Silverlight Viewer for Reporting Services Getting Started (Remote Connection)

Generation of a simple web-application in the Microsoft Visual Studio 2008 with the use of Silverlight Viewer for Reporting Services

Prerequisites

- .NET Framework 3.5 SP1
- Silverlight v3
- Silverlight v3 SDK
- Silverlight v3 Developer Tool
- Visual Studio 2008
- Microsoft SQL Reporting Services Sample Reports

Attachment

SilverlightViewerForReportingServicesGettingStartedSample.zip

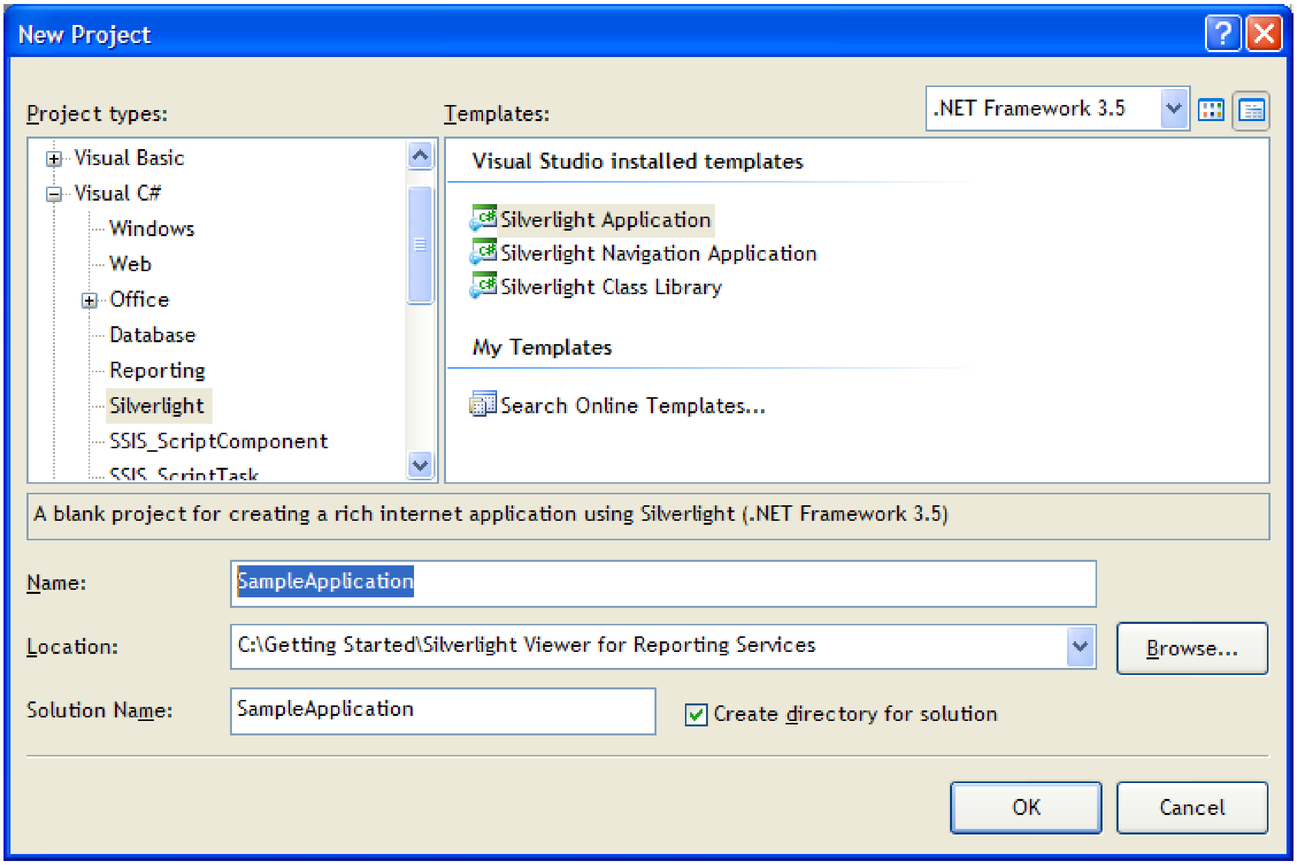
Introduction

The target of the following guide is to demonstrate the basic moments of the use of Silverlight Viewer for Reporting Services. It gives minimum necessary knowledge in order to start working with the component. We will examine the process of generation of web application with the use of Silverlight Viewer for Reporting Services step by step. We will consider creation and configuration of the service and at last integration of the report viewer component on the application pages.

Creating Web application

Step 1.

Create a new Silverlight Application project named “SampleApplication”.



New Project

Project types:

- Visual Basic
- Visual C#
  - Windows
  - Web
  - Office
  - Database
  - Reporting
  - Silverlight
  - SSIS_ScriptComponent
  - SSIS_ScriptTask

Templates: .NET Framework 3.5

Visual Studio installed templates

- Silverlight Application
- Silverlight Navigation Application
- Silverlight Class Library

My Templates

- Search Online Templates...

A blank project for creating a rich internet application using Silverlight (.NET Framework 3.5)

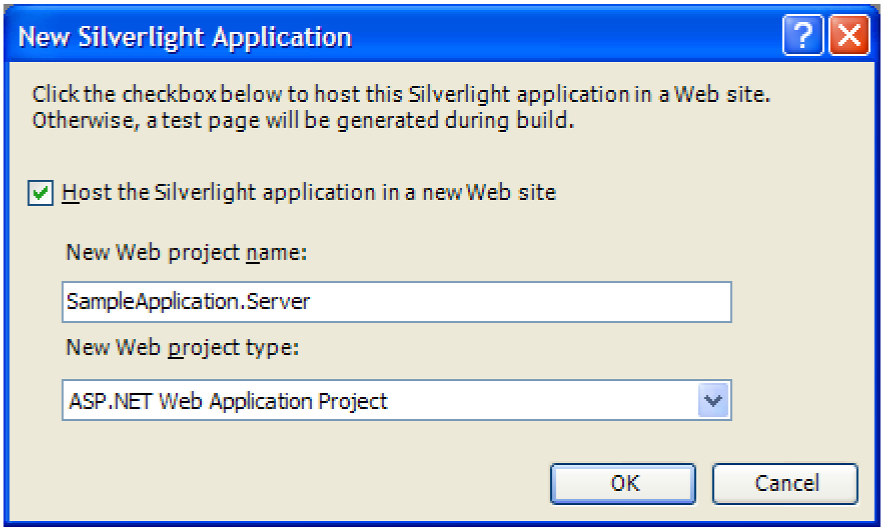
Name: SampleApplication

Location: C:\Getting Started\Silverlight Viewer for Reporting Services

Solution Name: SampleApplication

☑ Create directory for solution

While creation of the project select the menu item “Add a new ASP.NET Web project to the solution to host Silverlight” and set name of the web-project to SampleApplication.Server



New Silverlight Application

Click the checkbox below to host this Silverlight application in a Web site. Otherwise, a test page will be generated during build.

