SSRS REPORT

Table of contents

SQL Server 2012 Reporting Services

SQL Server 2012 : Reporting Services

Create a Data-Driven Subscription

Reporting Services provides data-driven subscriptions so that you can customize the
distribution of a report based on dynamic list of subscribers that will receive the report. Data-driven subscriptions are typically created and maintained by report server administrators. The ability to create data-driven subscriptions requires expertise in building queries, knowledge of data sources that contain subscriber data, and elevated permissions on a report server.

Creating a Report Server Project

CREATING A REPORT SERVER PROJECT(REPORTING SERVICES):

To create a report in SQL Server, you must first create a report server project where you will save your report definition (.rdl) file and any other resource files that you need for your report. Then you will create the actual report definition file, define a **DATA SOURCE** for your report, define a **DATASET**, and define the **REPORT LAYOUT**. When you run the report, the actual data is retrieved and combined with the layout, and then rendered on your screen,from where you can export it, print it, or save it.

A report server project is used to create reports that run on a report server.

PROCEDURES:

To create a report server project:

1.Click **Start**, point to **All Programs**, point to , and then click **SQL Server Data Tools**. If this is the first time you have
opened SQL Server Data Tools, click **Business Intelligence Settings** for the default environment settings.

2. On the **File** menu, point to **New**, and then click **Project**.
3. In the **Installed Templates** list, click **Business Intelligence**.
4. Click **Report Server Project**.
5. In **Name**, type example:**Tutorial**(your project name).

6. Click **OK** to create the project.

To create a new report definition file:

1. In Solution Explorer, right-click Reports, point to Add, and click New Item.

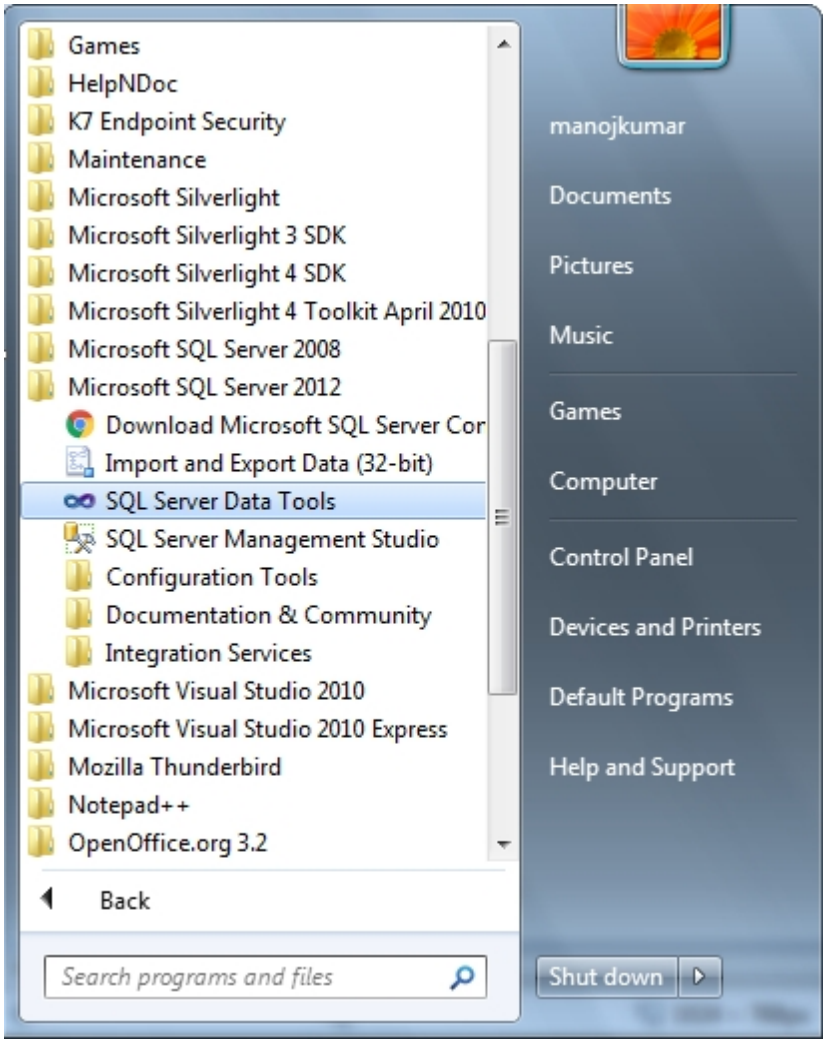
Note: If the **Solution Explorer** window is not visible, from the **View** menu, click **Solution Explorer**.

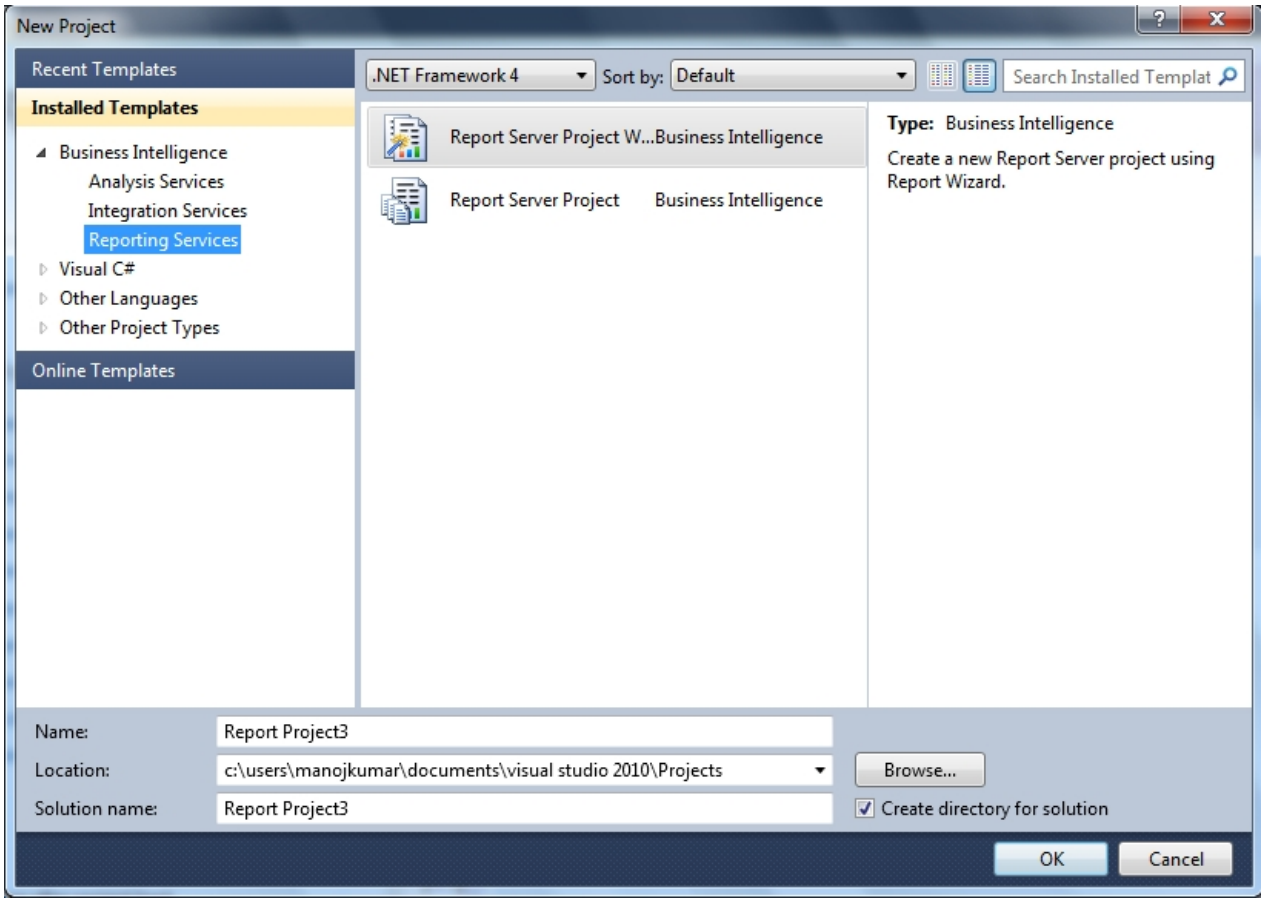
2. In the **Add New Item** dialog box, under **Templates**, click **Report**.
3. In **Name**, type **Sales Orders.rdl** and then click **Add**.

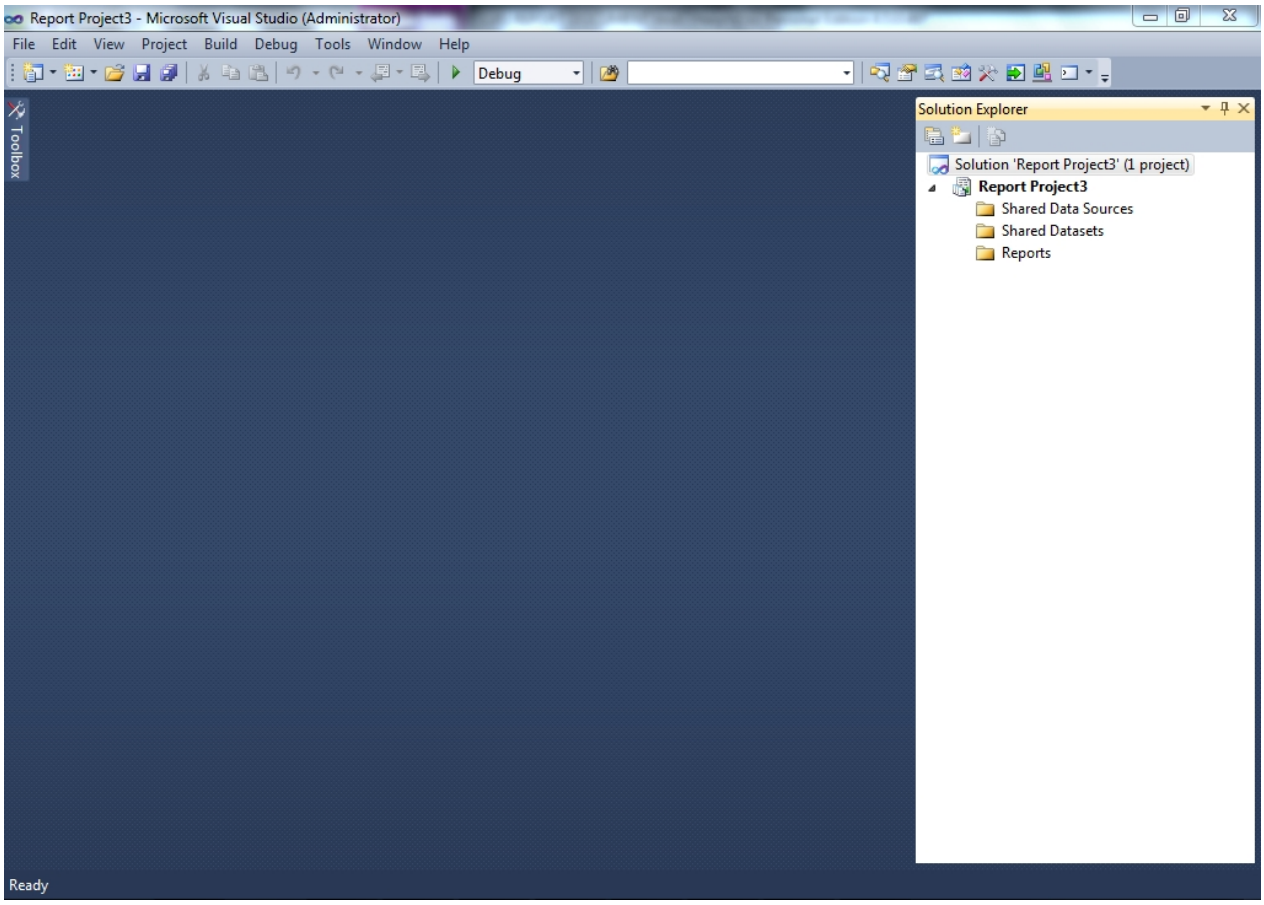
Report Designer opens and displays the new .rdl file in Design view. Report Designer is a Reporting Services component that runs in SQL Server Data Tools (SSDT). It has two views: **Design** and **Preview**. Click each tab to change views. You define your data in the **Report Data** pane. You define your report layout in **Design** view. You can run the report and see what it looks like in **Preview** view.

