ACTIVE REPORTS HANDBUCH

Dieses Handbuch beschreibt den Designer von ActiveReports2 von DataDynamics, wie er in IOS2000 Anwendung findet. Leider ist das Handbuch zur Zeit nur in englischer Sprache verfügbar.

Using Documentation

Document Conventions

Documentation Overview

Sample Reports

Document Conventions

This book uses the following typographical conventions

Convention	Description
bold	Programming language terms in text
italic	In Syntax italics indicate placeholders for information you would supply. Dialog box elements are displayed in italics
EmbeddedCaps	Language terms are capitalized for readability. The language is not case sensitive.
SMALL CAPS	File Names in text, command names and button names.
`monospace font`	Code Syntax and Examples
Keyboard Shortcuts	• **Delete** means the Delete or Del key on your numeric keypad. • **Escape** means the Escape or Esc key on your keyboard. • **Enter** means Enter, Return, or the Carriage Return key on your keyboard. • **Ctrl-Key, Shift Key, and Alt-Key** are two key combinations. Press and hold the first key then press the second key and release. For example, Ctrl-Z, press and hold the Control key and press the Z letter key on your keyboard.
Mouse Actions	• **Click** – Click the primary (left) mouse button once. • **Right-Click** – Click the secondary (right) mouse button once. • **Double-click** – Click the primary (left) mouse button twice without moving the mouse while clicking. • **Shift-Click** – Press and hold the SHIFT key on your keyboard and click the primary (left) mouse button.

USING ACTIVEREPORTS

This chapter introduces the ActiveReports design environment and describes how to add and modify controls in your report. It includes a detailed description of ActiveReports user interface elements, including the menus and toolbars items and preference settings.

In addition, you will learn about some basic operations while using ActiveReports, such as, selecting and moving, formatting and aligning controls in your report.

ActiveReports includes a Report Wizard that steps you through creating simple reports without writing any code. The wizard is integrated into the Visual Basic environment as an add-in module. This chapter describes where to find the wizard and how to use it in creating your reports.

Creating a New Report

User Interface

Adding Controls to your Report

Quick Start: Creating your First Report

Common Operations

Adding OLE Objects and ActiveX Controls

Report Explorer

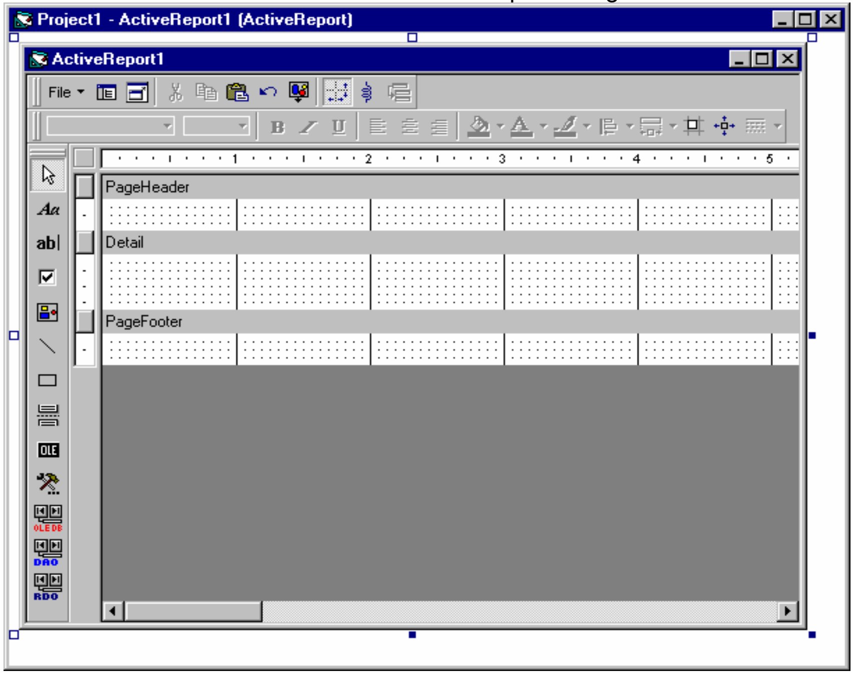
Fields List

Using ActiveReports Wizard

Creating a New Report

To create a new report in your project:

1. Choose **Project** from Visual Basic menu.
   Note: Visual Basic adds the menu item either directly under the Project menu or under **More Active Designers** menu item.
2. Choose **Add Data Dynamics ActiveReports** from the submenu.
3. Visual basic creates a new window with the report designer inside it as shown below.



User Interface

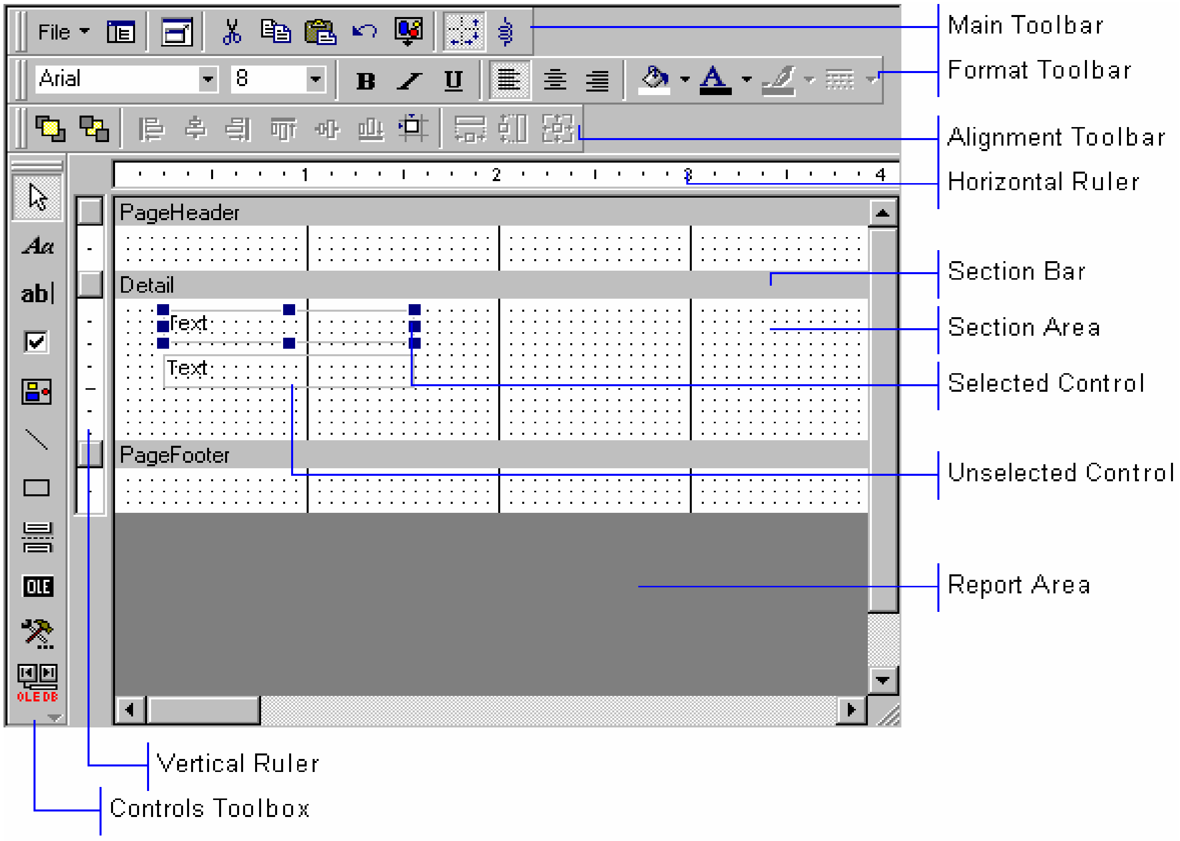
ActiveReports user interface is similar to Visual Basic's form designer interface. It leverages your current knowledge and provides full integration within your Visual Basic environment.

ActiveReports Window

Toolbars

Context Menus

ActiveReports Window



Toolbars

You can easily customize the toolbars in ActiveReports. You can re-arrange the buttons and menu options, create your own custom toolbars, hide or display toolbars, dock or move toolbars.

Moving a toolbar

1. Click the grab handle ‖ on a docked toolbar or the title bar on a floating toolbar.
2. Drag the toolbar to a new location.
   The edges of the window act as magnets and allow the toolbar to become a docked toolbar.

