Crystal Reports®

For Visual Studio .NET®

Exposing Reports as Web Services

Version 2.0

Crystal Decisions, Inc.

895 Emerson Street

Palo Alto, CA 94301

Exposing Reports as Web Services

Introduction

Crystal Reports for Visual Studio .NET extends the powerful reporting capability of Crystal Reports on the Microsoft .NET platform. You can use the Crystal Report Designer in Visual Studio .NET to create a new Crystal report or to modify an existing Crystal report. You can then keep the report on a local machine, or publish it as a Web Service on a Web server. Depending on whether you are developing a Windows or Web application, you can host the report with either the Windows Forms Viewer or the Web Forms Viewer respectively.

This paper describes how to publish a Crystal report as a Web Service on a server, and how to consume this Web Service with a Windows application or a Web application on a client.

If you are interested in creating or adding a Crystal report to a local machine and viewing it with a Windows application, please refer to the paper "Designing and Viewing a Report in a Windows Application." For a description on viewing such reports with a Web application, please refer to the paper "Viewing a Report in a Web Application."

For more information and updates, please visit our Web site at:

http://www.crystaldecisions.com/net.

What is a Web Service?

Just as the Internet has improved communication within and between organizations, browsing data-driven pages over the Internet is falling short of the growing needs of these organizations. These organizations require the capability of remotely accessing internal business systems as well as carrying on business-to-business functions through integrated applications. Web Services provide the answer to their needs.

Architecturally, a Web Service is any middle-tier business function exposed through standard Web protocols. It uses HTTP as the Web protocol, thus allowing remote business requests to pass through company firewalls. It supports the Secure Socket Layer (SSL) for security. It is independent of any particular component technology or object-calling convention; hence programs written in any language, using any component model, and running on any operating system can access the Web Service.

Consequently, companies can develop integrated applications that consume Web services to access internal business functions as well as functions exposed by other companies. Developers use Extensible Markup Language (XML) to describe the capabilities exposed by a Web Service.

Report Web Services

A Report Web Service is a Crystal report published as a Web Service. Windows and Web applications alike can connect to a Report Web Service and host the Crystal report exposed by the Report Web Service in a Windows Forms Viewer or a Web Forms Viewer respectively.

Creating the Report Web Service on a Server

When you create a Crystal report in a project (or add an existing Crystal report to a Web Service project on a web server) and generate a Report Web Service, Visual Studio compiles the Web Service to a DLL and generates an XML file describing the public functions, input parameters, data types, and return data types exposed by the Report Web Service. Both the DLL and XML files are published on the Web server as a Report Web Service. A client on a Web browser can now invoke the Report Web Service via HTTP, whereby XML is used to pass data to and from the Web Service.

Generating the Report Web Service File on the Client

Once the Report Web Service has been published on a Web server, you can consume the Report Web Service in an application on the client side. You can add the Report Web Service from the Visual Studio Server Explorer to a Windows application or a Web application. This action has the same outcome as invoking the Report Web Service in a browser (e.g., to access the Report Web Service, Chart_src.asmx, residing on a local Web server, enter the URL http://localhost/Chart_src.asmx?wsdl in a browser). Upon doing so, a Service Description Language (.sdl) file will be automatically generated on the client.

Binding the Report Web Service to a Viewer on the Client

You can insert the appropriate viewer (i.e., a Windows Forms Viewer for a Windows application, or a Web Forms Viewer for a Web application) into the application and bind the Report Web Service to the viewer.

There are several ways to bind a Report Web Service to a viewer. For detailed information on these methods, please refer to "Appendix A: Binding a Report Web Service to a Windows Forms Viewer and a Web Forms Viewer".

Building the Application on the Client

After binding the Report Web Service to the viewer, you can build the application.

The following sections illustrate the steps to create a Web Service on a server, and the steps to create a Windows application and a Web application on the client to consume the Report Web Service.

Server for Report Web Services

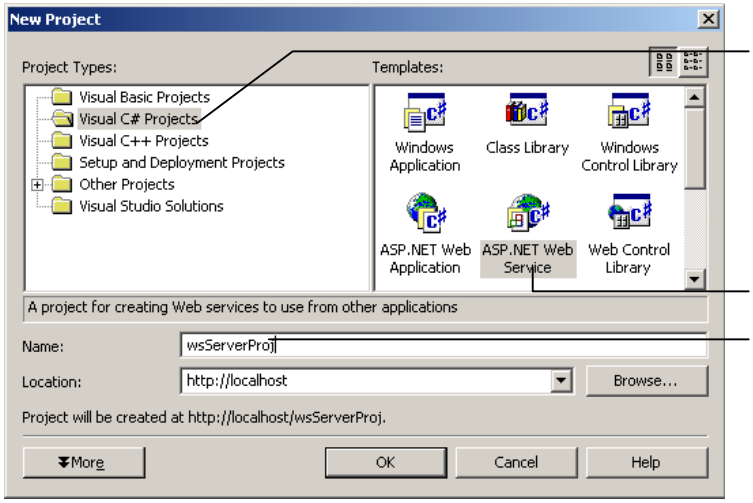
For this example, you will create a Report Web Service for the sample report, World Sales Report.rpt, on the local host http://localhost. In general, you can create and publish the Report Web Service on any Web server.

NOTE When installing Visual Studio .NET, World Sales Report.rpt is automatically added to C:\Program Files\Microsoft Visual Studio .NET\Crystal Reports\Samples\Reports\General Business.

