

Create Great Reports

VB's reporting capabilities once seemed like an afterthought. Now you have a choice between the speed of reporting utilities and the flexibility of doing it yourself.

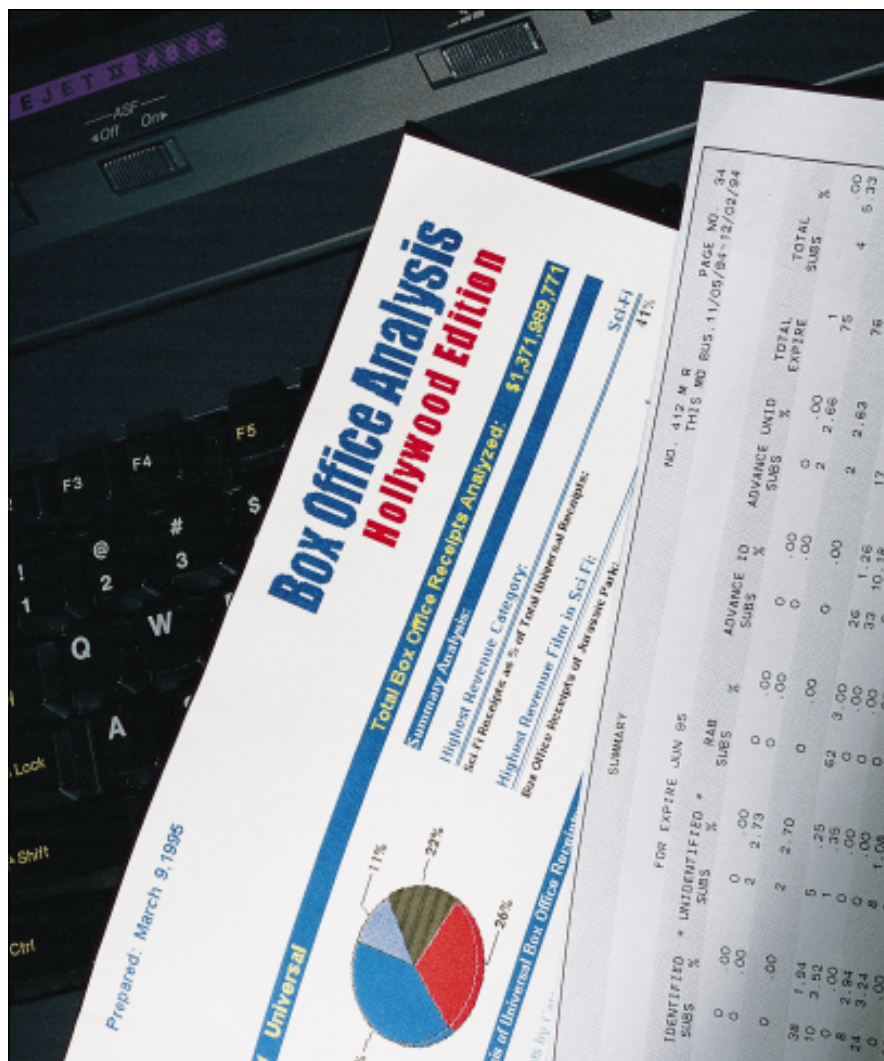
BY PHIL WEBER

No matter how good the data in your applications looks on screen, sooner or later you're going to have to print out something. And if you're like me, after spending days fine-tuning your user interface, you won't be satisfied with anything less than perfect output.

In contrast to the control VB programmers have over the onscreen appearance of their applications, Visual Basic's printing capabilities sometimes seem like they were slapped on at the last minute. I mean, they give us this cool interactive development environment, with point-and-click user interface design and versatile database support, and then for output . . . the *printer object*?

The release of VB 3.0, with its bundled version of Crystal Reports 2.0, improved the situation somewhat. But Crystal Reports 2.0 has limitations of its own, and if your reporting needs don't fit neatly into Crystal Reports 2.0's rows-and-columns format, or if you're concerned about the size and speed of your programs, you may need to explore the tradeoffs be-

Phil Weber is the founder of Micro Business Services, a contract programming firm specializing in Visual Basic. He is also a member of the Visual Basic Programmer's Journal Technical Review Board. Phil will speak on the subject of creating reports at VBITS '95 in New York this June. Reach him on CompuServe at 72451,3401.



tween using another commercial report writer and rolling your own reporting applications in VB.

If you're looking for an alternative to Crystal Reports 2.0 (the version included with VB 3.0), your options are simple: purchase another report writer, or create your own reporting routines in VB itself. A number of third-party reporting tools are out there (see the accompanying sidebar, "Third-Party Reporting Tools"), but this article will focus on using Crystal Reports Pro, Version 4.0, for two reasons: it's roughly comparable to the other tools in capability, and it's available as a relatively inexpensive upgrade to users of VB 3.0.

How does the latest version of Crystal Reports compare to the one you already have? As you may know, most of Crystal Reports' work is done by its print engine, CRPE.DLL; CRYSTAL.VBX is just a simple interface to the routines in the DLL. The Crystal Reports 2.0 print engine included with VB has 35 functions, 25 of which are exposed as VBX properties (you can access the other 10 functions by calling them directly from your code. See the CR20DEMO example in this article for details). In contrast, the new VBX (and OCX) in Crystal Reports Pro 4.0 makes nearly all of its 80-plus print engine functions available as properties. This added functionality (and the accompanying documentation) is by itself an excellent reason to upgrade.