Toolbars Context Menu

To access the toolbars context menu, click the right mouse button anywhere in the toolbars area.

The context menu allows you to show or hide toolbars by selecting the toolbar name from the menu. In addition, you can customize the toolbars or create new toolbars from the Customize option on that menu.

Customized toolbar settings are saved in user.tb file and they are restored the next time you start ActiveReports.

Main Toolbar

Button	Name
☰	Report Explorer – shows or hides the report explorer tree and the fields list.
🗖	Full-Screen – maximizes the ActiveReports designer to a full-screen view outside the Visual Basic IDE.
✂	Cut – Cuts the selected controls to the clipboard.
⎘	Copy – Copies the selected controls to the clipboard.
📋	Paste – Pastes the contents of the clipboard into the current selected section.
↶	Undo – the last action.

	View Grid – turns the grid display on or off.
	Reorder groups – displays the groups order dialog.

Format Toolbar

Button	Name
Arial ▾	Font – sets the typeface of the selected label or field control.
8 ▾	Size – sets the font size of the selected label or field control.
B	Bold – sets the bold typeface on or off.
I	Italic –sets the italic typeface on or off.
U	Underline – sets the underline typeface on or off.
	Text Align Left – aligns the text left within the control area.
	Text Align Center – align the text centered within the control area.
	Text Align Right – aligns the text right within the control area.
	BackColor – sets the background style to normal and the background color of the selected control to the specified color.
A	ForeColor – sets the text color of the selected control to the specified color.
	LineColor – sets the line color for the selected line control.
	LineStyle – sets the line style of the selected line control.

Controls Toolbox

Button	Name
	Select – allows you select controls on the report
Aa	Label – Insert a new static label control
ab|	Field – insert a field textbox, bound to a database field or unbound.
☑	Checkbox – insert a field checkbox, bound to a database field or unbound.
	Image – insert a picture, loaded from a file.
	Line – insert a line control.
□	Shape – insert a rectangle, circle or square shape.
	PageBreak – insert a page break within a section.
	Subreport – insert a subreport control to link to another report.
OLE	OLE Object – insert an OLE object, bound to a database field or unbound.
	ActiveX Control – insert an ActiveX control.
OLEDB	OLEDB Data Control – define an OLEDB data source.
DAO	DAO Data Control – define a DAO data source.
RDO	RDO Data Control – define an RDO data source.

Alignment Toolbar

Button	Name
	Bring to Front – Brings the selected controls to the top Z-Order
	Send to Back – Sends the selected controls to the bottom Z-Order

	Snap to Grid – Turns controls snap-to-grid on or off.
	Align Left – aligns selected controls to the same left coordinate of the last selected control.
	Align Center – aligns selected controls to the same center coordinate of the last selected control.
	Align Right – aligns selected controls to the same right coordinate of the first selected control.
	Align Top – aligns selected controls to the same top coordinate of the last selected control.
	Align Middle - aligns selected controls to the same middle coordinate of the last selected control
	Align Bottom - aligns selected controls to the same bottom coordinate of the last selected control
	Align to Grid – aligns the selected controls to the closest grid point.
	Make Same Width – makes all selected controls the same width as the last selected control.
	Make Same Height – makes all selected controls the same height as the last selected control.
	Make Same Size – makes all selected controls of the same height and width as the last selected control.
	Displays the position of the selected control within its parent section.
	Displays the dimensions of the selected control.

Print Preview Toolbar

Button	Name
	Table of Contents
Print...	Print Report
	Zoom Out/In
100 % ▾	Zoom
	Page Up/Down
2/3	Page p Of n
← →	History Back/Next

Context Menus

You can access context menus by clicking the right (secondary) mouse button on the (dark-colored) report area, the section area, the section bar or a control placed in any of the sections.

Menu Item	Function
Insert > Group Header/Footer	Adds a new group header / footer pair to the report.
Insert > Page Header / Footer	Adds a new page header / footer pair to the report. A single pair is allowed per report. This option is disabled once the first pair is added.
Insert > Report Header / Footer	Adds a new report header / footer pair to the report. A single pair is allowed per report. This option is disabled once the first pair is added.
Delete Section	Deletes the current selected section from the report. This option does not apply to the Detail section in the report.
Reorder Groups	This option is available when more than one group section are added to the report. It displays a dialog box to allow changes to the nesting order of the group sections in the report.

Cut	Cuts the selected control to the clipboard.
Copy	Copies the selected control to the clipboard
Paste	Pastes the contents of the clipboard into the current section.
Bring to Front	Brings the selected control to the top of the Z-Order
Send to Back	Sends the selected control to the bottom of the Z-Order
Align	Aligns controls to any of their vertical or horizontal coordinates.
Size	Sizes the controls to same width, height or both.
Format Border	Displays the border dialog box.

Adding Controls to Your Report

To add print controls to your report:

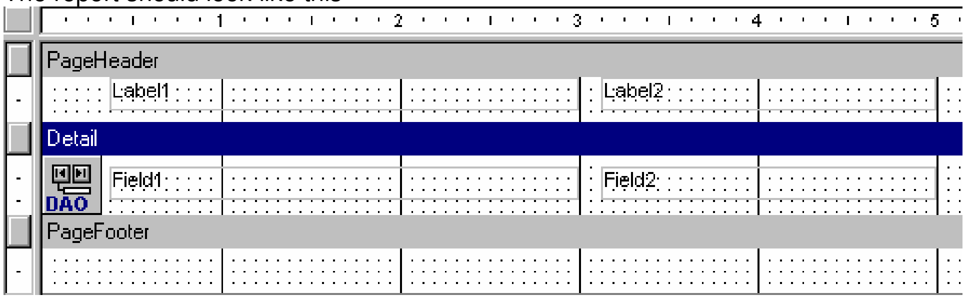
1. Click the control you want to add in the toolbox.
2. Move the mouse pointer to the section where you want to add the control; the mouse pointer will change to a cross hair.
3. Click and drag the mouse to size the rubber band to the desired size of the control.
4. The control is placed at the specified location and the toolbox current selection changes back to a Select pointer .

Note: *To add multiple copies of the same control, you can hold the Ctrl key while selecting the control from the toolbox and placing the controls in the section.*

Note: *Line controls can be set to draw horizontally or vertically by holding the Ctrl (horizontal) or Shift (vertical) key while clicking and dragging the mouse.*

Quick Start: Creating Your First Report

1. Start Visual Basic
2. Create a new Standard EXE project
3. Select Project > Add ActiveX Designer > Data Dynamics ActiveReports
4. Select and place a **DAO Data Control** DAO in the Detail section.
5. Drag the Page Footer section bar upward to size the Detail section to a one-line height.
6. Click on the DAO Data Control to select it.
7. Set the **DatabaseName** property to *<VB Path>\NWIND.MDB*
8. Set the **RecordSource** property to *Customers*
9. Place 2 label controls Aa horizontally in the Page Header section.
10. Place 2 text controls horizontally in the Detail section.
11. The report should look like this



12. Set the Label1 **caption** property to *Company Name*
13. Set the Label2 **caption** property to *Phone Number*
14. Set the Field1 **DataSource** property to *DAODataControl1* (select from dropdown)

15. Set the Field1 **DataField** property to *CompanyName* (select from dropdown)
16. Set the Field2 **DataSource** property to *DAODataControl1* (select from dropdown)
17. Set the Field2 **DataField** property to *Phone* (select from dropdown)
18. Add a button control to Form1.
19. In the Command1_Click event add the following code:

    ```
    ActiveReport1.Show
    ```
20. Run the project.
21. Click on Command1 button to preview your report.

