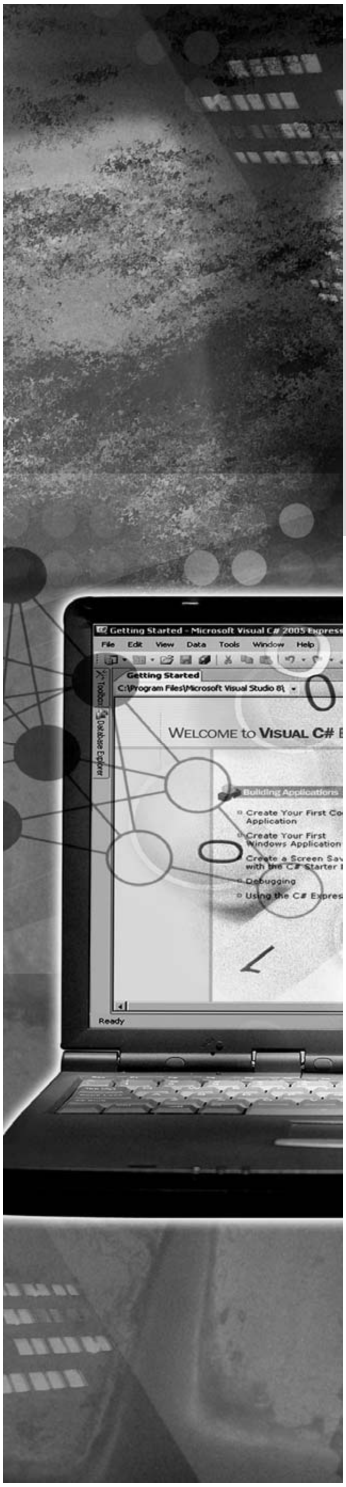


Implementing Printing and Reporting in Windows Forms Applications

Any application that requires data management is incomplete without the printing and reporting functionality.

This chapter discusses the functionality of various print components. In addition, the chapter provides a detailed description of Crystal Reports and the implementation of the CrystalReportViewer control.

Objectives

In this chapter, you will learn to:

- Identify the functionality of print components
- Identify the functionality of Crystal Reports

Introducing Print Components

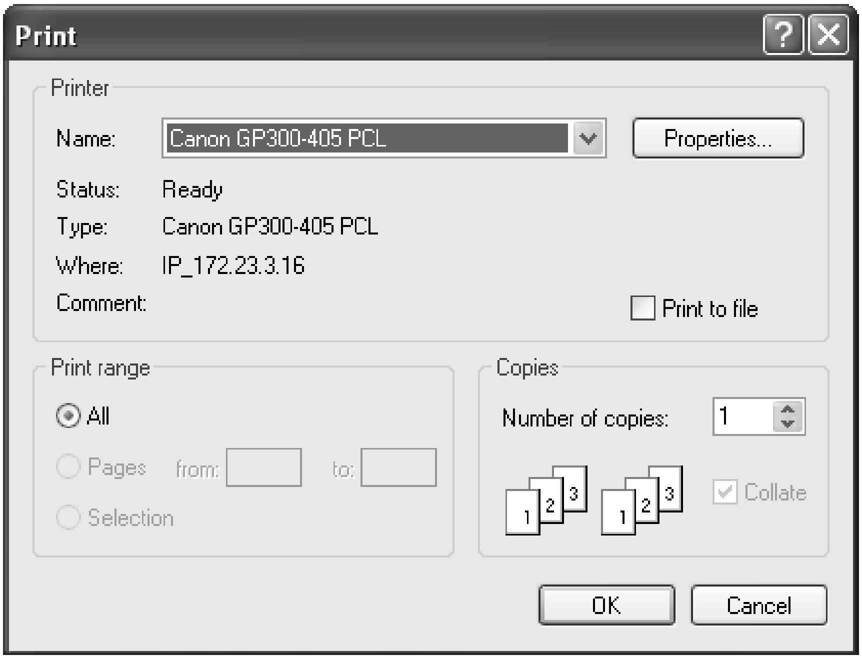
While writing an application, you might need to print data entered by the user or print the data stored in a file. You can use the PrintDialog control or the object of the `PrintDialog` class to print the documents or data.

You can use various controls or classes available in VC# to do the following tasks:

- Print text or graphics
- Change the printer attached to the computer
- Modify the Page Setup information
- Enable the Print Preview option
- Enable security permissions

Printing Text and Graphics

In VC#, you can use the default Print dialog box to print any text or graphics. The following figure shows the default Print dialog box.



Printer
Name: Canon GP300-405 PCL
Status: Ready
Type: Canon GP300-405 PCL
Where: IP_172.23.3.16
Comment:
☐ Print to file

Print range
[✓ All] Pages from: to: Selection

Copies
Number of copies: 1
✓ Collate

The Print Dialog Box

To invoke the default Print dialog box, you need to add the PrintDialog control and the PrintDocument control to the form. Alternatively, you can create an instance of the

`PrintDialog` and `PrintDocument` classes. You also need to set the Document property of the PrintDialog object either to the PrintDocument control that you have added to the form or to the instance of the `PrintDocument` class.

Printing Text in a Windows Form

Use the following example to invoke the default Print dialog box by using the PrintDialog control and a PrintDocument control. The same can also be done by instantiating the `PrintDialog` class. Before you execute the example, you need to add a button and a text box to the form. You also need to change the Name and Text property of the button to Print.

Use the following example to invoke the default Print dialog box by using the PrintDialog and PrintDocument controls:

```
//Code for the Click event of the Print button
private void Print_Click(object sender, EventArgs e)
{
     printDialog1.Document = printDocument1;
     DialogResult result = printDialog1.ShowDialog();
   if (result == DialogResult.OK)
         printDocument1.Print();
}
//Code for the PrintPage event of the printDocument1
private void printDocument1_PrintPage(object
sender,System.Drawing.Printing.PrintPageEventArgs e)
  {
        e.Graphics.DrawString(textBox1.Text, new Font("Arial",
        40,FontStyle.Bold), Brushes.Black, 150, 125);
  }
```

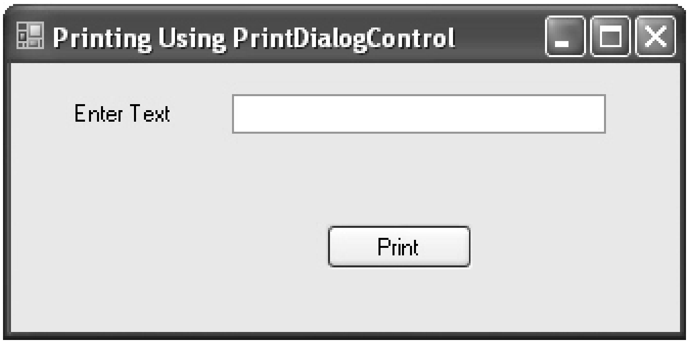
Use the following example to invoke the default Print dialog box by instantiating the PrintDialog class:

```
using System.Drawing.Printing;
public class Form1 : System.Windows.Forms.Form
{
//Instantiating object of the PrintDialog class

    PrintDialog pdlg = new PrintDialog();
    internal PrintDocument PDocument = new PrintDocument();
//Code for the Click event of the print button

    private void Print_Click(object sender, System.EventArgs e)
    {
        pdlg.Document = PDocument;
        DialogResult result = pdlg.ShowDialog();
        if (result == DialogResult.OK)
            PDocument.Print();
    }
//Code for the PrintPage event of the PrintDocument control
```

```
        private void PDocument_PrintPage(object sender,
        System.Drawing.Printing.PrintPageEventArgs e)
        {
            e.Graphics.DrawString(textBox1.Text, new Font("Arial", 40,
            FontStyle.Bold),Brushes.Black, 150, 125);
        }
    }
```

In the example, the `ShowDialog()` method is used to display the Print dialog box on the screen, and the DialogResult property is used to get the button that a user clicks in the Print dialog box. The PrintDocument control is used to set the properties that specify the content to be printed and then to print the content within Windows applications using the `Print()` method. The content to be printed is specified in the PrintPage event of the PrintDocument control.

The following figure shows the form that will appear after you execute the example.



User Interface Form

The default Print dialog box will appear when the Print button is clicked.

Just a minute:

Which property is used to trap the button that a user clicks in the Print dialog box?

1. *DialogResult Property*
2. *Document Property*
3. *Filter Property*
4. *Tag Property*

Answer:

1. *DialogResult Property*

Some of the properties of the `PrintDocument` class are described in the following table.

Property	***Description***
DefaultPageSettings	*Gets or sets page settings.*
PrinterSettings	*Gets or sets the printer that prints the document, modify the properties of the selected printer, and modify settings associated with the print job. For example, number of copies that will be printed.*
DocumentName	*Gets or sets the document name to display while printing the document.*
PrintController	*Gets or sets the print controller.*

Some Properties of the PrintDocument Class

Using the `PrintDocument` class you can print text as well as graphics. To print graphics you can use the various methods of the `Graphics` class.

Printing Graphics in a Windows Form

You can print graphics in a form by using the various methods of the `Graphics` class. The `Graphics` class provides methods for sending objects to a device, such as a screen or printer.

To print graphics in a Windows Forms, you need to perform the following steps:

1. Add a **PrintDocument** component and a **PrintDialog** component to your form or use `PrintDocument` and `PrintDialog` classes.
2. In the PrintPage event handler, use the **Graphics** property of the `PrintPageEventArgs` class to instruct the printer on what kind of graphics to print.

   The following example shows how to print a line and an ellipse using the `Graphics` class:

```
public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

       //Declaring a Button, a PrintDocument and a PrintDialog object

        private Button printButton = new Button();
        private PrintDocument printDocument1 = new PrintDocument();
        private PrintDialog printDialog1 = new PrintDialog();
        void printButton_Click(object sender, EventArgs e)
        {
            //calling the CaptureScreen method
            CaptureScreen();
            //Setting the Document to be printed
            printDialog1.Document = printDocument1;
            DialogResult res = printDialog1.ShowDialog();
            if (res == DialogResult.OK)
                printDocument1.Print();
        }

        //Declaring a Bitmap object
        Bitmap memoryImage;

        private void CaptureScreen()
        {
            //creating Graphics object for the Form
            Graphics myGraphics = this.CreateGraphics();
            myGraphics.DrawEllipse(Pens.Black, 50, 50, 200, 200);
            myGraphics.DrawEllipse(Pens.Black, 100, 100, 20, 20);
            myGraphics.DrawEllipse(Pens.Black, 170, 100, 20, 20);
            myGraphics.DrawLine(Pens.Black, 150, 140, 150, 170);
            myGraphics.DrawArc(Pens.Black, 110, 100, 85, 110, 45,
            85);

            //creating a Size object s and initialising it with the
              height and width of the form
            Size s = this.Size;

            //initialising the memoryImage object with the height and
              width of the form
            memoryImage = new Bitmap(s.Width, s.Height, myGraphics);
```

```
                //creating a new grphics object with the memoryImage
                  object
                Graphics g = Graphics.FromImage(memoryImage);
                //performing the bitblck trnsfer of data from the
                  specified coordinates to the graphics object
                g.CopyFromScreen(this.Location.X, this.Location.Y, 0, 0,
                s);
            }

            private void printDocument1_PrintPage(System.Object sender,
            System.Drawing.Printing.PrintPageEventArgs e)
            {
                e.Graphics.DrawImage(memoryImage, 0, 0);
            }

            private void Form1_Load(object sender, EventArgs e)
            {
                printButton.Text = "Print Form";

                //Adding the printButton_click event handler
                printButton.Click += new EventHandler(printButton_Click);

                //Adding the printDocument1_PrintPage event handler
                printDocument1.PrintPage += new
                PrintPageEventHandler(printDocument1_PrintPage);

                //Adding the printButton button to the controls array
                this.Controls.Add(printButton);
            }
        }
```

When a user executes the preceding example, a form with a Print Form button is displayed on the screen. When the user clicks the Print Form button a graphics is drawn on the form and the Print dialog box appears on the screen. If the user presses the OK button in the Print dialog box the form including the button and the graphics, gets printed. The following example uses some methods of the `Graphics` class.

Some of the methods of the `Graphics` class are described in the following table.

Methods	***Description***
DrawArc	*Draws an arc representing a portion of an ellipse specified by a System.Drawing.Rectangle structure.*
DrawLine	*Draws a line connecting two System.Drawing.Point structures.*
DrawEllipse	*Draws an ellipse defined by a bounding rectangle specified by a pair of coordinates, a height, and a width.*

Some Methods of the `Graphics` Class

You can also change the printer specifications or attach another printer to your computer.

Changing Printer Attached to a User's Computer

If you have more than one printer installed on your computer, you can change the printer by using the PrintDialog control or `PrintDialog` class.

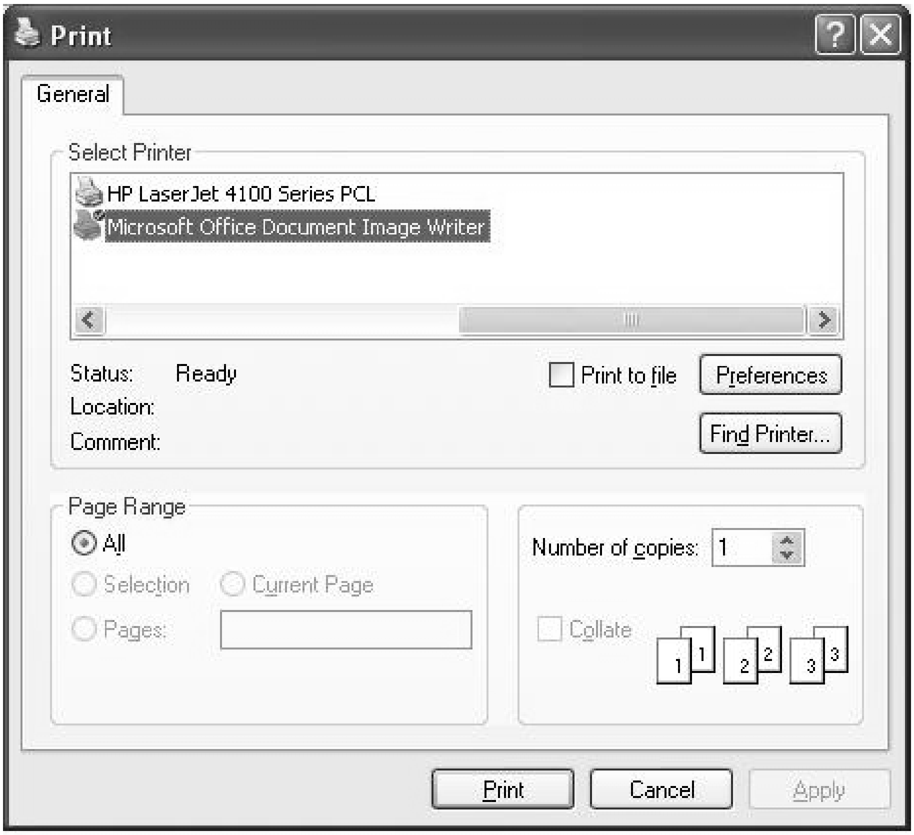
To change the printer using the PrintDialog control, you need to perform the following steps:

1. Add a **PrintDialog** control to the form.

   Write the following code to invoke the **PrintDialog** box:

```
printDialog1.ShowDialog();
```

The preceding code will invoke a Print dialog box with a list of printers installed on the computer. You can select the desired printer from the Select Printer list, as shown in the following figure.



HP LaserJet 4100 Series PCL
Microsoft Office Document Image Writer

Status: Ready
Location:
Comment:

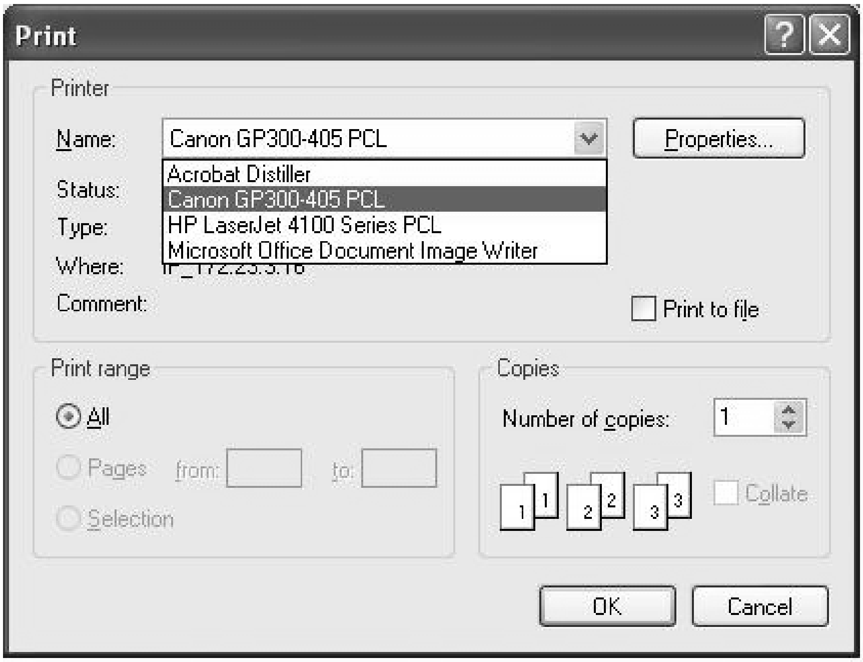
[✓ All]
Number of copies: 1

The Print Dialog Box Showing the List of Printers Installed on the Computer

Alternatively, you can change the printer by creating an instance of the `PrintDialog` class, as shown in the following example:

```
PrintDialog p = new PrintDialog();
p.ShowDialog();
```

This will invoke a Print dialog box with a list of printers installed on the computer. You can select the desired printer form the Name drop-down list, as shown in the following figure.



