C6109_AppendixC_CTP.qxd 18/7/06 02:55 PM Page 1

APPENDIX C

C

WORKING WITH REPORTS

After completing this appendix, you will be able to:

- Describe how to create a report
- Create a data source for a report
- Create report sections
- Create control instances on a report
- Use the `ReportViewer` control to display a report

»NOTE

The Report template and the `ReportViewer` control are new to Visual Studio 2005. In addition, the Report template and the `ReportViewer` control are not available with the Visual Basic Express edition.

INTRODUCTION TO PRINTING WITH THE REPORTVIEWER CONTROL

Chapter 11 discussed how to work with a database using bound control instances. In this appendix, you will learn how to print database data using a Report template along with the `ReportViewer` control.

Reporting in Visual Studio involves the following two types of components:

- Report Definition Files have a file suffix of .rdlc. These files are XML documents and define the structure and format of a report. .rdlc files are edited using the Report Designer.
- After a Report Definition File has been created, it is displayed to the end user in an instance of the `ReportViewer` control.

CREATING A REPORT

The process of creating a report is quite simple, as described in the following list:

1. First, one or more data sources need to be defined. Chapter 11 described how to create a data source. The process of creating a data source for a report is the same as the process of creating a data source for any database.
2. Second, you add a .rdlc file to the project using an installed template. After the .rdlc file has been added to the project, the report is configured using a visual designer much in the same way that a form's visual interface is created.
3. Finally, an instance of the `ReportViewer` control is created on the form and associated with the report.

DEFINING THE REPORT DATA SOURCE

The steps to define the data source for a report are the same as those discussed in Chapter 11. That is, you use the Data Source Configuration Wizard and the Data Sources window to add a data source to a project.

In this exploration exercise, you will add a data source that will be used by the report you will create later in the appendix.

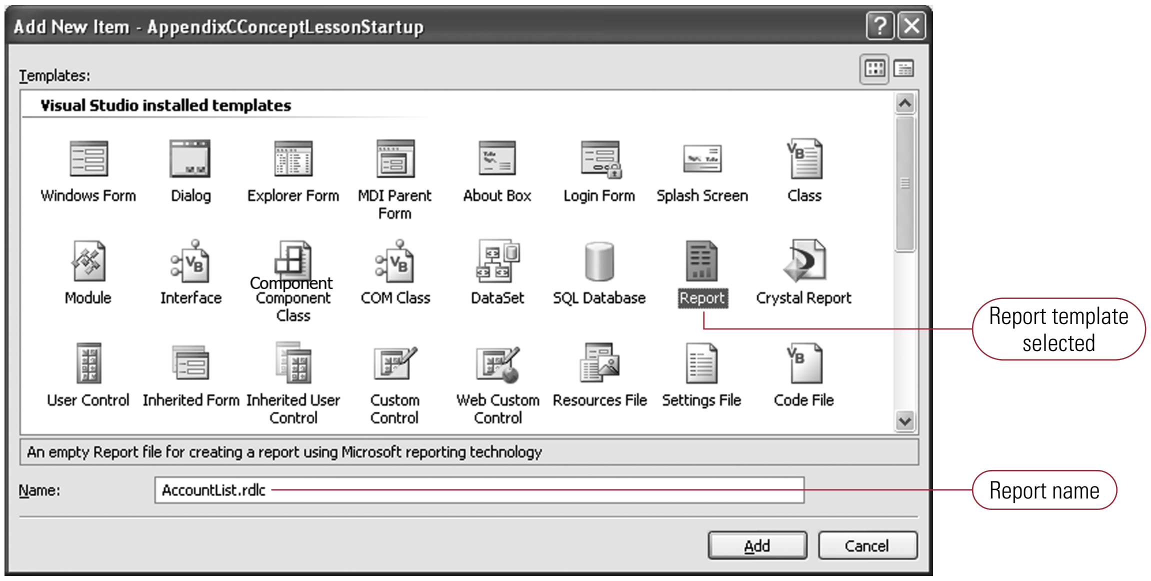
1. Start Visual Studio, if necessary, and open the solution file in the folder named **Appendix.C\AppendixCConceptLessonStartup**. Make sure that the form named **frmMain** is visible. Note that a completed version of the application appears in the folder Appendix.C\AppendixCConceptLessonPreview.
2. On the menu bar, click **Data**, **Add New Data Source** to activate the Data Source Configuration Wizard.
3. In the Choose a Data Source Type dialog box, select **Database**, and then click **Next** to activate the Choose Your Data Connection dialog box.
4. Click the **New Connection** button. Configure the connection to operate with an Access database, if necessary. Click the **Browse** button. Select the database file named **Appendix.C\Data\ AccountsReceivable.mdb**. When complete, the connection will appear in the Choose Your Data Connection dialog box.
5. Click **Next** in the Choose Your Data Connection dialog box. A message box appears asking you whether to copy the file to the current project. Save the connection string to the application configuration file by clicking **Yes** in the dialog box.
6. In the Save the Connection String to the Application Configuration File dialog box that opens, use the default options to save the connection string to the application's configuration file. Click **Next** to display the Choose Your Database Objects dialog box.
7. Select all of the fields from the table named **tblAccounts**. Use the default name for the DataSet.
8. Click **Finish** to configure the data source.
9. Next, you need to generate the strongly typed DataSet on the form. Open the Windows Forms Designer for the form named **frmMain**, if necessary. Using the Data Sources window, drag any field to the form to create the DataSet and BindingSource. An instance of the `BindingNavigator` control appears across the top of the form.