Common Operations

Select Multiple Controls

Move and Size Controls

Sizing Sections

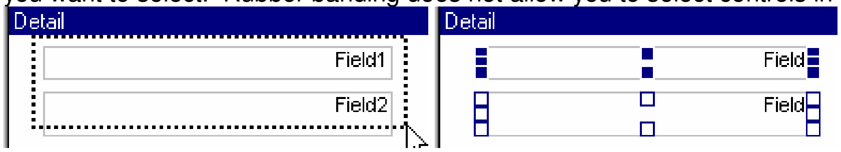
Control Alignment and Sizing

Formatting Controls

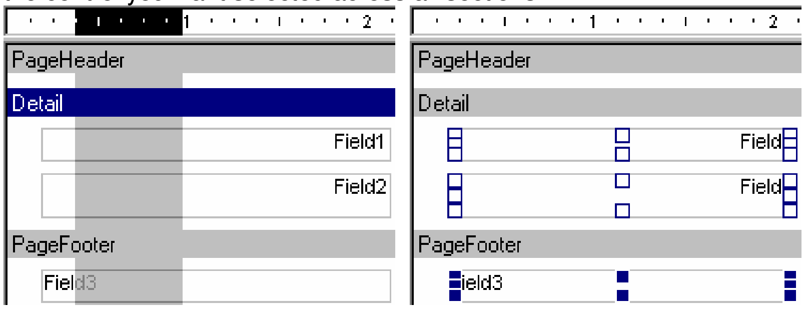
Select Multiple Controls

You can select multiple controls and then move, copy or do other things with them as a group. There are three methods to select more than one control:

1. Hold down the Shift or Ctrl key while you click on the control that you want to select.
2. Click in an empty area, and then draw a “rubber-banding” rectangle around all the controls that you want to select. Rubber banding does not allow you to select controls in different section.

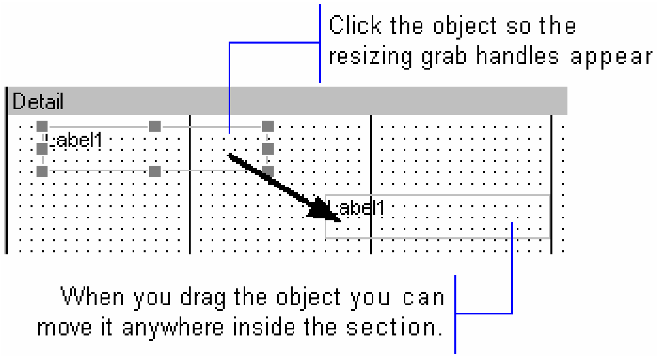


3. Click on either the horizontal or the vertical ruler and drag the pointer to draw a shadow covering the control you want selected across all sections.



Move and Size Controls

To move a control or a set of selected controls: select the controls you need to move and drag them with the mouse.



Click the object so the resizing grab handles appear

When you drag the object you can move it anywhere inside the section.

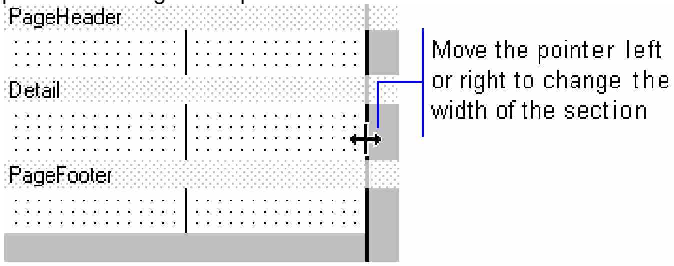
You can also adjust the size and location of controls precisely by specifying their coordinate properties in Visual Basic's property editor.

Note: *When you size a control beyond the boundaries of a section, the section will adjust to contain the control's new size.*

Sizing Sections

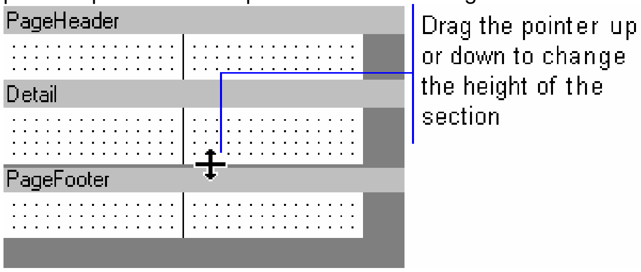
You can change the width and height of the sections in your report. The height of each section can be modified individually. However, you can only change the width of all sections at the same time, changing the width of sections will change the PrintWidth of the report.

- To change the width of the sections, place the pointer at the right edge of the section. Click and drag the pointer left or right to expand or shrink the width of the section.



Move the pointer left or right to change the width of the section

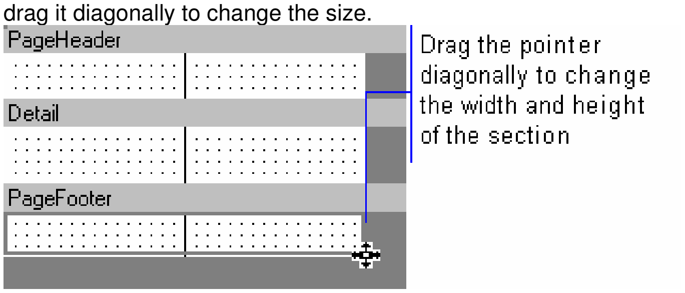
- To change the height of a section, place the pointer at the bottom edge of the section. Click and drag the pointer up or down to expand or shrink the height of the section.



Drag the pointer up or down to change the height of the section

- You can change the section height by using the vertical ruler thumb ▤ and dragging it up or down to adjust the height.
- Double clicking on the section thumb allows you to quickly set the section height to perfectly fit its contents.

- To change both width and height of a section, place the pointer at the lower-right corner of the section and drag it diagonally to change the size.

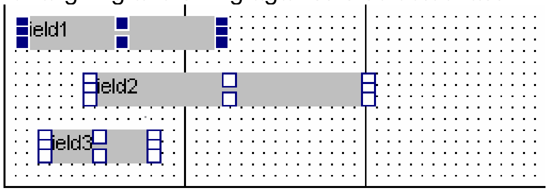


Drag the pointer diagonally to change the width and height of the section

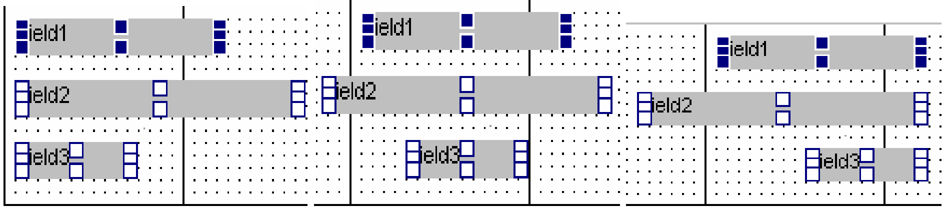
Control Alignment and Sizing

Control alignment and sizing toolbar buttons make it easy to organize the layout of your report and make sure that controls are sized and aligned precisely.

- Select multiple controls in your report, making sure that the control you want to use as a template for aligning and sizing against is selected last.



- Click the Align Lefts, Centers or Rights to align the controls as shown below:



- Follow similar steps to vertically align or size the selected controls.

Formatting Controls

Setting Font Properties

To format the text of a label or a field control, click on the Ellipse button (…) of the Font property in Visual Basic’s property window. You can set the typeface name, size and other font settings from the standard Font dialog.

In addition, you can use the Format toolbar button and combo-boxes to set those properties for any selected control or controls.

Setting Foreground and Background Colors

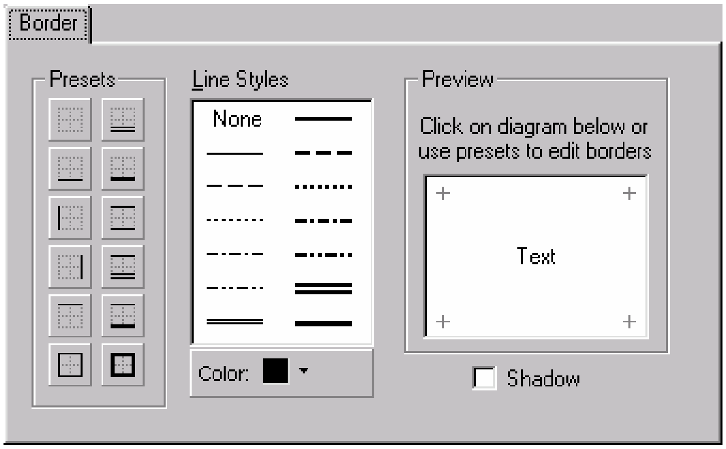
Foreground and background colors can be set using the color drop-down in either Visual Basic’s property window or the color palette toolboxes in the Format toolbar.

The Format toolbar background color palette sets both the BackStyle property of the control to Normal and the BackColor property to the specified color. Background colors will not show if the BackStyle of the control is set to Transparent.

The color palette toolboxes can be dragged and floated or docked to any of the window edges for quick access.

