

# Client-Side Reporting with Visual Studio in C#



Asif Sayed

## **Client-Side Reporting with Visual Studio in C#**

**Copyright © 2007 by Asif Sayed**

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-59059-854-2

ISBN-10 (pbk): 1-59059-854-7

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Java™ and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc., in the US and other countries. Apress, Inc., is not affiliated with Sun Microsystems, Inc., and this book was written without endorsement from Sun Microsystems, Inc.

Lead Editors: Jim Huddleston, Jeff Pepper

Technical Reviewer: Ty Anderson

Editorial Board: Steve Anglin, Ewan Buckingham, Gary Cornell, Jonathan Gennick, Jason Gilmore,

Jonathan Hassell, Chris Mills, Matthew Moodie, Jeffrey Pepper, Ben Renow-Clarke,

Dominic Shakeshaft, Matt Wade, Tom Welsh

Project Manager: Kylie Johnston

Copy Edit Manager: Nicole Flores

Copy Editor: Heather Lang

Assistant Production Director: Kari Brooks-Copony

Production Editor: Ellie Fountain

Compositors: Dina Quan and Linda Weidemann

Proofreader: Nancy Riddiough

Indexer: Carol Burbo

Artist: Kinetic Publishing Services, LLC

Cover Designer: Kurt Krames

Manufacturing Director: Tom Debolski

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit <http://www.springeronline.com>.

For information on translations, please contact Apress directly at 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail [info@apress.com](mailto:info@apress.com), or visit <http://www.apress.com>.

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at <http://www.apress.com> in the Source Code/Download section.



# Reporting with Visual Studio 2008 Web Forms

In Chapter 14, you learned to develop reports using Visual Studio 2008 with Windows Forms. In this chapter, we will look at how to develop the report using Visual Studio 2008 with ASP.NET web forms. Developing VS 2008 reports with web forms is not hugely different from developing VS 2005 Web Forms reports, so you will see similarities between this chapter and Chapter 5, in which you saw VS 2005 Web Forms in action.

As we are developing reports for the new client here, let me also show another reporting technique of RS—the drill-down feature of RS. You'll see how certain parts of a report can be toggled to hide and unhide based on a user's choice. So, what are we waiting for? Let's start.

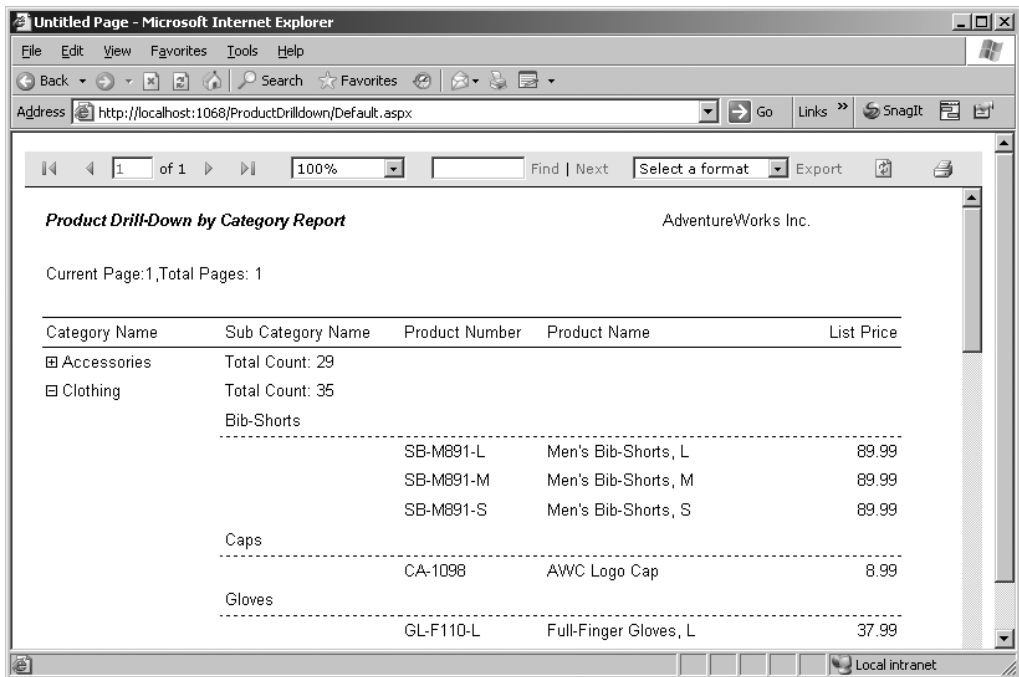
This chapter will cover creating Visual Studio 2008 reports with ASP.NET web forms by developing a drill-down report.

