm(inlineReportSlot1))
{
    previewForm.WindowState = FormWindowState.Maximized;
    inlineReportSlot1.RenderDocument();
    previewForm.ShowDialog(this);
}
}
```

Step 21

Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



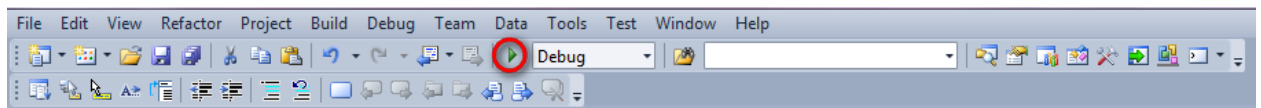
Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

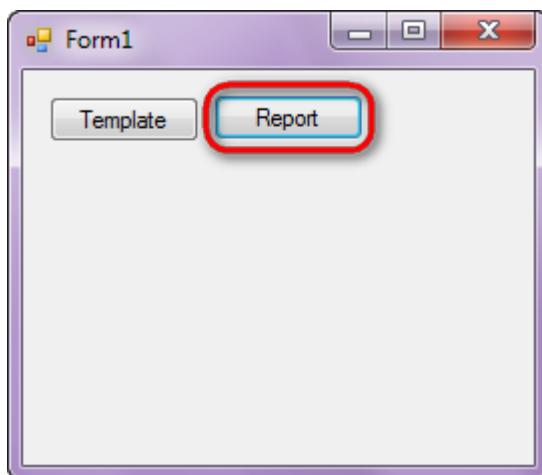
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

Click "Start Debugging" on the Visual Studio toolbar in order to start application.



Click the "Report" button in the opened application window.



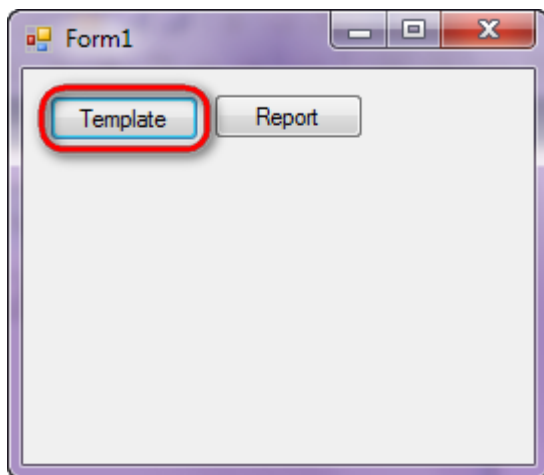
Generated report is viewed in the Report Viewer.



The screenshot shows a report preview window titled "document1 - Preview report". The report is a pivot table with the following data:

		Sales		
		Company		
Category	Product	Alfreds Futterkiste	Karkki Oy	Total
Beverages	Chai	505,88	511,20	1017,08
	Ipoh Coffee		1010,00	1010,00
	Steeleye Stout	525,00		525,00
	Total	1030,88	1521,20	2552,08
Confections	Maxilaku	130,00		130,00
	Total	130,00		130,00
Dairy Products	Flotemysost	2256,00		2256,00
	Geitost	154,84		154,84
	Raclette Courdavault	937,50		937,50
	Total	3348,34		3348,34
Grains/Cereals	Filo Mix		332,50	332,50
	Total		332,50	332,50
Total		4509,22	1853,70	6362,92

To edit report template, close Report Viewer and click "Template" on the application form.



Similar application sample is located in the following folder "\\Perpetuum Software\\Net ModelKit Suite\\Samples\\Report Sharp-Shooter\\CSharp\\PivotTableGettingStarted".

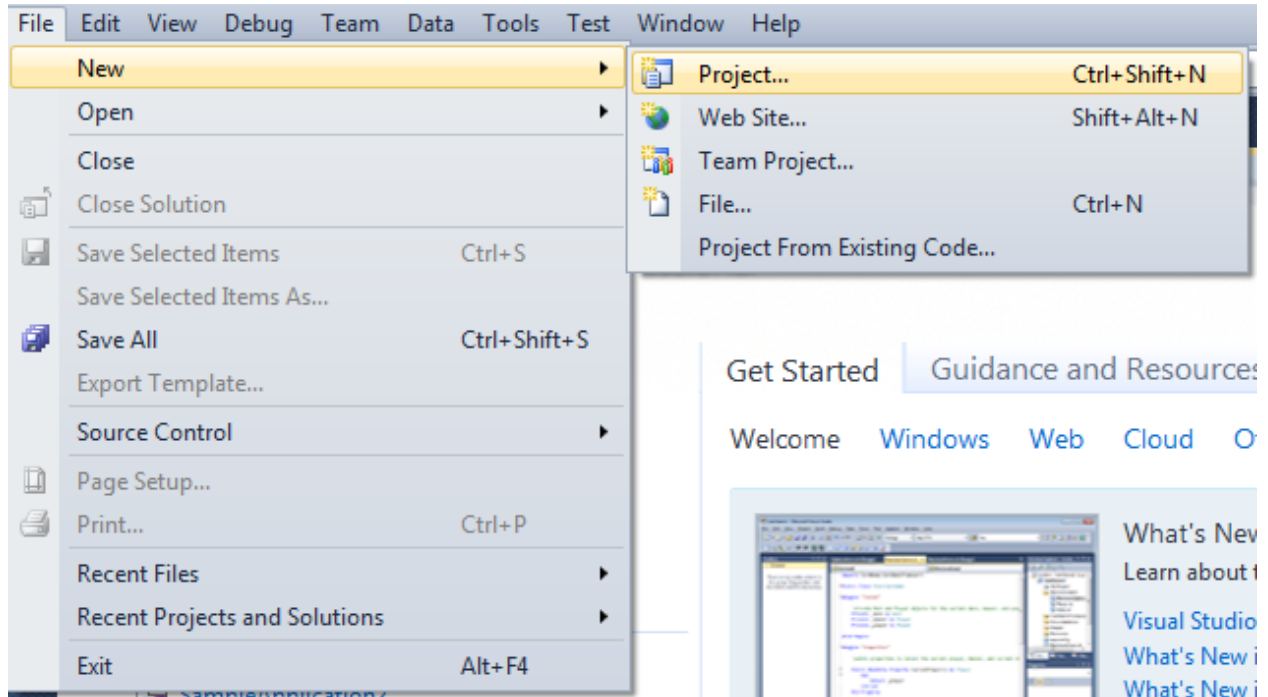


Subreports

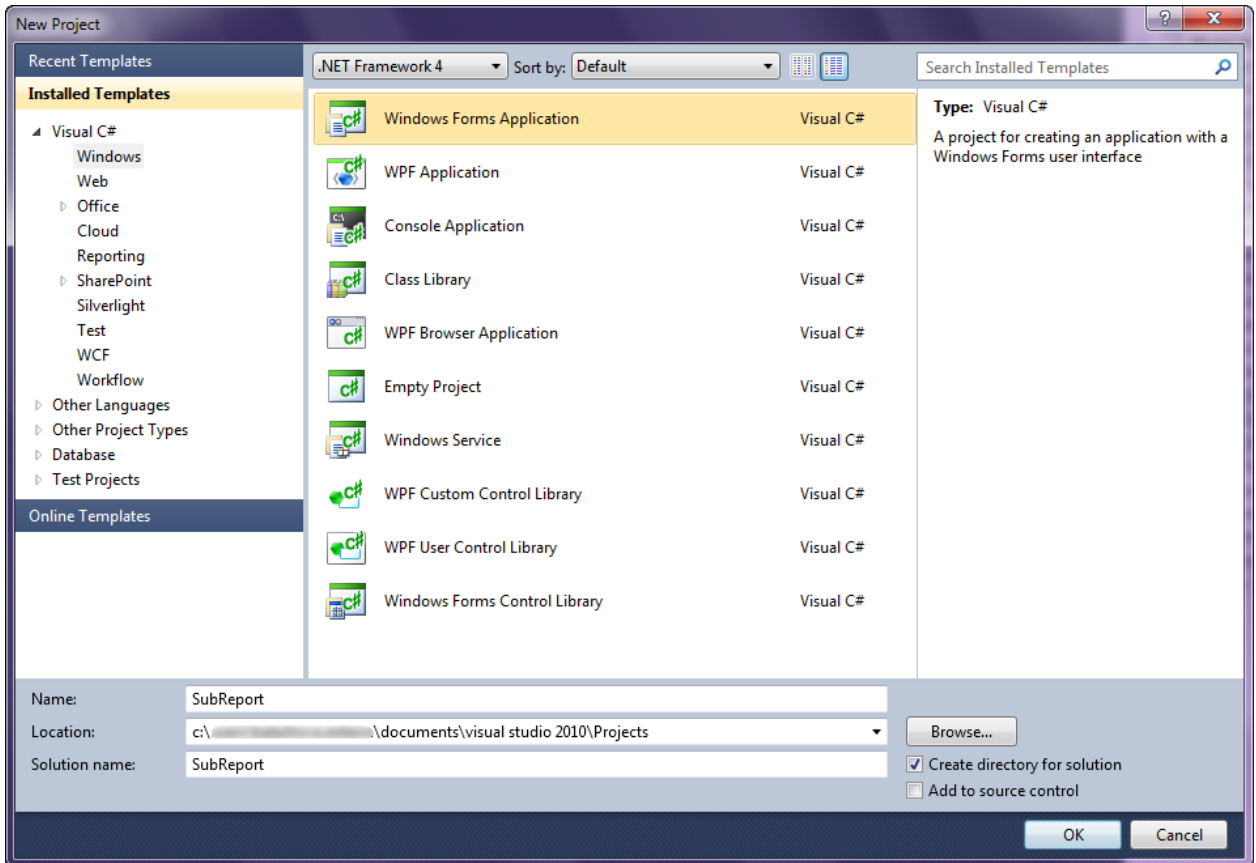
Template of a report containing three subreports – reports containing charts, gauges and shapes. Any report can be used as subreport. Before you create master report, design necessary templates and save them to files.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

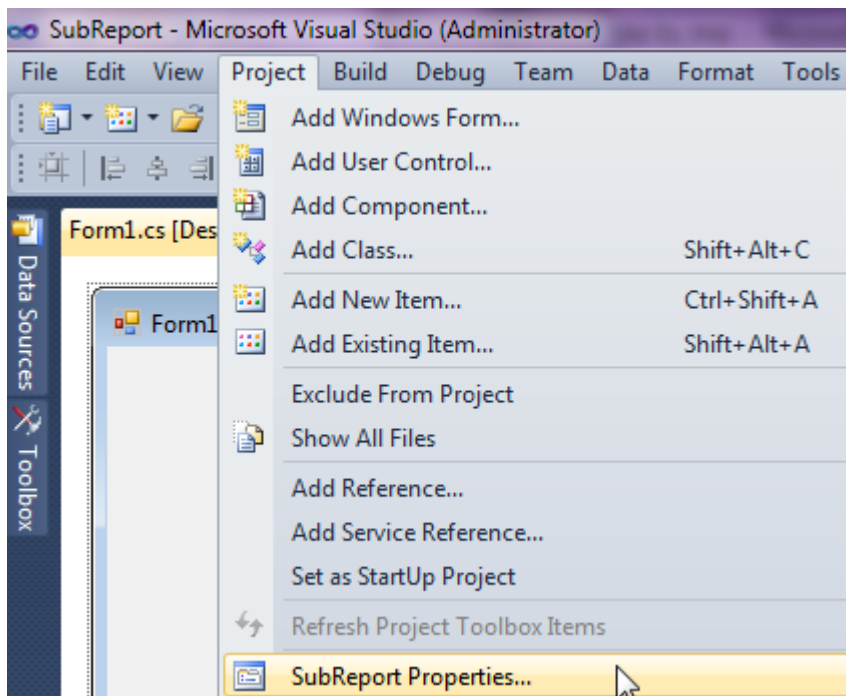


Select Windows Forms Application, set project name – “SubReport”, set directory to save the project to.

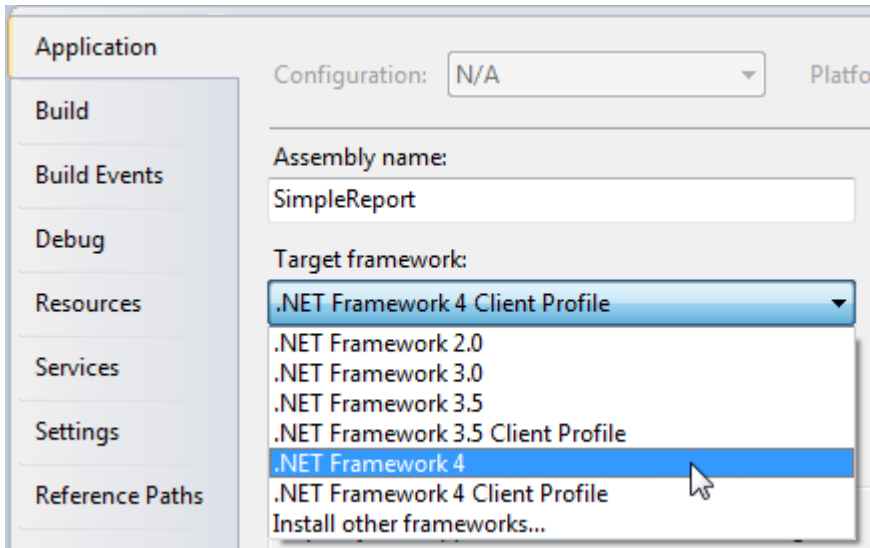


Step 2

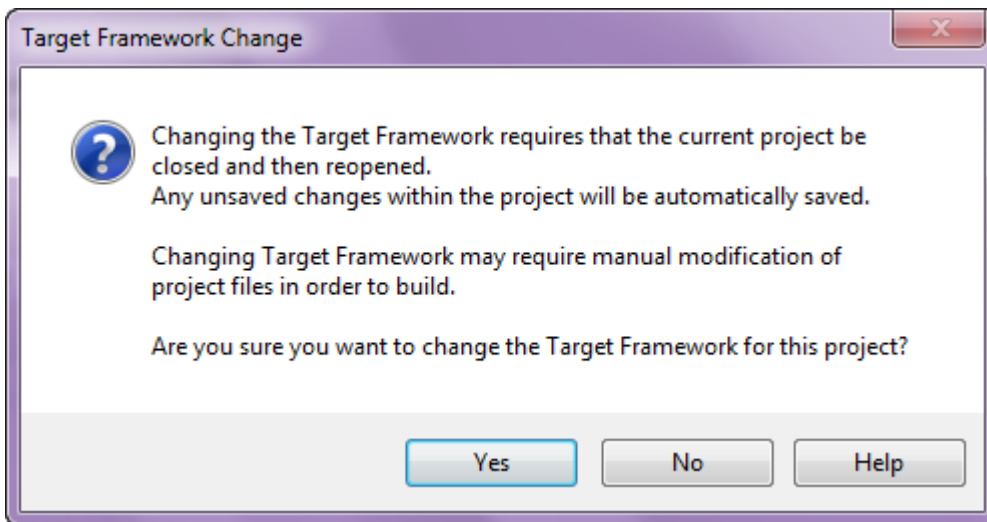
Change the project properties. Select the Project\SubReport Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

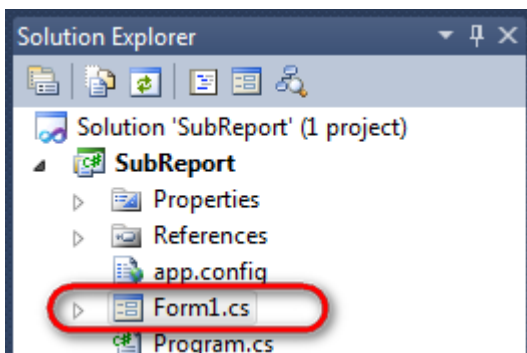


In the opened window press the “Yes” button.

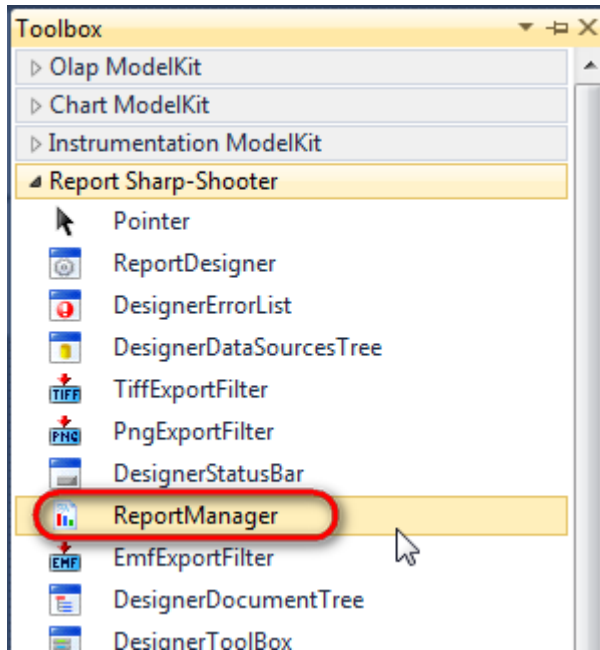


Step 3

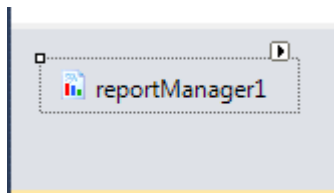
Open main form of the application by double click on the “Form1.cs” in the Solution Explorer.



Click on the “ReportManager” on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

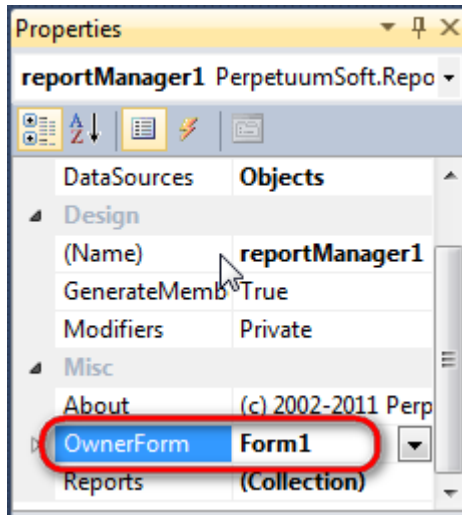


The component is available in the lower part of the window.



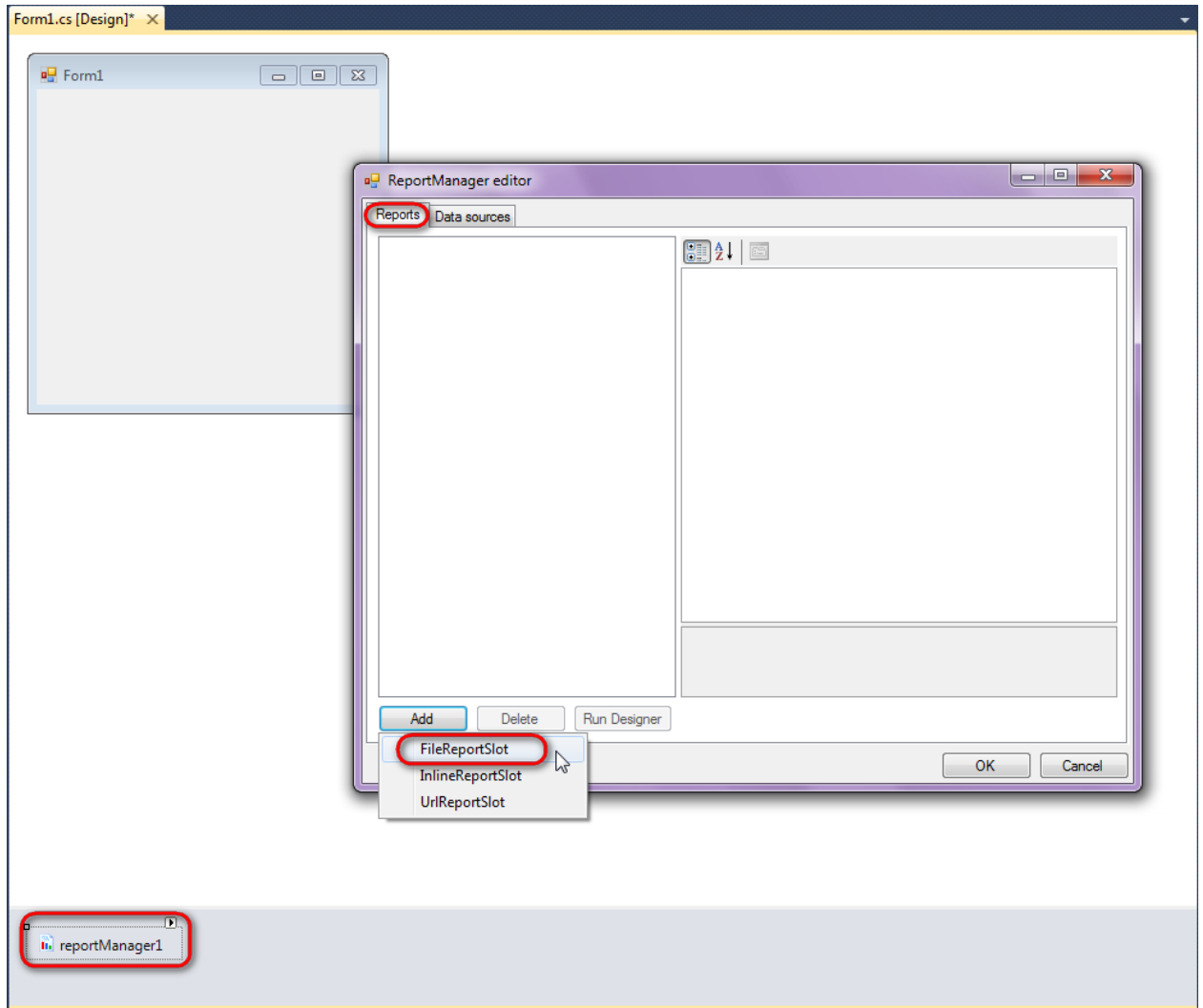
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.




Step 5

Double click on ReportManager to open ReportManager editor.

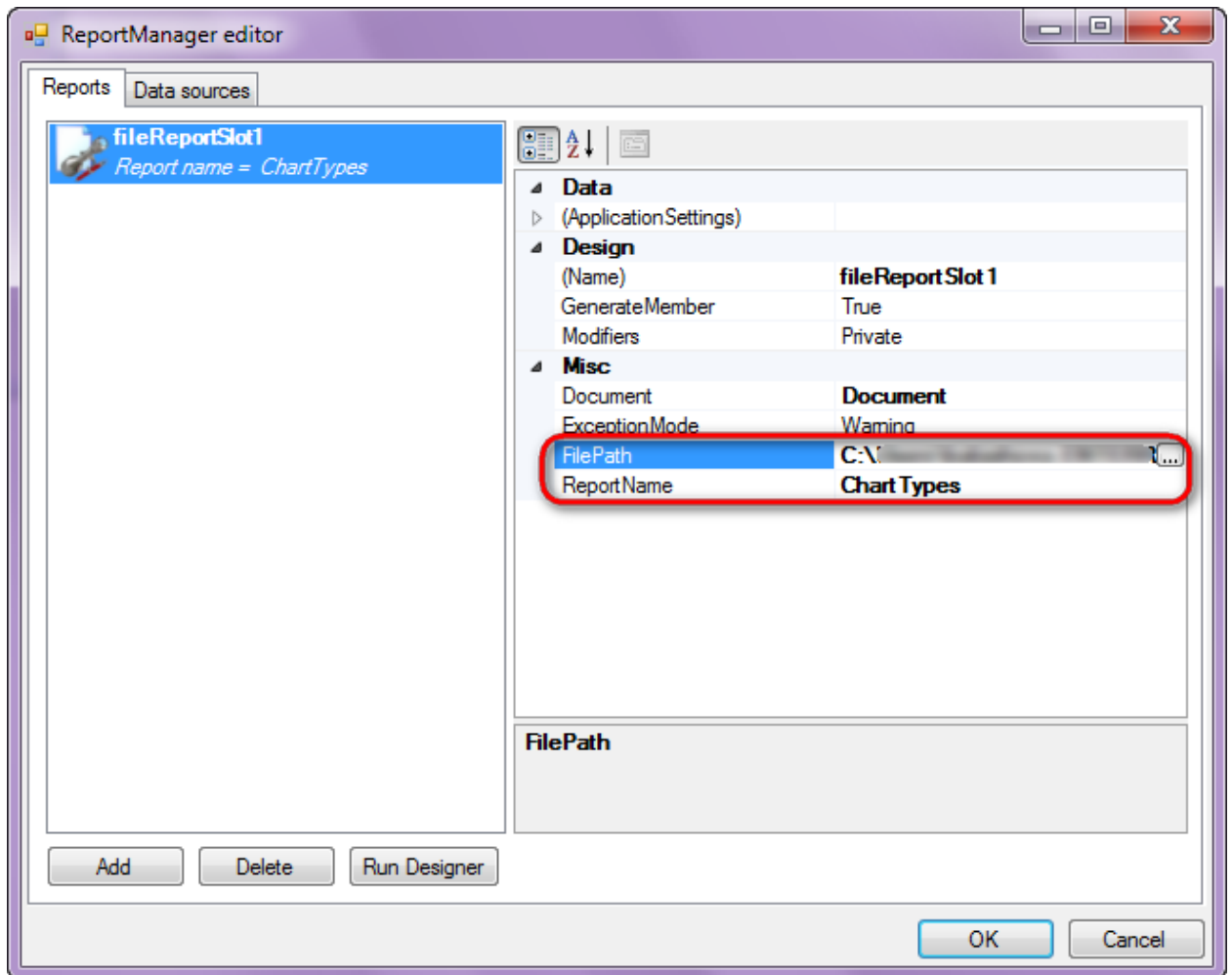


On the "Reports" tab, click "Add" and select "FileReportSlot".

Step 6

Set name of the report in property ReportName = "ChartTypes". Select FilePath property, click button  to open dialog to select report template.

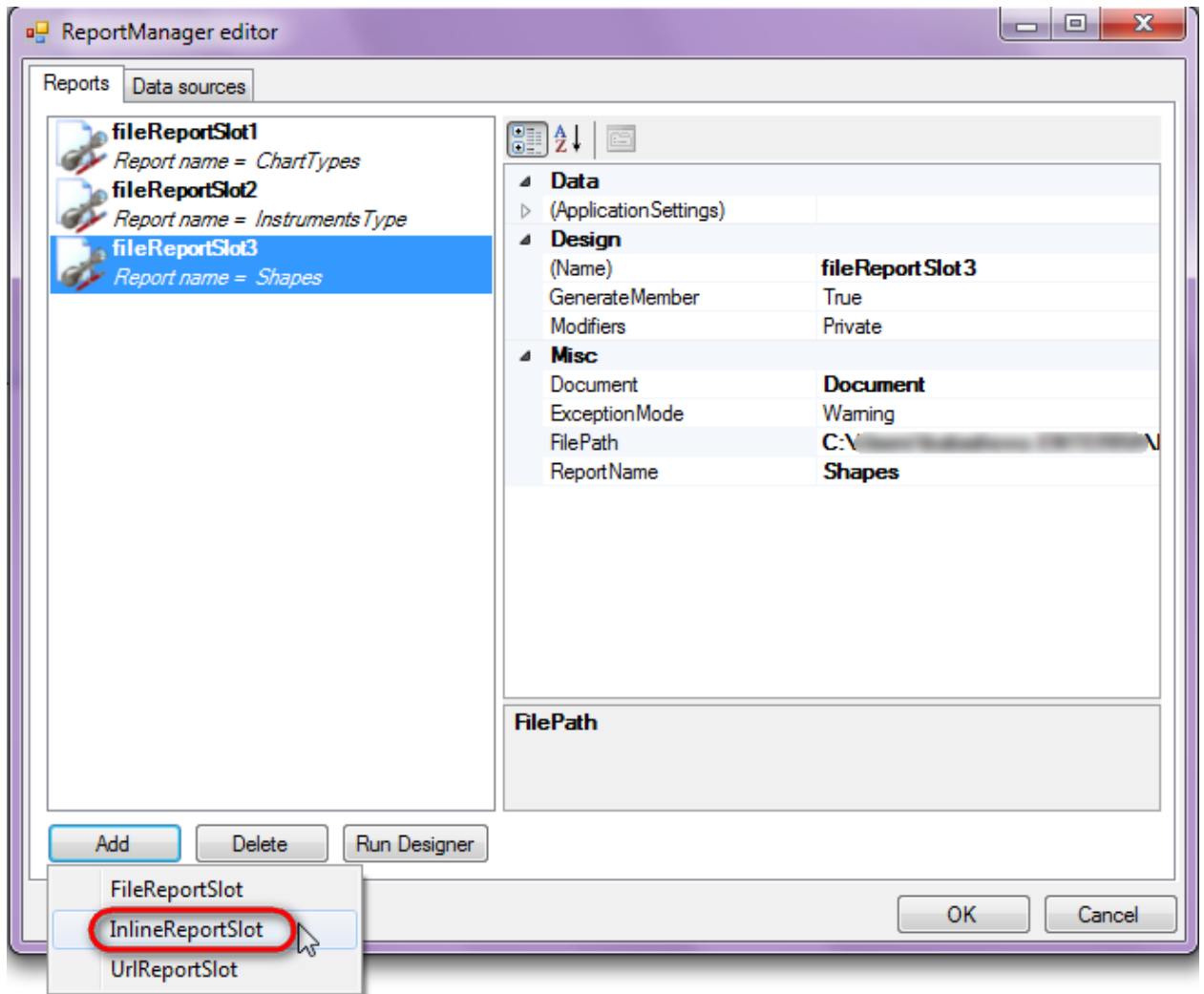
Select previously saved report template "ChartTypes" and click "OK".



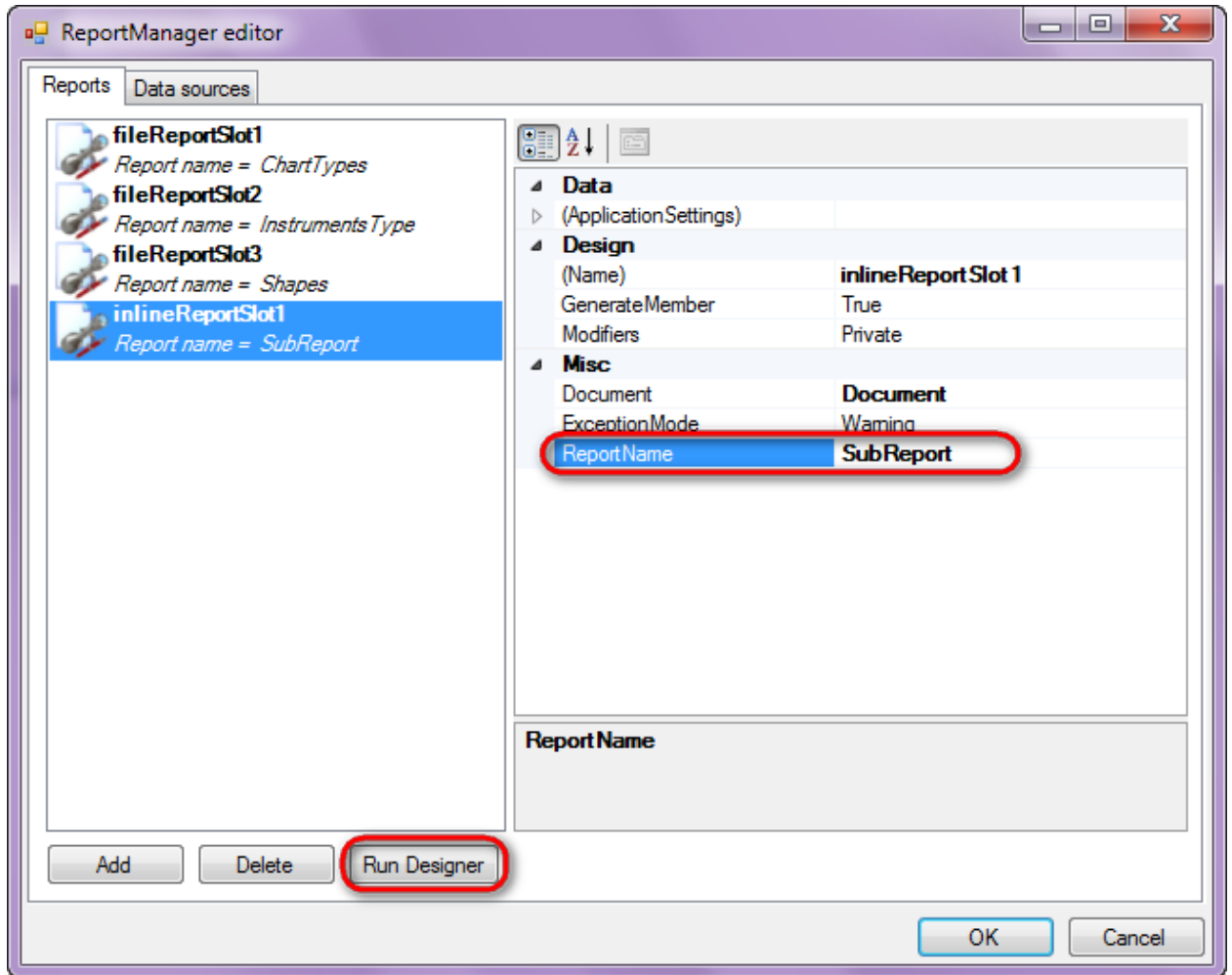
In the same way add two more report templates previously saved, set properties ReportName = InstrumentsType and ReportName = Shapes.

Step 7

Click "Add" and select "InlineReportSlot".



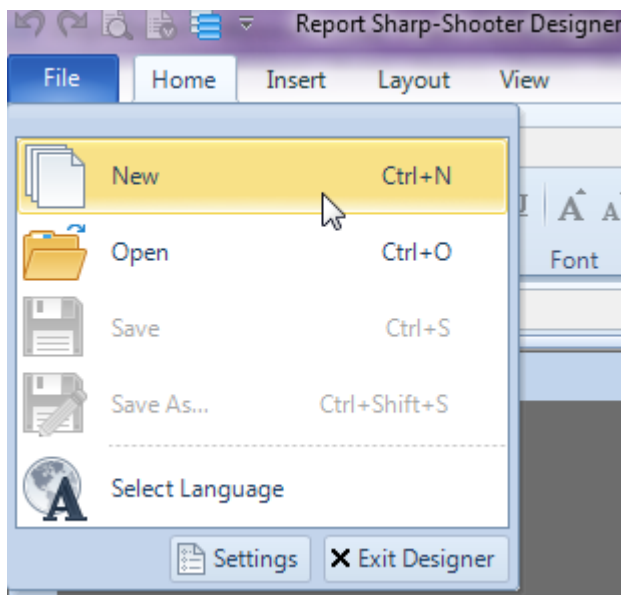
Set property ReportName = SubReport.



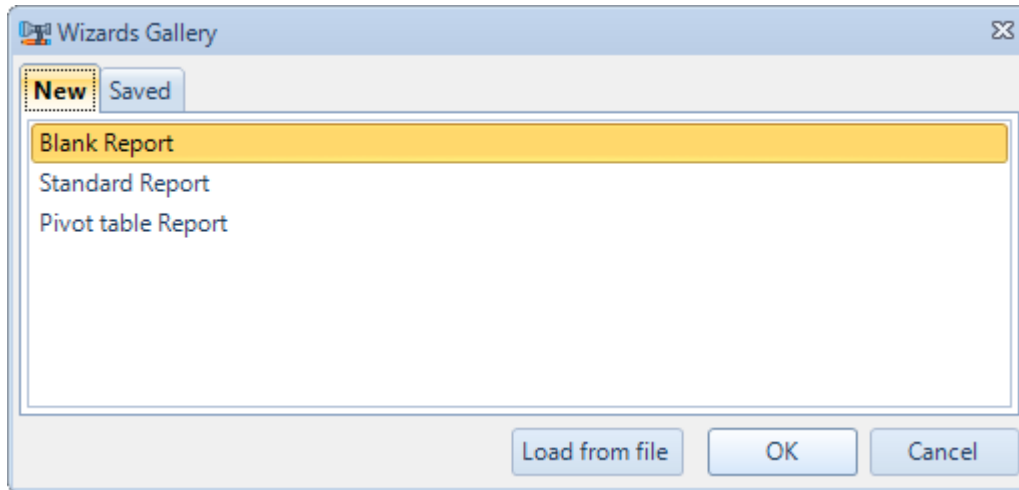
Click "Run Designer" in order to open template editor - Report Designer.

Step 8

Create new empty template – select item File\New from the main menu.

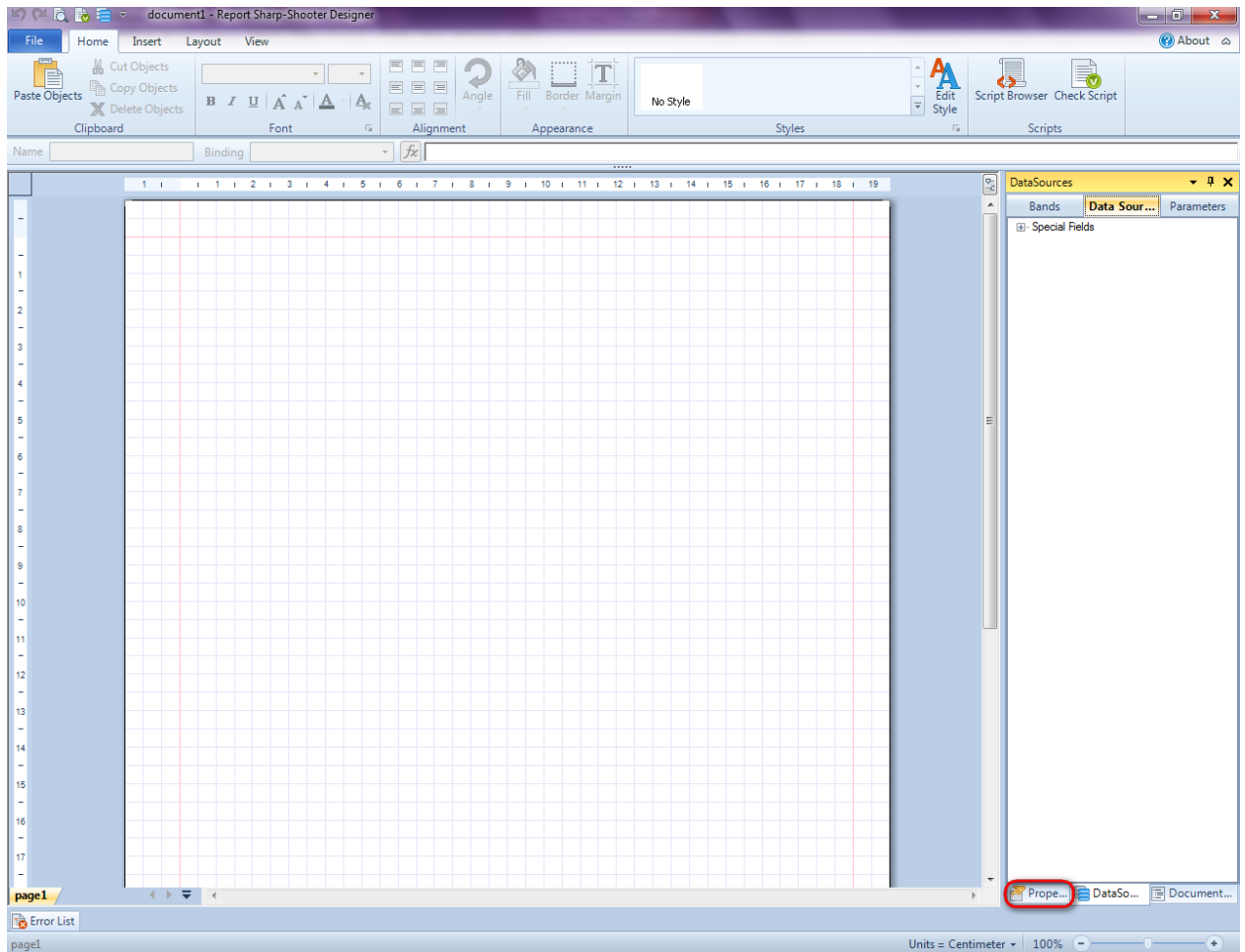


Select "Blank Report" in the Wizards Gallery and click "OK".



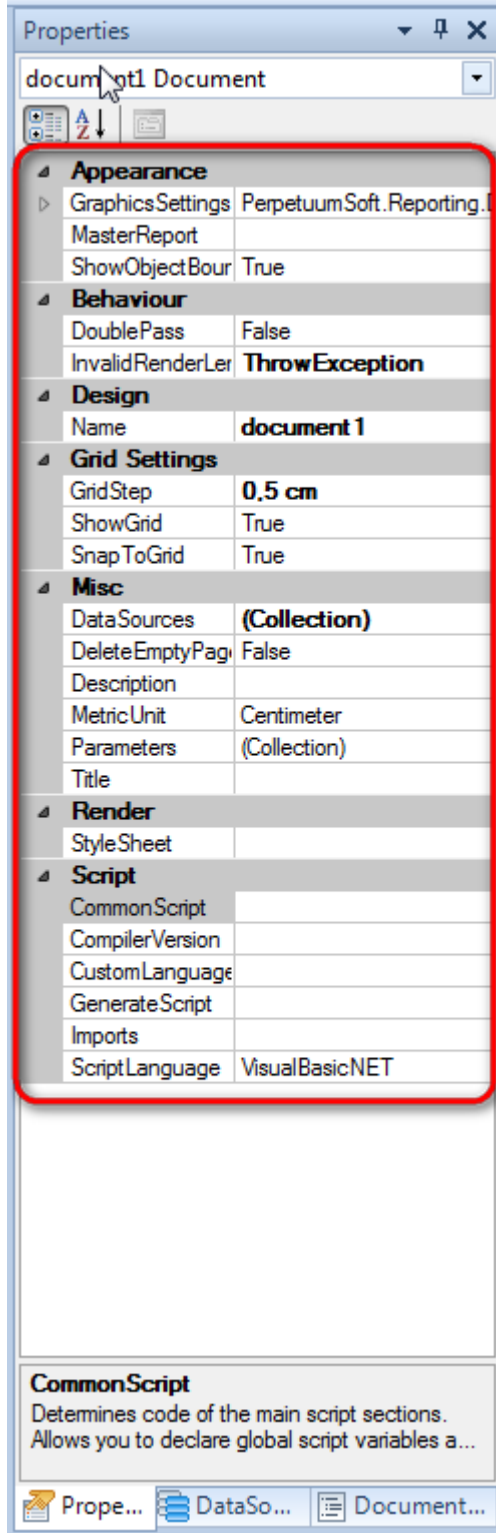
Step 9

Click the "Properties" tab of the tool window in the right part of the designer.

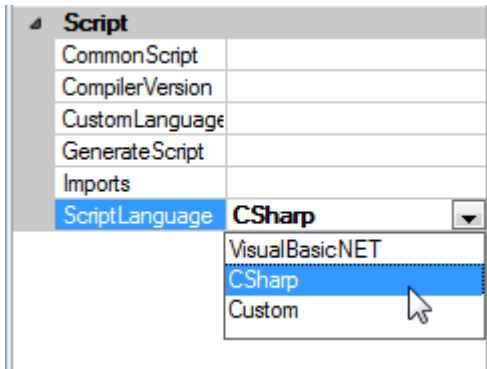




You will see properties of the edited template on the "Properties" tab

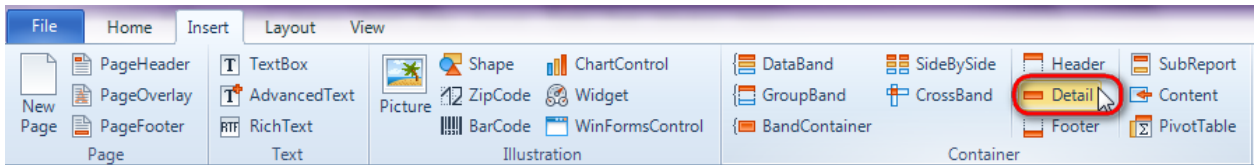


Set property ScriptLanguage = CSharp.



Step 10

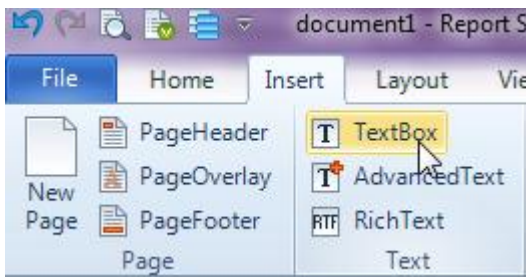
Press "Detail" button on the Insert tab in the group Container.



Click on the template area to add Detail band to the template.

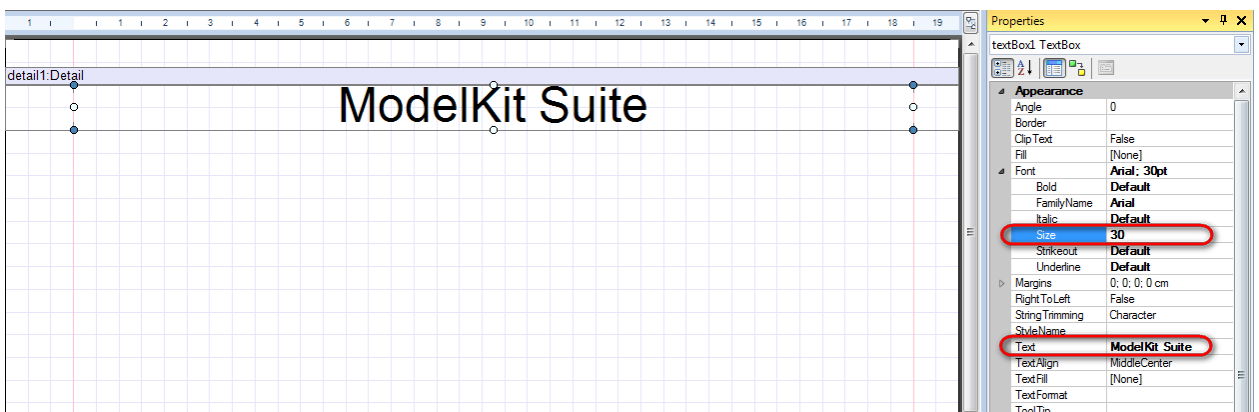
Step 11

Press button "TextBox" on the Insert tab in the group Text.



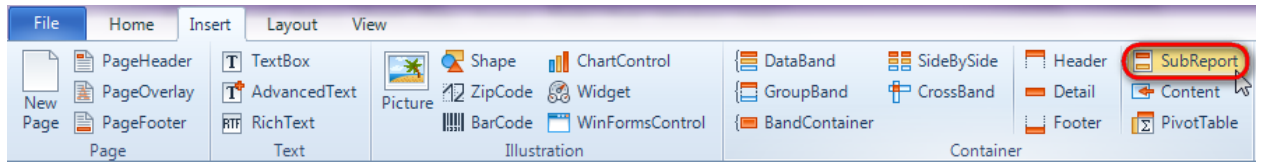
Step 12

Click on the Detail band area to add TextBox element inside it. Set the following properties: Text = "ModelKit Suite", Font.Size = 30.



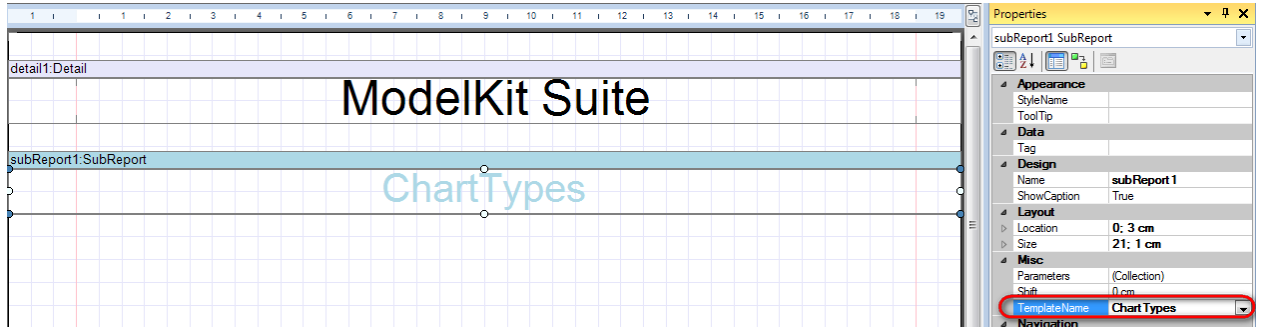
Step 13

Press "SubReport" button on the Insert tab in the group Container.



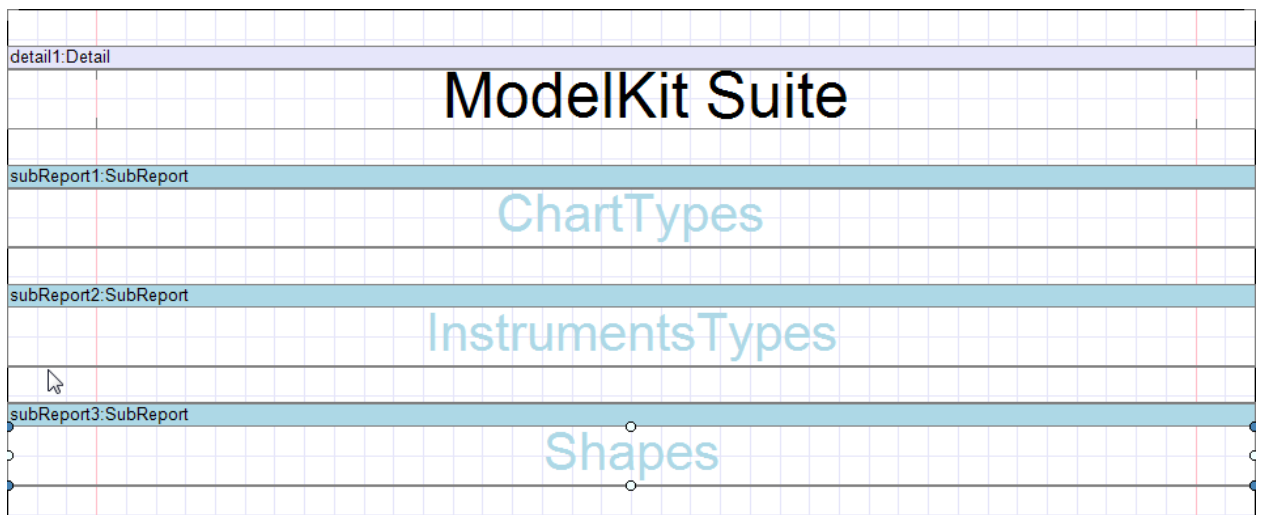
Click on the template area to add SubReport band to the template.

Set property `TemplateName = "ChartTypes"`.



Step 14

In the same way add two more SubReport bands. Set property `TemplateName` to "InstrumentsTypes" and "Shapes".

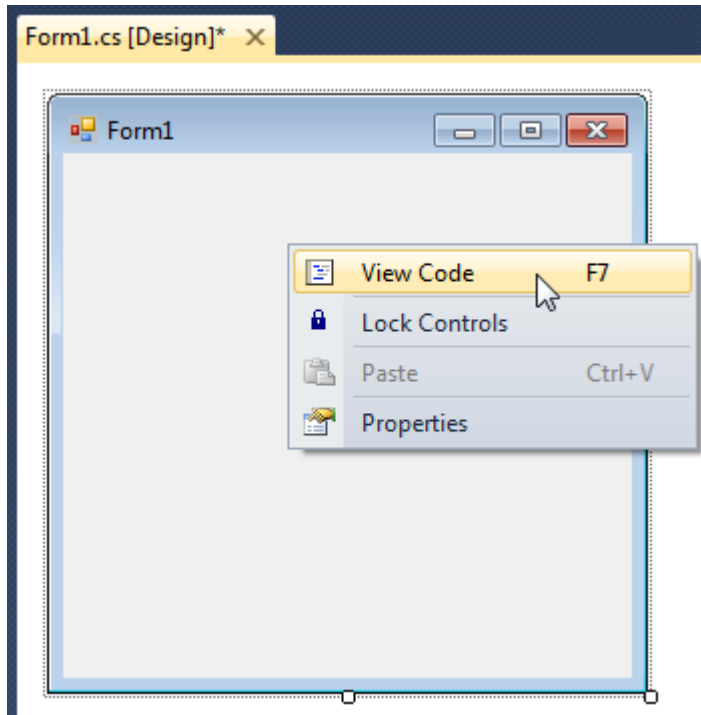


Step 15

Save template, close Report Designer.

Step 16

Right click on the application form and select "View Code" in the context menu to view code.

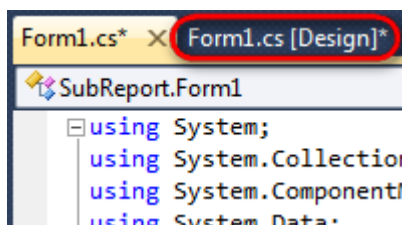


Add code to display report to the class constructor. Create `RenderCompleted` event handler of the `InlineReportSlot` object.

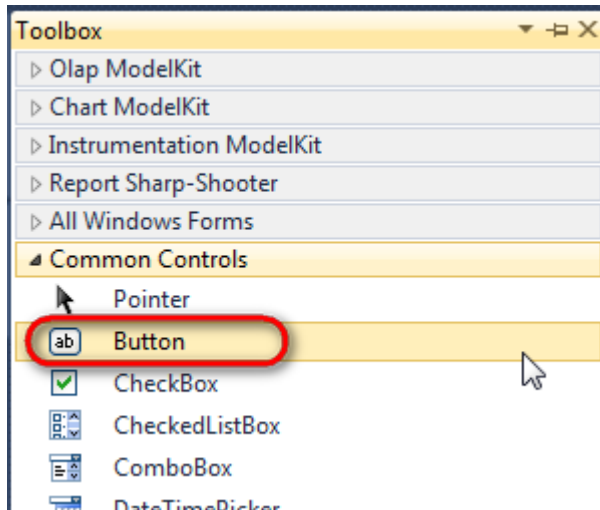
```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