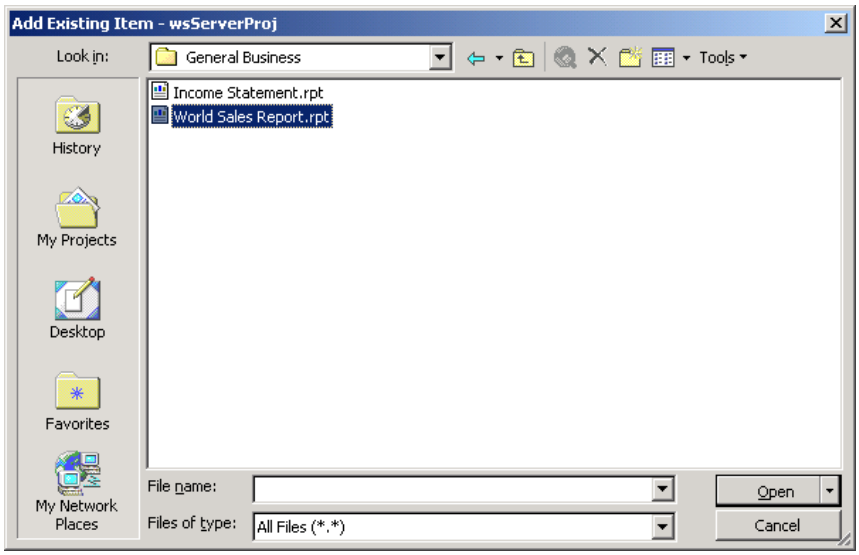
You will create a Visual C# Web Service project, wsServerProj, on http://localhost. You will then add World Sales Report.rpt to wsServerProj, and build and publish the Report Web Service on http://localhost.

TIP For the following walkthrough, the local host machine is named "angelgolfer." You can enter "localhost" in place of "angelgolfer" whenever you are referring to the local host.

1. In Visual Studio .NET, select **File** | **New** | **Project**. This invokes the **New Project** dialog box.
2. In the **New Project** dialog box, select **Visual C# Projects** on the left pane and **ASP.NET Web Service** on the right pane.
3. To change the default project name, enter "wsServerProj" in the **Location** field after http://localhost/. Click **OK**.



4. In the Solution Explorer, right-click wsServerProj and select **Add** | **Add Existing Item** from the shortcut menu. This will invoke the **Add Existing Item – wsServerProj** dialog box.
5. In the **Add Existing Item – wsServerProj** dialog box, select World Sales Report.rpt from the folder C:\Program Files\Microsoft Visual Studio .NET\Crystal Reports\Samples\Reports\General Business.



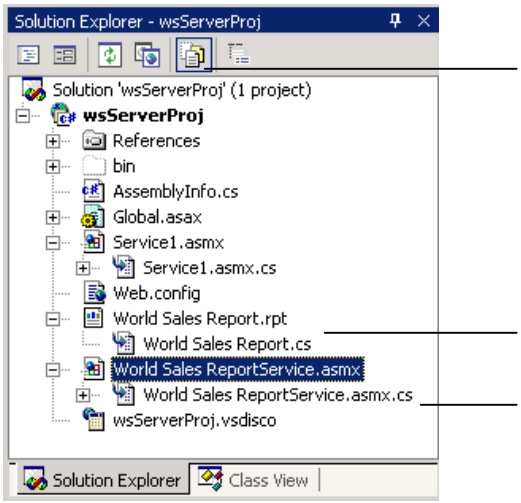
NOTE Select **All Files (*.*)** for **Files of type** in order to display .RPT files in a folder.

6. Upon adding World Sales Report.rpt to wsServerProj, World Sales Report.cs that contains a World_Sales_Report class is generated. This file is normally hidden; you can click the **Show All Files** icon in the Solution Explorer to display it under the World Sales Report.rpt node.

7. Right-click World Sales Report.rpt and select **Publish as web service** from the shortcut menu. The Report Web Service for World Sales Report.rpt, World Sales ReportService.asmx, is generated and displayed in the Solution Explorer.

   A source file for the Report Web Service, World Sales ReportService.asmx.cs, is also generated. This file allows users to customize the Report Web Service, e.g., to call another Web Service. It is normally hidden; you can click the **Show All Files** icon in the Solution Explorer to display it under the World Sales ReportService.asmx node.

8. Select **Build** | **Build Solution** to build and make the Report Web Service for World Sales Report.rpt available on the local host. In Visual Studio .NET, you can anchor the Server Explorer by clicking the "pin" icon at the top-right corner of the Server Explorer.



Setting Up the Sample Database

When you install Visual Studio .NET, the sample database, xtreme.mdb is automatically installed in C:\Program Files\Microsoft Visual Studio .NET\Crystal Reports\Samples\Database. World Sales Report.rpt assumes xtreme.mdb is at this location.

To change the location of xtreme.mdb

If you have installed xtreme.mdb at a location other than the default (i.e., C:\Program Files\Microsoft Visual Studio .NET\Crystal Reports\Samples\Database), you should do a Set Location for World Sales Report.rpt to point the report to the actual location of the database.

1. In Visual Studio .NET, double click on the World Sales Report.rpt in the Solution Explorer. This opens the report in the Visual Studios .NET designer.
2. In the Crystal Report Designer, right-click in a report area. Choose **Database** | **Set Location**. This invokes the **Set Location** dialog box.
3. In the **Set Location** dialog box, click the down-arrow under **Current Data Source**. Select the data source assumed by World Sales Report.rpt, by default the World Sales Report.rpt is base off ODBC data source Xtreme Sample Database that points to **C:\Program Files\Microsoft Visual Studio .NET\Crystal Reports\Samples\Database\xtreme.mdb**. You will point this assumed location to the actual location of the database.
4. You may use any one of these database technologies to connect the report to the actual data source: OLE DB (ADO), ODBC (RDO), Database Files (i.e., using native drivers), or ADO.NET (XML) (which is under **More Data Sources**). For the purpose of this walkthrough, under **Replace with**, double-click **OLE DB (ADO)**. This invokes the **OLE DB (ADO)** dialog box.