CREATING A NEW REPORT

The steps to create a new report are the same as the steps to create a form or other object. The Add New Item dialog box is used to create a report from a template.

In this exploration exercise, you will create a new report from a template.

1. On the menu bar, click **Project**, **Add New Item** to display the Add New Item dialog box.
2. Click the **Report** template, and then set the name to **AccountList.rdlc**, as shown in Figure C-1.

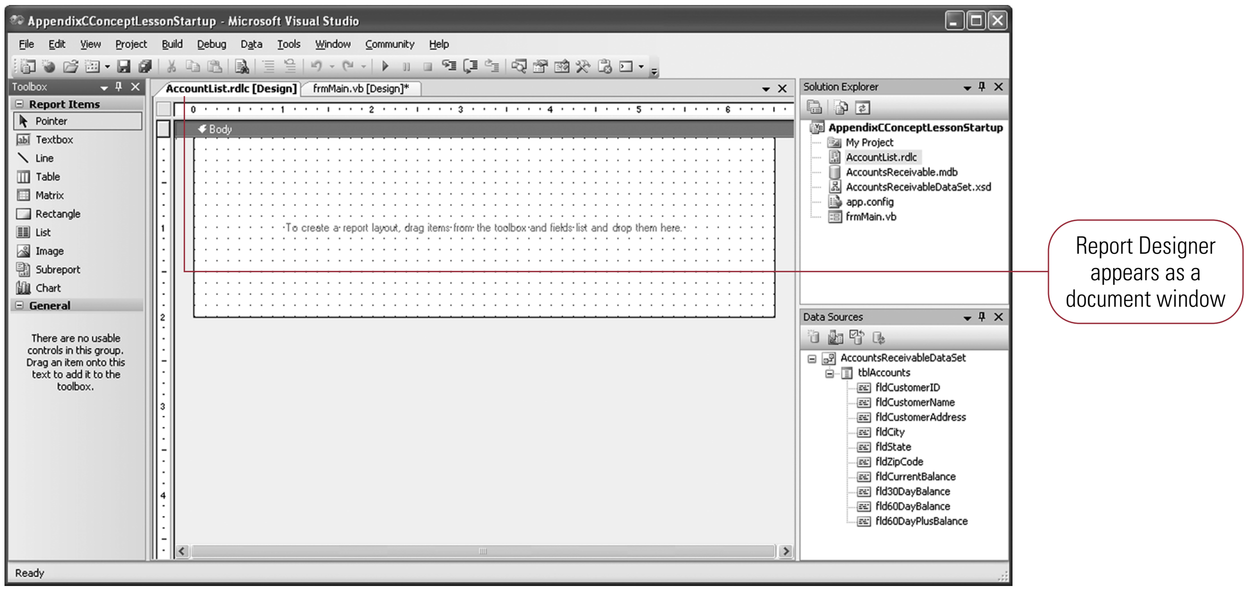


An empty Report file for creating a report using Microsoft reporting technology

Name: AccountList.rdlc

Figure C-1: Add New Item dialog box

3. Click **Add** to add the report to the solution. After the report has been added to the solution, the Report Designer appears, as shown in Figure C-2.



There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

To create a report layout, drag items from the toolbox and fields list and drop them here.

Figure C-2: Report Designer

REPORT DESIGNER CONTROLS

As shown in Figure C-2, the Report Designer has a visual surface in which you create a report similar to the way you create a form with the Windows Forms Designer. The controls used to create reports work much differently than the controls that work with forms. That is, even though a report has a control named `Textbox`, it is a different control than the `TextBox` control used with a form.

You will learn about the following Report Designer controls as you work through the steps in this appendix:

- The `Textbox` control displays a single item of data. It can be used to display a prompt, a field, or calculated data.
- The `Line` control displays a visible line on the report.
- The `Rectangle` control displays a visible rectangle on the report.
- The `Image` control displays a graphical image on the report.
- The `Table` control is used to create a tabular report made up of rows and columns.