Step 17

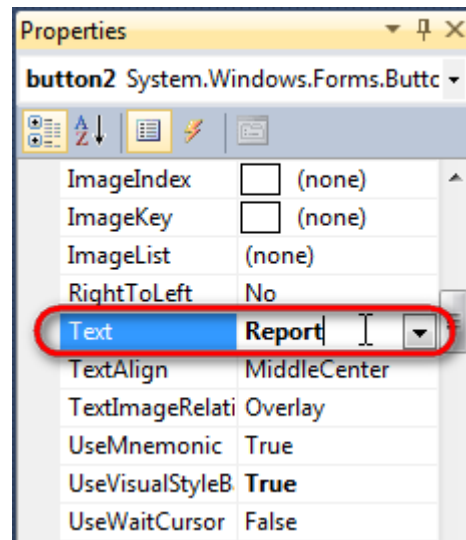
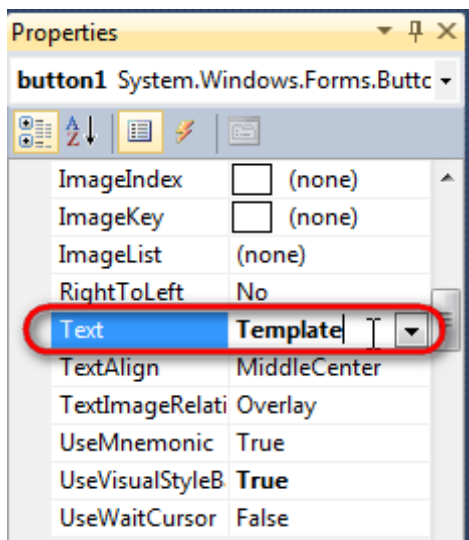
Get back to the application form by clicking "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



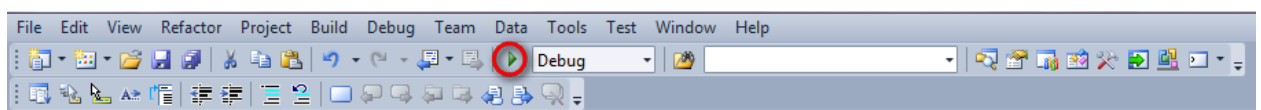
Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

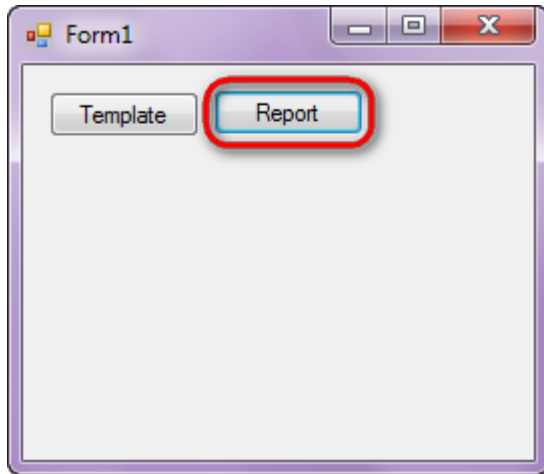
private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 18

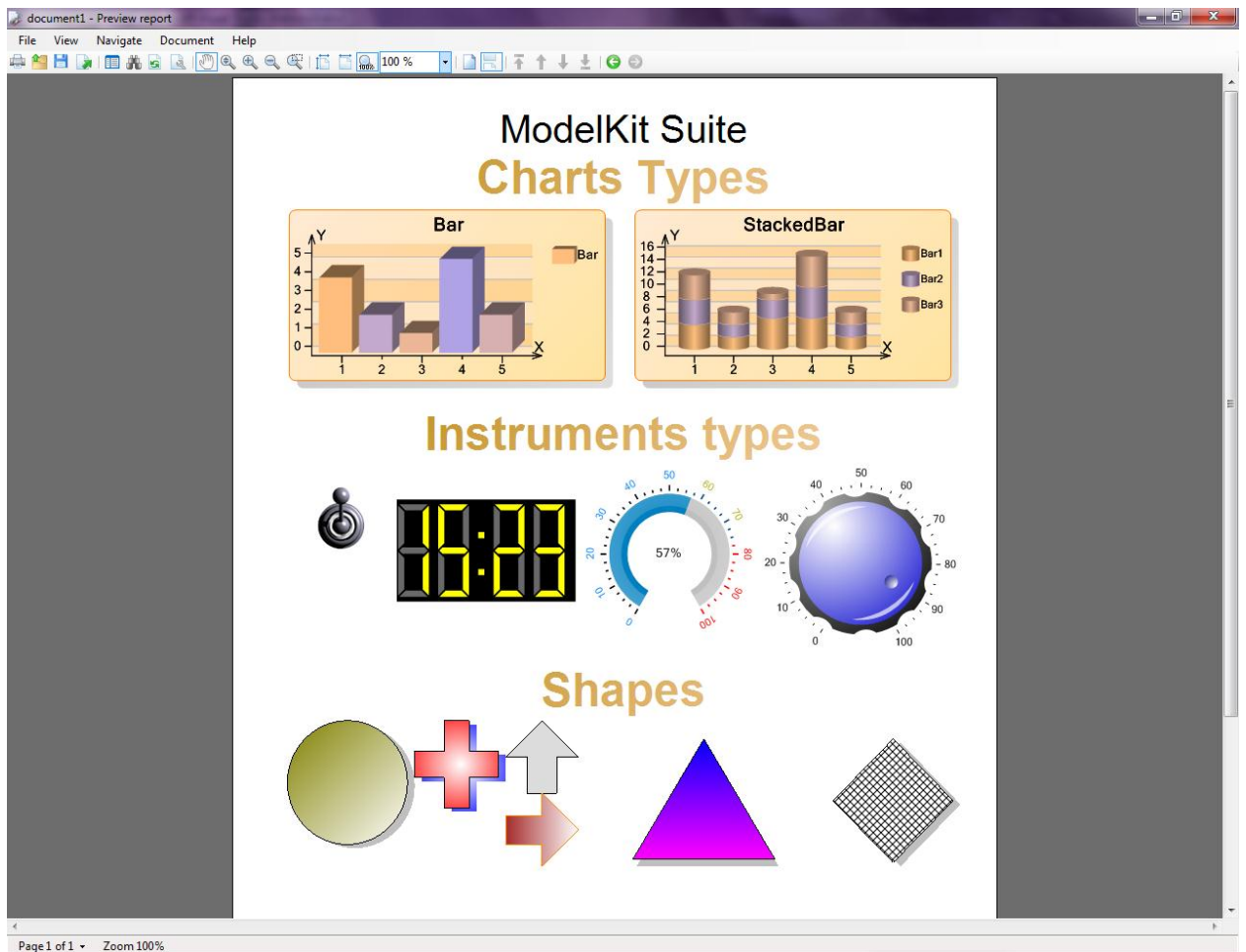
Click "Start Debugging" on the Visual Studio toolbar in order to start application.



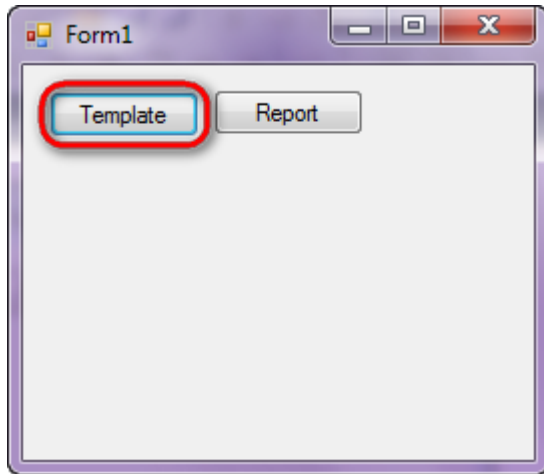
Click the "Report" button in the opened application window.



Generated report is viewed in the Report Viewer.



To edit report template, close Report Viewer and click "Template" on the application form.



Similar sample in the Samples Center is Reports\Sub-Reports\The use of sub-reports.

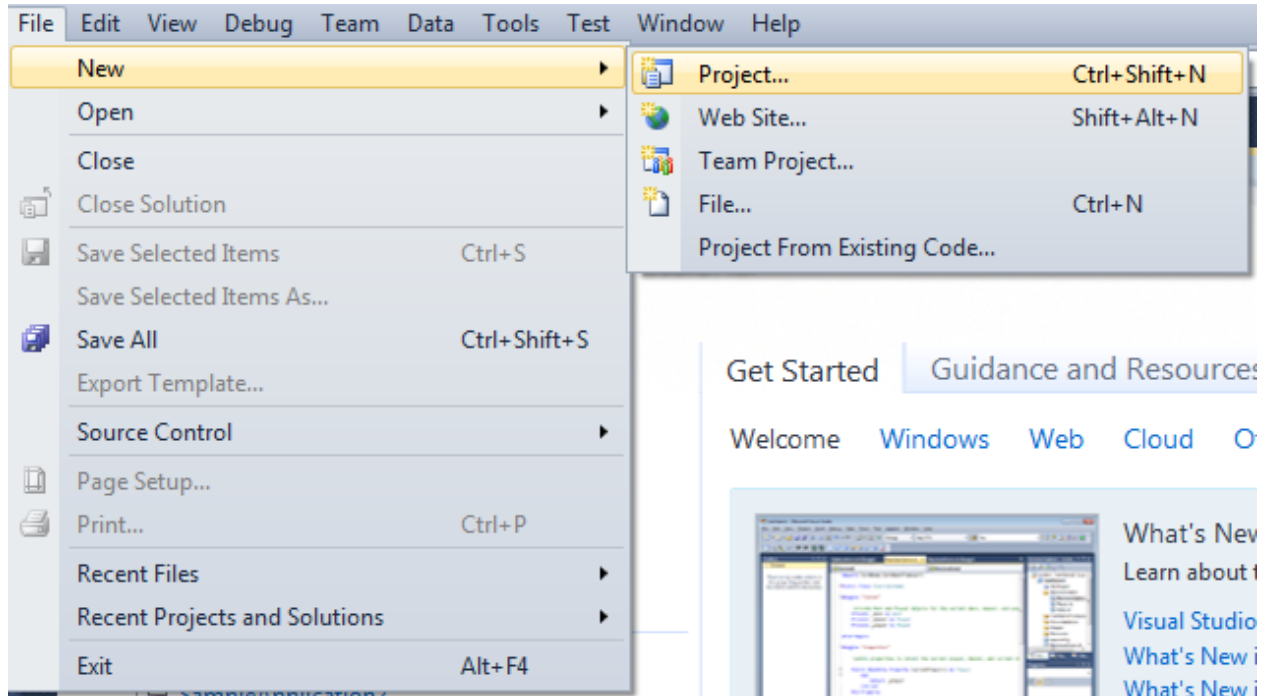


Master Report

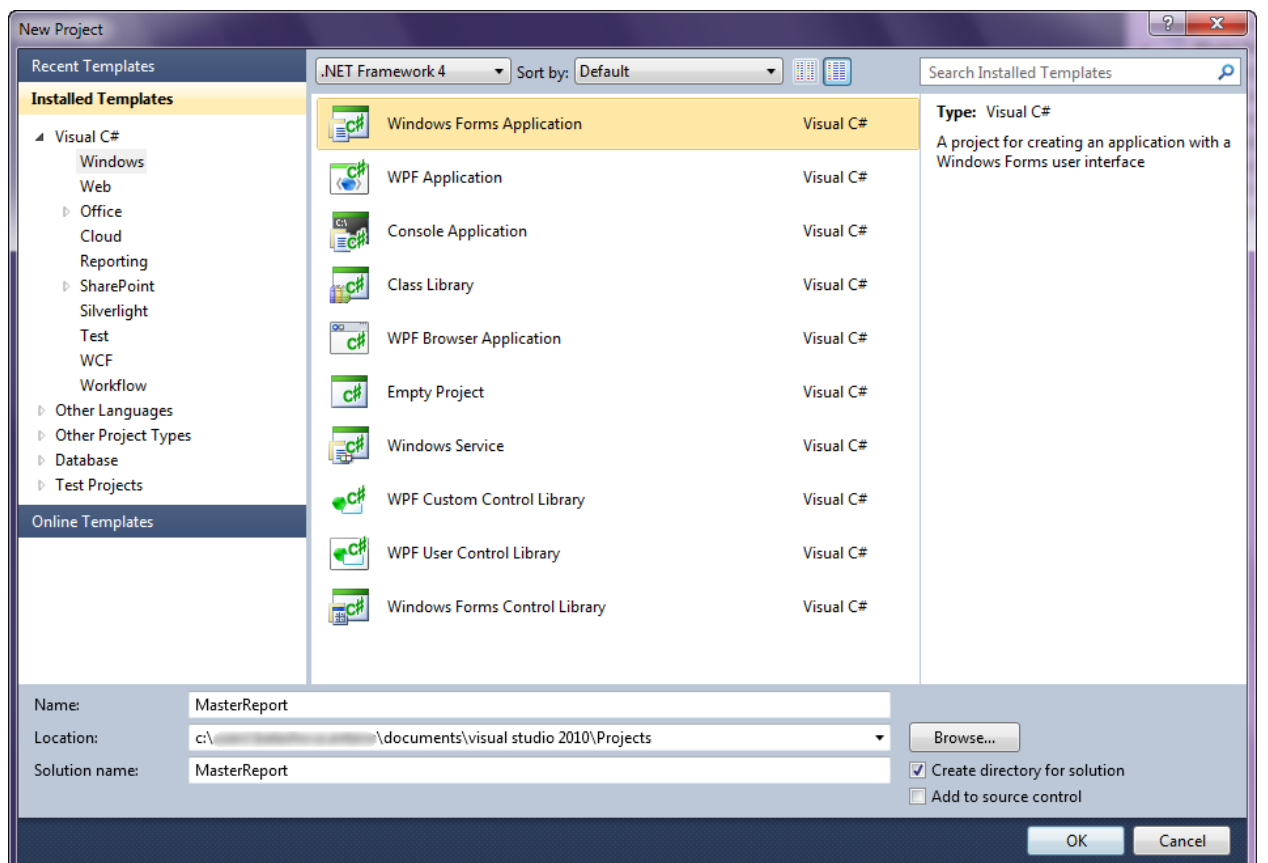
Application provides the ability to select header and contents of the report.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.