   NOTE If you are already connected to xtreme.mdb, proceed with step 9. Or, if you are connected to another database through OLE DB (ADO), select **Make New Connection**.

5. In the **OLE DB (ADO)** dialog box, highlight the OLD DB provider, **Microsoft Jet 4.0 OLE DB Provider**. Then click the **Next** button.
6. Provide the necessary information to access the actual data source: for **Database Name**, click the square button adjacent to the entry box. Select xtreme.mdb according to where you have installed it. Click **Next**.
7. Update any advanced information for your data source, if necessary. Then click **Finish**. You will be returned to the **Set Location** dialog box.
8. Redirect the report to look for the database in the actual location: in the **Set Location** dialog box, under **Replace with**, highlight the actual location of the database. Click **Replace**. You will find this location updated under **Current Data Source**. Then click **Close**.
9. If you have modified fields in xtreme.mdb, right-click in a report area, point to **Database** and select **Verify Database**. A message titled **Verify Database** appears. Click **OK** to fix unmapped fields in the database.
10. Choose **File** | **Save World Sales Report.rpt** to save the report with the updated database location.

Client for Report Web Services

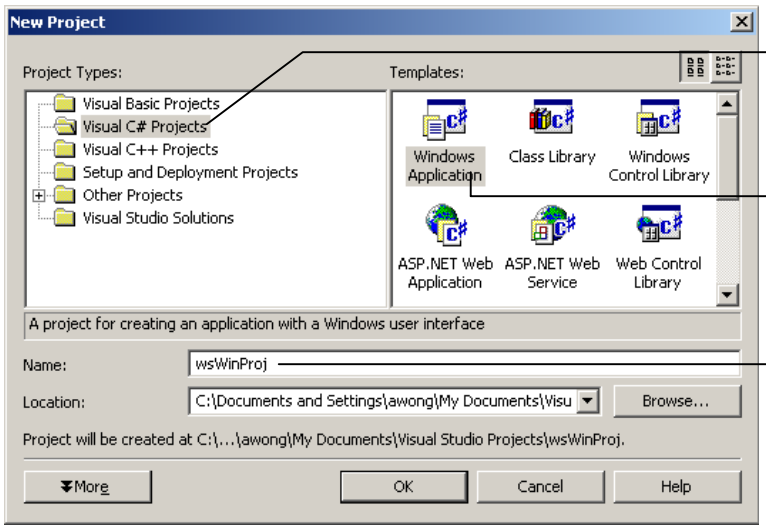
The client for a Report Web Service can be either a Windows application with a Windows Forms Viewer, or a Web application with a Web Forms Viewer. In either scenario, the corresponding viewer hosts the Crystal report by connecting to the Report Web Service for that report.

Windows Client for Report Web Services

You will create a Visual C# Windows project called wsWinProj, insert a Windows Forms Viewer into the Windows Form Form1.cs, and connect to the Report Web Service for World Sales Report.rpt created above. When you build and run wsWinProj, this application will host World Sales Report.rpt by consuming its Report Web Service.

1. In Visual Studio .NET, select **File** | **New** | **Project**. This invokes the **New Project** dialog box.
2. In the **New Project** dialog box, select Visual C# Projects in the left pane. Select Windows Application in the right pane.

   You can log on as a regular user and choose to create your project in the folder of your choice. For this example, you will create the project wsWinProj in the default project folder if you are logged on as Administrator: C:\Documents and Settings\Administrator\My Documents\Visual Studio Projects.
3. Enter "wsWinProj" in the **Name** field. Accept the default path, "C:\Documents and Settings\Administrator\My Documents\Visual Studio Projects", in the **Location** field. Click **OK**. This creates a Visual C# Windows project, wsWinProj, in C:\Documents and Settings\Administrator\My Documents\Visual Studio Projects.