The Print Dialog Box Showing the List of Printers Installed on the Computer

Sometimes you might want to alert a user about the completion of the print job. This could be done using the PrintDocument component.

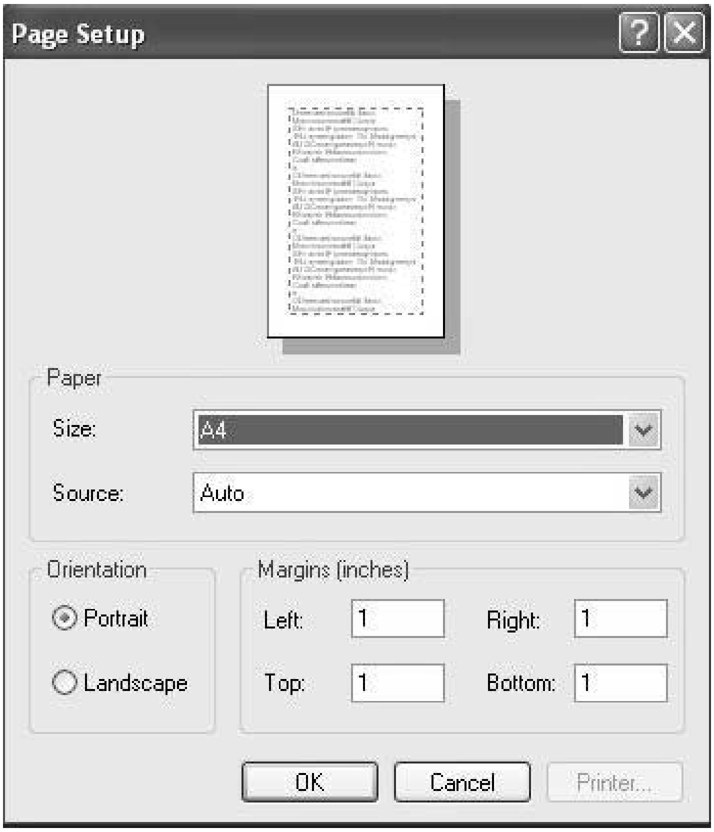
Alerting User about the Completion of a Print Job

You can implement this functionality in your application by handling the EndPrint event of the PrintDocument component. The following example shows how to alert a user when the print job is completed:

```
private void printDocument1_EndPrint(object sender,
System.Drawing.Printing.PrintEventArgs e)
        {
            MessageBox.Show("Printing is completed");
        }
```

PageSetupDialog Class

You can use the default Windows Page Setup dialog box in your application to set the page details for printing. The following figure shows the default Windows Page Setup dialog.



Size: A4
Source: Auto
Orientation: [✓ Portrait] Landscape
Left: 1
Right: 1
Top: 1
Bottom: 1

The Page Setup Dialog Box

To invoke the default Windows Page Setup dialog box, you need to add a PageSetupDialog control to the form. Alternatively, you can create an instance of the `PageSetupDialog` class. You also need to create an instance of the `PageSettings` class.

The `PageSettings` class is included in the `System.Drawing.Printing` namespace and is used to specify the settings to be applied on every printed page. The `PageSetupDialog` class is used to display the Print Setup page on the screen by using the `ShowDialog()` method. The following code snippets show how you can invoke the default Print dialog box by using the PageSetupDialog control or by instantiating the `PrintDialog` class. To execute the following code snippets, you need to add a button to the form. You also need to change the Name and Text property of the button to PageSetup.

Use the following code snippet to invoke the default Page Setup dialog box by using the PageSetupDialog control:

```
using System.Drawing.Printing;
public class Form1 : System.Windows.Forms.Form
{
    PageSettings PgSettings = new PageSettings();
//Code for the Click event of the PageSetup button
    private Sub PageSetup_Click(object sender, System.EventArgs e)
    {
      pageSetupDialog1.PageSettings = PgSettings;
      pageSetupDialog1.ShowDialog();
    }
}
```

Code to invoke the default Page Setup dialog box by instantiating the `PageSetupDialog` class is given as follows:

```
using System.Drawing.Printing;
public class Form1 : System.Windows.Forms.Form
{
    private PageSettings PgSettings = new PageSettings();
//Code for the Click event of the PageSetup button

    private void PageSetup_Click(object sender, System.EventArgs e)
    {
        PageSetupDialog pdlg = new PageSetupDialog();
        pdlg.PageSettings = PgSettings;
        pdlg.ShowDialog();
    }
}
```

Just a minute:

Name the method which is used to show the dialog box?

- ***Answer:***

  `ShowDialog()` method

PrintPreview Dialog and PrintPreview Control

The PrintPreviewDialog control displays how a document will look when printed. It provides the user with the facility to print, zoom, set page layout options, and to move between pages. If the options and the interface provided by the PrintPreviewDialog control do not fulfill your requirements, you can use the PrintPreviewControl control to create your custom print preview interface.

You have to set the Document property of the PrintPreviewDialog component to a PrintDocument object to specify the document to be printed.

There are various properties of the PrintPreviewControl control. Some of the properties of the PrintPreviewControl control are described in the following table.

Property	***Description***
Document	*Gets or sets the document to be previewed.*
UseAntiAlias	*Gets or sets a value indicating whether printing uses the antialiasing features of the operating system. It provides a more accurate display of the document, but it is slower.*
AutoZoom	*Gets or sets a value indicating whether resizing the control or changing the number of pages shown automatically adjusts the Zoom property.*
Columns	*Gets or sets the number of pages displayed horizontally across the screen.*
Rows	*Gets or sets the number of pages displayed vertically down the screen.*
StartPage	*Gets or sets the page number of the upper-left page in the PrintPreviewControl control.*
Zoom	*Gets or sets a value indicating how large the pages will appear.*

Some Properties of the PrintPreviewControl Control

You can also provide secure printing in Windows Forms applications by enabling the security for printing.

Enabling Security for Printing in Windows Forms

In Windows Forms applications printing abilities are frequently used. The .NET Framework uses the `PrintingPermission` class to control access to printing capabilities and the associated PrintingPermissionLevel value to indicate the level of access.

The following table shows the functionality available at each printing permission level.

PrintingPermissionLevel	***Description***
AllPrinting	*Provides full access to all installed printers.*
DefaultPrinting	*Enables programmatic printing to the default printer and safer printing through a restrictive printing dialog box. DefaultPrinting is a subset of AllPrinting.*
SafePrinting	*Provides printing only from a more-restricted dialog box. SafePrinting is a subset of DefaultPrinting.*
NoPrinting	*Prevents access to printers. NoPrinting is a subset of SafePrinting.*

The Printing Permissions in PrintingPermissionLevel Enumeration

Note

To use the `PrintingPermission` class and the PrintingPermissionLevel value, you need to use the `System.Drawing.Printing` namespace.

Just a minute:

Identify the property of the PrintPreviewControl, which is used to get or set the page settings.

1. *Document*
2. *DefaultPageSettings*
3. *StartPage*
4. *Zoom*

Answer:

2. DefaultPageSettings

Activity: Creating a Customized PrintPreview Dialog Control

Problem Statement

Sigma Pvt. Ltd., requires a custom version of the PrintPreviewDialog control that contains additional options such as to select the font of the text to be printed, to select file to be printed, and to type the header to be printed. The company wants to design a form that contains a PrintPreviewControl control and an OpenFileDialog control. The form should enable a user to choose the file to be printed from the OpenFileDialog and use the properties of the customized PrintPreviewControl control to print the file.

Provide a solution to the problem.

Solution

To meet the preceding requirement, you need to perform the following tasks:

1. Create a new VC# application.
2. Design the Printing Application form.
3. Add code to perform the desired task.
4. Execute the program and verify the output.

Task 1: Creating a New VC# Application

To create a new VC# application, you need to perform the following steps:

1. Select Start→All Programs→Microsoft Visual Studio 2005→Microsoft Visual Studio 2005. The Start Page - Microsoft Visual Studio window is displayed.
2. Select **File→New→Project**. The **New Project** dialog box is displayed.
3. Select the project type as **Visual C#** from the **Project types** pane and **Windows Application** from the **Templates** pane.
4. Type the name of the new project as **CustomizePrinting** in the **Name** text box.
5. Specify the location where the new project is to be created as **c:\chapter6\Activity1** in the **Location** combo box.
6. Click the **OK** button.

Task 2: Designing the Printing Application Form

When you create a VC# project, by default, the **Form1** form is added to the project. To add controls to **Form1**, you need to perform the following steps:

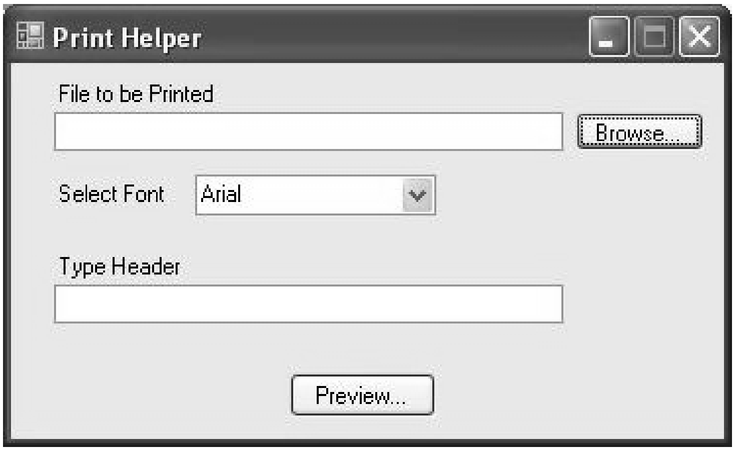
1. Drag three **Label** controls, two **TextBox** controls, a **ComboBox** control, two **Button** control, a **PrintPreviewDialog** control, and an **OpenFileDialog** control to the form from the **Toolbox** window.
2. Right-click the control and then select **Properties** from the shortcut menu to open the **Properties** window for each control added in Step 1.
3. Set the properties of the form and controls as listed in the following table.

Controls	***Properties***	***Values***
Form1	*Name*	*frmPrint*
	MaximizeBox	*False*
	StartPosition	*CenterScreen*
	Text	*Print Helper*
Label1	*Name*	*lblFileToPrint*
	Text	*File to be Printed*
Label2	*Name*	*lblFont*
	Text	*Select Font*
Label3	*Name*	*lblHeader*
	Text	*Type Header*
TextBox1	*Name*	*txtFileToPrint*
TextBox2	*Name*	*txtHeader*
Button1	*Name*	*btnBrowse*
	Text	*Browse...*
Button2	*Name*	*btnPreview*
	Text	*Preview...*
ComboBox1	*Name*	*cmbFonts*

Controls	***Properties***	***Values***
	Items	*Arial* *Courier* *Book Antiqua* *Times New Roman* *Comic Sans MS*
	Text	*Arial*
printPreviewDialog1	*Name*	*printPreviewDialog1*
openFileDialog1	*Name*	*openFileDialog1*

Properties and Values of the Form's Controls

The form should resemble the following figure.



The Print Helper Form

Task 3: Adding Code to Perform the Desired Task

To add a new class that inherits from the `PrintDocument` class, you need to perform the following steps:

1. Right-click the **CustomizePrinting** project in the **Solution Explorer** window.
2. Select **Add→Class**. The **Add New Item - CustomizePrinting** dialog box is displayed.
3. Type **CustomDocument** in the **Name** text box in the **Add New Item - CustomizePrinting** dialog box.
4. Click the **Add** button.

5. Include the following namespaces in the project to enable the class to perform the printing functionality:

```
using System.Drawing;
using System.Drawing.Printing;
using System.IO;
using System.Windows.Forms;
```

6. Change the class declaration for the **CustomDocument**, as given in the following code:

```
class CustomDocument : PrintDocument
```

7. Write the given code in the **CustomDocument** class:

```
private Font _printFont;
private string _fileToPrint;
private Single _headerHeight;
private TextReader _printStream;
// Private bitmap field

// String property to set and retrive the file to print.
public string FileToPrint
{
     get
     {
           return _fileToPrint;
     }
     set
     {
           if (File.Exists(value))
           {
                 _fileToPrint = value;
           }
           else
                 throw(new Exception("File not found."));
     }
}

// Font property to get and set the font to be used.
public Font PrintFont
{
     get
     {
           return _printFont;
     }
     set
     {
           _printFont = value;
     }
}

// Get and Set the height of header section.
```

```
public Single HeaderHeight
{
     get
     {
            return _headerHeight;
     }
     set
     {
            _headerHeight = value;
     }
}

protected virtual Single printPageHeader(RectangleF bounds,
PrintPageEventArgs e)
{
     // Create drawing surface.
     Graphics g = e.Graphics;
     // Specify font for header.
     Font drawFont = new Font("Arial", 16,FontStyle.Bold);
     // To Get the text to be displayed in the header.
     string headerText = this.DocumentName;
     // To Get the bounds of the header.
     RectangleF headerTextLayout = new RectangleF(bounds.X, bounds.Y,
     bounds.Width, bounds.Height);

     Single localHeaderHeight;

     // Header should be center aligned.
     StringFormat headerStringFormat = new StringFormat();
     headerStringFormat.Alignment = StringAlignment.Center;

     // Calculate the height of the header.
     localHeaderHeight = g.MeasureString(headerText, drawFont,
      headerTextLayout.Size, headerStringFormat).Height;

     // Draw the header.
     g.DrawString(headerText, drawFont,Brushes.Black,
     headerTextLayout, headerStringFormat);

     // Return the header height.
     return localHeaderHeight;
}

protected override void OnBeginPrint(PrintEventArgs e)
{
     base.OnBeginPrint(e);
     // Initialize the text reader.
     _printStream = new StreamReader(FileToPrint);
}

protected override void OnEndPrint(PrintEventArgs e)
{
     base.OnEndPrint(e);
     // Close the text reader.
     _printStream.Close();
}

protected override void OnPrintPage(PrintPageEventArgs e)
{
      base.OnPrintPage(e);

      // Create the drawing surface.
      Graphics gdiPage = e.Graphics;

      // Get the bounds of the print area.
      Single leftMargin = e.MarginBounds.Left;
      Single topMargin = e.MarginBounds.Top;
      Single width = e.MarginBounds.Width;
      Single height = e.MarginBounds.Height;

      // Calculate line height and lines per page;
      Single lineHeight = _printFont.GetHeight(gdiPage);
      Single linesPerPage = e.MarginBounds.Height / lineHeight;

      int lineCount = 0;
      string lineText = null;
      Single headerSize;
      Single currentPosition = topMargin;

      // Specify the header area.
      RectangleF headerBounds = new RectangleF(leftMargin, topMargin,
       width, height);
      // Get the value of the height property.
      headerBounds.Height = this.HeaderHeight;
      // Print the page header and retrieve the height of the header.
      headerSize = printPageHeader(headerBounds, e);

      // Specify the current position to ensure that line printing
      // begins below the header.
      currentPosition += (headerSize + 20);

      // Draw each line of text in turn.
      while (lineCount < linesPerPage &&
        ((lineText = _printStream.ReadLine()) != null))
      {
            gdiPage.DrawString(lineText, _printFont, Brushes.Black,
            leftMargin, (currentPosition + (lineCount++ *
            lineHeight)));
      }

       // Check if the end of the file has been reached and set
          HasMorePages property.
      if(lineText != null)
            e.HasMorePages = true;
      else
            e.HasMorePages = false;
      }
```

8. Double-click the **Form1.cs** file in the **Solution Explorer** window.

9. Double-click the **btnBrowse** button control to open the **Click** event handler. Write the following code in the **Click** event of the **btnBrowse** button control, as follows:

```
private void btnBrowse_Click(object sender, EventArgs e)
        {
            // Set initial properties of the openFileDialog.
            openFileDialog1.InitialDirectory = "c:\\";
            openFileDialog1.FileName = "";
            // Show the dialog.
            DialogResult result = openFileDialog1.ShowDialog(this);
            // If user clicked ok, display file name and path in text
box on form.
            if (result == DialogResult.OK)
            {
                txtFileToPrint.Text =
openFileDialog1.FileName.ToString();
            }
        }
```

10. Double-click the **Form1.cs** file in the **Solution Explorer** window.

11. Double-click the **btnPreview** button control to open the **Click** event handler. Write the following code in the **Click** event of the **btnPreview** button control, as follows:

```
private CustomDocument Doc = new CustomDocument();
private void btnPreview_Click(object sender, EventArgs e)
        {
            // If the FileToPrintTextBox is not empty:
            if (txtFileToPrint.Text != "")
            {
                //  Set properties of the CustomPrintDocument object
                // Using the values selected on the Printing form.
                Doc.FileToPrint = txtFileToPrint.Text;
                Doc.PrintFont = new Font(cmbFonts.Text, 10);
                Doc.DocumentName = txtHeader.Text;
                // Set the headerheight property.
                Doc.HeaderHeight = 100;
                // Specify the print document for the
                   PrintPreviewDialog.
                printPreviewDialog1.Document = Doc;
                // Open the PrintPreviewDialog.
                printPreviewDialog1.ShowDialog(this);
            }
            else
            {
                MessageBox.Show("Please select the file to be
                printed.", "File not selected", MessageBoxButtons.OK,
                MessageBoxIcon.Exclamation);
                txtFileToPrint.Focus();
            }
        }
```

Task 4: Executing the Program and Verifying the Output

To execute the program, you need to perform the following steps:

1. Press the **F5** key to build and execute the application.
2. Click the **Browse** button on the form to select the file.
3. Choose the font from the **Select Font** drop-down list.
4. Type the heading of the page that will be displayed on the printout.
5. Click the **Preview** button. The **Print preview** window is displayed.
6. Verify the output and close the Visual Studio application.
7. Close the Visual Studio application.

