Report Sharp-Shooter Getting Started

Last modified on: May 6, 2011

Preface

This user guide contains instructions on how to create templates of the common reports using Report Sharp-Shooter.

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

System Requirements

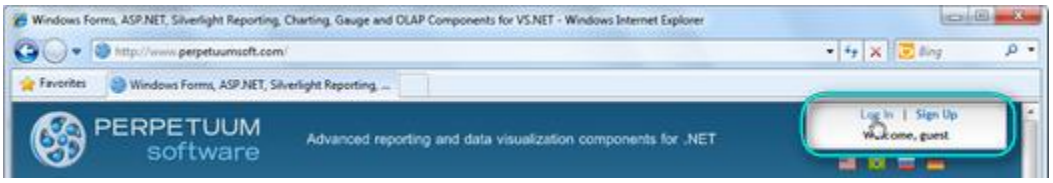
In order to use .NET ModelKit Suite successfully in WinForms applications you will need:

- .NET Framework 2.0, 3.5 or 4.0
- Visual Studio 2005/2008/2010

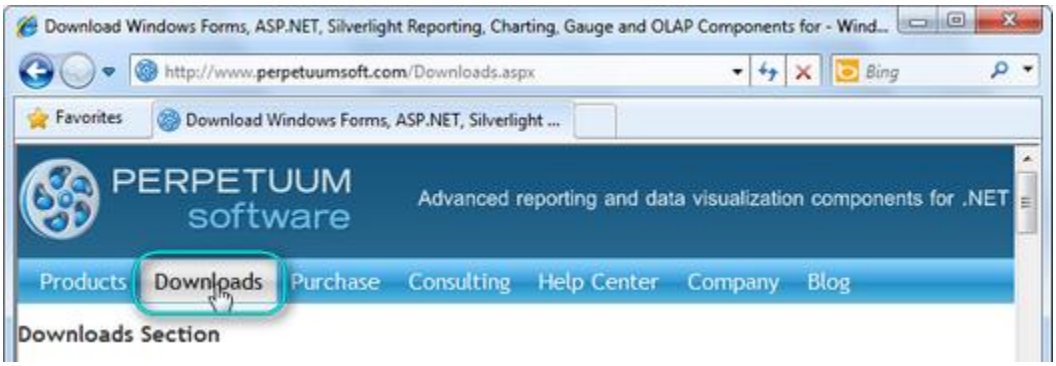
Downloading

You can download the product on the page:
http://www.perpetuumsoft.com/Downloads.aspx

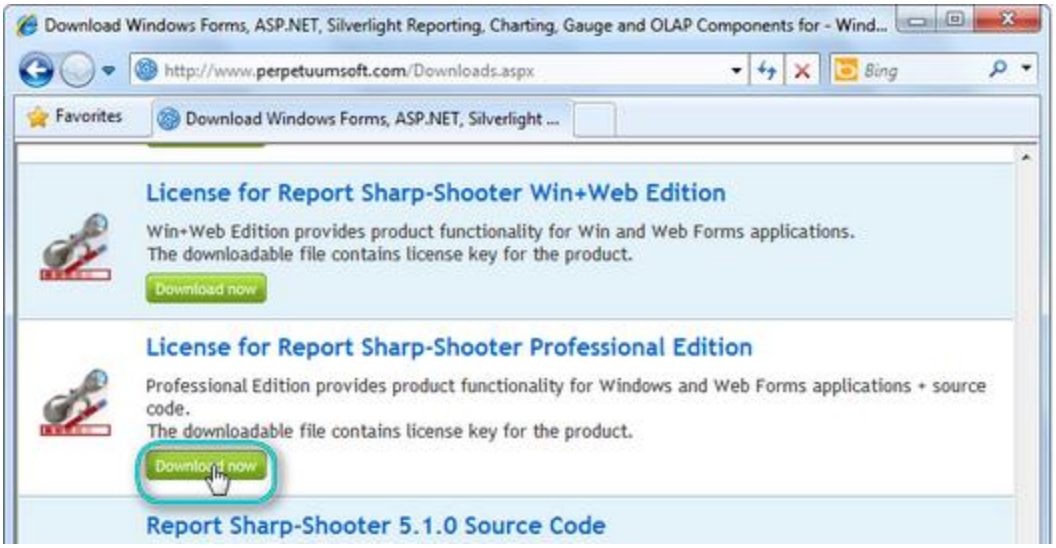
If you purchased the license, then to download the license key, you should enter our site under your account.



Then proceed to the "Downloads" section.



Find the license key for the product in the products' list.



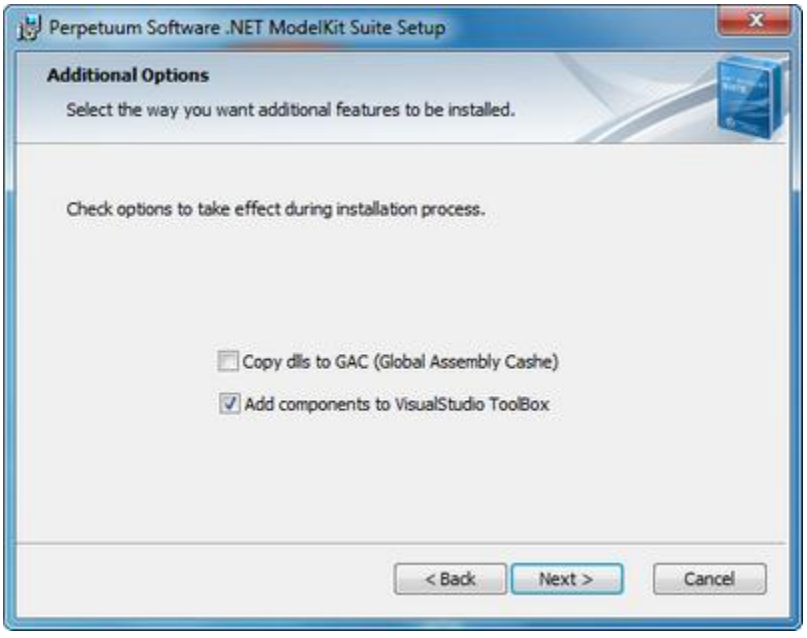
Installation

After you downloaded the license key, run the NETModelKitSuite.msi file. To install it you need administrative rights.