Crystal Reports Pro 4.0 includes dozens of other enhancements, including the ability to view live data while designing your reports and to create bound reports based on the record set of a data control. It can also export your reports in a variety of formats, including Excel 4.0, Word for Windows 6.0, and standard mail formats (VIM and MAPI). Crystal also offers a server-based product (Crystal Reports Server) that allows a workgroup administrator to design reports for end users and to specify which machines should function as distributed report processors (DRPs). Report processing is off-loaded to the DRPs; they are the only nodes that require connections to the remote data, so LAN traffic is minimized and network resources can be used more efficiently.

The downside of these additional features is that Crystal Reports now leaves a significantly larger footprint on your applications. VB programs that use Crystal Reports 2.0 require less than 1.5 MB of support files, not including Access' JET Engine, whereas apps using Crystal Reports Pro 4.0 must carry an additional 2.3 MB of baggage. And while Crystal claims that its "Smart Engine Technology" makes Version 4.0 between two and 10 times faster than previous versions, I found the

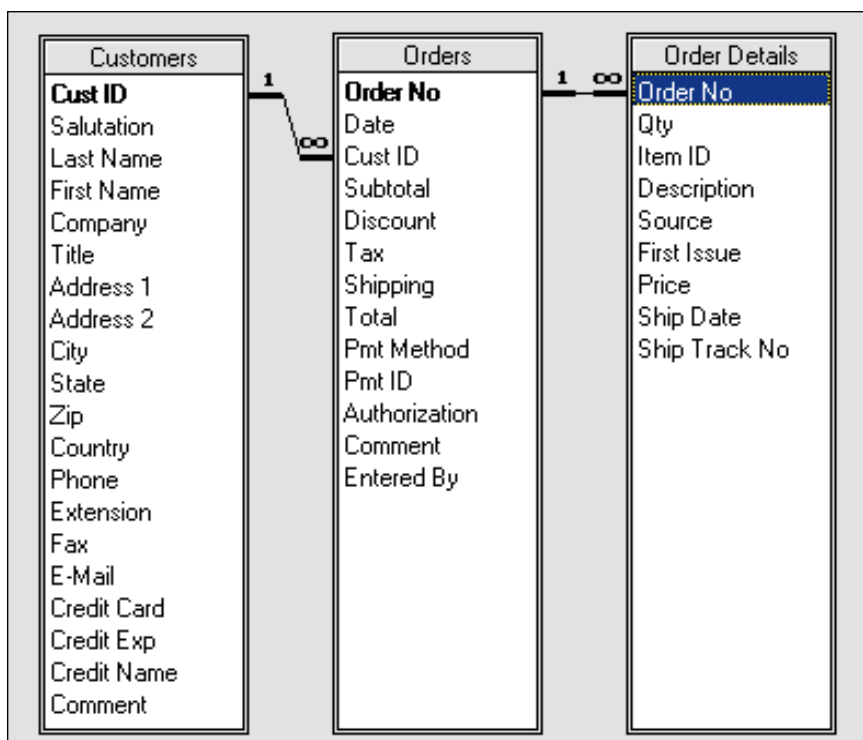


FIGURE 1 The Tables Used in the Customer History Report. The one-to-many relationships of the data present a challenge: How can I display a single customer record alongside multiple order records? Unlike a sort by a finite category (such as State or Country), my desired reports will create a large number of relationships for the report writer to handle.

Customer History Report					
Customer	Order No	Date	Qty	Item ID	Description
Weber, Phil	00001	02/10/95	1	VBPJ-12	VBPJ Subscription, 12 issues
Micro Business Services	00001	02/10/95	1	VB-CD	VB-CD Subscription, 4 issues
10450 SW McDonald St. #4					
Portland, OR 97224-4851					

FIGURE 2 The Goal. This mock-up, created in Microsoft Excel, was the design specification for the Customer History Report. The report writers had no trouble creating the reverse-type column headings, but neither of them could pad the order numbers with leading zeroes.

opposite to be true: the best performance I could wring out of 4.0 was still 47 percent slower than 2.0, using identical reports and data. I suspect this is because Crystal Reports Pro 4.0 uses more memory than earlier versions, and my machine is equipped with only 8 MB of RAM. If you can live with its additional disk space and memory requirements, however, Crystal Reports Pro 4.0 is a significant upgrade, well worth a closer look.

To demonstrate the relative strengths and weaknesses of various reporting techniques, I've created three Visual Basic programs that each use a different method to preview and print the same report. The source code for these programs is in REPORT.ZIP, in the Magazine Library (Library3) of the VBPI Forum on CompuServe (GOVBPIFO), and on the third edition of the VB-CD Quarterly. The first example (CR20DEMO) uses Crystal Reports Stan-

dard 2.0. The second (CR40DEMO) uses Crystal Reports Pro 4.0, and the third (VBDEMO) is straight VB code, with substantial help from VideoSoft's VSVIEW.VBX.

For the uninitiated, VSVIEW includes vsPrinter, a control that replaces VB's printer object and provides greatly enhanced functionality, and vsViewPort, which simplifies the creation of scrollable regions on a form. I could have created my own printing routines using Windows GDI calls, but why reinvent the wheel? VideoSoft has already done the dirty work. If you're the do-it-yourself type and want to know how to print and preview without custom controls, see my article, "Overcome Printing Limitations," in the June/July 1994 issue of VBPI.

The sample report is a customer history based on three database tables: Customers, Orders, and Order Details (see

Figure 1). Specifically, I wanted to know which customers had purchased more than one item, which items they had purchased, and when they had ordered them. Ideally, the report would be laid out like a previously created mock-up (see Figure 2).