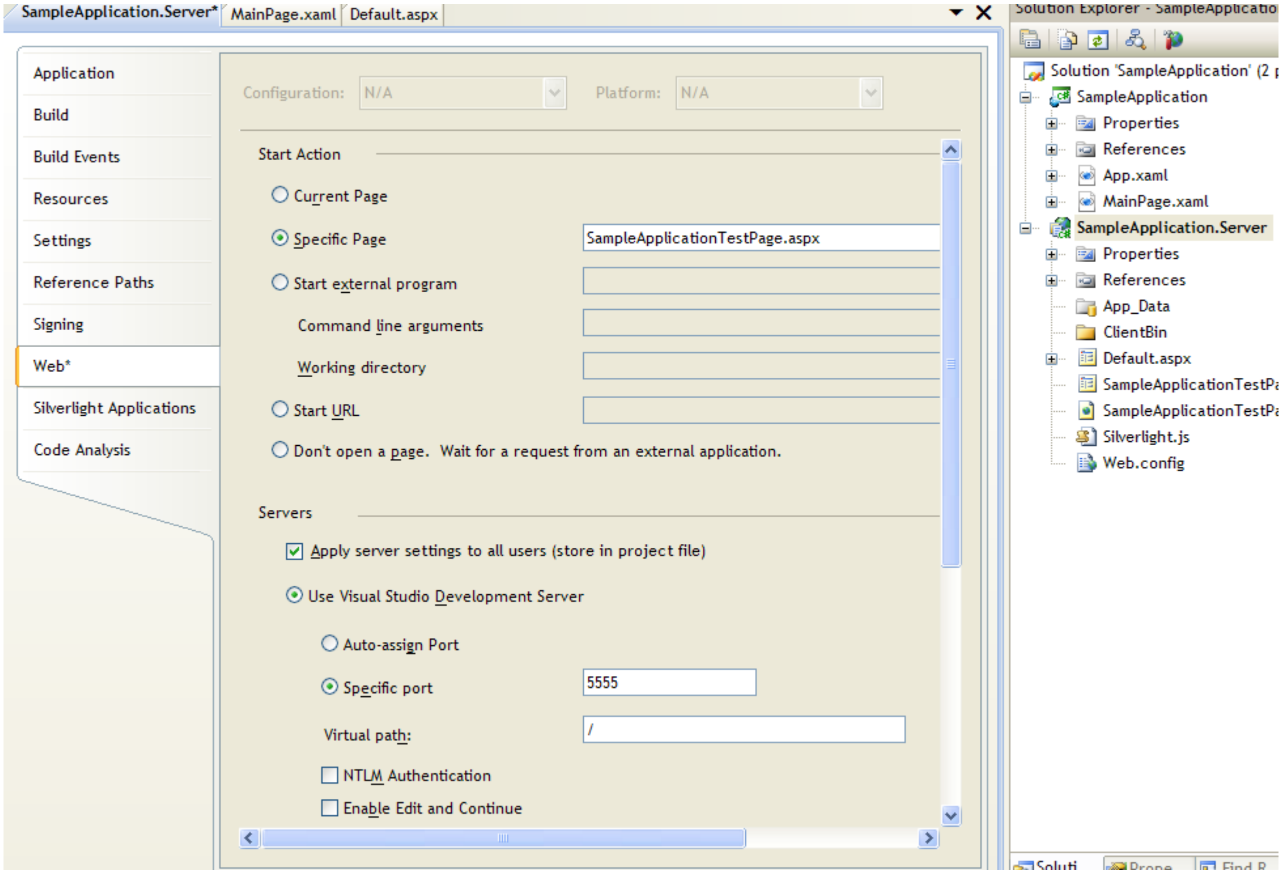
☑ Host the Silverlight application in a new Web site

New Web project name: SampleApplication.Server

New Web project type: ASP.NET Web Application Project

Step 2.

Set specific port to 5555 in the Web tab of the SampleApplication.Server properties (in the "Solution Explorer", item Properties in contextual menu of the SampleApplication.Server).



Configuration: N/A Platform: N/A

Start Action

○ Current Page

● Specific Page SampleApplicationTestPage.aspx

○ Start external program

Command line arguments

Working directory

○ Start URL

○ Don't open a page. Wait for a request from an external application.

Servers

☑ Apply server settings to all users (store in project file)