## Product Drill-Down by Category Report

Assume you're working for AdventureWorks Incorporated as a developer. You have the task of developing a Product Drill-Down by Category report. Initially, the report should only show the product categories and the total number of products belonging to each category. Once the user drills down, the report should show a list of products by subcategory. The report should meet all the characteristics described in Table 15-1, and the report output should match Figure 15-1.

**Table 15-1.** *Report Characteristics*

Characteristics	Value
Report title	Product Drill-down by Category Report
Company title	AdventureWorks Inc.
Page number	Yes (Current Page: n, Total Pages: n)
Data source	ProductDrilldown
Columns to report	ProductNumber, ProductName, CategoryName, SubCategoryName, ListPrice
Page size	Letter
Page orientation	Portrait
Layout design	Header and body sections



**Figure 15-1.** *The Product Drill-Down by Category report*

## Business Case

A product drill-down report is a common report requested by various departments in organizations. The beauty of a drill-down report is the ability to empower the user to see only the needed information from the output. The other highlight of this report is the ability to show the count of products belonging to each product category.

A user can expand or collapse the group sections to find more details as needed. You might be wondering how it works. Well, with drill-down reports, detail rows are simply hidden and become visible only when a user toggles them.

This is similar to the tree view structure you are familiar with in software commonly in use today (we use this a lot while mapping data columns from the data source). You can toggle a tree node by clicking either the plus (+) or minus (-) symbol on the node to view or hide the underlying details, respectively. In Figure 15-1, a plus sign (+) is displayed to the left of the product category. A user can click the icon, and as a result, the Hidden property of the details row is toggled between true and false.

---

**Note** If you decide to export the drill-down report, it will only work in Excel format. PDF format doesn't support this; the output in the PDF file will be the last state of the report in preview mode, either expanded or collapsed.

---

## Getting the ASP.NET Web Site Ready

Please open Visual Studio, and use the following steps, illustrated in Figure 15-2, to create an ASP.NET Web Site:

1. Click File ► New ► Web Site.
2. In the Templates pane, select ► ASP.NET Web Site.
3. In the Templates pane, select language ► Visual C#.
4. Please give the application a name; I've called the project ProductDrilldown. You may choose a different location for storing the application files according to your preference. Also make sure to select .NET Framework 2.0.
5. Click the OK button to finish the process. After you click OK, Visual Studio will create a new ASP.NET Web Site. You'll notice that a new page with the name Default.aspx is added to the solution. Please see Figure 15-3 for generated code and a view of Solution Explorer.