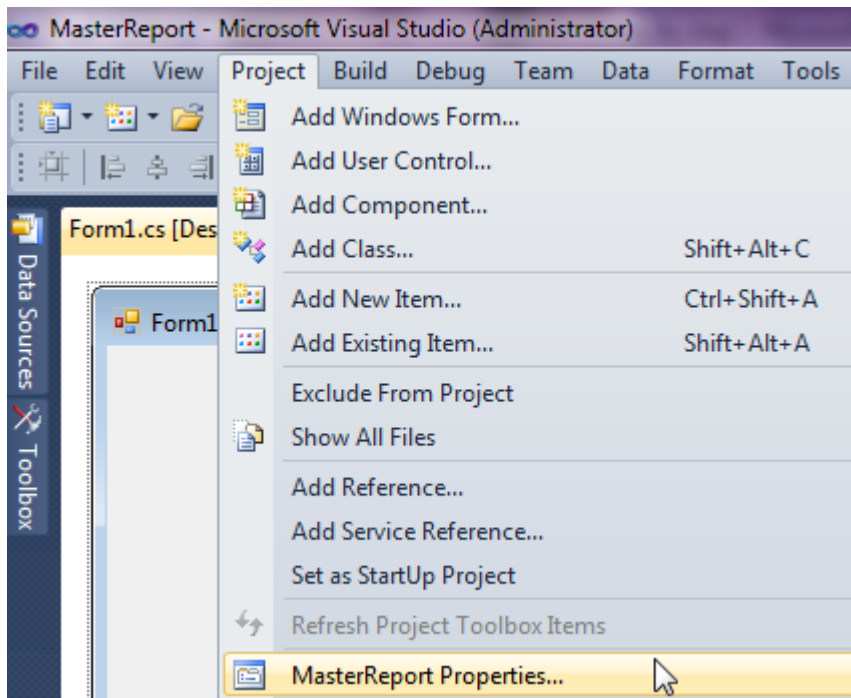
Select Windows Forms Application, set project name – “MasterReport”, set directory to save the project to.



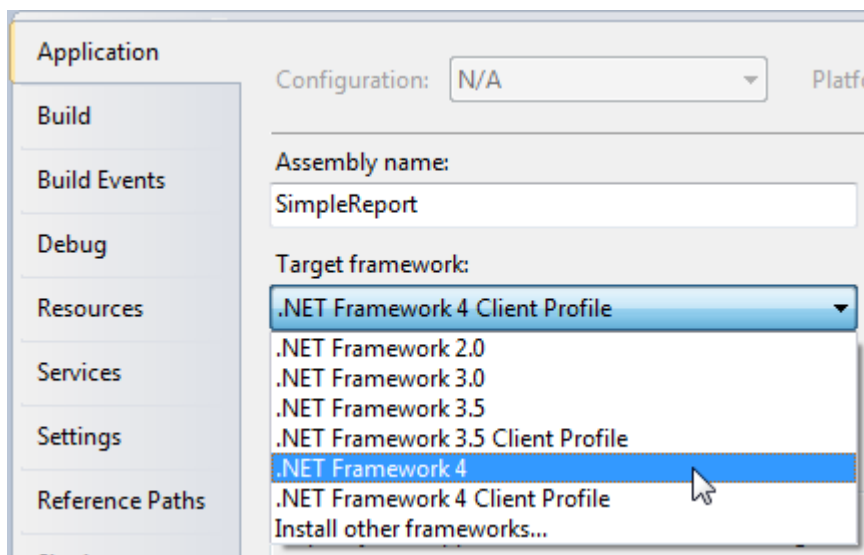


Step 2

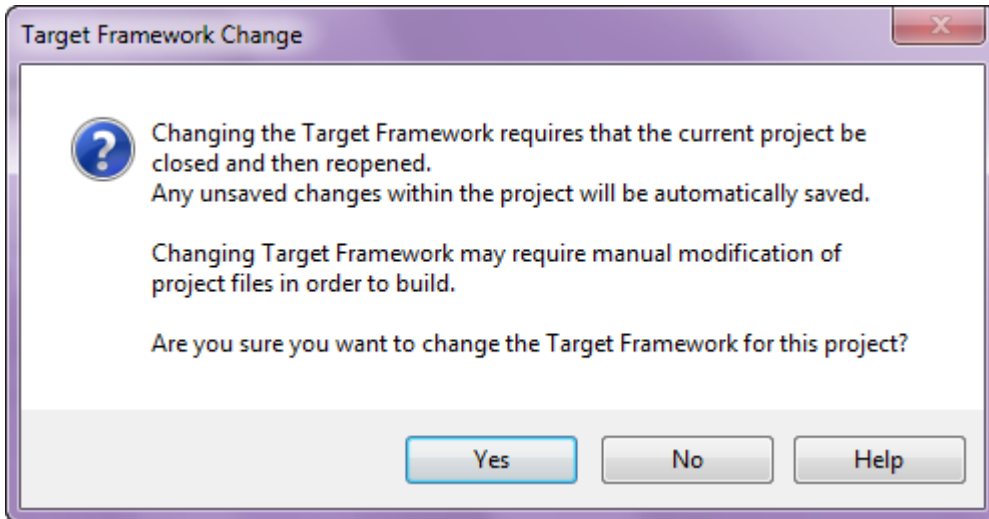
Change the project properties. Select the Project\MasterReport Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

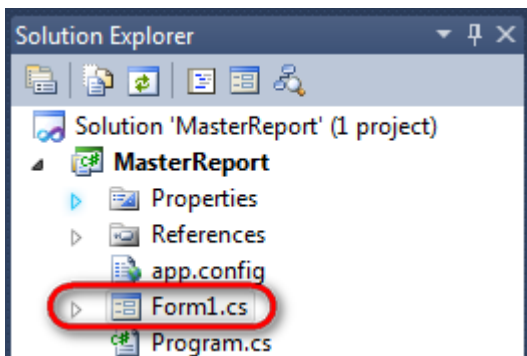


In the opened window press the "Yes" button.

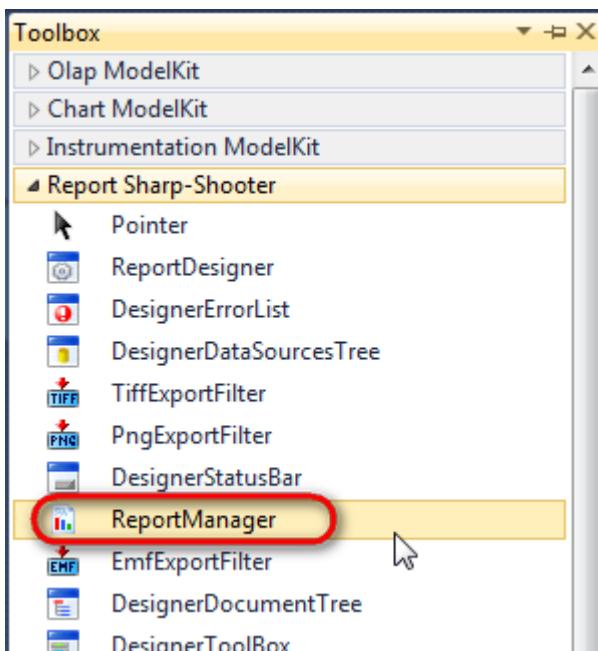


Step 3

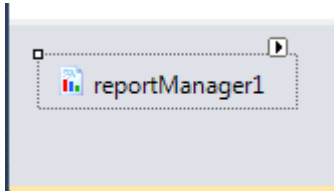
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

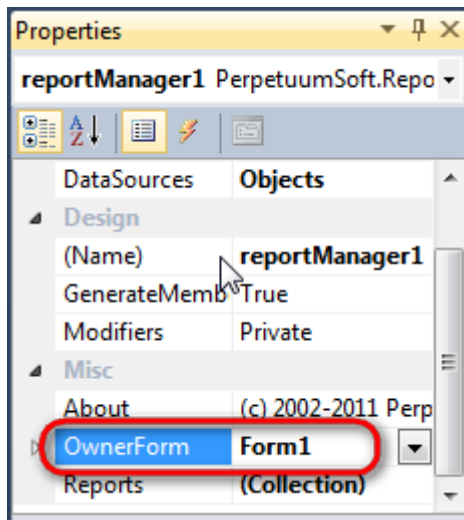


The component is available in the lower part of the window.



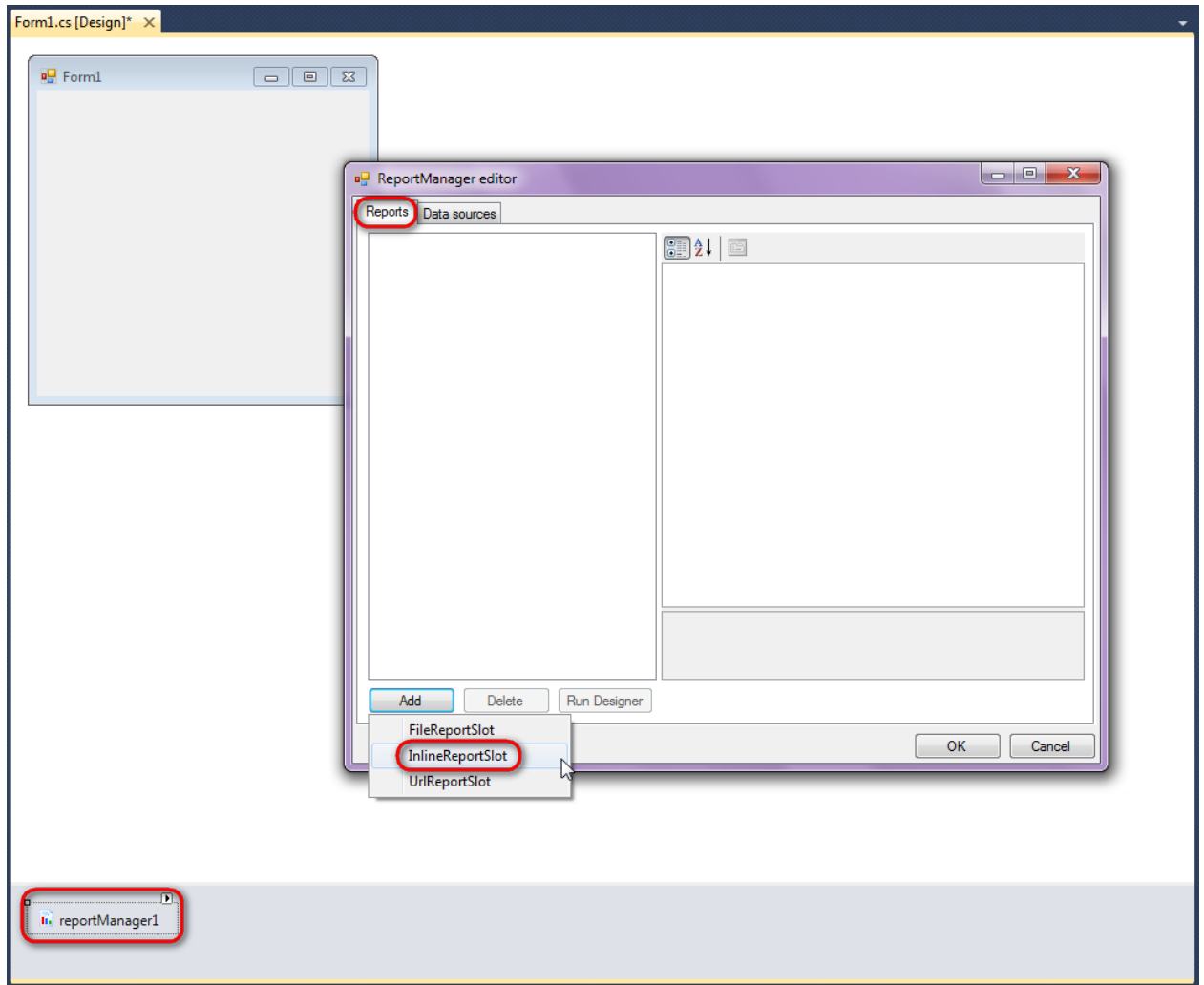
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



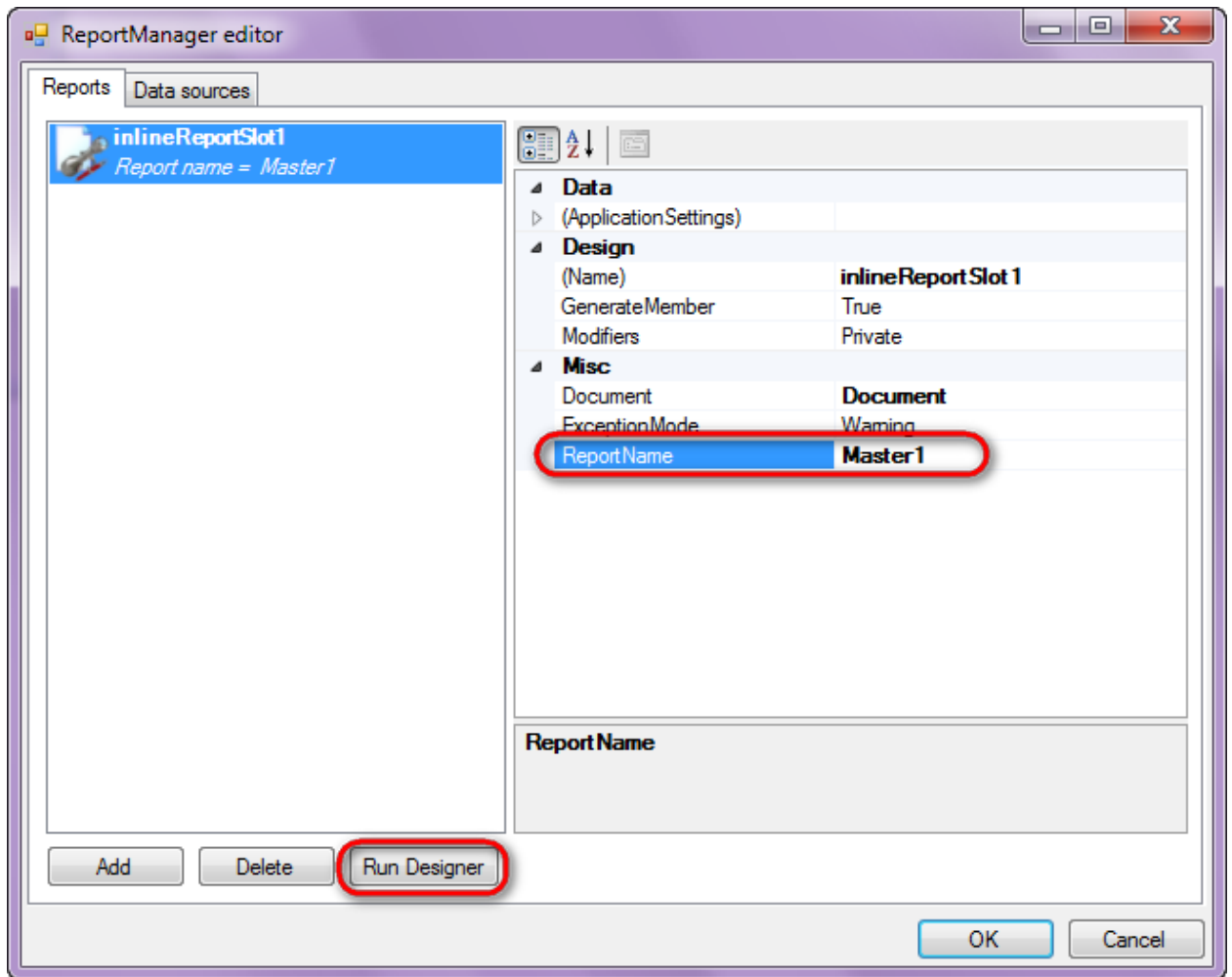
Step 5

Double click on ReportManager to open ReportManager editor.



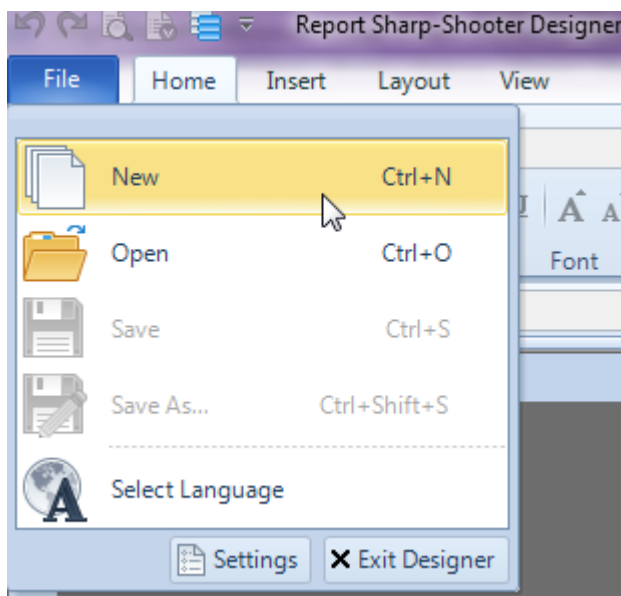
On the "Reports" tab, click "Add" and select "InlineReportSlot".

Set name of the report in property ReportName = Master1. Click "Run Designer" in order to open template editor - ReportDesigner.

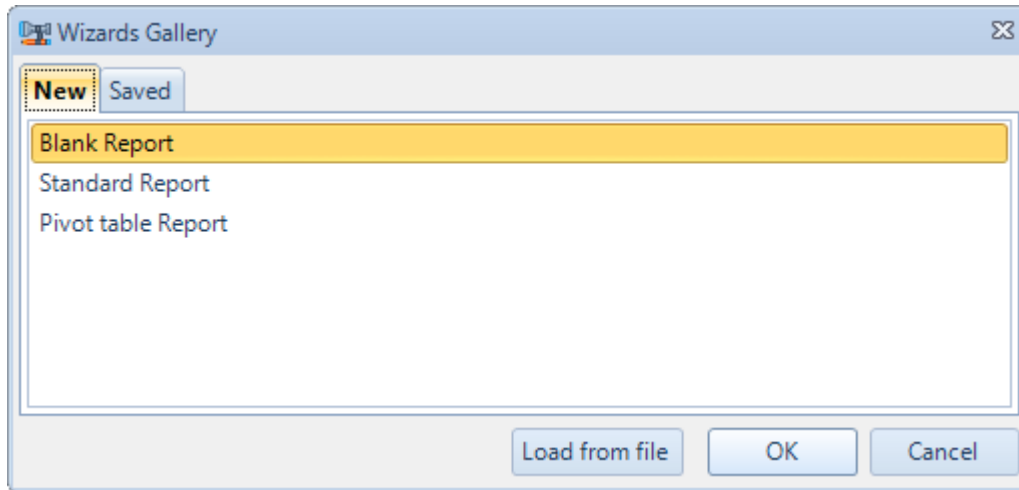


Step 6

Create new empty template – select item File\New from the main menu.

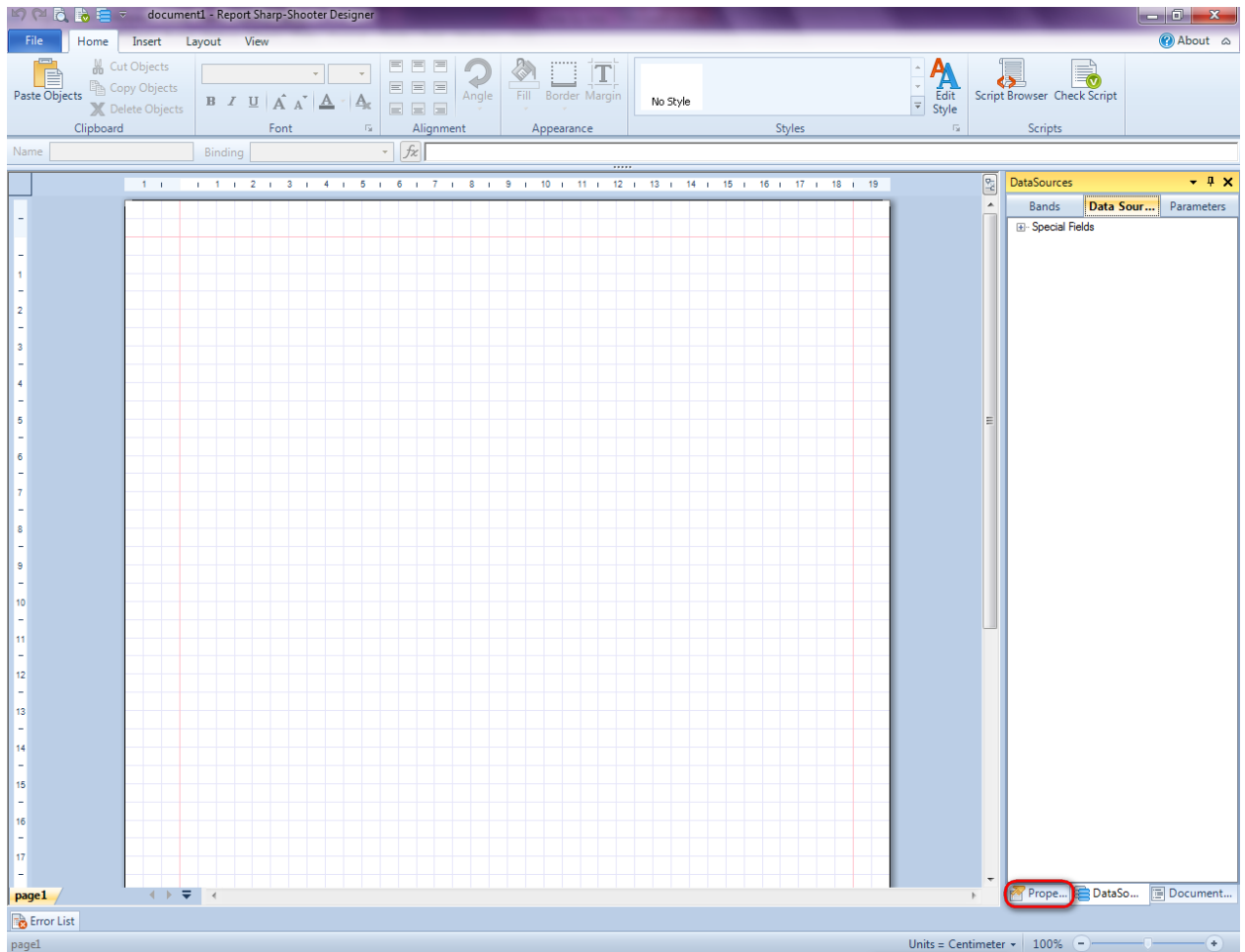


Select "Blank Report" in the Wizards Gallery and click "OK".

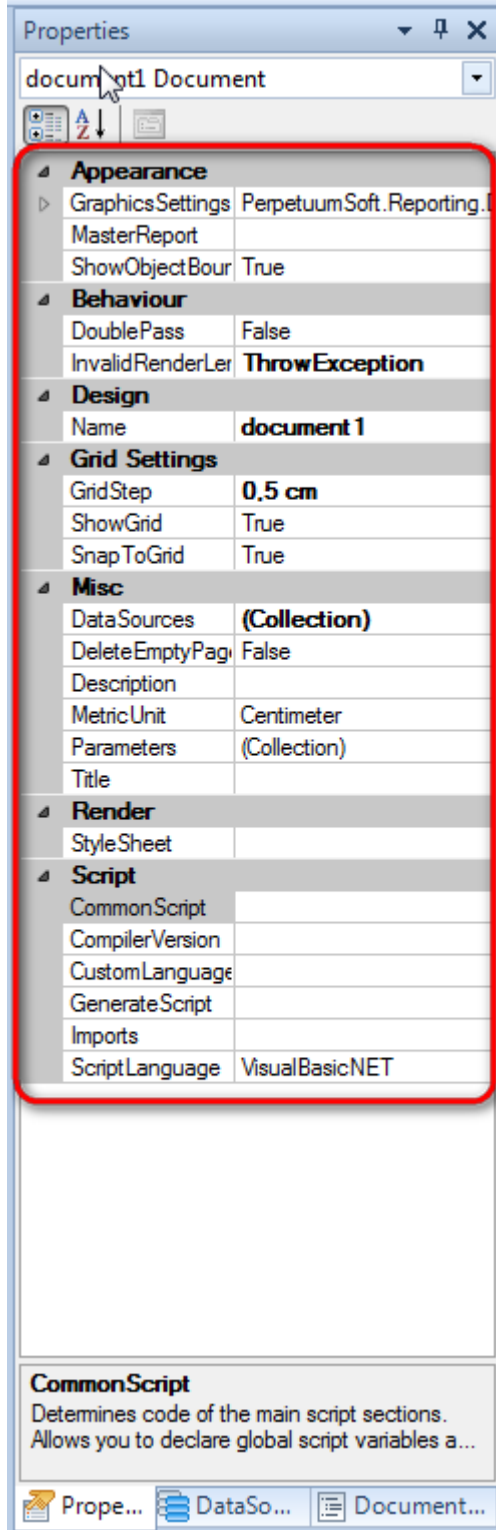


Step 7

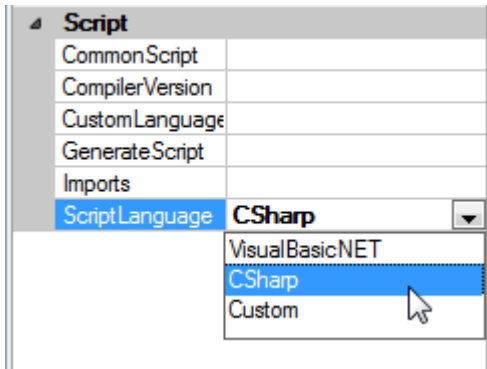
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

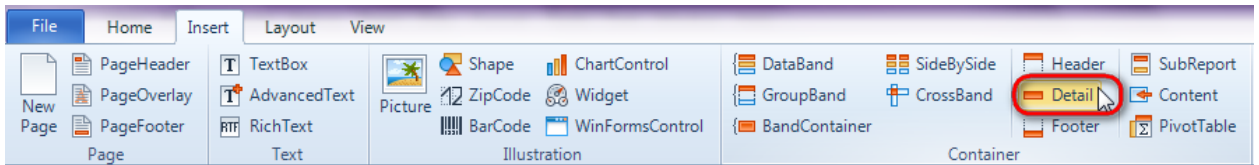


Set property ScriptLanguage = CSharp.



Step 8

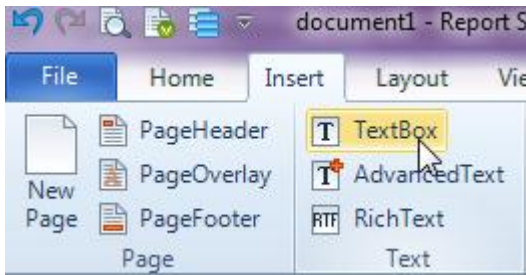
Press "Detail" button on the Insert tab in the group Container.



Click on the template area to add Detail band to the template.

Step 9

Press button "TextBox" on the Insert tab in the group Text.



Click on the Detail band area to add TextBox element inside it. Set property Text = Master Report.

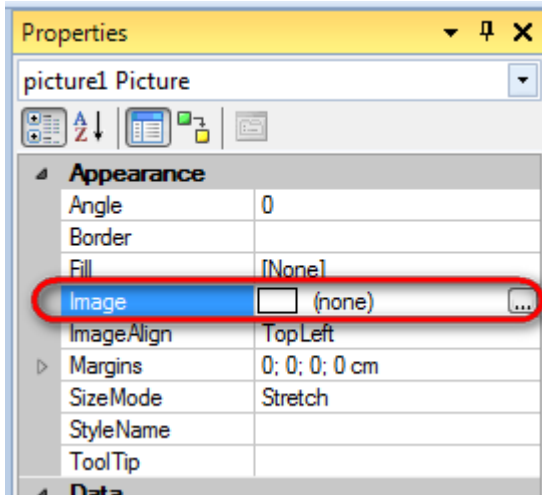
Step 10

Press "Picture" button on the Insert tab in the group Illustration.

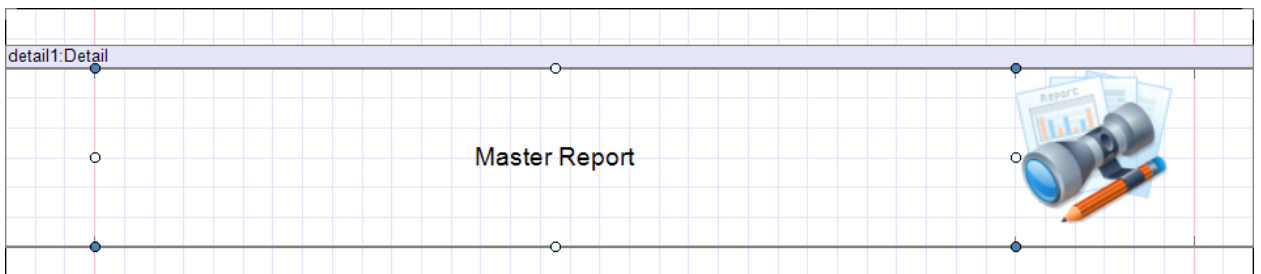


Click on the Detail band area to add Picture element inside Detail.

Select Image property, click button .