Control instances are created on a report in the same way that control instances are created on a form. That is, a control instance is created by clicking the desired control in the Toolbox, and then drawing the control instance on the report. After a control instance has been created on a report, it can be configured using the Properties window.

REPORT SECTIONS

By default, a report has a single section named Body. However, it is possible to create two additional report sections:

- The Page Header section is used to create a region that appears at the top of every page when the report is generated. To add a page header to a report, click Report, Page Header.
- The Page Footer section is used to create a region that appears at the bottom of every page when the report is generated. To add a page footer to a report, click Report, Page Footer.

NOTE

Unlike the other applications you have created in this book, you need not write any code to create a report. All of the logic to print records and control pagination is controlled by the report itself.

The `Page Header` and `Page Footer` objects support properties. The `PrintOnFirstPage` property controls whether the header or footer is printed on the first page. The `PrintOnLastPage` property controls whether the header or footer is printed on the last page.

In this exploration exercise, you will add page headers and page footers to the report.

1. Click **Report**, **Page Header** to add a page header to the report.
2. Click **Report**, **Page Footer** to add a page footer to the report. The Report Designer with a page header and page footer is shown in Figure C-3.

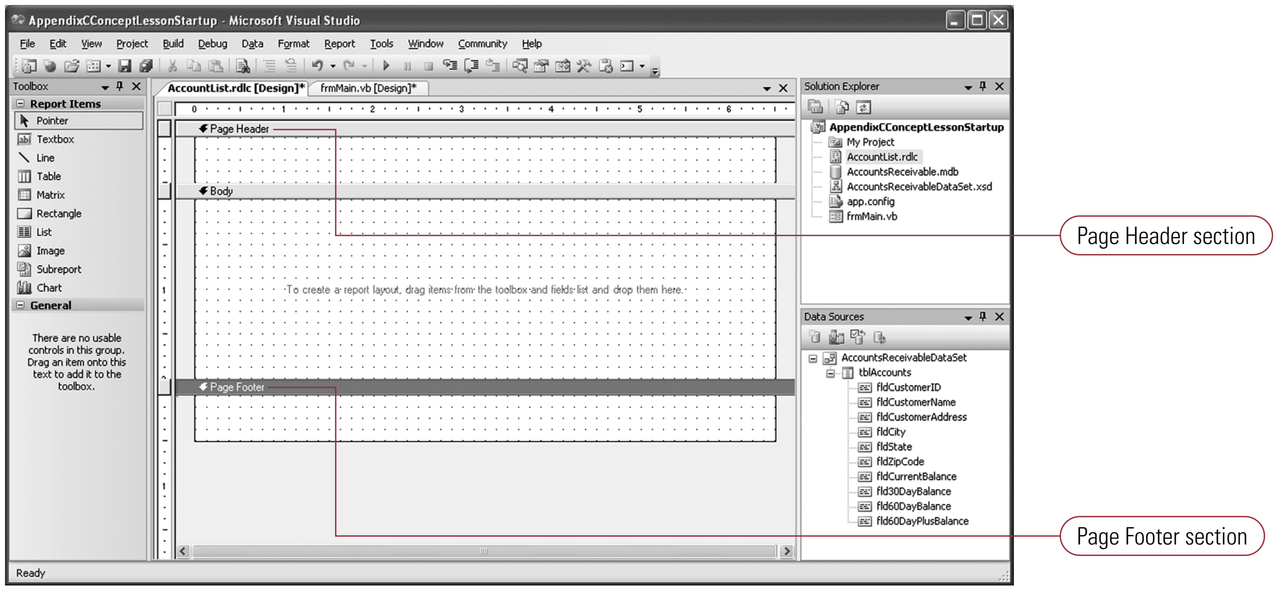

AppendixCConceptLessonStartup - Microsoft Visual Studio
File Edit View Project Build Debug Data Format Report Tools Window Community Help
Toolbox
Report Items
Pointer
Textbox
Line
Table
Matrix
Rectangle
List
Image
Subreport
Chart
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
AccountList.rdlc [Design]*
frmMain.vb [Design]*
Page Header
Body
To create a report layout, drag items from the toolbox and fields list and drop them here.
Page Footer
Solution Explorer
AppendixCConceptLessonStartup
My Project
AccountList.rdlc
AccountsReceivable.mdb
AccountsReceivableDataSet.xsd
app.config
frmMain.vb
Data Sources
AccountsReceivableDataSet
tblAccounts
fldCustomerID
fldCustomerName
fldCustomerAddress
fldCity
fldState
fldZipCode
fldCurrentBalance
fld30DayBalance
fld60DayBalance
fld60DayPlusBalance
Ready
Page Header section
Page Footer section

Figure C-3: Report with a page header and a page footer

As shown in Figure C-3, the report now has three sections. The height of each section can be changed by dragging the horizontal bar that separates each section.