Selecting a Border Style

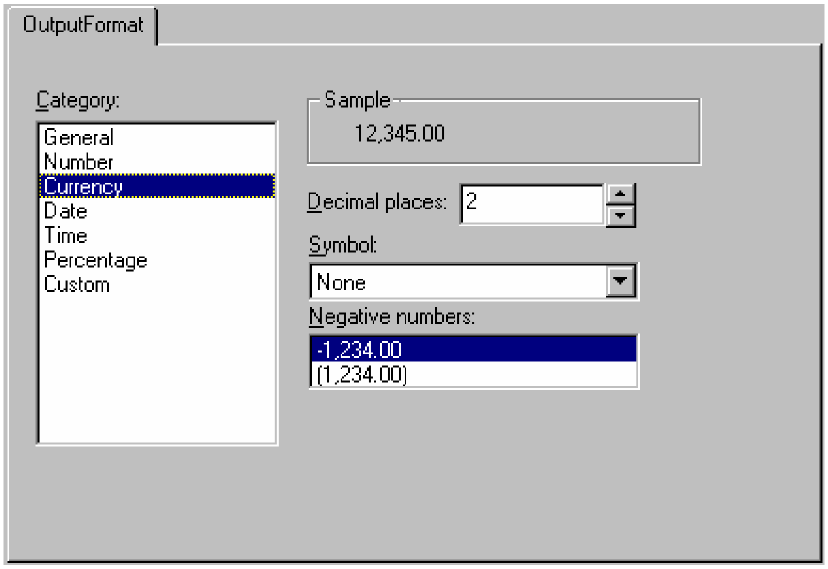
1. ActiveReports allows you to set the border of most controls to a variety of line styles and options. You can set these border styles using the Borders property sheet available from the control's context menu. You can set the borders of a single or multiple controls at the same time.
2. Select a control such as a label or field on your report.
3. Click the right mouse button for the context menu.
4. Select **Format Border** from the menu.



5. You can set the border on each side of the control by selecting the line style and color then clicking the side (between the + signs) you wish to set.
6. Presets contains a list of common border settings, you can click any of the buttons to set the border to the shown style.

Setting Output Formats

ActiveReports provides an easy to use Number Format dialog box. You can use this dialog to set the OutputFormat property of field controls to a valid formatting mask.



Sample: 12,345.00

Decimal places: 2

Symbol: None

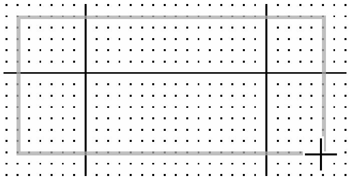
To access this dialog, click on the ellipse (…) button of the OutputFormat property in Visual Basic's property window.

Adding OLE Objects and ActiveX Controls

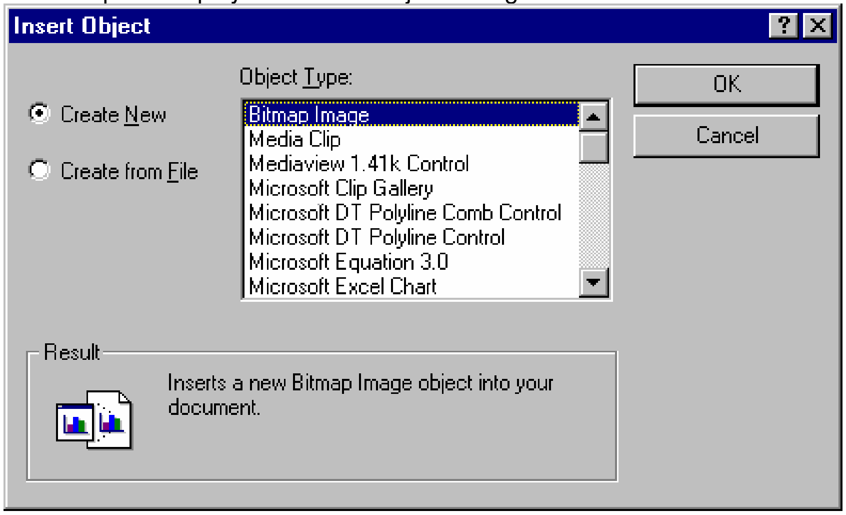
ActiveReports allows full access to insertable OLE objects and ActiveX controls.

To insert an OLE control into your report:

1. Click the control icon OLE from the toolbox.
2. Select the area of the control



3. ActiveReports displays the Insert Object dialog



Insert Object

● Create New

○ Create from File

Object Type:

- Bitmap Image
- Media Clip
- Mediaview 1.41k Control
- Microsoft Clip Gallery
- Microsoft DT Polyline Comb Control
- Microsoft DT Polyline Control
- Microsoft Equation 3.0
- Microsoft Excel Chart

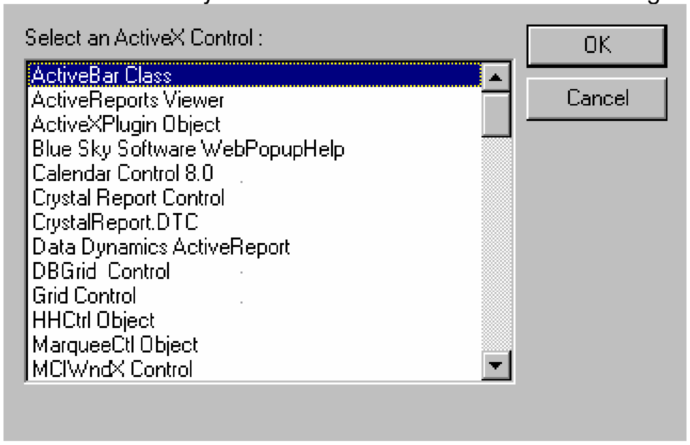
Result

Inserts a new Bitmap Image object into your document.

4. You can select the type of the OLE object to be inserted and Click OK to close the dialog.
5. Or, click Cancel to close the dialog and allow ActiveReports to select the object class from the bound data field.

To insert an ActiveX control into your report:

1. Select the ActiveX control icon ⚒ from the toolbox.
2. Select the control you want to insert from the ActiveX dialog



Select an ActiveX Control :

- ActiveBar Class
- ActiveReports Viewer
- ActiveXPlugin Object
- Blue Sky Software WebPopupHelp
- Calendar Control 8.0
- Crystal Report Control
- CrystalReport.DTC
- Data Dynamics ActiveReport
- DBGrid Control
- Grid Control
- HHCtrl Object
- MarqueeCtl Object
- MCIWndX Control

3. Size the area of the control. The ActiveX will always be printed within that area. ActiveReports will not grow or change the size of the ActiveX control based on its content. The ActiveX control will be rendered just as it would appear on a Visual Basic form at run-time.

Report Explorer

ActiveReports provides an easy navigation through your report sections and controls using the report explorer. To access the report explorer, click on the explorer icon in the main toolbar.

You can navigate by clicking on the tree nodes representing the sections and controls on your report; ActiveReports will select each section or control as you click in the tree.

Fields List

The second pane in the report explorer view displays a list of data source fields based on the data control in your report. The list is populated when you click on the data control in your report.

You can drag and drop fields from the field list to any section on your report. ActiveReports will place a textbox control for the field you dropped and bind it to the data source on your report.

DESIGNING REPORTS

This chapter presents a description of common and advanced reports. It outlines the steps to create these types of reports.

Connecting to your Data Source

Common Reports

Dynamic Reports at run-time

Connecting to your Data Source

ActiveReports uses data controls that are similar to Visual Basic's data controls. Data Controls are used to retrieve the report data from your data source and then load the field values into their corresponding bound fields. You can set up each data controls' properties to create the database connection. Then you bind the controls and groups on your report to the data control.

Note: *Data Controls work only when placed in the detail section of your report.*

Connecting Using Data Access Objects (DAO) Data Control

Connecting Using Remote Data Objects (RDO) Data Control

Connecting Using OLEDB Data Control

Setting the RecordSet and Connection Properties

Binding Controls to a Data Control

DAO (Data Access Objects)

DAO Data Control allows you to connect to your data source through the Visual Basic DAO libraries. DAO uses the Microsoft JET engine to process your SQL and manage your result sets.

You can use the DAO control to access Microsoft Access databases, ISAM databases and ODBC data sources.