At the "Additional Options" stage select the following options:

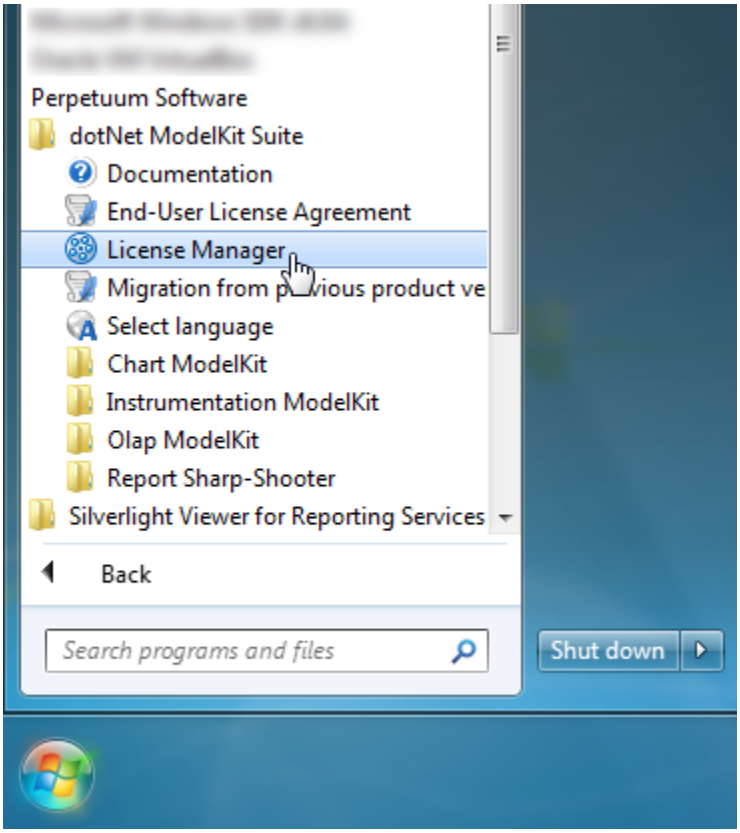
- Copy dlls to GAC;

- Add components to Visual Studio Toolbox. It is possible to change the list of displayed elements on Toolbox after you install the component into Visual Studio.

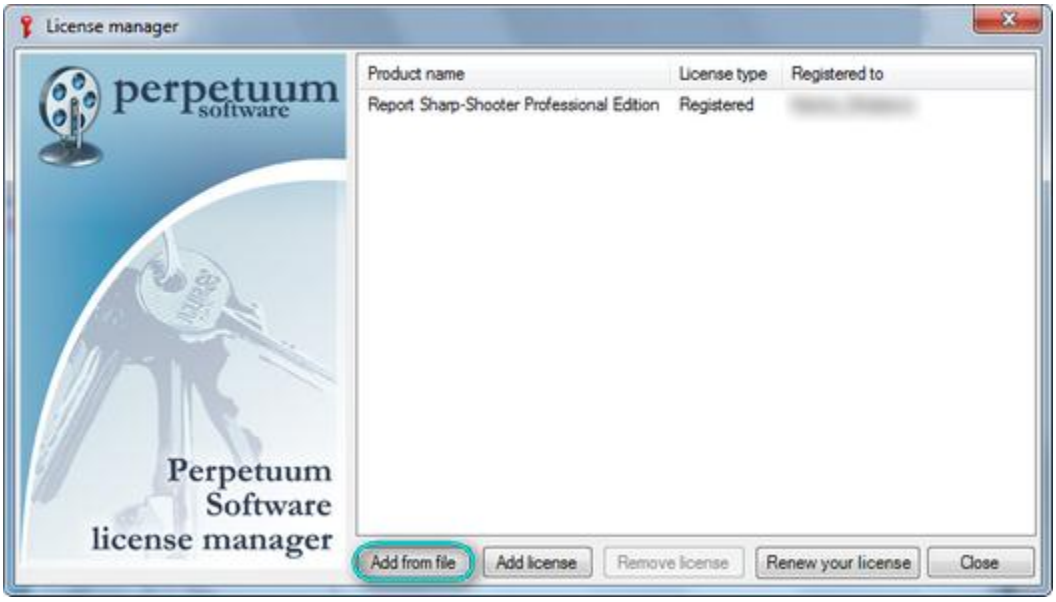


Licensing

If you have the license, launch the License Manager from the Start menu to install the license.

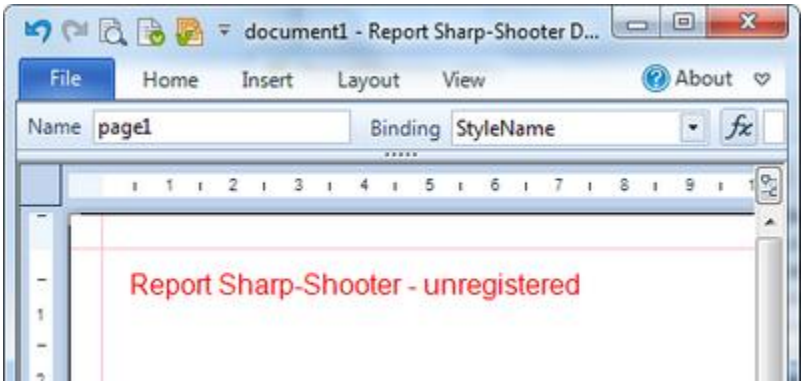


To add the license press the "Add from file" button, and select the *. elic which you downloaded. Press the "Close" button to close License Manager.



If you don't have the license, then there will appear "Trial" pop-ups during the work with the Perpetuum Software components.

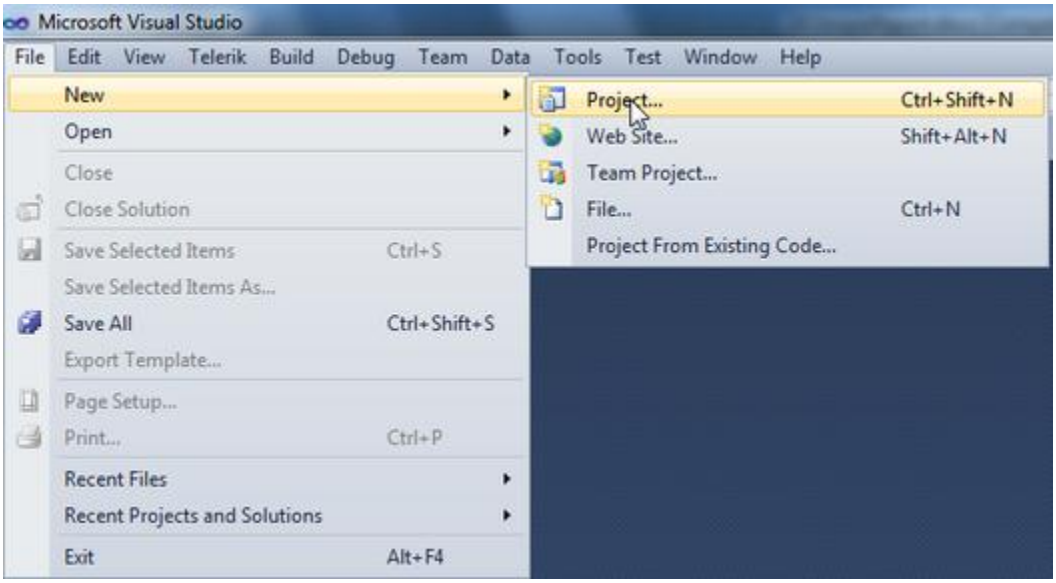
There will be shown watermark during the work in designer.