CREATING CONTROL INSTANCES ON A REPORT

Having created the report and its page header and page footer, the next step is to create and configure the control instances on the report.

THE LINE CONTROL

As its name implies, the `Line` control is used to create a visible line on a report. The following properties define the visual appearance of the line:

- » The `LineColor` property defines the color of the line.
- » The `LineStyle` property defines the appearance of the line.
- » The `LineWidth` property defines the thickness of the line.

THE RECTANGLE CONTROL

The `Rectangle` control is used to create a visible rectangle on the report. The following properties define the visual appearance of the rectangle:

- » The `BorderColor` property defines the color of the rectangle's border.
- » The `BorderStyle` property defines the appearance of the rectangle.

» The BorderWidth property defines the thickness of the border surrounding the rectangle.

» The BackgroundColor property is used to fill the region of the rectangle with a background color.

THE IMAGE CONTROL

The Image control is used to display images on a report. Images can be stored in the database, in an external file, or can be embedded into the report itself. To embed images into a report, click Report, Embedded Images to display the Embedded Images dialog box shown in Figure C-4.

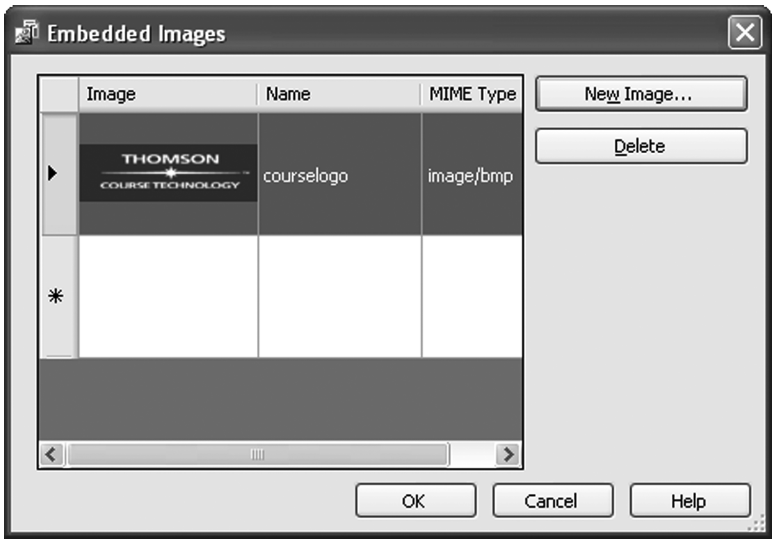


	Image	Name	MIME Type
▶	THOMSON COURSE TECHNOLOGY	courselogo	image/bmp
*			

Figure C-4: Embedded Images dialog box

To add an embedded image to a report, click the New Image button. In the Import Image dialog box that opens, select the image. After an image has been embedded into a report, the Value property of the Image control can be set to the image name. In addition, the Source property should be set to Embedded. The image will appear inside of the control instance created on the report.

THE TEXTBOX CONTROL

The `Textbox` control has various purposes, as described in the following list:

- It can be used to display static text that does not change. In this situation, the `Textbox` has the same purpose as a `Label` control applicable to a form.
- It can be used to display the value of a database field.
- It can be used to display the result of a calculation.

The `Textbox` control used with reports works much differently than the `TextBox` control used with the Windows Forms Designer. The data displayed in the `Textbox` control is stored in the `Value` property instead of the `Text` property. The `Value` property always contains an expression, which begins with an equal sign. The following code segment shows how to store a literal value in a text box:

```
="Employee Report"
```

The preceding expression stores the literal value “Employee Report” in the text box.

In addition to storing literal values in a `Textbox` control instance, it is possible to display the value of a database field, as shown in the following expression:

```
=Fields!fldCustomerID.Value
```

The `Fields` collection is used to reference the fields in the data source. An exclamation point separates the `Fields` collection and the field name, which in the preceding expression is “fldCustomerID”. A period separates the field name and the property name, which in the preceding expression is the `Value` property.

