WinRT XAML Viewer for Reporting Services 2008/2008 R2/2012
Getting Started

Last modified on: November 12, 2012

Table of Content

Introduction

The target of the following guide is to demonstrate how to build-in WinRT XAML Viewer for Reporting Services. It gives minimum necessary knowledge in order to start working with the component. We will examine the process of creating a server side project with a WCF service as well as a client side Windows Store application. We will consider creation and configuration of the service and at last integration of the report viewer component into the application pages. Your application can be configured to use the processing capability of Microsoft SQL Server 2008/2008 R2/2012 Reporting Services report server.

Product Prerequisites

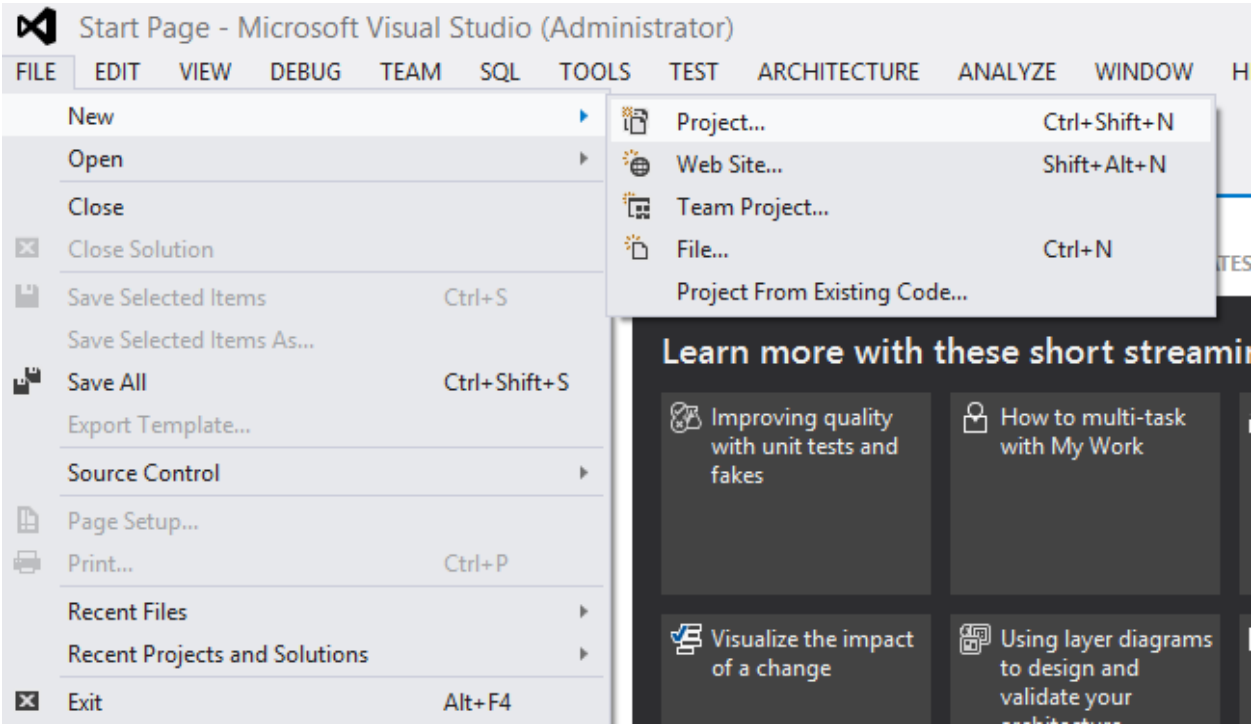
Product works with the following configuration.

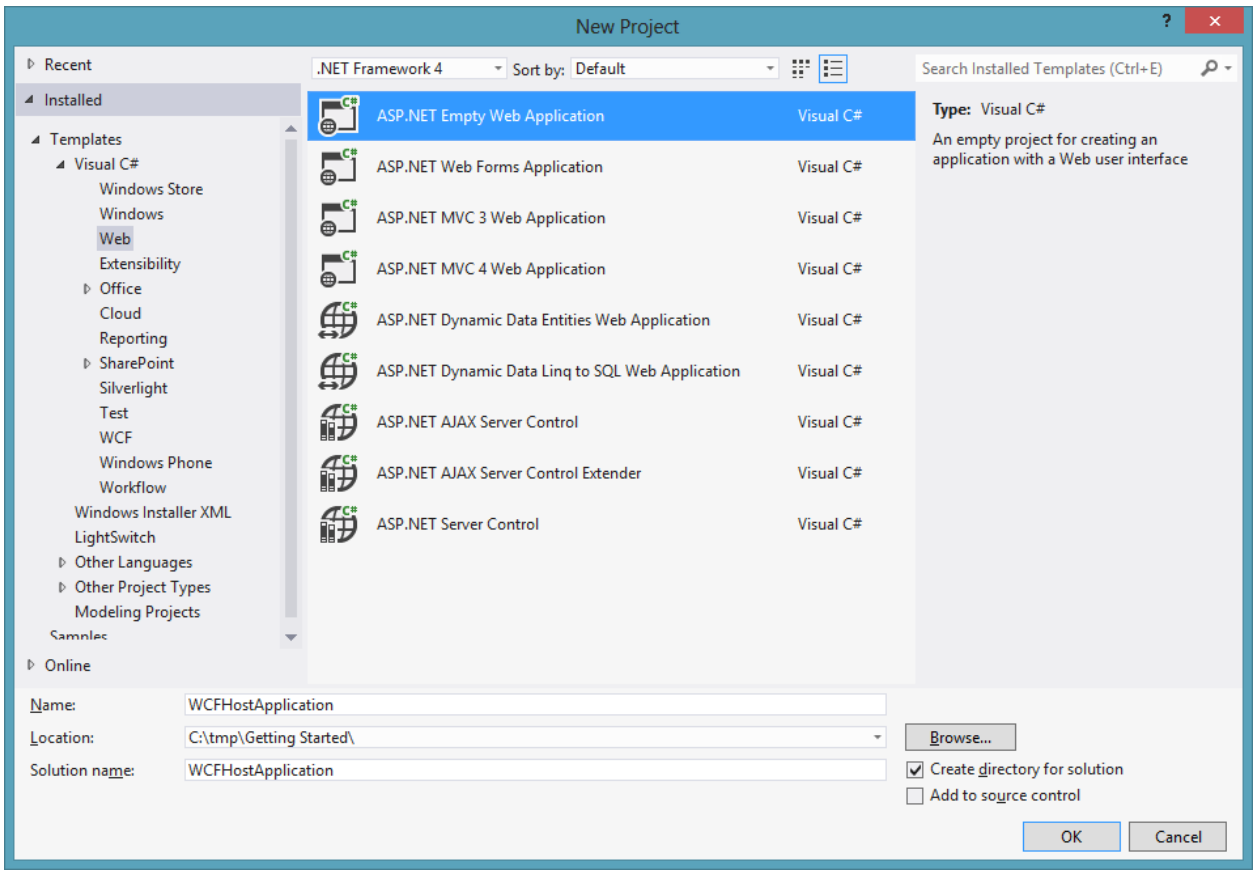
- *MS Visual Studio 2012 or higher;*
- *.NET Framework 3.5 or higher for the Server side project and .NET 4.5 or higher for Client side project;*
- *ASP.NET 2.0 or higher;*
- *Microsoft SQL Server 2008 Reporting Services (Developer ,Enterprise, Standard or higher edition) or Microsoft SQL Server 2008 R2 Reporting Services (Developer, Enterprise, Standard or higher edition) or Microsoft SQL Server 2012*
- *Microsoft SQL Reporting Services 2008 or 2008R2 or 2012 Sample Reports;*
- *WinRT XAML Viewer for Reporting Services 3.1 or higher.*

Creating a server side project using Microsoft SQL 2008/2008 R2/2012 Reporting Services report server

Step1

Create a new Web project.