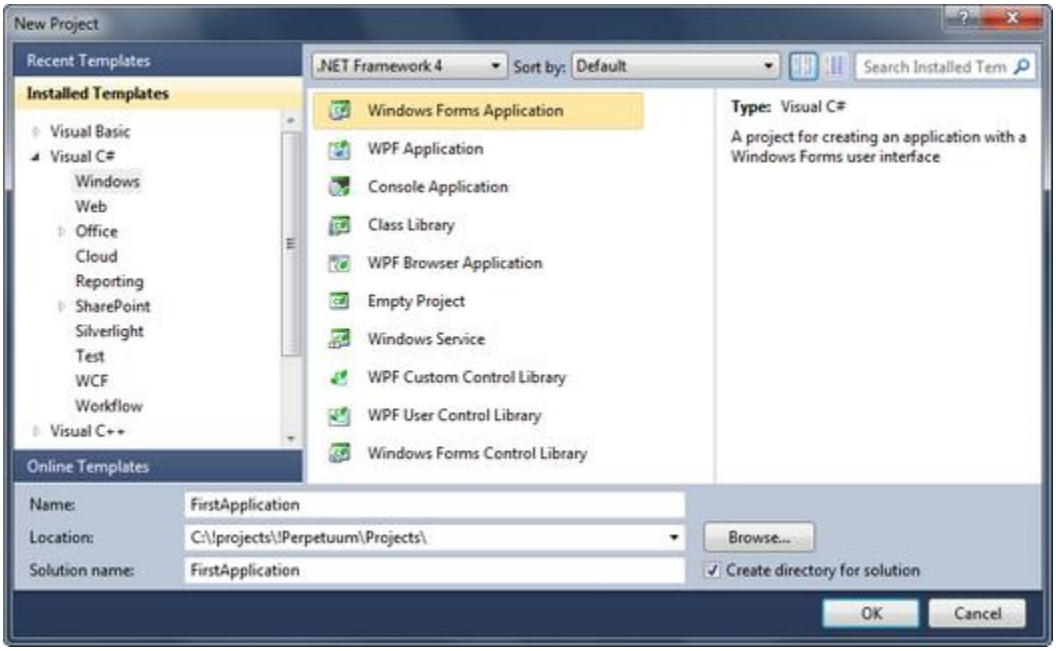
Developing

Step 1. Applications Creation

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

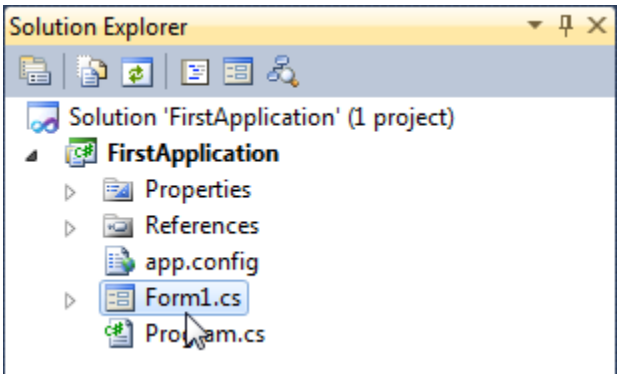


Select "Windows Forms Application", set a name of the project, set the directory to save the project to and click the "OK" button.

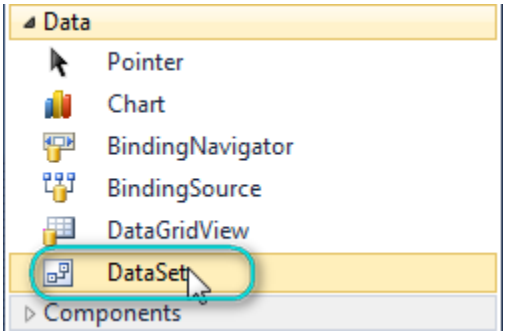


Step 2. Creation and Filling a Data Source

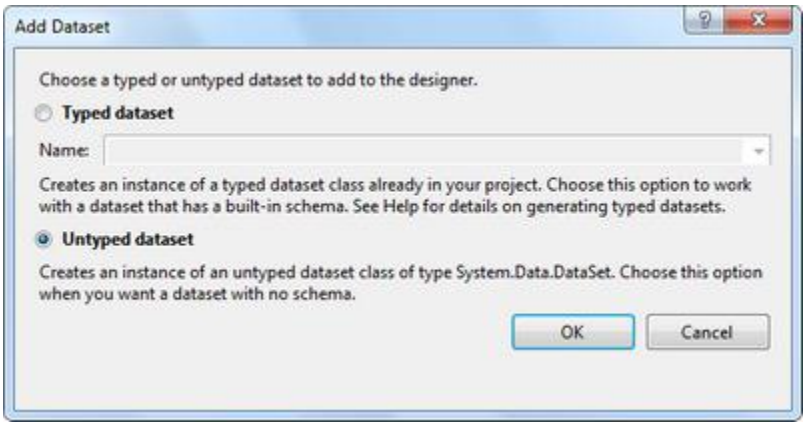
Open the main form of the application in the editor by double click on the "Form1.cs" in the Solution Explorer.



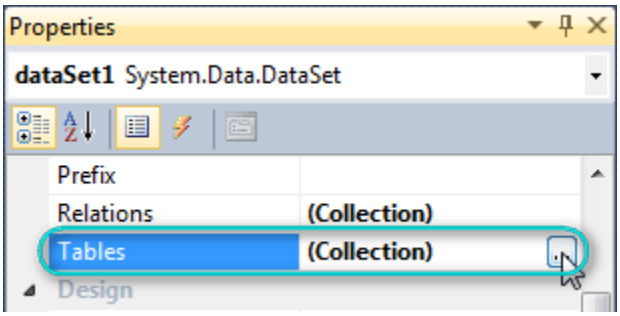
Use the DataSet element to set data structure. Add the element from Toolbox by double clicking.