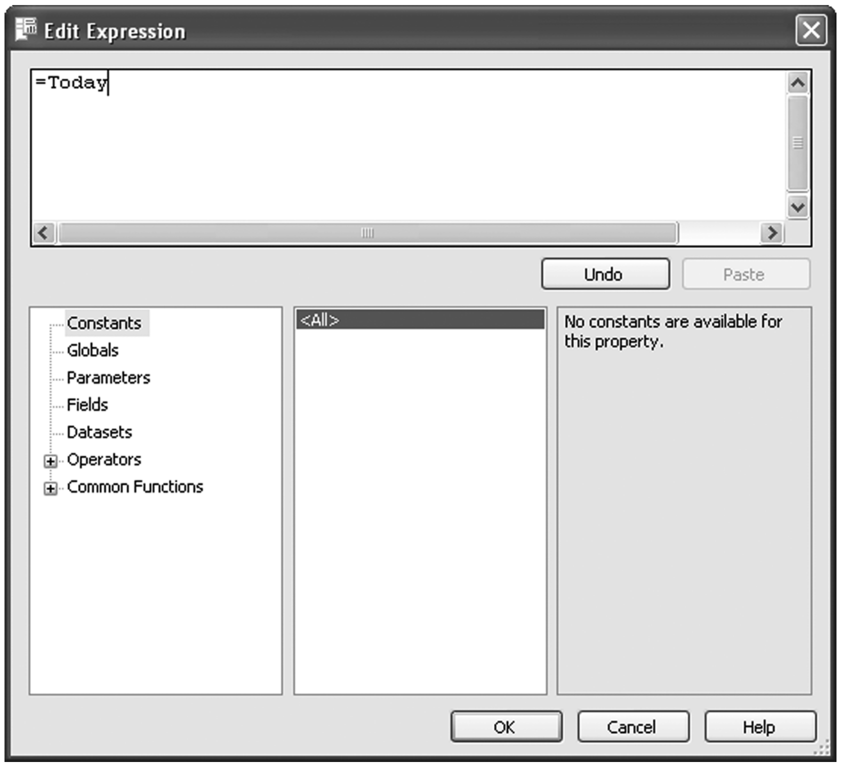
Note that expressions can call functions. For example, the following expression calls the `SUM` function to summarize all of the values across a group (multiple rows):

```
=SUM(Fields!fldCurrentBalance.Value)
```

The preceding expression calculates the sum for all rows of the field named fldCurrentBalance. Again, you need not write any code to calculate the total of the rows. This logic is controlled by the reporting system itself.

BUILDING EXPRESSIONS

The Edit Expression dialog box shown in Figure C-5 helps you build expressions.



```
=Today
```

No constants are available for this property.

Figure C-5: Edit Expression dialog box

As shown in Figure C-5, the Edit Expression dialog box has a drill-down interface. The left pane lists Operators and Common Functions; the right pane displays the applicable operators and functions. The current expression appears in the text box at the top of the dialog box. Figure C-5 shows an expression to call the `Today` function, which returns the current date. Note that all expressions begin with an equal (=) sign.

THE GLOBALS OBJECT

Reports support another object called the `Globals` object, which, in turn, supports properties used by most reports, as described in the following list:

- The `PageNumber` property displays the current report page number.
- The `TotalPages` property displays the total number of report pages.
- The `ReportName` property displays the name of the report.

The following expression shows how to use these properties:

```
=Globals.PageNumber & " of " & Globals.TotalPages
```

The preceding expression displays the current page number and total number of pages. The ampersand (&) operator is the string concatenation operator as usual.

In this exploration exercise, you will create and configure instances of the `Line`, `Rectangle`, `Image`, and `Textbox` controls on the report.

1. Click **Report**, **Embedded Images** to display the Embedded Images dialog box.
2. Click the **New Image** button. In the Import Image dialog box, select the image named **Appendix.C\Data\CourseLogo.bmp**.
3. Click **OK** to close the Embedded Images dialog box.
4. Using the Toolbox, create an instance of the `Image` control in the Page Header section.
5. Set the Source property to **Embedded**.
6. Using the Properties window, set the Value property to **courselogo** using the drop-down list.
7. Create an instance of the `Line` control in the Page Header section.
8. Create an instance of the `Textbox` control in the Page Header section. Using the Properties window, set the Value property to **="Accounts Receivable Report"**. This is an expression to display a literal value. Set the font size to **18** points.
9. Create another instance of the `Textbox` control in the Page Footer section. Right-click the control instance, and then click **Expression**. In the Edit Expression dialog box that opens, expand the **Common Functions** folder, and select **Date & Time**. Double-click the **Today** function. The function is added to the text box at the top of the dialog box. Click **OK** to close the Edit Expression dialog box.
10. Create another instance of the `Textbox` control in the Page Footer. Set the expression as follows to display the page number:

```
=Globals.PageNumber & " of " & Globals.TotalPages
```

Figure C-6 shows the Report Designer with the control instances created on the report.