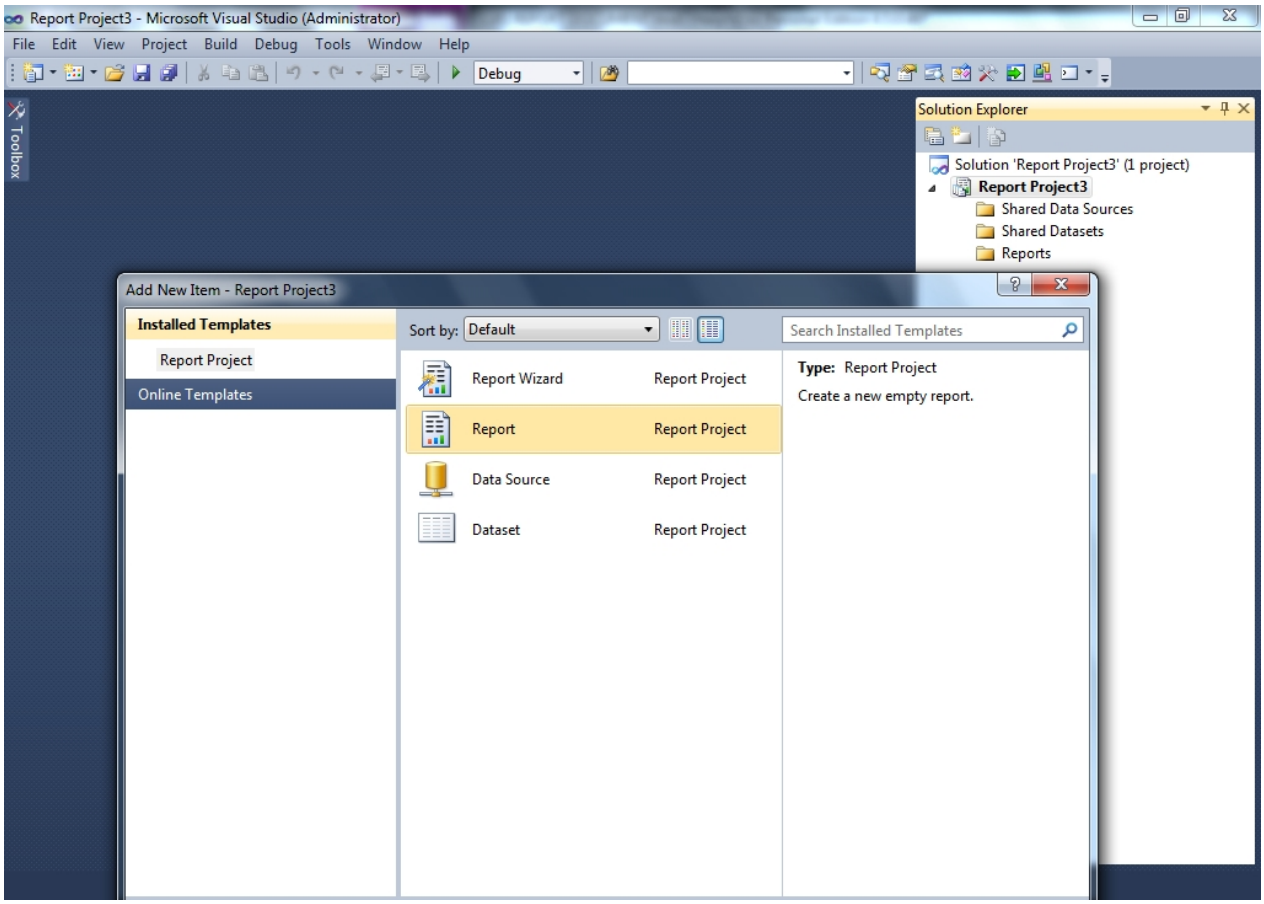
Next Task

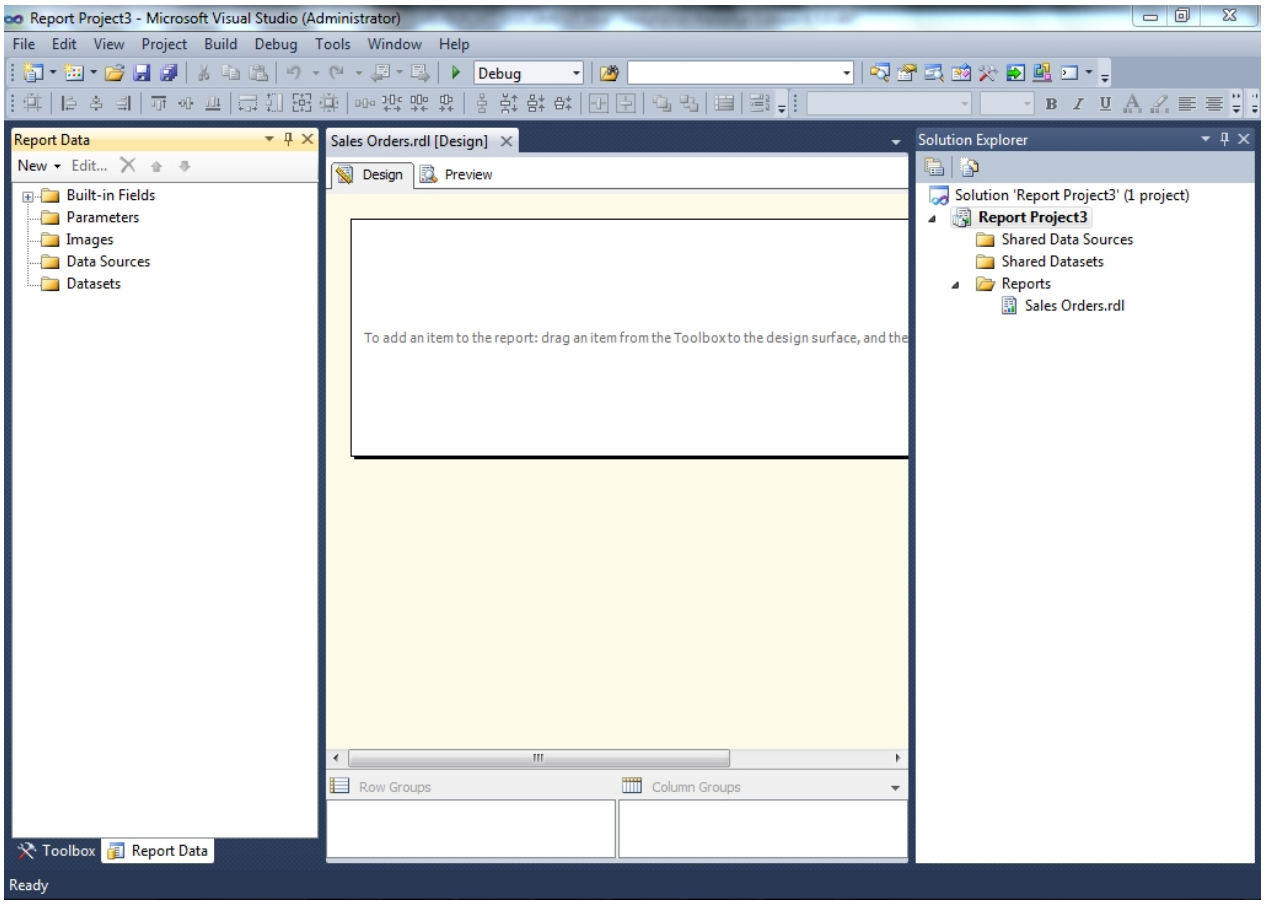
You have successfully created a report project called "Tutorial" and added a report definition (.rdl) file to the report project. Next, you will specify a data source to use for the report.




New Project
Recent Templates
Installed Templates
Business Intelligence
Analysis Services
Integration Services
Reporting Services
Visual C#
Other Languages
Other Project Types
Online Templates
.NET Framework 4
Sort by:
Default
Search Installed Templat
Report Server Project W...
Business Intelligence
Report Server Project
Business Intelligence
Type: Business Intelligence
Create a new Report Server project using Report Wizard.
Name:
Report Project3
Location:
c:\users\manojkumar\documents\visual studio 2010\Projects
Browse...
Solution name:
Report Project3
Create directory for solution
OK
Cancel


Report Project3 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Tools Window Help
Debug
Toolbox
Solution Explorer
Solution 'Report Project3' (1 project)
Report Project3
Shared Data Sources
Shared Datasets
Reports
Ready


Report Project3 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Tools Window Help
Debug
Toolbox
Solution Explorer
Solution 'Report Project3' (1 project)
Report Project3
Shared Data Sources
Shared Datasets
Reports
Add New Item - Report Project3
Installed Templates
Report Project
Online Templates
Sort by: Default
Search Installed Templates
Report Wizard Report Project
Report Report Project
Data Source Report Project
Dataset Report Project
Type: Report Project
Create a new empty report.


Report Project3 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Tools Window Help
Debug
Report Data
New Edit...
Built-in Fields
Parameters
Images
Data Sources
Datasets
Sales Orders.rdl [Design]
Design Preview
To add an item to the report: drag an item from the Toolbox to the design surface, and the
Row Groups
Column Groups
Solution Explorer
Solution 'Report Project3' (1 project)
Report Project3
Shared Data Sources
Shared Datasets
Reports
Sales Orders.rdl
Toolbox Report Data
Ready

Specifying Connection Information

SPECIFYING CONNECTION INFORMATION(REPORTING SERVICES):

After you add a report to the Tutorial project, you need to define a **DATA SOURCE**, which is connection information the report uses to access data from either a relational database, multidimensional database, or other resource.

Procedures:

To set up a connection*:*

1. In the **Report Data** pane, click **New** and then click **Data Source…**.

Note:

If the **Report Data** pane is not visible, from the **View** menu, click **Report Data**.

2. In **Name**, type .
3. Make sure **Embedded connection** is selected.
4. In **Type**, select **Microsoft SQL Server**.
5. In **Connection string**, type the following:

Data source=localhost; initial catalog=AdventureWorks2012

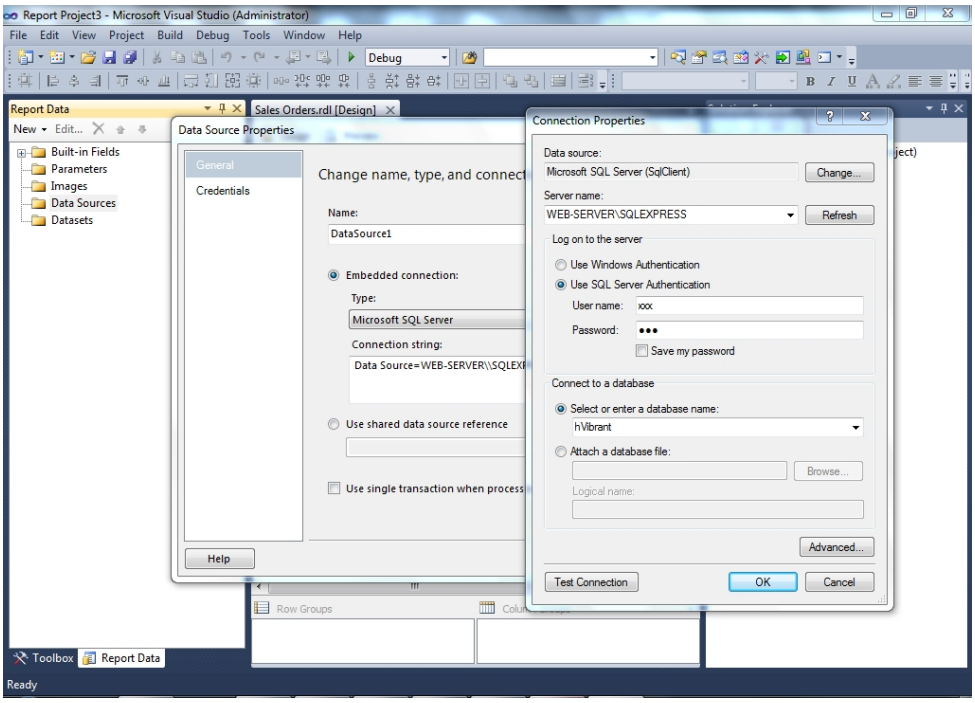
e.g:Data Source=WEB-SERVER\\SQLEXPRESS;Initial Catalog=hVibrant

This connection string assumes that SQL Server Data Tools (SSDT), the report server, and the database are all installed on the local computer and that you have permission to log on to the database.

Note:

If you are using SQL Server Express with Advanced Services or a named instance, the connection string must include instance information:

6. Click **Credentials** in the left pane and click **Use Windows Authentication (integrated security)**.



You have successfully defined a connection to the sample database. Next, you will create the report.

Defining a Data-set for the Table Report

DEFINING A DATA-SET FOR THE TABLE REPORT(REPORTING SERVICES):

After you define the data source, you need to define a data-set. In Reporting Services, data that you use in reports is contained in a data-set. A data-set includes a pointer to a data source and a query to be used by the report, as well as calculated fields and variables. You can use the query designer in Report Designer to design the query.

Procedures:

To define a Transact-SQL query for report data

*1. In the **Report Data** pane, click **New**, and then click **Dataset…**. The **Dataset** **Properties** dialog box opens.*

2. In the **Name** box, type <**Datasetname>**.
3. Click **Use a dataset embedded in my report**.
4. Make sure the name of your data source, AdventureWorks2012, is in the **Data**

source text box, and that the **Query type** is **Text**.