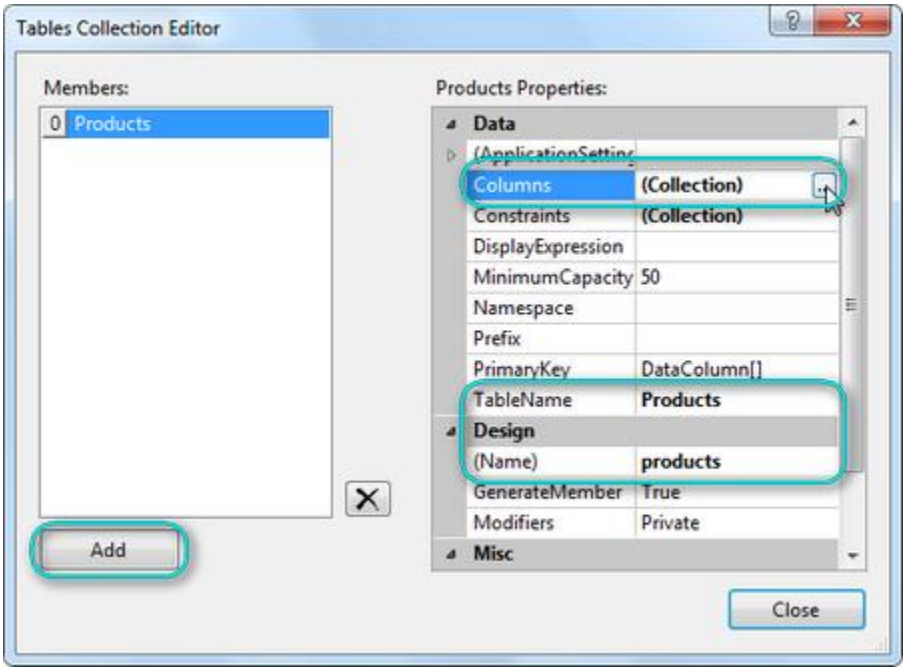
Select "Untyped dataset" and click the "OK" button.



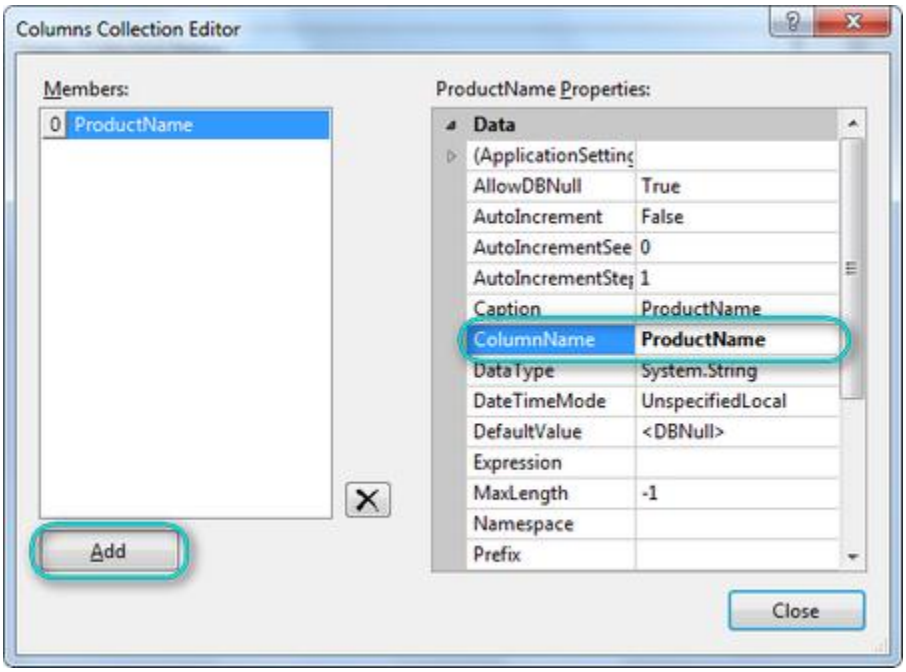
Select the dataSet1 component in the form editor. Select the *Tables* property on the Property Grid and click the button in order to open Tables Collection Editor.



Click the "Add" button in order to add a table. Set the *TableName* property and *Name* to "Products".

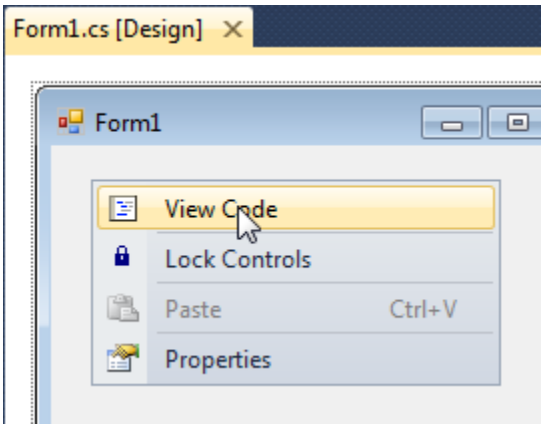


Select the *Columns* property and click the button in order to open Columns Collection Editor. Click the "Add" button to add a new column. Set the *ColumnName* property to "ProductName".



Close editors.

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



To fill data source write the LoadData function.

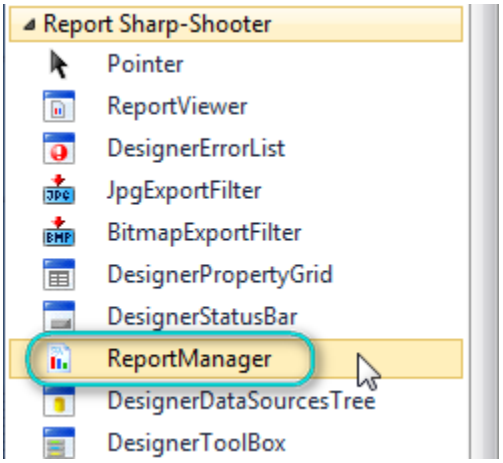
```
public void LoadData()
{
    DataRow row = products.NewRow();
    row["ProductName"] = "Report Sharp-Shooter";
    products.Rows.Add(row);
    row = products.NewRow();
    row["ProductName"] = "Report Sharp-Shooter for Silverlight";
    products.Rows.Add(row);
    row = products.NewRow();
```

```
            row["ProductName"] = "OLAP ModelKit";
            products.Rows.Add(row);
            row = products.NewRow();
            row["ProductName"] = "Instrumentation ModelKit";
            products.Rows.Add(row);
            row = products.NewRow();
            row["ProductName"] = "Chart ModelKit";
            products.Rows.Add(row);
        }
```