Introducing Crystal Reports

Consider the case of an online airways reservation system. Everyday, the system executives make reservations for thousands of customers. At the end of the month, the management may need to view the summarized transaction details, such as total number of seats booked for the month, the average customer feedback for the month, or the reservations made by a particular employee during the month for analysis.

Crystal Reports is the standard reporting tool for Visual Studio .NET, and is used to display data in a presentable manner. You can use Crystal Reports to display multiple-level totals, charts to analyze data, and much more. Creating a Crystal Reports requires minimal coding because it provides an inbuilt designer interface.

Data Access Through Crystal Reports

Crystal Reports requires database drivers to connect to a data source for accessing data. Crystal Reports in Visual C# support methods to access data from a data source. These methods are:

- The Pull Model
- The Push Model

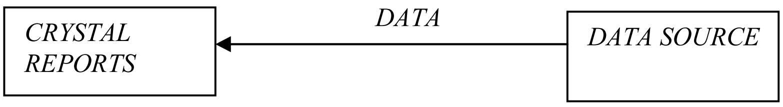
Note

A database driver is written by database vendors. It is a piece of software that forms a sort of bridge between a server hoisting a database and a client running an application.

A data source, such as an SQL server, is the application where data is actually stored.

The Pull Model

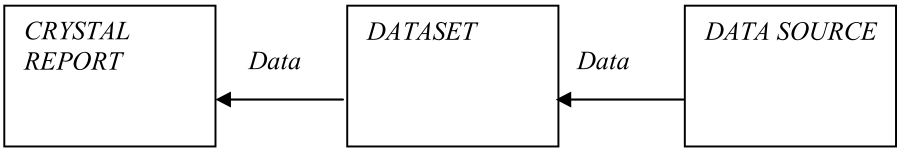
When this model is used to access data from the data source, the database driver directly retrieves data from the data source. This model does not require the developer to write code for creating a connection and retrieving data from the data source. Crystal Reports creates and manages the SQL commands for connecting to the data source and retrieving data from it, as shown in the following figure.



The Pull Model

The Push Model

When this model is used to access data from the data source, the developer writes code to connect to the data source and retrieves data from it. The data from the data source is cached in a dataset and multiple Crystal Reports accesses the data from the dataset. Thus, the Push model allows filtered data access by Crystal Reports. The Push model is generally useful for connection sharing scenarios. The following figure shows the Push Model.



The Push Model

When a dataset is created, it does not contain the actual data. It only stores the description of the data table, called metadata. Therefore, while using the Push model, you have to fill the dataset before executing the application to view the Crystal Reports.

Note

A dataset is a series of records which were originally created and managed on a computer system. A dataset usually takes the form of fields and tables within which items of data are contained and structured.

Just a minute:

Name the reporting model that does not require the developer to write code for creating a connection and retrieving data from the data source.

- ***Answer:***

  The Pull Model

Creating Crystal Reports

You can create Crystal Reports:

- Manually
- Using Standard Report Wizard
- From an existing report

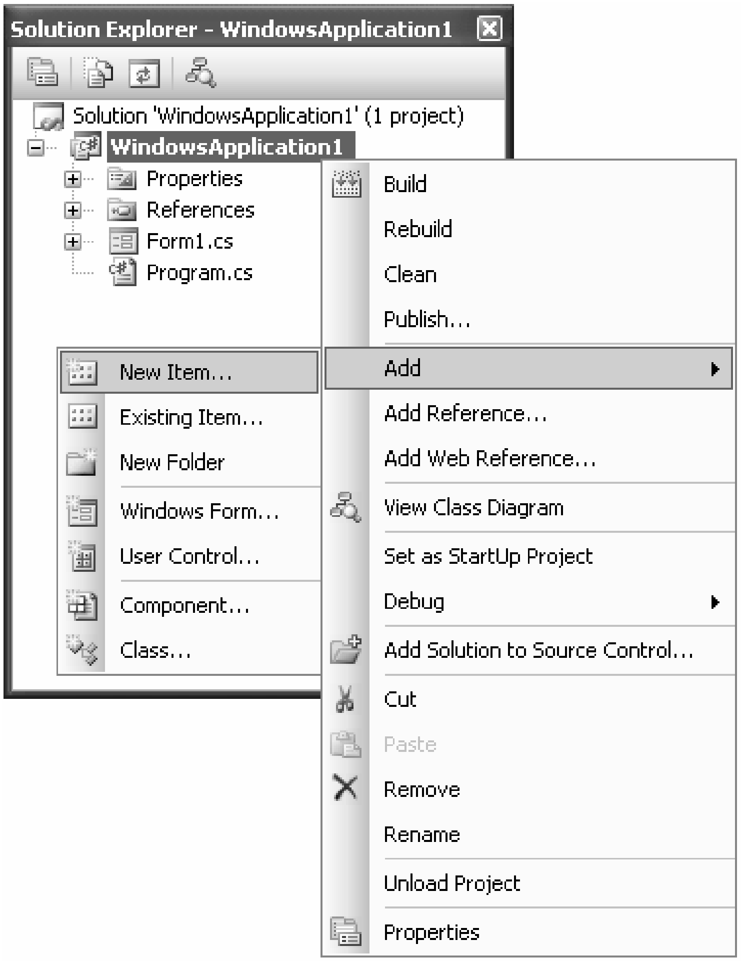
Creating Crystal Reports Manually

When Crystal Reports is created manually, data is added manually to Crystal Reports by using various tools from the **Crystal Reports Toolbox** window and the **Field Explorer** window.

To create Crystal Reports manually, you need to perform the following steps:

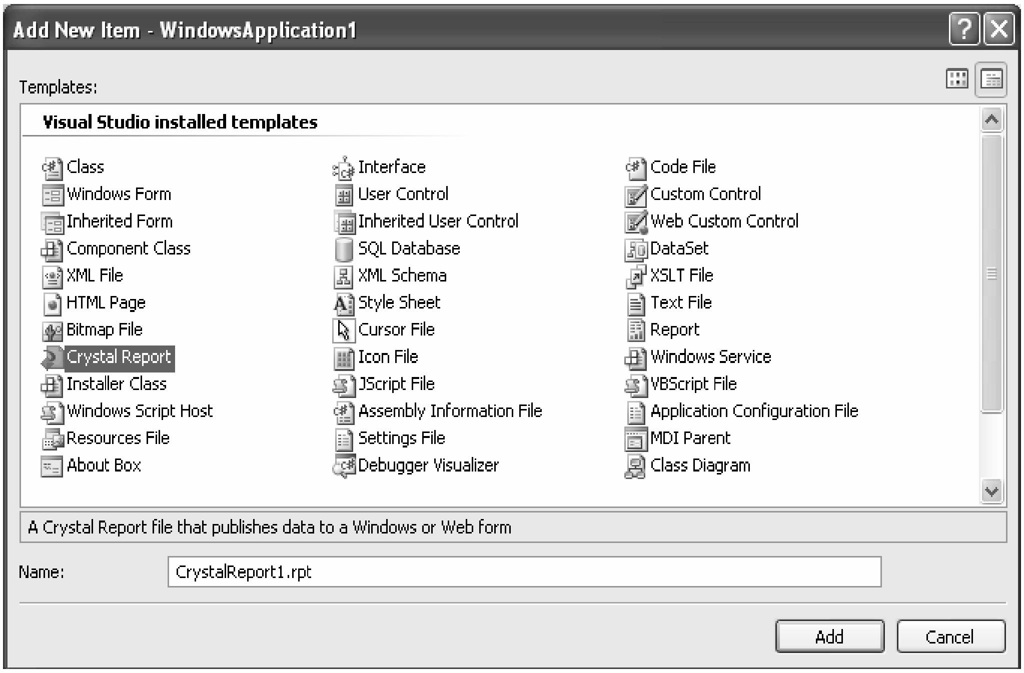
1. Create a new **Windows Application** project.

2. Right-click the project name in the **Solution Explorer** window and select **Add→New Item** from the shortcut menu, as shown in the following figure.



Selecting New Item from the Add Submenu

3. Select **Crystal Report** from the **Templates** pane in the **Add New Item** dialog box, as shown in the following figure.

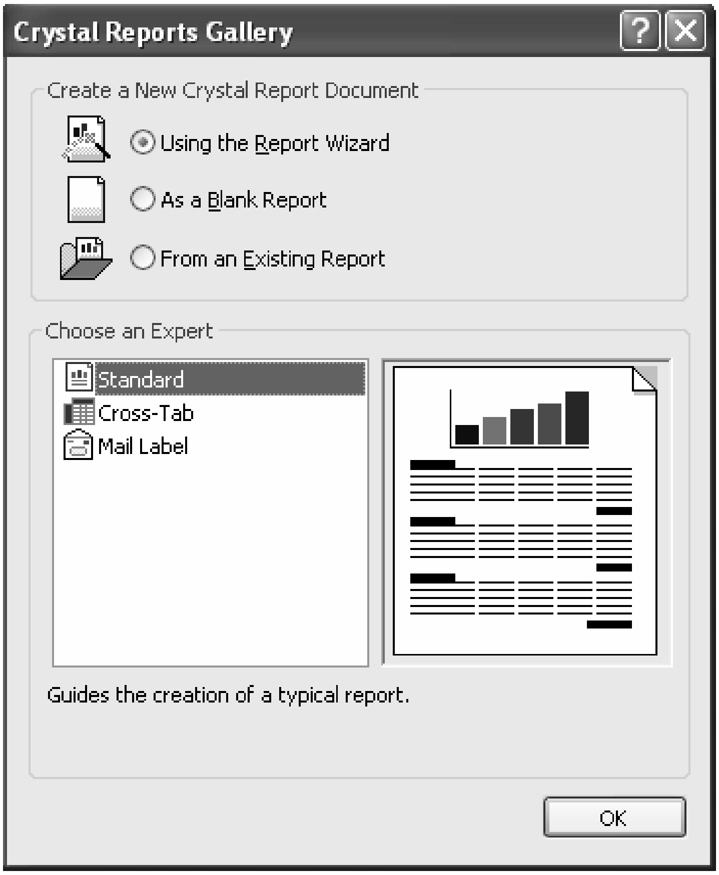


A Crystal Report file that publishes data to a Windows or Web form

Name: CrystalReport1.rpt

Crystal Reports Selected in the Add New Item – WindowsApplication1 Dialog Box

4. Rename the default Crystal Reports name, **CrystalReport1.rpt,** as required. Click the **Add** button and the **Crystal Reports Gallery** dialog box is displayed, as shown in the following figure.



The Crystal Reports Gallery Dialog Box

Note

If you are using the Crystal Reports for the first time, Crystal Decision Registration Wizard is displayed, which can be used to access additional resources, such as product updates and technical support. To add a Crystal Reports in an application, you need to click the Register Later button on the first page of the wizard.

5. Crystal Reports Gallery dialog box displays three option buttons, these are:

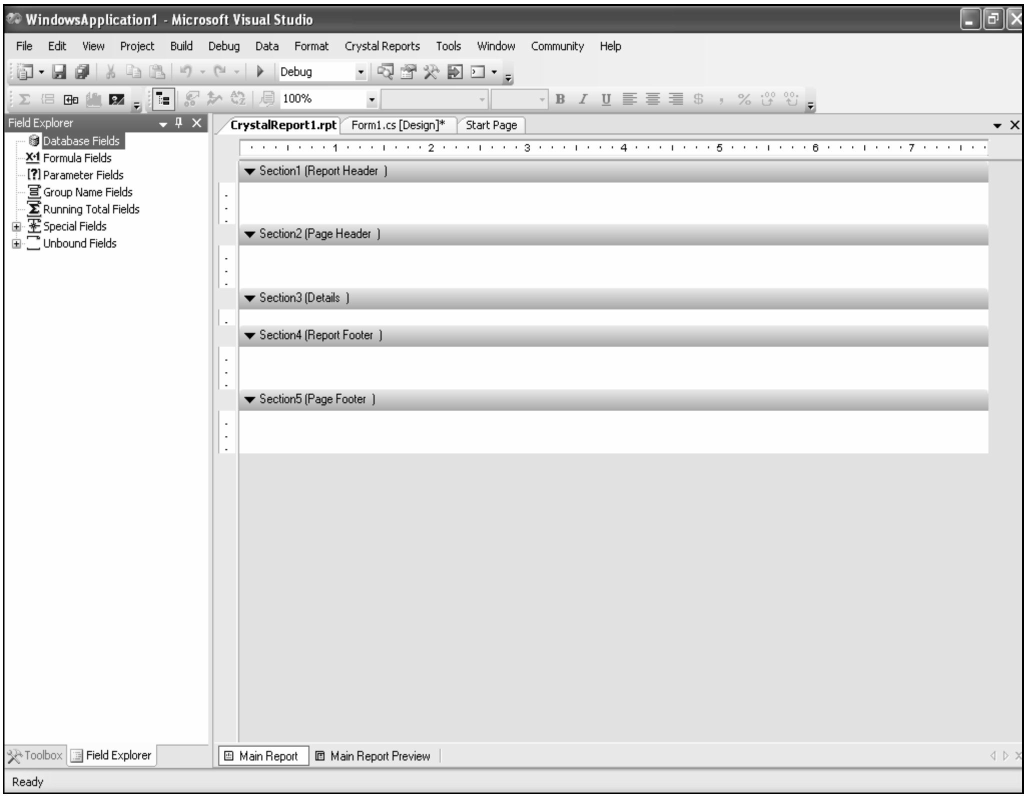
   a. **Using the Report Wizard:** Guides you through the process of report creation and creates a report based on the choices that you make, such as the fields to be included in the report, fields to group the data, and the filtering criteria for displaying data in the report.

b. **As a Blank Report:** Opens the Crystal Reports Designer where you can add the fields to be displayed in the Crystal Reports.

c. **From an Existing Report:** Creates a new Crystal Reports with the same design as another report that you specify.

For this example, select the **As a Blank Report** option button and click the **OK** button.

Clicking the **As a Blank Report** option, a Report Designer is displayed with the **Report Header**, **Page Header**, **Details**, **Report Footer**, and **Page Footer** sections, as shown in the following figure.



The Report Designer Containing Blank Sections

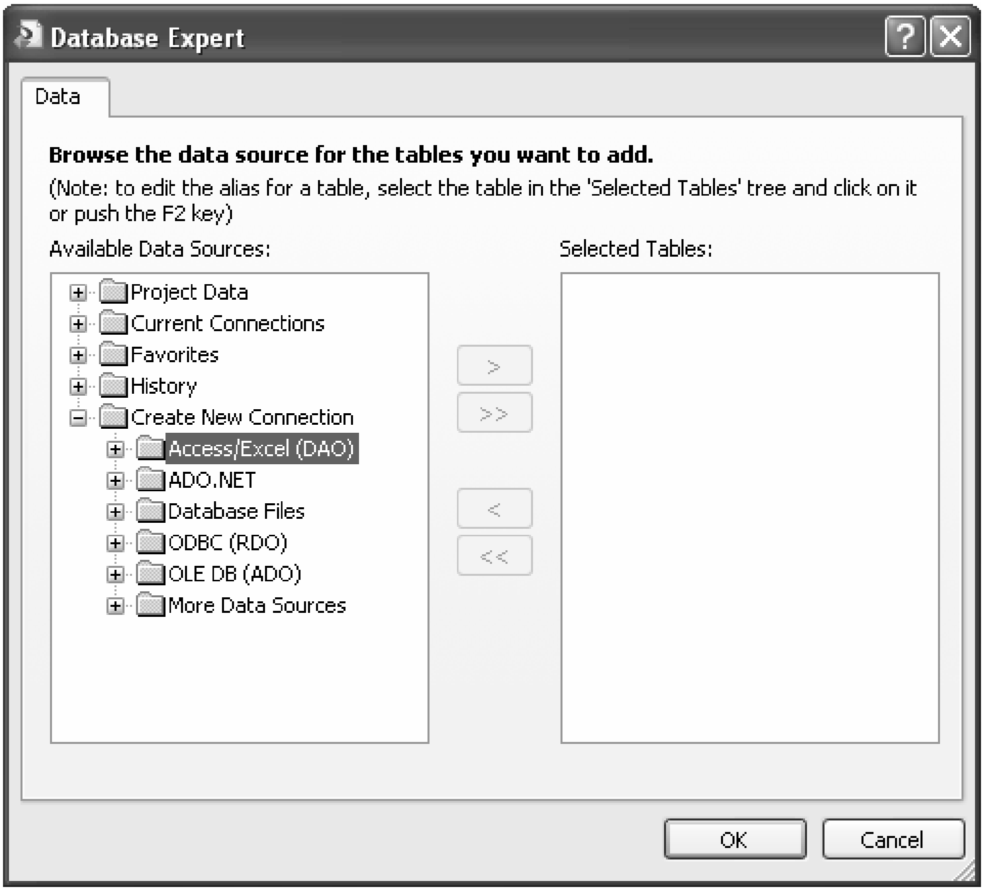
As can be seen in the preceding figure, the **Field Explorer** window is displayed on the left side of the screen.