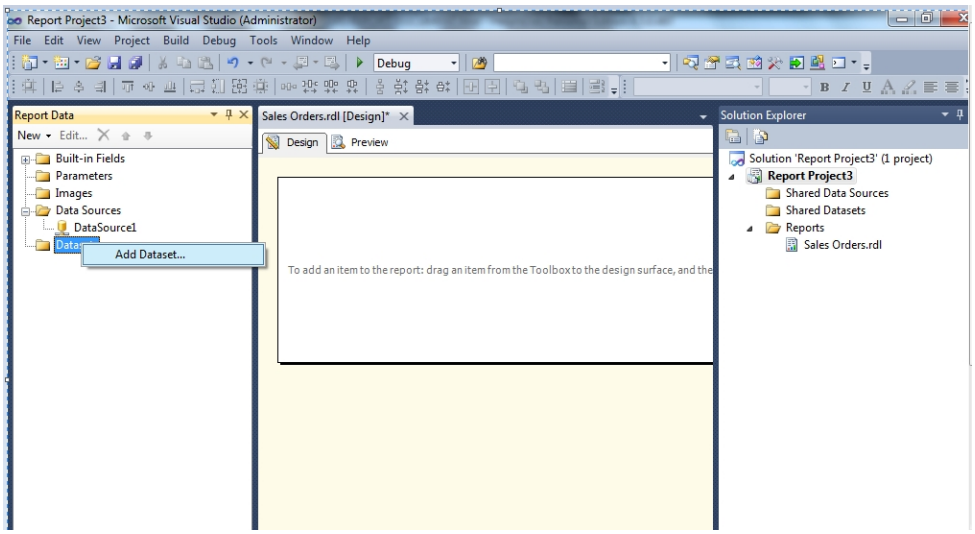
5. Type, or copy and paste, the following Transact-SQL query into the **Query** box.

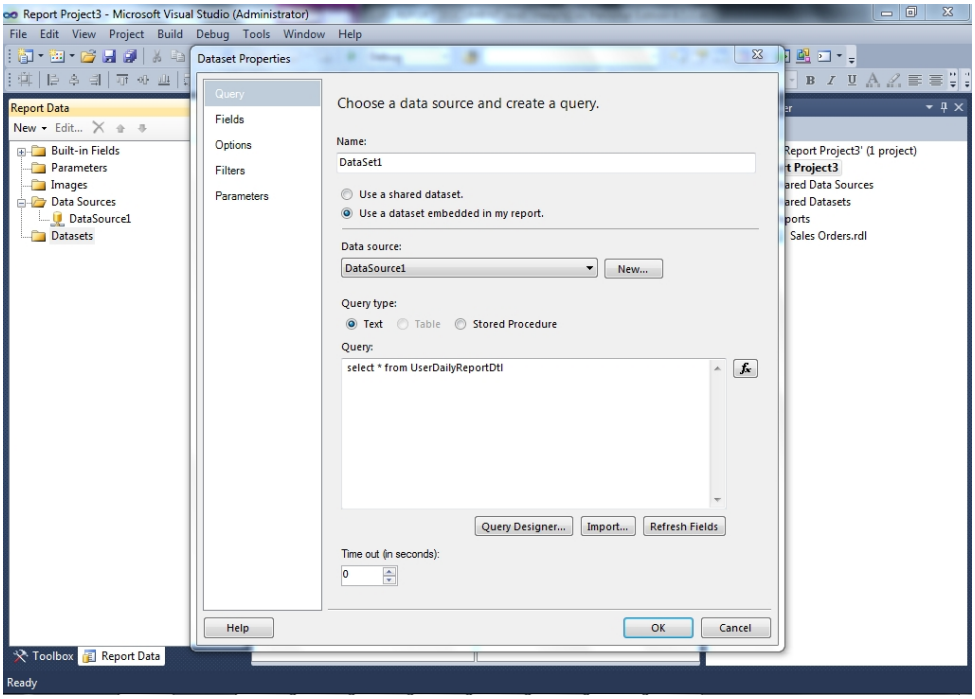
Select * from tablename.

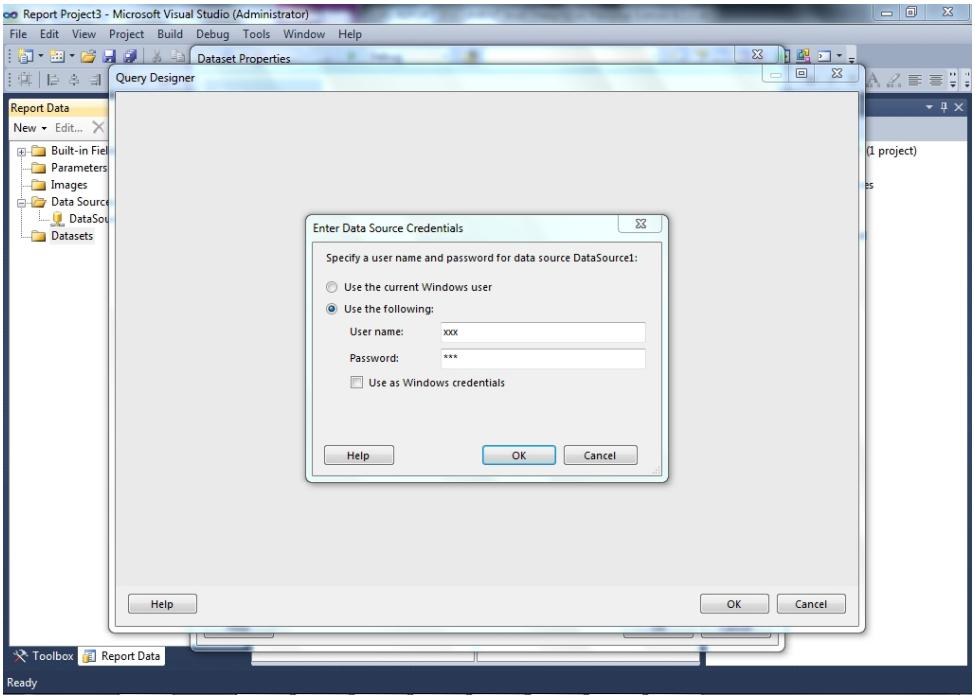
6. (Optional) Click the **Query Designer** button. The query is displayed in

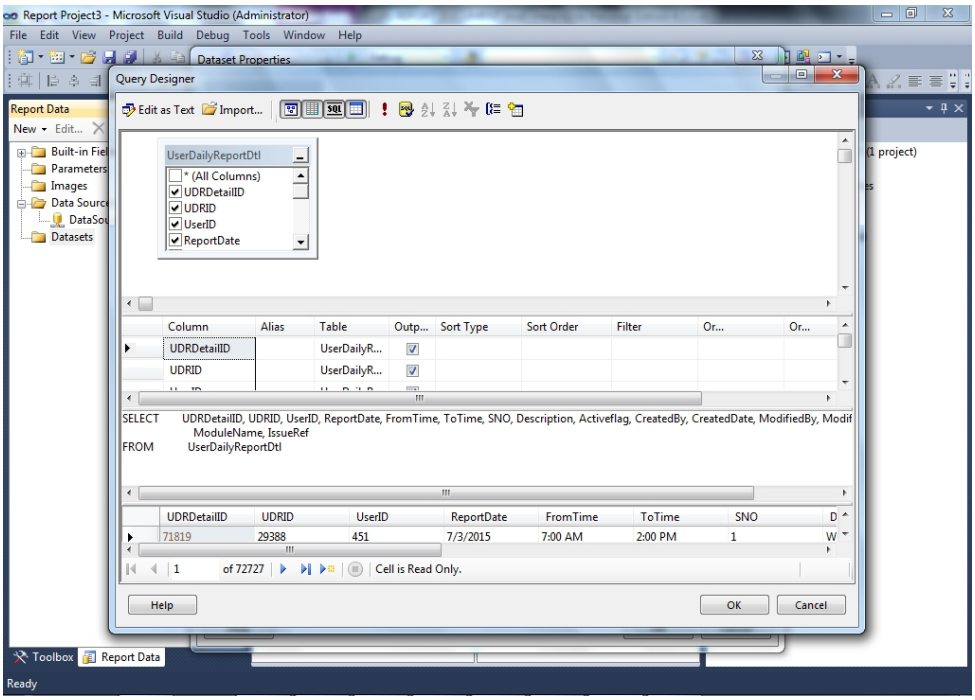
the text-based query designer. You can toggle to the graphical query designer by clicking **Edit As Text**. View the results of the query by clicking the run **(!)** button on the query designer toolbar. You see the data from six fields from four different tables in the database. The query makes use of Transact-SQL functionality such as aliases. For example, the SalesOrderHeader table is called soh. Click **OK** to exit the query designer.

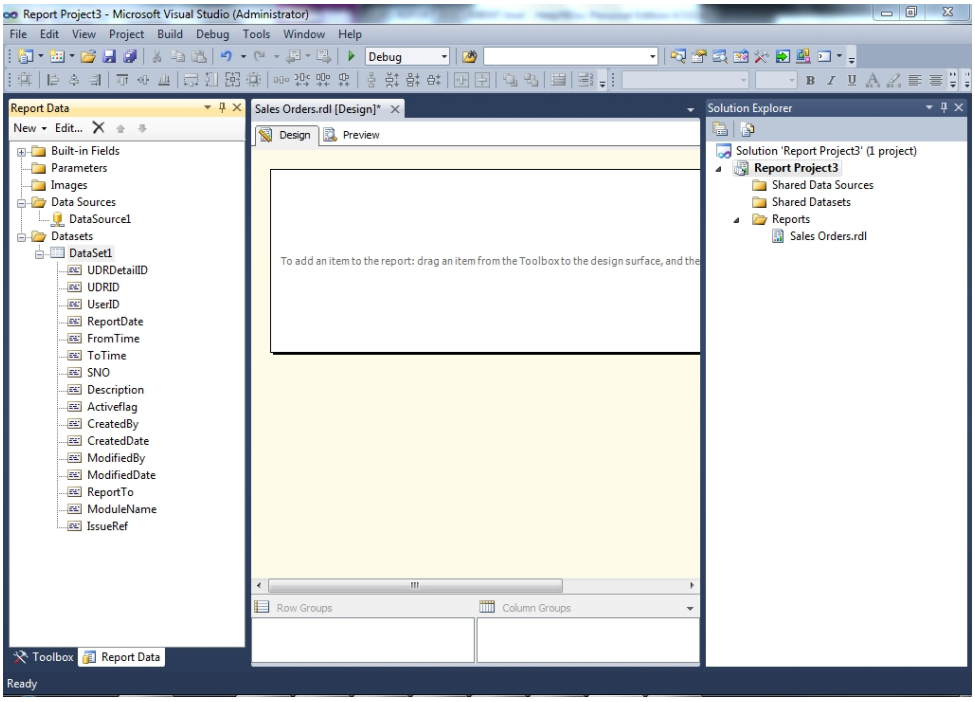
7. Click **OK** to exit the **Dataset Properties** dialog box. Your <Datasetname> **data-set** and fields appear in the Report Data pane.






Report Project3 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Tools Window Help
Dataset Properties
Query Designer
Report Data
New Edit...
Built-in Fiel
Parameters
Images
Data Source
DataSou
Datasets
Enter Data Source Credentials
Specify a user name and password for data source DataSource1:
Use the current Windows user
Use the following:
User name:
xxx
Password:
***
Use as Windows credentials
Help
OK
Cancel
Help
OK
Cancel
Toolbox
Report Data
Ready


Report Project3 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Tools Window Help
Dataset Properties
Query Designer
Edit as Text
Import...
Report Data
New Edit...
Built-in Fiel
Parameters
Images
Data Source
DataSou
Datasets
UserDailyReportDtl
* (All Columns)
UDRDetailID
UDRID
UserID
ReportDate
Column
Alias
Table
Outp...
Sort Type
Sort Order
Filter
Or...
Or...
UDRDetailID
UserDailyR...
UDRID
UserDailyR...
SELECT UDRDetailID, UDRID, UserID, ReportDate, FromTime, ToTime, SNO, Description, Activeflag, CreatedBy, CreatedDate, ModifiedBy, Modif
ModuleName, IssueRef
FROM UserDailyReportDtl
UDRDetailID
UDRID
UserID
ReportDate
FromTime
ToTime
SNO
71819
29388
451
7/3/2015
7:00 AM
2:00 PM
1
1
of 72727
Cell is Read Only.
Help
OK
Cancel
Toolbox
Report Data
Ready



You have successfully specified a query that retrieves data for your report. Next, you will create the report layout.

Adding a Table to the Report

Adding a Table to the Report (Reporting Services):

After the dataset is defined, you can start designing the report. You create a report layout by dragging and dropping data regions, text boxes, images, and other items that you want to include in your report to the design surface.

Items that contain repeated rows of data from underlying datasets are called *data regions*. A basic report will have only one data region, but you can add more, for example, if you want to add a chart to your tabular report. After you add a data region, you can add fields to the data region.

Procedures:

To add a Table data region and fields to a report layout

1. In the **Toolbox**, click **Table**, and then click on the design surface and drag the mouse. Report Designer draws a table data region with three columns in the center of the design surface.