Step 3. Template Creation

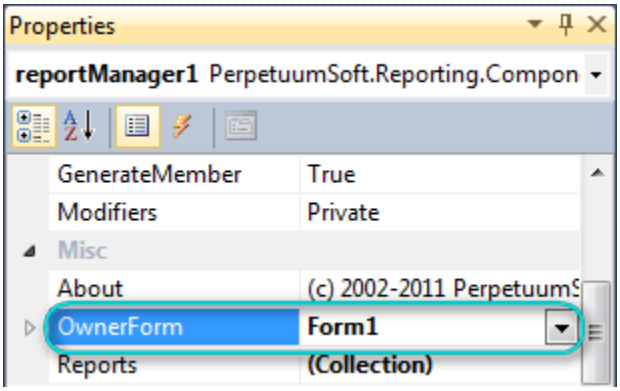
Get back to the application form.

Add the ReportManager element from Toolbox.



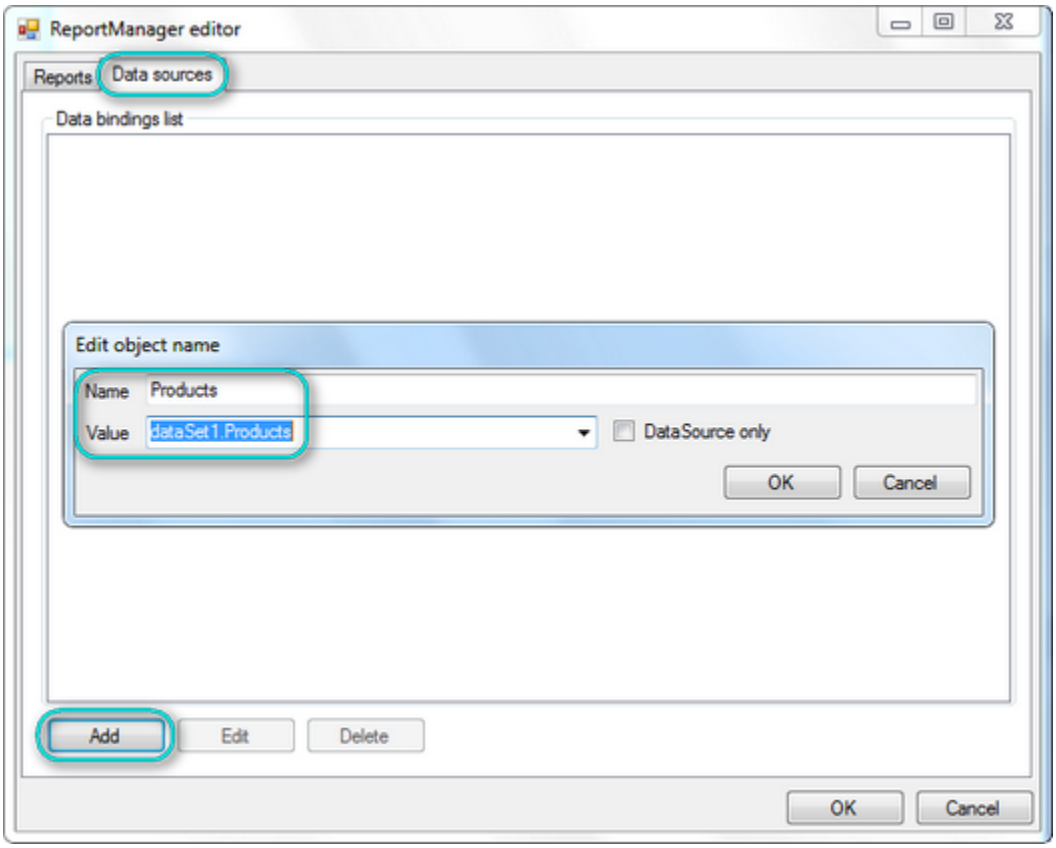
The component is displayed in the bottom of the window.

Set the *OwnerForm* property of ReportManager to "Form1" – the form on which it is located.

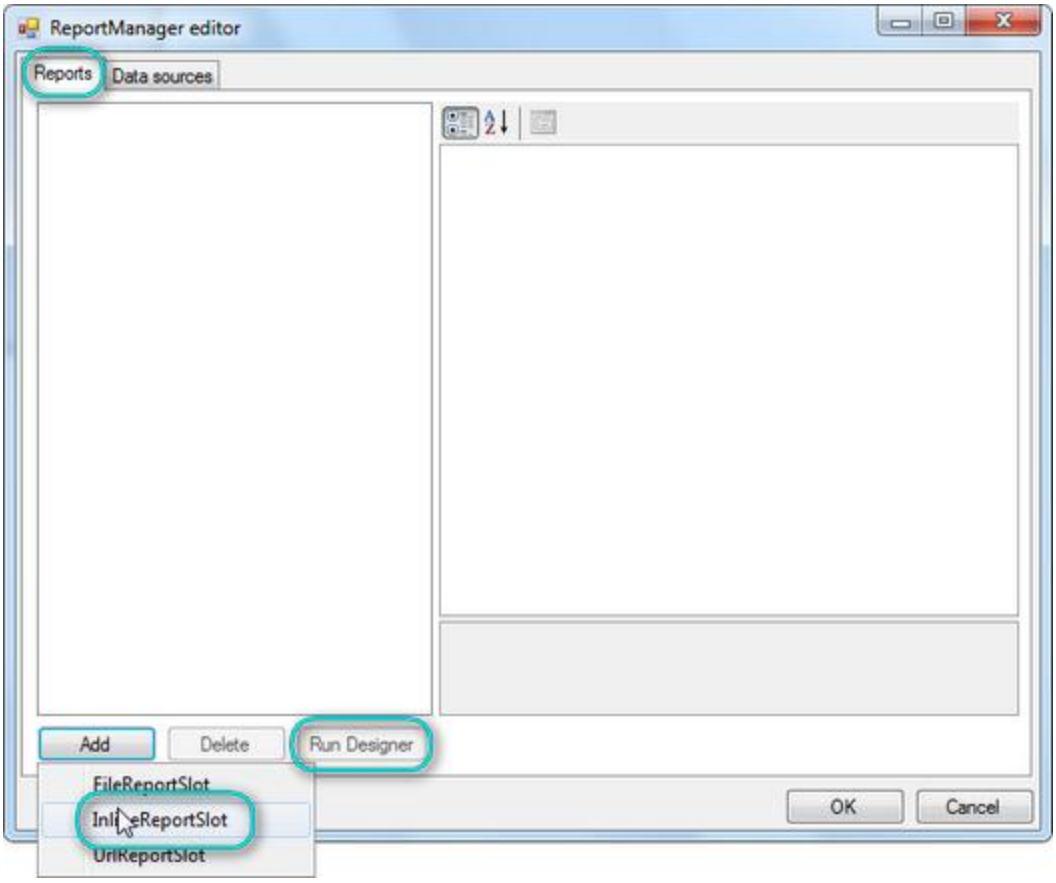


Double click on the ReportManager component to open ReportManager editor.