Note

If the Field Explorer window is not visible by default, select View→Other Windows→ Document Outline to view the Field Explorer window.

You have to connect to a database and select the fields from the database table to be displayed in Crystal reports. To do so, you need to perform the following steps:

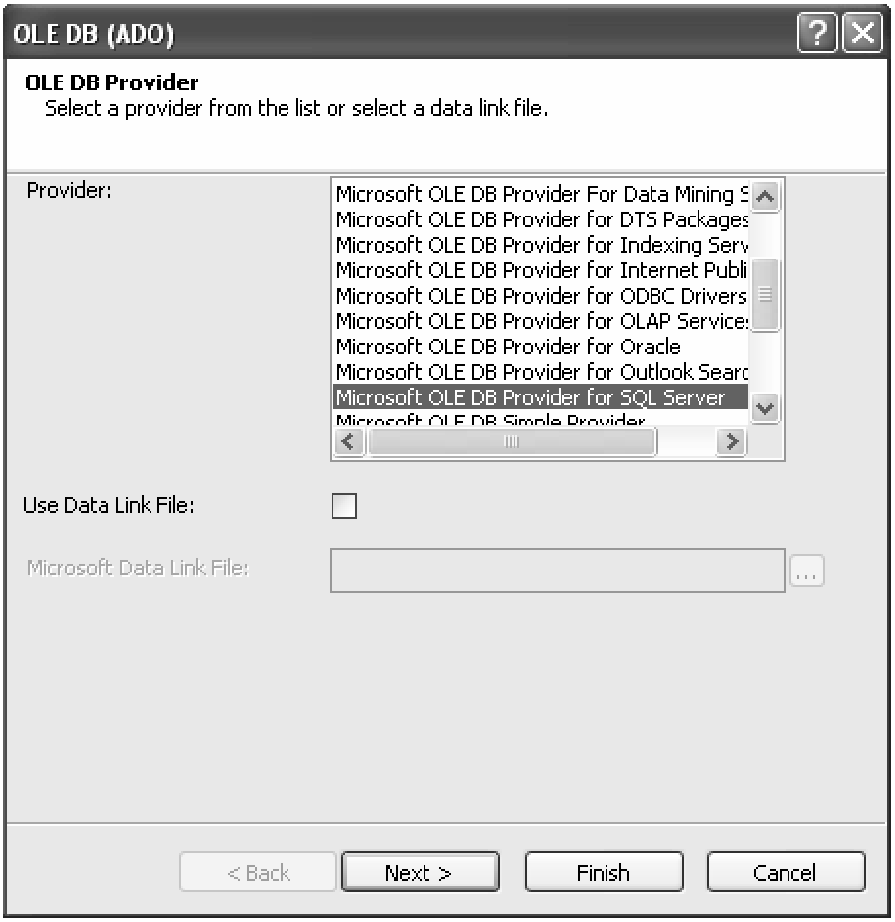
6. Right-click **Database Fields** in the **Field Explorer** window and then select the **Database Expert** option. A list of data sources is displayed under the **Data** tab of the **Database Expert** dialog box, as shown in the following figure.



Expanding the OLE DB (ADO) Folder in the Database Expert

7. Expand the **Create New Connection** folder within the **Available Data sources** list box and then, expand the **OLE DB (ADO)** list item to open the **OLE DB (ADO)** dialog box.

8. Select **Microsoft OLE DB Provider for SQL Server** within the **Provider** list in the **OLE DB (ADO)** dialog box and Click the **Next** button. The **OLE DB (ADO)** dialog box will be displayed, as follows.

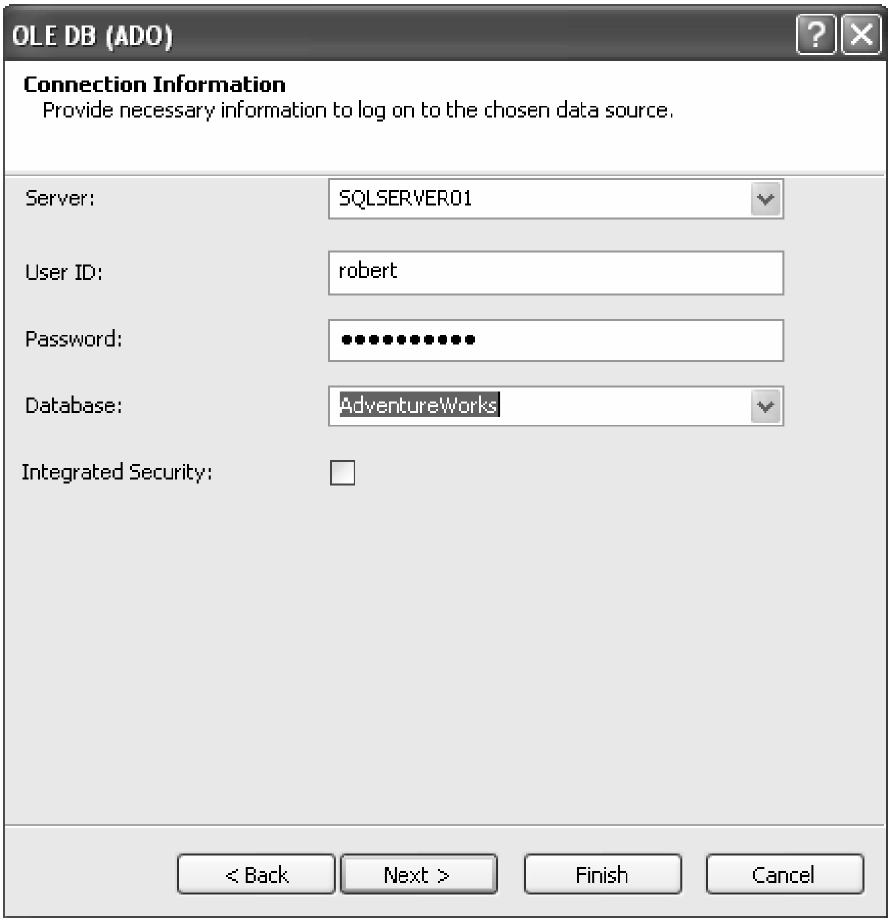

OLE DB (ADO)
OLE DB Provider
Select a provider from the list or select a data link file.
Provider:
Microsoft OLE DB Provider For Data Mining S
Microsoft OLE DB Provider for DTS Packages
Microsoft OLE DB Provider for Indexing Serv
Microsoft OLE DB Provider for Internet Publi
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for OLAP Service
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for Outlook Searc
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider
Use Data Link File:
Microsoft Data Link File:
< Back
Next >
Finish
Cancel

Selecting Microsoft OLE DB Provider for SQL Server from the Provider List

Note

You can select any provider depending in which DBMS/RDBMS you have created the database.

9. Provide the information, such as the SQL server name for the connection, the user ID and password for connecting to the server, and the database from which you want to add data to the Crystal Reports, as shown in the following figure.


OLE DB (ADO)

Connection Information

Provide necessary information to log on to the chosen data source.

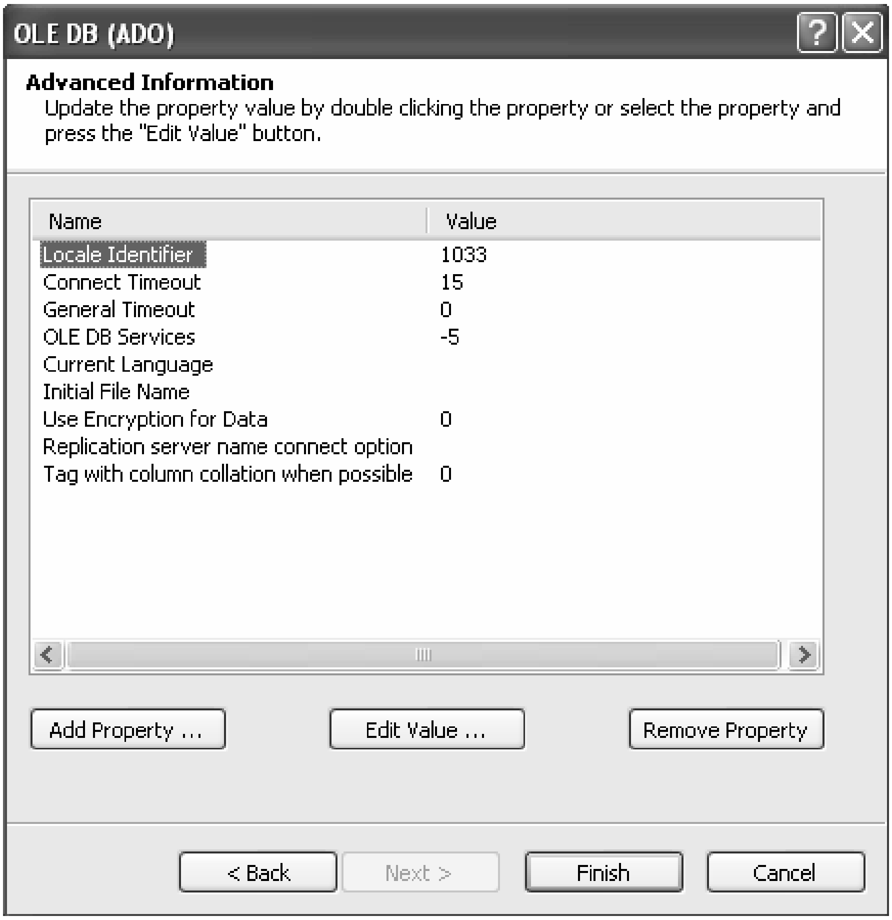
Server:	SQLSERVER01
User ID:	robert
Password:	●●●●●●●●●●●
Database:	AdventureWorks
Integrated Security:	☐

Providing the Connection Information from the Connection Information Dialog Box

10. Click the **Next** button.

    The advanced information containing the list of connection properties is displayed. You might need to edit a property, such as the Connect Timeout or the Initial File Name.

11. Edit any property, if required, by selecting the property name and clicking the **Edit Value** button or by double-clicking the property name. Click the **Finish** button. The **advance Information** dialog box is shown in the following figure.



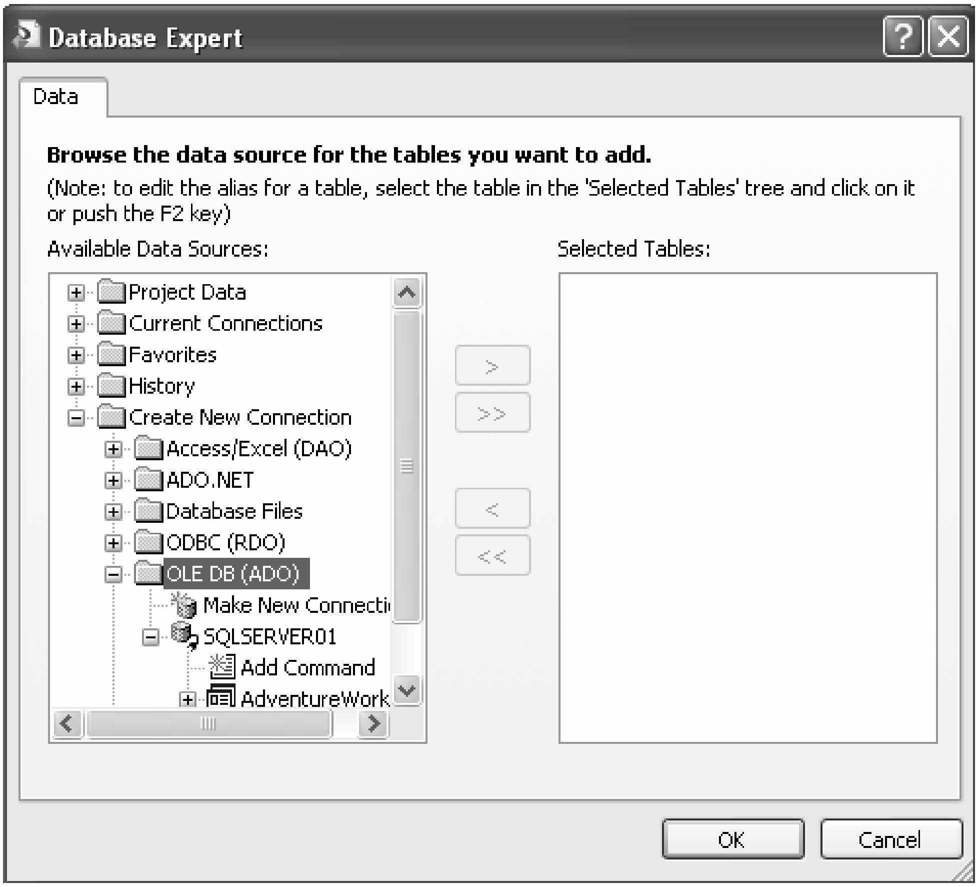
Update the property value by double clicking the property or select the property and press the "Edit Value" button.

Name	Value
Locale Identifier	1033
Connect Timeout	15
General Timeout	0
OLE DB Services	-5
Current Language	
Initial File Name	
Use Encryption for Data	0
Replication server name connect option	
Tag with column collation when possible	0

The Advanced Information About the Connection Displayed

The Database Expert dialog box is displayed, containing the Available data sources in the Available Data Sources pane.

12. Add tables from a data source by expanding the database, expanding the **Tables** category, selecting the table, and clicking on the button with the symbol (>), as shown in the following figure.


Database Expert

Data

Browse the data source for the tables you want to add.

(Note: to edit the alias for a table, select the table in the 'Selected Tables' tree and click on it or push the F2 key)

Available Data Sources:

- Project Data
- Current Connections
- Favorites
- History
- Create New Connection
  - Access/Excel (DAO)
  - ADO.NET
  - Database Files
  - ODBC (RDO)
  - OLE DB (ADO)
    - Make New Connecti
    - SQLSERVER01
      - Add Command
      - AdventureWork

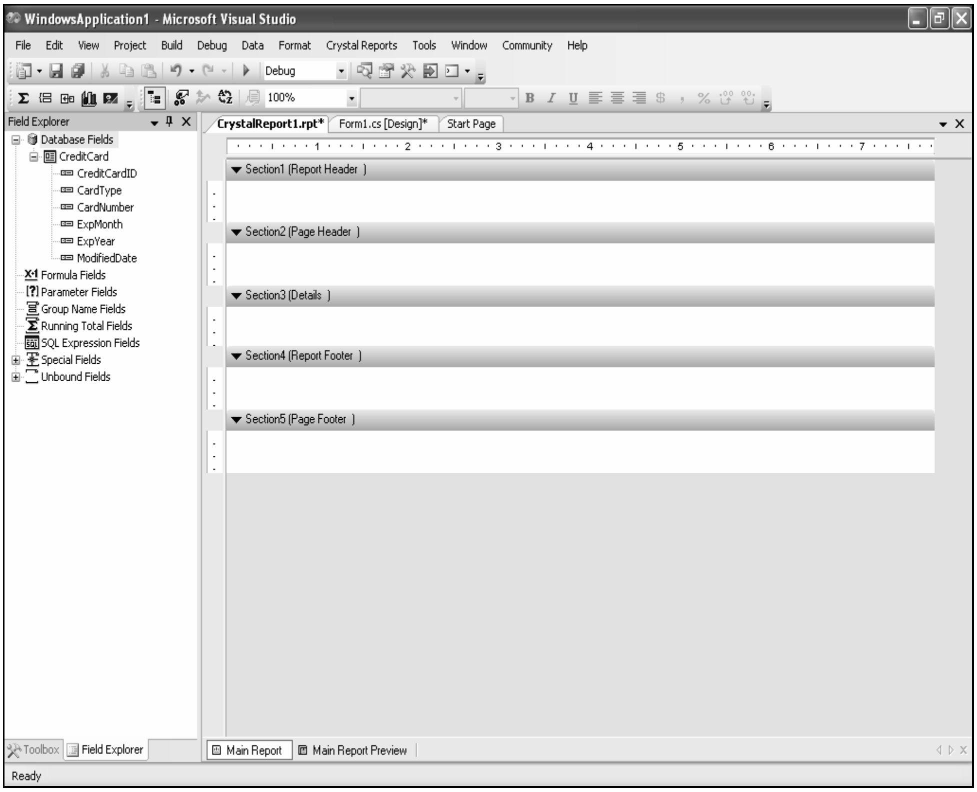
Selected Tables:

Adding Tables in a Crystal Reports

Note

If you add more than one table in the Database Expert and the added tables have matching fields, you need to click the OK button. Notice the links between the added tables is displayed under the Links tab. You can remove a link by clicking the Clear Links button.