Note:

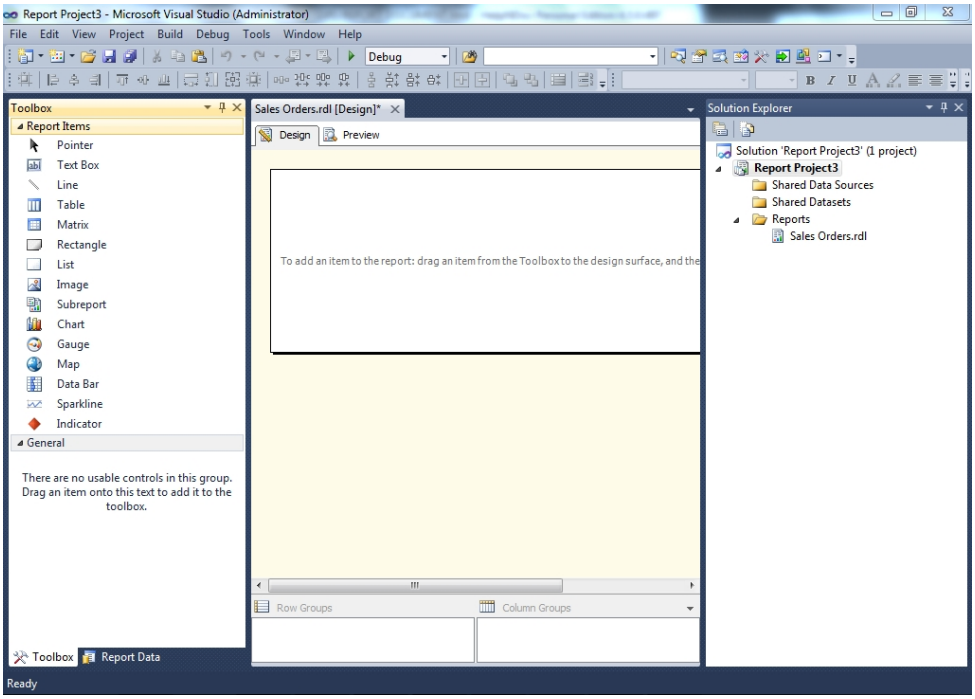
The **Toolbox** may appear as a tab on the left side of the **Report Data** pane. To open the **Toolbox**, move the pointer over the **Toolbox** tab. If the **Toolbox** is not visible, from the **View** menu, click **Toolbox**.

2. In the **Report Data** pane, expand the **<Datasetname>** dataset to display the fields.

3. Drag the Date field from the **Report Data** pane to the first column in the table. When you drop the field into the first column, two things happen. First, the data cell will display the field name, known as the *field expression*, in brackets: [Date]. Second, a column header value is automatically added to Header row, just above the field expression. By default, the column is the

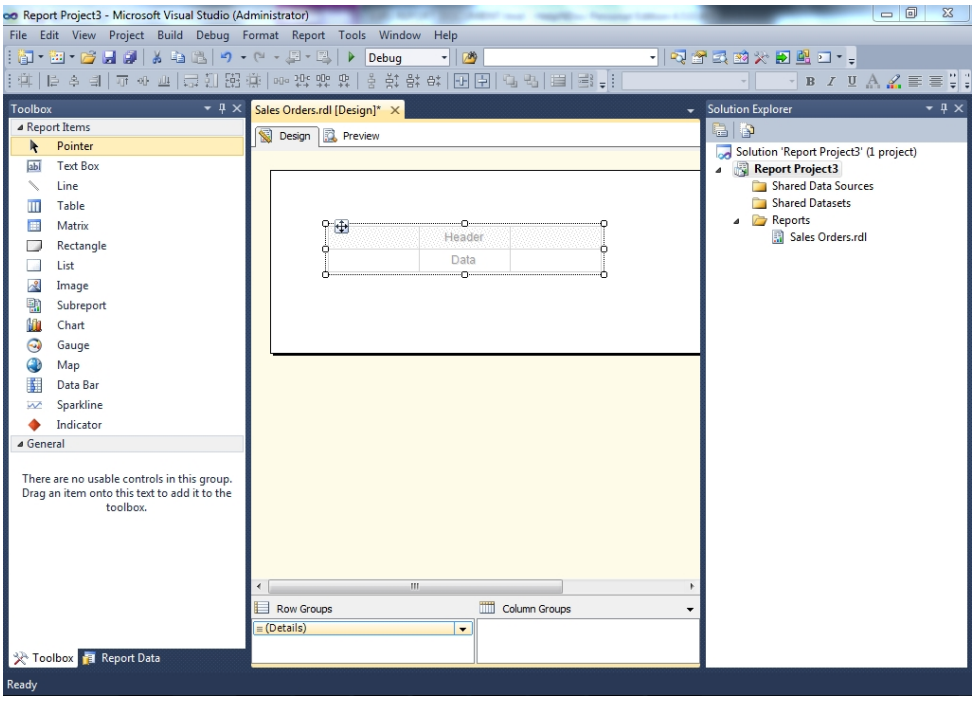
name of the field. You can
select the Header row text and type a new name.

4. Drag the Order field from the **Report Data** pane to the second column in the table.
5. Drag the Product field from the **Report Data** pane to the third column in the table.
6. Drag the Qty field to the right edge of the third column until you get a vertical
cursor and the mouse pointer has a plus sign [+]. When you release the mouse button, a fourth column is created for [Qty].
7. Add the LineTotal field in the same way, creating a fifth column.



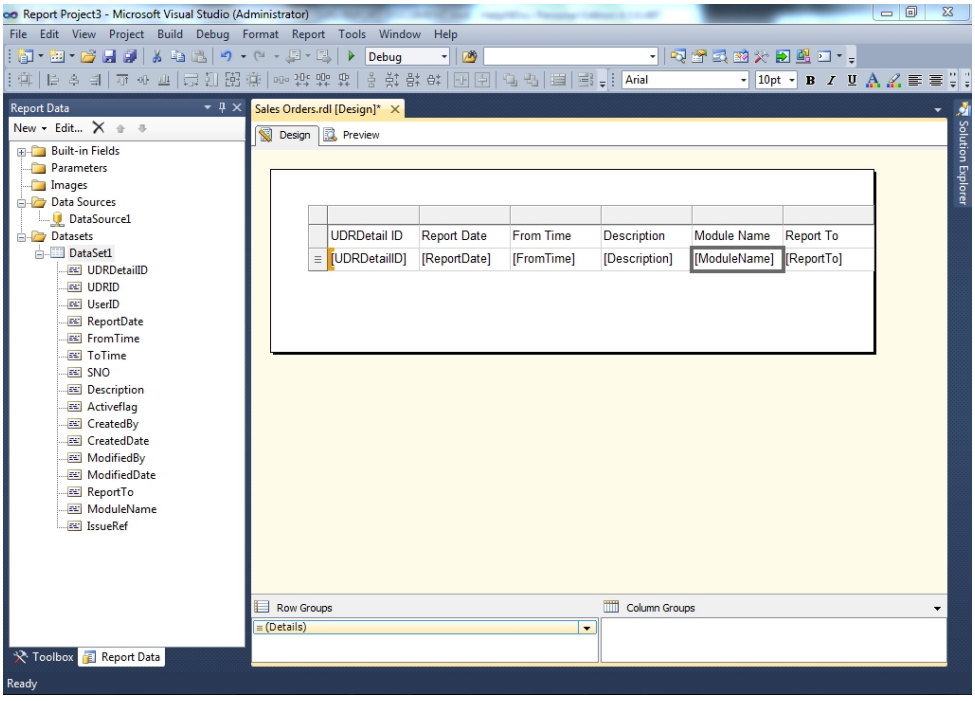
Note:

Click on the report items -->table. drag the table and drop in the Design.



Note:

The column header is Line Total. Report Designer automatically creates a friendly name for the column by splitting LineTotal into two words. The following diagram shows a table data region that has been populated with these fields:UDRDetail ID,Report Date,From Time,Description,Module Name,Report To



Preview Your Report

Preview Your Report:

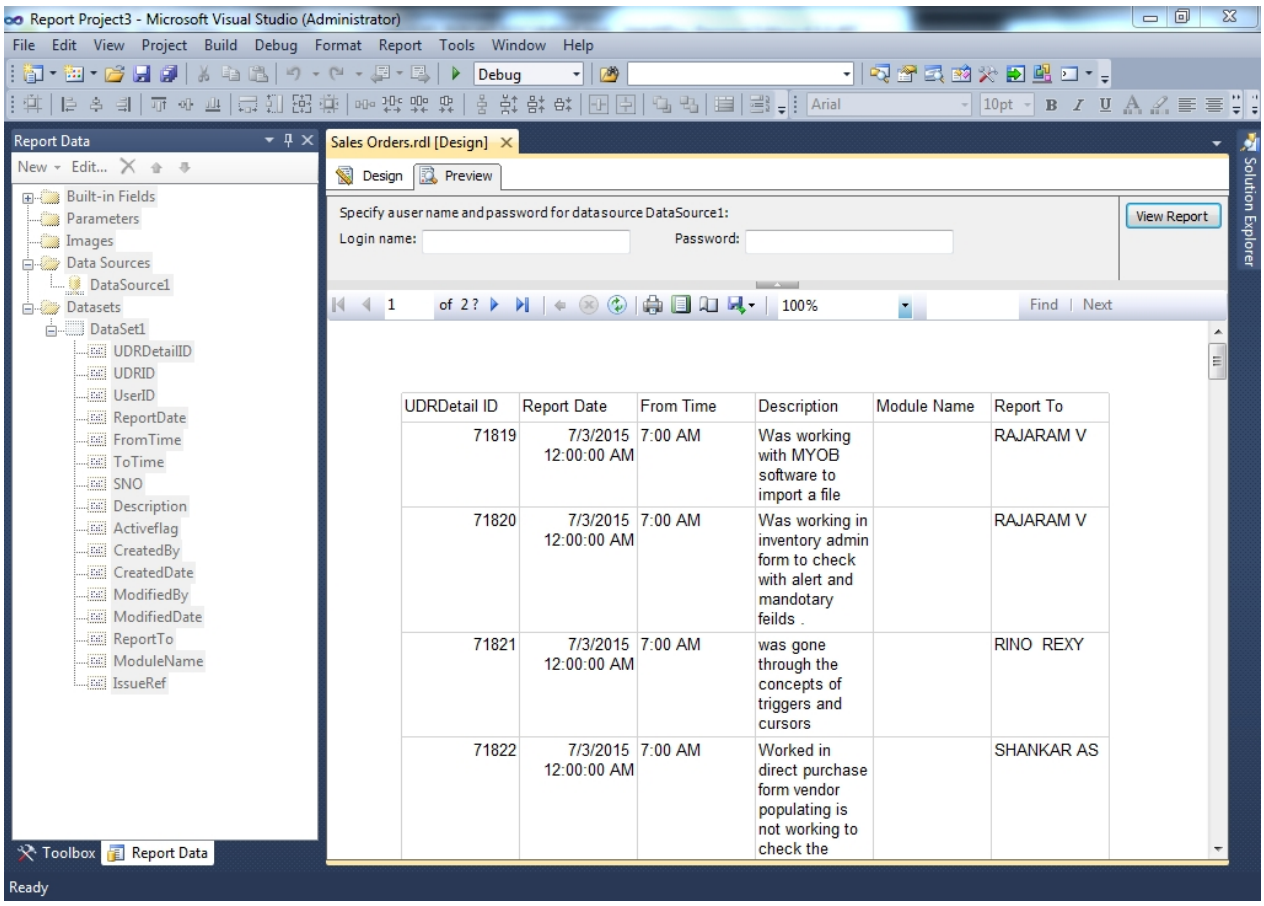
Previewing a report enables you to view the rendered report without having to first publish it to a report server. You will probably want to preview your report frequently during design time. Previewing the report will also run validation on the design and data connections so you can correct errors and issues before publishing the report to a report server.

To preview a report:

Click the Preview tab. Report Designer runs the report and displays it in Preview view.

Note:

On the **File** menu, click **Save All** to save the report.



Integrating ssrs reports in mvc application

INTRODUCTION:

SQL Server Reporting Services is rich and popular reporting solution that you have for free with SQL Server. It is widely used in the industry: from small family businesses running on SQL Server 2008/2012 express to huge corporations with SQL Server clusters.

There is one issue with the solution. Microsoft has **not release SSRS viewer for ASP.NET MVC yet**. That is why people usually mixing modern ASP.NET MVC enterprise applications with ASP.NET Web Forms pages to view report.

Installation

Installation:

1.Install Microsoft® System CLR Types for Microsoft® SQL Server® 2012 if needed.
Go to https://www.microsoft.com/en-us/download/details.aspx?id=29065 and scroll page to Microsoft® System CLR Types for Microsoft® SQL Server® 2012.

2.Install Microsoft Report Viewer 2012. The library has be deployed to developer machines and to servers.

3.Install MvcReportViewer package from NuGet.

Configuration:

1.Make sure you reference Microsoft.ReportViewer.WebForm (Microsoft.ReportViewer.WebForms, Version=11.0.0.0, Culture=neutral, PublicKeyToken=89845dcd8080cc91) and MvcReportViewer assemblies in the application.

2.Configure the ASP.NET Web Forms Report Viewer in the web.config.

3.Add <add path="Reserved.ReportViewerWebControl.axd" verb="*"

type="Microsoft.Reporting.WebForms.HttpHandler, Microsoft.ReportViewer.WebForms, Version=11.0.0.0, Culture=neutral, PublicKeyToken=89845dcd8080cc91" validate="false"/> to system.web/httpHandlers section.