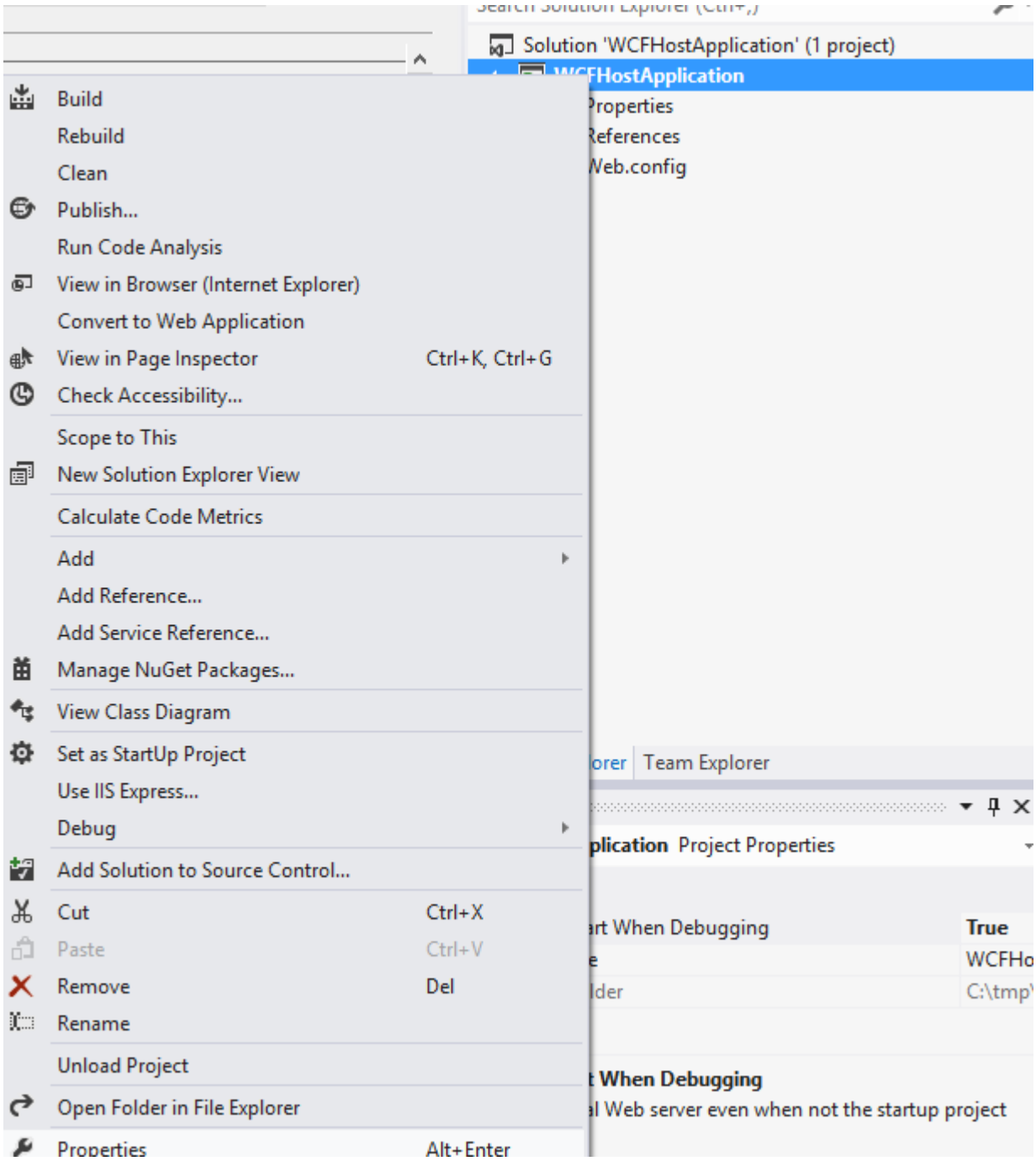


Step2

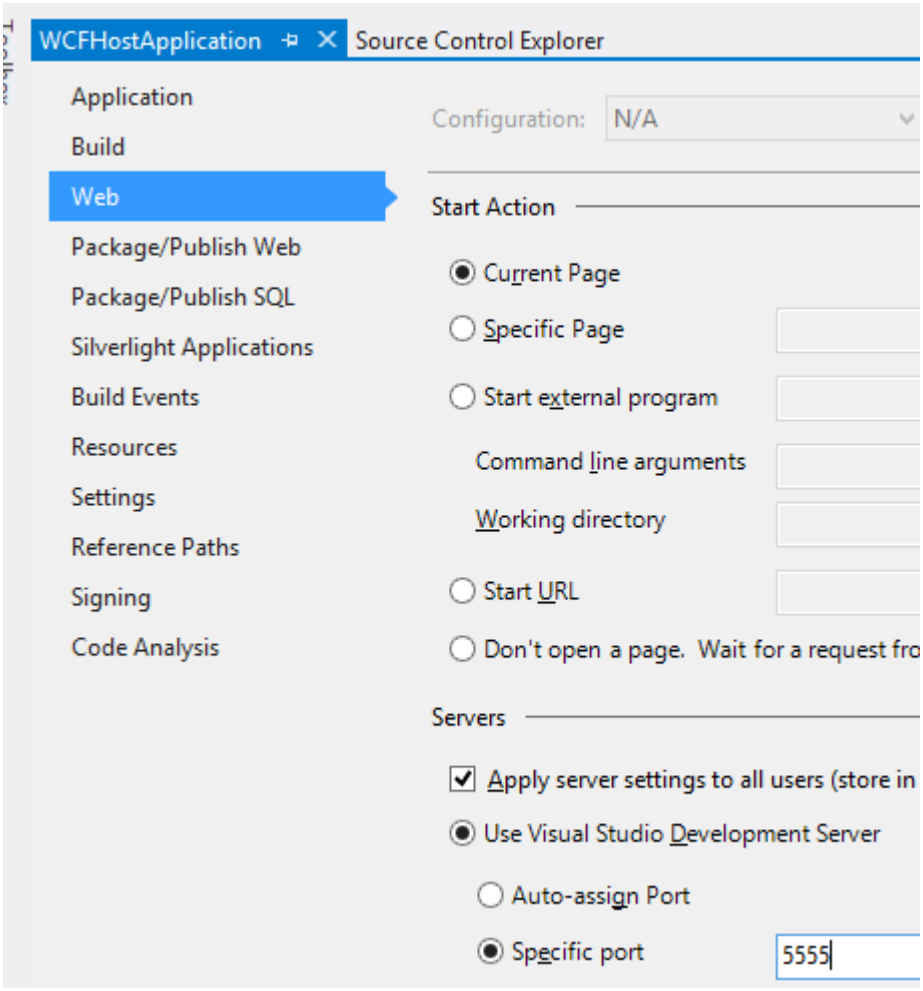
Set a web application to use static port 5555.