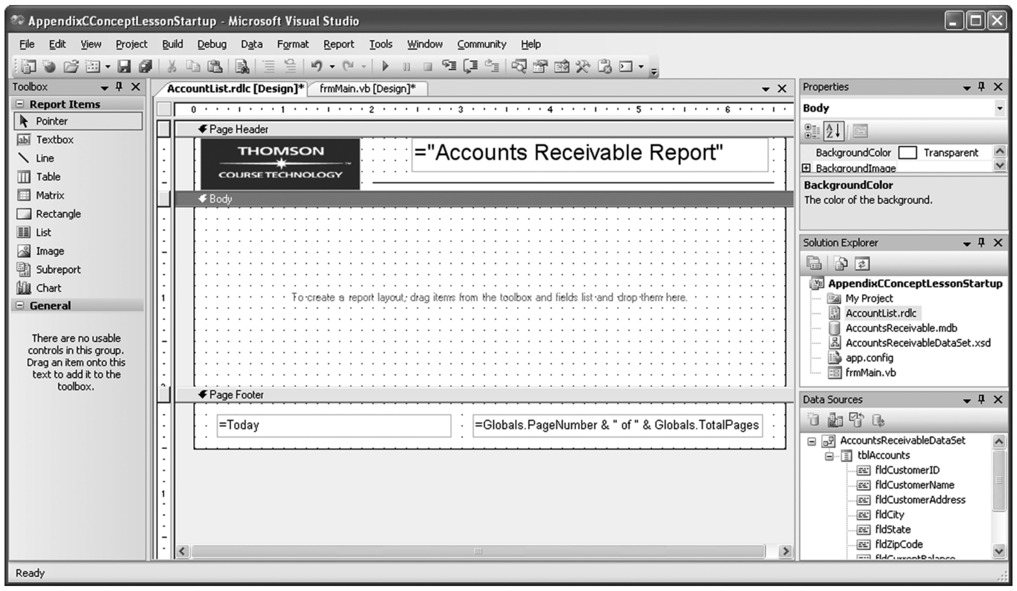

AppendixCConceptLessonStartup - Microsoft Visual Studio
File Edit View Project Build Debug Data Format Report Tools Window Community Help
Toolbox
Report Items
Pointer
Textbox
Line
Table
Matrix
Rectangle
List
Image
Subreport
Chart
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
AccountList.rdlc [Design]*
frmMain.vb [Design]*
Page Header
THOMSON COURSE TECHNOLOGY
="Accounts Receivable Report"
Body
To create a report layout, drag items from the toolbox and fields list and drop them here.
Page Footer
=Today
=Globals.PageNumber & " of " & Globals.TotalPages
Properties
Body
BackgroundColor Transparent
BackgroundImage
BackgroundColor
The color of the background.
Solution Explorer
AppendixCConceptLessonStartup
My Project
AccountList.rdlc
AccountsReceivable.mdb
AccountsReceivableDataSet.xsd
app.config
frmMain.vb
Data Sources
AccountsReceivableDataSet
tblAccounts
fldCustomerID
fldCustomerName
fldCustomerAddress
fldCity
fldState
fldZipCode
Ready

Figure C-6: Report Designer with lines, images, and text boxes

DISPLAYING REPEATING DATA

Many business reports are columnar reports that display repeating data. That is, one or more lines are printed on a report for each database record. In addition, many reports also contain subtotals and totals. To create a tabular report, you use the `Table` control.

Just as a report is divided into sections, so too is the `Table` control. The following list describes the default sections appearing in the `Table` control:

» The Header section contains column headers.

» The Detail section displays a row for each row in the underlying data source.

» The Footer section is used to display totals.

Each cell in the table grid contains an instance of the `Textbox` control. This is the same `Textbox` control discussed in the preceding section. Thus, you store expressions in the `Value` property for each `Textbox` control to display the data in the table.

The appearance of the `Table` control can be configured in a visual way, as follows:

» To create columns in the table, right-click on the header above the column and click Insert Column to the Left or Insert Column to the Right. By default, the table has three columns.

» To remove a column, right-click the column header, and click Remove Columns.

» To configure a cell, enter an expression into the cell. Format each text box using the Properties window.

In this exploration exercise, you will use the `Table` control to display repeating data.

1. Create an instance of the **Table** control in the Body section of the report.
2. Insert additional columns so that the table contains six columns. To insert a column, click a column header to select it. Then, click **Insert a Column to the Left**.
3. Each of the cells in the table is an instance of the `Textbox` control. Thus, to configure the table, you enter expressions in the `Value` property for each `Textbox` control instance. Use the information in Table C-1 to build the expressions for each cell in the first row of the table. Each expression stores a literal value in the column header.

Column	Expression
Column 1	`="Customer ID"`
Column 2	`="Customer Name"`
Column 3	`="Current Balance"`
Column 4	`="30 Days"`
Column 5	`="60 Days"`
Column 6	`="90 Days"`

Table C-1: Column header expressions

4. Next, the expressions for the detail rows need to be created. These expressions will display the data for each detail row in the underlying data source. Use the information in Table C-2 to build the expressions for each cell in the second row of the table. These expressions display detail data for the respective fields.