After you click the OK button, the Database Fields node in the Field Explorer window will contain a list of selected tables and their columns, as shown in the following figure.

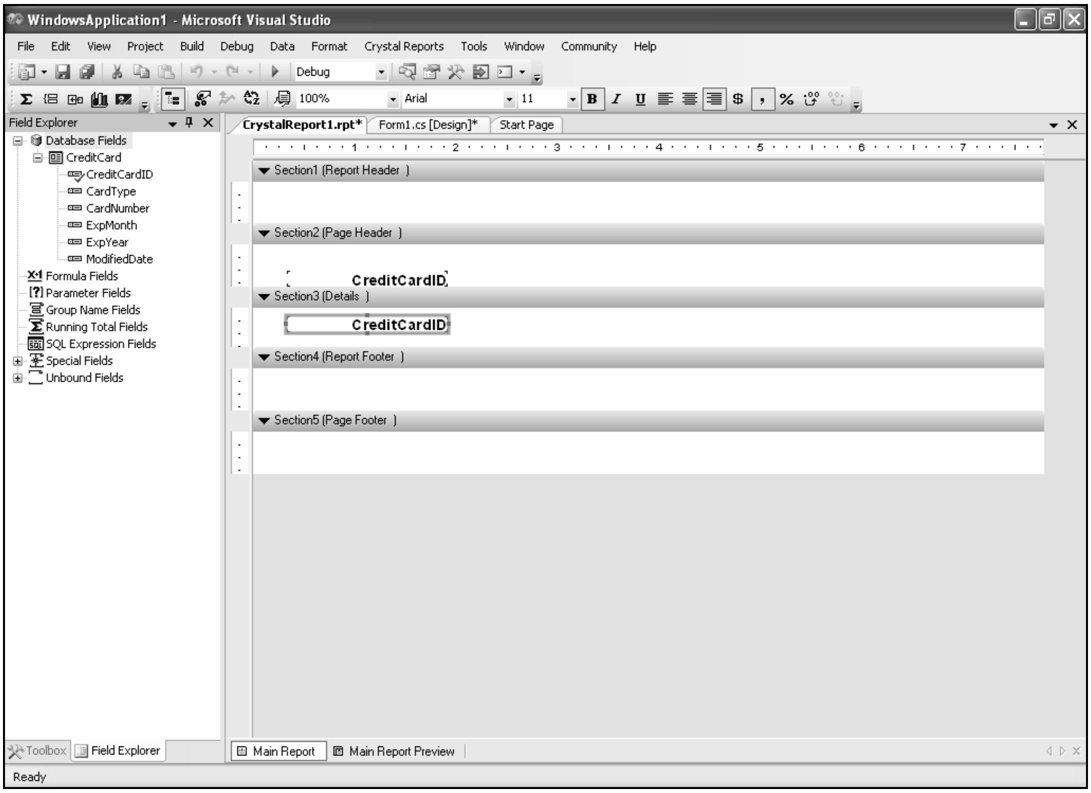


The Hierarchy of Tables and their Column under Database Fields

The preceding figure shows Report Designer. This designer displays various sections of the Report. These sections are:

- **Report Header**: Contains the report title and appears at the start of the report. Any component that you need to display once for the report, you can add to the Report Header section.
- **Page Header**: Contains field titles of the report. The component that you add to the Page Header section is displayed at the top of each page of the report.
- **Details**: Displays the actual values in the report. For example, if you need to display the value of a field in the report, you can add that field to the Details section.
- **Report Footer**: Displayed at that end of the report.
- **Page Footer**: Displayed at that end of each page of the report.

You can add a table column in the Crystal Reports by dragging the column name from the Field Explorer window into Report Designer. A sample report using the As a Blank Report option is displayed, as shown in the following figure.



A Column Added in the Crystal Reports

In the left pane the Field Explorer window contains various fields to add data to a Crystal Reports. Some commonly used fields are:

- **Formula Fields**: Used to insert a calculated field in the Crystal Reports. Formula Fields can be created by right-clicking the Formula Fields option in the Field Explorer window, selecting New from the shortcut menu, and specifying a name and the formula for the field.
- **Group Name Fields**: Used to group data in a Crystal Reports. This field can be inserted by right-clicking the Group Name Fields in the Field Explorer window and selecting Insert Group from the shortcut menu.
- **Special Fields**: Used to insert a field such as Record Number and Page Number. The Special Fields can be inserted in a Crystal Reports by dragging the available list of field from the Special Fields section to the Report Designer.

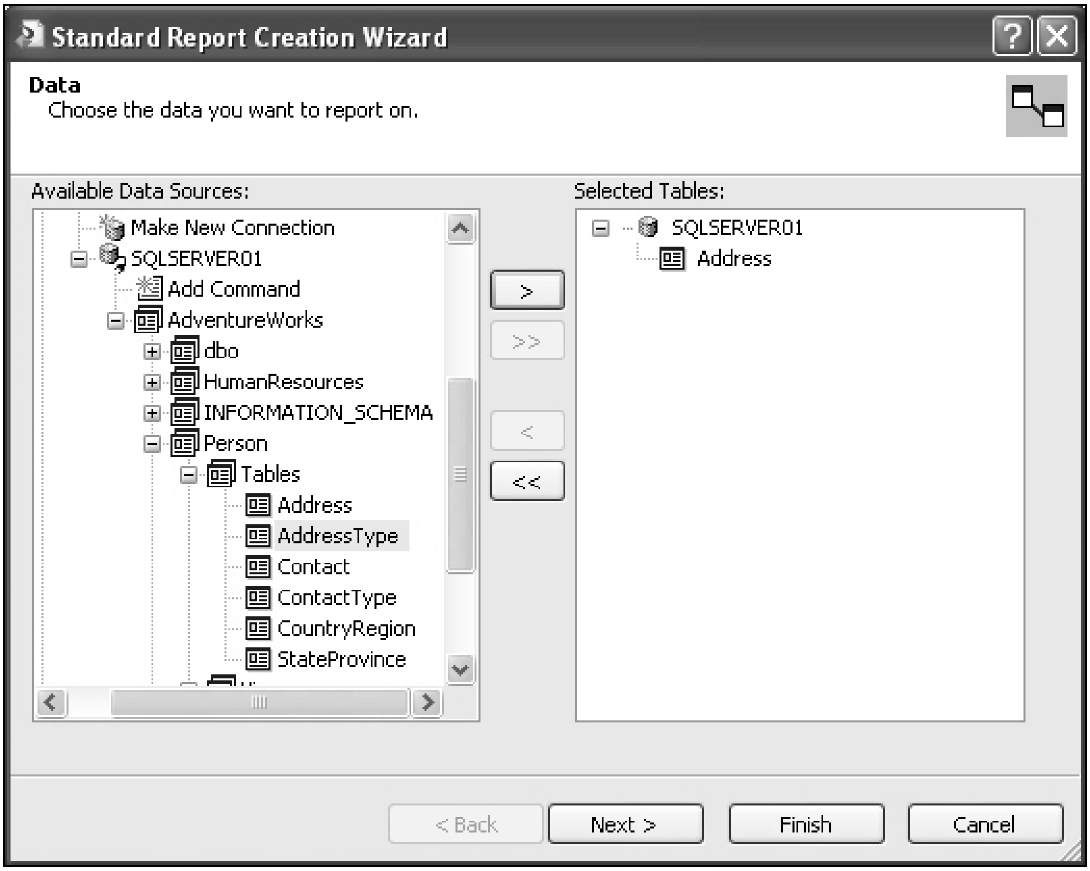
Creating Crystal Reports Using Standard Report Wizard

The Standard Report Wizard allows the creation of Crystal Reports through a wizard.

Consider a scenario. You are organizing your birthday party. You need to invite all your friends. The addresses of all the friends are stored in the Address table of the AdventureWorks database. You need to print a report containing the addresses so that you could send them an invitation.

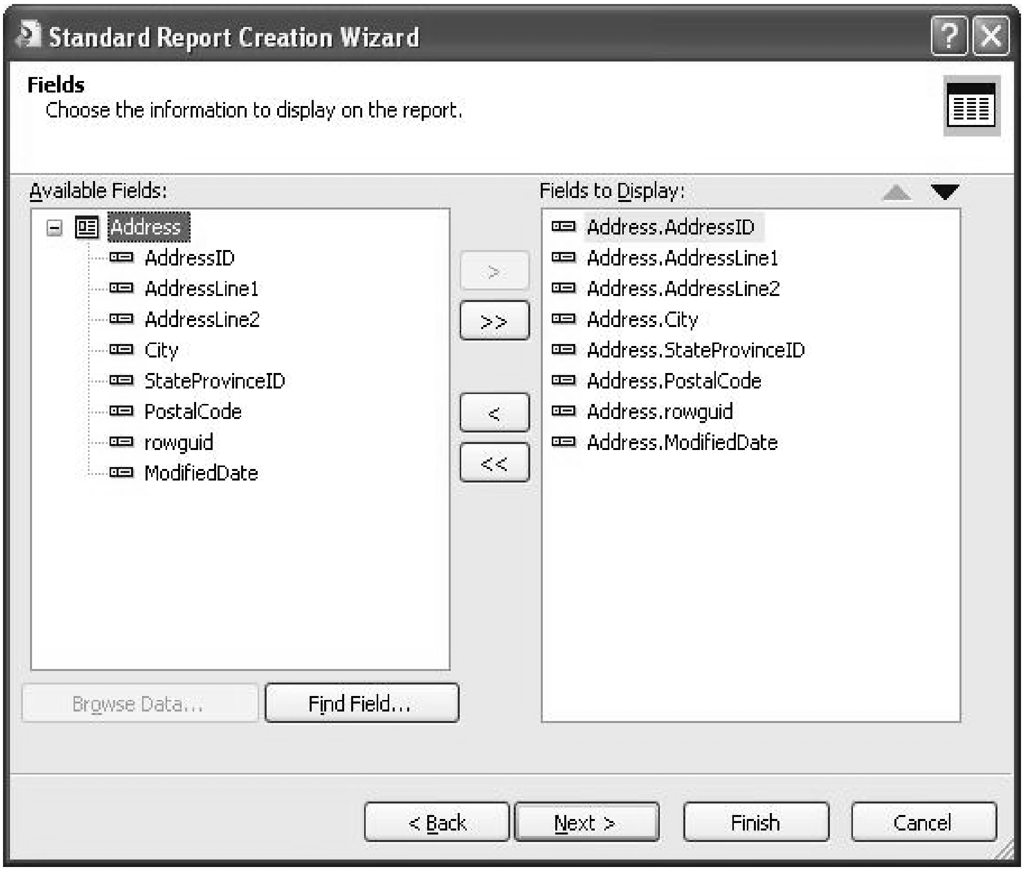
To create a Crystal Reports, which will fulfill the above requirement, by using Standard Report Wizard, you need to perform the following steps:

1. Select **Using the Report Wizard** option from the **Crystal Reports Gallery**, and click the **OK** button. The **Standard Report Creation Wizard** appears. If you have already established a connection with a database, then expand that database and insert the desired tables from the **Available data sources** list by selecting the table and clicking the **Insert Table** button. You can also create a new connection by selecting **Make New Connection** from the **Available data sources** list. Add tables from a data source by expanding the database, then expanding the **Tables** category, selecting the table, and clicking on the button with the symbol (>), as shown in the following figure.



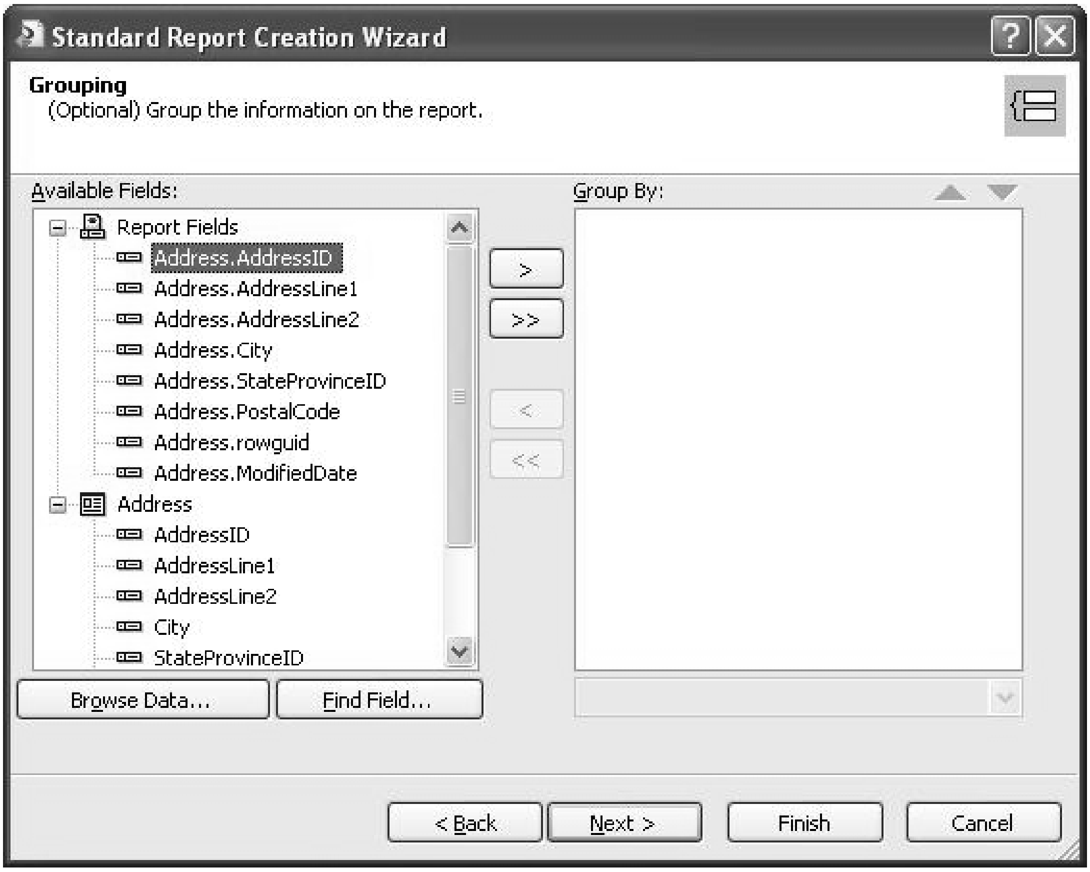
Adding Tables in a Crystal Reports

2. Click the **Next** button after inserting tables. The **Standard Report Creation Wizard** dialog box is displayed. It will show a list of available fields in the selected table. You can select the fields that you want to show in the report. By selecting the field and clicking the button with the symbol (>) to add the required field to the **Fields to Dsiplay** section of the dialog box. All the fields of a table can be added by clicking the button with the symbol (>>), as shown in the following figure.



The Fields of the Selected Table

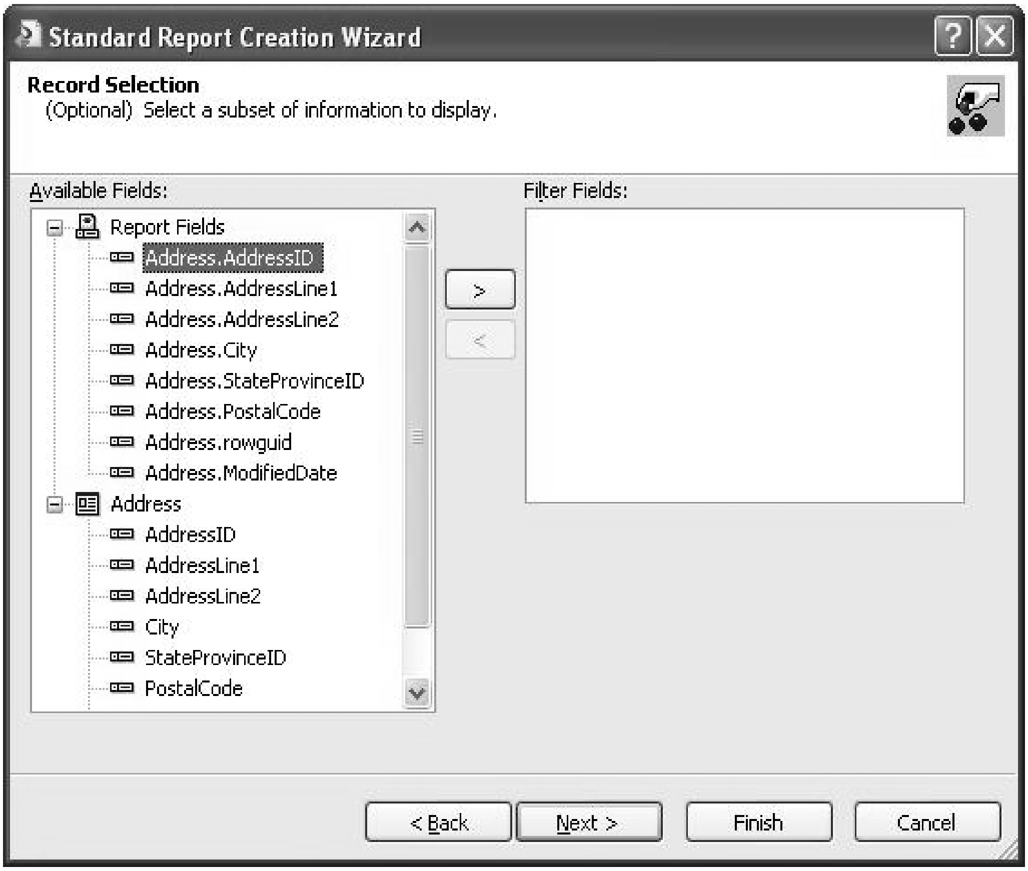
3. Clicking the **Next** button will open the **Group** tab, as shown in the following figure.