Set the specific port to 5555 on the Web tab of the SampleApplication.Web properties (the “Properties” item in contextual menu of the WCFHostApplication in the "Solution Explorer").

Open the WCFHostApplication properties:



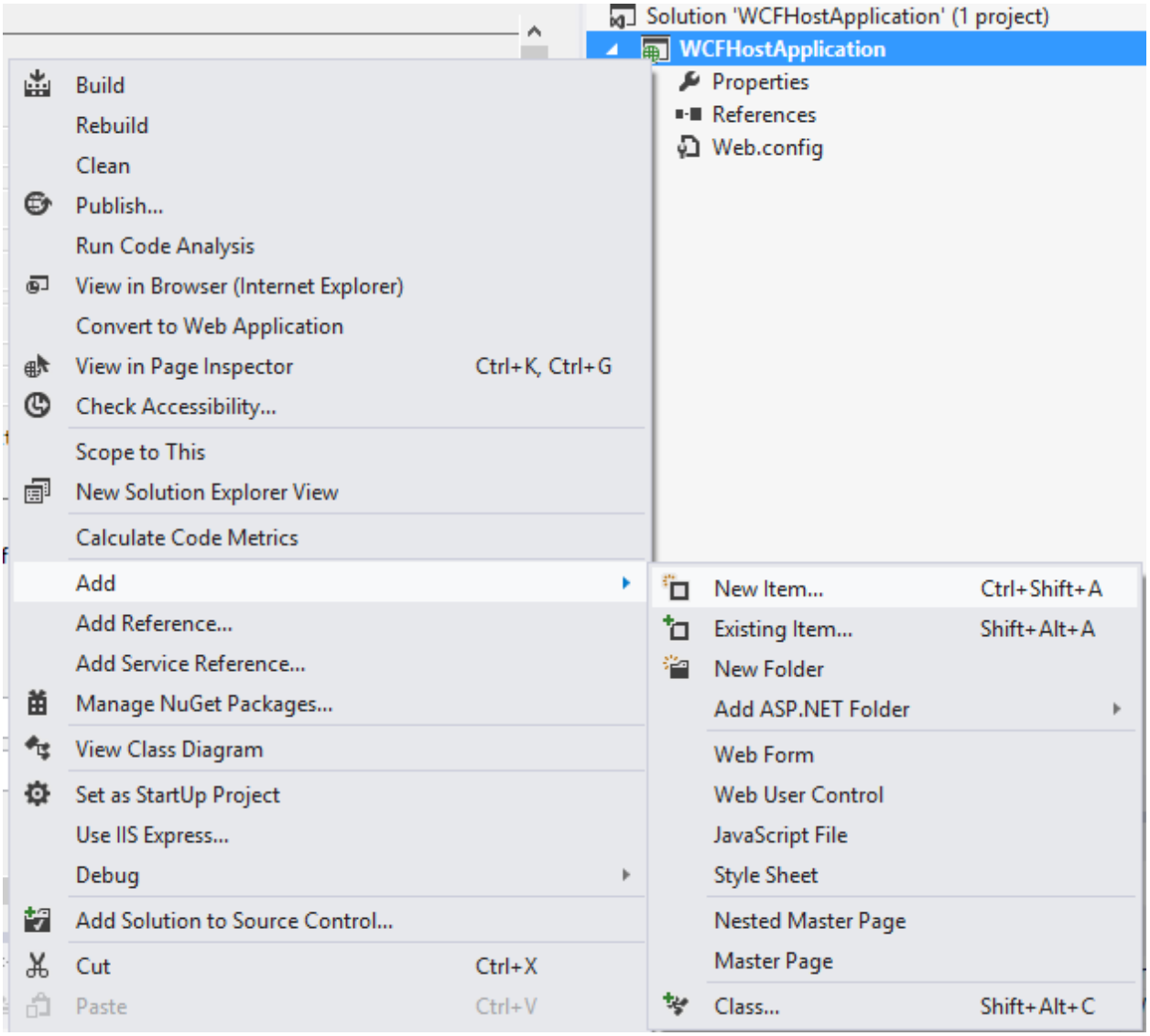
Check the Specific port option and set it to 5555 on the Web tab of the WCFHostApplication properties.



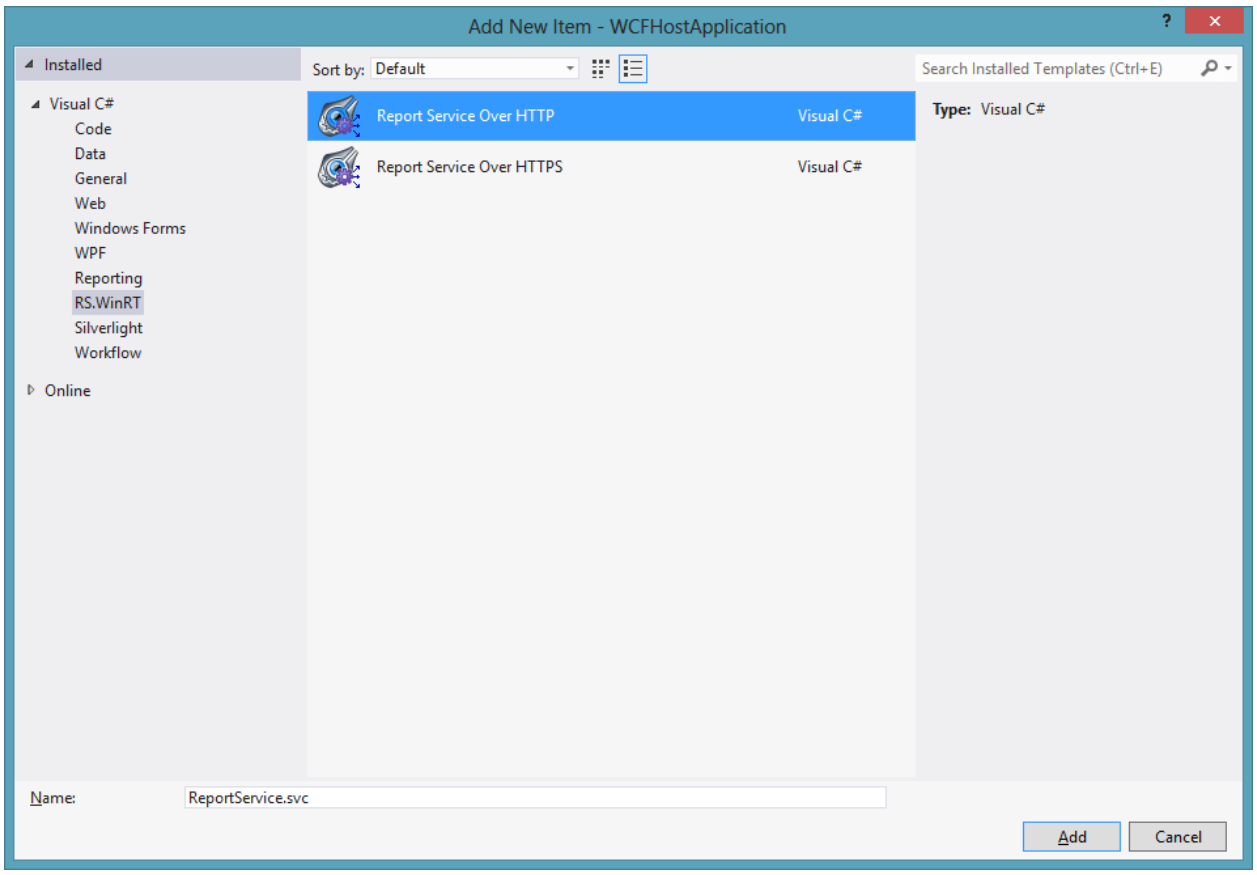
Step3

To configure the server side you can use Templates.