● Use Visual Studio Development Server

○ Auto-assign Port

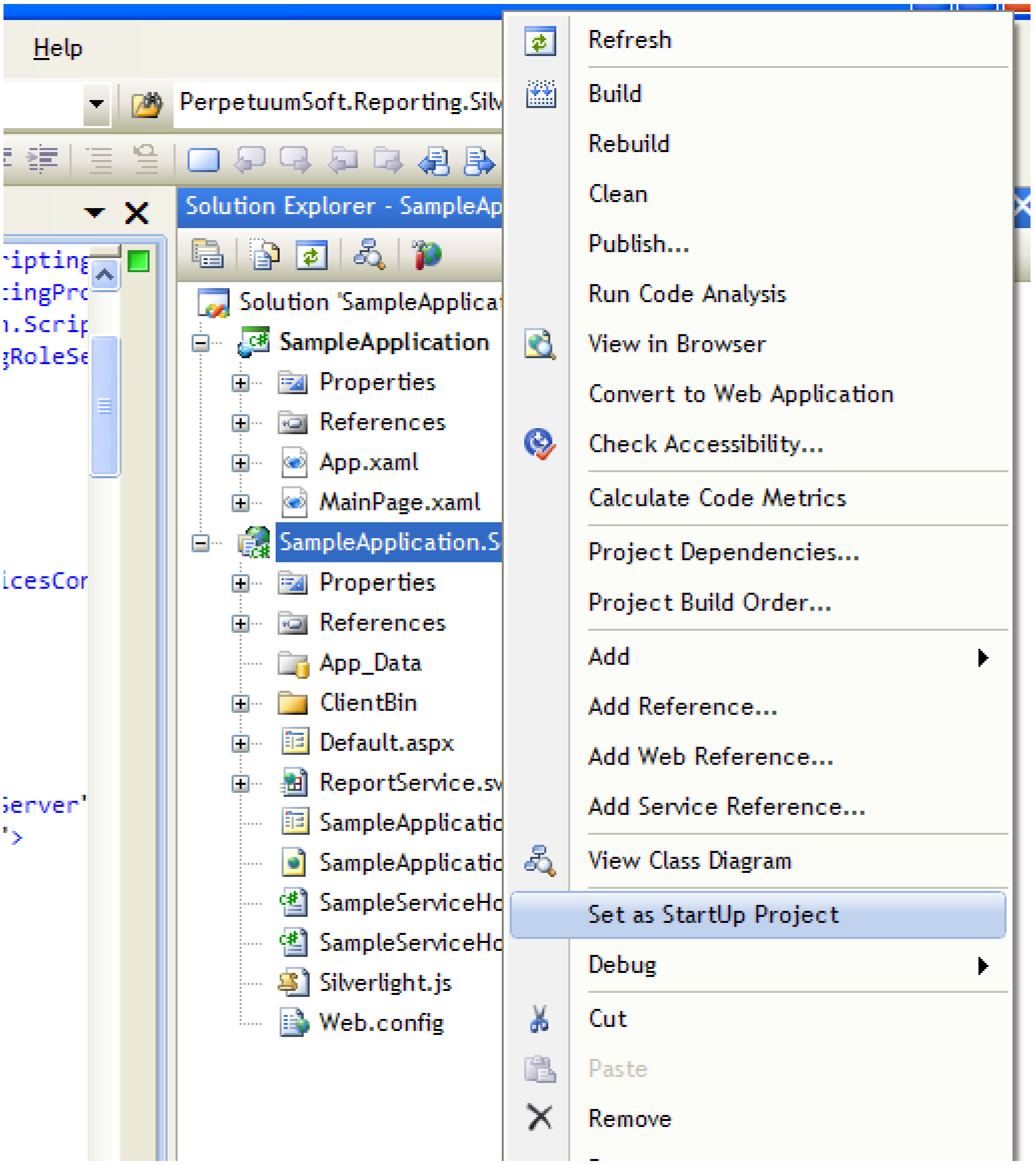
● Specific port 5555

Virtual path: /

□ NTLM Authentication

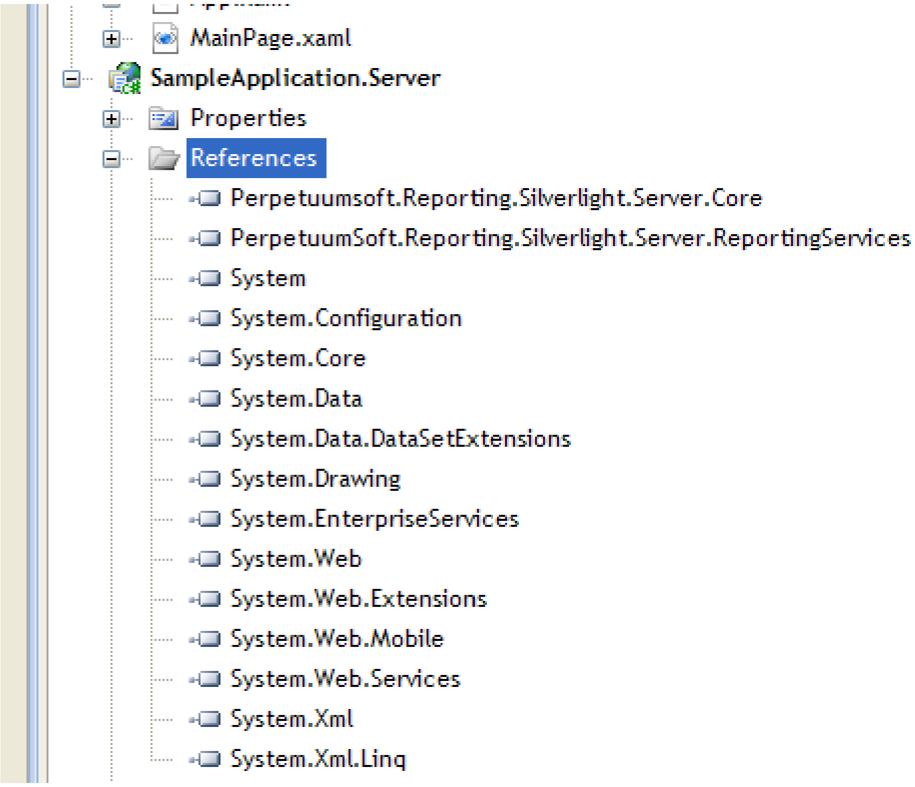
□ Enable Edit and Continue

Next, set SampleApplication.Server project as Startup Project



Step 3.