This exercise revealed a number of trade-offs and “gotchas” you should consider before you decide how to create reports in your own applications.

SPEED VERSUS FLEXIBILITY

The primary advantage offered by third-party reporting tools is development

speed. Designing a report in a report writer is similar to designing your user interface in VB: you simply drag database fields from a list and place them as desired on your report (see Figure 3). It took me only about an hour to design my sample report, and that includes the learning curve; I had never worked with Crystal Reports before. You (or users of your app) can modify a report’s design without having to change your program. On the other hand, it took me more than four hours to write and debug the VB version of my report, and any changes made to my

handcrafted report writer after the fact will require a recompile.

What you gain in development speed, however, you lose in reporting flexibility. For example, because report writers use a grid model to display records as rows and fields as columns, it’s nearly impossible to create a report with information from one table printed next to data from another table. My report design specified that a customer’s order information should appear alongside his name and

NO MATTER HOW GOOD
YOUR APPS LOOK ON
SCREEN, SOONER OR
LATER YOU’RE GOING TO
HAVE TO PRINT
SOMETHING.

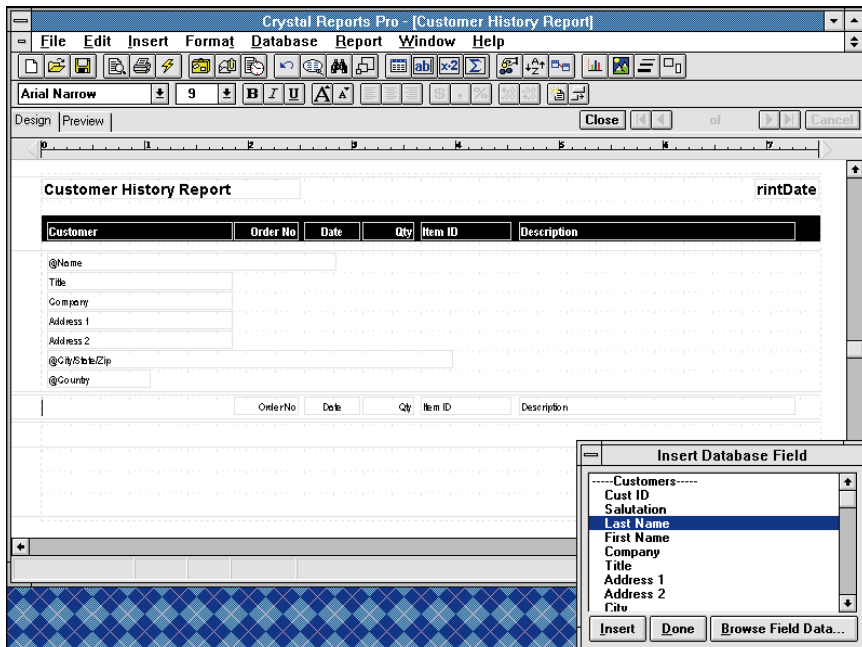


FIGURE 3 *The Crystal Reports Pro 4.0 Design Environment. Database fields are selected from the list and placed on the report as desired. Report functions such as @City/State/Zip allow you to format and combine fields as you wish. A click on the Preview tab lets you design your report while viewing your actual data.*

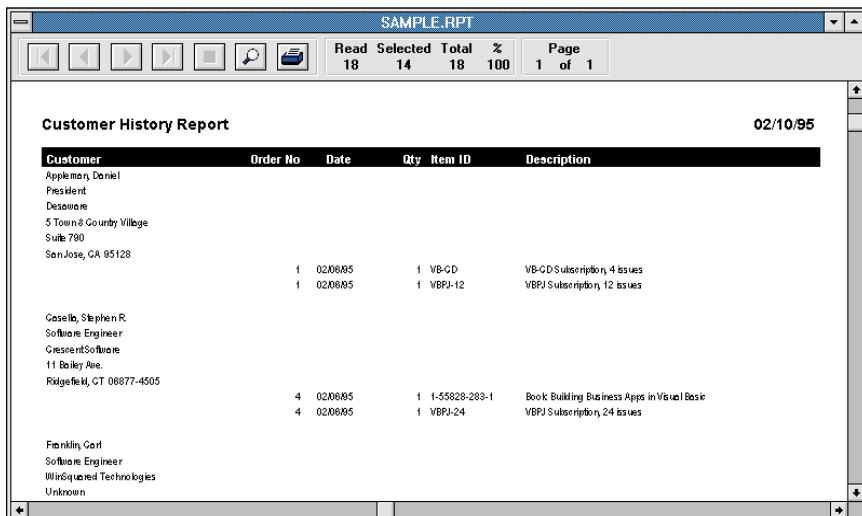


FIGURE 4 *Crystal Reports 2.0's Preview Window. While it's certainly functional, Crystal Reports Version 2.0's Preview Window might look out of place in many of your applications, particularly if your program emulates the look and feel of Microsoft Office applications.*

mailing address, but both flavors of Crystal Reports wanted to reprint the customer data with each associated order item. I was able to group records by Cust ID and suppress duplicates, so that the customer information printed only once, but it had to appear above the order history, rather than to the left of it (see Figures 4 and 5).

If you *want* tabular output, the grid model won't be a problem; just be aware of this limitation before you decide to use a report writer in your app (for more hints on using Crystal Reports, see the sidebar, “Eight Common Questions About Crystal Reports”).