Add a new item to the WCFHostApplication project (right-click the WCFHostApplication in the "Solution Explorer" and choose the Add\New Item…):



Then choose Visual C#->RS.WinRT->Report Service Over HTTP.



Step4

Now we need to configure the automatically created ReportService WCF-service. Make modifications to the web.config file (double-click the WCFHostApplication\Web.config in the "Solution Explorer").

In order to configure your WCF service you just need to set up a Url attribute of the MsReportingServer section. Also you should provide your Credentials to access Reporting Services including Domain, UserName and Password.

```
  <PerpetuumSoftServices>
    <!--
          Set LogMode value="On" to log errors.
          If hosting provider does not allow to write into files set LogMode value
    to "Off".
        -->
    <LogMode value="On"/>
    <Service Type="SampleApplication.Server.ReportService,
SampleApplication.Server">>
      <MsReportingServer
Url="http://localhost/ReportServer/ReportExecution2005.asmx">
        <Credentials Domain="myDomain.com" UserName="user1" Password="myPassword"/>
      </MsReportingServer>
    </Service>
  </PerpetuumSoftServices>>
```

The *LogMode* element is designated to setup exception logging. The Log.txt file is saved under the {Web Application Root}\Log directory on your server. Please note that some Hosting Providers don't allow your application to access the file system. Try to change the attribute value to "Off" if your SampleApplication cannot be executed properly.

MsReportingServer attributes and elements description:

The *Url* attribute specifies path to your Microsoft SQL Reporting Services 2008/2008 R2/2012 web service.

The *Credentials* element defines credentials to access Reporting Services' web service.

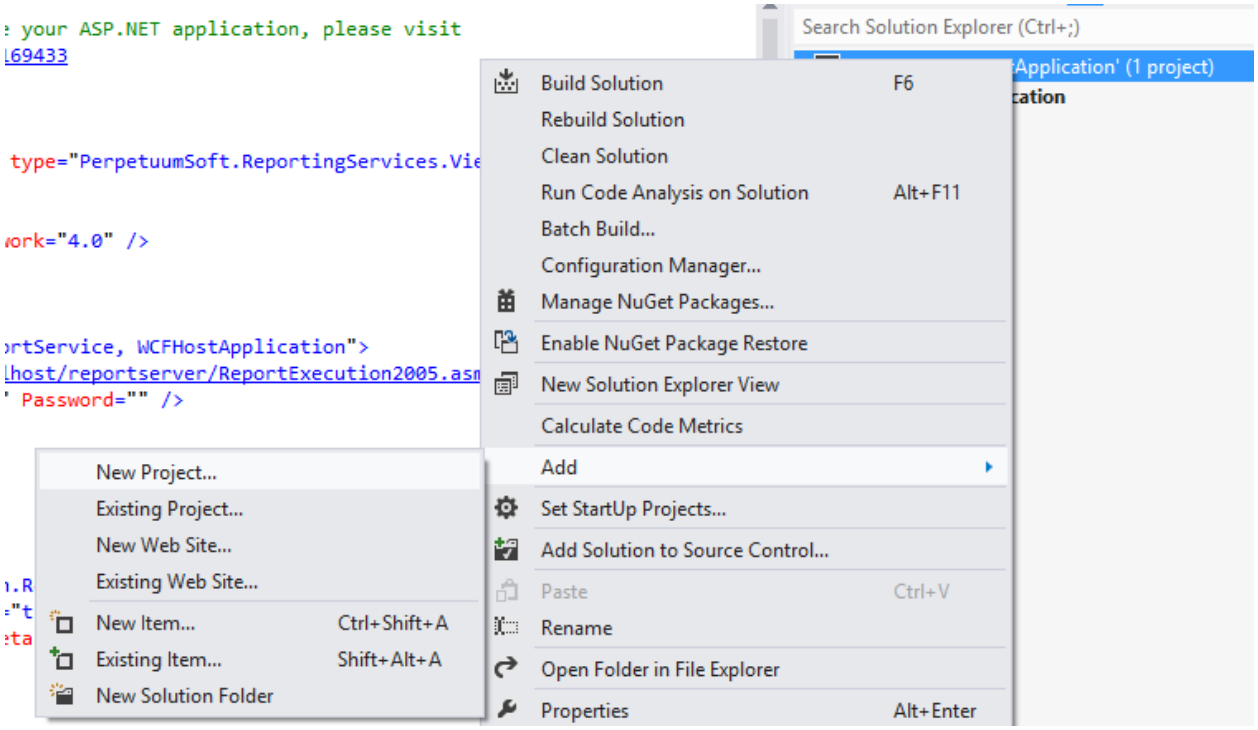
If you omit the Credentials element or leave the attributes blank, the server will use the Default Network Credentials of a Web Site.