Go to the “Data sources” tab and click the “Add” button. Set the name of the data source to “Products”, select “dataSet1.Products” data source value and click the “OK” button.

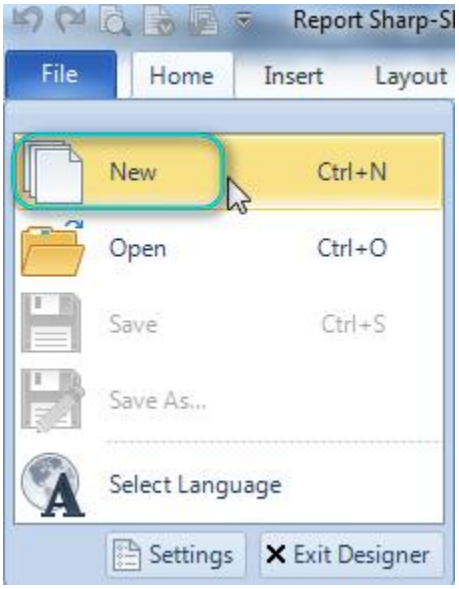


Go to the “Reports” tab. Click the “Add” button and select the “InlineReportSlot” item.

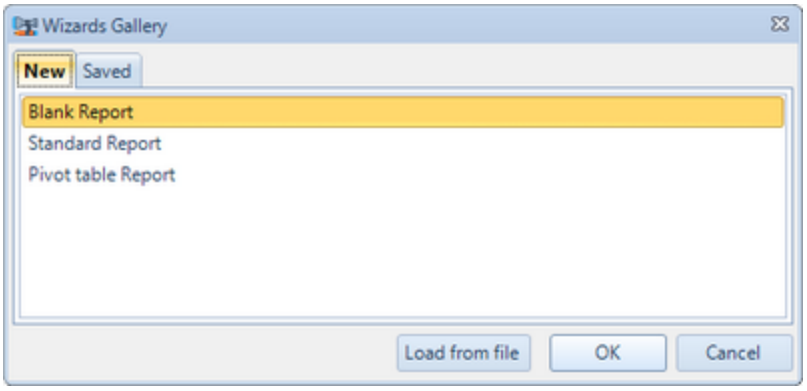


Click the “Run Designer” button to create a template.

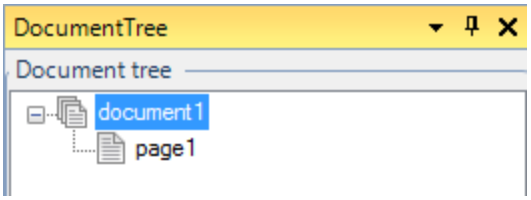
Create a new empty template – select the “New” item in the Application menu.



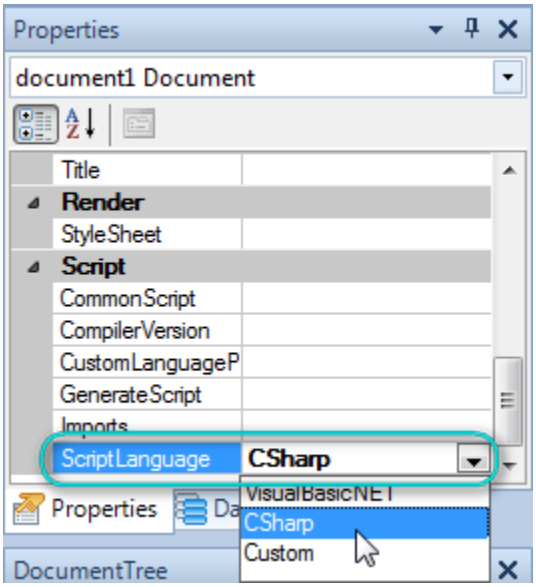
Select the "Blank Report" item in the Wizards Gallery and click the "OK" button.



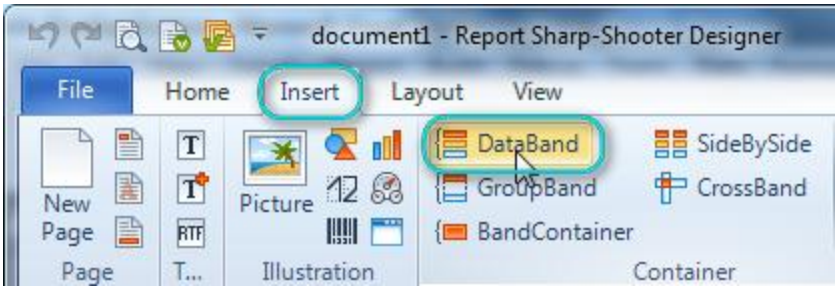
To view template properties select "document1" in the Document Tree or click the button on Quick Access Toolbar.



Set the *ScriptLanguage* property to "CSharp".

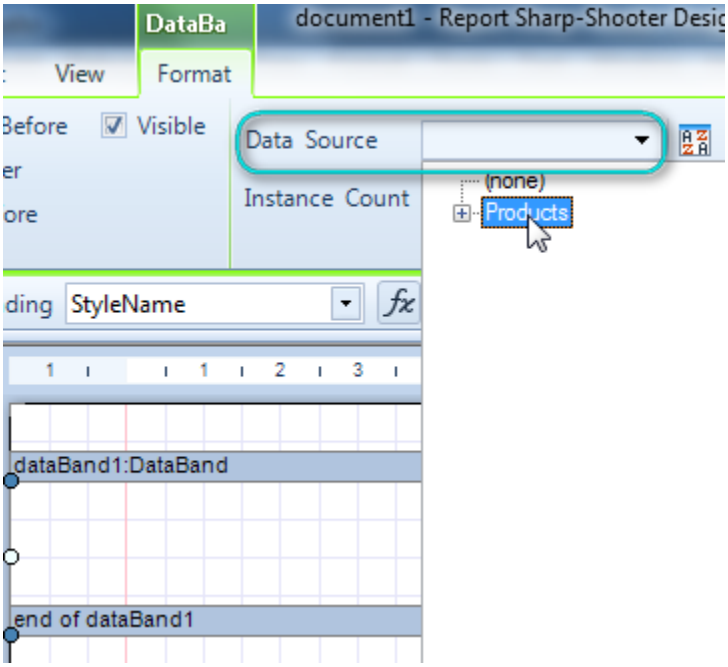


Go to the "Insert" tab and click the "DataBand" button.