I was also unable to get either version of Crystal Reports to pad my order numbers to five digits, as required by my “spec.” The order numbers are stored in the database as long integers, and Crystal Reports has no built-in Format\$ function. Crystal Reports Pro 4.0 supports User Function Libraries (UFLs)—DLLs containing custom C or C++ functions—so I could have created my own Format\$ function had I really needed it. I also could have exported the report's records to a temporary table, and reformatted the order numbers in the process. Neither solution, however, is particularly convenient—I would have appreciated more built-in formatting options.

I ran into another limitation in trying to generate my report. The version of

CRYSTAL.VBX that ships with VB can print only to Windows' default printer. Now, maybe I shouldn't gripe about this (after all, even Word for Windows has this problem), but in my opinion there's no excuse for a program of any consequence not to transparently support all the printers on a system. This is a feature that I insist on in all the apps I create, and that means my reporting tools must do so as well. Fortunately, it's not difficult to make Crystal Reports 2.0 work with nondefault printers—if you know how to call the functions in CRPE.DLL. Crystal didn't provide this information with VB 3.0, so as a public service, I'm going to do it now.

CRYSTAL.BAS (included in REPORT.ZIP) contains VB declarations for the functions in Version 4.0 of CRPE.DLL. Thirty-five of those functions are also available in the version that ships with VB. The one we're interested in is PESelectPrinter. Simply pass PESelectPrinter the name of the desired printer, the name of its driver file, and the port to which you want to print (see Listing 1).

I've also included a VBSelectPrinter function that fills a list box with the names of printers installed on a system and allows the user to select one (see Listing 2). SelectPrinter returns the device, driver, and port names required by PESelectPrinter. Using these two functions, it's easy to present your users with a list of installed printers and to print reports to any device they select—even if you're using Crystal Reports 2.0. You won't have to mess with PESelectPrinter if you use Version 4.0 because it has PrinterName, PrinterDriver, and PrinterPort properties. But you'll still need some way to let the user choose a printer, so hang on to the SelectPrinter function.

DOING IT YOURSELF

I'm sure you've heard the obsessive-compulsive's motto, "if you want some-

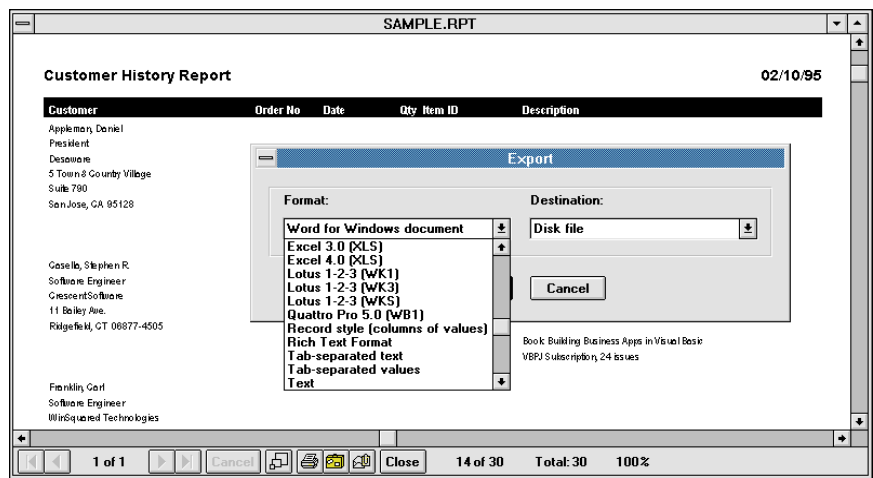


FIGURE 5 *That Modern Look.* Crystal Reports Pro 4.0's preview window looks more up to date. It also allows your programs to export reports in a variety of popular formats without requiring you to write additional code.

thing done right, you'll have to do it yourself." Unfortunately, when it comes to creating reports, it's true: the only way I was able to meet my design specification was to hand-code the report in VB. I've already mentioned the overwhelming drawback to this approach: it takes *forever!* Before you decide to forego a report writer, consider the productive programmer's motto: "Good enough is just right."

I estimate that it took me just over four hours to write and debug the code to create my report in VB. That doesn't include the additional four hours I spent designing and coding the report viewer. But I'll only have to do that once, and now that it's done, you won't have to do it at all. I'm pretty darn proud of it, too (check it out in Figure 6). I expected the formatting aspect of the code to take a while, but I was surprised at how difficult it was to select records to include in the report.

Remember, I wanted the data for all customers who had ordered more than one item. Here's the selection formula I used in Crystal Reports to get that information:

```
Count ({Order Details.Item ID}, _
      {Customers.Cust ID}) > 1
```

And here's the SQL query I ended up with to get the same set of records in Visual Basic:

```
SELECT DISTINCTROW Orders.[Cust ID], _
      Orders.[Order No], Orders.Date, _
      [Order Details].Qty, _
      [Order Details].[Item ID], _
      [Order Details].Description
FROM (Customers INNER JOIN Orders ON _
      Customers.[Cust ID] = _
      Orders.[Cust ID])
INNER JOIN [Order Details] ON _
      Orders.[Order No] = _
      [Order Details].[Order No]
```

```
DefInt A-Z
Option Explicit
Declare Sub PECloseEngine Lib "CRPE.DLL" ( )
Declare Sub PEClosePrintJob Lib "CRPE.DLL" ( _
    ByVal hJob)
Declare Function PEOpenEngine Lib "CRPE.DLL" ( )
Declare Function PEOpenPrintJob Lib "CRPE.DLL" ( _
    ByVal sRptName As String)
Declare Function PEOutputToPrinter Lib _
    "CRPE.DLL" (ByVal hJob, ByVal nCopies)
Declare Function PESelectPrinter Lib "CRPE.DLL" ( _
    ByVal hJob, ByVal sDrv As String, ByVal sDev _
    As String, ByVal sPort As String, lpDevMode As _
    Any)
Declare Function PESTartPrintJob Lib "CRPE.DLL" ( _
    ByVal hJob, ByVal bWait)

Dim hJob As Integer
Dim bAbort As Integer
Dim iResult As Integer

'Initialize CRPE.DLL
```

```
iResult = PEOpenEngine()

'Obtain a job handle for this report
hJob = PEOpenPrintJob(sRptName)

'Let user choose a printer
bAbort = SelectPrinter(sDev, sDrv, sPort)

'If user didn't click 'Cancel'...
If Not bAbort Then
    'Tell CRPE which printer to use
    iResult = PESelectPrinter(hJob, sDrv, sDev, _
        sPort, ByVal 0)
    'Print the report
    iResult = PEOutputToPrinter(hJob, 1)
    iResult = PESTartPrintJob(hJob, True)
End If

'Free resources
Call PEClosePrintJob(hJob)
Call PECloseEngine
```