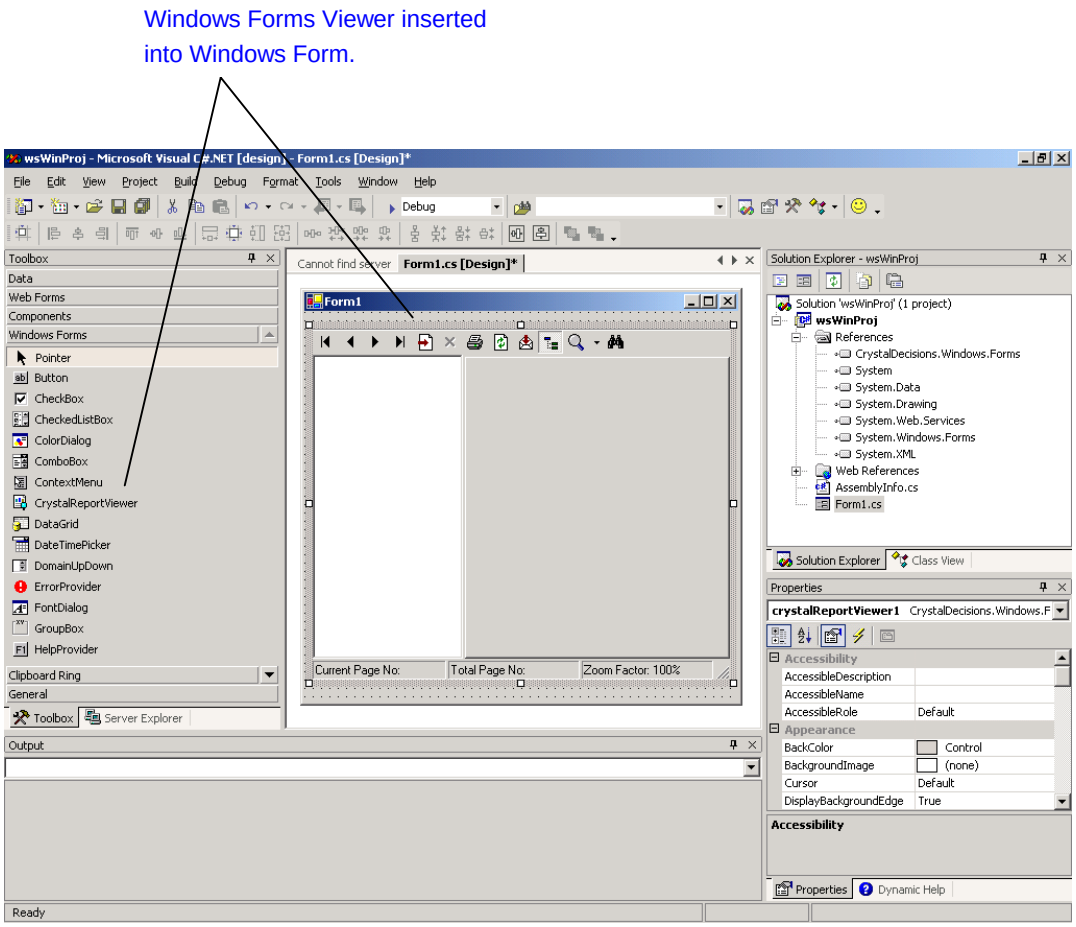


4. Upon creating wsWinProj, Visual Studio opens the Windows Form, Form1.cs. Open the Visual Studio Toolbox. The **Windows Forms** tab of the Toolbox displays a list of controls applicable to Windows Forms, including the Windows Forms Viewer control (which is labeled as the **CrystalReportViewer**).

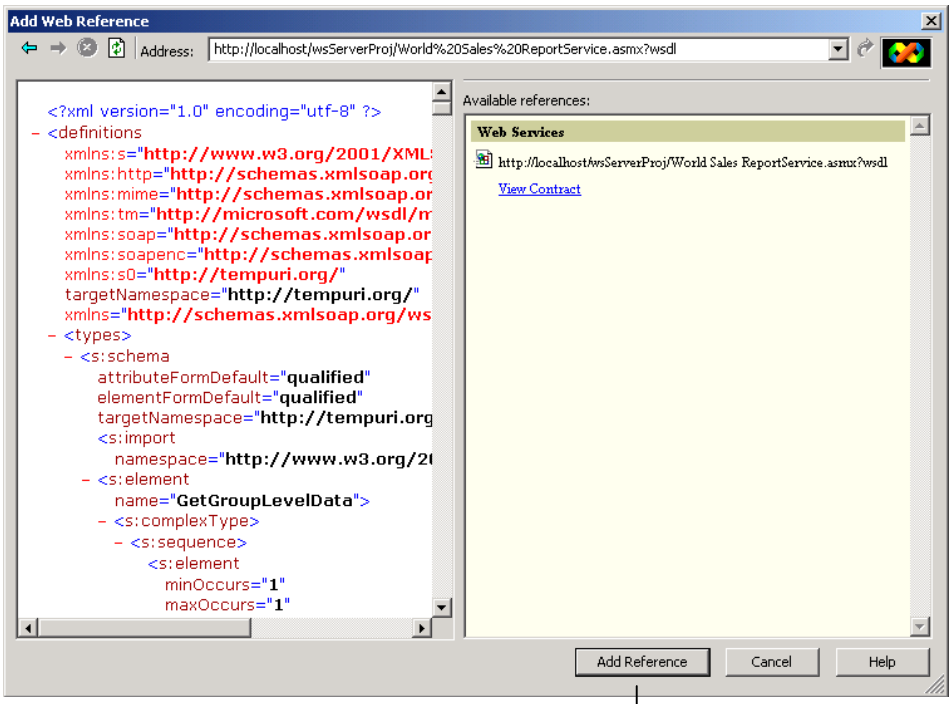
   TIP In Visual Studio .NET, you can anchor the Toolbox by clicking the "pin" icon at the top-right corner of the Toolbox.

   Also, items in a Toolbox tab can be displayed alphabetically by right-clicking the tab and selecting **Sort Items Alphabetically** from the shortcut menu.

5. From the **Windows Forms** tab of the Toolbox, drag and drop the Windows Forms Viewer control to Form1.cs. This action inserts a Windows Forms Viewer into the Windows Form. The Windows Forms Viewer is named "crystalReportViewer1" by default. You can move and size the Windows Forms Viewer as desired.