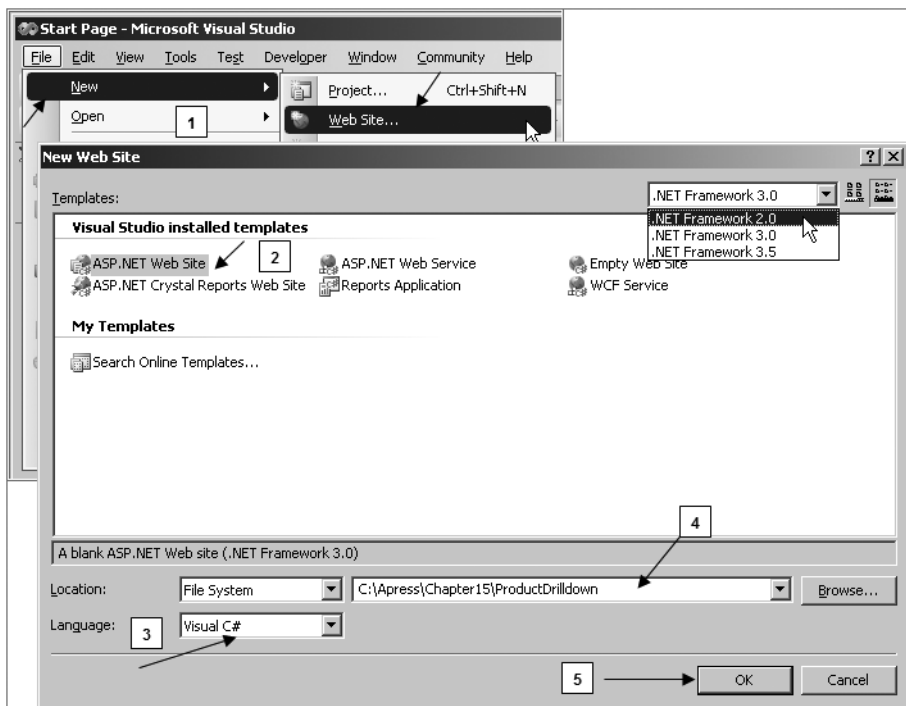
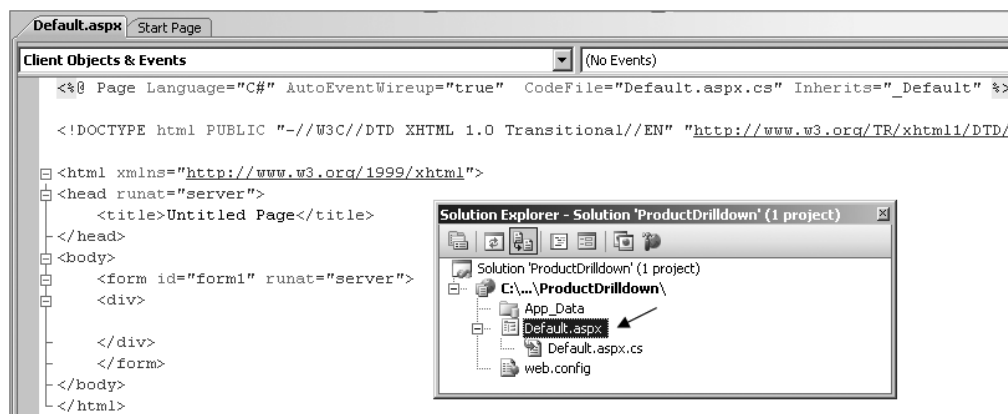


Figure 15-2. Create a new ASP.NET Web site



**Figure 15-3.** Generated code and view of Solution Explorer

It's time to add the dataset and ReportViewer. Let's start by selecting the project in Solution Explorer, right-clicking it, and selecting **Add ► New Item ► Dataset**. Please name the dataset `dsProductDrilldown`. You'll notice that Visual Studio will ask you to put the dataset inside the `App_Code` folder; go ahead and click the Yes button.

Before you add the ReportViewer, please make sure that the `Default.aspx` page is open in the designer and that HTML source mode is selected. Now, let's add the ReportViewer to the project by dragging **Data ► ReportViewer** from the toolbox and dropping it between the `<div>` tags. As a result of this action, you'll notice that `ReportViewer1` is added to the `Default.aspx` page, and the generated code will look like the following:

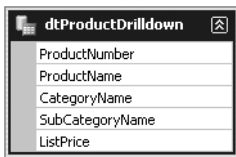
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <rsweb:ReportViewer ID="ReportViewer1" runat="server" Width="100%">
        </rsweb:ReportViewer>
    </div>
  </form>
</body>
</html>
```

You'll also notice that I've set `Width="100%"` to make sure the ReportViewer takes the maximum space available on the page to display the report.

## Step 1: Creating a Data Table

Use the following steps to add a data table inside the dataset:

1. You can go to the dataset designer in two ways: double-click `dsProductDrilldown` inside Solution Explorer, or right-click the `dsProductDrilldown` node and select View Designer.
2. Let's add the data table by right-clicking the design surface and selecting Add ► data table.
3. Click the header of the newly created data table, and name it `dtProductDrilldown`. Let's start adding columns to `dtProductDrilldown` by right-clicking the data table and selecting Add ► Column.
4. Please add the following columns into the data table, which should then look similar to Figure 15-4:
  - ProductNumber (System.String)
  - ProductName (System.String)
  - CategoryName (System.String)
  - SubCategoryName (System.String)
  - UnitsInStock (System.Double)



**Figure 15-4.** Final look of the `dtProductDrilldown` data table

## Step 2: Designing the Report Layout

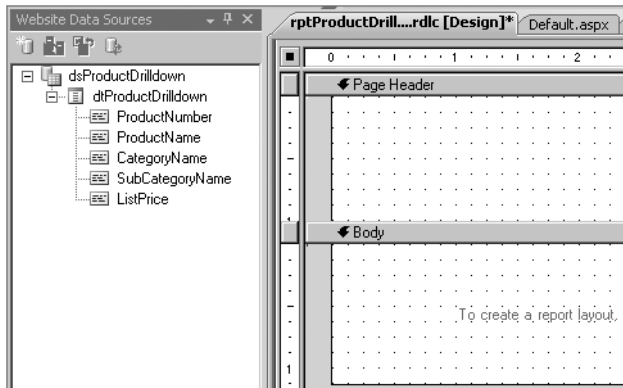
You must be thinking designing a drill-down report will be a daunting task! Well, although it might look like one, the process is simple—as simple as adding a few data groups and setting up a few properties.

As you can see in Figure 15-1, the report shows the product category as a tree view that is ready for the user to toggle. Underlying this hidden view is the product subcategory and other information related to the products. Another interesting point here is the use of the `Count()` function to count the number of products for each product category.

Add the report by selecting the project inside Solution Explorer and right-clicking it; select Add ► New Item, and select Report from the Add New Item dialog box. Please name the report `rptProductDrilldown.rdlc`. Click the Add button to complete the process and make the new report part of the project. You'll also notice that a new toolbox called Data Sources is available with our dataset information inside.

## Adding a Header

Let's add the header to the report. As usual, adding a header is simple: right-click the open area inside the report designer and select Page Header. Your report design surface should look similar to Figure 10-5.



**Figure 15-5.** *The report design surface with header and body sections*

## Setting Up the Page

According to the requirements defined earlier, let's set up the page. We need to make sure the report is letter sized and has a portrait page orientation. Right-click the open area inside the design surface, and select Properties; you may wish to put your name in the Author field and fill in the Description field.

## Designing the Page Header

Now, we have the header and the body sections added to our report. Next, please drag and drop the following report items inside the header section:

- Textbox item for the report title
- Textbox item for the company title
- Textbox item for the page number

Report item properties are changed in one of the following ways: you need to select the Report Item, right-click it, and select Properties or access the general properties toolbox. Let's start changing the properties; after selecting each of the text boxes, please specify the values for each report item's properties according to Table 15-2.

**Table 15-2.** Report Item Properties for the Header

Report Item	Property	Value