The Group Tab of the Standard Report Creation Wizard

Grouping is optional if you want to group the information on the report, select the field on which you want to group the report, otherwise click the **Next** button without selecting any field.

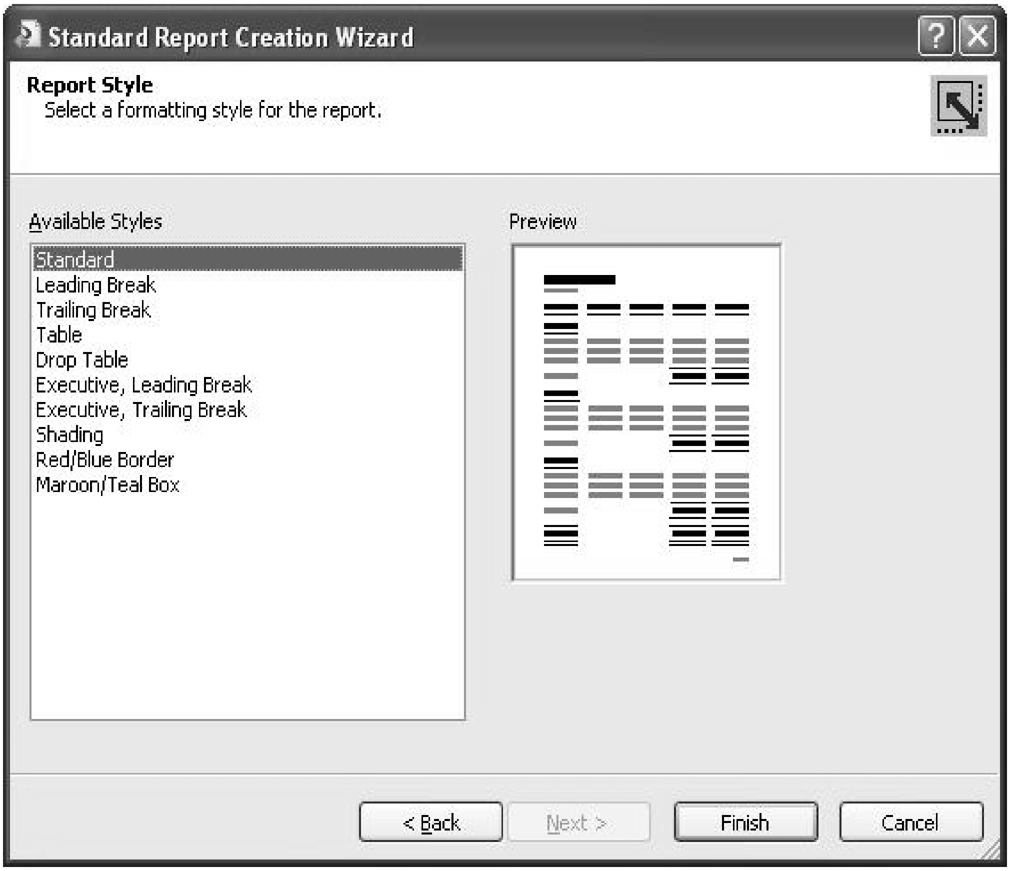
4. Click the **Next** button without selecting any field. Clicking the **Next** button will open the **Record Selection** tab, as shown in the following figure.



The Record Selection Tab of the Standard Report Creation Wizard

Record selection is optional, if you want to select a subset of information to display, select the required field, otherwise click the **Next** button without selecting any field.

5. Click the **Next** button without selecting any field. Clicking the **Next** button will open the **Report Style** tab, as shown in the following figure.



The Report Style Tab of the Standard Report Creation Wizard

6. Select the desired style and click the **Finish** button in the **Report Style** tab of the **Standard Report Creation Wizard**, this will complete the creation of the report.

Creating Crystal Reports from an Existing Report

To create a report based on the design of an existing report, perform the following steps:

1. Select the Crystal Reports in the **Add New Item** dialog box and click the **Add** button.
2. Select the **From an Existing Report** option from **Crystal Reports Gallery** and click the **OK** button. The **Open** dialog box is displayed.

3. Select the Crystal Reports file based on which you want to create the report and click the **Open** button.

The Report Designer is displayed with the design of the selected file. All the fields of the report from which you create a report are also added to the new report. In case a field is not required in the Crystal Reports, it can be deleted from the report by selecting it and pressing the Delete key.

When you have created Crystal reports, you can enhance it by using the Charts and Cross-tab Objects components.

Enhance the Crystal Reports

The presentation quality of a Crystal Reports can be enhanced by adding the following components to the Crystal Reports:

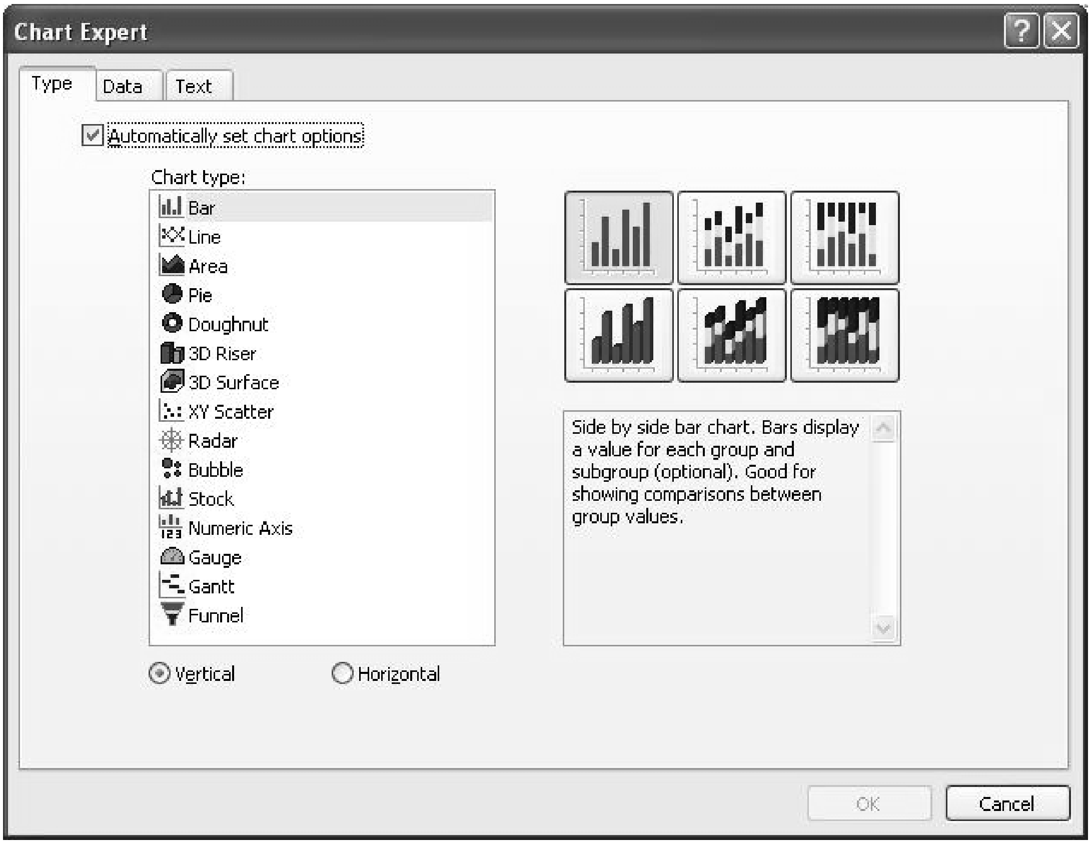
- Charts
- Cross-tab objects

Charts

Presenting data using charts allows easy analysis of the presented data. Generally, charts are created to present summarized fields. For example, presenting the number of products sold during each month would be best analyzed using charts. To insert a chart in a Crystal Reports, perform the following steps:

1. Right-click **Report Designer** and select **Insert** and then **Chart** from the shortcut menu.

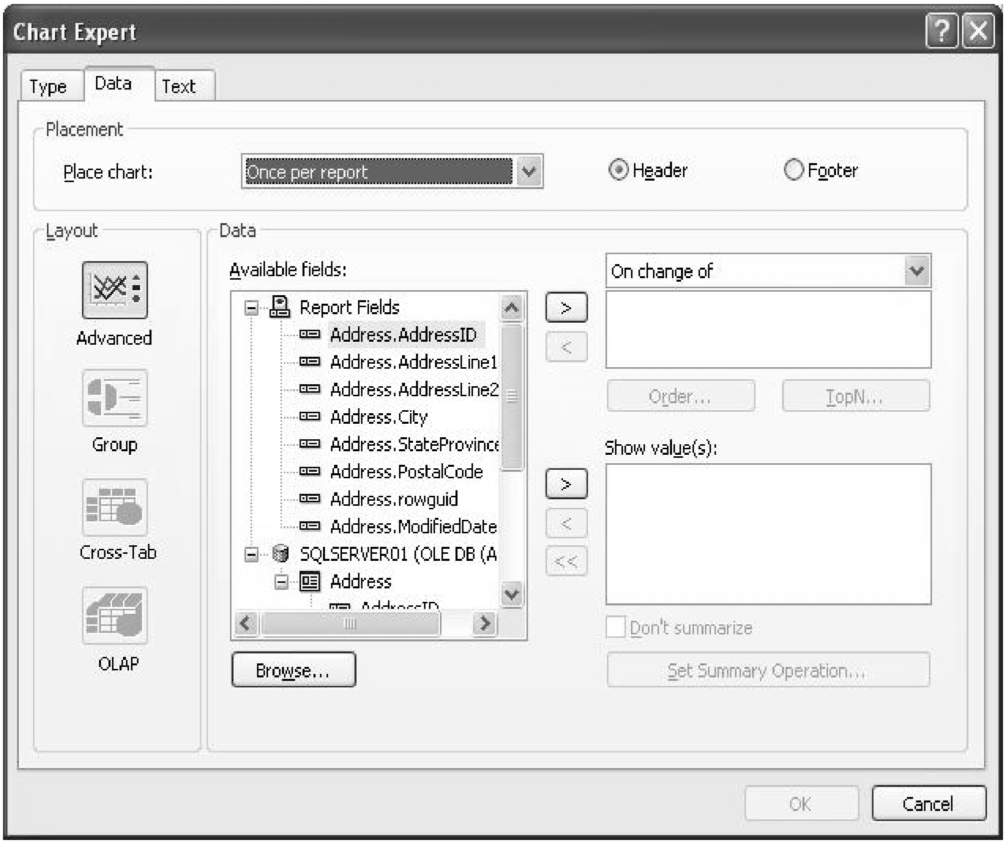
2. Specify the chart type in the **Chart Expert** dialog box by selecting the chart from the **Chart type** list, as shown in the following figure.



Side by side bar chart. Bars display a value for each group and subgroup (optional). Good for showing comparisons between group values.

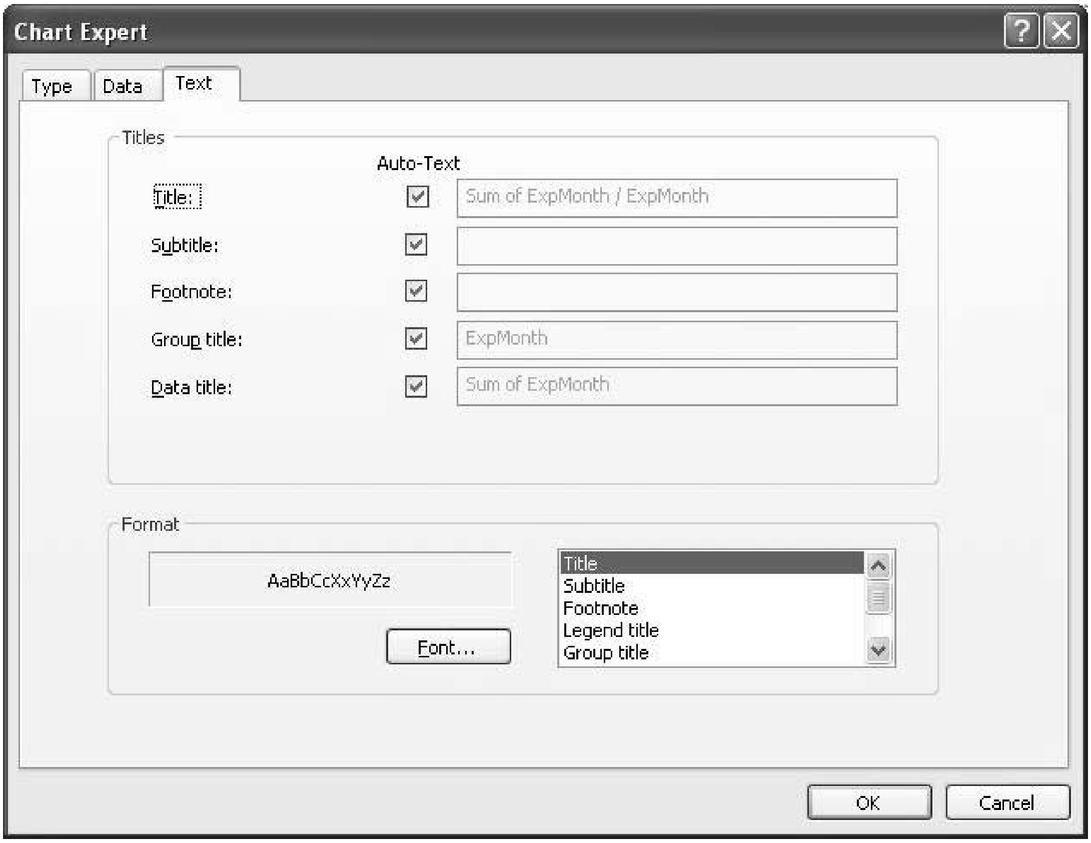
The Type Tab of the Chart Expert Dialog Box

3. Specify the fields for grouping data and the fields to be displayed for each change in the group field value by selecting the field and clicking the (>) button under the **Data** tab, as shown in the following figure.



The Data Tab of the Chart Expert Dialog Box

4. Specify titles for the report, such as the report **Title** and **Subtitle** in the **Text** tab, as shown in the following figure.



	Auto-Text	
Title:	✓	Sum of ExpMonth / ExpMonth
Subtitle:	✓	
Footnote:	✓	
Group title:	✓	ExpMonth
Data title:	✓	Sum of ExpMonth

Text Tab of the Chart Expert Dialog Box

Cross-Tab Objects

Cross-tab object is a grid and is inserted in a Crystal Reports when data is to be displayed in the form of compact rows, columns, and summary fields. This format helps to compare and identify trends. A sample output in a cross-tab format is shown in the following figure.