Microsoft Access

1. Set the **Connect** property to Access – from the dropdown list.
2. Set the **DatabaseName** property to the full pathname of your mdb file drive:\path\filename.mdb.
3. Set the **RecordSource** property to the name of a table, querydef or a valid SQL statement.

4. If your mdb file is secured using a system.mdw file, set the SystemDB, UserID and Password properties to connect to your database.
5. If the mdb is locked using a password, you need to set the Connect property to ";DATABASE=<database path and filename>;Pwd=<password>"

ISAM files (dBase, FoxPro, Paradox)

1. Set the **Connect** property to the ISAM access type from the dropdown list.
2. Set the **DatabaseName** property to the data file path drive:\path.
3. Set the **RecordSource** property to a valid SQL statement.

Excel Files

1. Set the **Connect** property to the ISAM access type from the dropdown list.
2. Set the **DatabaseName** property to the full pathname of your xls file drive:\path\filename.xls.
3. Set the **RecordSource** property to the name of the table.

ODBC

1. Set the **Connect** property to "ODBC;DSN=<dsn>;[DATABASE=dbname;][UID=UserID;][PWD=pwd;]
2. Leave the **DatabaseName** property blank
3. Set the **RecordSource** property to a valid SQL statement.

Since a report would print all records in the record source, it is recommended that you use Snapshot RecordsetType. Dynaset type contains bookmarks to the actual records in your resultset and requires going back to the database to retrieve the records as they are accessed. Snapshots on the other hand, retrieve the complete record and do not require accessing the database for each record.

RDO (Remote Data Objects)

The RDO Data control allows you to connect to your data source using Remote Data Objects (RDO) libraries. RDO is smaller than DAO and better suited for ODBC connections than DAO. RDO is only available in the Enterprise Edition of Visual Basic.

DSN Connection

You can connect using a DSN by using the Connect property or DataSourceName, UserName and Password properties.

To connect using the DataSourceName property, you can select the DSN from the dropdown list in the property window then type your UserName and Password in the corresponding properties.

To use the Connect property, you should type a valid ODBC connection string in the Connection property. A valid connection string with a DSN is

```
DSN=<dsn>;[UID=<userid>;][PWD=<password>]
```

Note: *The Microsoft Access ODBC driver allows you to specify a security database using the SystemDB parameter and a database name using the DBQ parameter. For example, DSN=MyAccessMDB; DBQ=C:\Data\mymdb.mdb; SystemDB=c:\data\mymdb.mdw; uid=admin; pwd=secure;*

DSN-Less Connection

You can connect to your data source without the use of a Data Source Name (DSN) by using the Connect Property. You will need to specify the driver name in your connect string.

For example:

SQL server

```
DRIVER={SQL Server};SERVER=MySQLServer;DATABASE=pubs;UID=sa;PWD=admin;
```

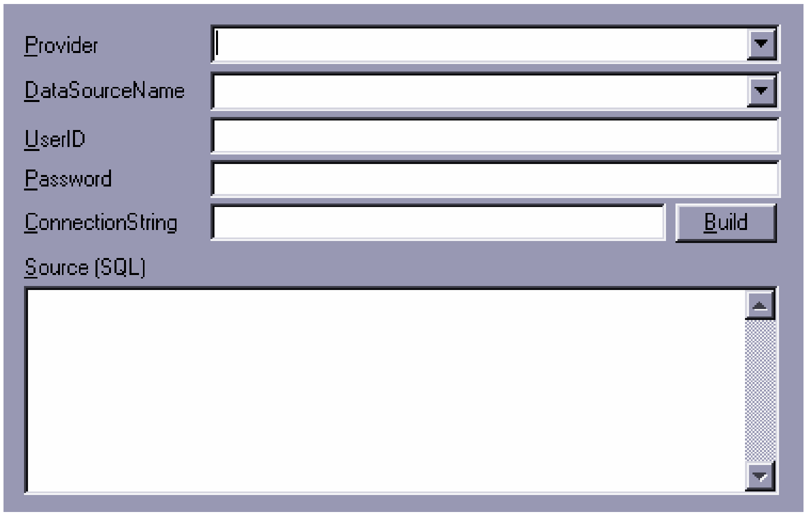
MS Access

```
DRIVER={Microsoft Access Driver (*.mdb)};DBQ=c:\data\mydb.mdb;
```

OLEDB (ADO)

OLEDB (ADO) Data Control allows you to connect to your data sources using OLEDB libraries. OLEDB requires having an OLEDB Provider for your data source or Microsoft’s OLEDB provider for ODBC.

To connect using OLEDB you must have an OLEDB provider installed on the machine. Then, set the connectionstring properties to connect. Use the "Build" button of the Custom properties dialog.



The Provider, DataSourceName, UserID and Password are used only for the MSDASQL provider and an ODBC DSN. If you are using any other provider you should leave all these properties blank and use the ConnectionString only. The Build button displays the connection string wizard to help you set the connection string properly. The following is a listing of common connection strings used with different providers.

Microsoft Access using MSDASQL Provider

```
DRIVER={Microsoft Access Driver (*.mdb)};DBQ=<DBName>
```

Microsoft SQL Server using MSDASQL Provider

```
DRIVER={SQL
Server};SERVER=<Server>;DATABASE=<DBName>;UID=<User>;PWD=<password>;
```

Microsoft Access using Microsoft JET 3.51 OLEDB Provider

```
Provider=Microsoft.Jet.OLEDB.3.51;Persist Security Info=False;Data
Source=<DBName>
```

Microsoft SQL Server using Microsoft OLEDB Provider for SQL Server

```
Provider=SQLOLEDB.1;Persist Security Info=False;User ID=<user id>;Initial
Catalog=<DBName>;Data Source=<servername>
```

Oracle using Microsoft OLEDB Provider for Oracle

```
Provider=MSDAORA.1;User ID=<user id>;Data Source=<servername>;Persist
Security Info=False
```

Setting the RecordSet and Connection Properties

In addition to setting the connection and record source properties for data controls at design-time or run-time, you can set the RecordSet property to a recordset object at run-time. This allows ActiveReports to reuse the recordset that you might have open in your VB application.

In addition, RDO and OLEDB data controls allow you set the Connection property at run-time. ActiveReports will open the recordset using the specified connection object. This reduces the connection requirements for your report.

```
’ Setting the recordset of a data control at run-time
Dim db As Database   ’ DAO database
Dim rs As Recordset  ’ DAO recordset
   Set db = dbEngine.Workspaces(0).OpenDatabase( & _
      App.Path & "\NWind.mdb”)
   Set rs = db.OpenRecordset("SELECT * FROM invoices”)
   Load rptInvoice   ' Load the report
   ’ set the recordset
   Set rptInvoice.dcRptData.Recordset = rs
   rptInvoice.Show      ’ Preview the report
```

Binding Controls to a Data Control

You can bind text field controls, image controls and OLE object controls to fields in your data source control. ActiveReports reads each record in the data source and loads the bound field values into the bound control.

Controls are bound using the DataSource and DataField properties. You can select a data source from the dropdown list in the property window then select the binding field from the DataField dropdown list.

In addition to the above, you can use the Fields List Window to drag and drop fields on any of your report’s sections. ActiveReports sets the name, DataSource and DataField properties accordingly.

Common Reports

Each one of the following reports assumes that you have created a new Visual Basic project and added a new ActiveReport to the project. You can execute the report using the Show or PrintReport methods or link the report to an ActiveReports Viewer control on one of your forms. (See the Tutor directory for an implementation example)

Simple Table or List Report

Grouping Records

Group on Calculated Expression

Adding a Summary Field in Group Footer

Conditional Summary Fields

Adding Formulas and Calculated Fields

Top n Reports

Master Detail Reports

Summary Reports

Mail-Merge Letters

Columnar Reports

Labels

SubReports

Unbound Reports

More Report Tips

Simple Table or List Report

The simplest form of a report is a tabular listing of fields from your record source. This exercise uses the NorthWind database included with Microsoft Visual Basic.