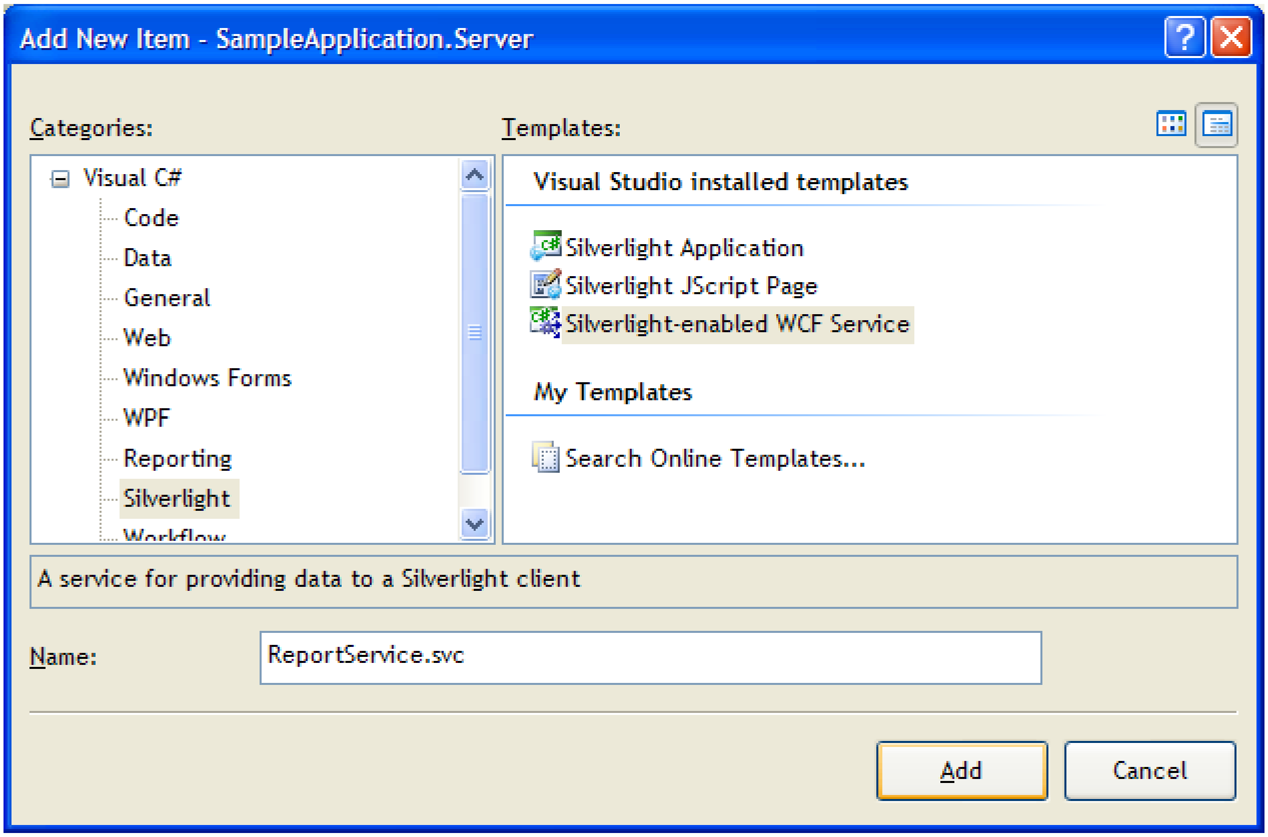
Add Perpetuumsoft.Reporting.Silverlight.Server.Core and PerpetuumSoft.Reporting.Silverlight.Server.ReportingServices assemblies to the project references, (in "Solution Explorer", item Add Reference in context menu of SampleApplication.Server). These assemblies are located in the Bin folder of the Silverlight Viewer for Reporting Services installation folder.



```
MainPage.xaml
SampleApplication.Server
    Properties
    References
        Perpetuumsoft.Reporting.Silverlight.Server.Core
        PerpetuumSoft.Reporting.Silverlight.Server.ReportingServices
        System
        System.Configuration
        System.Core
        System.Data
        System.Data.DataSetExtensions
        System.Drawing
        System.EnterpriseServices
        System.Web
        System.Web.Extensions
        System.Web.Mobile
        System.Web.Services
        System.Xml
        System.Xml.Linq
```

Step 4.

Add WCF service named ReportService.svc to the SampleApplication.Server project.



Add New Item - SampleApplication.Server

Categories: Visual C# — Silverlight

Templates: Silverlight-enabled WCF Service

A service for providing data to a Silverlight client

Name: ReportService.svc

Inherit a new ReportService class from the MsReportServiceBase class for implementation and opportunity to change standard behavior. In order to make ASP.Net context available for the service, service class needs to be marked by a special AspNetCompatibilityRequirements attribute. The ASP.Net context is required for the service to cache document data, if other mechanism of data storage is not implemented.

```
// ReportService.svc.cs
using System.ServiceModel.Activation;
using PerpetuumSoft.Reporting.Silverlight.Server.ReportingServices;

namespace SampleApplication.Server
{
    [AspNetCompatibilityRequirements(RequirementsMode                    =
AspNetCompatibilityRequirementsMode.Required)]
    public class ReportService : MsReportServiceRemote
    {
    }
}
```

Step 5.

Now it’s necessary to setup created web-service. Make changes in the web.config file. System.serviceModel section must be as follows:

```
<system.serviceModel>

  <behaviors>
    <endpointBehaviors>
      <behavior name="webBehavior">
        <webHttp/>
      </behavior>
    </endpointBehaviors>
    <serviceBehaviors>
      <behavior name="SampleApplication.Server.ReportServiceBehavior">
        <serviceMetadata httpGetEnabled="true"/>
        <serviceDebug includeExceptionDetailInFaults="false"/>
      </behavior>
    </serviceBehaviors>
  </behaviors>
  <serviceHostingEnvironment aspNetCompatibilityEnabled="true"/>
  <services>
    <service
      behaviorConfiguration="SampleApplication.Server.ReportServiceBehavior"
      name="SampleApplication.Server.ReportService">
      <endpoint
        address=""
         binding="basicHttpBinding"
        bindingConfiguration="basicHttpBindingConf"

      contract="PerpetuumSoft.Reporting.Silverlight.Server.Core.IReportService">
        <identity>
          <dns value="localhost"/>
        </identity>
      </endpoint>
      <endpoint
            address="rest"
            behaviorConfiguration="webBehavior"
            binding="webHttpBinding"
            bindingConfiguration="webHttpBindingConf"
            contract="PerpetuumSoft.Reporting.Silverlight.Server.Core.IReportSer
            viceResources"/>
     <endpoint
            address="mex"
            binding="mexHttpBinding"
            contract="IMetadataExchange"/>
    </service>
   </services>
   <bindings>
      <webHttpBinding>
        <binding name="webHttpBindingConf"/>
      </webHttpBinding>
      <basicHttpBinding>
        <binding name="basicHttpBindingConf"/>
      </basicHttpBinding>
   </bindings>
</system.serviceModel>
```

The first endpoint is intended for work with the report, for instance, to get pages. The second endpoint is intended for work with the resources, for instance, to get localization. And, at last, the third endpoint is used to get meta-information about the service.

The Report Service can work over SSL. To do it you need extended service definition in the web.config file. Add some definition into binding element.

```
<security mode="Transport">
   <transport clientCredentialType="None"/>
</security>
```

NOTE: If you set Security Mode equal to Transport you have to set ServiceUrl of ReportViewer to https://...

Example:

```
<webHttpBinding>
   <binding name="webHttpBindingConf">
      <security mode="Transport">
         <transport clientCredentialType="None"/>
      </security>
    </binding>
</webHttpBinding>
   <basicHttpBinding>
      <binding name="basicHttpBindingConf">
         <security mode="Transport">
           <transport clientCredentialType="None"/>
         </security>
      </binding>
</basicHttpBinding>
```

Step 6.