	P001	P002	P003	Total
Jan	2	0	5	7
Feb	5	4	21	30
Mar	1	6	12	19
Apr	10	14	9	33
Total	18	24	47	89

Sample of the Cross-Tab Format Output

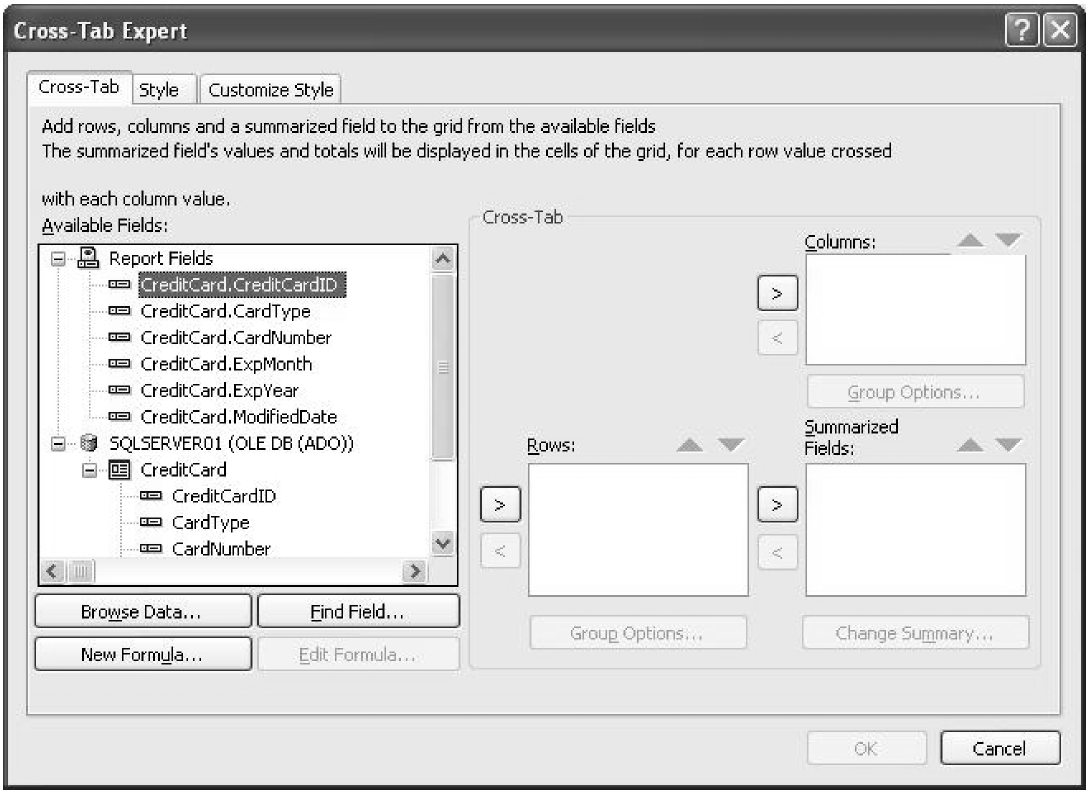
Note

Cross-tab objects are read-only. Width of all columns in a cross-tab must be same.

To insert a cross-tab object in Crystal Reports, you need to perform the following steps:

1. Right-click **Report Designer** and select **Insert** and then **Cross-Tab** from the shortcut menu.
2. Click the **Report Header** section of the **Report Designer** to create a cross tab.
3. Specify the fields to be displayed as rows, columns, and summarized fields under the **Cross-Tab** tab in the **Cross-Tab Expert** dialog box, as shown in the following figure.

Fields can be added to the **Rows**, **Columns**, and **Summarized Fields** lists by selecting the field and clicking the **Add Row**, **Add Column**, or **Add Summarized Field** button, as shown in the following figure.



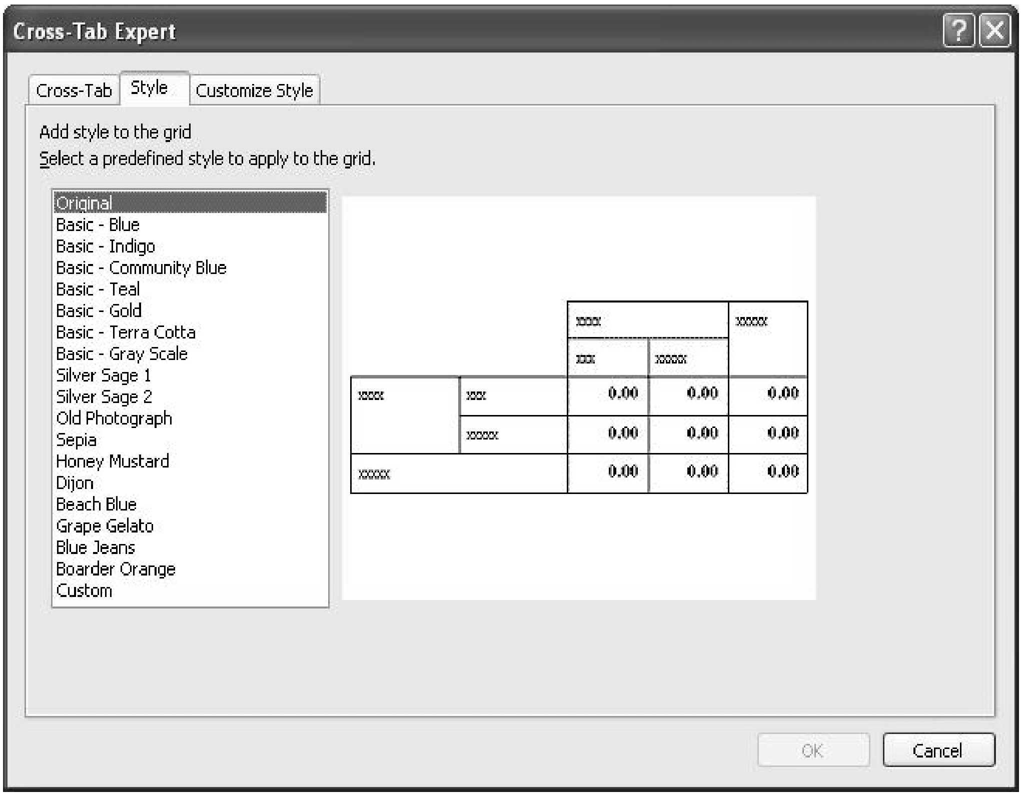
Add rows, columns and a summarized field to the grid from the available fields
The summarized field's values and totals will be displayed in the cells of the grid, for each row value crossed

with each column value.

- Report Fields
  - CreditCard.CreditCardID
  - CreditCard.CardType
  - CreditCard.CardNumber
  - CreditCard.ExpMonth
  - CreditCard.ExpYear
  - CreditCard.ModifiedDate
- SQLSERVER01 (OLE DB (ADO))
  - CreditCard
    - CreditCardID
    - CardType
    - CardNumber

The Cross-Tab Tab of the Cross-Tab Expert Dialog Box

4. Specify the style for the grid to display the data under the **Style** tab, as shown in the following figure.

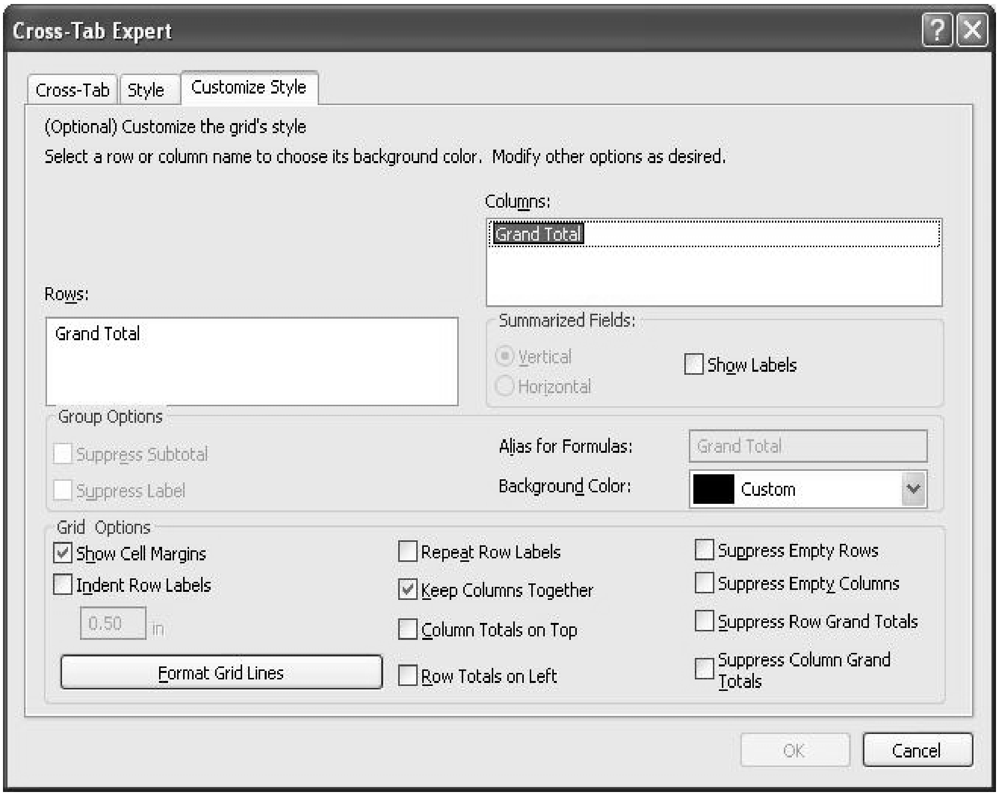


Add style to the grid
Select a predefined style to apply to the grid.

- Original
- Basic - Blue
- Basic - Indigo
- Basic - Community Blue
- Basic - Teal
- Basic - Gold
- Basic - Terra Cotta
- Basic - Gray Scale
- Silver Sage 1
- Silver Sage 2
- Old Photograph
- Sepia
- Honey Mustard
- Dijon
- Beach Blue
- Grape Gelato
- Blue Jeans
- Boarder Orange
- Custom

The Style Tab of the Cross-Tab Expert Dialog Box

5. Specify the background color for rows and columns of the grid under the **Customize Style** tab, as shown in the following figure.



(Optional) Customize the grid's style

Select a row or column name to choose its background color. Modify other options as desired.

Columns: Grand Total

Rows: Grand Total

Summarized Fields:

- Vertical (selected)
- Horizontal
- [ ] Show Labels

Group Options

- [ ] Suppress Subtotal
- [ ] Suppress Label

Alias for Formulas: Grand Total

Background Color: ■ Custom

Grid Options

- [✓] Show Cell Margins
- [ ] Indent Row Labels
- 0.50 in
- [ ] Repeat Row Labels
- [✓] Keep Columns Together
- [ ] Column Totals on Top
- [ ] Row Totals on Left
- [ ] Suppress Empty Rows
- [ ] Suppress Empty Columns
- [ ] Suppress Row Grand Totals
- [ ] Suppress Column Grand Totals

The Customize Style Tab of the Cross-Tab Expert Dialog Box

After Crystal Reports has been created it needs to be hoisted. The Crystal reports can be hoisted using the CrystalReportViewer control.

Implementing CrystalReportViewer Control in .NET Application

The CystalReportViewer is present as a control in the Toolbox and can be inserted into a Windows application by dragging the **CystalReportViewer** control from the Toolbox into the form.

The CrystalReportViewer control contains the following components:

- **Toolbar**: The toolbar of the CrystalReportViewer control contains the following icons:
  - **Go to First Page**: Used to move to the first page of the Crystal Reports.

    - **Go to Previous Page**: Used to move to the previous page of the Crystal Reports.
    - **Go to Next Page**: Used to move to the next page of the Crystal Reports.
    - **Go to Last Page**: Used to move to the last page of the Crystal Reports.
    - **Go to Page**: Used to move to the page specified by the user.
    - **Close Current View**: Used to close the currently open Crystal Reports view. This is used for groups and sub reports only.
    - **Print Report**: Used to print the Crystal Reports.
    - **Refresh**: Used to refresh the Crystal Reports view.
- **Export Report**: Used to export the Crystal Reports to other formats, such as Adobe Acrobat, Microsoft Excel, Microsoft Rich Text, HTML, and Microsoft Word.
- **Toggle Group Tree**: Used to toggle the state of the Group Tree to the opposite of the current state. For example, if the Group Tree is visible in the Crystal Reports, clicking the Toggle Group Tree hides the Group Tree.
- **Zoom**: Used to display the Crystal Reports in a customized size.
- **Search Text**: Used to search for a text in the Crystal Reports.
- **Group Tree**: Used to display the groups into which the Crystal Reports is divided. For example, if the Crystal Reports is grouped to display data on a weekly basis, the Group Tree would contain the weeks into which the Crystal Reports has been broken down.
- **Main Report Window**: Used to display the Crystal Reports. The data of the Crystal Reports can be viewed in the Main Report window.

You can create interaction between the **CrystalReportViewer** control and other controls on the Windows Form by handling the events of the Windows Form controls and the **CrystalReportViewer** control.

Some of the events of the **CrystalReportViewer** control that can be handled to add code and interact with other controls are listed in the following table.

Events	***Description***

Load	*This event is raised when the CrystalreportViewer control is loaded in memory, as shown in the following code:*
	`private void CrystalReportViewer1_Load(object sender, System.EventArgs e)`
	`{`
	`    label1.Text = "Report Viewer loaded";`
	`}`
	The preceding code displays the specified text in the Label control

Events	*Description*
	when the CrystalReportViewer control is loaded into the memory.

ReportRefresh	*This event is raised when the report view is refreshed. One of the methods to raise this event is to click the Refresh icon on the Crystal Reports toolbar, as shown in the following code:*
	 `private void CrystalReportViewer1_ReportRefresh (object source, CrystalDecisions.Windows.Forms.ViewerEventArgs e)`
	 `{`
	 `label1.Text = "Report Viewer Refreshed";`
	 `}`
	 The preceding code displays the specified text in the Label control when the Report view is refreshed.

Some Important Events of CystalReportViewer

Similarly, the events raised by other controls can also be used for interaction between the Crystal Reports and controls. For example, if you want to display a Crystal Reports selected by the user at run time, you write a code using the `OpenFileDialog` class to display the Crystal Reports files. Then you use the ReportSource property of the CrystalReportViewer control to display the selected Crystal Reports.

Hosting Crystal Reports

After the Crystal Reports has been created, it needs to be hosted in the Windows application so that the user can view the Crystal Reports at run time. Hosting a Crystal Reports in a Windows application consists of two tasks:

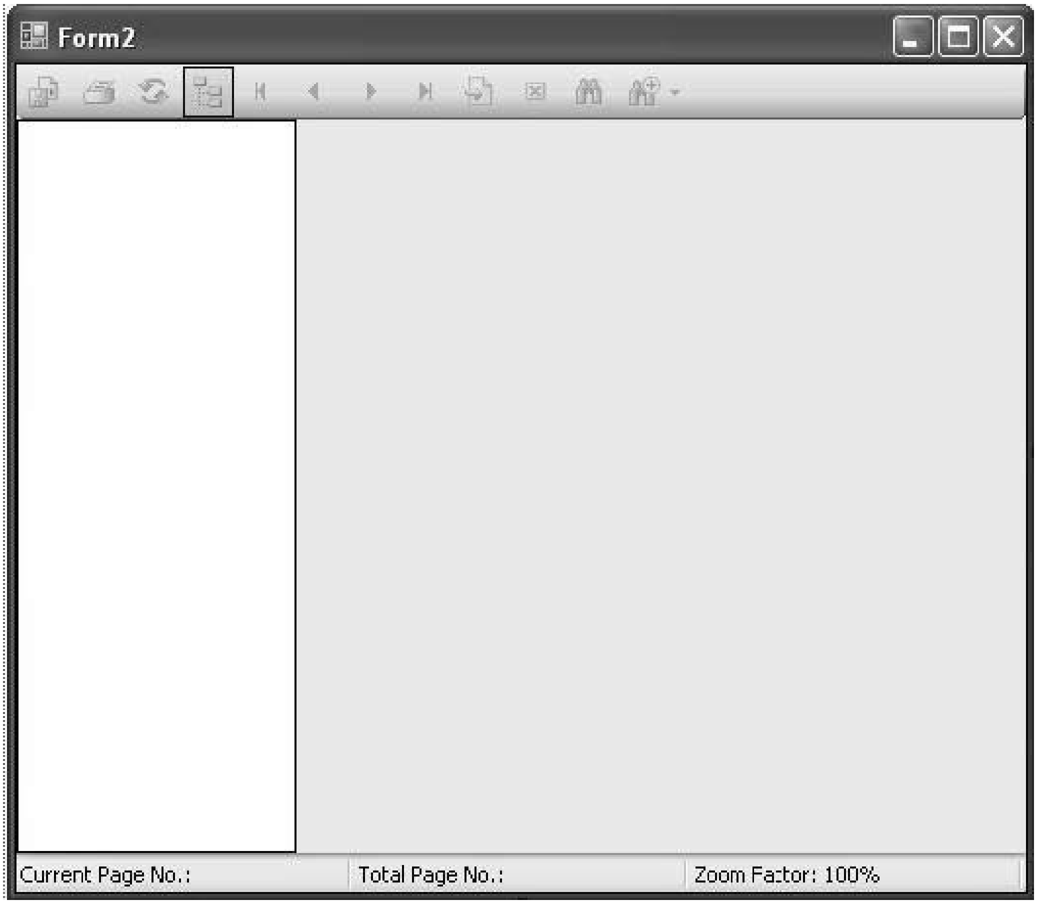
- Adding a CrystalReportViewer control to the Windows Form.
- Binding the CrystalReportViewer control to the created Crystal Reports.

Adding a CrystalReportViewer Control to the Windows Form

To add a CrystalReportViewer control to the Windows Forms, perform the following steps:

1. Drag the **CrystalReportViewer** control into the form from the **Windows Forms** tab of the **Toolbox** and name it as RevenueReportViewer.

2. Resize the inserted **CrystalReportViewer** control so as to display all the Crystal Reports data at run time, as shown in the following figure.