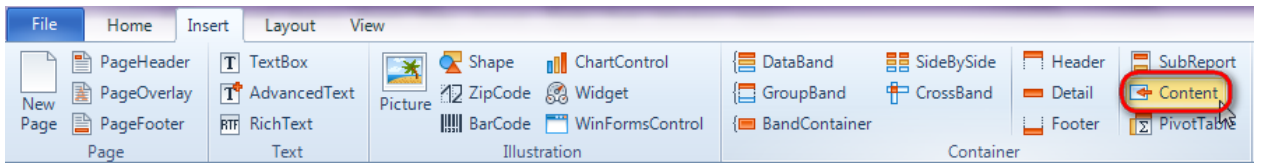


Select file.

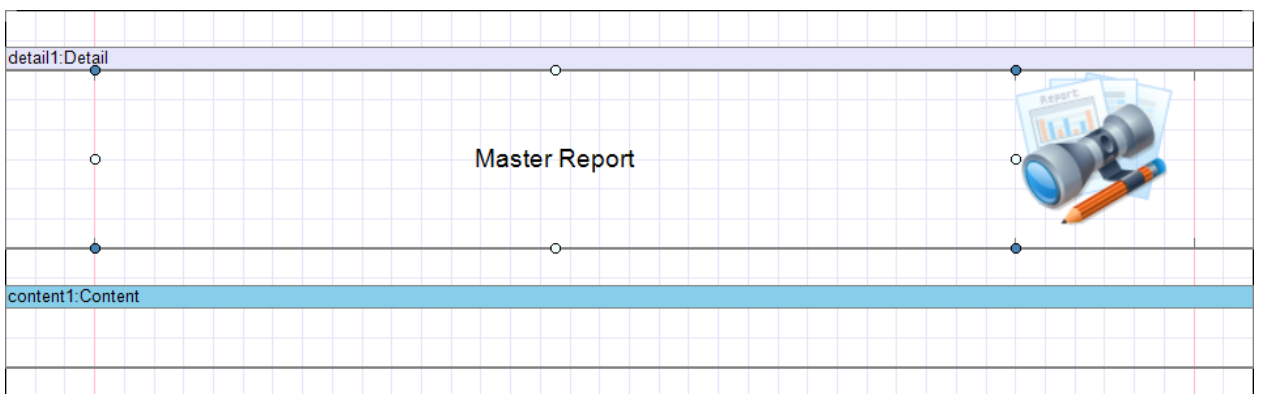


Step 11

Press "Content" button on the Insert tab in the group Container.



Click on the template area to add this band to the template.

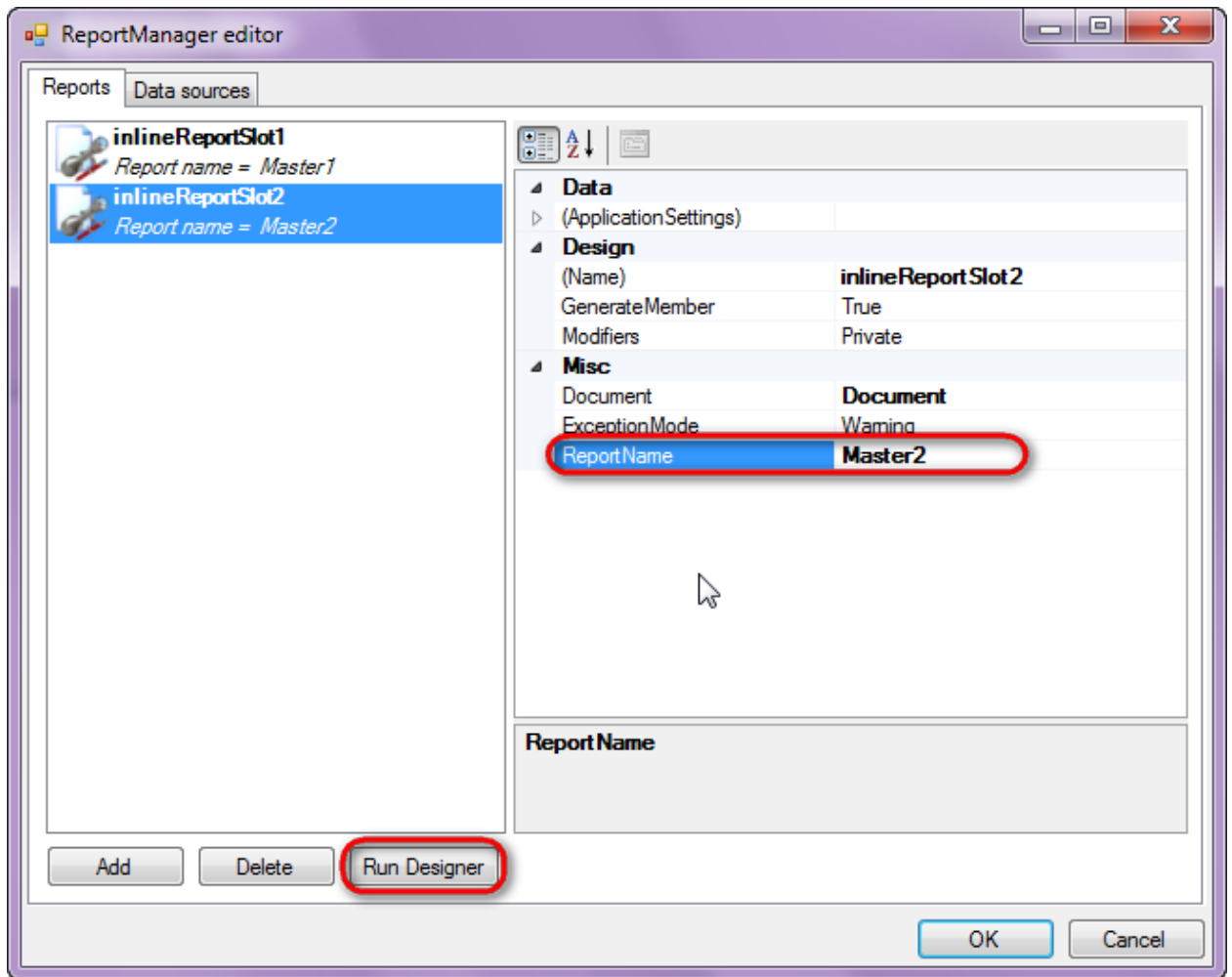


Step 12

Save template, close Report Designer.

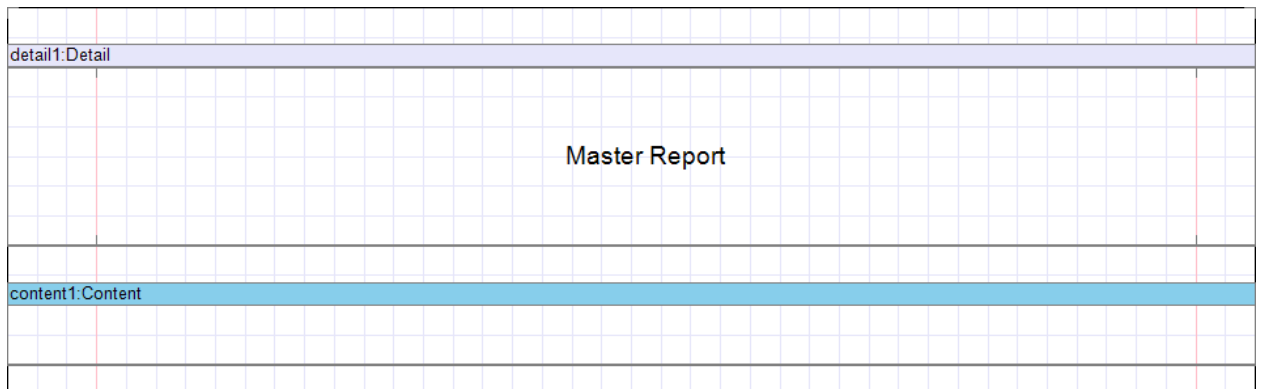
Step 13

Add one more template in the ReportManager editor. Open Report Designer. Set property ReportName = Master2. Open Report Designer.



Step 14

Create a template similar to the first one, but without a picture. Report template should look as follows:



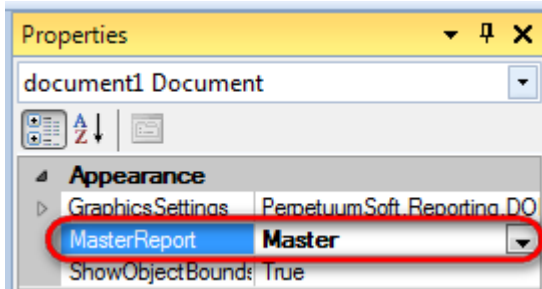
Save template, close Report Designer.

Step 15

Add one more template in the ReportManager editor. Set property ReportName = Shapes.

Step 16

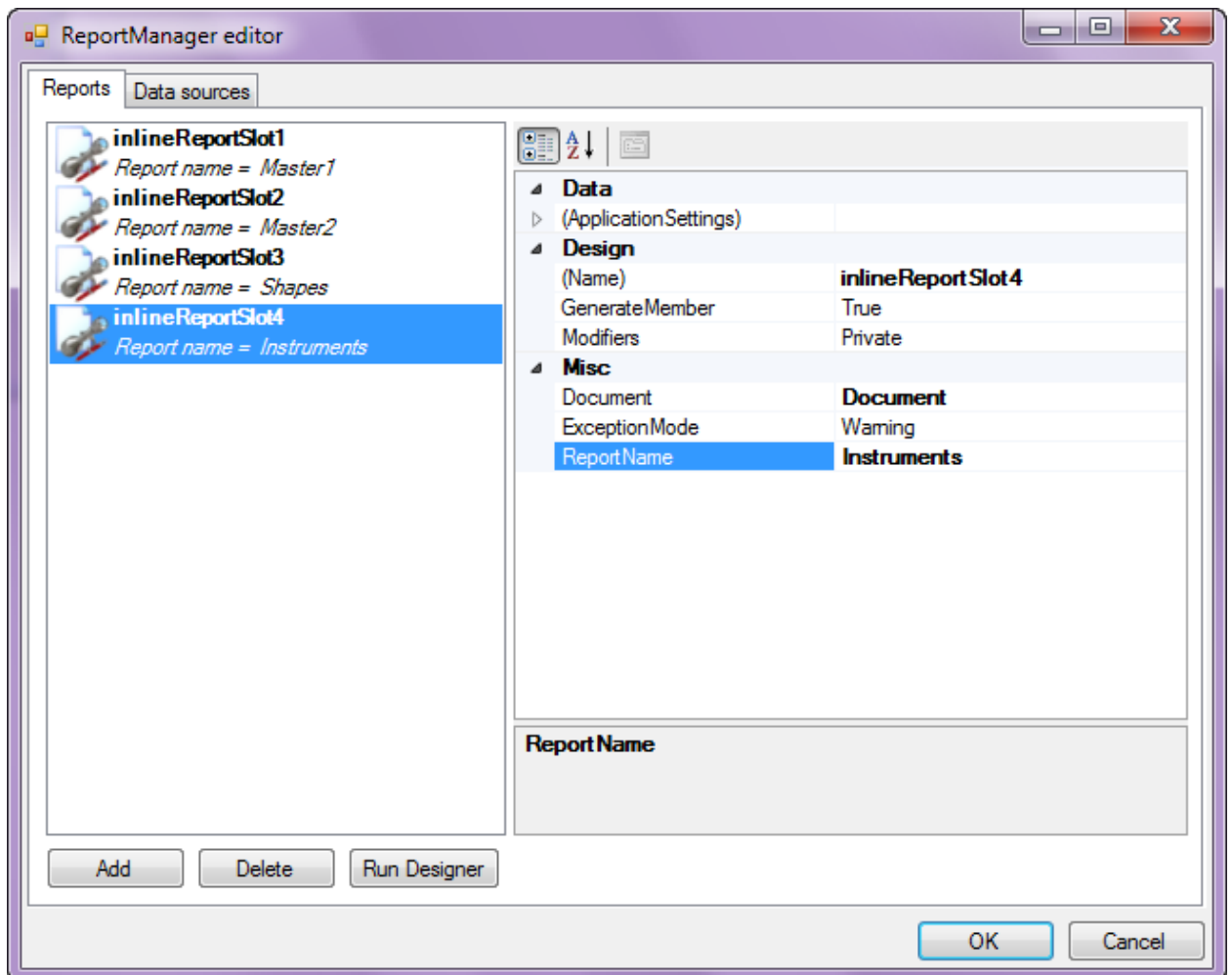
Open Report Designer. Open report properties. Set property MasterReport = Master.



Add Detail bands to the template, add Shape and TextBox elements inside Detail bands. Save template, close Report Designer.

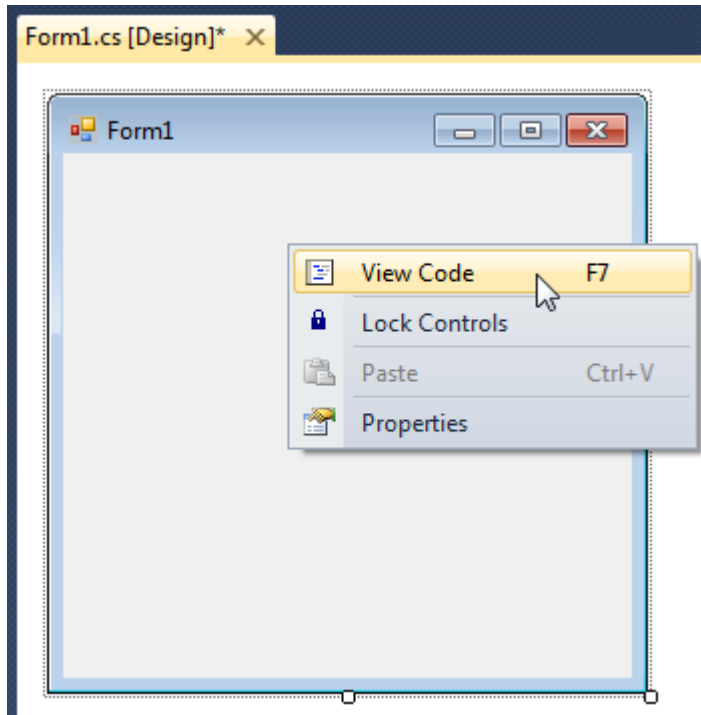
Step 17

Add one more template in the ReportManager editor. Open Report Designer. Set property Name = Instruments. Open report properties. Set property MasterReport = Master. Add Detail bands to the template, add TextBox and Widget elements inside Detail bands. Click on the Widget element and open Instrumentation ModelKit, design widget. Save template, close Report Designer.



Step 18

Right click on the application form and select "View Code" in the context menu to view code.



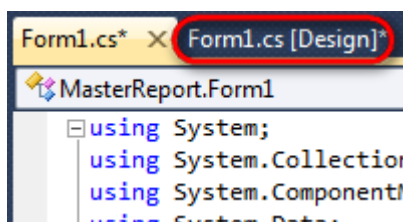
Add code to display report to the class constructor. Create RenderCompleted event handler of the InlineReportSlot object.

```
public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot3.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
    inlineReportSlot4.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}

private void reportSlot_RenderCompleted(object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm ( (PerpetuumSoft.Reporting.Components.
IReportSource) sender))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}
```

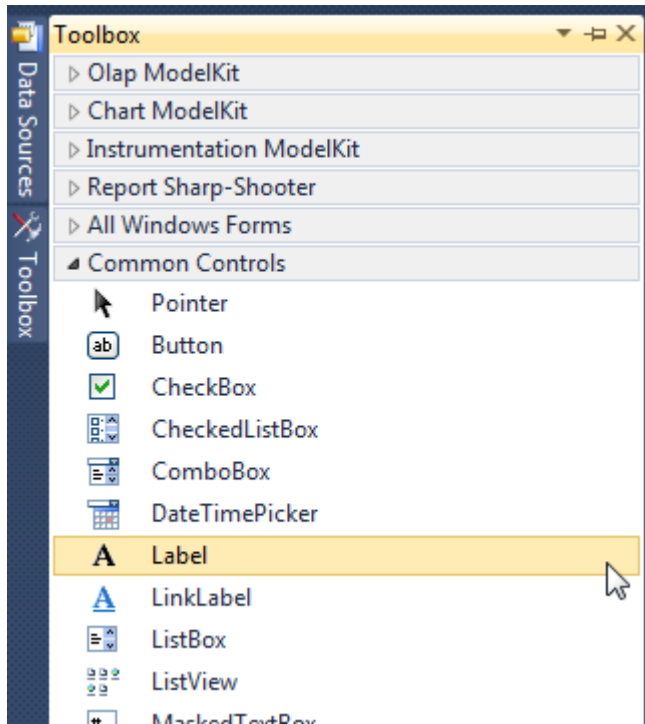
Step 19

Get back to the application form by clicking "Form1.cs[Design]" tab.



Step 20

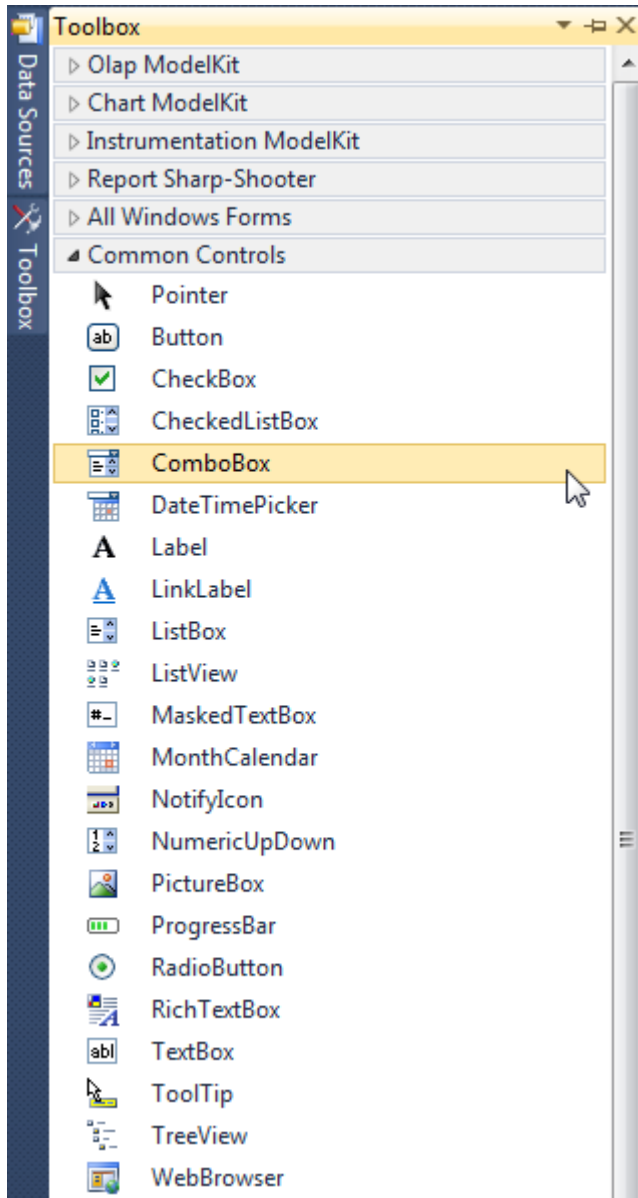
Add two Label elements onto the form (drag and drop "Label" element from the Toolbox onto the form).



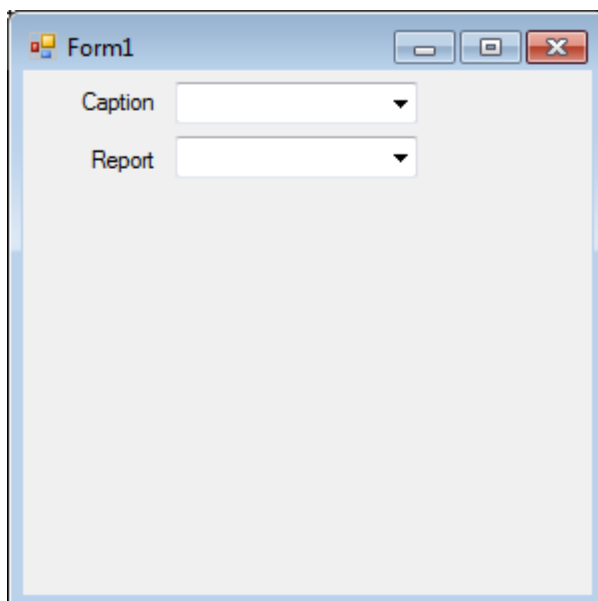
Select Label on the form, edit Text property on the property grid. Set Text = Caption for one label and Text = Report for the second.

Step 21

Add two ComboBox elements onto the form (drag and drop "ComboBox" element from the Toolbox onto the form).



Place elements so that comboBox1 is located next to the "Caption" label and comboBox2 is located next to the "Report" label.



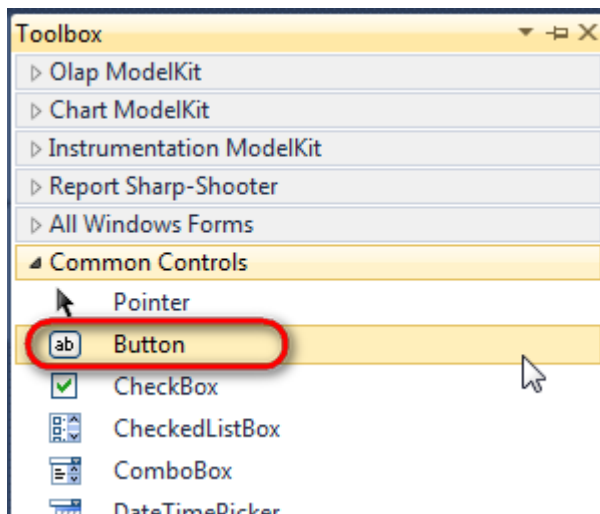


View code and add class constructor to fill lists.

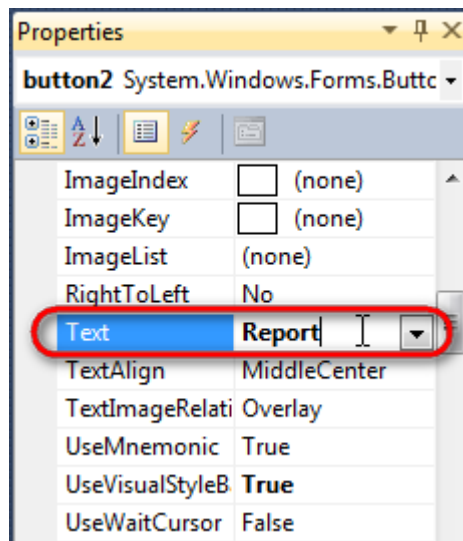
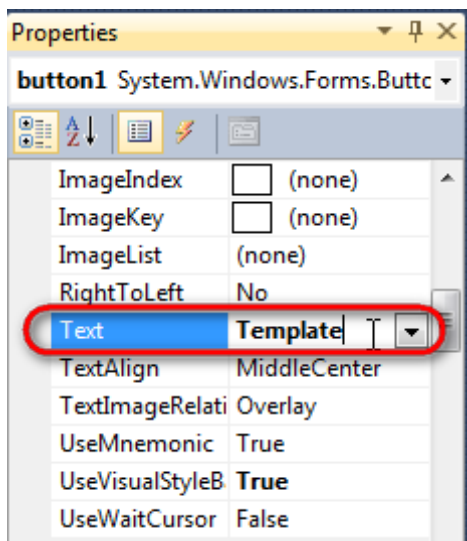
```
public Form1 ()
{
    InitializeComponent ();
    comboBox1.Items.Add ("With logo");
    comboBox1.Items.Add ("Without logo");
    comboBox2.Items.Add ("Shapes");
    comboBox2.Items.Add ("Instruments");
    inlineReportSlot3.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
    inlineReportSlot4.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
```

Step 22

Add three buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select the second Button element on the form, edit its Text property. For the first and second one, set Text = Template; for the third, set Text = Report.



Create Click event handlers – double click on the Button element on the form. Add code launching report generation. Depending on the selected values in ComboBox elements, Template buttons will open different templates in the Report Designer. Final report will be generated depending on the selected values.

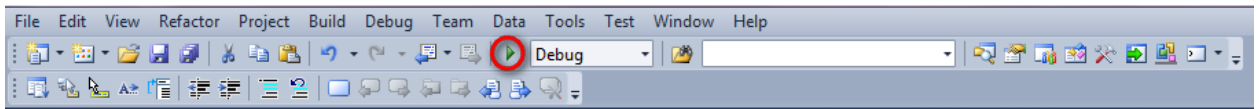
```
private void button1_Click(object sender, EventArgs e)
{
    if (comboBox1.SelectedIndex == 0)
inlineReportSlot1.DesignTemplate();
    if (comboBox1.SelectedIndex == 1)
inlineReportSlot2.DesignTemplate();
}

private void button2_Click(object sender, EventArgs e)
{
    if (comboBox2.SelectedIndex == 0)
inlineReportSlot3.DesignTemplate();
    if (comboBox2.SelectedIndex == 1)
inlineReportSlot4.DesignTemplate();
}

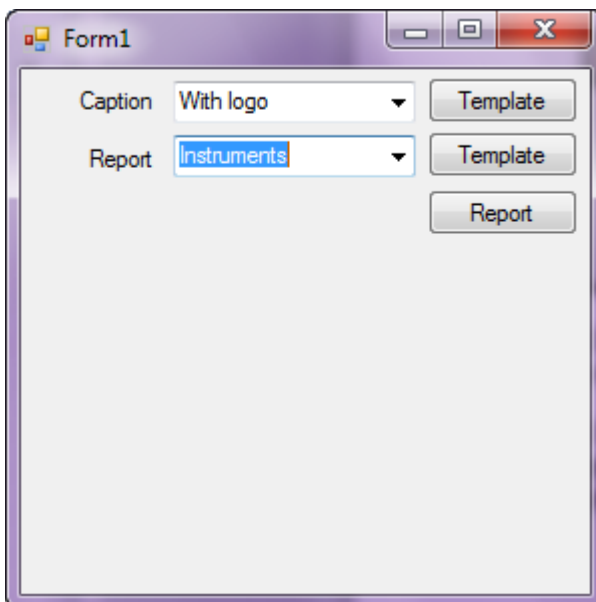
private void button3_Click(object sender, EventArgs e)
{
    if (comboBox1.SelectedIndex == 0)
    {
        inlineReportSlot1.ReportName = "Master";
        inlineReportSlot2.ReportName = "Master2";
    }
    else
    {
        inlineReportSlot1.ReportName = "Master1";
        inlineReportSlot2.ReportName = "Master";
    }
    if (comboBox2.SelectedIndex == 0) inlineReportSlot3.Prepare();
    if (comboBox2.SelectedIndex == 1) inlineReportSlot4.Prepare();
}
```

Step 23

Click "Start Debugging" on the Visual Studio toolbar in order to start application.



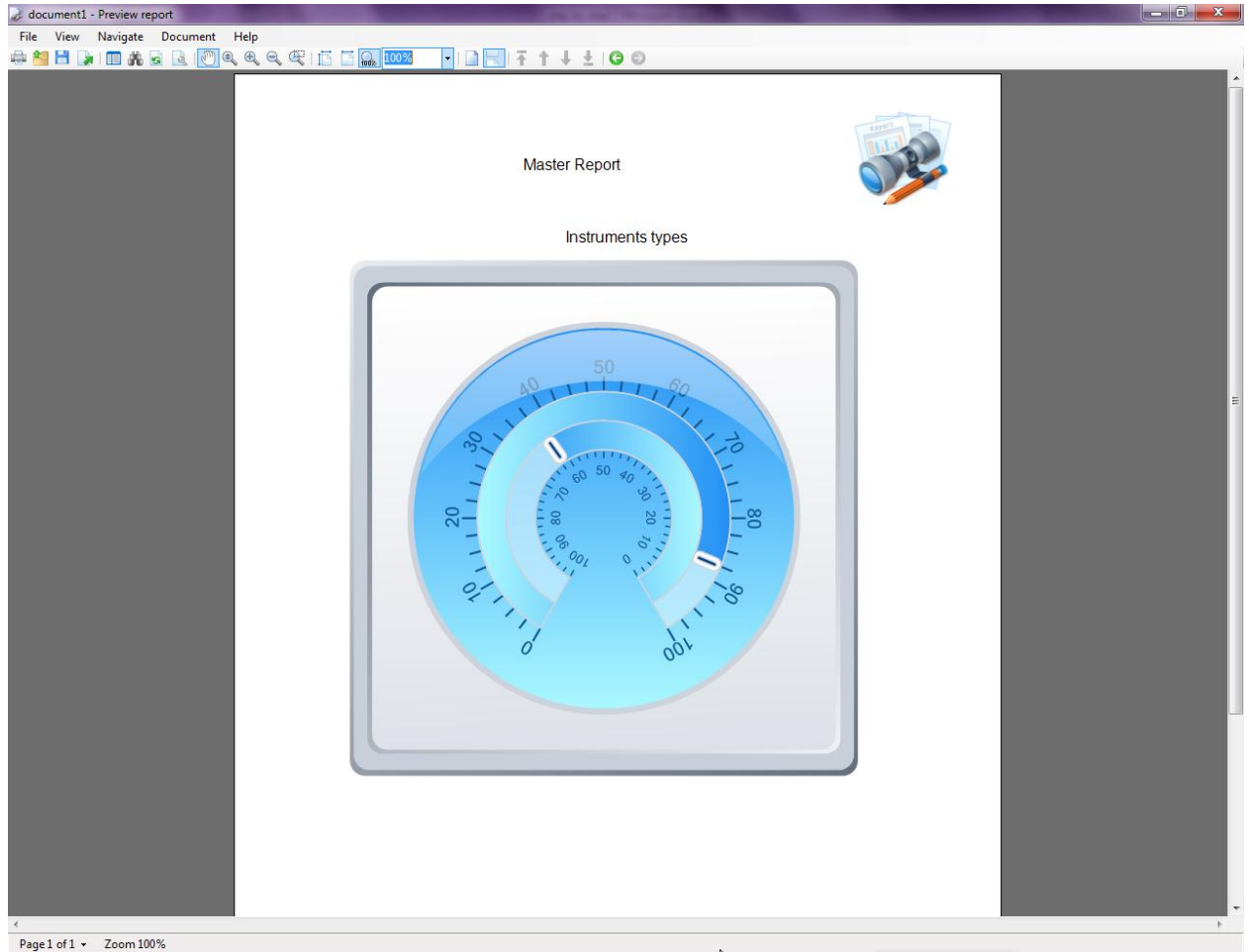
Application form runs.





Click "Template" button and you will open Report Designer with a template selected in the corresponding ComboBox.

Click "Report" button and the final report will be generated; and document header will be generated by the template selected in the Caption section, and the document itself will be generated by the template selected in the Report section.



Similar sample in the Samples Center is Reports\Master Report.