Now it's necessary to setup Microsoft SQL Reporting Services web-service. Make changes in the web.config file.

Add section element inside the configSections element.

```
<section
name="PerpetuumSoftServices"
type="PerpetuumSoft.Reporting.Silverlight.Server.ReportingServices.ServicesConfi
gurationSection, PerpetuumSoft.Reporting.Silverlight.Server.ReportingServices"
       allowLocation="true"
       allowDefinition="Everywhere"/>
```

Add PerpetuumSoftServices below the configSection element in the web.config file.

```
<PerpetuumSoftServices>
    <Service
Type="SampleApplication.Server.ReportService, SampleApplication.Server">
      <MsReportingServer
Url="http://localhost/reportserver/ReportExecution2005.asmx">
        <Credentials Domain="" UserName="" Password=""/>
      </MsReportingServer>
    </Service>
  </PerpetuumSoftServices>
```

MsReportingServer attributes and elements description:

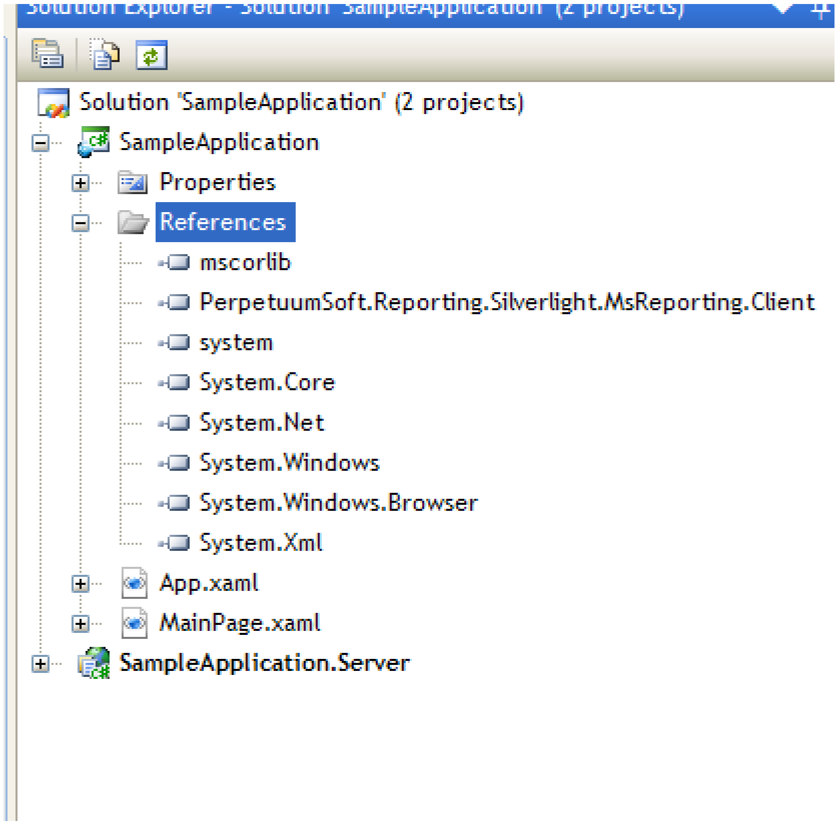
Url attribute specifies path to your Microsoft SQL Reporting Services 2005 web service.

Credentials element defines credentials to access to the Reporting Service web service.

If you omit the Credentials element or leave attributes blank, the server will use Default Network Credentials of Web Site.

Step 7.

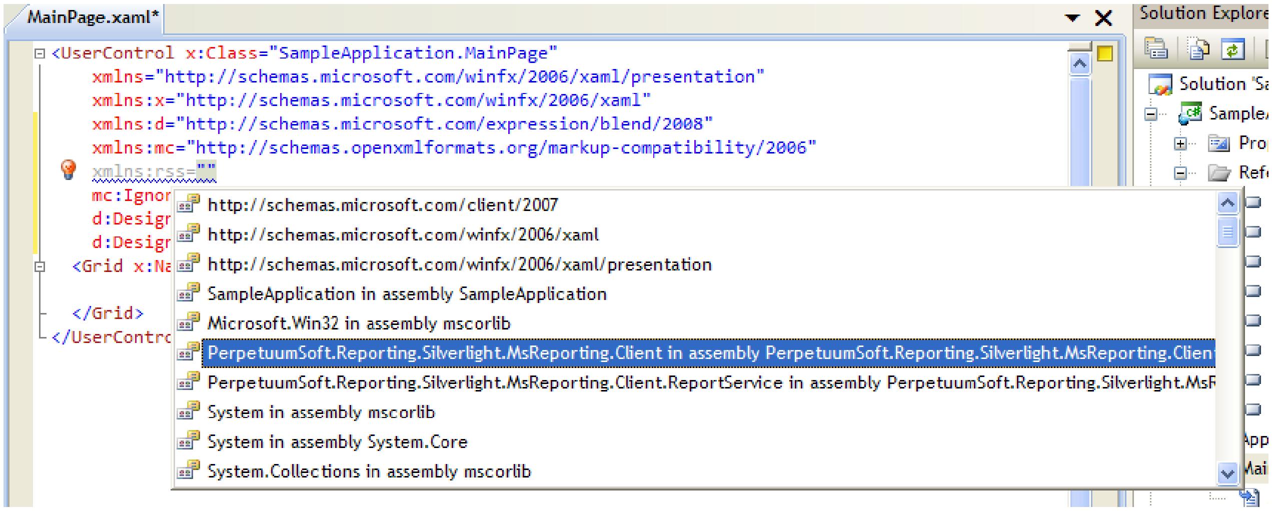
It's necessary to add the report viewer component to the Silverlight application for the report display. Therefore you should add a reference to the PerpetuumSoft.Reporting.Silverlight.MsReporting.Client assembly, containing ReportViewer (in "Solution Explorer", item Add Reference in context menu of SampleApplication). This assemble is located in the Bin folder of the Silverlight Viewer for Reporting Services installation folder.