LISTING 1 *Picking a Printer in Crystal Reports.* This code calls the SelectPrinter function to allow the user to choose an output device. It then calls the Crystal print engine functions to send a report to the selected printer.

```
WHERE ((Orders.[Cust ID] In
(SELECT DISTINCTROW _
Customers.[Cust ID]
FROM (Customers INNER JOIN Orders ON _
Customers.[Cust ID] = _
Orders.[Cust ID])
INNER JOIN [Order Details] ON _
Orders.[Order No] = [Order _
Details].[Order No]
GROUP BY Customers.[Cust ID]
HAVING (Count([Order Details]._
[Item ID]))>1)))
ORDER BY Customers.[Last Name], _
Customers.[First Name];
```

As you might imagine, devising that rat's nest took some time (I must acknowledge the assistance of Bruce Hawkins. Without his help this article would have been tardy).

But the additional development time pays greater dividends than just control over the printed output. Earlier, I mentioned that Crystal Reports requires you to distribute up to 2.3 MB of support files with your application. The custom controls I used in VBDEMO total under 172K (and my EXE file is 3K smaller than CRxxDEMO). Also, when I ran the three programs against a database containing

more than 10,500 customers and 9,800 orders, VBDEMO was more than twice as fast as CR20DEMO (see Table 1).

The sample report definition for this

article's CR20DEMO program contains a pair of nested "groups" (Crystal Reports lingo for logical subsets of records, usually used for sorts or subtotals). In