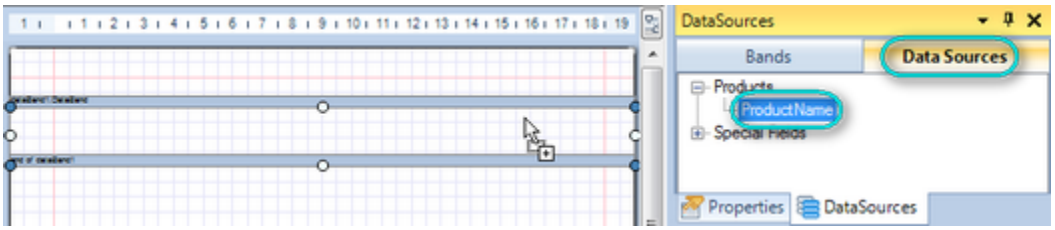


Click on the template area to add DataBand to the template.

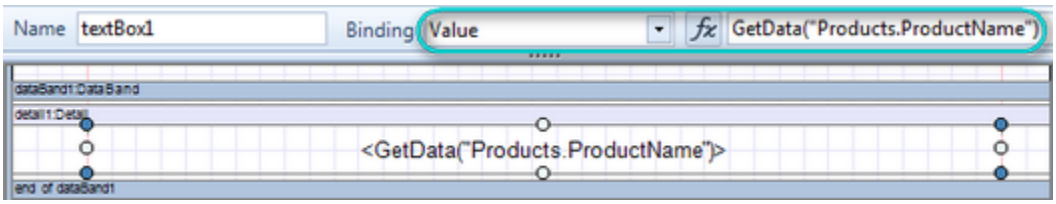
Go to the DataBand "Format" contextual tab. Set the *Data Source* property to "Products".



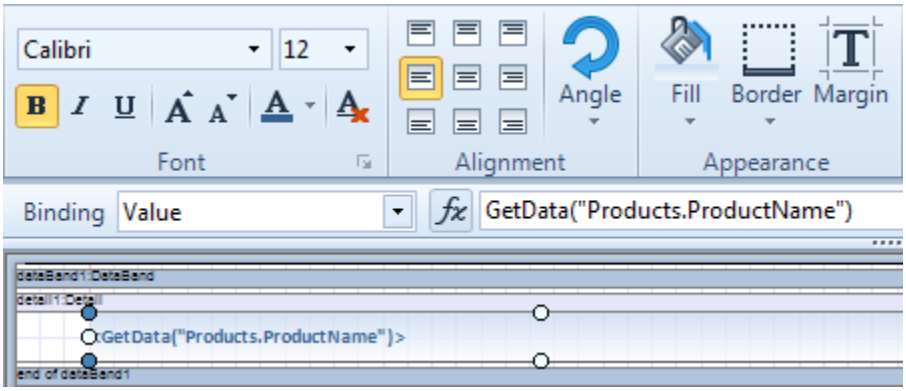
Go to the Data Sources section of the DataSources window. Drag-and-drop ProductName field and put it onto the DataBand.



The detail section and the TextBox element are created automatically. The TextBox's *Value* binding property contains script for data loading.



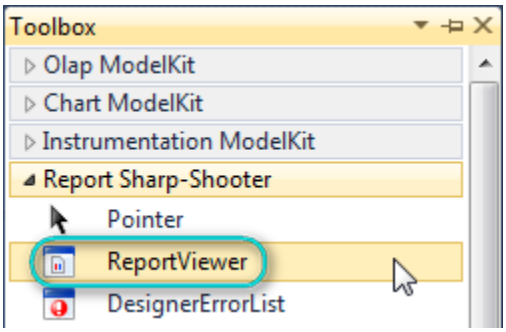
Select the TextBox element and set appearance properties using controls on the "Home" tab.



Save the template and close Report Designer. Click "OK" to close ReportManager Editor.

Step 4. Addition of Report Viewer

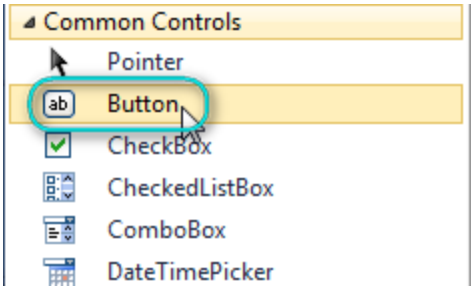
In order to display the generated report, add the ReportViewer element from Toolbpox.



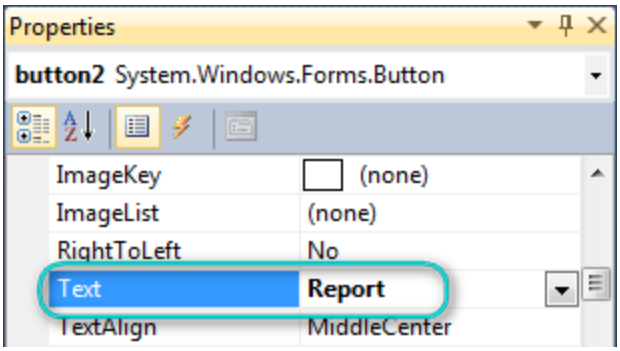
Add code to the class constructor: invocation of the data loading function, the report generation function and initialization of the *Source* property of the viewer.

```
public Form1()
{
    InitializeComponent();
    loadData();
    inlineReportSlot1.RenderDocument();
    reportViewer1.Source = inlineReportSlot1;
}
```