Solution Explorer - Solution 'SampleApplication' (2 projects)

Solution 'SampleApplication' (2 projects)
SampleApplication
Properties
References
mscorlib
PerpetuumSoft.Reporting.Silverlight.MsReporting.Client
system
System.Core
System.Net
System.Windows
System.Windows.Browser
System.Xml
App.xaml
MainPage.xaml
SampleApplication.Server

Open MainPage.xaml in the markup designer and add xml namespace for the PerpetuumSoft.Reporting.Silverlight.MsReporting.Client assembly.

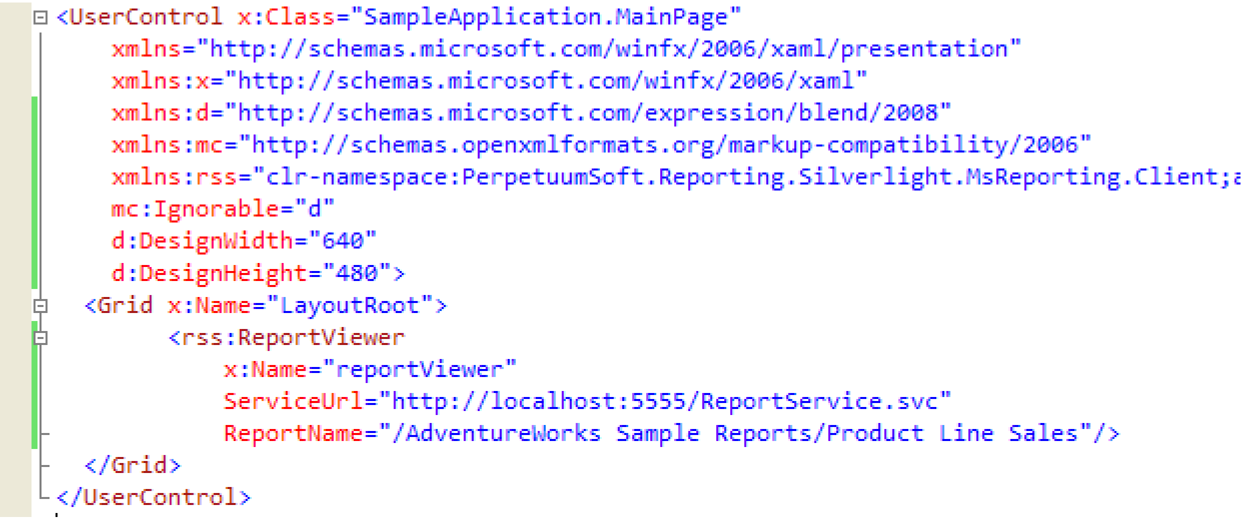


```
<UserControl x:Class="SampleApplication.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:rss=""
    mc:Ignor
    d:Desigr
    d:Desigr
  <Grid x:Na

  </Grid>
</UserContrc
```

- http://schemas.microsoft.com/client/2007
- http://schemas.microsoft.com/winfx/2006/xaml
- http://schemas.microsoft.com/winfx/2006/xaml/presentation
- SampleApplication in assembly SampleApplication
- Microsoft.Win32 in assembly mscorlib
- PerpetuumSoft.Reporting.Silverlight.MsReporting.Client in assembly PerpetuumSoft.Reporting.Silverlight.MsReporting.Clien
- PerpetuumSoft.Reporting.Silverlight.MsReporting.Client.ReportService in assembly PerpetuumSoft.Reporting.Silverlight.MsR
- System in assembly mscorlib
- System in assembly System.Core
- System.Collections in assembly mscorlib

Then add ReportViewer and set ServiceUrl="http://localhost:5555/ReportService.svc" and ReportName="/AdventureWorks Sample Reports/Product Line Sales".

NOTE: We suppose you have Microsoft SQL Reporting Services installed and configured.



```
<UserControl x:Class="SampleApplication.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:rss="clr-namespace:PerpetuumSoft.Reporting.Silverlight.MsReporting.Client;a
    mc:Ignorable="d"
    d:DesignWidth="640"
    d:DesignHeight="480">
  <Grid x:Name="LayoutRoot">
        <rss:ReportViewer
            x:Name="reportViewer"
            ServiceUrl="http://localhost:5555/ReportService.svc"
            ReportName="/AdventureWorks Sample Reports/Product Line Sales"/>
  </Grid>
</UserControl>
```

Open the MainPage.xaml.cs source code and add the following code.

```
public partial class MainPage : UserControl
{
    public MainPage()
    {
        InitializeComponent();
        Loaded += new RoutedEventHandler(Page_Loaded);
    }

    void Page_Loaded(object sender, RoutedEventArgs e)
    {
        reportViewer.ApplyTemplate();
        reportViewer.RenderDocument();
    }
}
```

RenderDocument method invocation leads to the rendering of the current report on the server and its displaying in the Report Viewer.

Step 8.