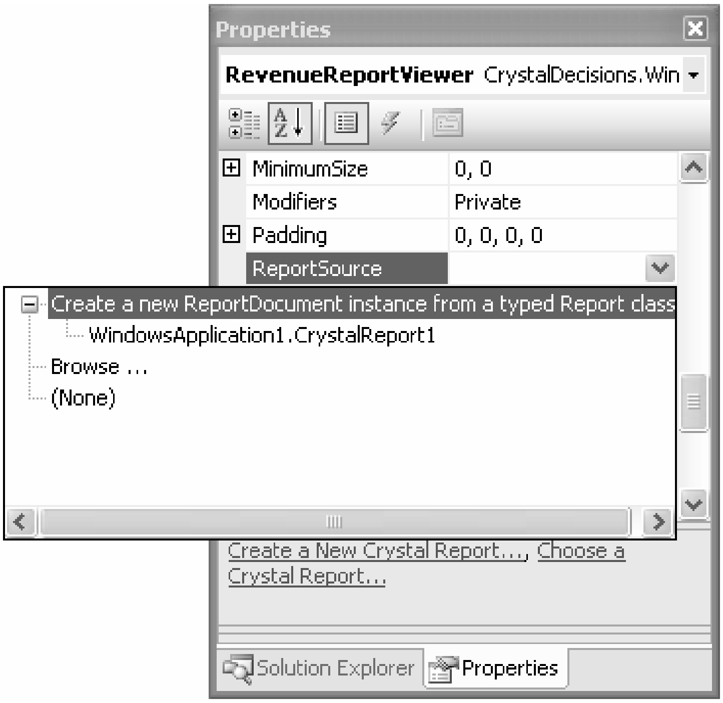
The Resized CrystalReportViewer Control in the Windows Form

Binding the CrystalReportViewer Control to the Created Crystal Reports

To display the Crystal Reports in the CrystalReportViewer control, it has to be bound to the CrystalReportViewer control. To do so, you need to perform the following steps:

1. Display the **Properties** window for the **CrystalReportViewer** control either by right-clicking the **CrystalReportViewer** control and selecting **Properties** from the shortcut menu or by selecting the **CrystalReportViewer** control and then selecting **Properties Window** from the **View** menu on the menu bar.

2. Click the **ReportSource** property under the **Data** category in the **Properties** window. Using the **ReportSource** property, you can specify the source from where the Crystal Reports would be accessed. The **ReportSource** property has two options, **None** and **Browse**. Click the **Browse** option to select the Crystal Reports to be opened, as shown in the following figure.


Properties
RevenueReportViewer CrystalDecisions.Win
MinimumSize 0, 0
Modifiers Private
Padding 0, 0, 0, 0
ReportSource
Create a new ReportDocument instance from a typed Report class
WindowsApplication1.CrystalReport1
Browse ...
(None)
Create a New Crystal Report..., Choose a Crystal Report...
Solution Explorer
Properties

The Properties Window for the CrystalReportViewer Control

3. Select any report and click the **Open** button in the **Open an Existing Crystal Reports dialog box**. The Crystal Reports is bound to the **CrystalReportViewer** control to display the data.

Just a minute:

Name the property which is used to specify the source from where the Crystal Reports would be accessed.

Answer:

The ReportSource property

After creating the Crystal Reports and connecting it to the CrystalReportViewer control, the report needs to be viewed.

Viewing Crystal Reports

To view the Crystal Reports data, select Debug from the menu bar. Then, select Start from the Debug menu. You can also run the application by clicking the Start icon on the Standard toolbar.

Practice Questions

1. Identify the category under which the ReportSource property of the CrystalReportViewer control appears in the property window.
   a. Data
   b. Layout
   c. Design
   d. Misc

2. Name the object which is inserted in a Crystal Reports when the data is to be displayed in the form of compact rows, columns, and summary fields.
   a. Charts
   b. Cross-tab objects
   c. ReportLabels
   d. Date and Time

3. Name the section of Crystal Reports that displays the actual values in the report.
   a. Report Footer
   b. Report Header
   c. Page Header
   d. Details

4. Identify the property which can be used to get or set a value indicating how large the pages will appear.
   a. AutoZoom
   b. Document
   c. Zoom
   d. AntiAlias

5. Which event of the PrintDocument object is used to specify the text which is to be printed?
   a. QueryPageSettings
   b. EndPrint
   c. BeginPrint
   d. PrintPage

Summary

In this chapter, you learned that:

- The default Print dialog box can be invoked either by adding a PrintDialog control and a PrintDocument control to the form or by creating an instance of the PrintDialog and `PrintDocument` classes.
- The `ShowDialog()` method is used to display the Print dialog box on the screen.
- The DialogResult property is used to trap the button that a user clicks in the Print dialog box
- To invoke the default Windows Page Setup dialog box, either a PageSetupDialog control is added to the form or an instance of the `PrintSetupDialog` class is created.
- The PrintPreviewDialog control displays how a document will look when printed.
- The PrintPreviewControl can be used to create a custom print preview interface.
- Crystal Reports is the standard reporting tool for Visual Studio .NET that is used to display data of presentation quality.
- Crystal Reports in VC# .NET supports two methods to access data from a data source:
  - The Pull model
  - The Push model
- The presentation quality of a Crystal Reports can be enhanced by adding the following components to the Crystal Reports:
  - Chart
  - Cross-tab objects
- A Crystal Reports can be hosted in a Windows Form by using the CrystalReportViewer control.
- An interaction can be created between the CrystalReportViewer control and other controls on the Windows Form by handling the events of the Windows Form controls and the CrystalReportViewer control.
- To display a Crystal Reports, it has to be bound to a CrystalReportViewer control.
- A Crystal Reports can be bound to a CrystalReportViewer control by setting the ReportSource property of the CrystalReportViewer control to the path of the Crystal Reports file.

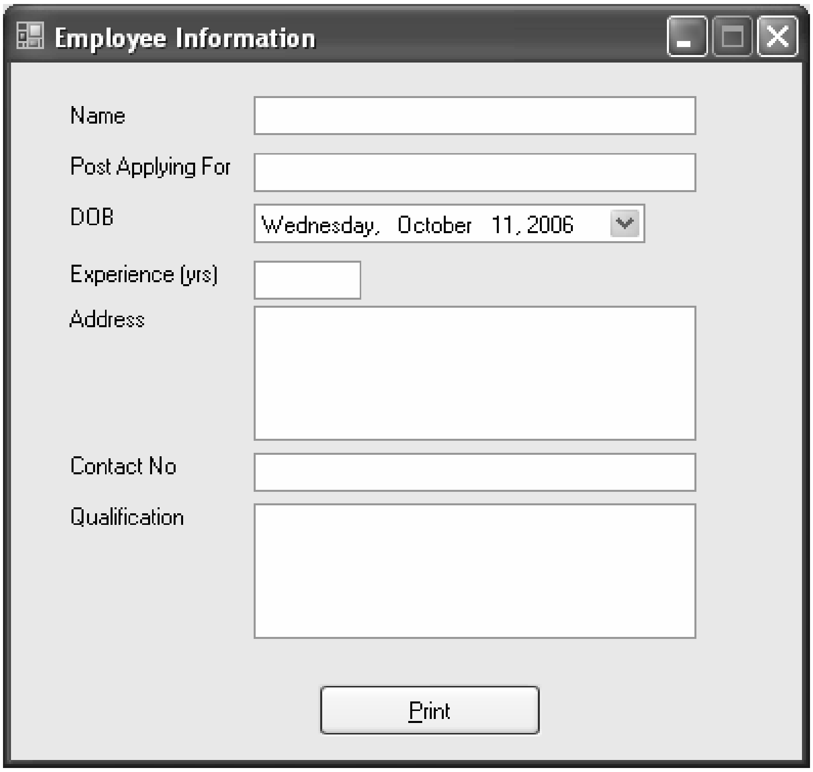
Exercises

Exercise1

A consultancy company, Sigma Pvt. Ltd. frequently conducts interviews. For uniformity and to save time, the company requires software that takes the information from candidates and prints that information.

Use PrintDialog and PrintDocument controls to implement printing.

The completed form should look like the following form.



Employee Information

Name	
Post Applying For	
DOB	Wednesday, October 11, 2006
Experience (yrs)	
Address	
Contact No	
Qualification	

Exercise2

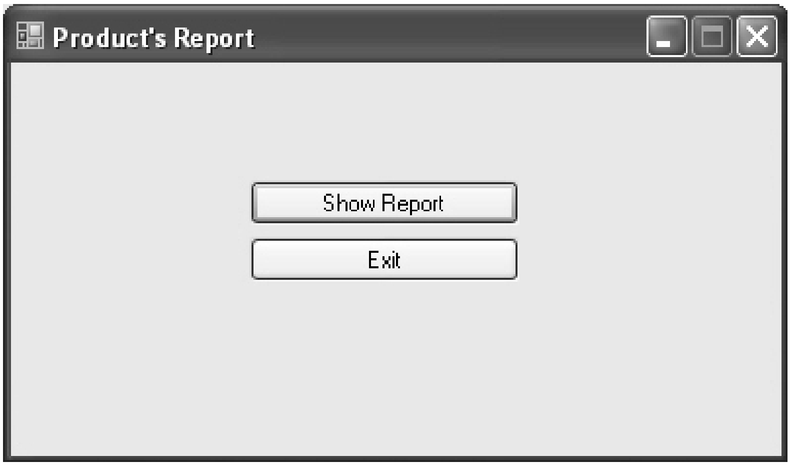
A company requires an application in which a form is displayed. The application should have two buttons, Exit and Show Report. On clicking the Show Report button a report of the products sold by the company should be displayed. The report must include the following fields from the Products table:

- ProductID
- ProductName
- QuantityPerUnit
- UnitPrice
- UnitsInStock
- ReorderLevel

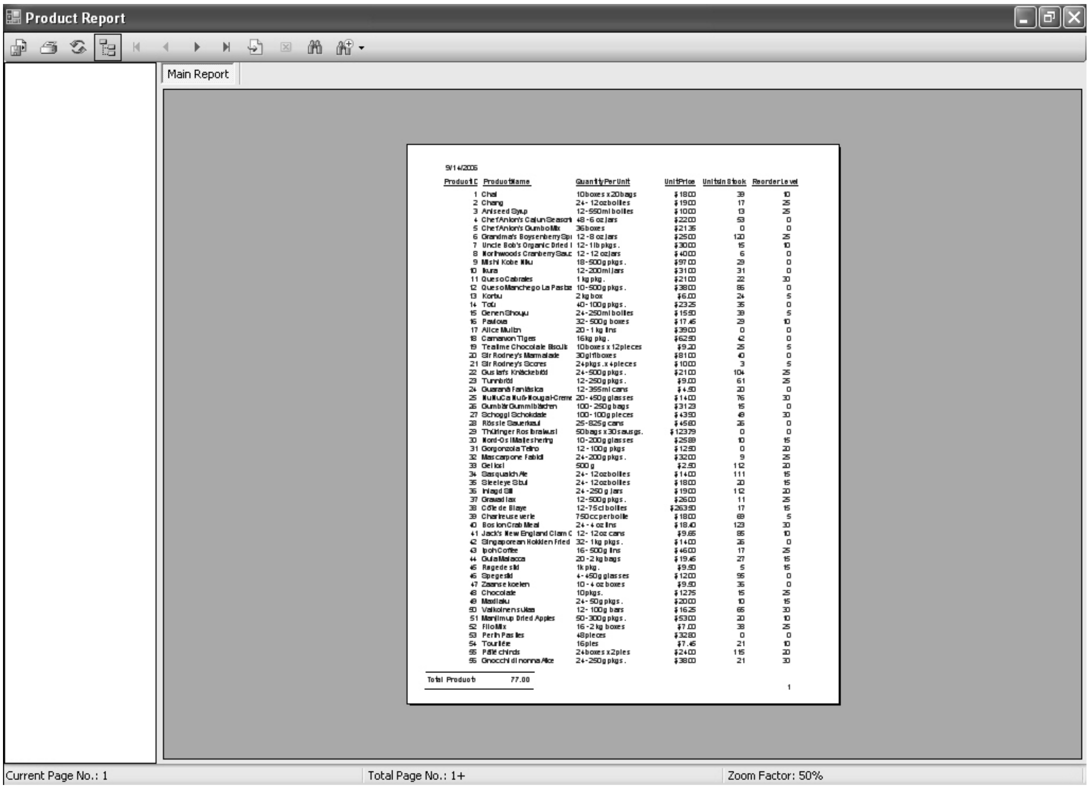
The company also wants the total number of products sold to be displayed in the Page Footer. For this purpose a formula field is to be used that counts the total number of ProductId.

The function of the Exit button is to close the application.

The first form of the application should look like the following form.



The report should look like as shown in the following form.



9/14/2006

ProductID	ProductName	QuantityPerUnit	UnitPrice	UnitsInStock	ReorderLevel
1	Chai	10 boxes x 20 bags	$18.00	39	10
2	Chang	24 - 12 oz bottles	$19.00	17	25
3	Aniseed Syrup	12 - 550 ml bottles	$10.00	13	25
4	Chef Anton's Cajun Seasoni	48 - 6 oz jars	$22.00	53	0
5	Chef Anton's Gumbo Mix	36 boxes	$21.35	0	0
6	Grandma's Boysenberry Spr	12 - 8 oz jars	$25.00	120	25
7	Uncle Bob's Organic Dried I	12 - 1 lb pkgs.	$30.00	15	10
8	Northwoods Cranberry Sauc	12 - 12 oz jars	$40.00	6	0
9	Mishi Kobe Niku	18 - 500 g pkgs.	$97.00	29	0
10	Ikura	12 - 200 ml jars	$31.00	31	0
11	Queso Cabrales	1 kg pkg.	$21.00	22	30
12	Queso Manchego La Pasto	10 - 500 g pkgs.	$38.00	86	0
13	Konbu	2 kg box	$6.00	24	5
14	Tofu	40 - 100 g pkgs.	$23.25	35	0
15	Genen Shouyu	24 - 250 ml bottles	$15.50	39	5
16	Pavlova	32 - 500 g boxes	$17.45	29	10
17	Alice Mutton	20 - 1 kg tins	$39.00	0	0
18	Carnarvon Tigers	16 kg pkg.	$62.50	42	0
19	Teatime Chocolate Biscuit	10 boxes x 12 pieces	$9.20	25	5
20	Sir Rodney's Marmalade	30 gift boxes	$81.00	40	0
21	Sir Rodney's Scones	24 pkgs. x 4 pieces	$10.00	3	5
22	Gustaf's Knäckebröd	24 - 500 g pkgs.	$21.00	104	25
23	Tunnbröd	12 - 250 g pkgs.	$9.00	61	25
24	Guaraná Fantástica	12 - 355 ml cans	$4.50	20	0
25	NuNuCa Nuß-Nougat-Creme	20 - 450 g glasses	$14.00	76	30
26	Gumbär Gummibärchen	100 - 250 g bags	$31.23	15	0
27	Schoggi Schokolade	100 - 100 g pieces	$43.90	49	30
28	Rössle Sauerkraut	25 - 825 g cans	$45.60	26	0
29	Thüringer Rostbratwurst	50 bags x 30 sausgs.	$123.79	0	0
30	Nord-Ost Matjeshering	10 - 200 g glasses	$25.89	10	15
31	Gorgonzola Telino	12 - 100 g pkgs	$12.50	0	20
32	Mascarpone Fabioli	24 - 200 g pkgs.	$32.00	9	25
33	Geitost	500 g	$2.50	112	20
34	Sasquatch Ale	24 - 12 oz bottles	$14.00	111	15
35	Steeleye Stout	24 - 12 oz bottles	$18.00	20	15
36	Inlagd Sill	24 - 250 g jars	$19.00	112	20
37	Gravad lax	12 - 500 g pkgs.	$26.00	11	25
38	Côte de Blaye	12 - 75 cl bottles	$263.50	17	15
39	Chartreuse verte	750 cc per bottle	$18.00	69	5
40	Boston Crab Meat	24 - 4 oz tins	$18.40	123	30
41	Jack's New England Clam C	12 - 12 oz cans	$9.65	85	10
42	Singaporean Hokkien Fried	32 - 1 kg pkgs.	$14.00	26	0
43	Ipoh Coffee	16 - 500 g tins	$46.00	17	25
44	Gula Malacca	20 - 2 kg bags	$19.45	27	15
45	Rogede sild	1k pkg.	$9.50	5	15
46	Spegesild	4 - 450 g glasses	$12.00	95	0
47	Zaanse koeken	10 - 4 oz boxes	$9.50	36	0
48	Chocolade	10 pkgs.	$12.75	15	25
49	Maxilaku	24 - 50 g pkgs.	$20.00	10	15
50	Valkoinen suklaa	12 - 100 g bars	$16.25	65	30
51	Manjimup Dried Apples	50 - 300 g pkgs.	$53.00	20	10
52	Filo Mix	16 - 2 kg boxes	$7.00	38	25
53	Perth Pasties	48 pieces	$32.80	0	0
54	Tourtière	16 pies	$7.45	21	10
55	Pâté chinois	24 boxes x 2 pies	$24.00	115	20
56	Gnocchi di nonna Alice	24 - 250 g pkgs.	$38.00	21	30

Total Products 77.00

1

Current Page No.: 1 Total Page No.: 1+ Zoom Factor: 50%

Note

Use the Products table of the Northwind database for the report. In the OLE DB Provider dialog box you have to choose Microsoft Jet 4.0 OLE DB Provider from the Provider list. You can also choose Access/Excel DAO and select the Northwind database in the Database Name text box.