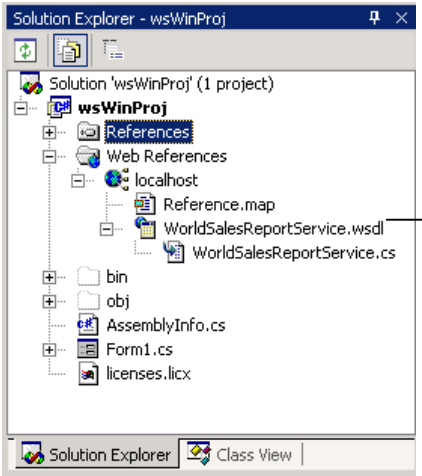


6. Add the Report Web Service to wsWinProj by right-clicking **References** in the Solution Explorer, and selecting **Add Web Reference**.
7. In the Add Web Reference dialog box, type the following in the **Address** field: **http://localhost/wsServerProj/World Sales ReportService.asmx?wsdl**
8. Click the green arrow icon beside the **Address** field. The contract for the Report Web Service for World Sales Report.rpt is displayed.
9. Select **Add Reference** to add this Report Web Service to wsWinProj.



Click Add Reference to add the Report Web Service for World Sales Report.rpt to wsWinProj.

Once you have added the Report Web Service file to wsWinProj, if you have selected **Show All Files** in the Solution Explorer, you will find **WorldSalesReportService.wsdl** and an associated **References.cs** Web References defining a class for this Report Web Service displayed under the **Web References** node.



Report Web Service on localhost added to client project.

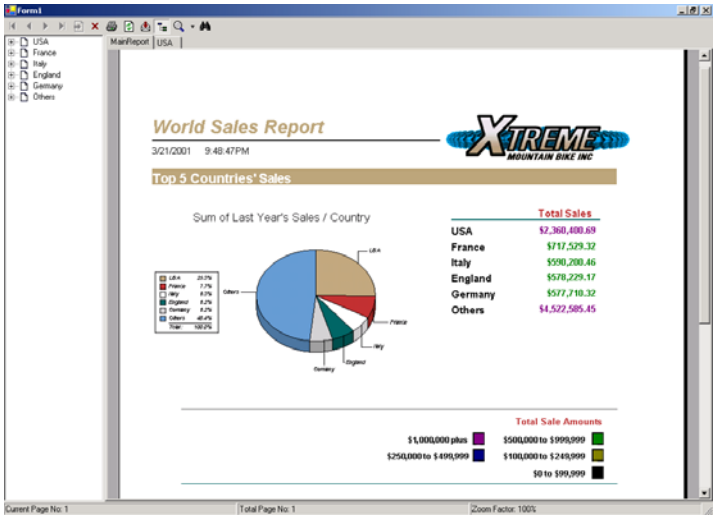
You can now bind the Report Web Service for World Sales Report.rpt to the Windows Forms Viewer in Form1.cs. You can do so through code in the source file for the Windows Form, Form1.cs.

10. Select Form1.cs in the Solution Explorer, and click the **View Code** icon.
11. In the `Form1()` function after the call to `InitializeComponent()`, enter the following lines of (case sensitive) code:

```
crystalReportViewer1.ReportSource = new
localhost.World_Sales_ReportService();
```

where "localhost" is the namespace defined in WorldSalesReportService.cs (or, as an alternative, to be more complete, specify "wsWinProj.localhost"), and World_Sales_ReportService is the proxy class name defined in the same source file as well.

12. Select **Build** | **Build Solution** to build the Windows application.
13. Select **Start Without Debugging** from the **Debug** menu to run the application that will host the World Sales Report.rpt by consuming its Report Web Service.



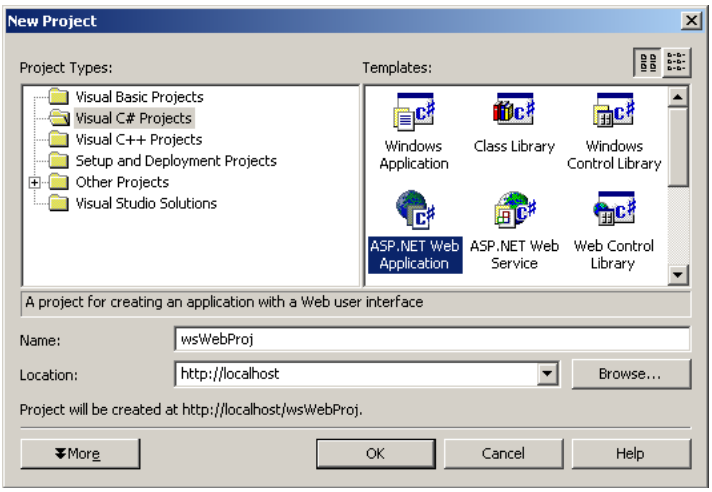
Web Client for Report Web Service

In this section, you will create a Visual C# Web project, wsWebProj, which is on the local host http://localhost, insert a Web Forms Viewer into the Web Form WebForm1.cs, and connect to the Report Web Service for World Sales Report.rpt created above. When you build and run wsWebProj, this application will host World Sales Report.rpt by consuming its Report Web Service.

1. In Visual Studio .NET, select **File** | **New** | **Project**. This invokes the **New Project** dialog box.
2. In the **New Project** dialog box, select **Visual C# Projects** on the left pane.
3. Select **ASP.NET Web Application** on the right pane.

You may choose to create your project on any Web server of your choice. For the purpose of this example, you will create the project in the default project location, http://localhost, which is the local IIS Web server.

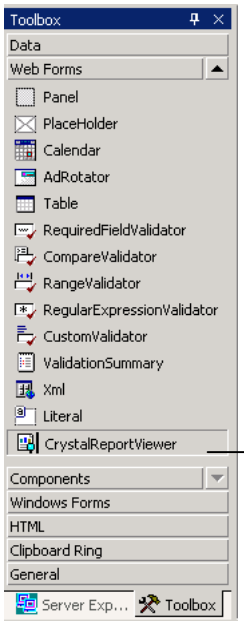
4. To change the default project name, enter "wsWebProj" in the **Location** field after http://localhost/. Click **OK**.



After you click **OK** in the **New Project** dialog, Visual Studio creates a Web Forms page (with the default name "WebForm1.aspx") in the project wsWebProj.

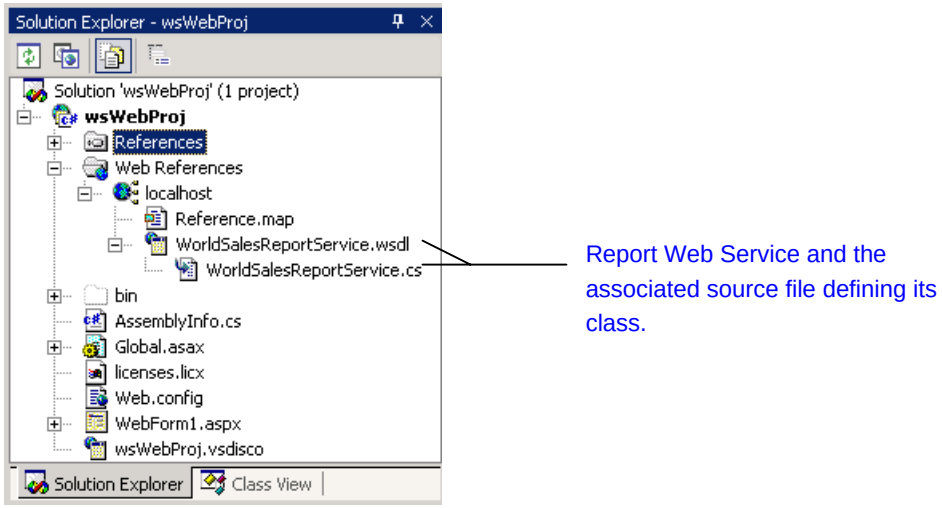
The window for the Web Forms page consists of two tabs: **Design** and **HTML**. The **Design** tab allows you to program with the Visual Studio IDE. The **HTML** tab contains the HTML source that supports ASP.NET.

Upon creating WebForm1.aspx, Visual Studio opens it in the **Design** tab. The **Web Forms** tab of the Visual Studio Toolbox displays a set of controls for the Web Form, including the Web Forms Viewer control that is labeled **CrystalReportViewer**.



By default, the Web Forms Viewer control is at the bottom of the list of Web Form controls.

5. Drag and drop the Web Forms Viewer control from the Toolbox into the **Design** tab of WebForm1.aspx.
6. Add the Report Web Service to wsWebProj by right-clicking **References** in the Solution Explorer, and selecting **Add Web Reference**.
7. In the Add Web Reference dialog box, type the following in the **Address** field: **http://localhost/wsServerProj/World Sales ReportService.asmx?wsdl**
8. Click the green arrow icon beside the **Address** field. The contract for the Report Web Service for World Sales Report.rpt is displayed.
9. Select **Add Reference** to add this Report Web Service to wsWebProj.
10. Once you have added the Report Web Service to wsWebProj, if you click the **Show All Files** icon in the Solution Explorer, you will find **WorldSalesReportService.wsdl** and an associated **References.cs** defining a class for this Report Web Service displayed under the **Web References** node.



You can now bind the Report Web Service for World Sales Report.rpt to the Web Forms Viewer. You can do so through code in the source file for the Web Form, WebForm1.aspx.cs.

11. Select WebForm1.aspx in the Solution Explorer, and click the **View Code** icon.

12. In the `OnInit()` function after the call to `base.OnInit(e)`, enter the following lines of code:

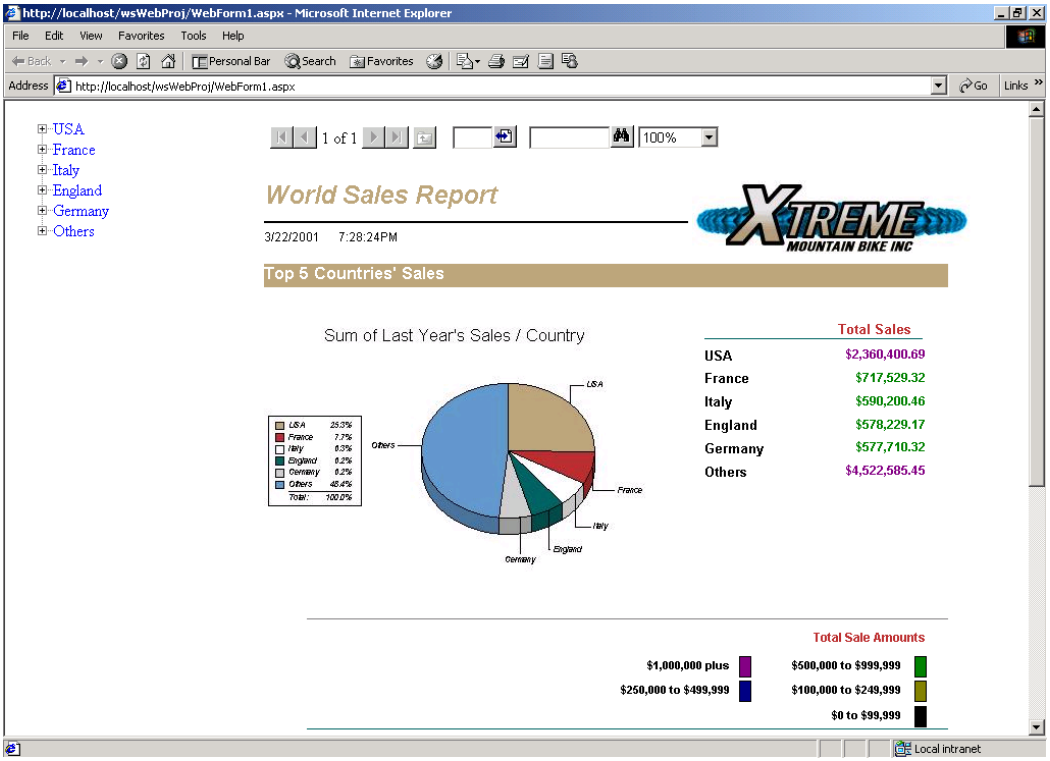
    ```
    CrystalReportViewer1.ReportSource = new
    localhost.World_Sales_ReportService();
    ```

    where "localhost" is the namespace defined in WorldSalesReportService.cs (or, as an alternative, to be more complete, specify "wsWebProj.localhost"), and World_Sales_ReportService is the proxy class name defined in the same source file.

13. Select **Build** | **Build Solution** to build the Web application.

14. Select **Start Without Debugging** from the **Debug** menu to run the application that will host the World Sales Report.rpt by consuming its Report Web Service.

    There are a number of ways to bind a Report Web Service to a Web Forms Viewer. For more information, please refer to Appendix A: Binding a Report Web Service to a Windows Forms Viewer and a Web Forms Viewer.



Appendix A: Binding a Report Web Service to a Windows Forms Viewer and a Web Forms Viewer

In general, there are four methods you can use to bind a Report Web Service to a Windows Forms Viewer or a Web Forms Viewer:

- By a Report Web Service
- By a proxy object
- By Enterprise Report object
- By Server File Report object

Windows Forms Viewer

The following section lists several scenarios and describes the methods you can use to bind a Report Web Service to a Windows Forms Viewer.

Scenario A – Report Web Service

Suppose the following conditions exist and a Crystal report is available as a Report Web Service:

- The report, My Report.rpt, is available as a Report Web Service, My Report.asmx.
- MyReportService.asmx has been published on the Web server, MyServer, in the project, ServerProject.
- On the client side, you have created a Visual C# Windows project, MyProject.
- You have inserted a Windows Forms Viewer in the Windows Form, Form1.cs, in MyProject.

You can bind the Report Web Service for My Report.rpt to the Windows Forms Viewer using the following method:

By Report Web Service:

You can specify the ReportSource property through the Code view for Form1.cs:

```
crystalReportViewer1.ReportSource = "http://MyServer/ServerProject/My
ReportService.asmx";
```

Scenario B – Report Web Service Added to Project

Suppose the conditions in Scenario A exist, plus the following:

- On the client, you have added the Report Web Service to MyProject as a Web Reference.

Adding a Report Web Service to a client project creates a proxy class. You can bind the Report Web Service for My Report.rpt to the Windows Forms Viewer using an instance of the proxy object.

By proxy object:

In the Code view of Form1.cs, after the call to `InitializeComponent()`, specify:

```
crystalReportViewer1.ReportSource = new localhost.My_ReportService();
```

Scenario C – Crystal Enterprise Report Added to Windows Form

Suppose the following conditions exist:

- On the client side, you have created a Visual C# Windows project, MyProject.
- You have inserted a Windows Forms Viewer in the Windows Form, Form1.cs, in MyProject.
- On the client, you have added a Crystal Enterprise report, MyCEReport.rpt, to the Windows Form (by dragging and dropping the report from the Server Explorer to the Design view of Form1.cs, or by right-clicking MyCEReport.rpt in the Server Explorer and selecting **Add to Designer**. For more information on Crystal Enterprise reports, please refer to the online help in Visual Studio .NET.)

By Crystal Enterprise report object:

You can bind the Crystal Enterprise report, MyCEReport.rpt, to the Windows Forms Viewer through the Properties window or through code in Form1.cs:

- Select the Windows Forms Viewer in the Design view of Form1.cs. In the Properties window, for the **ReportSource** property, select **enterpriseReport1[CrystalDecisions.ReportSource.EnterpriseReport]** from the drop down list.
- Bind the Crystal Enterprise report object to the Windows Forms Viewer by specifying the ReportSource property through the Code view for Form1.cs, in `Form1()` after the call to `InitializeComponent()`:

  ```
  crystalReportViewer1.ReportSource = enterpriseReport1;
  ```

Scenario D –Server File Report Added to Windows Form

Suppose the following conditions exist:

- On the client side, you have created a Visual C# Windows project, MyProject.
- You have inserted a Windows Forms Viewer in the Windows Form, Form1.cs, in MyProject.
- On the client, you have added a Server File report, MyServerFileReport.rpt, to the Windows Form (by dragging and dropping MyServerFileReport.rpt from the Server Explorer to the Design view of Form1.cs, or by right-clicking the Server Explorer and selecting **Add to Designer**. For more information on Server File reports, please refer to the online help in Visual Studio .NET.)