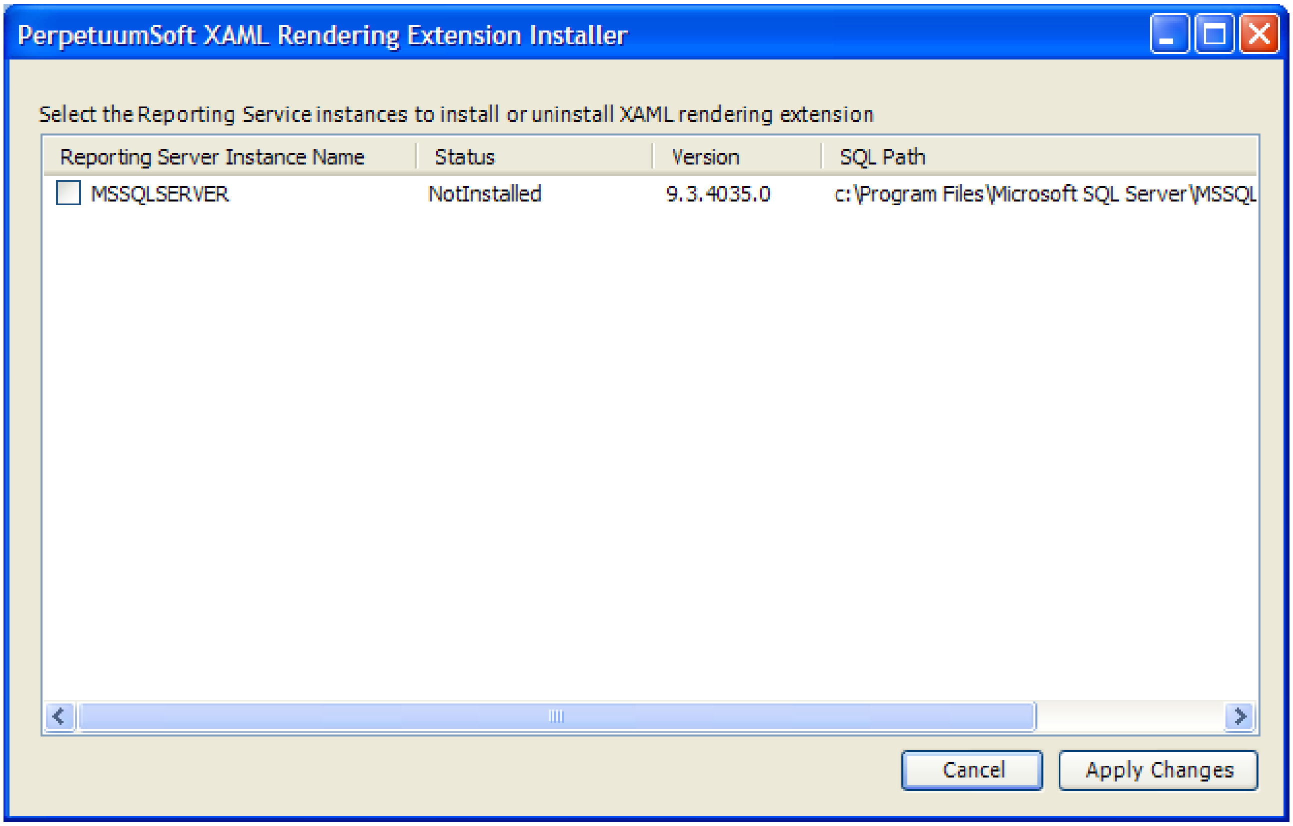
Install Microsoft SQL Reporting Services Xaml Rendering extension.

There are two ways to install rendering extension: using the PerpetuumSoft.XamlExtension.msi file and manually.

NOTE: The extension must be installed ONLY to Microsoft SQL Reporting Services 2005.

The use of installer.

Run PerpetuumSoft.XamlExtension.msi. The installer determines your SQL configurations and offers the list of the available Reporting Services instances.

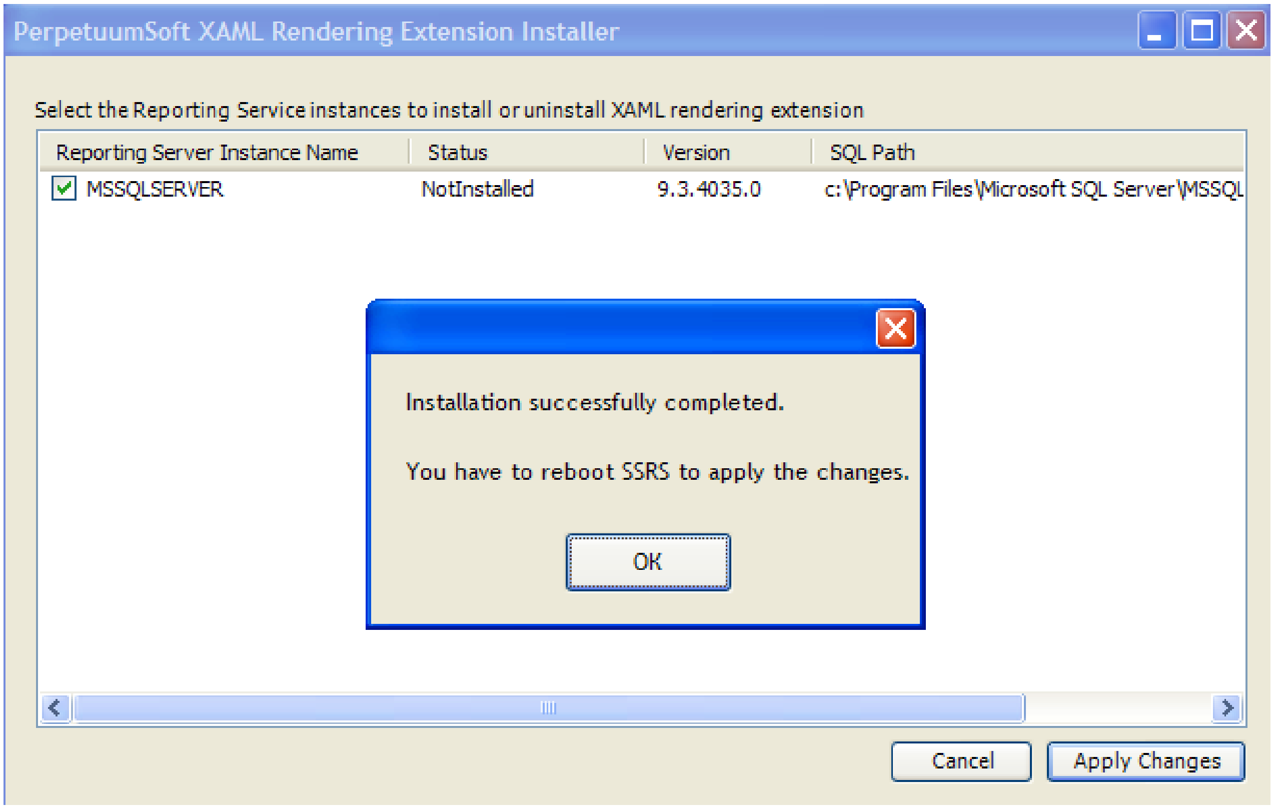


PerpetuumSoft XAML Rendering Extension Installer

Select the Reporting Service instances to install or uninstall XAML rendering extension

Reporting Server Instance Name	Status	Version	SQL Path
□ MSSQLSERVER	NotInstalled	9.3.4035.0	c:\Program Files\Microsoft SQL Server\MSSQL

Check/uncheck the instances to install/uninstall Rendering Extension to selected Reporting Services instances and click “Apply Changes” button.



PerpetuumSoft XAML Rendering Extension Installer

Select the Reporting Service instances to install or uninstall XAML rendering extension

Reporting Server Instance Name	Status	Version	SQL Path
☑ MSSQLSERVER	NotInstalled	9.3.4035.0	c:\Program Files\Microsoft SQL Server\MSSQL

Installation successfully completed.

You have to reboot SSRS to apply the changes.

Manual installation.