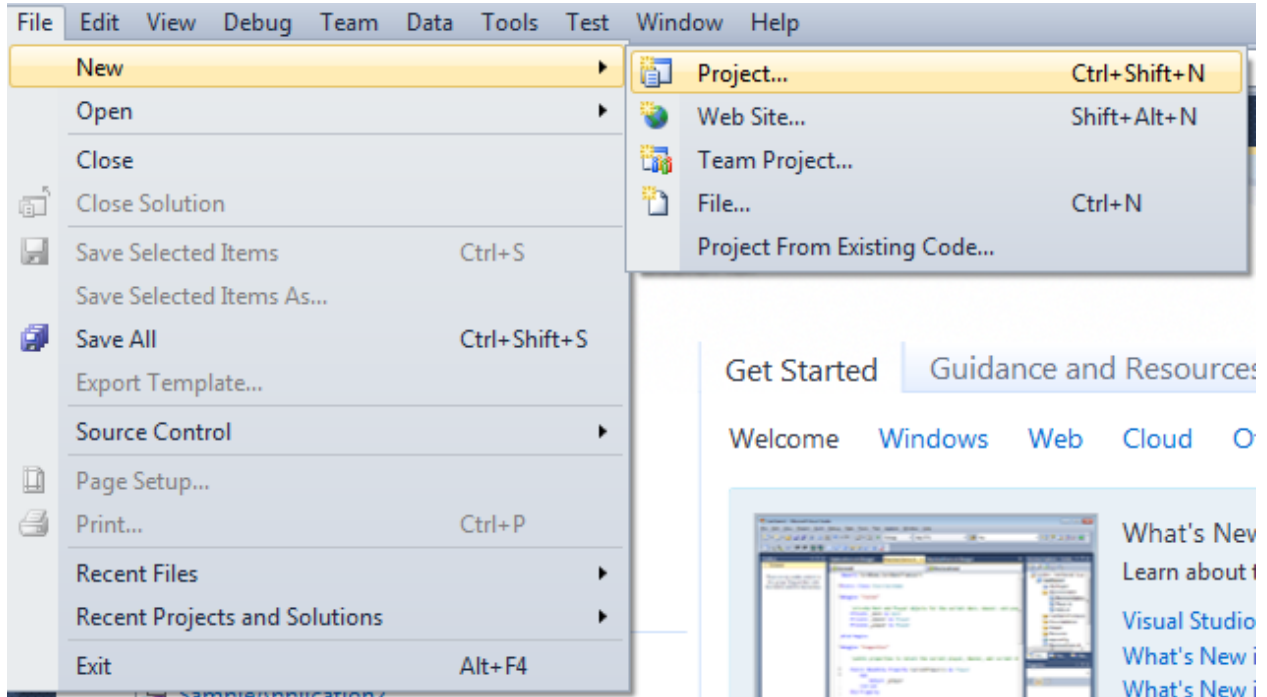


Side by Side Report

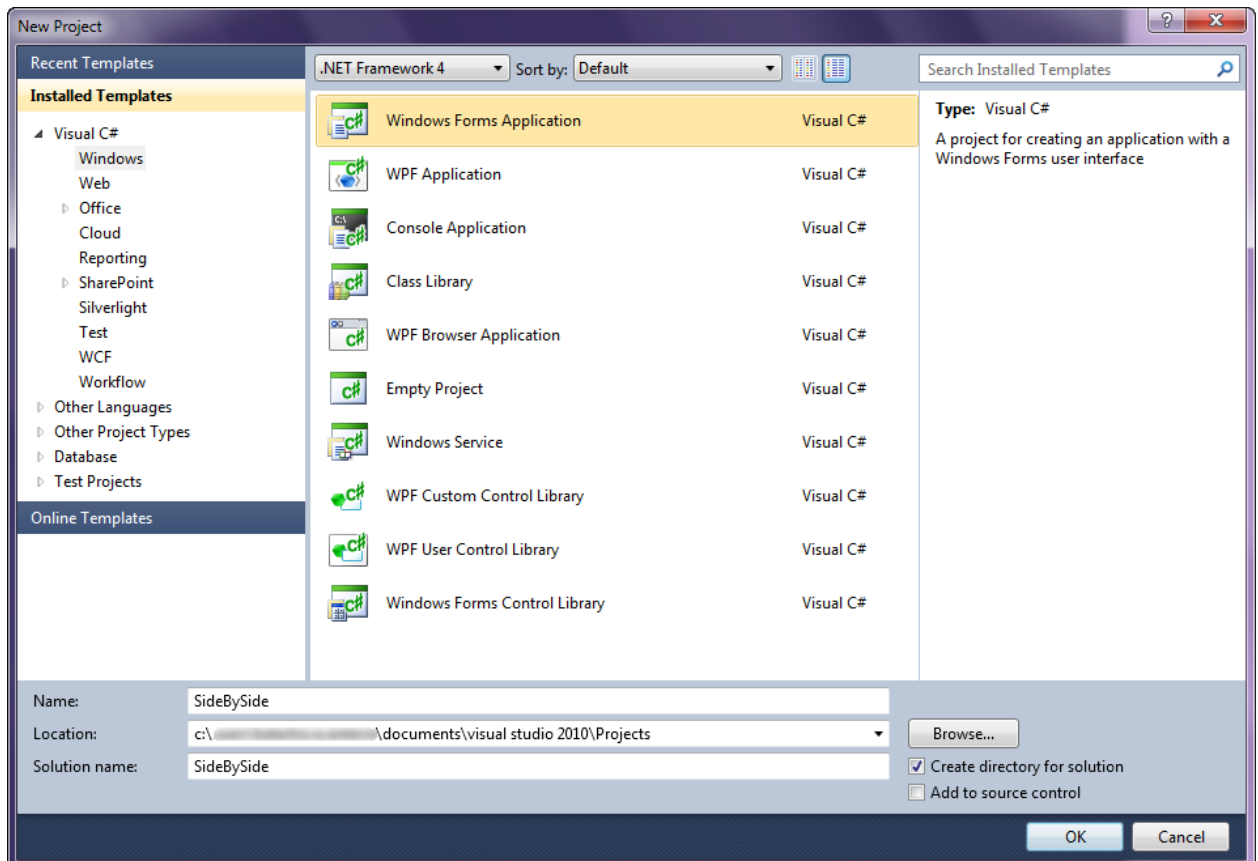
Template of a report containing a table – first column contains numbers from 1 to 20, and the second one contains numbers from 1 to 10.

Step 1

Create new project in Microsoft Visual Studio. Select New\Project from the main menu.

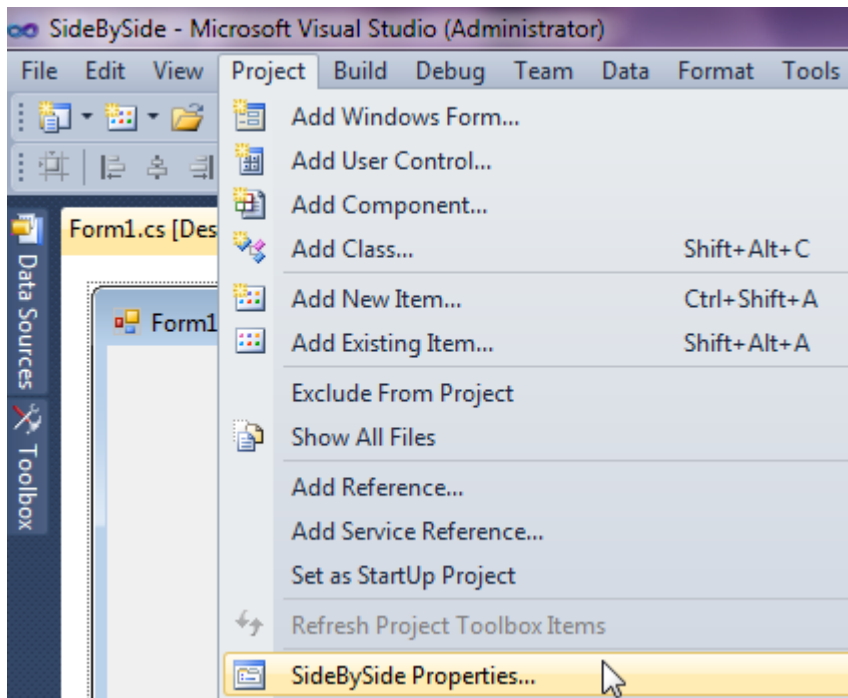


Select Windows Forms Application, set project name – “SideBySide”, set directory to save the project to.

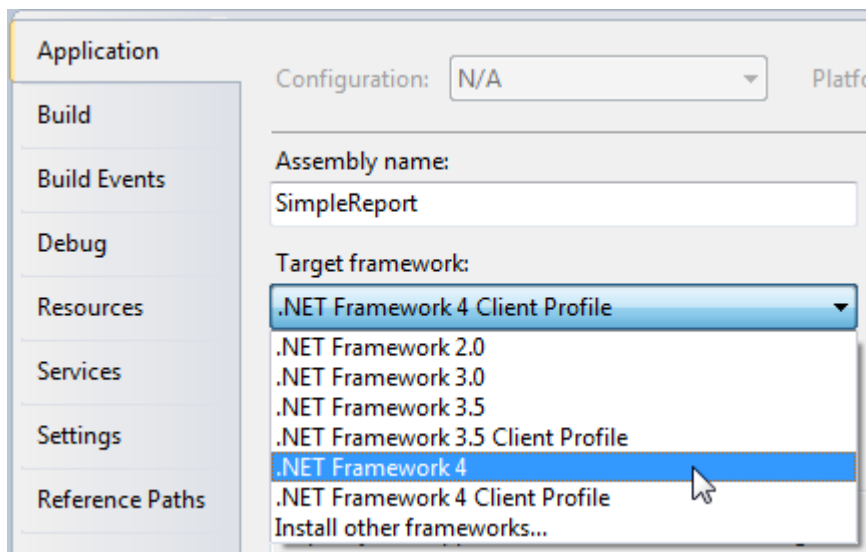


Step 2

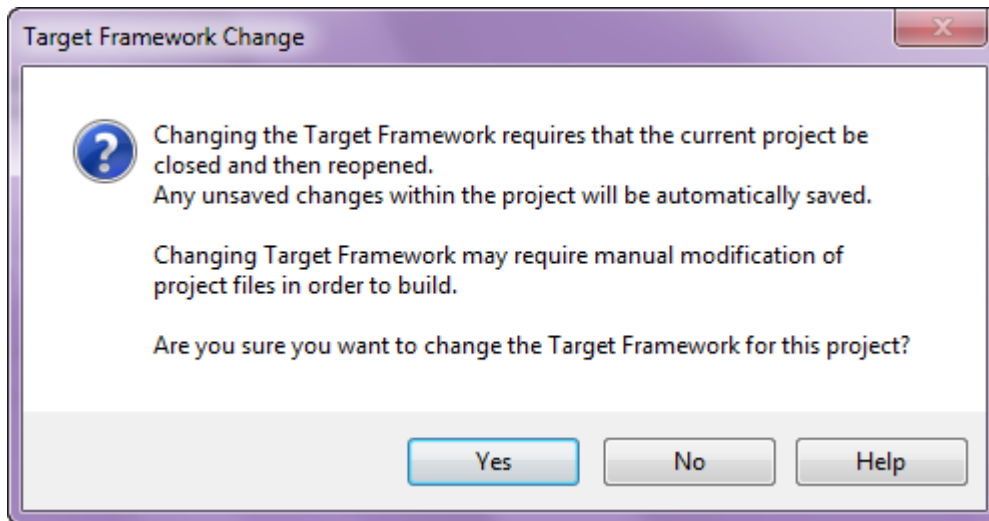
Change the project properties. Select the Project\SideBySide Properties... item in the main menu.



Select item Target framework\ .NET Framework4 from the tab Application.

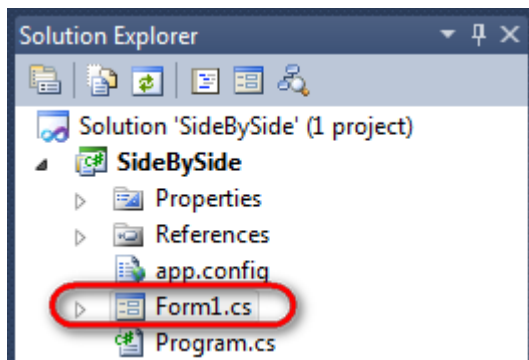


In the opened window press the "Yes" button.

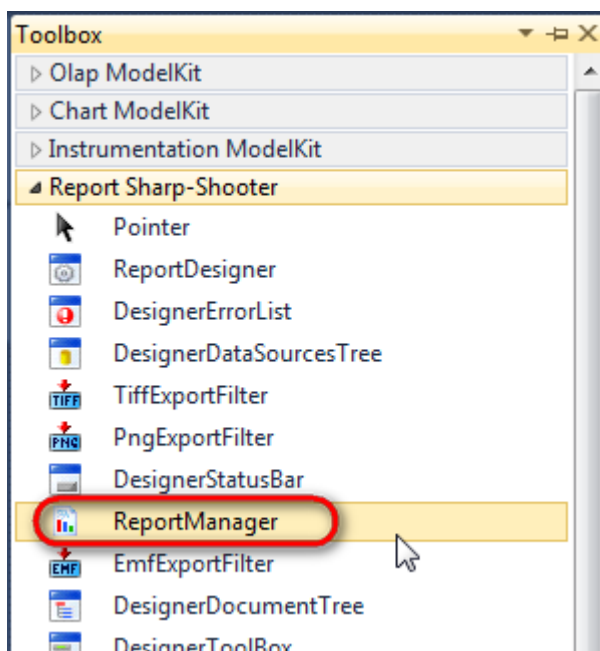


Step 3

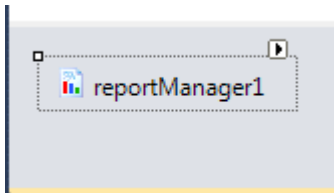
Open main form of the application by double click on the "Form1.cs" in the Solution Explorer.



Click on the "ReportManager" on the Toolbox and place this component onto the form. This component is designed to store collections of report templates and data sources.

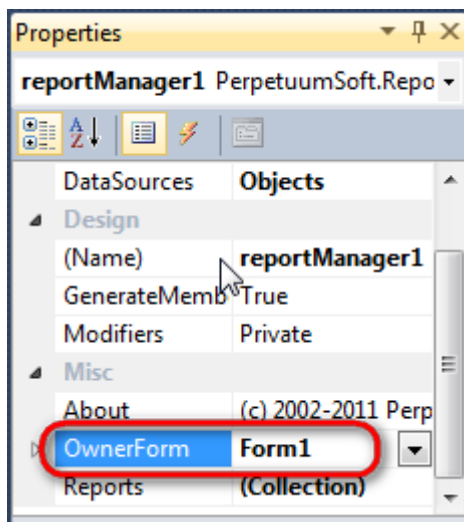


The component is available in the lower part of the window.



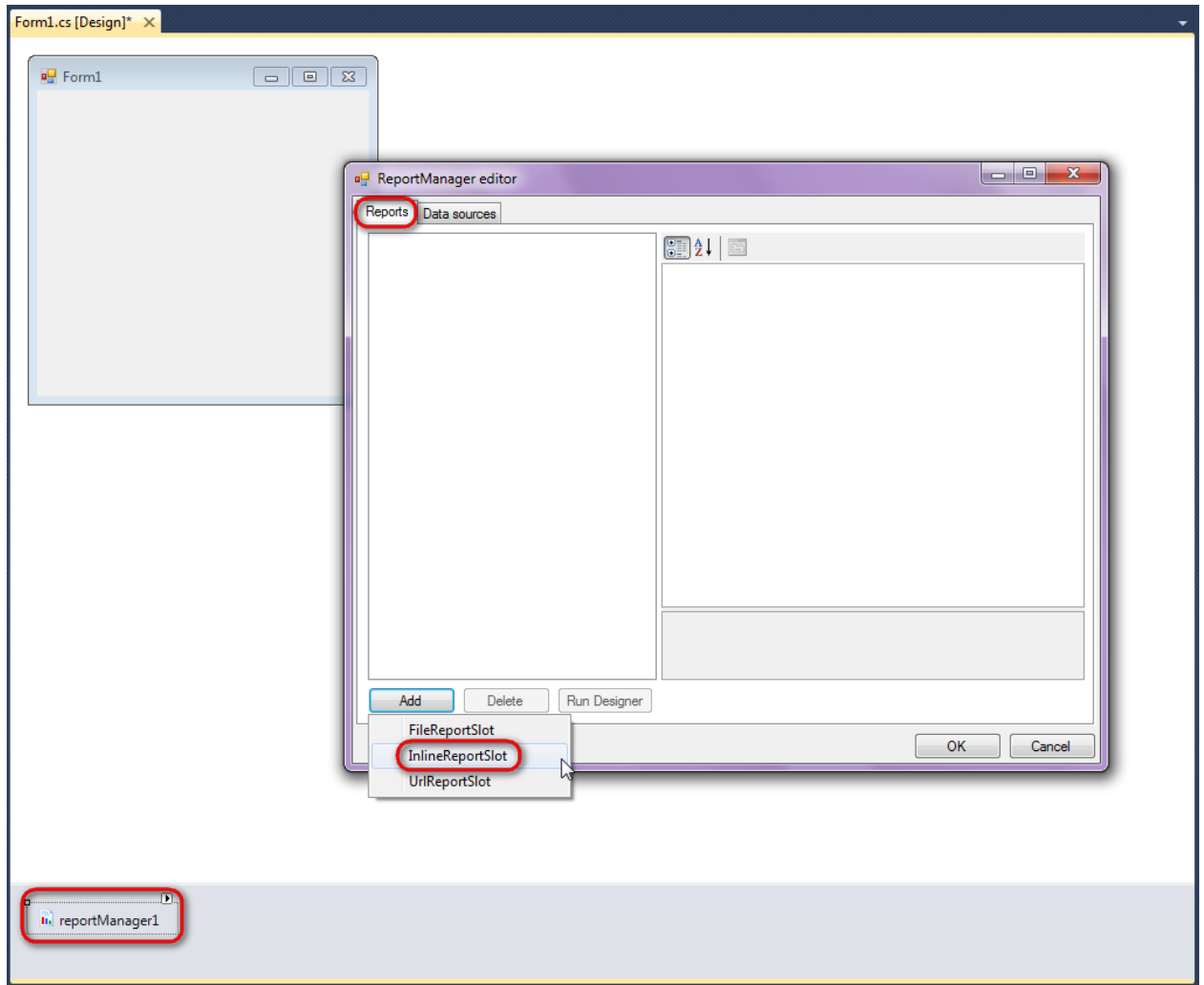
Step 4

On the property grid, initialize OwnerForm property of the ReportManager by selecting the form it is located on.



Step 5

Double click on ReportManager to open ReportManager editor.

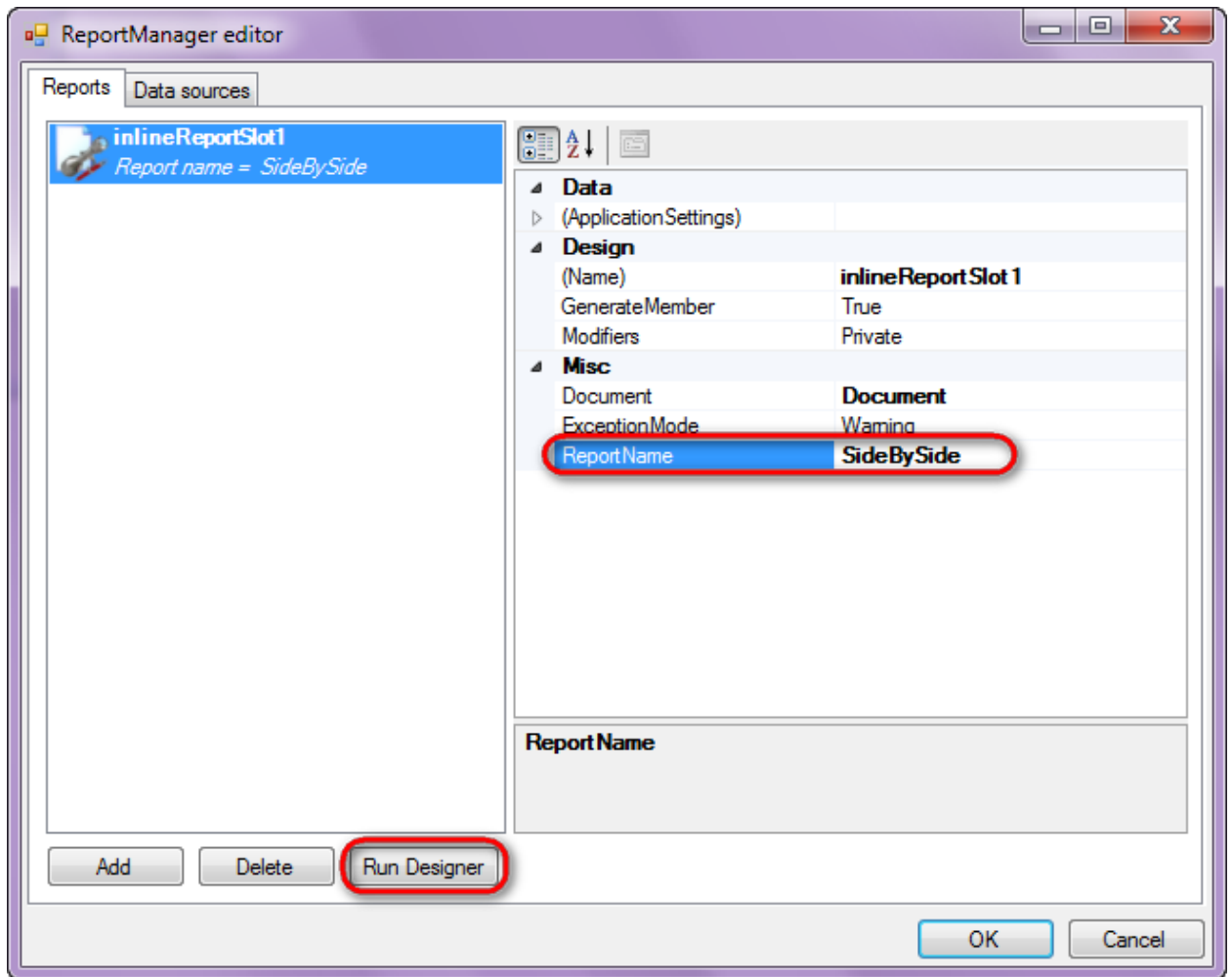


On the "Reports" tab, click "Add" and select "InlineReportSlot".

Step 6

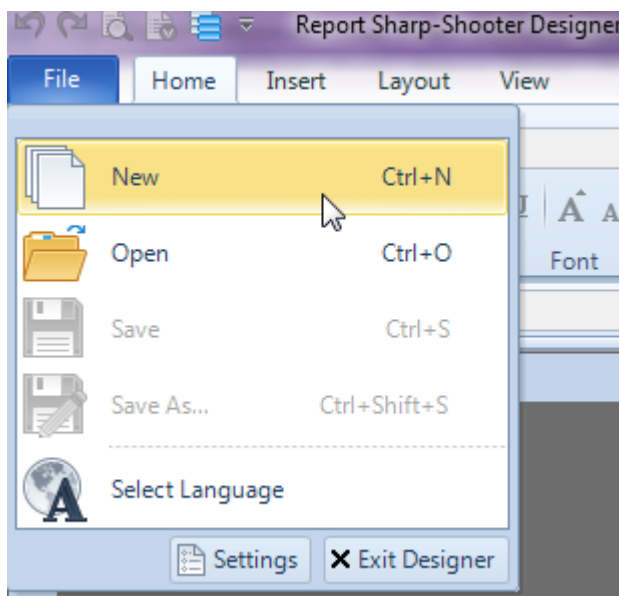
Set name of the report in the property ReportName – "SideBySide".

Click "Run Designer" in order to open template editor – Report Designer.

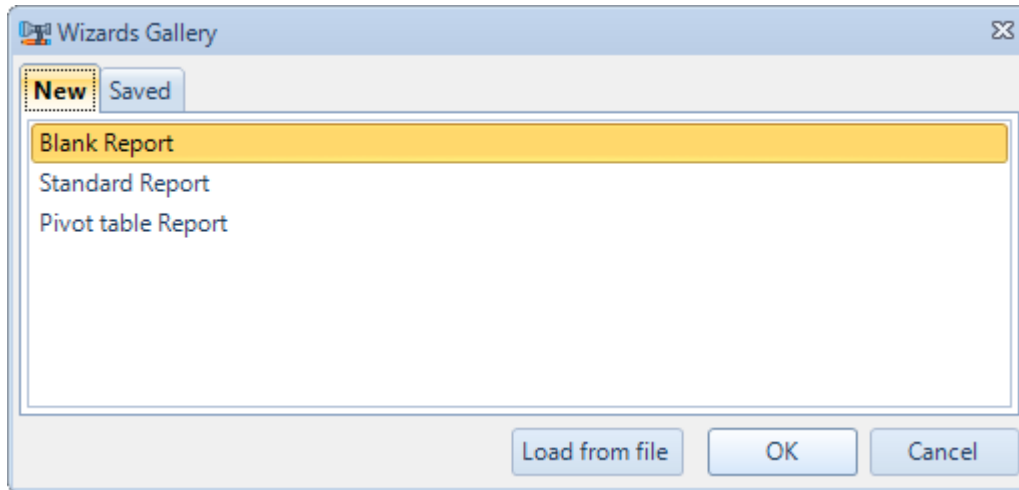


Step 7

Create new empty template – select item File\New from the main menu.

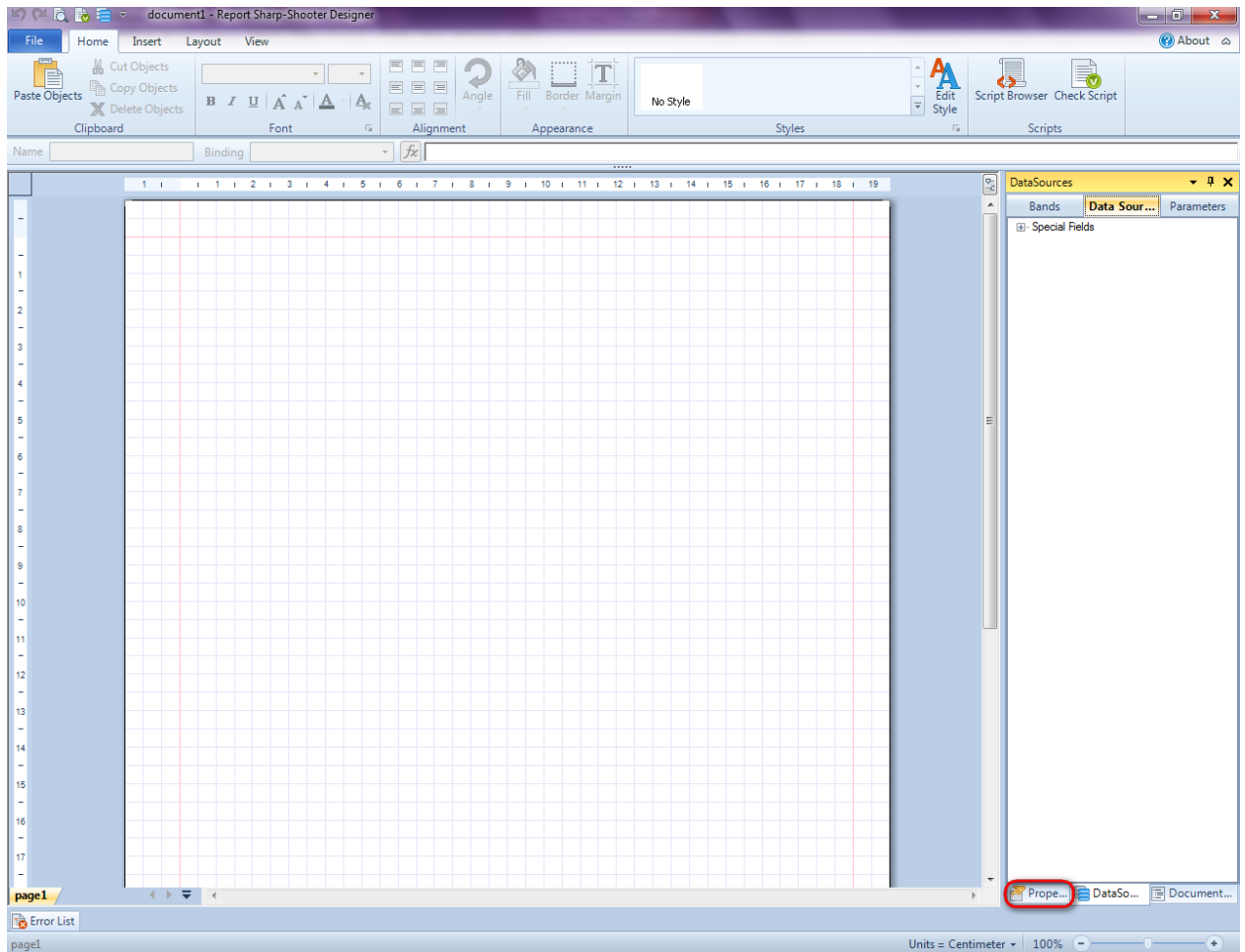


Select "Blank Report" in the Wizards Gallery and click "OK".

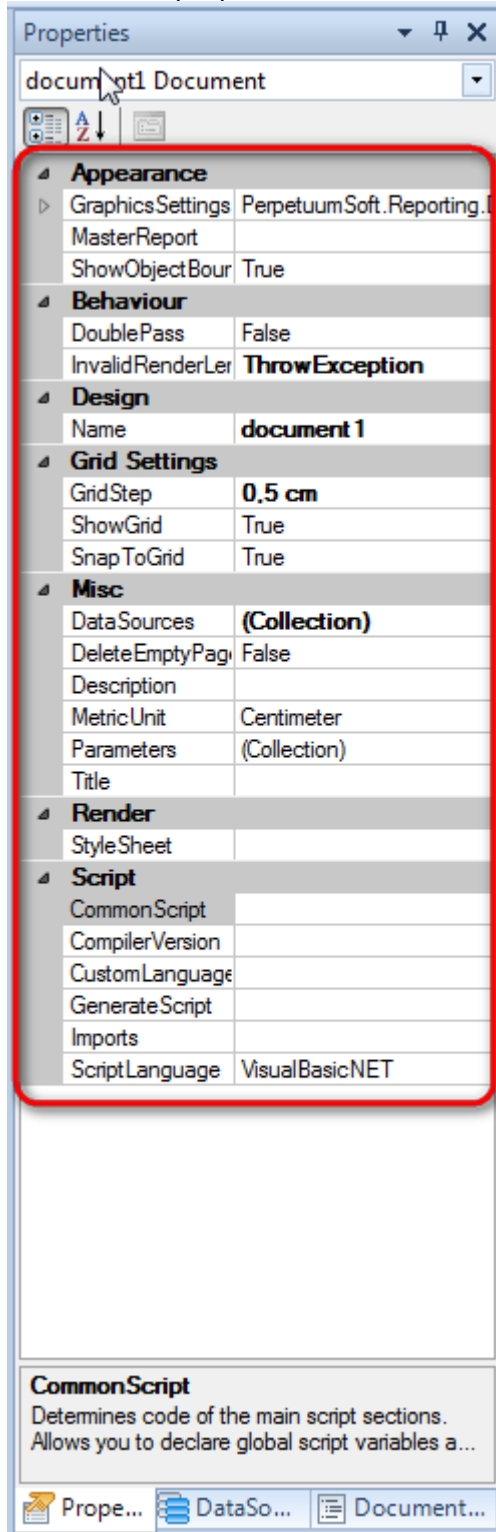


Step 8

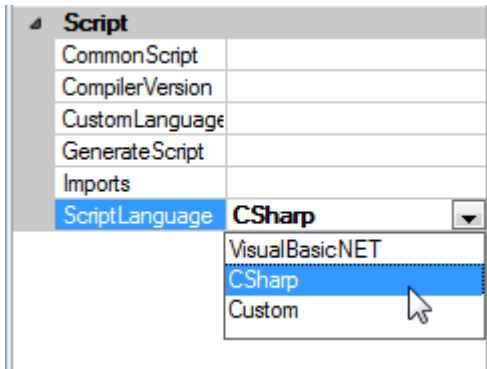
Click the "Properties" tab of the tool window in the right part of the designer.



You will see properties of the edited template on the “Properties” tab

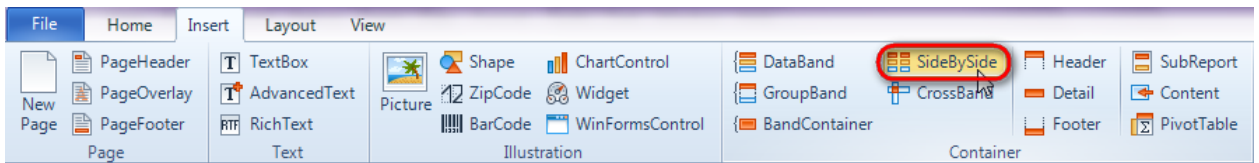


Set property ScriptLanguage = CSharp.



Step 9

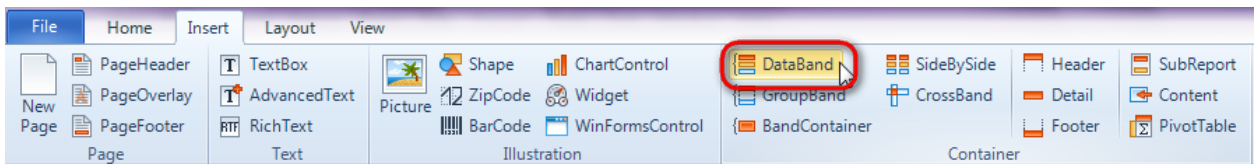
Press "SideBySide" button on the Insert tab in the group Container.



Click on the template area to add this band to the template

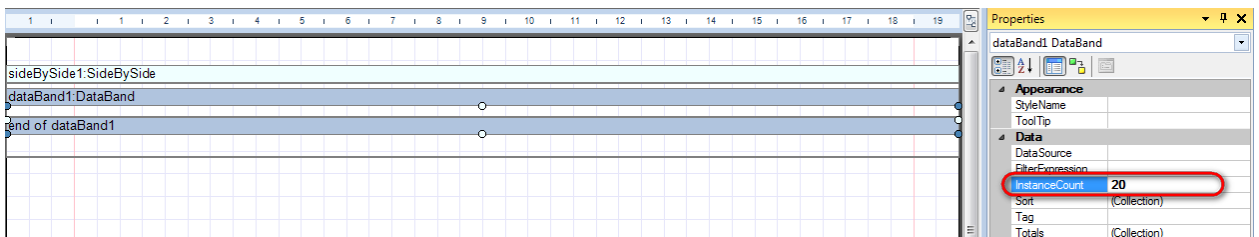
Step 10

Press "DataBand" button on the Insert tab in the group Container.