By Server File report object:

You can bind MyServerFileReport.rpt to the Windows Forms Viewer through the Properties window or through code in Form1.cs:

- Select the Windows Forms Viewer in the Design view of Form1.cs. In the Properties window, for the ReportSource property, select "serverFileReport1[CrystalDecisions.ReportSource.ServerFileReport]" from the drop down list.
- Bind the Crystal Enterprise report object to the Windows Forms Viewer by specifying the ReportSource property through the Code view for Form1.cs, after the call to `InitializeComponent()`:

```
crystalReportViewer1.ReportSource = serverFileReport1;
```

Web Forms Viewer

The following section lists several scenarios and describes the methods you can use to bind a Report Web Service to a Web Forms Viewer in each scenario.

Scenario E – Report Web Service

Suppose the following conditions exist and a Crystal report is available as a Report Web Service:

- The report, My Report.rpt, is available as a Report Web Service, My Report.asmx.
- MyReportService.asmx has been published on the Web server, MyServer, in the project, ServerProject.
- On the client side, you have created a Visual C# Web project, MyProject, on the local server, http://localhost.
- You have inserted a Web Forms Viewer in the Web Form, WebForm1.aspx, in MyProject.

You can bind the Report Web Service for My Report.rpt to the Web Forms Viewer using the following method:

By Report Web Service:

You can specify the ReportSource property through the Code view for WebForm1.cs:

```
CrystalReportViewer1.ReportSource =
      "http://MyServer/ServerProject/My ReportService.asmx";
```

Scenario F – Report Web Service Added to Project

Suppose the conditions in Scenario D exist, plus the following:

- On the client, you have added the Report Web Service to MyProject as a Web Reference.

Adding a Report Web Service to a client project creates a proxy class. You can bind the Report Web Service for My Report.rpt to the Web Forms Viewer using an instance of the proxy object.

By proxy object:

In the Code view for WebForm1.cs, after the call to `InitializeComponent()`, specify:

```
CrystalReportViewer1.ReportSource = new localhost.My_ReportService();
```

Scenario G – Crystal Enterprise Report Added to Web Form

Suppose the following conditions exist:

- On the client side, you have created a Visual C# Web project, MyProject.
- You have inserted a Web Forms Viewer in the Web Form, WebForm1.cs, in MyProject.
- On the client, you have added a Crystal Enterprise report, MyCEReport.rpt, to the Web Form (by dragging and dropping the report from the Server Explorer to the Design tab of WebForm1.cs, or by right-clicking MyCEReport.rpt in the Server Explorer and selecting **Add to Designer**. For more information on Crystal Enterprise reports, please refer to the online help in Visual Studio .NET.)

By Crystal Enterprise report object:

You can bind the Crystal Enterprise report, MyCEReport.rpt, to the Web Forms Viewer through the Properties window or through code in WebForm1.cs:

Through the Properties window

1. Go to the **Design** tab of WebForm1.aspx. Select the Web Forms Viewer.
2. In the Properties window, click the square button on the right of the property **DataBindings**. This invokes the **CrystalReportViewer1 DataBindings** dialog box.
3. In the **CrystalReportViewer1 DataBindings** dialog box, select **ReportSource** on the left pane under **Bindable Properties**.
4. Check the **Simple Binding** radio button. Expand the **Page** node and select **enterpriseReport1** from the tree. Click **OK**.
5. In WebForm1.aspx.cs, after the call to `base.OnInit(e)` add the following line:

   ```
   DataBind();
   ```

Through the Code view

1. Add reference to CrystalDecisions.ReportSource and CrystalDecisions.Shared to the project.
2. Bind the Crystal Enterprise report object to the Web Forms Viewer by specifying the ReportSource property through the Code view for WebForm1.cs, in `OnInit()` after the call to `base.OnInit(e)`:

   ```
   CrystalReportViewer1.ReportSource = enterpriseReport1;
   ```

Scenario H – Server File Report Added to Web Form

Suppose the following conditions exist:

- On the client side, you have created a Visual C# Web project, MyProject.

- You have inserted a Web Forms Viewer in the Web Form, WebForm1.cs, in MyProject.
- On the client, you have added a Server File report, MyServerFileReport.rpt, to the Web Form (by dragging and dropping the report from the Server Explorer to the Design tab of WebForm1.cs, or by right-clicking MyServerFileReport.rpt in the Server Explorer and selecting "Add to Designer". For more information on Server File reports, please refer to the online help in Visual Studio .NET.)

By Server File report object:

You can bind the Server File report, MyServerFileReport.rpt, to the Web Forms Viewer through the Properties window or through code in WebForm1.cs:

Through the Properties window

1. Go to the **Design** tab of WebForm1.aspx. Select the Web Forms Viewer.
2. In the Properties window, click the square button on the right of the property **DataBindings**. This invokes the **CrystalReportViewer1 DataBindings** dialog box.
3. In the **CrystalReportViewer1 DataBindings** dialog box, select **ReportSource** on the left pane under **Bindable Properties**.
4. Check the **Simple Binding** radio button. Expand the **Page** node and select **serverFileReport1** from the tree. Click **OK**.
5. In WebForm1.aspx.cs, after the call to `base.OnInit(e)`, add the following line:

   ```
   DataBind();
   ```

Through the Code view

1. Add reference to CrystalDecisions.ReportSource and CrystalDecisions.Shared to the project.
2. Bind the Server File report object to the Web Forms Viewer by specifying the ReportSource property through the Code view for WebForm1.cs, in `Page_Init()` after the call to `base.OnInit(e)`:

   ```
   CrystalReportViewer1.ReportSource = serverFileReport1;
   ```

Bibliography

Microsoft Corporation. Visual Studio Enables the Programmable Web: Web Services and XML (URL). February 25, 2000; <URL:http://msdn.microsoft.com/vstudio/nextgen/technology/webservices.asp>.[Accessed May 11, 2000]