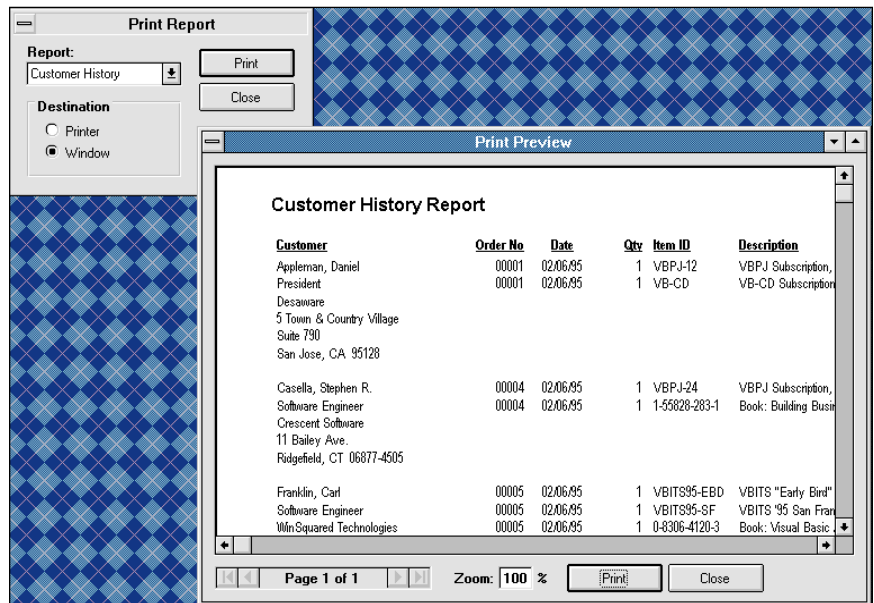


FIGURE 6 *Not Bad, If I Do Say So Myself!* This preview window was created with VB and VideoSoft's VSVIEW.VBX. Unlike Crystal Reports' preview windows, it allows the user to view the report at anywhere between zero and 200 percent of its actual size.

```
DefInt A-Z
Option Explicit
Global Const MODAL = 1
Declare Function GetProfileString Lib "Kernel" _
    (ByVal lpAppName$, ByVal lpKeyName As Any, _
    ByVal lpDefault$, ByVal lpReturnedString$, _
    ByVal nSize)

Function SelectPrinter (sDev As String, sDrv As _
    String, sPort As String)

    ' Calls GetProfileString to obtain a list of
    ' installed printers. Copies list to list box
    ' (lstBox on form frmList) and allows user to
    ' select a printer. Function then parses WIN.INI
    ' entry for selected device and returns device
    ' name(sDev), printer driver (sDrv) and default
    ' port (sPort).
    ' Return value: True if user clicks 'Cancel' on
    ' printer selection dialog, False otherwise.

    Dim I As Integer
    Dim iResult As Integer
    Dim iStart As Integer
    Dim sAppName As String
    Dim sNull As String
    Dim sTemp As String
    Dim sZ As String
    Dim sBuffer As String * 1024

    sAppName = "devices"
    sNull = "": sZ = Chr$(0)

    ' Get list of installed printers
    ' (Passing 0& as second parameter tells
    ' GetProfileString to return all items in the
    ' [devices] section of WIN.INI. List is
    ' delimited by nulls, with two nulls following
    ' the last item.)
    iResult = GetProfileString(sAppName, 0&, _
        sNull, sBuffer, Len(sBuffer))
```

```
frmList!lstBox.Clear
' Fill lstBox with device names
iStart = 1
Do
    I = InStr(iStart, sBuffer, sZ)
    sTemp = Mid$(sBuffer, iStart, I - iStart)
    If Len(sTemp) = 0 Then Exit Do
    frmList!lstBox.AddItem sTemp
    iStart = I + 1
Loop

' Select first item by default
' and show list
frmList!lstBox.ListIndex = 0
frmList.Show MODAL

' Selected text returned in Tag property...
sDev = frmList.Tag
If Len(sDev) Then
    ' Get info about selected device
    iResult = GetProfileString(sAppName, sDev, _
        sNull, sBuffer, Len(sBuffer))
    ' Parse driver and port info
    If iResult Then
        sTemp = Left$(sBuffer, iResult)
        I = InStr(sTemp, ",")
        sDrv = Left$(sTemp, I - 1)
        sPort = Trim$(Mid$(sTemp, I + 1))
    End If
    SelectPrinter = False
Else
    ' Set abort flag if user clicked
    ' the 'Cancel' button
    SelectPrinter = True
End If

' Free list form resources
' and return
Unload frmList

End Function
```

LISTING 2 *Freedom of the Printers.* The SelectPrinter function uses the GetProfileString API function to obtain a list of printers installed on the system, and to return the device, driver, and port names to the calling program.

order to see the items ordered by a specific customer, I created a group on the CustID field, a unique number assigned by Access to each customer record. But CustID corresponds to the sequence in which customers were entered into the database. This sequence is not very useful as the basis for a report. So I added a second grouping on each customer's last name, which causes the CustID groups to be sorted in alphabetical order. So far, so good.

With small data sets, the nested groups worked fine in both versions of

cratenates each customer's last name, first name, company name, and telephone number, creating a single value that can be used for both grouping and sorting. This approach allowed the original version of Crystal Reports Pro 4.0 to complete my report in 7 minutes, 37 seconds.

Curiously, Crystal Reports 2.0 took about a minute *longer* to run the report with this method—5:11—than it did using the nested groups.

By the time you read this, Crystal will have begun shipping the updated version of Crystal Reports Pro 4.0 that

WHAT YOU GAIN IN DEVELOPMENT SPEED, YOU LOSE IN REPORTING FLEXIBILITY.

Crystal Reports. But when I tested Crystal Reports Pro 4.0 against my 10,000-record customer database, it began to misbehave. About 80 percent into the generation of the report, Crystal Reports Pro acted like it was finished: the record counter stopped counting and the preview window's page navigation buttons were enabled, but the report was blank.

At first, Crystal tech support was baffled. But when I sent them my report definition and my 5 MB database file, the problem became clear: by nesting two groups on a large database, I was creating something like 600,000 record sets for Crystal Reports to keep track of! Interestingly, the version of Crystal Reports 2.0 included with VB was able to handle this ridiculous task (and rather speedily, too), but Crystal Reports Pro 4.0 just got confused and took the rest of the day off.

Crystal supplied me with two ways to fix this problem. First, they compiled a new version of Crystal Reports Pro (ver. 4.0.1.3) that no longer caches groups in memory. This new version was able to complete my report, but at a glacial pace: it took 9 minutes, 45 seconds to accomplish what Crystal Reports 2.0 did in 4:07.

The second fix involved a change to my report definition. Instead of grouping on CustID within Last Name, I created a single group based on the result of a user-defined function. The function con-

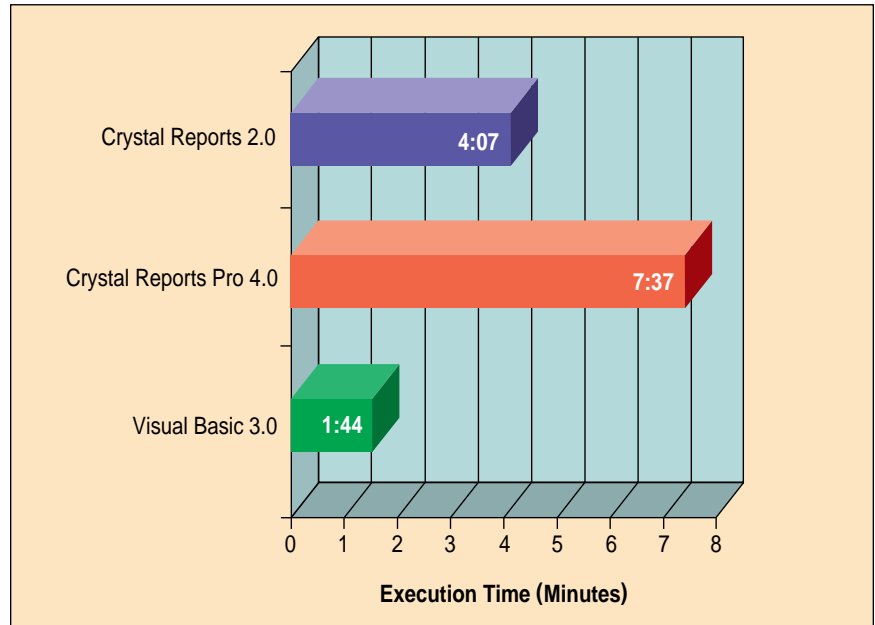


TABLE 1 Execution Times of CR20DEMO, CR40DEMO and VBDEMO. My report was run against a database of more than 10,500 customers and 9800 orders. I had to alter it to run against a user-defined function concatenating customer last name, first name, company, and telephone number to enable Crystal Reports 4.0 to run it.