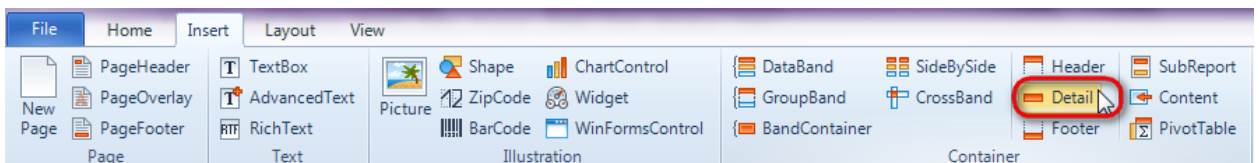
Click on the SideBySide band area to add DataBand inside SideBySide band.

Set property InstanceCount = 20.



Step 11

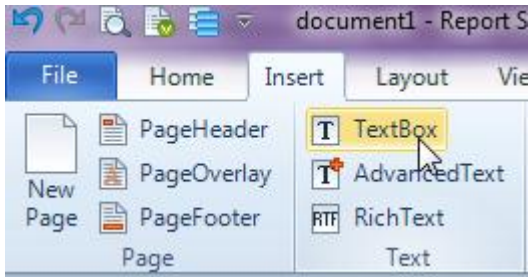
Press "Detail" button on the Insert tab in the group Container.




Click on the DataBand area to add Detail band inside DataBand.

Step 12

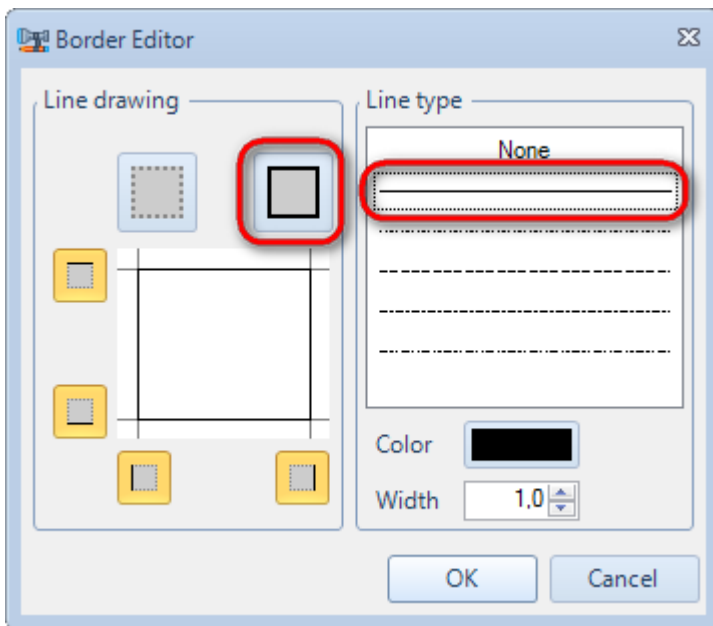
Press button "TextBox" on the Insert tab in the group Text.



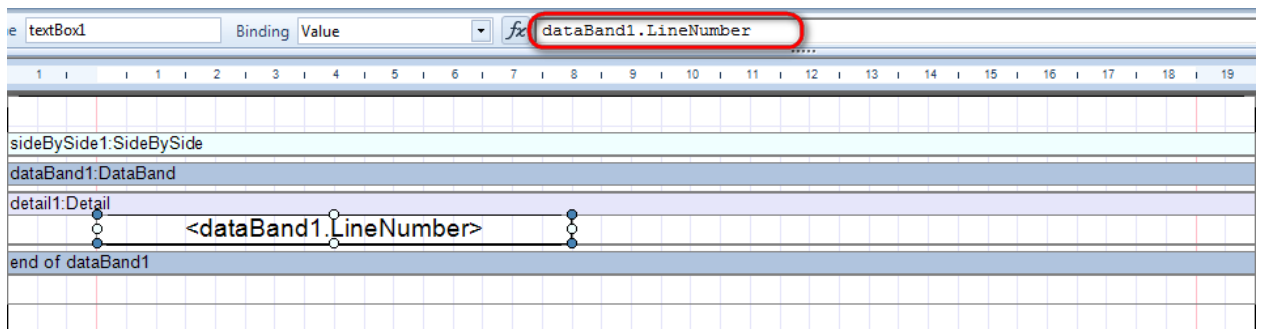
Click on the Detail band area to add TextBox element inside Detail band.

Select Border property, click button  to open Border Editor.

Set borders of the TextBox element.



Change TextBox size. Set property Value = dataBand1.LineNumber.



Step 13

Add one more DataBand inside sideBySide1 band. Set InstanceCount = 10.

Step 14

Add Detail band inside dataBand2. Add TextBox element so that it is located on the right of the TextBox from the dataBand1, set Border property. Set Value = dataBand2.LineNumber. Change element size.

Report template should look as follows:

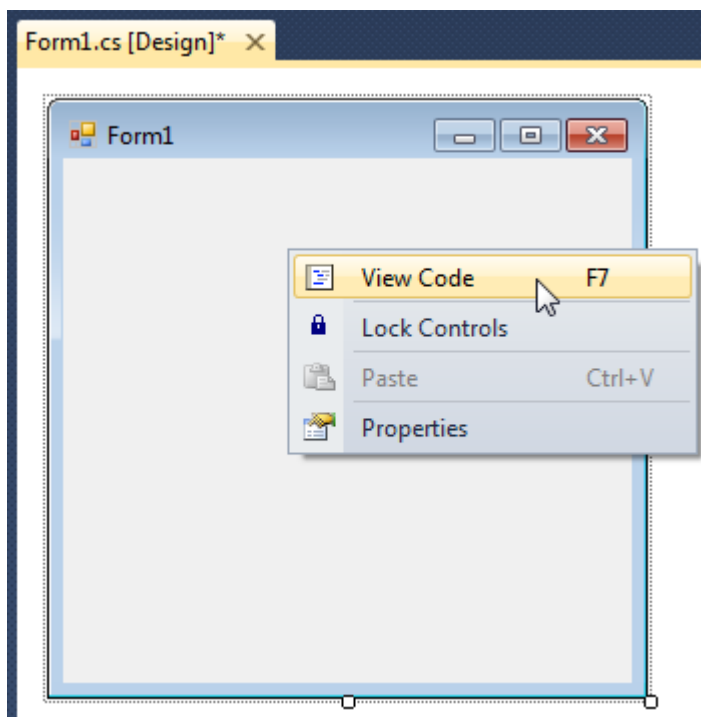
sideBySide1:SideBySide
dataBand1:DataBand
detail1:Detail
<dataBand1.LineNumber>
end of dataBand1
dataBand2:DataBand
detail2:Detail
<dataBand2.LineNumber>
end of dataBand2

Step 15

Save template, close Report Designer.

Step 16

Right click on the application form and select "View Code" in the context menu to view code.



Add code to display report to the class constructor. Create RenderCompleted event handler of the InlineReportSlot object.

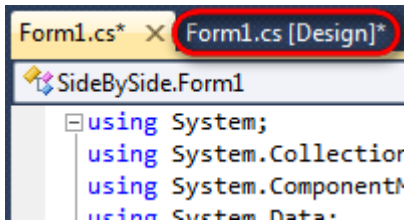
```

public Form1 ()
{
    InitializeComponent ();
    inlineReportSlot1.RenderCompleted += new
EventHandler (reportSlot_RenderCompleted);
}
private void reportSlot_RenderCompleted (object sender, EventArgs e)
{
    using (PerpetuumSoft.Reporting.View.PreviewForm previewForm = new
PerpetuumSoft.Reporting.View.PreviewForm (inlineReportSlot1))
    {
        previewForm.WindowState = FormWindowState.Maximized;
        previewForm.ShowDialog (this);
    }
}

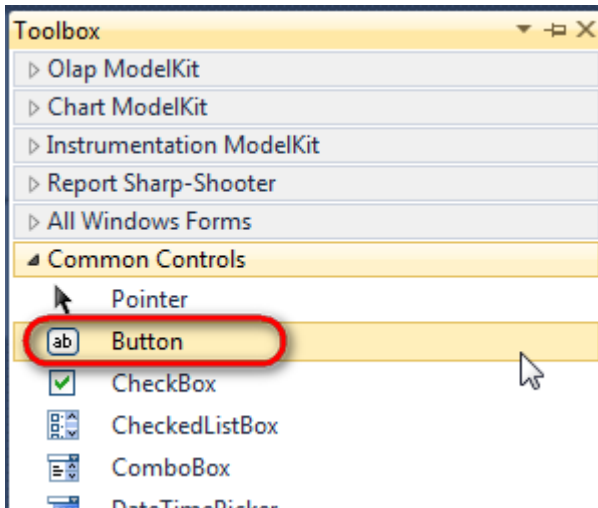
```

Step 17

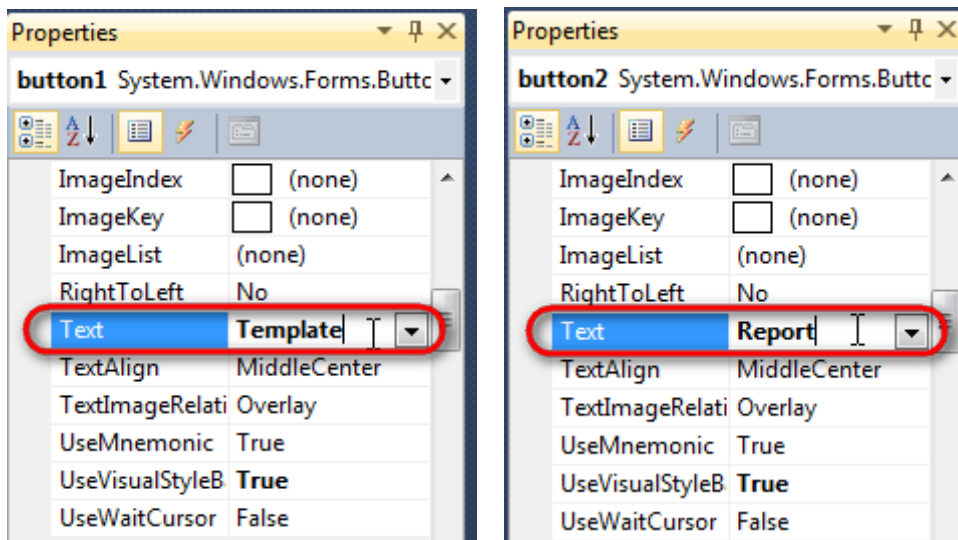
Get back to the application form by clicking "Form1.cs[Design]" tab.



Add two buttons onto the form (drag and drop "Button" element from the Toolbox onto the form).



Select Button element on the form, edit Text property on the property grid. Set Text = Template for one button and Text = Report for the other one.



Create Click event handlers for the buttons – double click on the Button on the form. Add code launching report generation to the event handler. For example, use the following code:

```
private void button1_Click(object sender, EventArgs e)
{
    inlineReportSlot1.DesignTemplate();
}

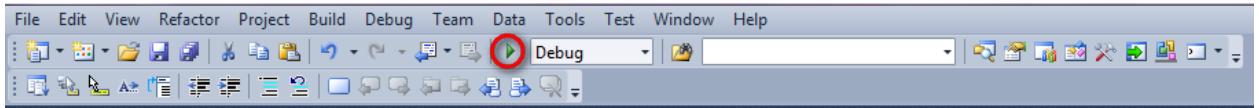
private void button2_Click(object sender, EventArgs e)
```



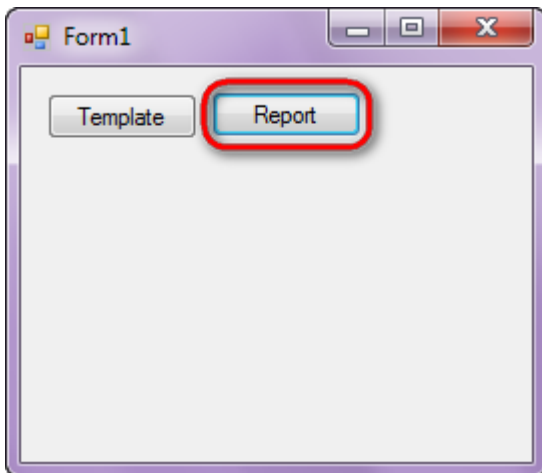
```
{  
    inlineReportSlot1.Prepare();  
}
```

Step 18

Click "Start Debugging" on the Visual Studio toolbar in order to start application.



Click the "Report" button in the opened application window.



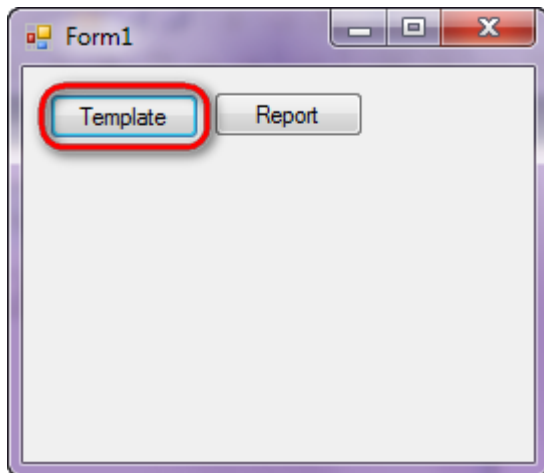
Generated report is viewed in the Report Viewer.



The screenshot shows a window titled "document1 - Preview report" with a menu bar (File, View, Navigate, Document, Help) and a toolbar. The main content is a table with 20 rows and 2 columns. The first column contains numbers 1 through 20, and the second column contains numbers 1 through 10. The status bar at the bottom indicates "Page 1 of 1" and "Zoom 158%".

1	1
2	1
3	2
4	2
5	3
6	3
7	4
8	4
9	5
10	5
11	6
12	6
13	7
14	7
15	8
16	8
17	9
18	9
19	10
20	10

To edit report template, close Report Viewer and click "Template" on the application form.



Similar application sample is located in the following folder "\\Perpetuum Software\Net ModelKit Suite\Samples\Report Sharp-Shooter\CSharp\SideBySideUsing.rst".

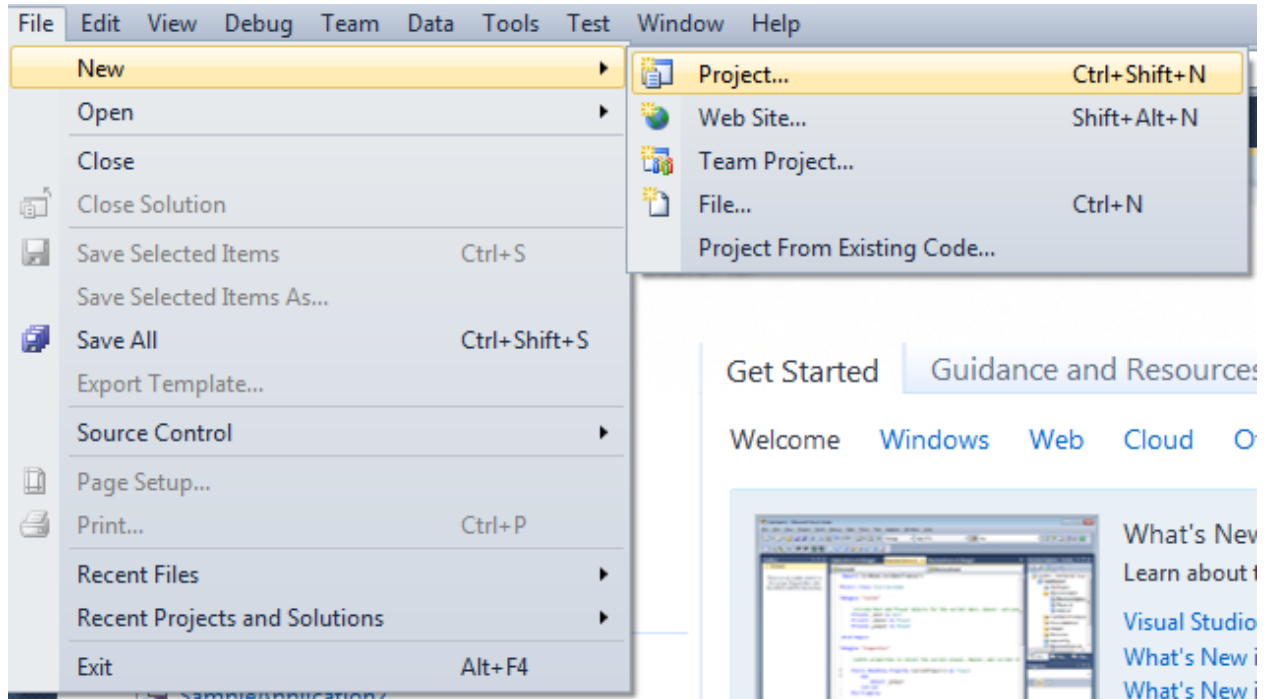


Using MS Charts in Report Sharp-Shooter 5.3+

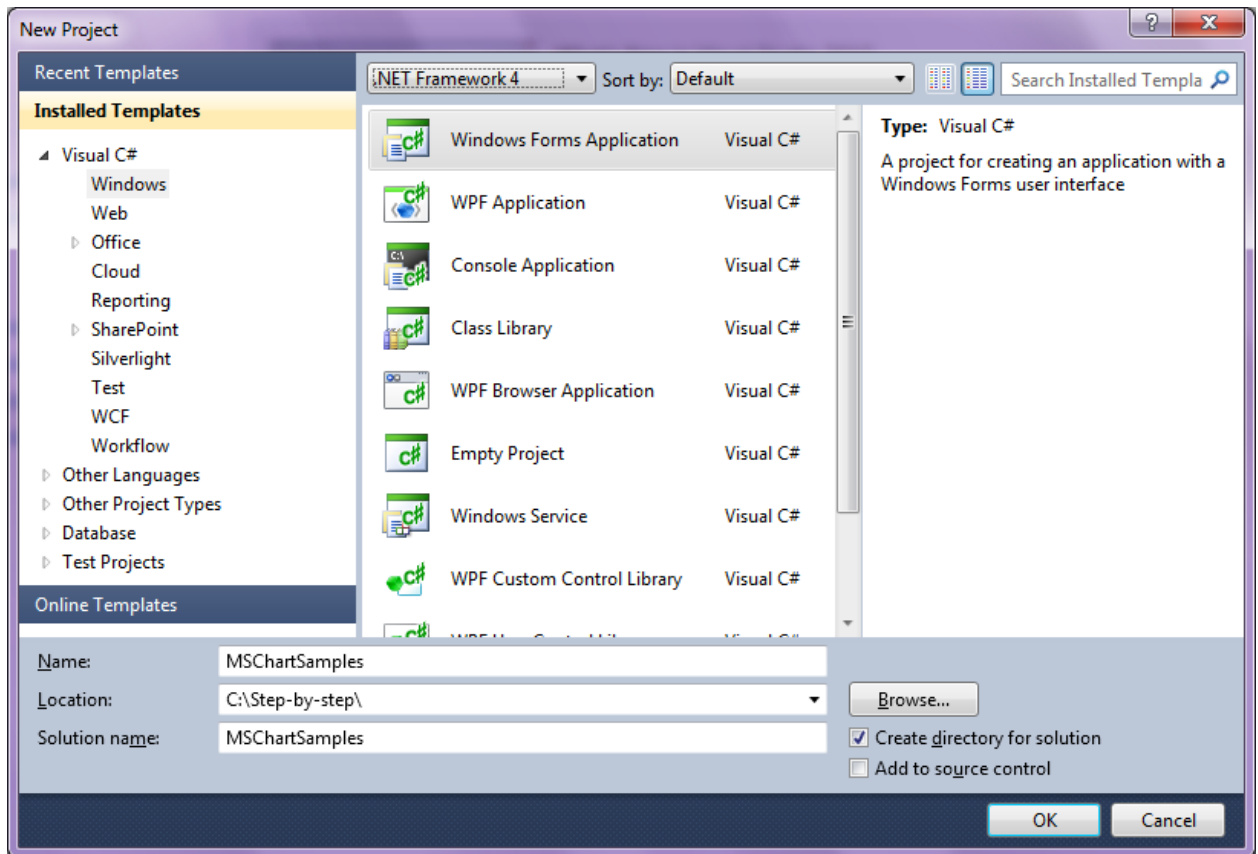
In this example we will illustrate how to use MS Charts in Report Sharp Shooter.

Step 1

Create a new project in Microsoft Visual Studio. Select New\Project from the main menu.

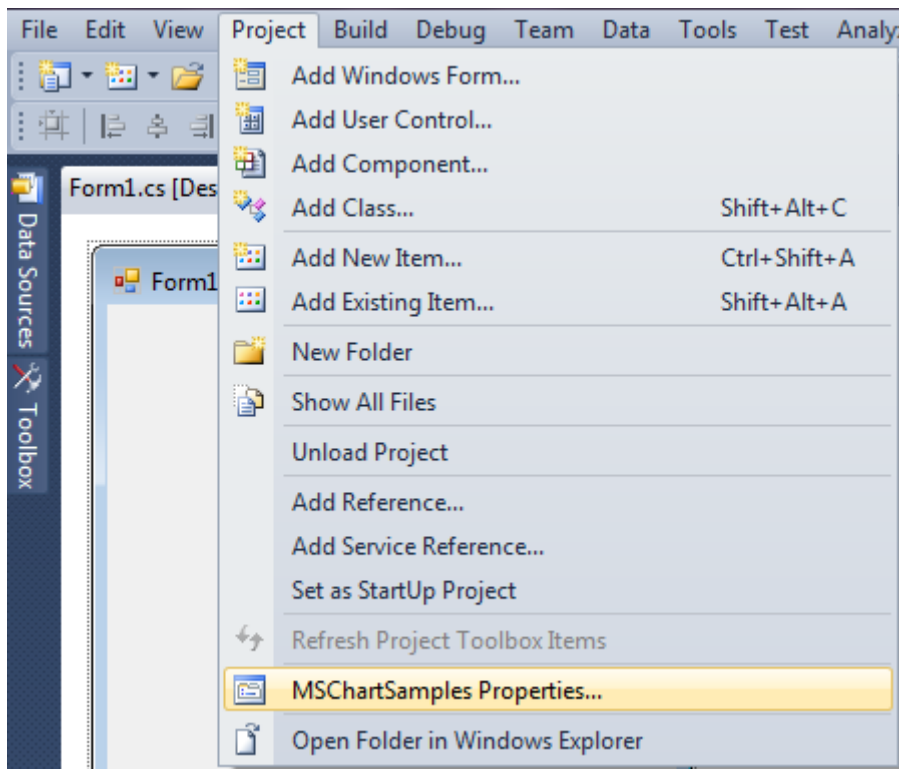


Select Windows Forms Application, set name of the project – “MSChartSamples” and set directory to save the project to.

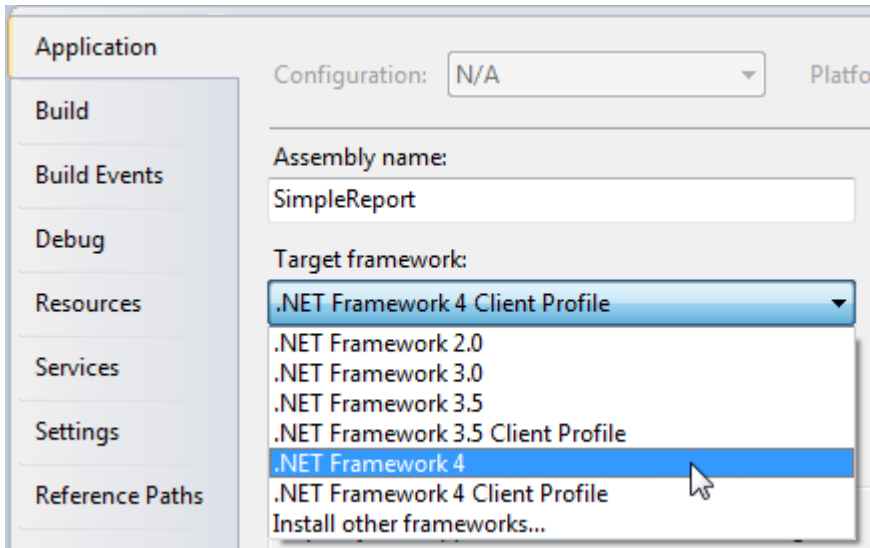


Step 2

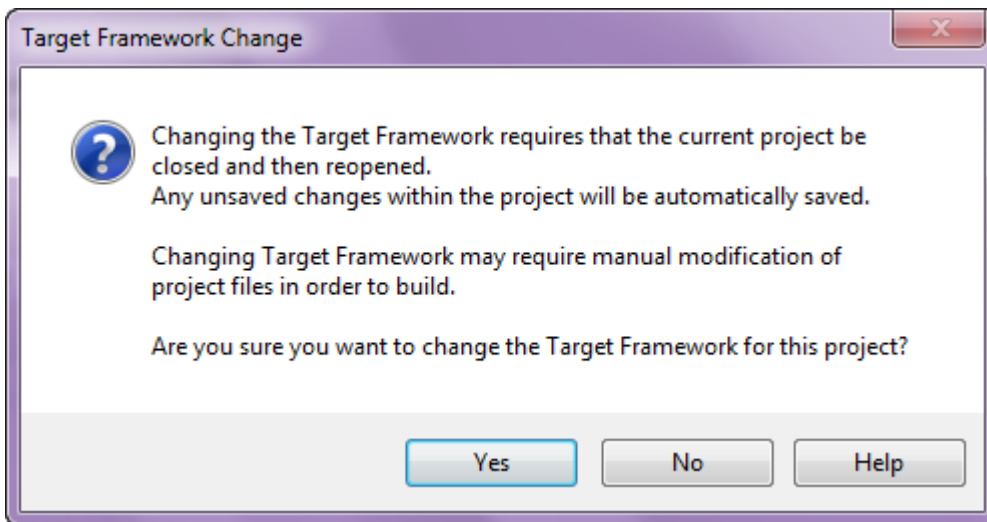
Change the project properties. Select the Project\MSChartSamples Properties... item in the main menu.



Select the Target framework\ .NET Framework4 item in the Application tab.

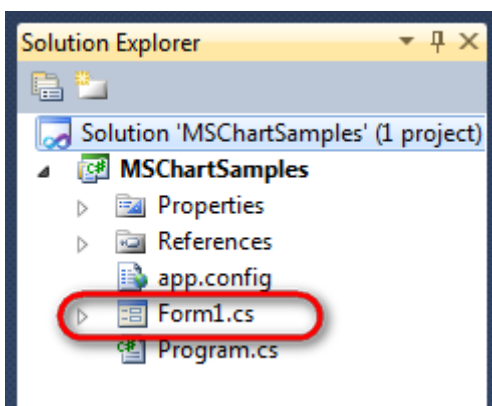


Press the “Yes” button in the opened window.

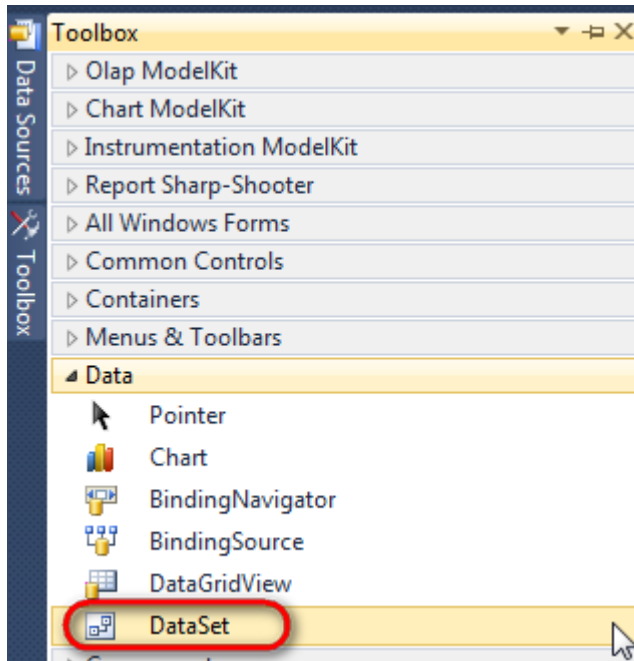


Step 3

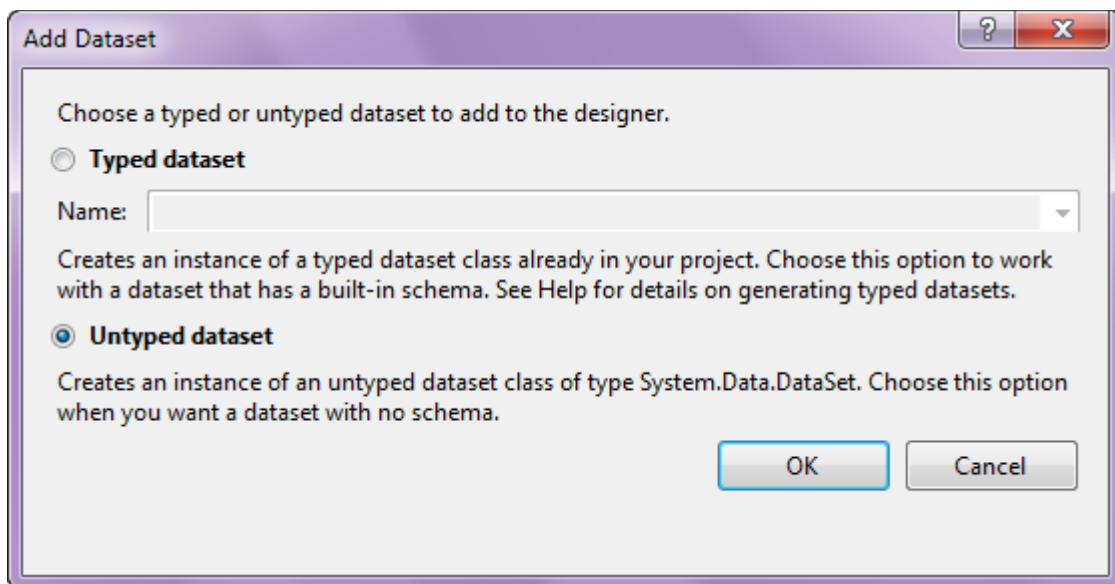
Open main form of the application in the editor by double click on “Form1.cs” in the Solution Explorer.



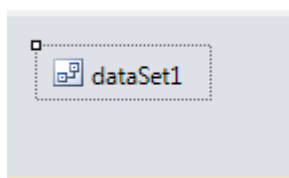
Click “DataSet” element in the Toolbox and place it onto the form.




Select "Untyped dataset", click "OK".

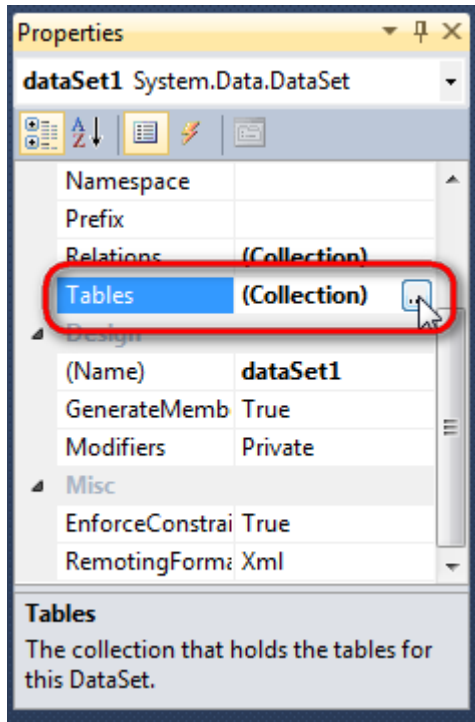


The component is displayed in the lower part of the window.