1. Start Visual Basic and create a new project.
2. Place 2 command buttons on the new form in your project.
3. Add an ActiveReport designer, Project > Add Active Designer > ActiveReport
4. Add the following code to your Print and Preview buttons click event

```
ActiveReport1.Show                     ' Preview click
ActiveReport1.PrintReport False        ' Print Click
```

5. Select **DAO Data Control** from the toolbox and place it in the Detail section.
6. Set the controls properties as follows

Name	*dcRptData*
DatabaseName	*<VB Path>\NWind.mdb*
RecordSource	*SELECT * FROM Customers*

7. Place 4 Label controls horizontally in the PageHeader section.
8. Set the Labels properties as follows

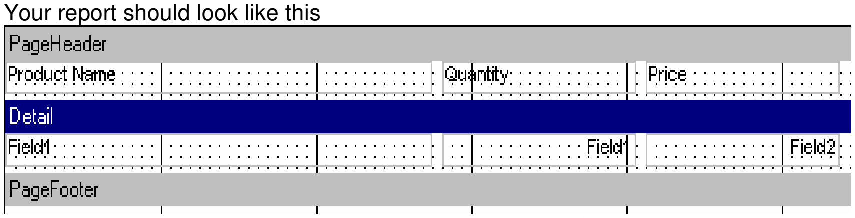
Name	lblCustomer	lblCity	lblCountry	lblPostalCode
Caption	Customer	City	Country	Postal Code
Left	0	2970	5490	7380
Height	270	270	270	270
Top	0	0	0	0
Width	2880	2430	1800	1800

Set the page header section height property to 285

9. Click to open the Report Explorer Window.
10. Drag and Drop the following fields into the Detail Section: CustomerName, City, Country, PostalCode
11. Set the Fields properties as follows

Name	txtCustomer	txtCity	txtCountry	txtPostalCode
DataSource	-------	dcRptData	-------	
DataField	CustomerName	City	Country	PostalCode
Left	0	2970	5490	7380
Height	270	270	270	270
Top	0	0	0	0
Width	3960	2430	1800	1800
Alignment	0-Left	0-Left	0-Left	1-Right

12. Select the detail section by clicking on the section bar (gray) area.
13. Set the **Height** property of the detail section to 285.
14. Your report should look like this



15. Run your project and click on the Preview button on your form.

Grouping Records

Using the first report we created in the previous exercise, we will group the product sales list by Customer and Product Name. ActiveReports allows up to **32 nested groups** in a single report. Groups can be bound to a field in the data source or set at run-time using unbound GroupValue property. In this exercise we will create a group section and bind it to the Country field in the data source.

Note: *ActiveReports does not order the records for grouping, it assumes that the data source recordset is already sorted in the same grouping order. For example, in this exercise, the data source needs to be sorted by Country to get the desired results.*

Open the report we created in exercise 1.

1. Select the data control dcRptData and modify the **RecordSource** property to: “SELECT * FROM Customers ORDER BY Country”

2. Select the Detail Section in the report.
3. Right-click the mouse and select **Insert > Group Header / Footer**.
4. Click on the new section “GroupHeader1” to select it.
5. Modify the section properties as follows:

Name	ghCountry
Height	360
DataSource	dcRptData
DataField	Country

6. Insert a Field control in the ghCountry section

Name	txtCountry
Top	0
Left	0
Height	360
DataSource	dcRptData
DataField	Country
Width	4230
Font.Size	12
Font.Bold	True

7. Run the project and press “Preview” to see your report grouped by Country.

Group on a Calculated Expression

Many types of reports require that your groups are based on a calculated expression such as first letter in CompanyName field or TotalProductSales ranges. ActiveReports allows you to create your groups in unbound mode by setting the GroupValue property of the group header section in your events code. **In order to get correct grouping the data source should be sorted by the grouped fields.**

In this example, we have a data source of company names and phone numbers. We will group the records based on the first letter in the company name for a directory-type report.

Set up the detail section in your report with CompanyName and PhoneNumber fields and bind them to the data source. The data source is sorted by CompanyName.

Create a group header/footer pair and name the sections ghCompany and gfCompany. Set the ghCompany.DataField property to "FirstLetter". FirstLetter is a user defined field that does not exist in the data source, instead it is added and fetched at run-time in the two new events DataInitialize and FetchData.

Add the following code

```
Private Sub ActiveReport_DataInitialize()
   Fields.Add "FirstLetter"
End Sub

Private Sub ActiveReport_FetchData(eof As Boolean)
   If dc.Recordset.EOF Then Exit Sub
   Fields("FirstLetter").Value = _
      Left$(dc.Recordset!CompanyName, 1)
End Sub
```

ActiveReports will call the FetchData event and group the records based on the value of the "FirstLetter" user defined field.

Adding a Summary Field in Group Footer

You can add summary fields that calculate the total, count, average and other aggregations in any of the report sections. The placement of the summary fields dictate when a section containing that field and sections after it will be printed. A section with a summary field will be delayed until all the calculations are completed. This allows you to place a summary field ahead of its detail.

Summary fields are calculated according to the Summary properties of the field control. A summary field control is updated with each new detail record read from the record set. When a field is placed ahead of the detail section (ReportHeader, PageHeader or GroupHeader) the detail section is formatted with each record and the summary field is updated. When all records for the summary level are read, the header section is printed followed by the delayed sections.

Summarization fields are controlled by the following field control properties:

SummaryType: Specifies the summarization level that the field should calculate. Summary types are: GrandTotal – Calculates the summary expression once for all detail records in the report; PageTotal – calculates the summary expression once for each page; SubTotal – calculates the summary expression once per SummaryGroup section.

SummaryFunc: Sets the type of aggregation that the summary field will calculate. Summary function can be set to Sum, Average, Count, Min, Max, Variance and Standard Deviation. In addition you can set the summary function to calculate based on the distinct values of another field.

Distinct summarization is valuable where the field's value repeats in many detail records and the summary function needs to include a single value of all repeating values. For example, transaction records of orders might look like this

Order ID	Date	Amount
1001	5/16/97	3000
1001	5/30/97	-500
1001	6/15/97	-2500
1002	4/20/97	2550
1002	4/30/97	-2500

A Count summary function would calculate **5** as the number of records. However, a distinct count on the **SummaryDistinct** Order ID field would properly return **2** as the number of orders. The same would apply for other distinct functions.

SummaryGroup: A subtotal type summary is calculated. It defines the group section level for which the subtotal should be calculated. It is usually the matching group header section associated with the section where the summary field is placed.

SummaryRunning: Allows you to create running totals that accumulate with each printing a reset at the specified level. A SummaryRunning in-group keeps a running total of the specified field within a single group span. In all setting keeps a running summary throughout the whole report.

In this exercise, we will count the number of customers in each country group in the previous report. We will add a summary field in the group footer gfCountry.

1. Start by opening the previous project.
2. Set the gfCountry section height to 285
3. Add a label control lblGFooter at Left=0, Height=270, Width=2070, Top=0 and set the caption to "Number of Customers"
4. Add Field control txtCustomersCount at Left=2160, Height=270, Width=1980, Top=0
5. Set the txtCustomerCount properties as follows

DataSource	*dcRptData*
DataField	*CompanyName*
SummaryFunc	*2-Count*
SummaryGroup	*ghCountry*
SummaryType	*3-SubTotal*
Alignment	*1-Right*

6. Run the project and select Preview to see the new report with customer count in each country group.

Conditional Summary Fields

Some report require that totals, or counts or other summary formulas to be based on a certain condition. For example, if you have a data set of product sales of different categories in different countries, you might want to print the total category sales by country in the category group footer without grouping the record on the Country field.

Category	Product	Country	Sales
Beverages	Product A	USA	1000
Beverages	Product A	UK	560
Beverages	Product B	USA	2000
Beverages	Product B	UK	1067
…	…	…	…

Your report looks like

Beverages Category		
Product A	USA	1000
Product A	UK	560
Product B	USA	2000
Product B	UK	1067
	Total USA	3000
	Total UK	1672

Create the detail and group sections in your report and add the detail and group header fields.

Add two text fields for the totals txtUSACatTotal and txtUKCatTotal, leave them unbound.

In the Detail_Format event update the DataValue property of the fields by adding the Sales amount based on your condition

```
Private Detail_Format()
If txtCountry.DataValue = "USA" Then
   txtUSACatTotal.DataValue = txtUSACatTotal.DataValue + _
                              txtSales.DataValue
Else If txtCountry.DataValue = "UK" Then
   txtUKCatTotal.DataValue = txtUKCatTotal.DataValue + _
                              txtSales.DataValue
End If
End Sub

Private Sub ActiveReport_ReportStart()
   ’ Reset the totals.
   txtUSACatTotal.DataValue = 0
   txtUKCatTotal.DataValue = 0
End Sub

Private Sub gfCategory_AfterPrint()
   ’ Reset the totals
   txtUSACatTotal.DataValue = 0
   txtUKCatTotal.DataValue = 0
End Sub
```

Adding Formulas and Calculated Fields

ActiveReports uses Visual Basic as its formula language, all calculated fields and expressions are handled during the section events.