Third-Party Reporting Tools

Crystal Reports may be the market leader among VB programmers, but it's not the only game in town. Borland bundled ReportSmith with its new Pascal language product. Concentric Data Systems' R&R Report Writer has long been popular among the xBase crowd, and the recent release of the SQL Edition is sure to gain popularity on other database platforms. And while it's not a dedicated report writer, VideoSoft's VSView is useful for controlling printed output for building do-it-yourself print reporting tools. Contact these vendors for more information.

Borland ReportSmith for Windows Version 2.5 (\$100)
Borland ReportSmith Server Version 2.5 (call for pricing)
Borland International
800-645-4559
408-431-1000

Crystal Reports Professional Version 4.0 (\$395)

CrystalReportsServerVersion4.0(\$1195)
Crystal, A Seagate Software Company
800-663-1244
604-893-6301

MicroHelp Report Generator 3 (\$249)
MicroHelp Inc.
800-922-3383
404-516-0899
Fax: 404-516-1099

R&R Report Writer Version 6 for Windows, SQL Edition (\$395)
R&R Report Writer Version 6 for Windows, xBase Edition (\$249)
Concentric Data Systems Inc.

800-325-9035
508-366-1122
Fax: 508-366-2954

VSView (\$99)
VideoSoft
800-547-7295
510-704-8200
Fax: 510-843-0174

—P.W.

solves the group limitation. If you already have Version 4.0 and run into this limit (Crystal says the current version can handle about 6000 groups on a typical machine), registered users of Crystal Reports Pro 4.0 can request an upgrade free of charge on Crystal's

CompuServe forum (GO REPORTS), by posting a message on Crystal's tech support BBS (604-681-9516), or by calling Crystal tech support.

So, if you want your app to print and preview reports, how do you choose between commercial report writers and the

do-it-yourself method? The answer depends on your priorities. If want the smallest, fastest possible programs and you must have complete control over the appearance of the output, you should create your reports in code.

However, you may be able to achieve

Eight Common Crystal Reports Questions

The Crystal Reports Forum on CompuServe (GO REPORTS) is always one of the most active discussion areas in the Windows Component Vendor forums. Here are the answers to eight of the most frequently asked questions about using Crystal Reports with Visual Basic, courtesy of Crystal Technical Support.

Q HOW CAN I PASS SELECTION FORMULAS AND FORMULAS WITH VARIABLES OR TEXT BOXES FROM VB?

A Suppose you want to allow users to pass a record selection criterion from their VB application, instead of simply hard-coding a criterion into their Crystal Reports RPT file. Consider the situation where you have created an application with various text boxes that pertain to fields from a database. Typically, you would like to allow users to print a report based on the information that resides in those text boxes. You might also want to allow users to pass formulas to Crystal Reports at run time in the form of a text string (such as a report title) or in the form of calculated data (such as X % Y).

A key thing to remember is that the formulas must conform to Crystal Reports' formula syntax, where strings must be embedded in single quotes, while numbers must not be embedded in quotes. In order for Crystal Reports to parse a formula it receives, the formula must be identical to a formula you would enter directly within Crystal Reports itself. Using the MsgBox function in Visual Basic, it is easy to verify that the formula string to be sent actually matches that expected by Crystal Reports.

This code will let you use the SelectionFormula and Formulas properties of the Crystal VBX when using a string in your record selection criterion:

```
Sub Command1_Click ()
    'if we hard-code the name
    Report1.SelectionFormula = "{file.field} = _
    'Bob Brown'"
    'If we use the value of the Textbox
    Text1.Text = "Bob Brown"
    Report1.SelectionFormula = "{file.field} = '" _
    & Text1.Text & "'"
    Report1.action = 1
End Sub
```

And here is an example of using a date in a record selection criterion:

```
Sub Command1_Click ()
    'if we hard-code the date
    'We are interested in reporting only records of
    'March 04, 1993
    RecSelect$ = "{file.field} = Date(1993,03,04)"
    MsgBox RecSelect$
    'The result of this MsgBox should look identical
```

'to your record

'selection criteria within Crystal Reports itself
Report1.SelectionFormula = RecSelect\$

'if we use the value of a date/time variable of type
'Variant (E.g. We are interested in reporting only
'today's records)

TodayDate = Now

RecSelect\$ = "{file.field} = Date(" + Str\$(Year _
(CriticalDate)) + ",,"

RecSelect\$ = RecSelect\$ + Str\$(Month _
(CriticalDate)) + ",,"

RecSelect\$ = RecSelect\$ + Str\$(Day _
(CriticalDate)) + ",,"

Report1.SelectionFormula = RecSelect\$

Report1.action = 1

End Sub

Note: you must manipulate the date input by the user within your VB code to fit the exact format of DATE(yyyy,mm,dd). If you try to pass any other format you will receive an "Error in Formula."

If you want to change a formula or a string (such as page title or author) at run time from Visual Basic, you would take similar steps.

First, in Crystal Reports, create your formula. If you are giving it a string value, you must place a string value in it. This will force Crystal to treat the formula as a string. A single space inside double quotes is a good idea because this will not print if left unchanged—yet it holds the string type for when you change the actual value later from VB. Place the new formula on your report where you would like the passed value displayed.