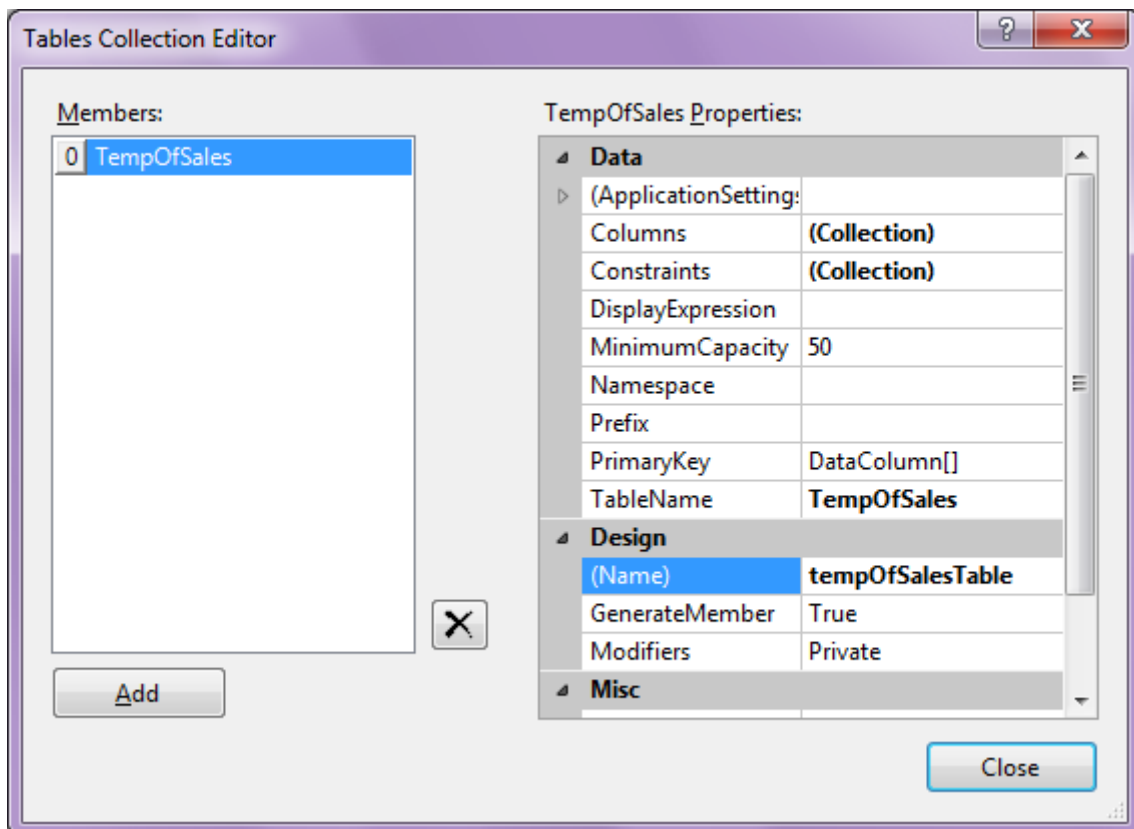


Step 4

Select dataSet1 component in the form editor. On the property grid, select "Tables" property, press button  in order to open Tables Collection Editor.

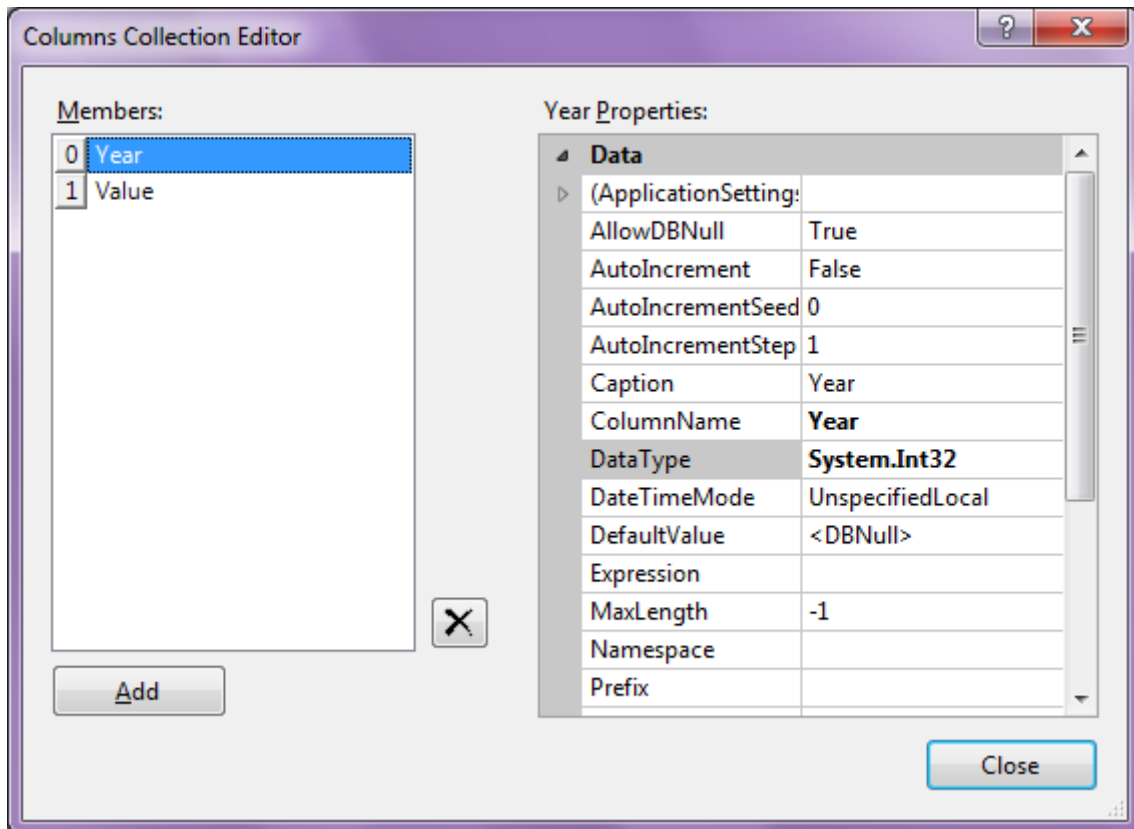


Press button "Add" in order to add a table. Set Name = tempOfSalesTable, TableName = TempOfSales.



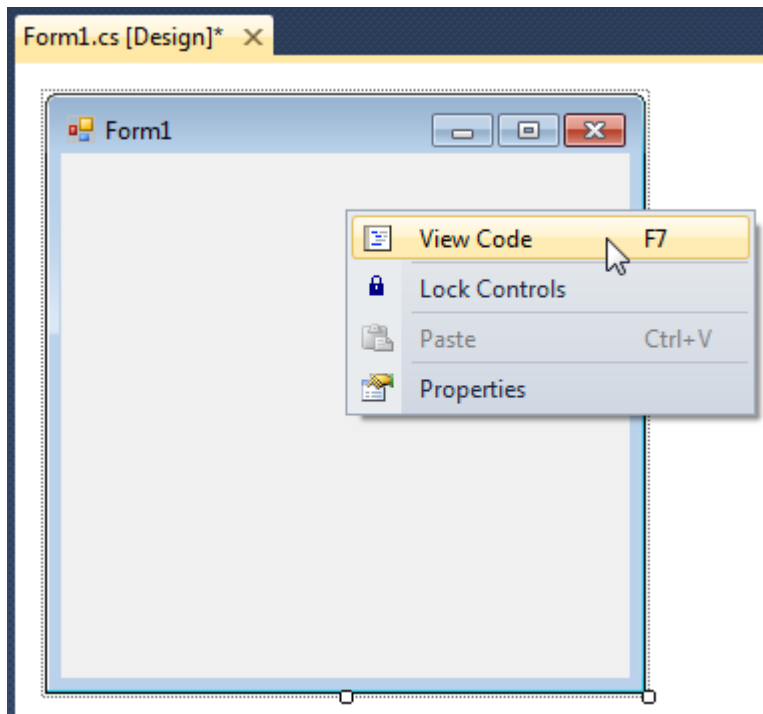
Step 5

Add the following columns: Year and Value are of the Int32 type.



Step 6

Right click on the form and select "View Code" in the context menu in order to view code.



Fill the DataSource with random information for a period from 2000 till current year-1. Add the FillDataSource function for this purpose:

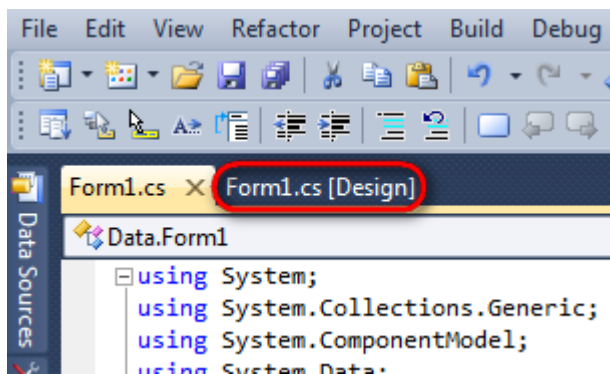
```
public Form1 ()  
{  
    InitializeComponent ();
```



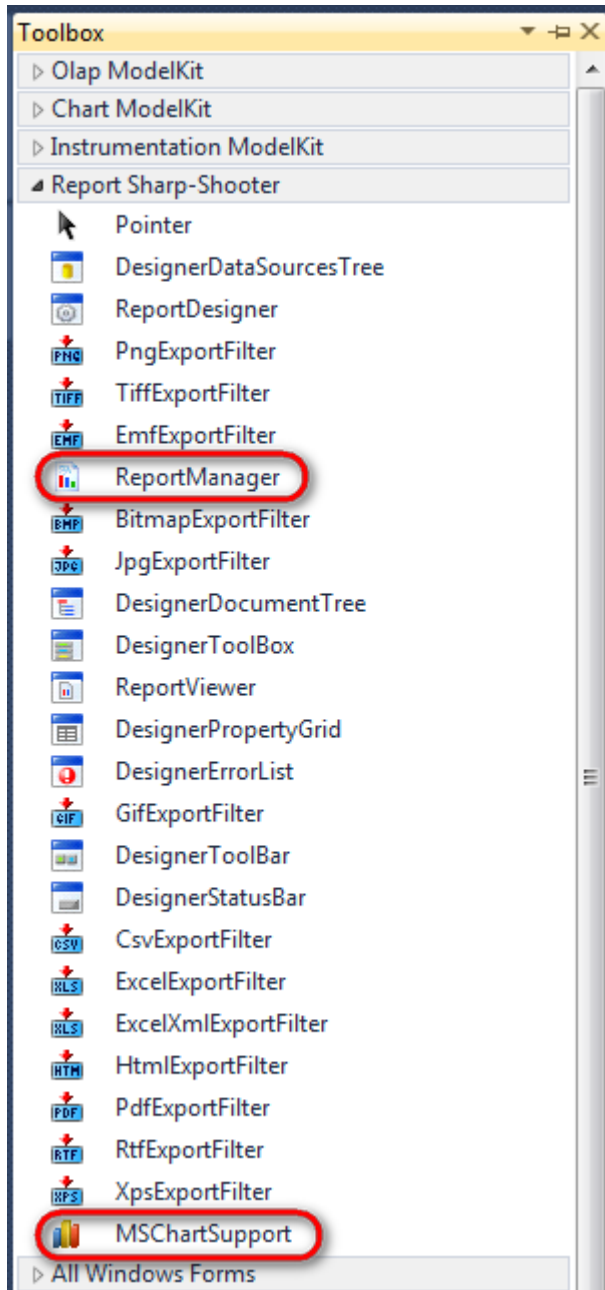
```
FillDataSource();  
}  
  
private void FillDataSource()  
{  
    // Let's add some data  
    Random random = new Random();  
    for (int year = 2000; year < DateTime.Now.Year; year++)  
    {  
        DataRow row = tempOfSalesTable.NewRow();  
        row["Year"] = year;  
        row["Value"] = random.Next(1000);  
        tempOfSalesTable.Rows.Add(row);  
    }  
}
```

Step 7

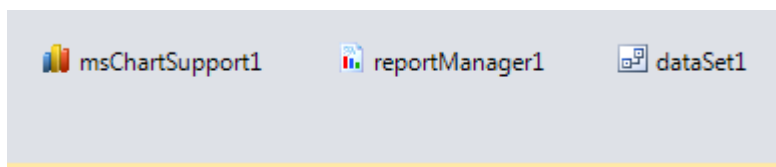
Get back to the application form by clicking the "Form1.cs[Design]" tab.



Click "ReportManager" element on the Toolbox and place it onto the form. This element is designed to store collections of report templates and data sources. Then click "MSChartSupport" element on the Toolbox and place it onto the form.

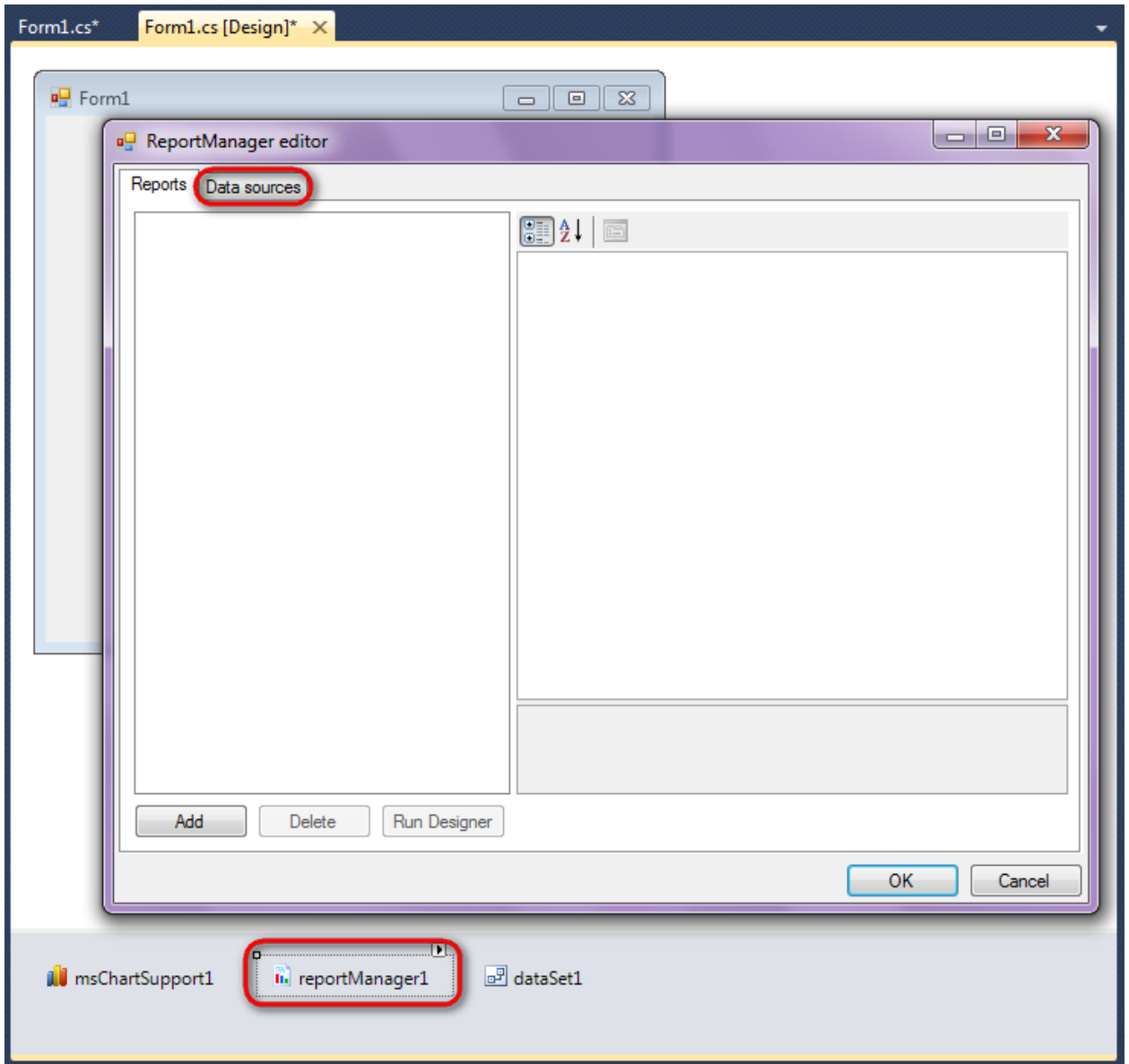


Components are displayed in the lower part of the window.

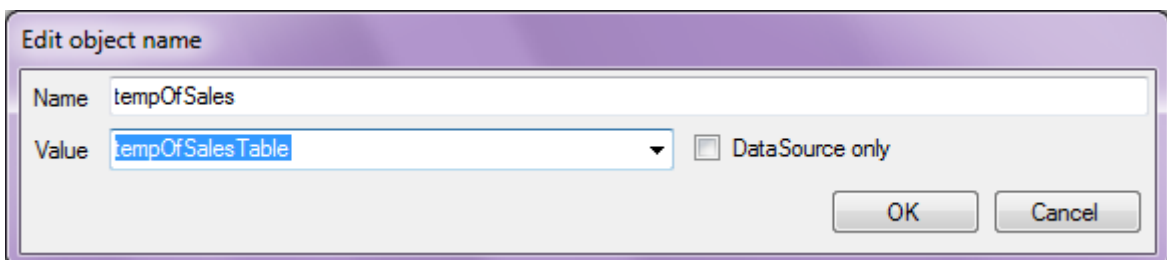


Step 8

Double click on the ReportManager component and open ReportManager editor.

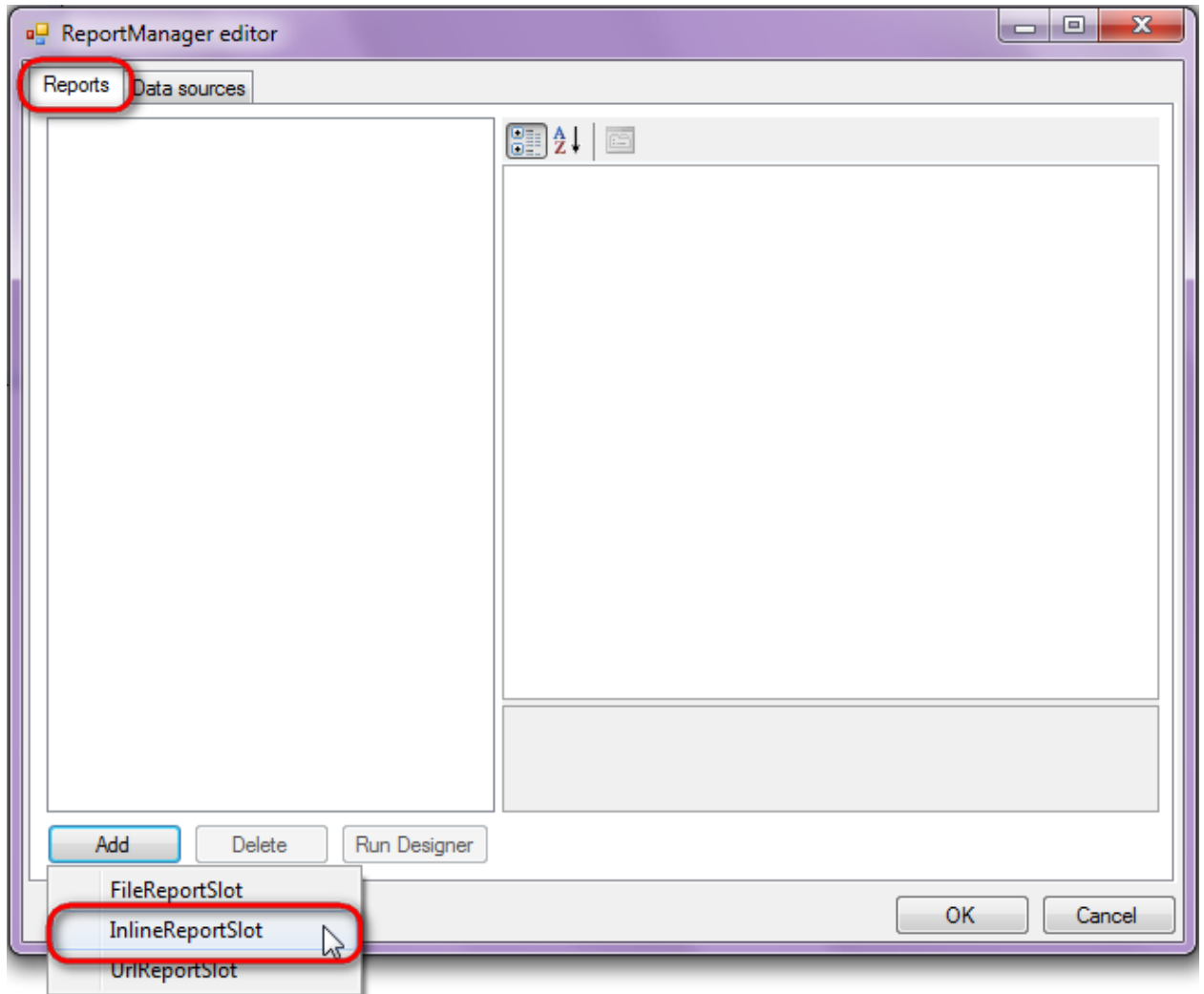


Go to "Data sources" tab, press "Add", set name of the data source - "tempOfSales", select data source value - "tempOfSalesTable".



Step 9

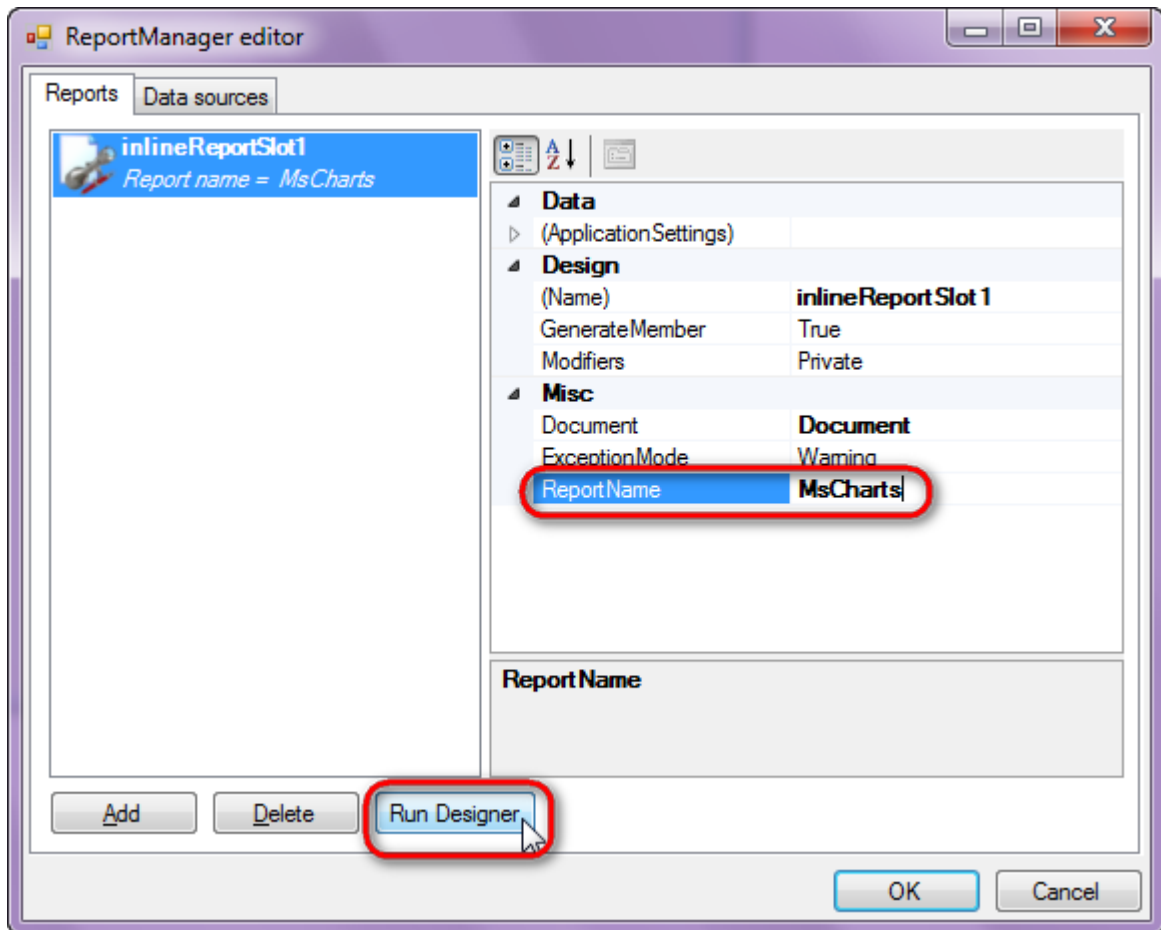
Go to "Reports" tab of the ReportManager editor, press "Add" and select "InlineReportSlot".



Step 10

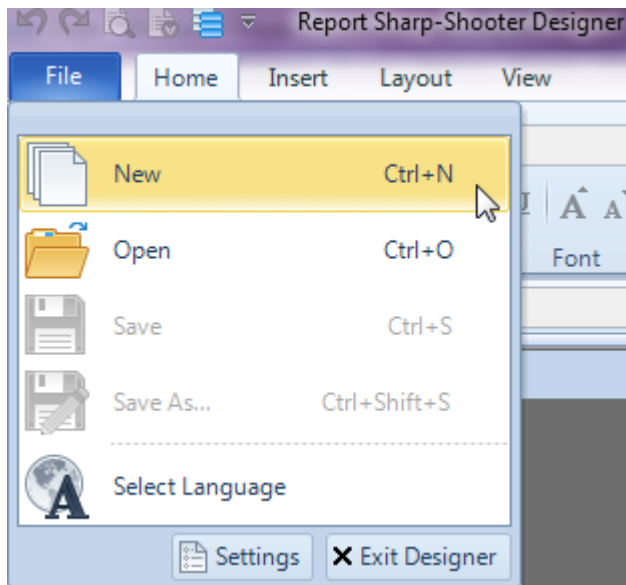
Set name of the report to "MsCharts" in the ReportName property.

Click "Run Designer" in order to open template editor – Report Designer.

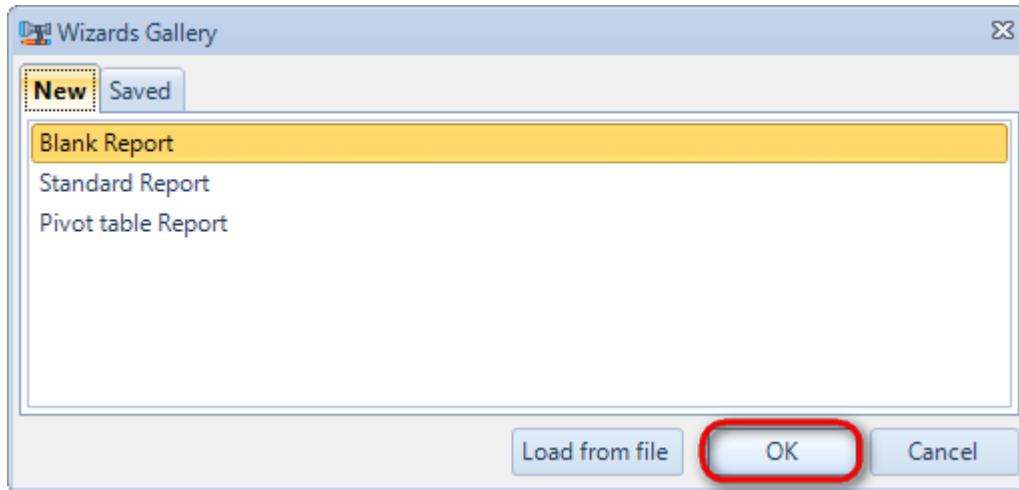


Step 11

Create new empty template – select File\New in the main menu.

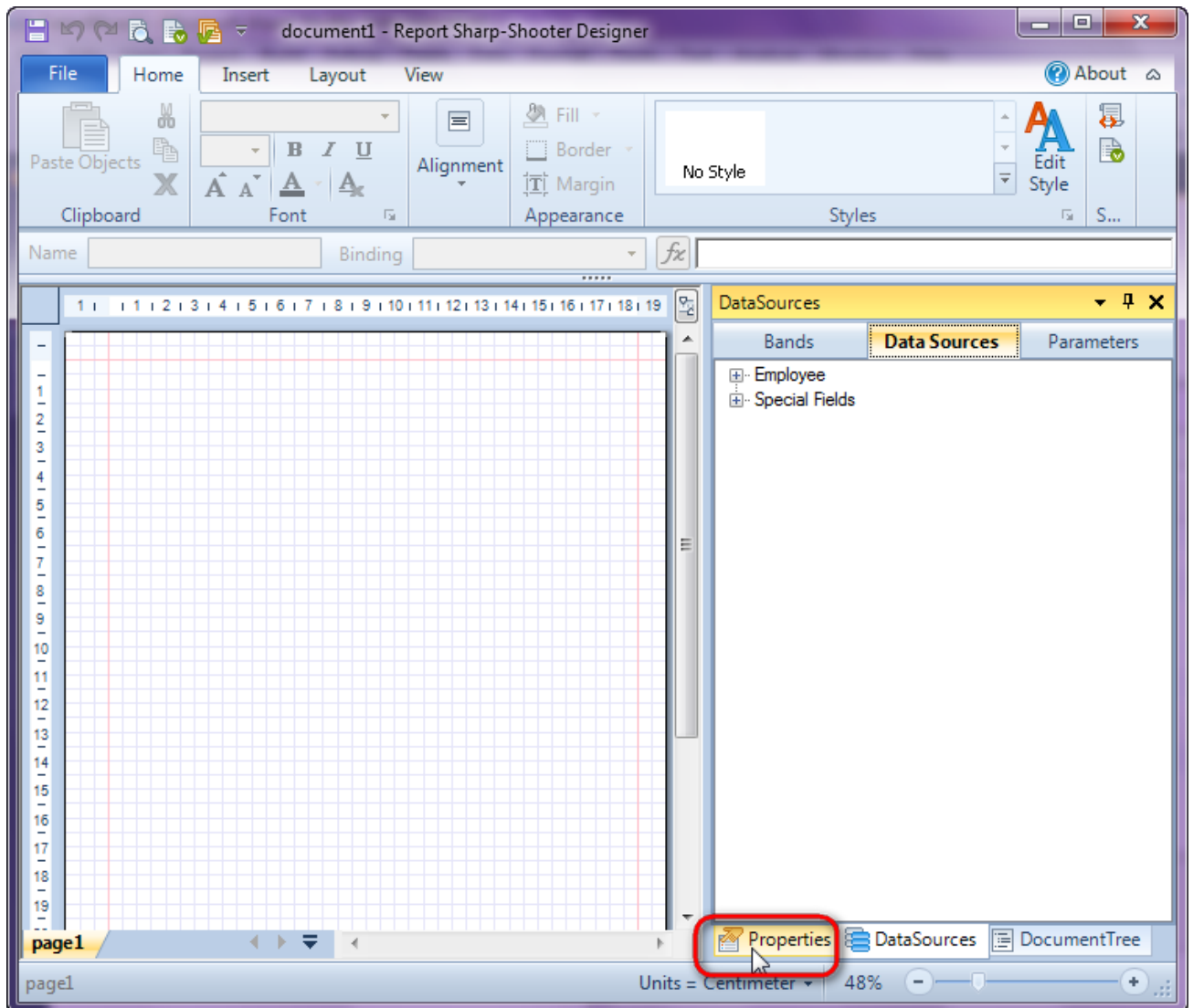


Select "Blank Report" in the Wizards Gallery and click "OK".



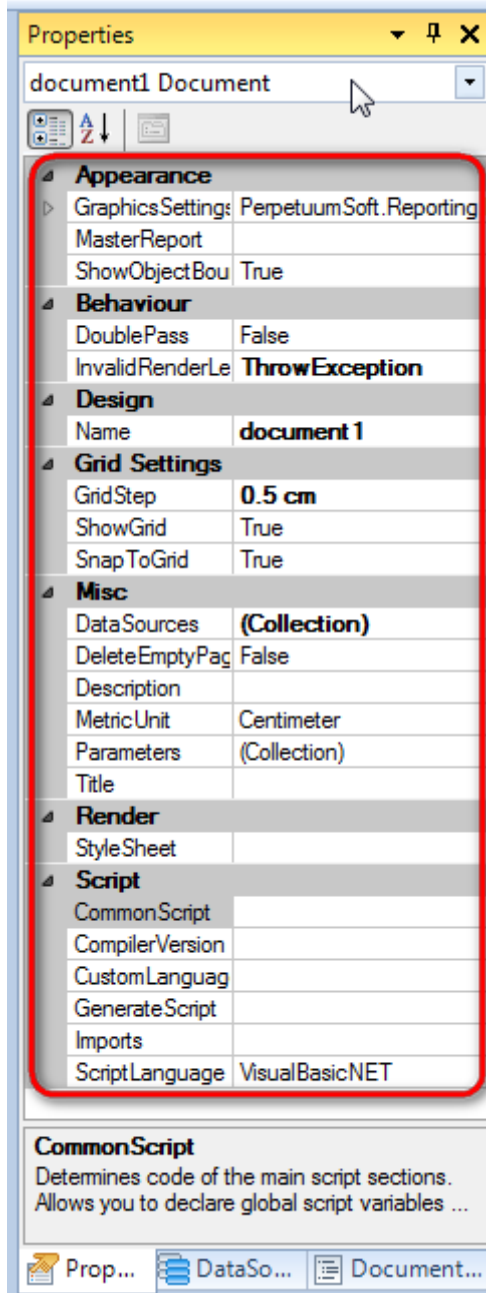
Step 12

Click the "Properties" tab of the tool window in the right part of the designer.