In this example, we will create a new report that prints customers sales from the order details records. The report will be grouped and summarized by customer, total sales will be calculated as the total of the quantity in each detail record multiplied by the product price.

1. Create a new ActiveReport as shown in the first tutorial.

2. Insert a DAO data control in the detail section

3. Set the properties as follows
   Name
      dcRptData
   DatabaseName
      <VB Path>\Nwind.mdb
   RecordSource

```
SELECT
        Customers.CompanyName,
        Products.ProductName,
        [Order Details].UnitPrice,
        [Order Details].Quantity
FROM Products
        INNER JOIN ((Customers
        INNER JOIN Orders
                ON Customers.CustomerID = Orders.CustomerID)
        INNER JOIN [Order Details]
                ON Orders.OrderID = [Order Details].OrderID)
        ON Products.ProductID = [Order Details].ProductID
WHERE (((DatePart("yyyy",[OrderDate]))=1995))
ORDER BY Customers.CompanyName, Products.ProductName
```

4. Insert a group header/footer section.

5. Set the GroupHeader properties as follows

Name	ghCustomer
DataSource	dcRptData
DataField	CompanyName
Height	765

6. Add a field control to the ghCustomer section

Name	txtCompanyName
DataSource	dcRptData
DataField	CompanyName
Top	0
Left	0
Height	360
Width	5670

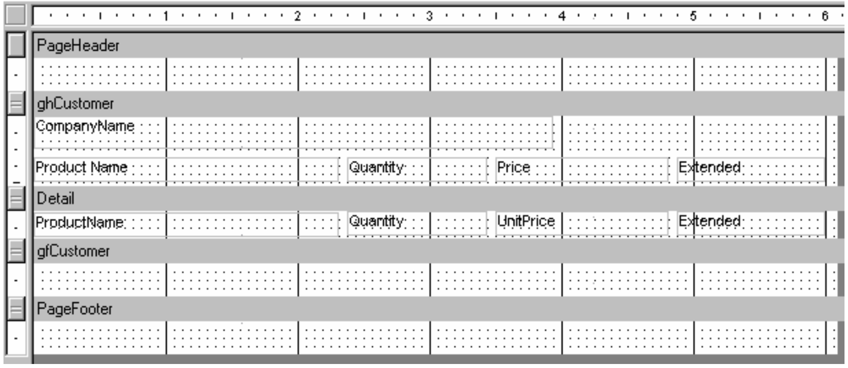
7. Add the following label controls to the ghCustomer section

Name	lblProductName	lblQuantity	lblUnitPrice	lblExtended
Caption	Product Name	Quantity	Unit Price	Extended
Top	450	450	450	450
Left	0	3420	5040	7020
Height	270	270	270	270
Width	3330	1530	1890	1620

8. Set the detail section height to 300 and add the following text controls

Name	txtProductName	txtQuantity	txtUnitPrice	txtExtended
DataSource	dcRptData	dcRptData	dcRptData	
DataField	ProductName	Quantity	UnitPrice	**Extended**
OutputFormat		Number	Currency	Currency
Top	0	0	0	0
Left	0	3420	5040	7020
Height	270	270	270	270
Width	3330	1530	1890	1620

9. Your report should look like this.



10. The txtExtended control DataSource property is left blank and the DataField is bound to a run-time field called "Extended". The field is added in the DataInitialize event and the value is set in the FetchData event

11. Add the following code to the report

```
Private Sub ActiveReport_DataInitialize()
   Fields.Add "Extended"
End Sub
Private Sub ActiveReport_FetchData(eof As Boolean)
   If dcRptData.Recordset.EOF Then Exit Sub
   Fields("Extended").Value = dcRptData.Recordset!Quantity * _
         dcRptData.Recordset!UnitPrice
End Sub
```

12. Run your project and click preview to see the new report with the extended price calculation.

Top n Reports

ActiveReports has no special handling for Top n Records report styles. You can easily implement such reports by setting your data source to a Top n filtered query. If your data source does not support Top n queries, you can set your query to return your records ordered by the Top n value descending. Then set the MaxRows property to n.

For example, to list the Top 10 customers by their sales number. You can create a query that returns all customer sales ordered by the sales value descending and set the MaxRows property of the data control to 10. ActiveReports will process only 10 records from the sorted query results.

Master Detail Reports

Master detail reports are implemented in ActiveReports using a JOIN SQL query and adding a group on the master key. The master records can be placed in the group header and footer sections and the detail records placed in the detail section.

For example, to create a master-detail of orders and order detail records, you can create a query: `“SELECT * FROM order, [order details] WHERE orders.OrderID = [order details].OrderID”` then create a group where the group field is the OrderID field. Place all order fields such as ID, Date in the group header sections and place the order detail fields in the detail section.

Summary Reports

Summary Reports are implemented by setting Visible property to False or setting its height to 0. The detail section will be processed and the summary group header and footer sections will be printed without the detail section.

For example, to create a summary of sales of all orders, you can create the same report mentioned above and set the detail height to 0. ActiveReports will print the summary of each order without printing its detail level.

Mail-Merge Letters

ActiveReports supports mail-merge reports using the RichText control. The RichText control can contain field placeholders that are replaceable with their values (merged) at run-time.

To create a mail-merged report, insert a RichText control in the detail section of your report; edit the text of your letter as you would a word-processing document. Use the insert field command to add a mail-merge field into the text.

In the detail format event, add the following code to update the field values in the RichText control for each record in your detail section.

```
Private Sub Detail_Format()
   With dcRptData.Recordset
      rtf.ReplaceField("LastName”,.Fields(“LName”).Value)
      rtf.ReplaceField("FirstName”,.Fields(“FName”).Value)
   End With
End Sub
```

Columnar Reports

ActiveReports supports newspaper column layout in both the detail and group sections. You can render the columns either horizontally or vertically in the section with options to break the column on group sections (start a new column on the change of a group).

Labels

You can use ActiveReports to print any label size by using the newspaper column layout.

To create a report that prints labels to a laser printer labels sheet:

1. Set the report width to the total width of the label sheet (if printing to a laser printer).
2. Set the columns property of your detail section to the number of labels across the page.
3. Remove the page header and footer sections from your report.
4. Set the height of the detail section to the exact height of the label.
5. Finally, place your text and label controls in the column area in the detail section.

Skipping Labels

ActiveReports allows you to skip labels on a label sheet using the LayoutAction property. LayoutAction has several uses, one of which is to skip a section i.e. do not print anything and move to the next printable area. So for each label you want to skip you can set the LayoutAction property to 2 - ddLAMoveLayout in the format event of the detail section.

The section will be skipped without moving to the next record in the data source.

```
Dim iSkipLabels As Integer

Private Sub Detail_Format()
   If iSkipLabels > 0 Then
      iSkipLabels = iSkipLabels - 1
      LayoutAction = ddLAMoveLayout
   End If
End Sub
```

SubReports

ActiveReports supports creating reports that contain any number of child reports using the SubReport control. Child reports or SubReports are executed each time their parent section (the section on which the subreport control is placed) is printed.

You can use subreports to print any of the following types of reports.

- Related child reports that contain data from a source different from the main report.
- Related or un-related lists to be printed in parallel horizontally or vertically. For example, use subreports to print a list of top 10 products, top 10 customers and top 10 employees. Each of these lists can be printed using a separate child report.

Note: *SubReports can be nested within each other, where a child report can contain a subreport control which links to grand-child and so on…*

Note: *SubReports print only their group and detail sections, page headers and footer are not printed in the parent report.*

Adding a SubReport

To insert a subreport into your report:

1. Click on the subreport control in the toolbox
2. Place the control in the section where the subreport will be printed. The subreport will be executed each time the specified section is printed.

Note: *The subreport will be confined by the width of the control, but the height will grow according the needs of the subreport.*

Linking the SubReport to your main report

You will link each subreport in your report to the actual report object at runtime. ActiveReport cannot instantiate the child report directly, you will need to set the object property of the subreport control to an instance of your child report in the format event of the parent section.