4.Add <remove name="ReportViewerWebControlHandler" /> <add name="ReportViewerWebControlHandler"
preCondition="integratedMode" verb="*"
path="Reserved.ReportViewerWebControl.axd"
type="Microsoft.Reporting.WebForms.HttpHandler, Microsoft.ReportViewer.WebForms, Version=11.0.0.0, Culture=neutral, PublicKeyToken=89845dcd8080cc91"/> to system.webServer/handlers section.

5.Configure MvcReportViewer HTML helper in the web.config. There are two

ways of doing this. You can use application settings or MvcReportViewer configuration section.

WE USE APPLICATION SETTINGS

Application Settings

```
<!-- we use only these keys inside application setting -->
<add key="reportserverurl" value="http://<servername>/<reportserver>" />
<add key="reportpath1" value="/TrackerReport Project3"/>

TrackerReport project3 = type your report folder name. for folder ask your TL
or Report publish manager.


<!-- Required by Microsoft ReportViewer control -->
<add key="MvcReportViewer.AspxViewer" value="~/MvcReportViewer.aspx" />
<add key="MvcReportViewer.AspxViewerJavaScript" value="~/Scripts/
MvcReportViewer.js" />
<add key="MvcReportViewer.ErrorPage" value="~/MvcReportViewerErrorPage.html" />
<add key="MvcReportViewer.ShowErrorPage" value="False" />
<add key="MvcReportViewer.ReportServerUrl" value="http://localhost/
ReportServer_SQLEXPRESS" />
<add key="MvcReportViewer.User name" value="" />
<add key="MvcReportViewer.Password" value="" />
<add key="MvcReportViewer.EncryptParameters" value="True" />
<add key="MvcReportViewer.IsAzureSSRS" value="false" />
<add key="MvcReportViewer.LocalDataSourceProvider"
value="MvcReportViewer.SqlLocalDataSourceProvider,MvcReportViewer" />
<add key="SqlLocalDataSourceProvider.ConnectionString" value="Products" />
```

Description

The ASP.NET MVC Component

Description:

1.Controller method to launch report

2.ASP.NET Web Form to host the Report Viewer control

That's right! In order to pull this solution off, you need to incorporate ASP.NET Web Forms. The good news is that ASP.NET MVC is based on Web Forms. Look at any ASP.NET MVC solution and you will find the System.Web namespace. Therefore, in a very real sense, you aren't adding anything new to an ASP.NET MVC solution.

With respect to the controller method used to launch the report, technically speaking, even that is not required. However, if at some point you wish to pass information to the SSRS context from the ASP.NET MVC context, then the controller method becomes required. On the other hand, if no such information passing requirement exists, then you are

free to just call the report URL itself. Going one step further, the ASP.NET Web Form to host the controller isn't 100% required. If all you want to do is launch the report, you can simply invoke the URL to launch the report in the browser

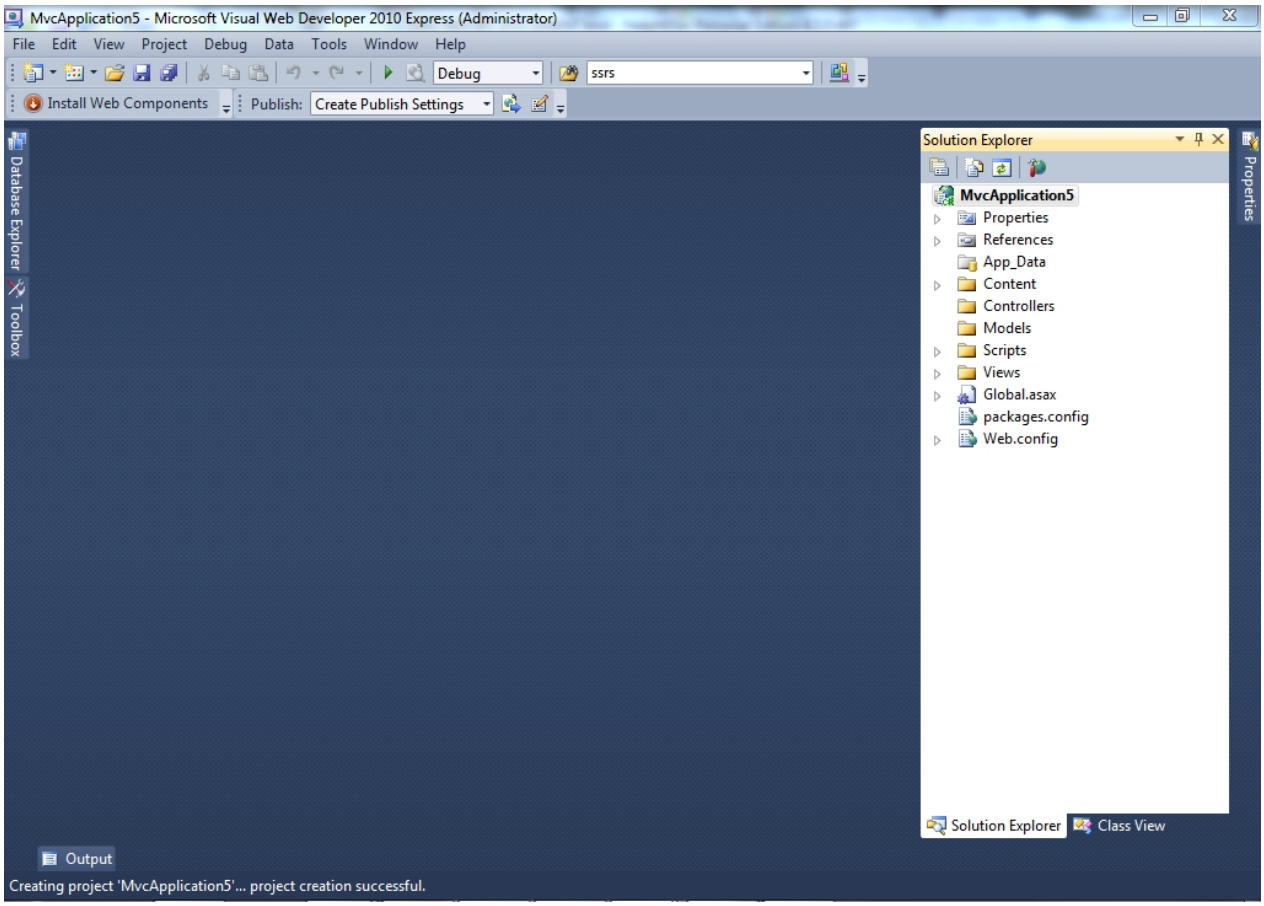
However, if you do need to pass information from one context to another, then just as you need the controller method to send the data, you will need the ASP.NET Web Form to receive the data. In order to set the stage for the second part of this article, I will go ahead and incorporate the ASP.NET Web Form.

This brings up an interesting question, if ASP.NET MVC Views are based on Web Forms, then why do you need to include a Web Form? The answer has to do with view state. If you look at the source of an ASP.NET Web Form, you will find a lot of code that is dedicated to maintaining state in a manner similar to that of a Windows Forms application. That, after all, was a main design goal of ASP.NET; to open the world of web development to Windows developers.

Adding a Web Form to the ASP.NET MVC Application

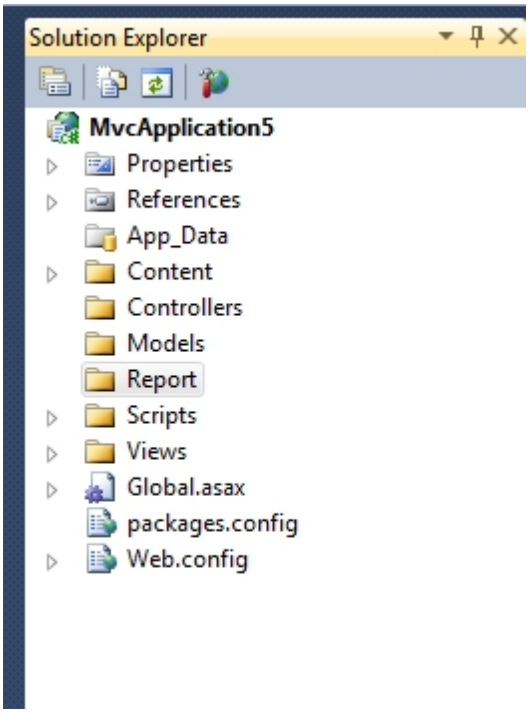
Step 1:

I'm going to create a sample mvc application 5



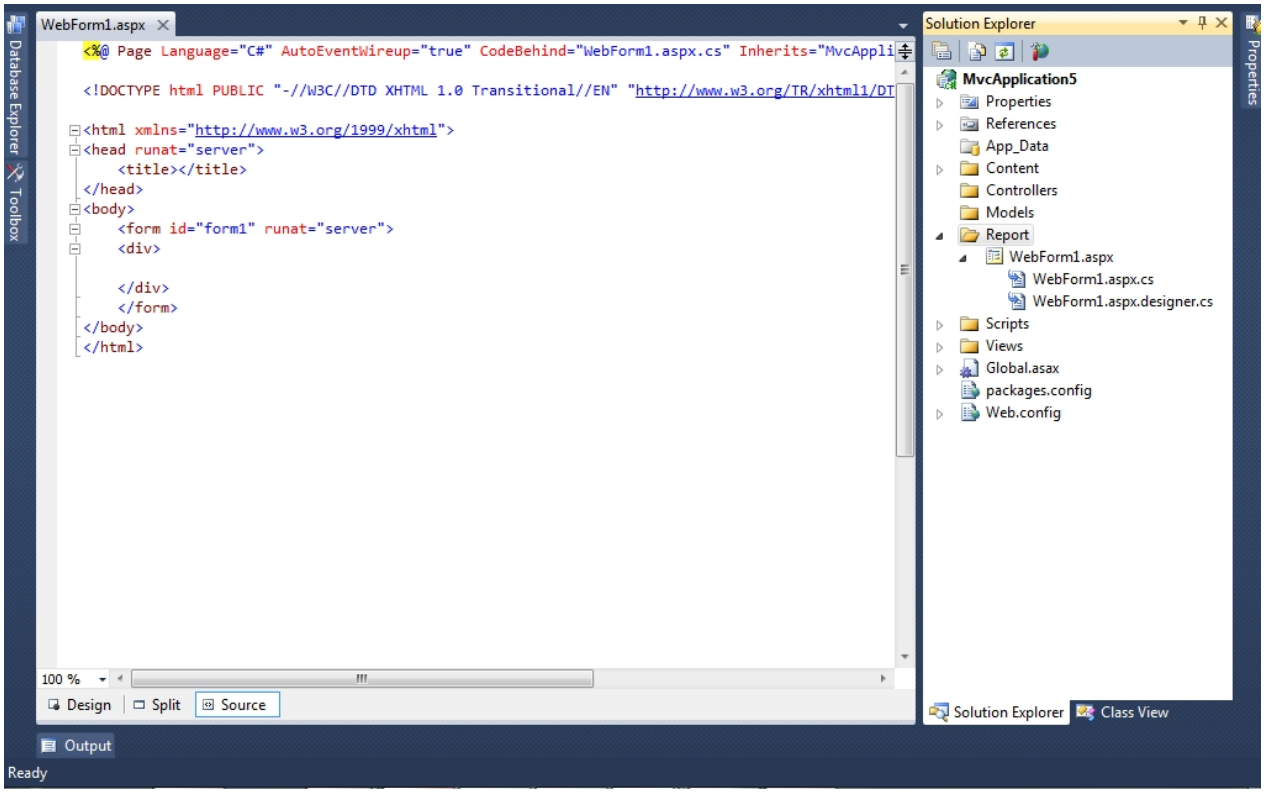
Step 2:

added a folder name as Report.