First, find location of your Microsoft SQL Reporting Services. Consider the path is C:\Program Files\Microsoft SQL Server\MSSQL.2\Reporting Services\ReportServer

Copy PerpetuumSoft.Reporting.Silverlight.LinkExtension.dll and PerpetuumSoft.ReportingService.XamlRendering.dllassemblies in the C:\Program Files\Microsoft SQL Server\MSSQL.2\Reporting Services\ReportServer\Bin folder. It can be found in the installation folder of the rendering extension.

Open the C:\Program Files\Microsoft SQL Server\MSSQL.2\Reporting Services\ReportServer\ rsreportserver.config file in notepad.

Find <Render> section and insert

<Extension Name="XAML" Type="PerpetuumSoft.ReportingService.XamlRendering.XamlRenderer, PerpetuumSoft.ReportingService.XamlRendering" Visible="false" /> inside it.

Save the file.

Open the C:\Program Files\Microsoft SQL Server\MSSQL.2\Reporting Services\ReportServer\ rssrvpolicy.config file in notepad.

Find <CodeGroup class="FirstMatchCodeGroup" version="1" PermissionSetName="Nothing"> section and insert

<CodeGroup class="UnionCodeGroup" version="1" Name="PerpetuumSoftExtensionCodeGroup" Description="" PermissionSetName="FullTrust">

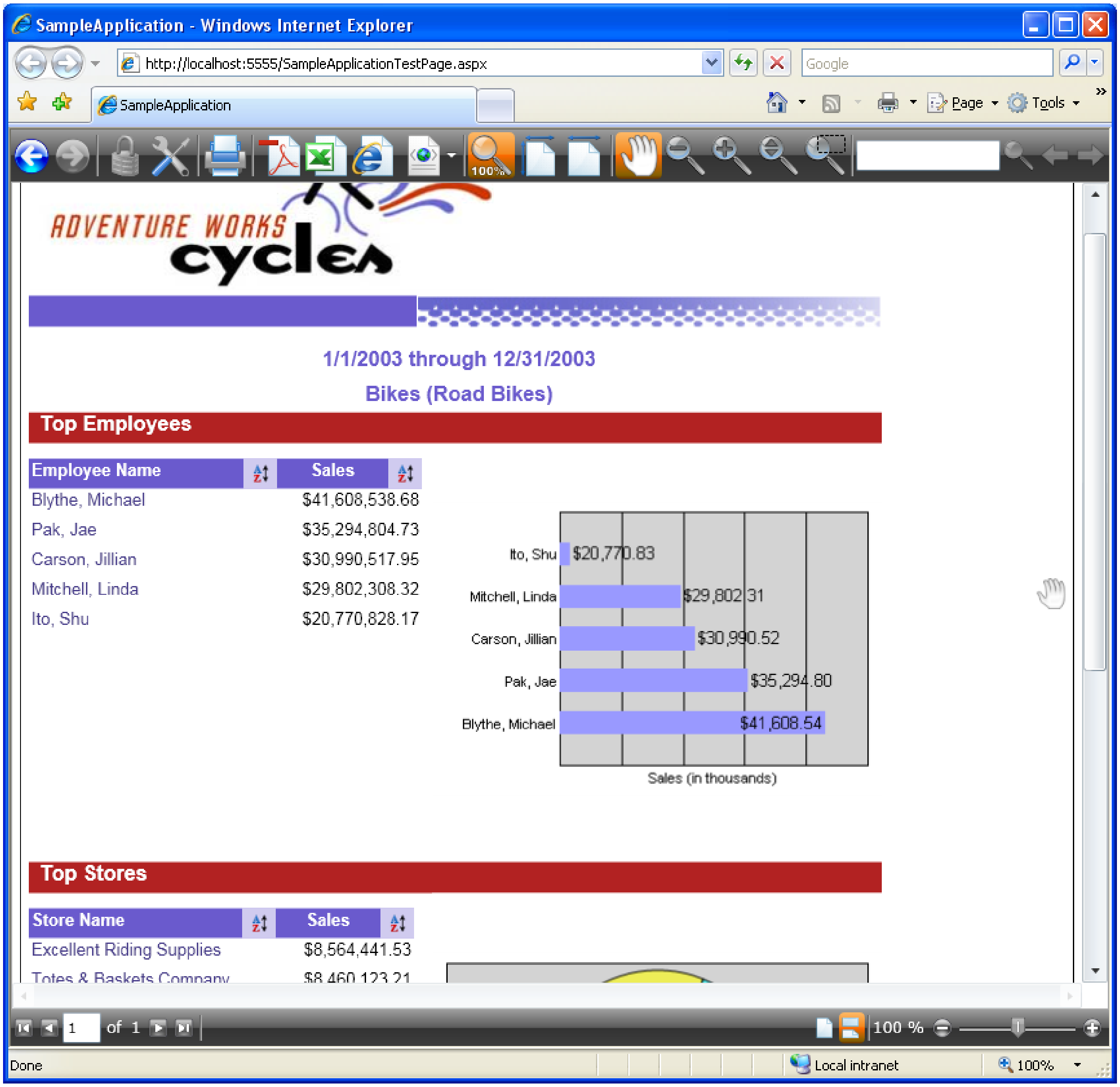
```
                <IMembershipCondition          class="UrlMembershipCondition"          version="1"
Url="c:\Program          Files\Microsoft          SQL          Server\MSSQL.2\Reporting
Services\ReportServer\bin\PerpetuumSoft.ReportingService.XamlRendering.dll" />

              </CodeGroup> inside in.
```

Save the file.

Restart the instance of Reporting Services.

Step 9.

Set the SampleApplication.Server application as a starting one and launch it.



1/1/2003 through 12/31/2003

Bikes (Road Bikes)

Top Employees

Employee Name	Sales
Blythe, Michael	$41,608,538.68
Pak, Jae	$35,294,804.73
Carson, Jillian	$30,990,517.95
Mitchell, Linda	$29,802,308.32
Ito, Shu	$20,770,828.17

Top Stores

Store Name	Sales
Excellent Riding Supplies	$8,564,441.53
Totes & Baskets Company	$8,460,123.21

Conclusion

We have examined basic steps and got a simple and quite operable application. We didn’t have to write thousand lines of code – we only used ready-made implementation. It will be enough in the most cases. If required behavior differs greatly from the one provided by default, you can change not only many aspects of the Silverlight Viewer for Reporting Services work but also the appearance of the report viewer.