Creating Windows Store application using Microsoft Visual Studio 2012

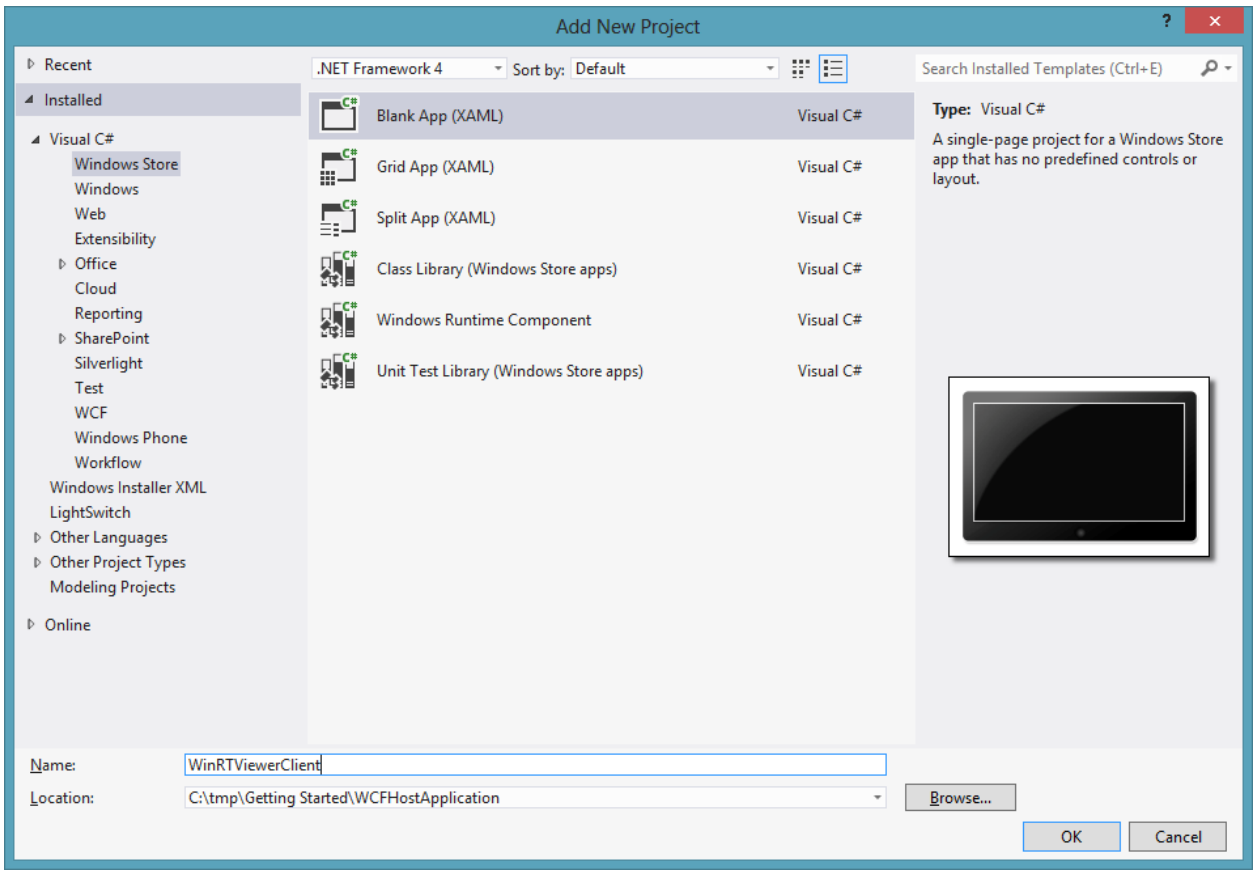
Step1

Creating Windows Store Application

Add a new project to the existing solution you've just created:



Choose Visual C#->Windows Store->Blank App (XAML) and specify a name for a new project.



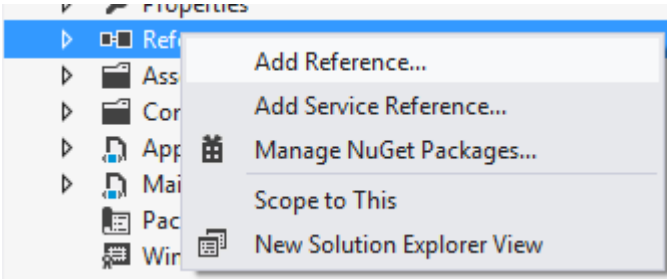
NOTE: We suppose you have Microsoft SQL Reporting Services installed and configured.

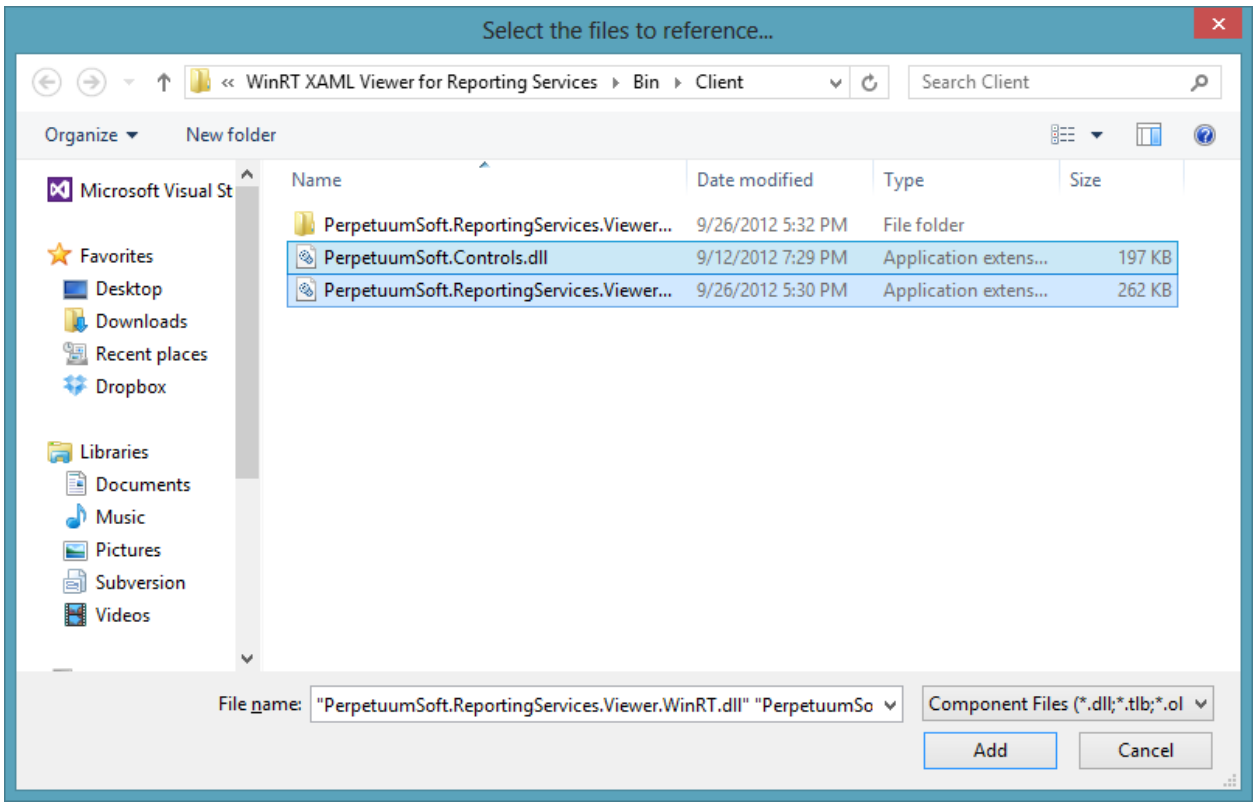
It's necessary to add the report viewer component to a Windows Store application page for report displaying. Therefore you should add references to the following libraries:

PerpetuumSoft.Controls.dll

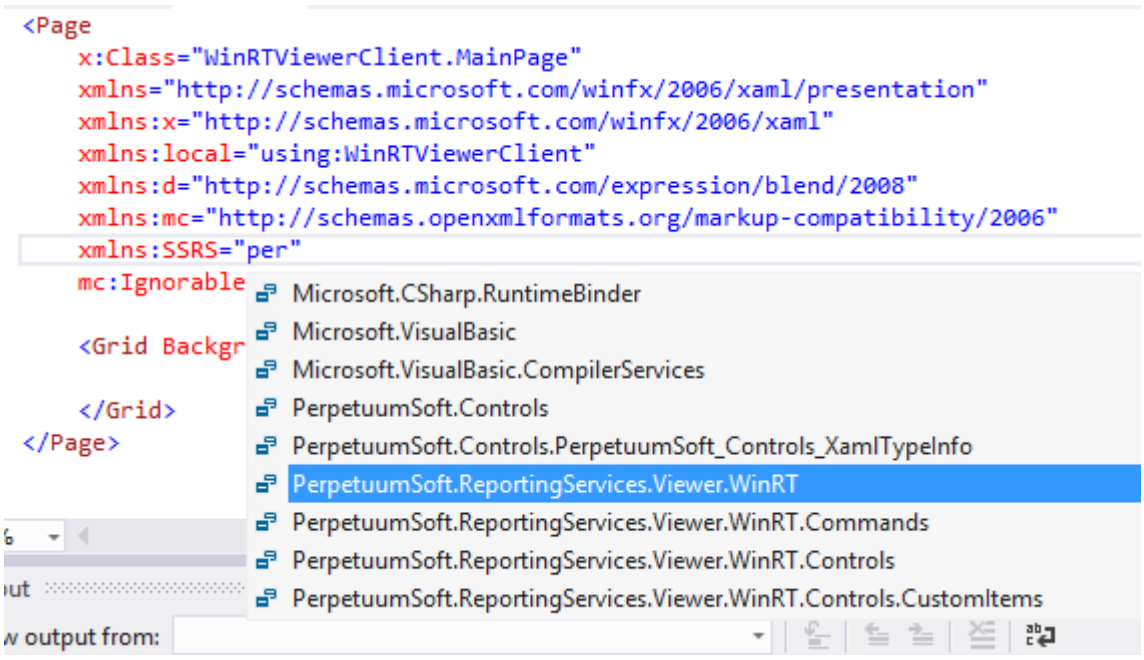
PerpetuumSoft.ReportingServices.Viewer.WinRT.dll

These assemblies are located in the Bin\Client folder of the WinRT XAML Viewer for Reporting Services installation folder.

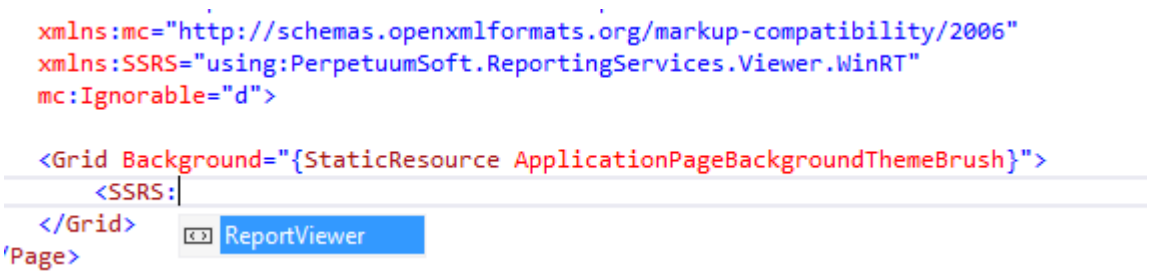




Open MainPage.xaml in the markup designer and add an xml namespace for the PerpetuumSoft.ReportingServices.Viewer.WinRT assembly.



Then add ReportViewer on your page:



And specify the required parameters:

```
<Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}">
        <SSRS:ReportViewer
            x:Name="winrtViewer"
            ServiceUrl="http://localhost:5555/ReportService.svc"
            ReportName="/Product Catalog 2008"
            ></SSRS:ReportViewer>
    </Grid>
```

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

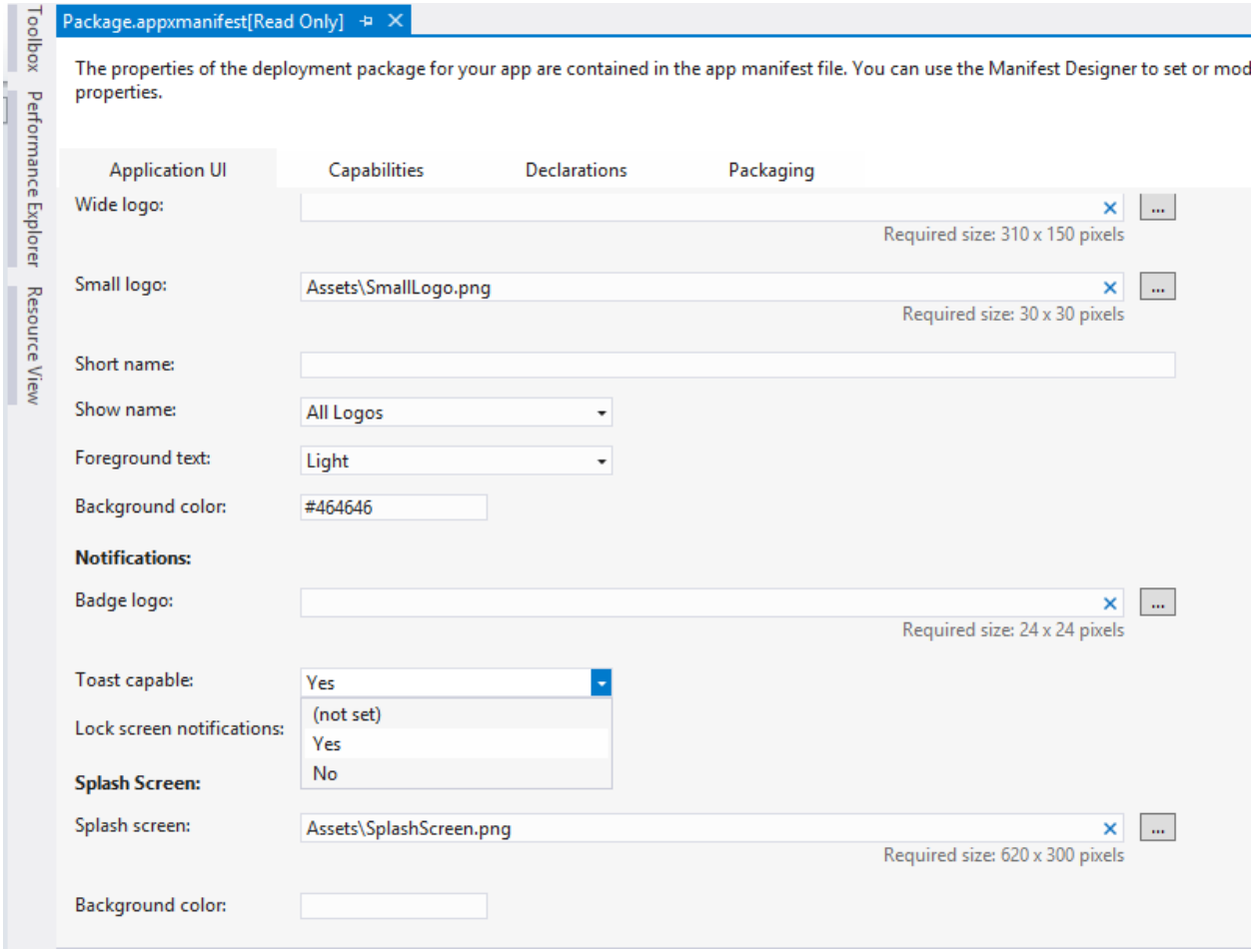
```
public sealed partial class MainPage : Page
    {
        public MainPage()
        {
            this.InitializeComponent();
            winrtViewer.Loaded += winrtViewer_Loaded;
        }

        void winrtViewer_Loaded(object sender, RoutedEventArgs e)
        {
            winrtViewer.Commands.RenderReportCommand.Execute(null);
        }

…

}
```

The RenderDocument command invocation leads to the rendering of the current report on the server and it will be displayed in the Report Viewer.