For example, let's assume you have two subreports in the detail section of your report, in your Detail_Format event add the following code:

```
Private Sub ActiveReport_ReportStart()
   Set srptProducts.Object = New rptTopProducts
   Set srptCustomers.Object = New rptTopCustomers
End Sub

Private Sub Detail_Format()
   ’ Optionally set record source filter
   srptProducts.Object.dcRptData.RecordSource = sTop10Products
   srptCustomers.Object.dcRptData.RecordSource = sTop10Customers
End Sub

Private Sub ActiveReport_ReportEnd()
   On Error Resume Next
   Unload srptProducts.Object
   Set srptProducts.Object = Nothing
   Unload srptCustomers.Object
   Set srptCustomers.Object = Nothing
End Sub
```

Linking SubReports to a Parent by Changing the Record Source

If you need to print each subreport with a different recordset based on key in the parent report, you can change the recordsouce of the subreport after creating the instance.

In this example, each subreport will print the products within each category.

```
Private Sub ActiveReport_ReportStart()
   Set srptProducts.Object = New rptProducts
End Sub

Private Sub Detail_Format
   ’ Modify the recordsource based on the current category
   Srpt.Object.dcRptData.RecordSource = _
      "SELECT * FROM Products WHERE CategoryID = ” & _
[illegible]
End Sub

Private Sub ActiveReport_ReportEnd()
   On Error Resume Next
   Unload srptProducts.Object
   Set srptProducts.Object = Nothing
End Sub
```

Unbound Reports

ActiveReports gives you complete control to bind your report to any data source - including arrays-through its programmable object model. You can create a report without any data binding then load the data from your data source and feed it into the report controls at runtime.

ActiveReports Fields property provides the data binding between the control and the runtime fields. It allows you to set the DataField property of the control to any of the runtime defined field names.

The DataInitialize and FetchData events are used to define the runtime fields and feeding the data values of these fields so they can be used with bound controls. These two events can also be used to create calculated fields in bound report (reports that have a data control).

The following example creates a set of runtime fields and loads the field values from an array.

```
Private Sub ActiveReport_DataInitialize()
    Fields.Add "OrderID"
    Fields.Add "ProductID"
    Fields.Add "ProductName"
    Fields.Add "Qty"
    Fields.Add "Price"
    Fields.Add "Amount"
    ’ iRow is the current record pointer
    iRow = LBound(arr)
End Sub

Private Sub ActiveReport_FetchData(eof As Boolean)
    ’ If we processed all element then we exit the event
    ’ leaving the eof paramter at its default True value
    ’ this will promp AR to end the report
    If iRow > UBound(arr) Then Exit Sub
    Fields("OrderID") = arr(iRow).OrderNo
    Fields("ProductID") = arr(iRow).ProductID
    Fields("ProductName") = arr(iRow).ProductName
    Fields("Qty") = arr(iRow).Qty
    Fields("Price") = arr(iRow).Price
    Fields("Amount") = arr(iRow).Amount
    iRow = iRow + 1
    ’ We must set the eof parameter to True as
    ’ long as there is more data to be processed
    eof = False
End Sub
```

More Report Tips

Page n of m in Page Footer

Conditional Printing

Green-bar Printout Report

“Continued…” In Page Footers

Phonebook Style: Begin-End in Page Headers

Multi-Page Sections

Charts

Multiple Records per Section

Modifying the Pages Collection of a Processed Report

Page n of m in Page Footer

You can add page numbers and page count to your report using the PageCount summary type. Page numbers are created using a running summary field.

Add two text fields and a label to your Page Footer section

Set the following properties

First field

Name	txtPageNumber
SummaryRunning	2-in all
SummaryType	4-PageCount

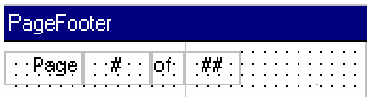
Label

Name	lblOf
Caption	of

Second Field

Name	txtPageCount
SummaryType	4-PageCount

Your page footer should look like this


PageFooter

Page # of ##

When you preview your report, you should get a continuous n of m page counts. Please note that using the PageCount causes all report sections to be delayed until the report is completed.

Conditional Printing

ActiveReports allows you print or suppress any control at runtime based on a condition in your data. This can be achieved by modifying the Visible property of the control in the Format event of the parent section.

For example, you can print an “Outstanding” label for employees that exceeded their sales goals for the period by adding the following code in the Detail Format event:

```
Private Sub Detail_Format()
   If txtEmployeeSales.DataValue > txtEmployeeGoal.DataValue Then
      lblOutstanding.Visible = True
   Else
      lblOutstanding.Visible = False
   End If
End Sub
```

Similarly, you can modify any of the print properties of controls in any of the format events in your report.

Green-bar Printout Report

You can create green-bar printouts by alternating the shading or background color of your detail section in the format event.

```
Dim I As Integer
Private Sub Detail_Format()
   If (I Mod 2) = 0 Then
      ’ Set the detail BackStyle to normal
      Detail.BackStyle = ddBKNormal
      Detail.BackColor = vbGreen
   Else
      Detail.BackStyle = ddBKTransparent
   End If
   I = I + 1
End Sub
```

“Continued…” in Page Footers

A common requirement of reports is to print a “Continued…” message at the bottom of each page in the report except the last page. You can achieve this in ActiveReports by checking the EOF flag of the data source in the PageFooter section Format event.

```
Dim bLastPage As Boolean

Private Sub ActiveReport_FetchData(eof As Boolean)
   bLastPage = dcRptData.Recordset.EOF
End Sub

Private Sub PageFooter_Format()
   If bLastPage Then
      ’ lblContinued is a label placed in the
      ’ Page Footer section
      lblContinued.Caption = ""
   Else
      lblContinued.Caption = "Continued…"
   End If
End Sub
```

Phonebook Style: Begin-End in Page Header

Phonebooks and address books can be printed using a format similar to the yellow pages or dictionary layout, where the page header shows the first and last entries on the page.

You can achieve the same behavior by placing two labels in the page header and adding the following code in your report.

```
Dim sLastEntry As String

Private Sub PageHeader_Format()
   ’ Set the first entry label, wait on the second
   lblFirstEntry.Caption = _
      dcRptData.Recordset.Fields("Name”).Value
End Sub

Private Sub Detail_AfterPrint()
   ' Assign the just-printed entry to the temp variable
   sLastEntry = txtName.Text
End Sub

Private Sub PageFooter_Format()
   lblLastEntry.Caption = sLastEntry
End Sub
```

The code above works as described because ActiveReports does not render the page header until all Detail sections are printed and the format event is fired for the page footer section.

Multi-Page Sections

By using the PageBreak control you can span a report header or report footer section across multiple pages. You can use this to add multiple summary pages to your reports that might contain charts, summary tables and table of contents.

Charts

ActiveReports does not include a built-in data chart control. However, its support for ActiveX controls allows you to use any charting control in your report. You can use the data in your report to set series and data points in the chart as the report is being processed.

If you place the chart in the report header (i.e. before its data is processed) you will need to place a summary field control in the same section. This allows ActiveReports to delay printing the section until all the required data is processed and you will get a chance to load your chart data correctly. See the EmployeeSales report for an example of this technique.

Another alternative is to place the chart control in a child report and link it to a subreport control in the main report. This allows you to fully process the data for the chart and then render it into the main report using the subreport control. However, this would require going through more than one recordset, one for the main report and another for the child report.

Multiple records per section

LayoutAction property allows you to control how many records are printed in a single section. In the Detail section format event you can set the LayoutAction to ddLANextRecord, which moves the record pointer but keeps the current section active for printing.

This allows you to print multiple records in the same section. You can use this functionality to print Aging reports, calendar of events in a calendar format, and more such reports. See the Samples directory for an example of using this powerful feature (Students and Classes – Schedule).

Modifying the pages collection of a processed report

ActiveReports caches the printed pages into canvas objects in the Pages collection. Once a report is completed you can access each canvas object in the collection and modify its contents using the canvas drawing methods.

In addition, you can save and load canvas objects into the collection using the Save and Load methods. You can overlay a preset layout to any of the pages (pre-printed forms) using the Overlay method.

ActiveReports allows you to reorder the pages in the collection and selectively print odd or even pages. You can also change the printer device of each page or group of pages.