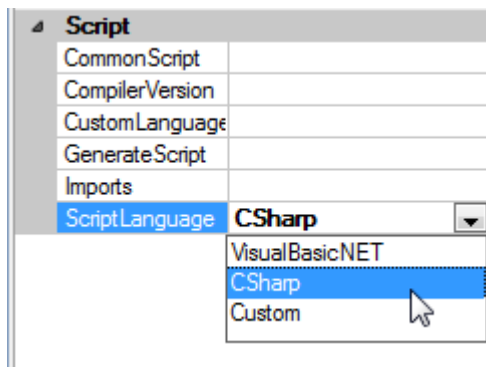




You will see properties of the edited template on the "Properties" tab

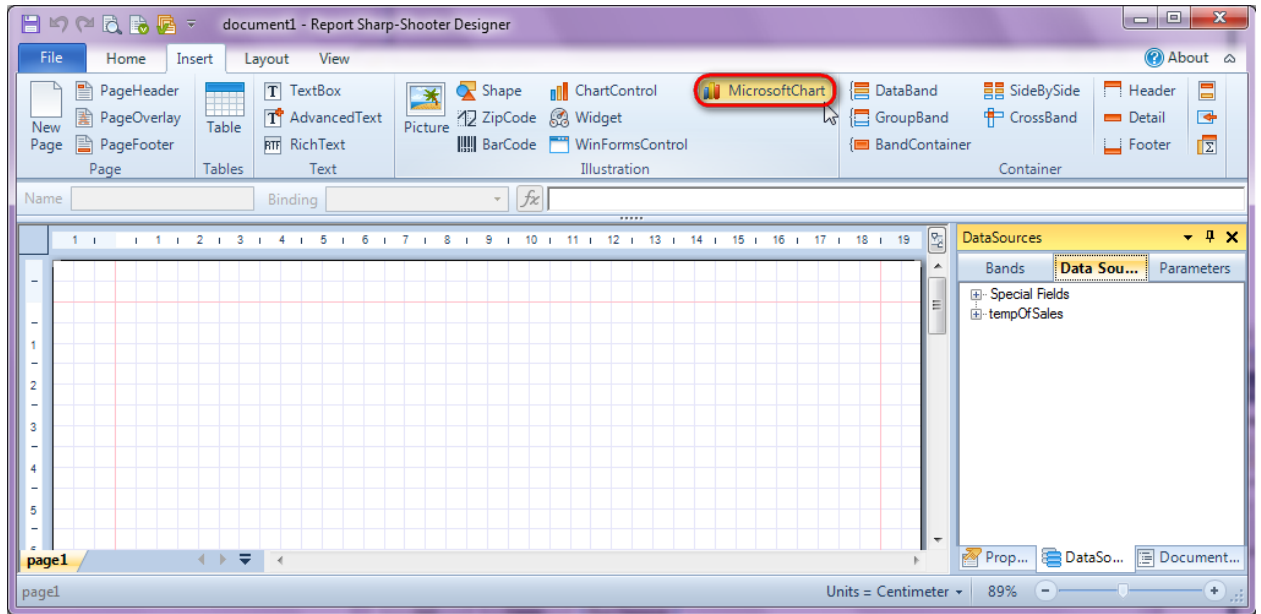


Set property ScriptLanguage = CSharp.



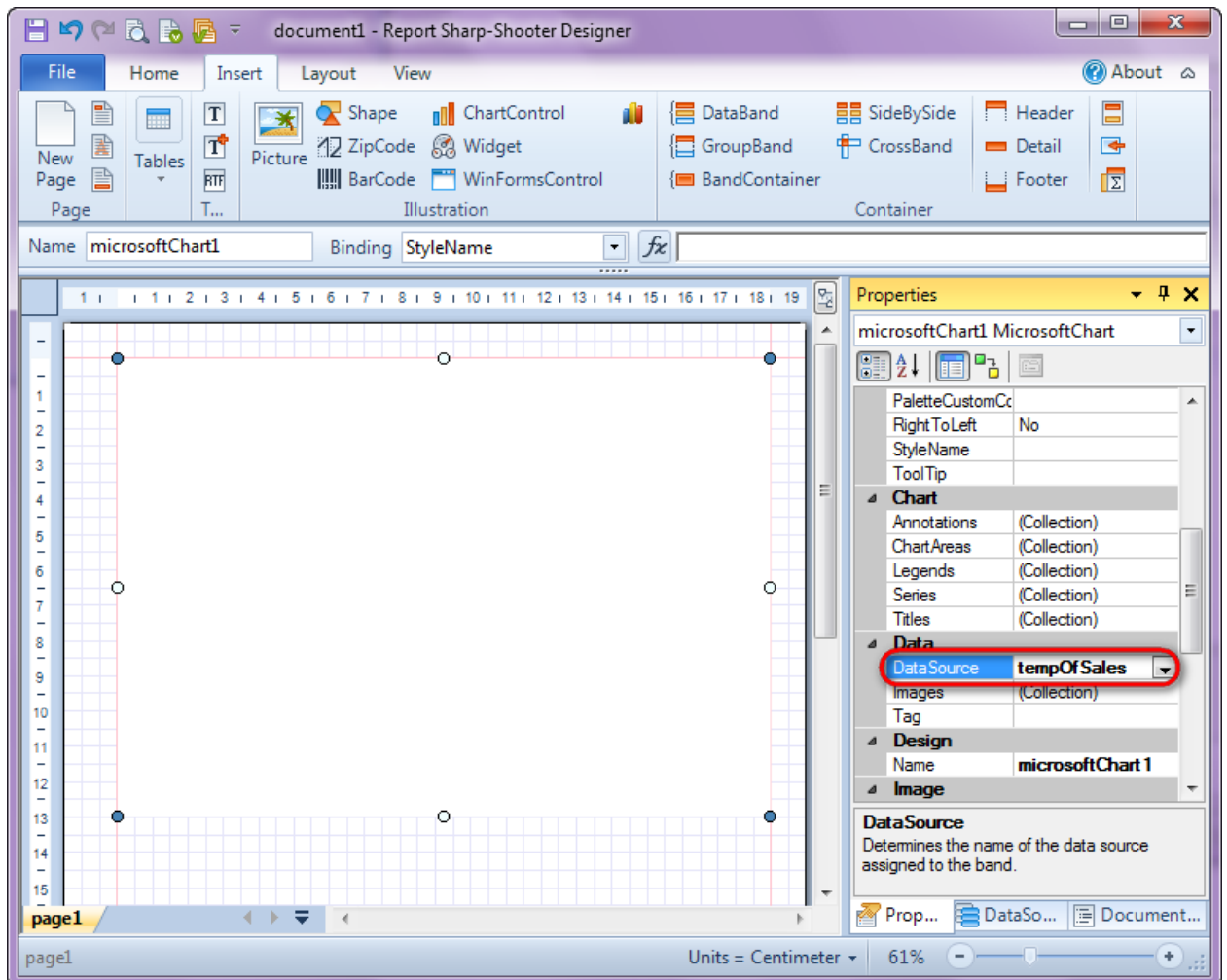
Step 13

Press "MicrosoftChart" button on the Insert tab in the Illustration Container.



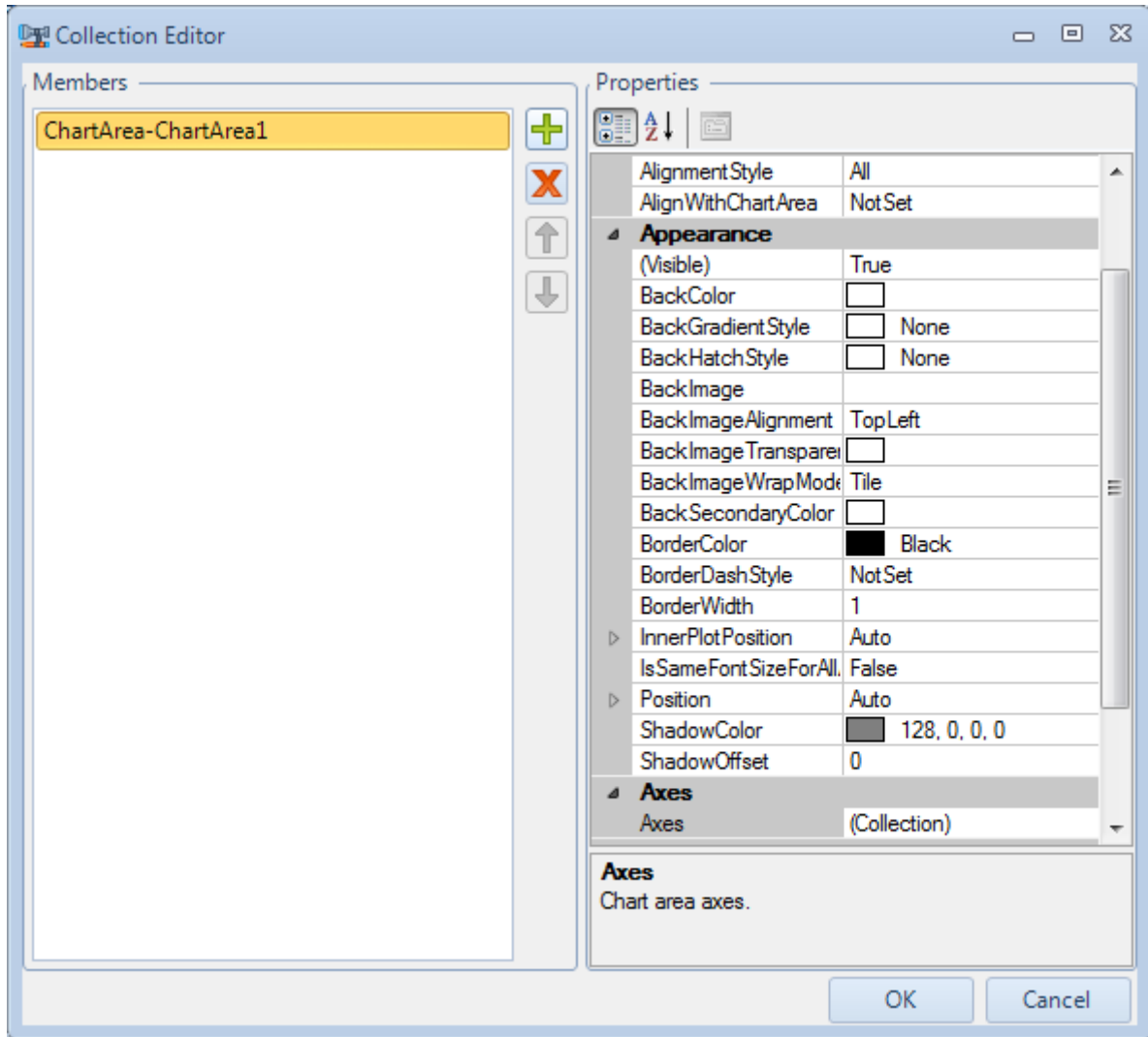
Step 14

Select the "tempOfSales" as the DataSource property value.



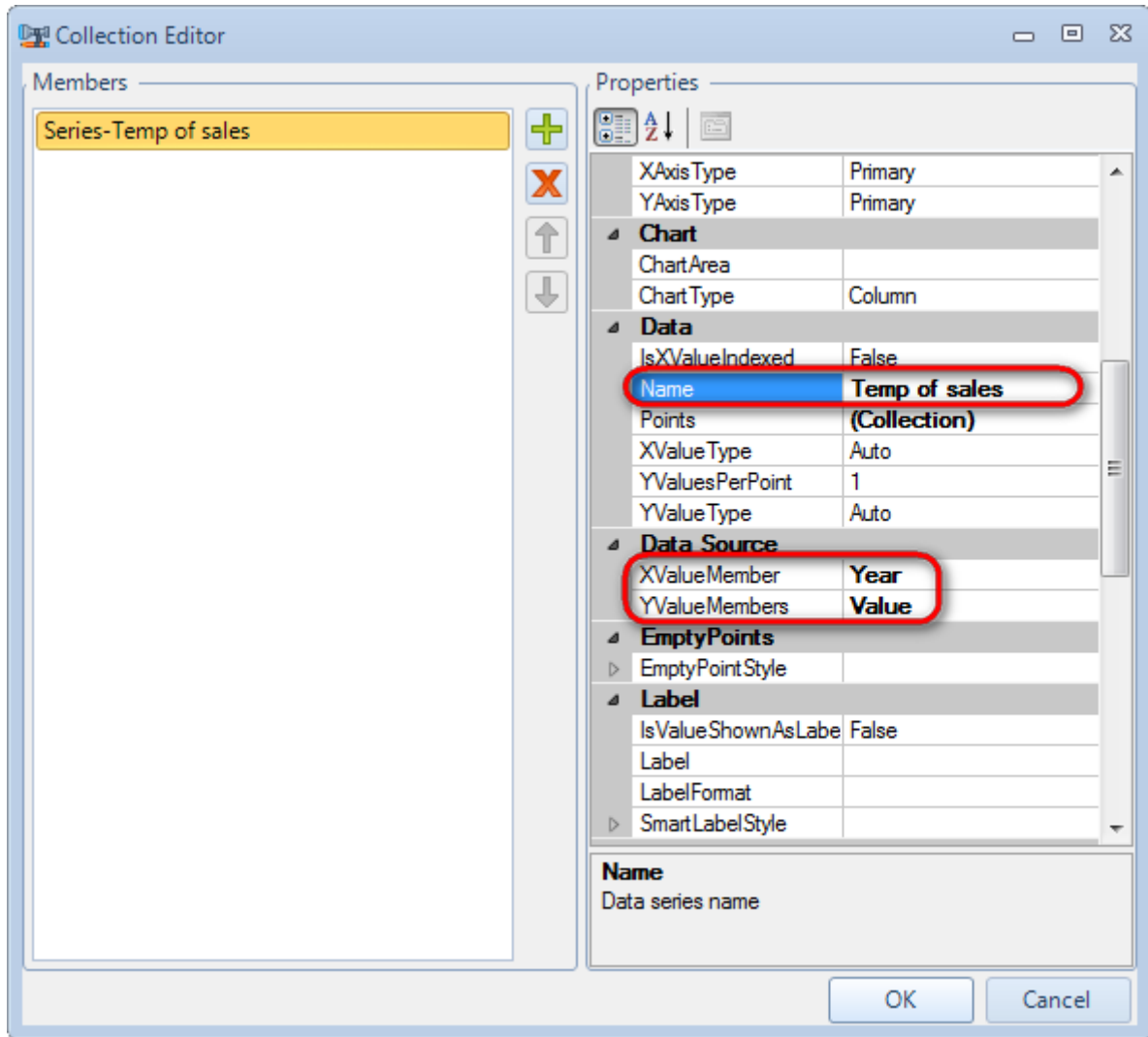
Step 15

Open the ChartAreas collection and add a Chart Area, keep all the parameters by default.



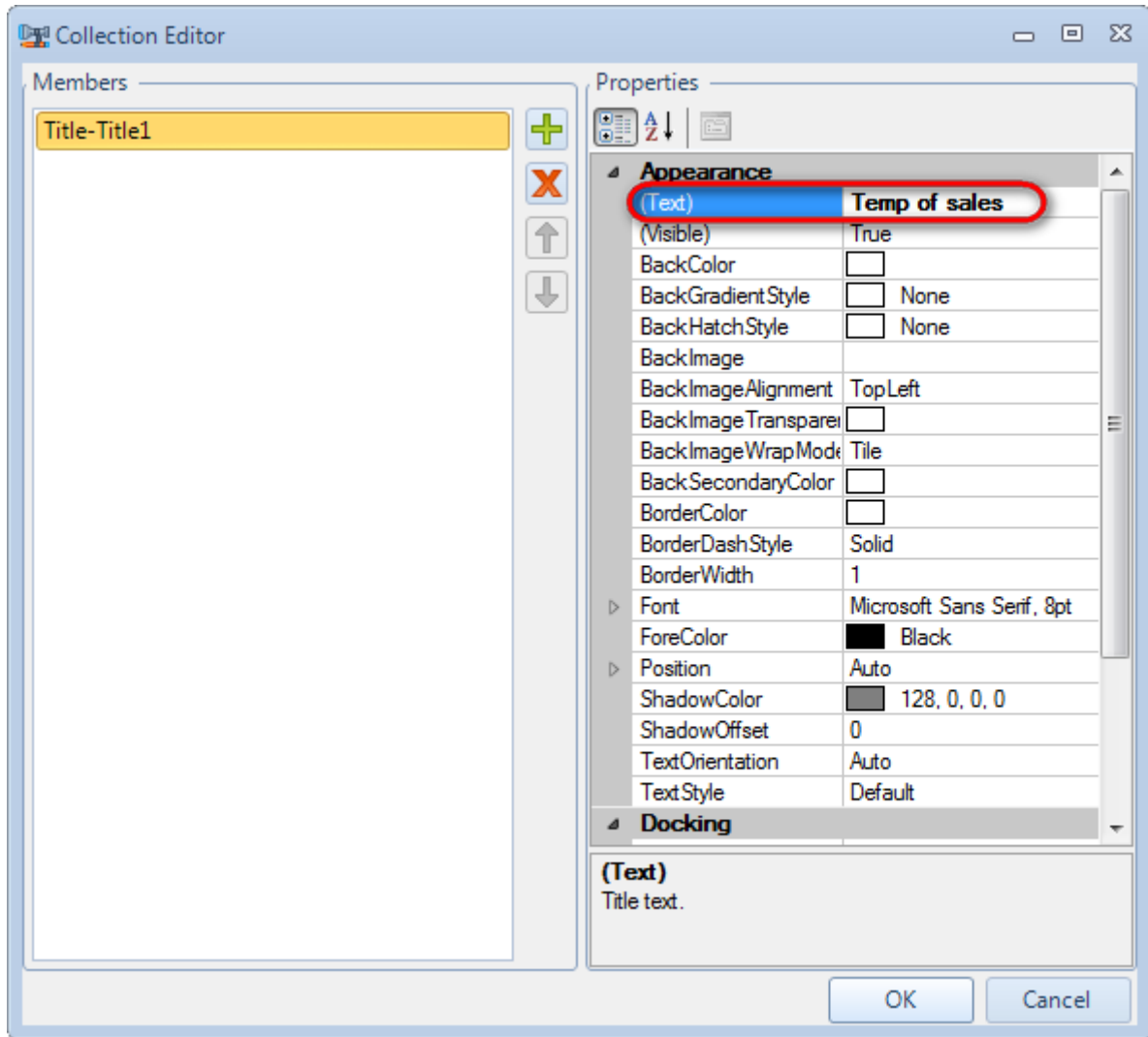
Step 16

Open Series collection and add a series. Select "Year" as XValueMember, "Value" as YValueMember, set Name = Temp of sales. The ChartArea is set automatically as "ChartArea1" after the editor is closed.



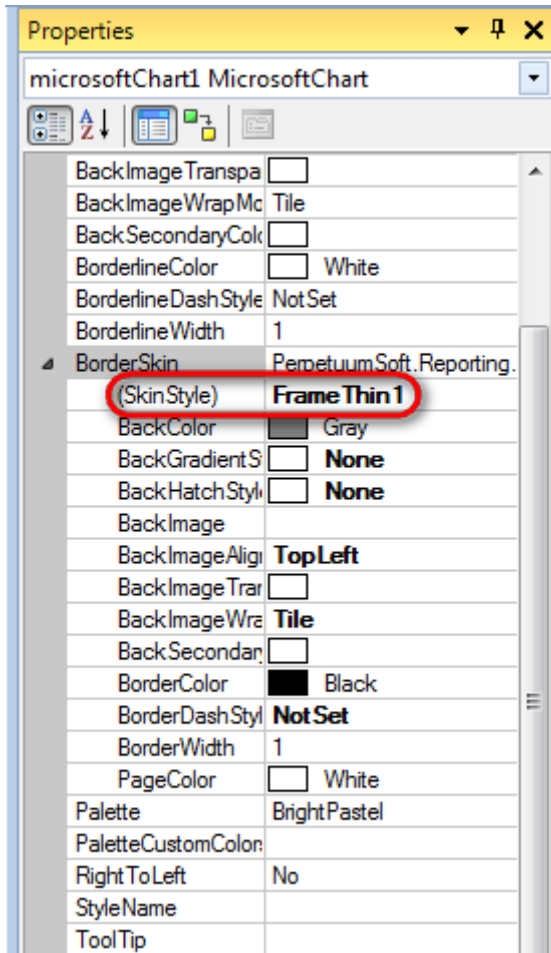
Step 17

Open the Titles collection, add a new Title, set Text = Temp of sales.



Step 18

Select the "FrameThin1" as the BorderSkin.SkinStyle.

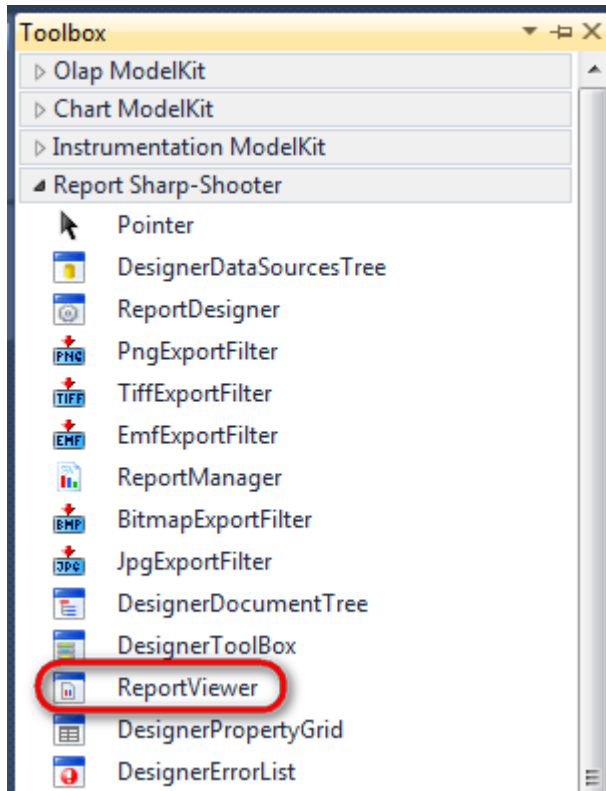


Step 19

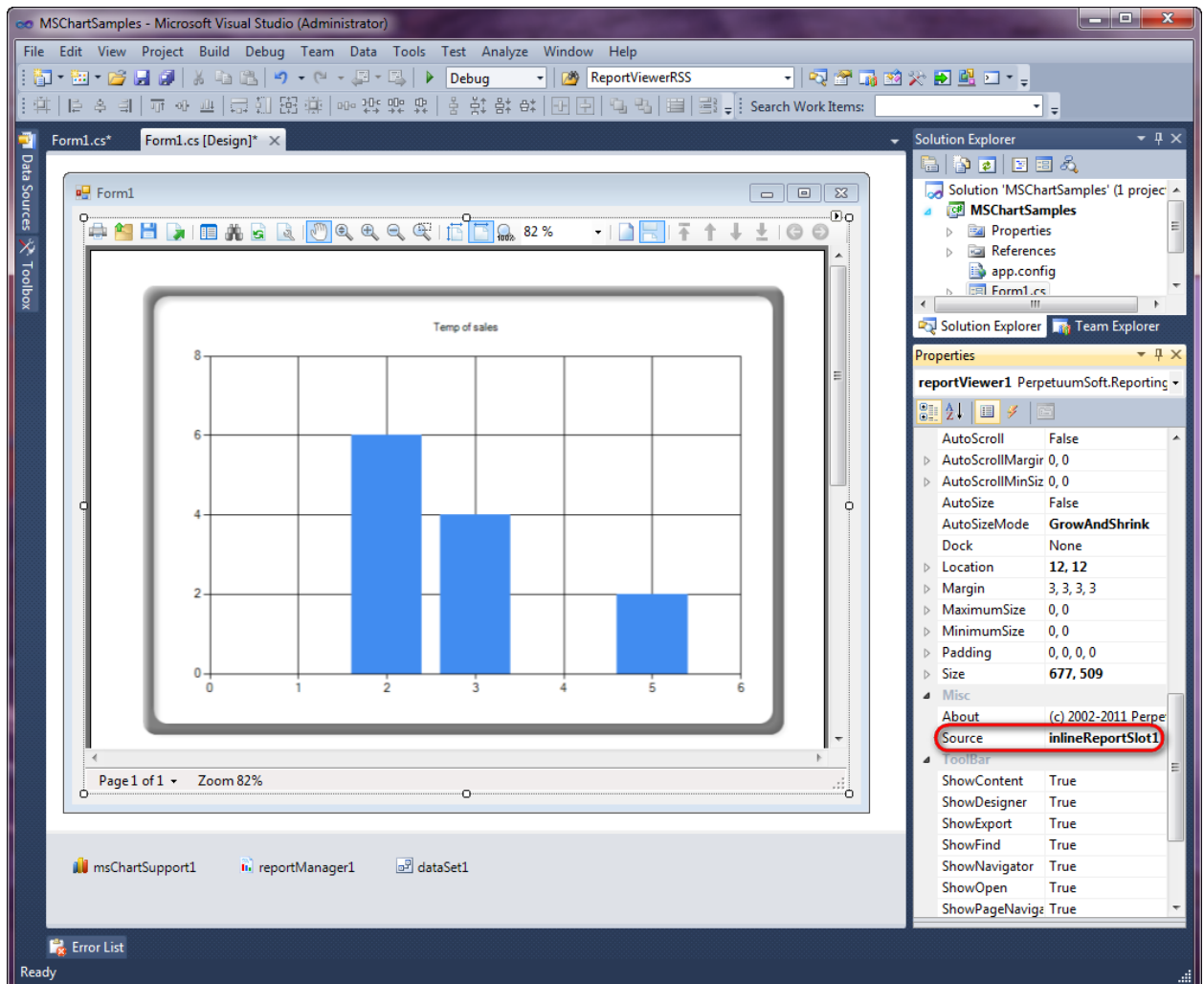
Save the template and close the Report Designer.

Step 20

Add ReportViewer onto the form.



Set the Source to inlineReportSlot1.



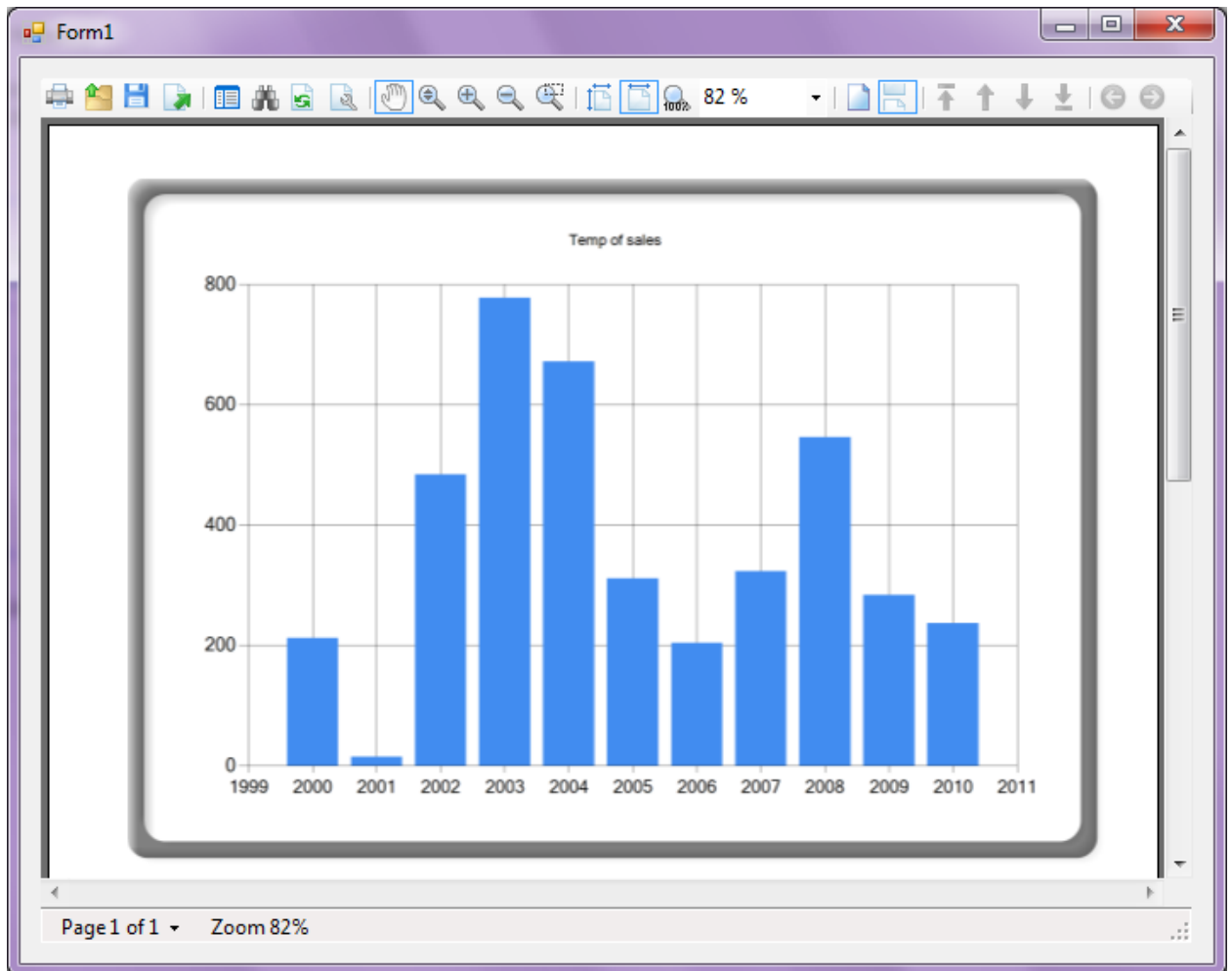
Step 21

In order to display the report on the startup, add the form's Load event handler:

```
private void Form1_Load(object sender, EventArgs e)
{
    inlineReportSlot1.Prepare();
}
```

Step 22

Run the application.



Configuring charts in Report Sharp-Shooter scripts.

There is some information to know in order to successfully work with Microsoft Charts using Report Sharp-Shooter scripts:

The implementation of Charts in Report Sharp-Shooter isn't our own implementation. It uses standard charts in the System.Windows.Forms.DataVisualization.Charting.Chart library. A "wrapper" for all used classes is implemented for support integration. It allows the user to use Charts inside Report Sharp-Shooter. Most of the wrapper-classes are inherited from the standard classes and they add interfaces to Report Sharp-Shooter. Therefore there are some requirements in order to successfully use the Charts.

Use the PerpetuumSoft.Reporting.MSChart.ChartModel namespace with the same type names instead of the System.Windows.Forms.DataVisualization.Charting namespace.



Try to avoid the use of functions used to add objects into collections that automatically create an object. In this case the object of a standard type will be created instead of the wrapped object.

It is not recommended to setup a chart using scripts if there is a possibility to do this without scripts.

An example of configuring a chart using scripts.

In order to configure the chart using scripts as described above, use the following code. For example, add the second chart to your report template and in the GenerateScript of the chart you will have:

```
microsoftChart2.DataSource = "tempOfSales";

PerpetuumSoft.Reporting.MSChart.ChartModel.ChartArea chartArea = new
PerpetuumSoft.Reporting.MSChart.ChartModel.ChartArea();
chartArea.Name = "ChartArea1";
microsoftChart2.ChartAreas.Add(chartArea);

PerpetuumSoft.Reporting.MSChart.ChartModel.Series series = new
PerpetuumSoft.Reporting.MSChart.ChartModel.Series();
series.Name = "One more series";
series.XValueMember = "Year";
series.YValueMembers = "Value";
microsoftChart2.Series.Add(series);

PerpetuumSoft.Reporting.MSChart.ChartModel.Title title = new
PerpetuumSoft.Reporting.MSChart.ChartModel.Title();
title.Text = "Temp of sales";
microsoftChart2.Titles.Add(title);

microsoftChart2.BorderSkin.SkinStyle =
System.Windows.Forms.DataVisualization.Charting.BorderSkinStyle.FrameThin1;
```