The ServiceUrl attribute specifies Url to ReportService which we've created before, the DebugMode attribute specifies the display mode exception. There are two values available "Simple" and "Full". The Full mode can be useful if any error occurs during execution.

Step2

Now it's time to configure exports. First, we need to configure the Toast notification for our application. For that purposes open *"Package.appxmanifest"*:



Then, select "*Toast capable*" -> *Yes*

The second option you can be interested in is the sharing. In order to turn it on you need to subscribe to the DataRequested event sharing:

```
        public MainPage()
        {
            this.InitializeComponent();
            …
            SubscribeSharing();
        }

        private void SubscribeSharing()
        {
            DataTransferManager dataTransferManager = DataTransferManager.GetForCurrentView();
            dataTransferManager.DataRequested += dataTransferManager_DataRequested;
        }

        void dataTransferManager_DataRequested(DataTransferManager sender,
DataRequestedEventArgs args)
        {
            if (reportViewer.HasContentToBeShared)
            {
                reportViewer.ShareContent(args.Request);
            }
        }
```

Step3

Install Reporting Services Rendering Extension.msi on your server PC.

You can find the extension installer by the following path:

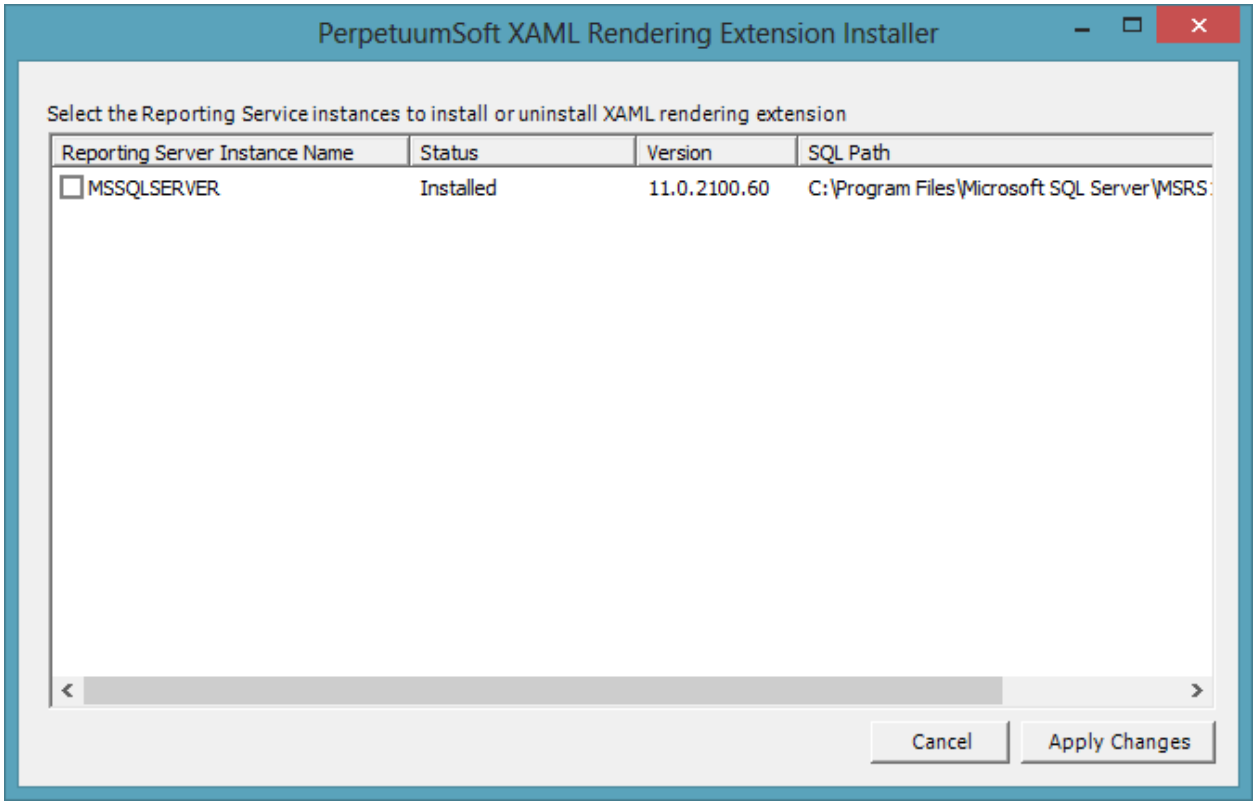
"C:\Program Files (x86)\Perpetuum Software\WinRT XAML Viewer for Reporting Services\Extension\Reporting Services Rendering Extension.msi"

NOTE: The extension must be installed ONLY to Microsoft SQL Server Reporting Services 2008/2008 R2/2012 Developer, Enterprise or Standard edtions. Make sure that your SQL Server edition supports custom rendering extensions.

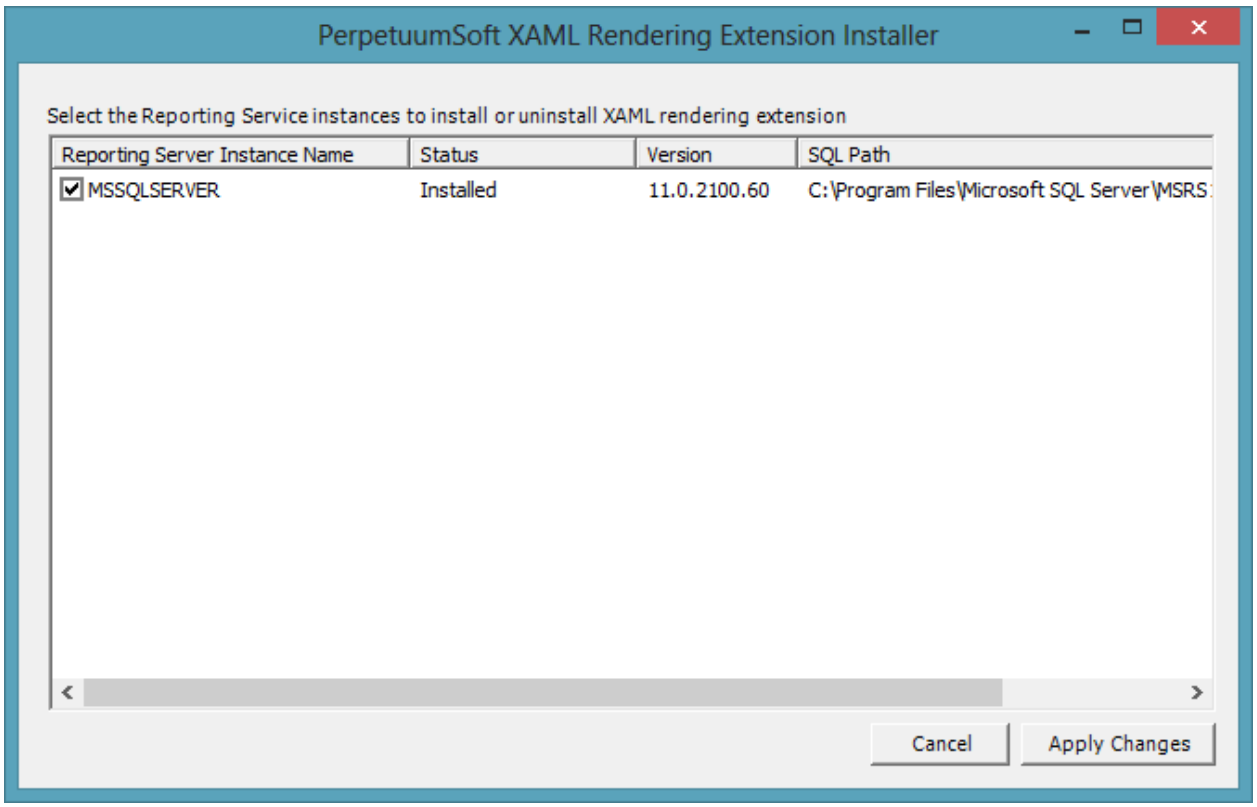
Run the extension installer on the machine where Microsoft SQL Server Reporting Services are installed.