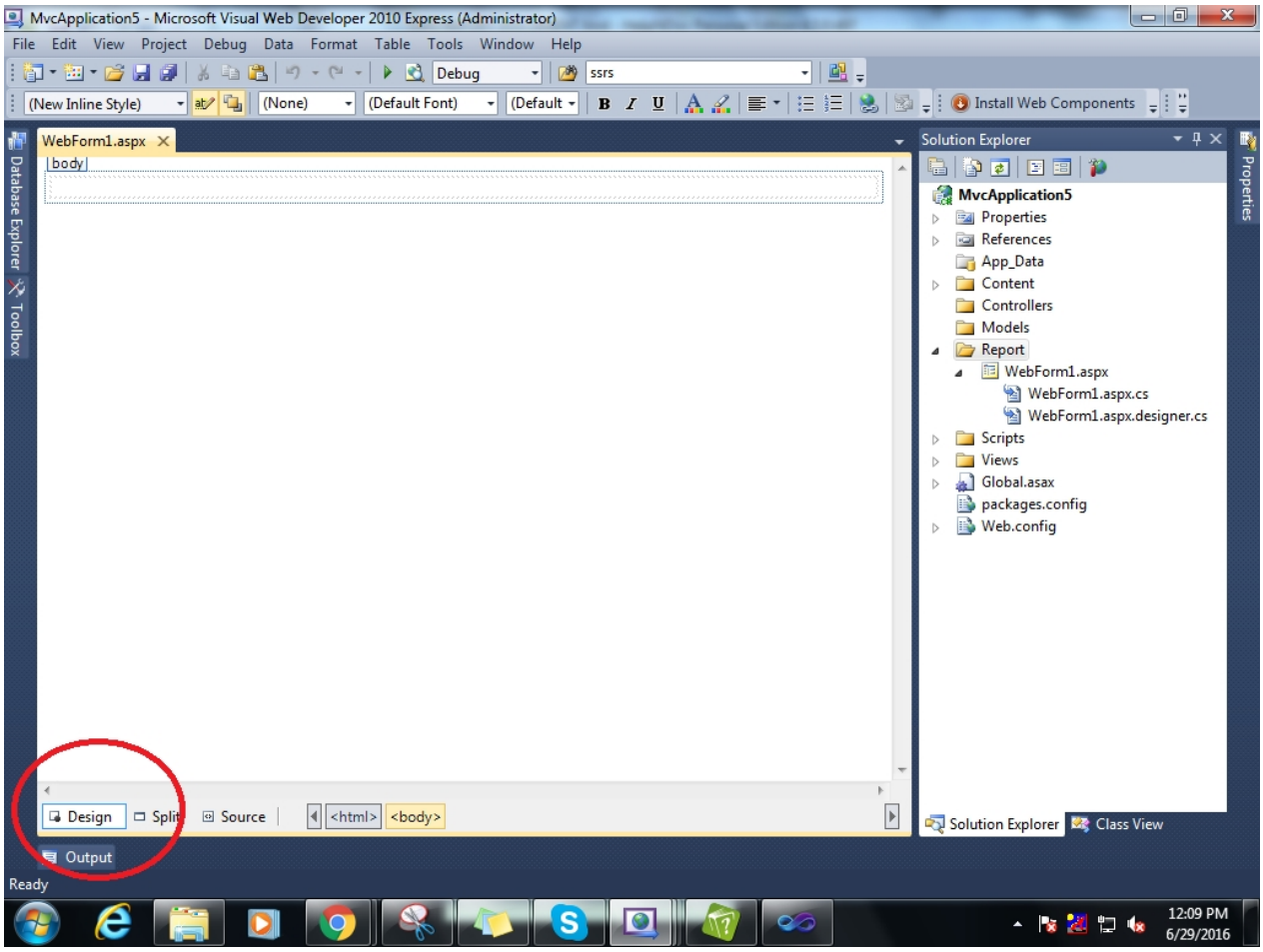
Step 3:

Added a web form inside a Report folder.



Step 4:

click the Design tab below normally it will be empty as shown below.

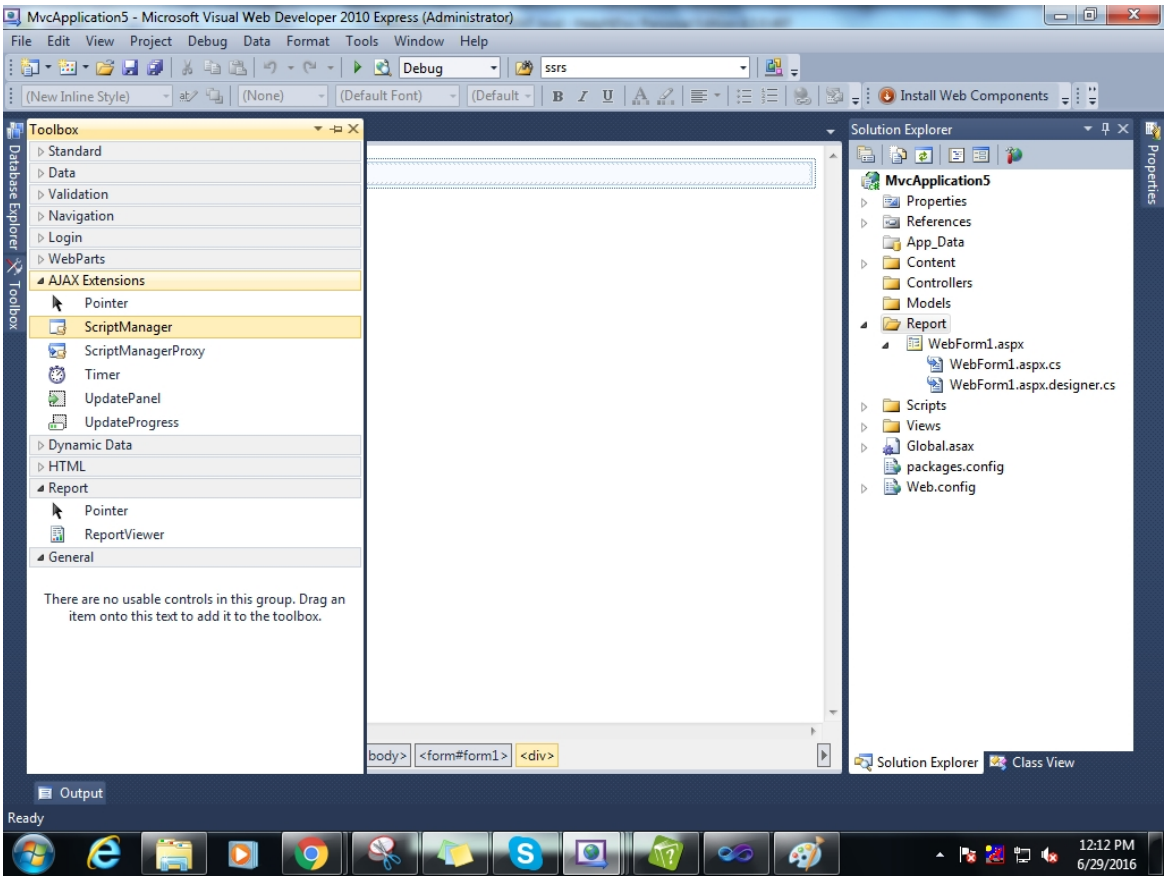


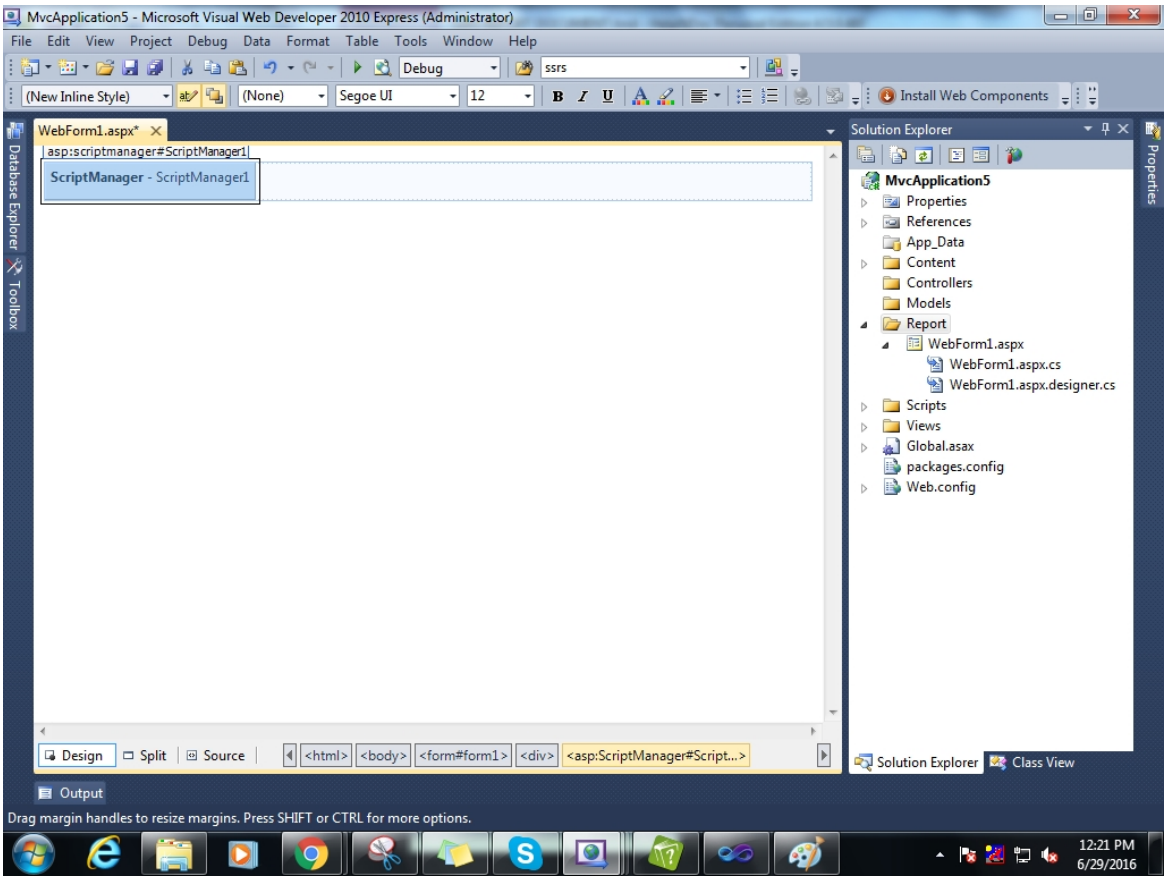
Step 4:

IMAGE 1

1.Click the toolbox -->AJAX Extensions-->script manager.

2.Drag and drop the script manager to design page as shown below.

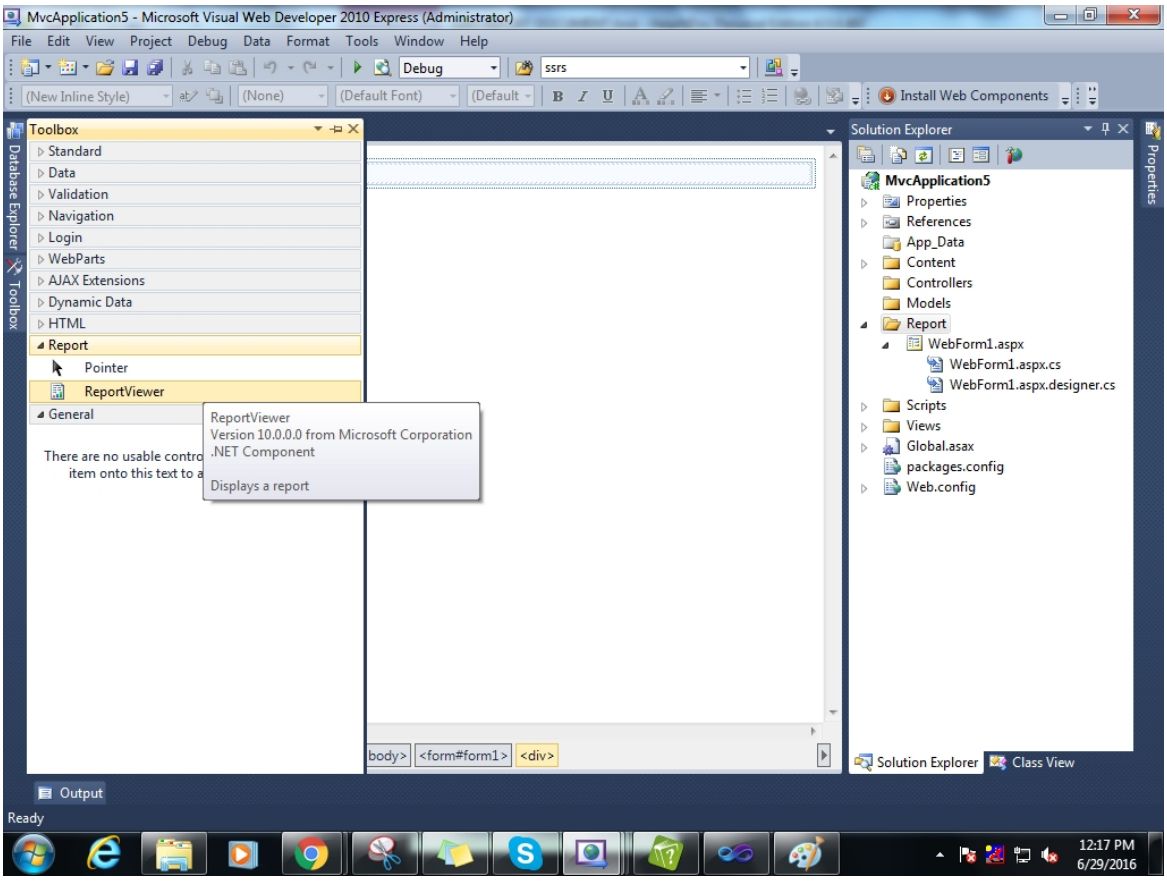

MvcApplication5 - Microsoft Visual Web Developer 2010 Express (Administrator)
File Edit View Project Debug Data Format Tools Window Help
Debug
ssrs
Install Web Components
Toolbox
Standard
Data
Validation
Navigation
Login
WebParts
AJAX Extensions
Pointer
ScriptManager
ScriptManagerProxy
Timer
UpdatePanel
UpdateProgress
Dynamic Data
HTML
Report
Pointer
ReportViewer
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
Solution Explorer
MvcApplication5
Properties
References
App_Data
Content
Controllers
Models
Report
WebForm1.aspx
WebForm1.aspx.cs
WebForm1.aspx.designer.cs
Scripts
Views
Global.asax
packages.config
Web.config
Solution Explorer
Class View
Output
Ready
12:12 PM
6/29/2016