Column	Expression
Column 1	`=Fields!fldCustomerID.Value`
Column 2	`=Fields!fldCustomerName.Value`
Column 3	`=Fields!fldCurrentBalance.Value`
Column 4	`=Fields!fld30DayBalance.Value`
Column 5	`=Fields!fld60DayBalance.Value`
Column 6	`=Fields!fld60DayPlusBalance.Value`

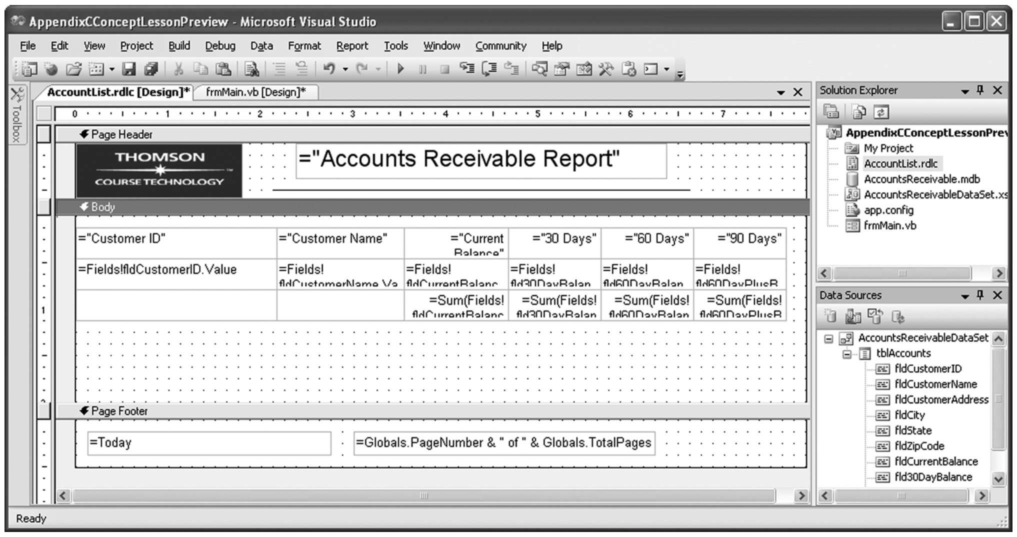
Table C-2: Detail expressions

5. Finally, you need to enter the expressions in the table footer to display the totals for the numeric columns. Use the information in Table C-3 to build the expressions for each cell in the third row of the table. Note that the first two columns do not have any expressions because totals are not calculated for those columns.

Column	Expression
Column 1	`N/A`
Column 2	`N/A`
Column 3	`=Sum(Fields!fldCurrentBalance.Value)`
Column 4	`=Sum(Fields!fld30DayBalance.Value)`
Column 5	`=Sum(Fields!fld60DayBalance.Value)`
Column 6	`=Sum(Fields!fld60DayPlusBalance.Value)`

Table C-3: Table footer expressions

6. It is possible to format a text box using a format string. Format strings work the same way with a text box as they do when calling the `ToString` method. For example, the `"c"` specifier denotes a currency format string. In columns 3 through 6, as detailed in Table C-3, activate the Properties window for each text box and set the `Format` property to `="c"`.
7. Finally, resize the columns by dragging the column separator. When complete, the table should look similar to Figure C-7.



="Accounts Receivable Report"

="Customer ID"	="Customer Name"	="Current Balance"	="30 Days"	="60 Days"	="90 Days"
=Fields!fldCustomerID.Value	=Fields! fldCustomerName.Va	=Fields! fldCurrentBalanc	=Fields! fld30DayBalan	=Fields! fld60DayBalan	=Fields! fld60DayPlusB
		=Sum(Fields! fldCurrentBalanc	=Sum(Fields! fld30DayBalan	=Sum(Fields! fld60DayBalan	=Sum(Fields! fld60DayPlusB

=Today

=Globals.PageNumber & " of " & Globals.TotalPages

Figure C-7: Configured table

You have completed configuring the table. The next step in the process is to create an instance of the `ReportViewer` control on the form and associate the control instance with the report.

USING THE REPORTVIEWER CONTROL

At this point, you have created a report. However, you have not defined a means to display the report. To display a report, you create an instance of the `ReportViewer` control on the form, and configure it to display the report that you need.

In this exploration exercise, you will create an instance of the `ReportViewer` control and configure it to display the report you just created.