If you need to send a numeric value or date, again place an appropriate value in the formula in order to establish the data type of the formula. To pass a numeric value to a formula, place a zero in the formula when you create the formula in Crystal Reports. To pass a date value to a formula, place the Today function in the formula.

If you want to send a title to your report from your VB application, this code uses the suggested syntax to change the title at run time:

```
Sub Command1_Click ()
    'if we hard-code the title
    Report1.Formulas(0) = "Title = 'Report on _
    Telephone Usage'"
    'If we use the value of the Textbox
    Text1.Text = "Report on Telephone Usage"
    Report1.Formulas(0) = "Title = '" & Text1.Text & "'"
    Report1.action = 1
End Sub
```

This code changes the value of a numeric formula at run time:

acceptable output and performance from a report writer—it depends on the complexity of the report and the organization of the data—and you will almost certainly save development time if you use one. At that point you'll have to consider whether it bothers you to ship

the reporting tool's support files with your app. If the program is already large (or uses the corpulent Access engine), another couple of megabytes won't make much difference.

I'm one of the more anal-retentive programmers you'll ever meet, but after

creating a moderately complex report by hand in VB, I think I'll learn to live with my report writer's limitations. Of course, there's no reason you can't use a third-party tool for the simple reports, and hand-optimize the really important ones. Hmmm . . . ■

```
Sub Command1_Click ()
    'if we hard-code the numeric value
    Report1.Formulas(1) = "Temperature = 65"
    'If we use the value of a variable
    new_temperature% = 65
    Report1.Formulas(1) = "Temperature = " & _
        str$(new_temperature)
    Report1.action = 1
End Sub
```

Q HOW CAN I CONNECT TO SQL USING THE CRYSTAL REPORTS VBX?

A Use this connect string:

```
Report1.Connect="DSN=DataSource name;UID=ID;_
    PWD=Password;DSQ=Tablename"
```

Q HOW CAN I USE CRYSTAL REPORTS AS A VB MDI CHILD WINDOW?

A The Crystal Reports print window can be displayed as an MDI child. The basic concept is to create an MDI Child frame in VB and then make the Crystal print window a child of that child form. In the PEOutPutToWindow call, if you are using the Print Engine, the last parameter allows you to specify a ParentWindow handle. If you are using the VBX, you would use the Report1.WindowParentHandle property (this is a runtime-only property). If the ParentWindow Handle is the handle of an MDI frame window then Crystal's print window will act as an MDI child. The next step is to use the Windows API calls GetWindow and MoveWindow.

The GetWindow function call is declared in the APPLICATION.BAS file and is used to retrieve the handle of a window that has the specified relationship to the given window. The function searches the system's list of top-level windows, their associated child windows, the child windows of any child windows, and any siblings of the owner of a window. The second parameter of the GetWindow function call allows you to specify a relationship flag:

GW_CHILD identifies the MDI frame window's first child, which would be the Crystal Reports print window.
 GW_HWNDFIRST returns the first sibling of the child window. Otherwise it will return the first top-level window in the list.
 GW_HWNDLAST returns the last sibling of the child window. Otherwise it will return the last top level window in the list.
 GW_HWNDNEXT returns the sibling window that follows the given window in the window manager's list.
 GW_HWNDPREV returns the previous sibling window in the window manager's list.
 GW_OWNER identifies the window's owner.

The MoveWindow function call is declared in the APPLICATION.BAS and is used to change the position and dimensions of a window. For top-level windows, the position and dimensions are relative to the upper-left corner of the

screen. For child windows, they are relative to the upper-left corner of the parent window's client area.

Both of these Windows API function calls are used in the MDI Child form Paint and Resize procedures to ensure that when the Crystal print window appears the MDI child frame will exactly match the size of the Crystal Reports Print window, and when the MDI child frame is resized the Crystal print window will resize accordingly.

Q HOW CAN I PASS A STORED PROCEDURE?

A This code will pass a stored procedure:

```
[form.]Report.StoredProcParam(Parameter Array Index%)_
    [= newParameter$]
```

For example, using this code:

```
Report1.StoredProcParam(0) = "06/14/1989"
```

sets the first stored procedure parameter to the date June 14, 1989.

Q HOW CAN I GET THE STRUCTURE SIZE?

A This code will retrieve the size of a Structure:

```
Dim logonInfo As PELogonInfo
logonInfo.StructSize = Len(logonInfo)

logonInfo.ServerName = "ODBCSQL" + Chr$(0)
logonInfo.DatabaseName = "pubs" + Chr$(0)
logonInfo.UserID = "tech" + Chr$(0)
logonInfo.Password = "tech" + Chr$(0)
```

Q HOW DO I SET THE LOCATION TO A SERVER NAME OTHER THAN THE ONE ON THE DEVELOPMENT MACHINE? THE SERVER NAMES ARE DIFFERENT BUT THE TABLE AND FIELD NAMES ARE THE SAME ON THE VBX.

A To change the location of a server, connect to the new server and use the Datafiles() property for each table.

Q HOW DO YOU SET THE TABLE LOCATIONS ON STORED PROCEDURES?

A Use the Datafiles() property. Note that you cannot link stored procedures in Crystal Reports.

Q WHY DOES CLOSING THE CRYSTAL REPORTS ENGINE DESTROY SQL CONNECTIONS WHEN USING NATIVE Q+E DRIVERS?

A This is a problem with all Q+E 1.x drivers. The workarounds for this problem are to use ODBC, log off before calling Crystal Reports, or use the Q+E 2.x drivers.