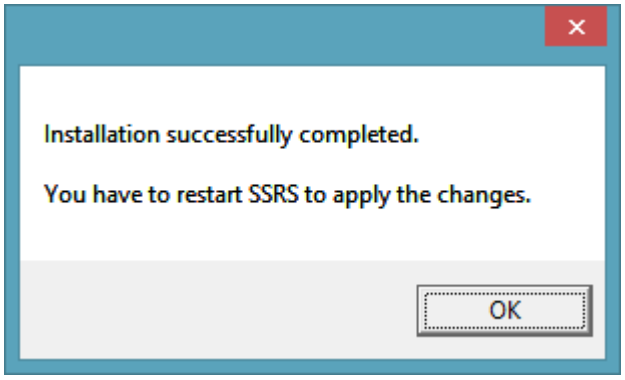
Follow the installer instructions on the screen.

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



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





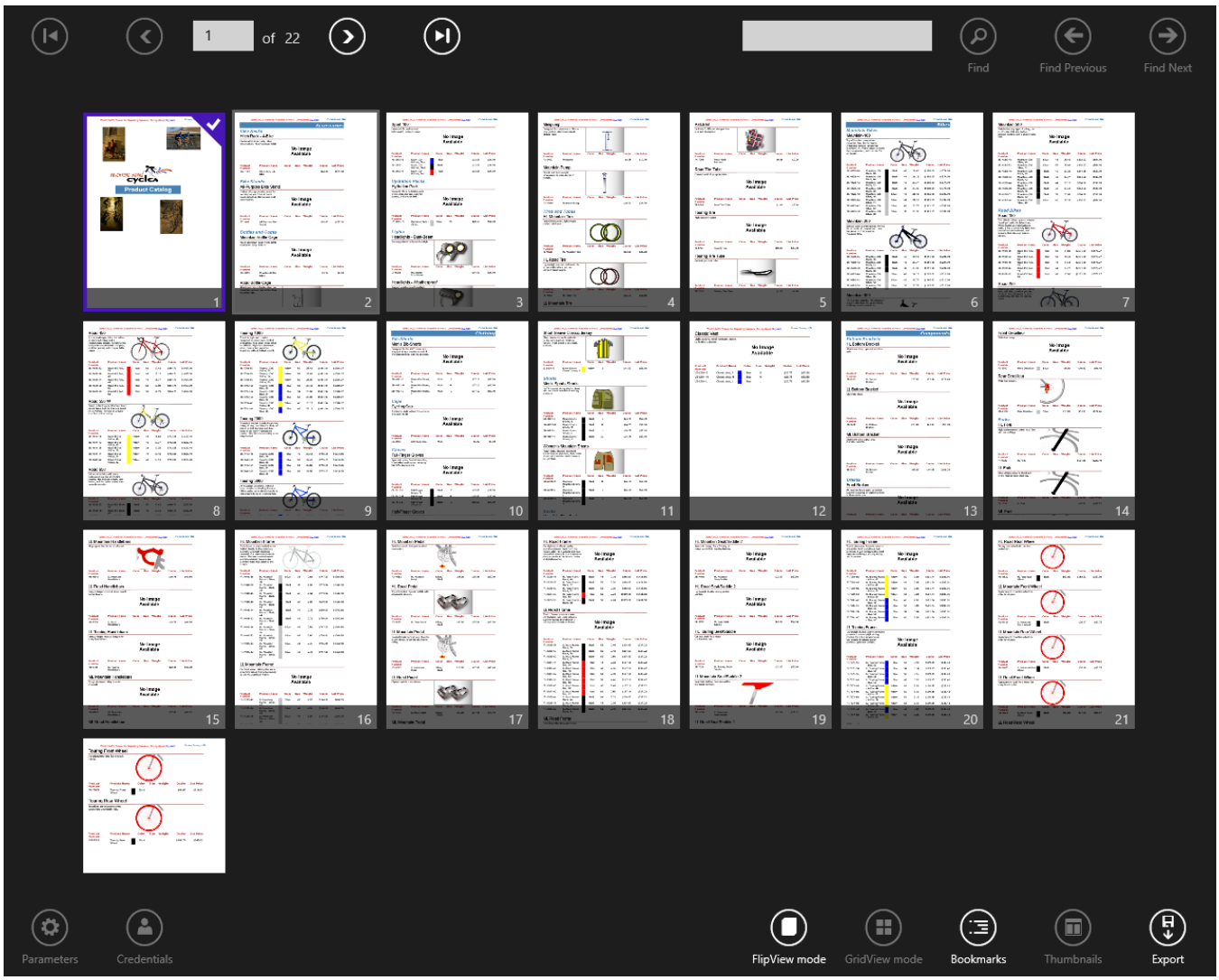
Restart the affected instances of Reporting Services.

Step4

Run the Sample Application.

Set your WinRT Client app as a startup project and hit F5 in Visual Studio environment to run the application.

NOTE: The first launch of your application may take some time. It's caused by the Slow Startup of SQL Reporting Services problem.



If you cannot run the application or any error occurs during execution please contact Perpetuum Software support team by submitting a ticket or by e-mail support@perpetuumsoft.com.

Please provide as much information as possible in the support request:

You can select the error message, copy it to clipboard and attach to your email.

You may also attach the log.txt file (if it exists) which can be found under the {Web Site Root}\Log directory.

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 most cases. If the required behavior differs greatly from the one provided by default, you can change not only many aspects of the WinRT Viewer for Reporting Services 2008/2008R2/2012 work but also the appearance of the report viewer.

More info

You can find more articles here:

http://helpcenter.perpetuumsoft.com/KB/c101/winrt-viewer-for-reporting-services.aspx

NOTE

There is a bug in Windows RTM and if you get the XAML parser error you need to apply the official Microsoft HotFix for this issue. You can find the details here:

http://social.msdn.microsoft.com/Forums/en-US/toolsforwinapps/thread/48a1d915-7f03-4058-84e3-8415d12e0b03

And if you don't want to call Microsoft in order to get this fix, you can download it here:

http://perpetuumsoft.com/Support/VS2012RTMHotfix/HotFix-KB2739194.zip