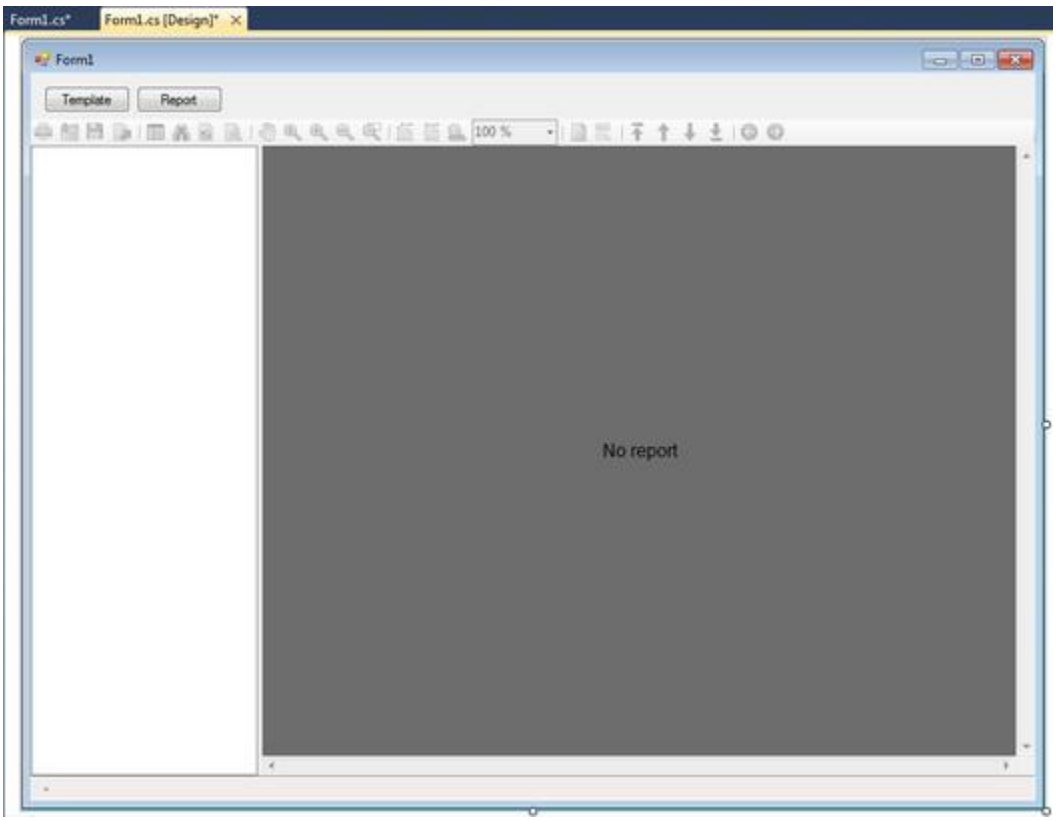
Place two buttons onto the form (drag and drop the Button element from the Toolbox).



Select Button on the form; edit the *Text* property on the Property Grid. Set the property to "Template" for one button and to "Report" for the second one.



Change the elements' location on the form.

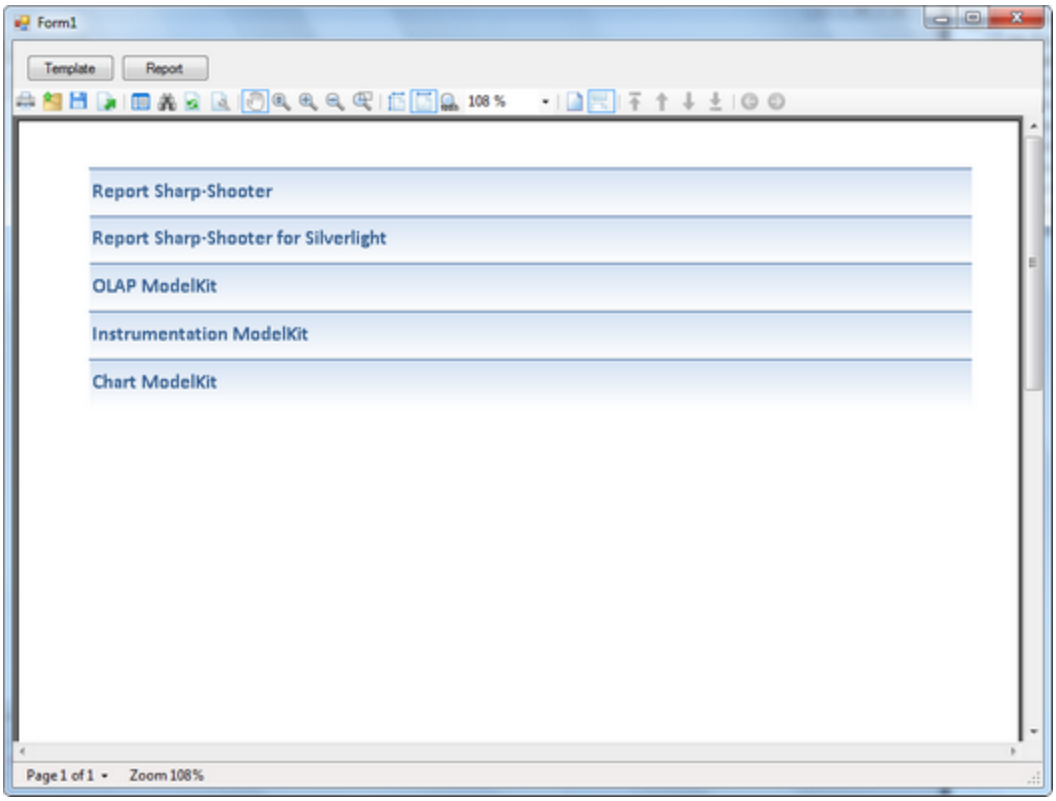


Create the Click event handlers for the buttons – double click on the Button. Add code which launches report generation to the event handler. Use the following code:

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

private void button2_Click(object sender, EventArgs e)
{
    inlineReportSlot1.RenderDocument();
    reportViewer1.Source = inlineReportSlot1;
}
```

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



Step 5. Editing of the license file

Make sure that the license data was included into the project. To do that, open the license.licx file; it must contain a list of the licensed components used in the application.

The data about the ReportManager and ReportViewer components was added to the file automatically when components were added to application.

The created application offers to use End User Designer to create a template (the "Template" button invokes the `DesignTemplate()` function). That's why you need to add the data about the ReportDesigner component to the licenses.licx file. This can be done with two methods:

1. Add the ReportDesigner element from Toolbox to the form. The string will be added to the file automatically. Then the ReportDesigner can be removed from the form.
2. Write the data about the component manually.

```
PerpetuumSoft.Reporting.Designer.ReportDesigner, PerpetuumSoft.Reporting, Version=<
Version>, Culture=neutral, PublicKeyToken=<Key Token>
```