textbox1	Value	Product Drill-Down by Category Report
	Font	Italic, Arial, 10pt, Bold
textbox2	Value	AdventureWorks Inc.
	TextAlign	Right
textbox3	Value	= "Current Page: " & Globals!PageNumber & ", " & "Total Pages: " & Globals!TotalPages

## Designing the Body Section

Since we need two data groups for this report, we will make use of the table report item. Let's start working on this section by dragging Report Items ► Table from the toolbox and dropping it inside the body section in the report designer. A new table item is part of the report now, and it has the default name of `table1`. To add two more columns, right-click the right-most column header on the table item, and select "Insert Column to the Right". Make sure you have a total of five columns inside `table1`.

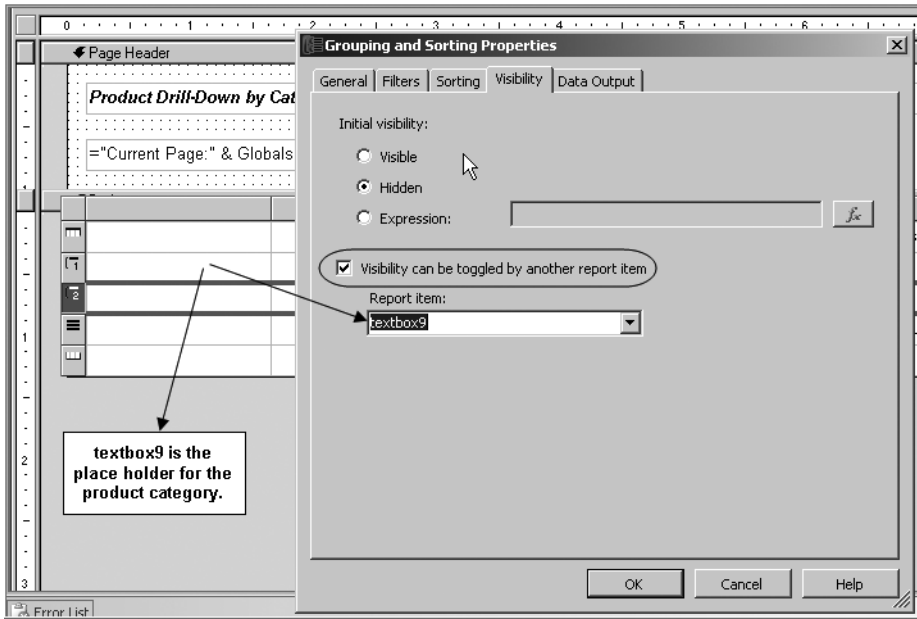
As usual, after adding a table you may choose your favorite method to map the data table's columns to the text box report items: type an expression or drag and drop from the data source.

Drag Data Source ► `dsProductDrilldown` ► `ProductNumber` and drop it inside the third column of the table item's detail section. Why start mapping with the third column? Well, as you can see in Figure 15-1, we are going to use the first two columns only as headers for the product category and subcategory.

Repeat the data mapping task for the rest of the columns in `dsProductDrilldown`.

As you know, we need to add two data groups to `table1`. Let's insert the group by selecting the entire detail row (`TableRow3`) of `table1`. Once the row is selected, right-click it, and select Insert Group. For the "group on" expression, select or type `=Fields!CategoryName.Value`. Since we don't need the group footer, please make sure to uncheck the Include Group Footer check box (you may revisit Chapter 4 to learn more about how to work with the table report item). Repeat the same steps to add the product subcategory group. However, this time, for the "group on" expression select or type `=Fields!SubCategoryName.Value`.

All right, we added the two necessary groups; however, we have not yet set the drill-down behavior of the report, which we need to hide the product subcategory and related production information. Therefore, while creating the group for the product subcategory, we also have to make sure to go to the Visibility tab of the Grouping and Sorting Properties dialog box to set the "Initial visibility" option to Hidden. Check the "Visibility can be toggled by another report item" check box, and select or type `textbox9` (a placeholder for the data column of the product category) in the "Report item" field. Please see Figure 15-6 for an illustration of these steps.



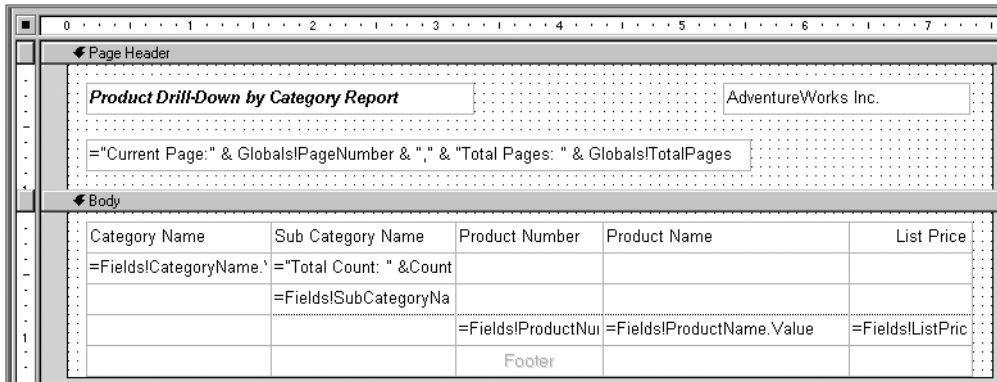
**Figure 15-6.** Toggle visibility on the product category to create a drill-down effect.

That's it; that is what we need to create a drill-down effect. Now, let's move on and take care of headings for the product category and subcategory. Select Data Source ► `dsProductDrilldown` ► `CategoryName` and drag and drop it inside the first column of group header 1. When you drag and drop, the default name of the text box gets changed to the data column name. Therefore, in this case, make sure to change the text box name from `CategoryName` back to `textbox9`. This action will also add the group header `Category Name` automatically. It is your choice to accept it or change the group header according to your needs. Repeat the step to target `TableRow3` and the second column for `SubCategoryName`.

As you know, one of the requirements for the report is the count of products for each product category. We can easily achieve this by specifying the following expression in the second column of `TableRow2`:

```
"Total Count: " & Count(Fields!SubCategoryName.Value, "table1_Group1")
```

As you can see, we are using the `Count()` function to include all the subcategories falling under the data group `table1_Group1`. Please make sure your report design surface looks similar to the one shown in Figure 15-7.



**Figure 15-7.** The report designer after adding the header and body sections

Please make sure all items inside table1 are properly set to property values matching those in Table 15-3.

**Table 15-3.** Items Property Values Inside table1

Item	Property	Value
textbox26	Value	Category Name
textbox4	Value	Sub Category Name
textbox6	Value	Product Number
textbox13	Value	Product Name
textbox16	Value	List Price
textbox16	TextAlign	Right
textbox9	Value	=Fields!CategoryName.Value
textbox14	Value	="Total Count: " & Count(Fields!SubCategoryName.Value, "table1_Group1")
SubCategoryName	Value	=Fields!SubCategoryName.Value
ProductNumber	Value	=Fields!ProductNumber.Value
ProductName	Value	=Fields!ProductName.Value
ListPrice	Value	= Fields!ListValue.Value
ListPrice	Format	N
TableRow1	BorderStyle	None, , , Solid, Solid

### Step 3: Writing the C# Code

Well, that's all we need on the front end of report design. Let's add the following code behind the `Default.aspx.cs`:

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;
using Microsoft.Reporting.WebForms;

public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        //Declare connection string
        string cnString = "Data Source=(local);Initial Catalog=RealWorld;➡
            Integrated Security=SSPI;";

        //Declare Connection, command and other related objects
        SqlConnection conReport = new SqlConnection(cnString);
        SqlCommand cmdReport = new SqlCommand();
        SqlDataReader drReport;
        Dataset dsReport = new dsProductDrilldown();

        try
        {
            conReport.Open();

            cmdReport.CommandType = CommandType.Text;
            cmdReport.Connection = conReport;
            cmdReport.CommandText = "Select * FROM dbo.ProductDrilldown ➡
                order by CategoryName,SubCategoryName,ProductNumber";

            drReport = cmdReport.ExecuteReader();
            dsReport.Tables[0].Load(drReport);

            drReport.Close();
            conReport.Close();

            //provide local report information to viewer
            ReportViewer1.LocalReport.ReportPath = "rptProductDrilldown.rdlc";
```