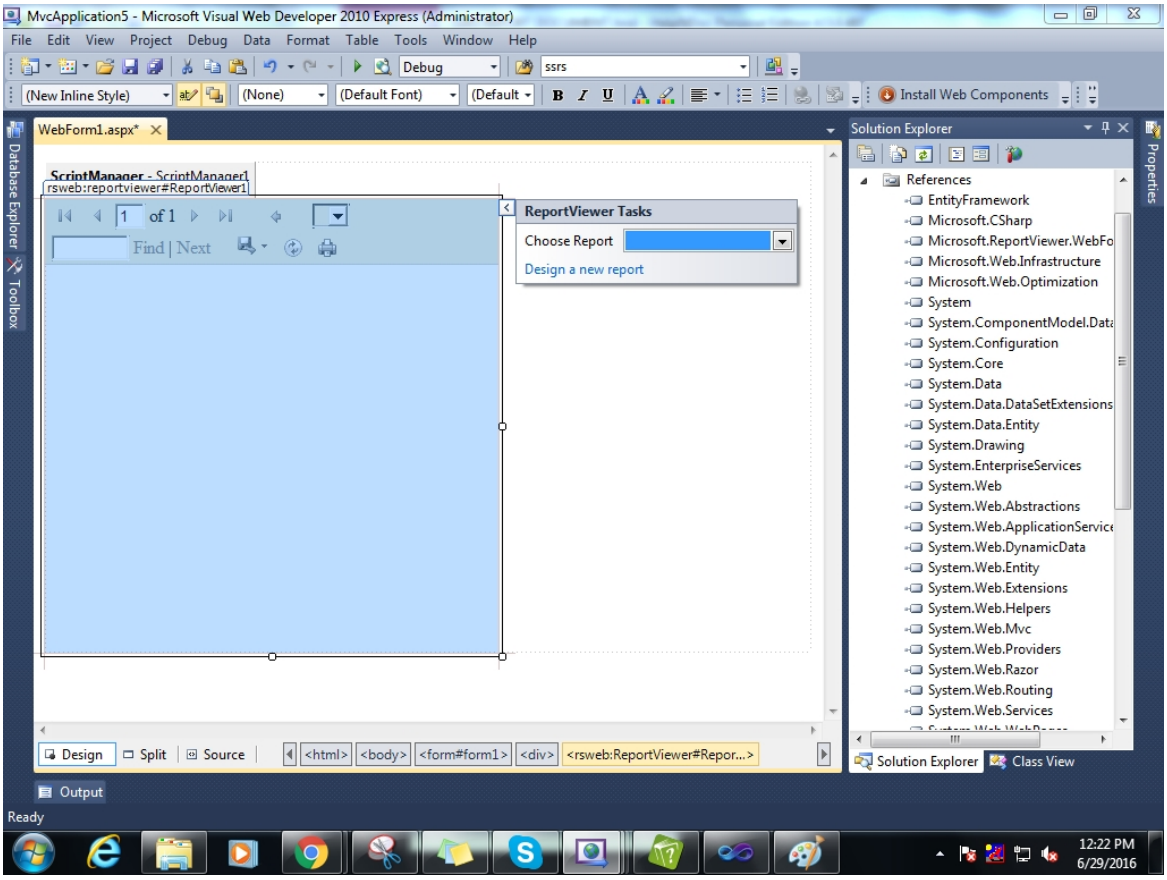

MvcApplication5 - Microsoft Visual Web Developer 2010 Express (Administrator)
File Edit View Project Debug Data Format Table Tools Window Help
Debug
ssrs
(New Inline Style)
(None)
Segoe UI
12
Install Web Components
WebForm1.aspx*
asp:scriptmanager#ScriptManager1
ScriptManager - ScriptManager1
Design
Split
Source
<html>
<body>
<form#form1>

<asp:ScriptManager#Script...>
Solution Explorer
MvcApplication5
Properties
References
App_Data
Content
Controllers
Models
Report
WebForm1.aspx
WebForm1.aspx.cs
WebForm1.aspx.designer.cs
Scripts
Views
Global.asax
packages.config
Web.config
Solution Explorer
Class View
Output
Drag margin handles to resize margins. Press SHIFT or CTRL for more options.
12:21 PM
6/29/2016

IMAGE 2

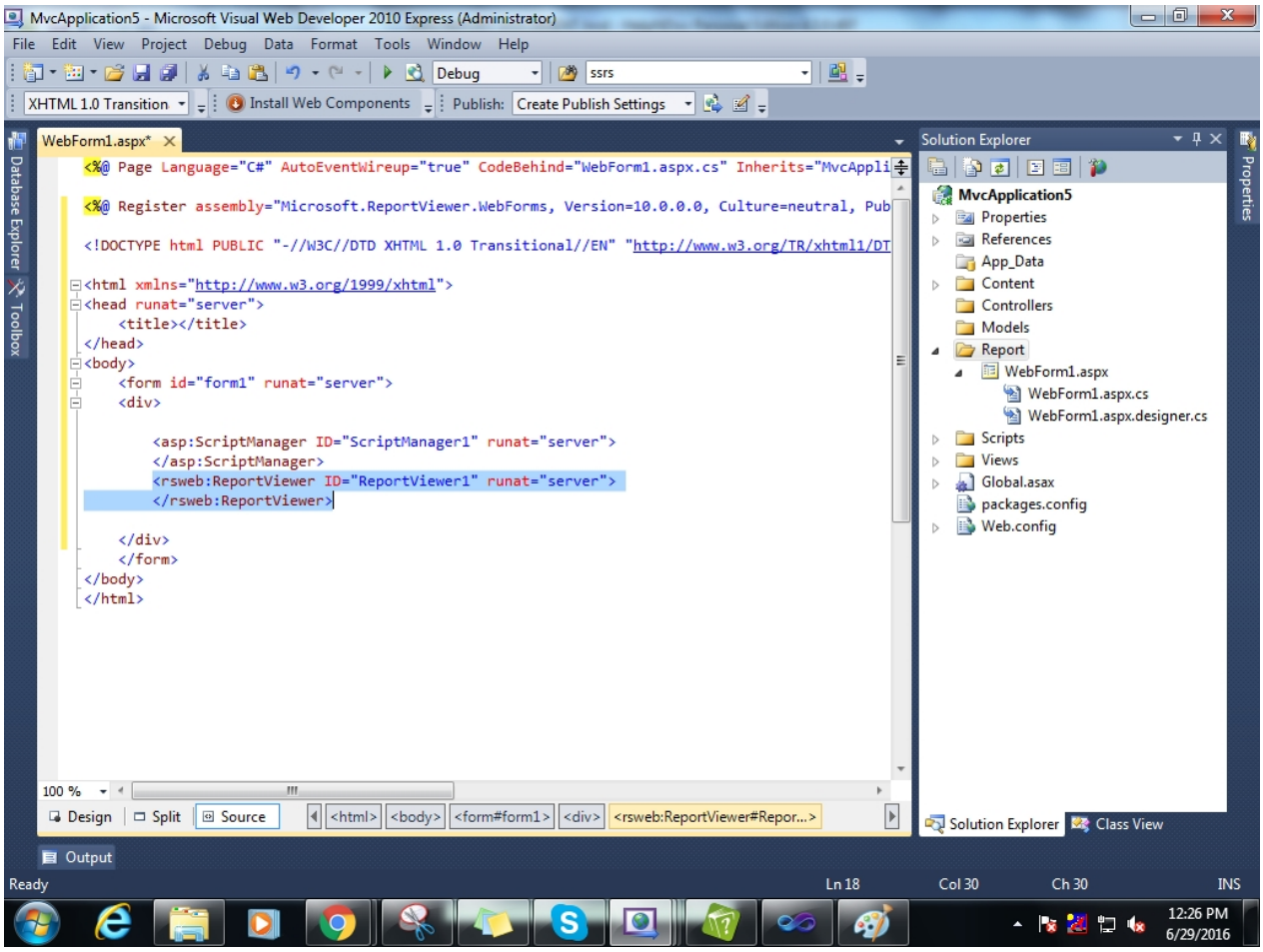
1.Click the toolbox-->Report-->Reportviewer.
2.Drag and drop the reportviewer in design page as shown below.





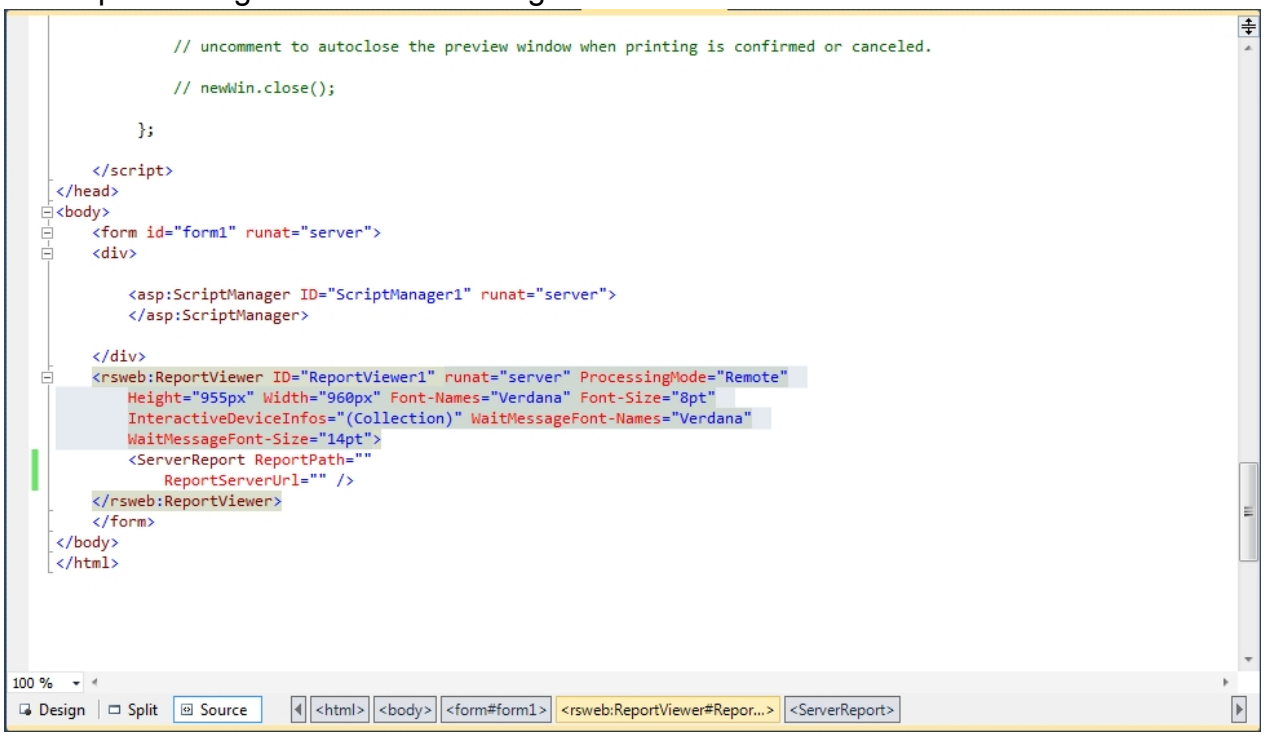
Step 5:

Your source page will looks like as shown below.



Step 6:

mention this in <rsweb:reportviewer> Tag processingmode="remote" Height="" and width="".