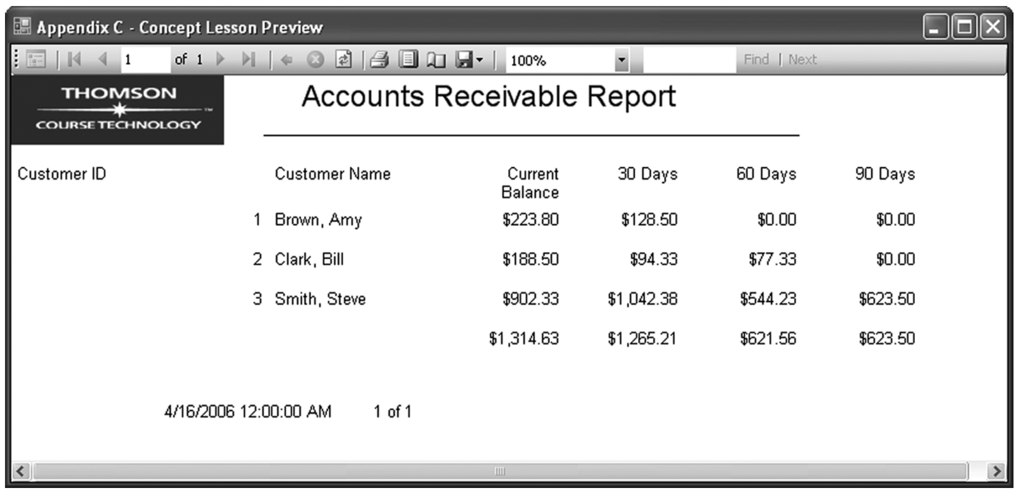
1. Activate the Windows Forms Designer for the form named **frmMain.vb**.
2. In the Data section of the Toolbox, click the **ReportViewer** control and create an instance of the control on the form.
3. Using the Properties window, set the Dock property to **Fill** so that the control instance fills the region of the form.

4. While the control instance is selected, activate the Report Viewer Tasks dialog box by clicking the arrow near the upper-right part of the control instance.
5. Using the Choose Report combo box, select the report named **AppendixC ConceptLessonStartup.rdlc**.
6. Click **Choose Data Sources** in the Report Viewer Tasks dialog box. In the dialog box that appears, set the Data Source Instance to **TblAccountsBindingSource**.

At this point, the `ReportViewer` control instance is associated with the report. Running the application will cause the data from the data source to be loaded into the report and the report to be generated.

In this exploration exercise, you will test the report.

1. Run the application. A message appears in the `ReportViewer` control instance to indicate that the report is being generated. When report generation is complete, the report appears, as shown in Figure C-8.



Accounts Receivable Report

Customer ID	Customer Name	Current Balance	30 Days	60 Days	90 Days
1	Brown, Amy	$223.80	$128.50	$0.00	$0.00
2	Clark, Bill	$188.50	$94.33	$77.33	$0.00
3	Smith, Steve	$902.33	$1,042.38	$544.23	$623.50
		$1,314.63	$1,265.21	$621.56	$623.50

4/16/2006 12:00:00 AM 1 of 1

Figure C-8: Completed report

This appendix has introduced the basics of creating a simple tabular report and displaying it with an instance of the `ReportViewer` control. Visual Studio allows you to create many different types of reports in addition to tabular reports. For instance, it's also possible to create reports with subtotals and perform complex formatting.

APPENDIX SUMMARY

- There are two basic steps to create a report. First, you create a report from a template. A report is nothing more than an XML document having a file extension of .rdlc. Second, you create an instance of the `ReportViewer` control and associate it with a report.
- A report gets its data from a data source. The data source for a report is no different than the data source used to work with a database. The data source is typically created using the Data Source Configuration Wizard.
- A report is created from a template just as a form is created from a template. Click Project, Add New Item to display the Add New Item dialog box. Select the Report template and give the report a name.
- After a report has been created, control instances can be created on the report in much the same way that a report is created on a form. Click the desired control in the Toolbox. Draw the visible region of the control instance in the desired report section.
- By default, a report has a single section named Body. In addition, it is possible to create a Page Header and Page Footer section. These sections print at the top and bottom of each page.
- The `Line` control is used to draw a visual line on the report. The `Rectangle` control is used to draw an outlined or filled rectangle on the report.
- The `Image` control is used to display an image on the report. Images can be retrieved from the data source or can be embedded directly into the report. The Embedded Images dialog box is used to embed images into the report itself. The `Value` property of the `Image` control contains a reference to the image.
- The `Textbox` control has multiple purposes. It can be used to display a literal text string or data from a data source. The `Value` property of the control instance contains an expression. The expression is evaluated, and the result is displayed in the control instance. Expressions can be created by hand or by using the Edit Expression dialog box.
- The `Globals` object has properties used to display common report information such as the `PageNumber` and `TotalPages`.
- The `Table` control is used to create a tabular report. By default, the `Table` control has three sections. The first section contains the column headers. The second section contains the detail lines, and the third section, called the footer, is used to display totals. Each cell in the table is a text box. Each text box, in turn, can contain an expression.
- After a report has been created, it is displayed using an instance of the `ReportViewer` control. Create an instance of the `ReportViewer` control on the form, and then associate the control instance with the desired report.