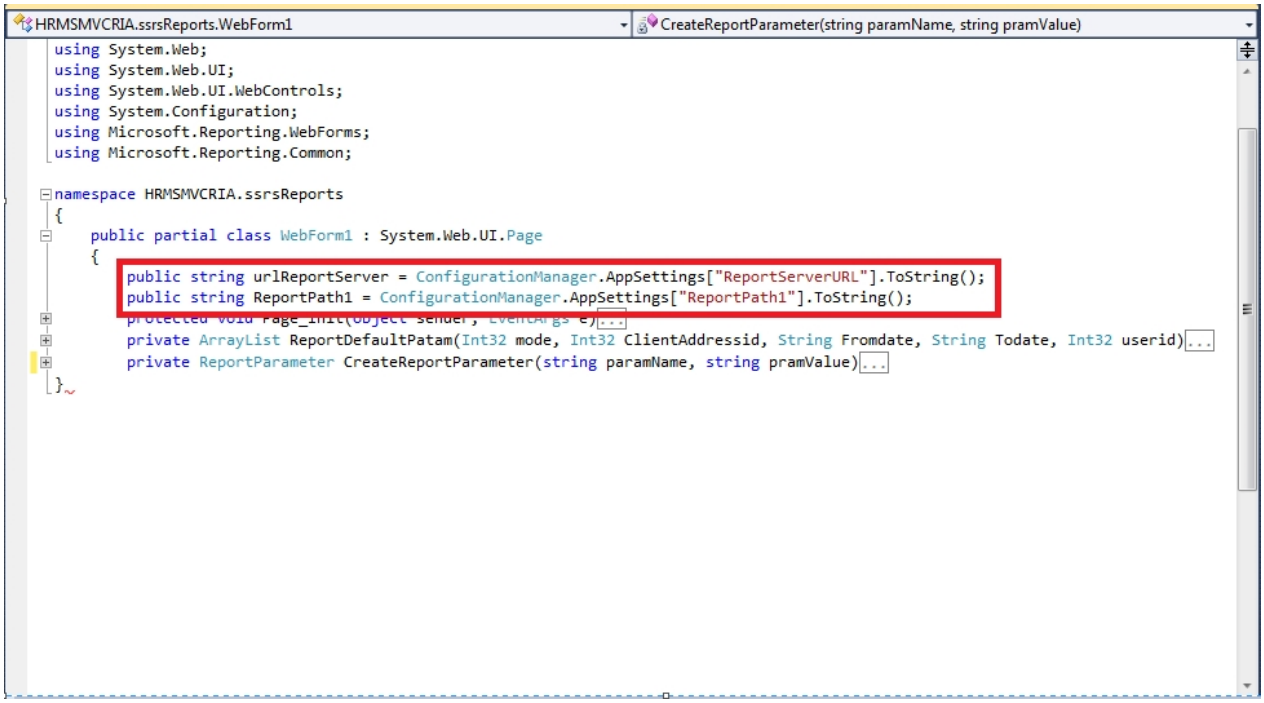


Step 7:

code in **aspx.cs** page

1.add the namespace `using Microsoft.Reporting.WebForms;` and `using Microsoft.Reporting.Common;`

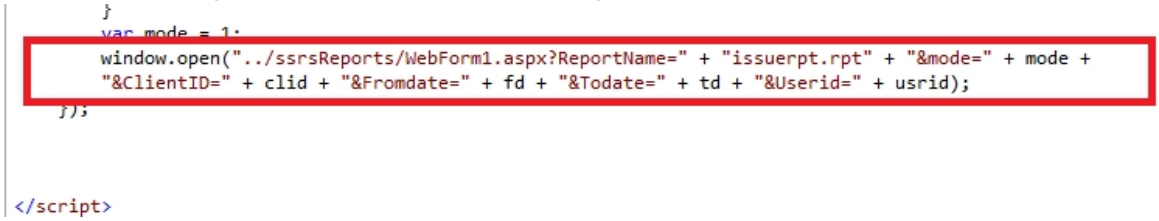
2.add urlreportserver and reportpath.



NOTE: Don't write a code inside page_load method. create a method name as page_init.

Step 8:

passing value to web form page.



Step 9:

1.we can pass the value to report from web form using two type of coding.

(i).using array[] or list.

(ii).Generally pass string value.

```
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        public string urlReportServer = ConfigurationManager.AppSettings["ReportServerURL"].ToString();
        public string ReportPath1 = ConfigurationManager.AppSettings["ReportPath1"].ToString();
        protected void Page_Init(object sender, EventArgs e)
        {

            ReportViewer1.Visible = true;
            ReportViewer1.ProcessingMode = ProcessingMode.Remote;
            ReportViewer1.ServerReport.ReportServerUrl = new Uri(urlReportServer);
            ReportViewer1.ServerReport.ReportPath = (ReportPath1 + "/clientwisedetailReport");


            Int32 mode = Convert.ToInt32(Request.QueryString["mode"]);
            Int32 ClientAddressid = Convert.ToInt32(Request.QueryString["ClientID"]);
            String Fromdate = Convert.ToString(Request.QueryString["Fromdate"]);
            String Todate = Convert.ToString(Request.QueryString["Todate"]);
            Int32 userid = Convert.ToInt32(Request.QueryString["Userid"]);


            ArrayList reportParam = new ArrayList();
            reportParam = ReportDefaultPatam(mode, ClientAddressid, Fromdate, Todate, userid);
            ReportParameter[] param = new ReportParameter[reportParam.Count];
            for (int k = 0; k < reportParam.Count; k++)
            {
                param[k] = (ReportParameter)reportParam[k];
            }
```

```
    protected void Page_Init(object sender, EventArgs e)
    {

        ReportViewer1.Visible = true;
        ReportViewer1.ProcessingMode = ProcessingMode.Remote;
        ReportViewer1.ServerReport.ReportServerUrl = new Uri(urlReportServer);
        ReportViewer1.ServerReport.ReportPath = (ReportPath1 + "/clientwisedetailReport");


        Int32 mode = Convert.ToInt32(Request.QueryString["mode"]);
        Int32 ClientAddressid = Convert.ToInt32(Request.QueryString["ClientID"]);
        String Fromdate = Convert.ToString(Request.QueryString["Fromdate"]);
        String Todate = Convert.ToString(Request.QueryString["Todate"]);
        Int32 userid = Convert.ToInt32(Request.QueryString["Userid"]);


        ArrayList reportParam = new ArrayList();
        reportParam = ReportDefaultPatam(mode, ClientAddressid, Fromdate, Todate, userid);
        ReportParameter[] param = new ReportParameter[reportParam.Count];
        for (int k = 0; k < reportParam.Count; k++)
        {
            param[k] = (ReportParameter)reportParam[k];
        }

        ReportViewer1.ServerReport.SetParameters(param);
        this.ReportViewer1.ServerReport.Refresh();

    }
    private ArrayList ReportDefaultPatam(Int32 mode, Int32 ClientAddressid, String Fromdate, String Todate, Int32 userid)
    {
        try
        {
            ArrayList arrLstDefaultParam = new ArrayList();
            arrLstDefaultParam.Add(CreateReportParameter("mode", mode.ToString()));
            arrLstDefaultParam.Add(CreateReportParameter("ClientID", ClientAddressid.ToString()));
            arrLstDefaultParam.Add(CreateReportParameter("Fromdate", Fromdate));
            arrLstDefaultParam.Add(CreateReportParameter("Todate", Todate));
            arrLstDefaultParam.Add(CreateReportParameter("UserId", userid.ToString()));
            return arrLstDefaultParam;
        }
        catch (Exception e)
        {
            throw e;
        }
    }
```

```
        private ReportParameter CreateReportParameter(string paramName, string pramValue)
        {
            try
            {
                ReportParameter aParam = new ReportParameter(paramName, pramValue);
                return aParam;
            }
            catch (Exception exx)
            {
                throw exx;
            }
        }
```

Step 10:

This another way of passing value to parameter:

```
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    public string urlReportServer = ConfigurationManager.AppSettings["ReportServerURL"].ToString();
    public string ReportPath1 = ConfigurationManager.AppSettings["reportpath1"].ToString();
    protected void Page_Init(object sender, EventArgs e)
    {

        ReportViewer1.Visible = true;
        ReportViewer1.ProcessingMode = ProcessingMode.Remote;
        ReportViewer1.ServerReport.ReportServerUrl = new Uri(urlReportServer);
        ReportViewer1.ServerReport.ReportPath = (ReportPath1 + "/meeting");


        string fromdate = Request.QueryString["Fromdate"];
        string todate = Request.QueryString["Todate"];
        ReportParameter rp1 = new ReportParameter("fromdate", fromdate);
        ReportParameter rp2 = new ReportParameter("todate", todate);
        this.ReportViewer1.ServerReport.SetParameters(new ReportParameter[] { rp1, rp2 });
        this.ReportViewer1.ServerReport.Refresh();
    }
}
```

step 11:

1.Print button won't work in chrome, fire fox other browser except InternetExplorer

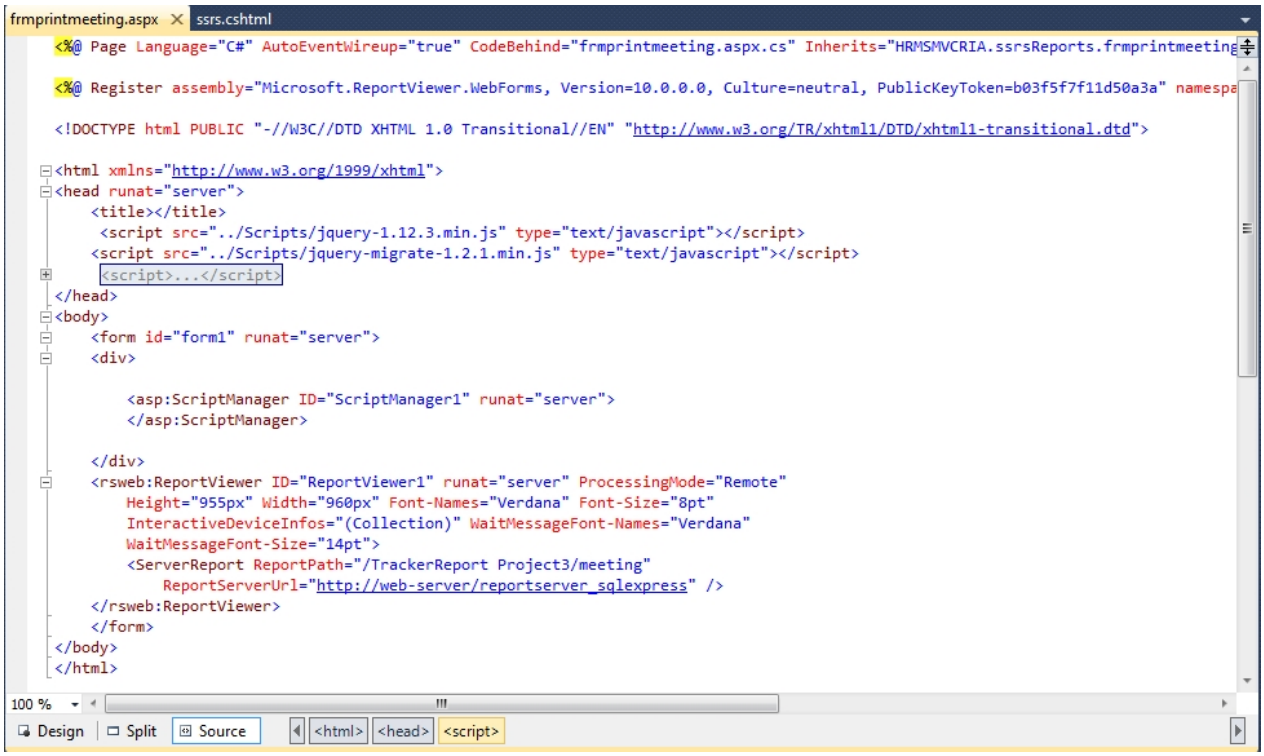
Print Button

Print option:

print button won't work in chrome, firefox and other browsers. Except InternetExplorer. so we included a Jquery for print button. Add `jquery-1.12.3.min.js` and `jquery-migrate-1.2.1.min.js`

Code:

Add this code in .aspx page.




```
<script type="text/javascript" language="javascript">

        $(document).ready(function () {

            if ($.browser.mozilla || $.browser.webkit) {

                try {

                    showPrintButton();

                }

                catch (e) { alert(e); }

            }

        });


        function showPrintButton() {

            var table = $("table[title='Refresh']");

            var parentTable = $(table).parents('table');

            var parentDiv = $(parentTable).parents('div').parents('div').first();

            parentDiv.append('<input type="image" style="border-width: 0px; padding:
3px;margin-top:2px; height:16px; width: 16px;" alt="Print" src="../
Reserved.ReportViewerWebControl.axd?
OpType=Resource&amp;Version=9.0.30729.1&amp;Name=Microsoft.Reporting.WebForms.Icons.Print
.gif";title="Print" onclick="PrintReport();">');


        }
```

```
        // Print Report function
        function PrintReport() {

            //get the ReportViewer Id
            var rv1 = $('#ReportViewer1');
            var iDoc = rv1.parents('html');

            // Reading the report styles
            var styles = iDoc.find("head style[id$='ReportControl_styles']").html();
            if ((styles == undefined) || (styles == '')) {
                iDoc.find('head script').each(function () {
                    var cnt = $(this).html();
                    var p1 = cnt.indexOf('ReportStyles":"');
                    if (p1 > 0) {
                        p1 += 15;
                        var p2 = cnt.indexOf('"', p1);
                        styles = cnt.substr(p1, p2 - p1);
                    }
                });
            }
            if (styles == '') { alert("Cannot generate styles, Displaying without
styles.."); }
            styles = '<style type="text/css">' + styles + "</style>";

            // Reading the report html
            var table = rv1.find("div[id$='_oReportDiv']");
            if (table == undefined) {
                alert("Report source not found.");
                return;
            }

            // Generating a copy of the report in a new window
```

```
            var docType = '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//
EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">';

            var docCnt = styles + table.parent().html();

            var docHead = '<head><style>body{margin:5;padding:0;}</style></head>';

            var winAttr = "location=yes, statusbar=no, directories=no, menubar=no,
titlebar=no, toolbar=no, dependent=no, width=720, height=600, resizable=yes, screenX=200,
screenY=200, personalbar=no, scrollbars=yes"; ;

            var newWin = window.open("", "_blank", winAttr);

            writeDoc = newWin.document;

            writeDoc.open();

            writeDoc.write(docType + '<html>' + docHead + '<body
onload="window.print();">' + docCnt + '</body></html>');

            writeDoc.close();

            newWin.focus();

            // uncomment to autoclose the preview window when printing is confirmed or
canceled.

            // newWin.close();

        };

    </script>
```

Screen-shot: