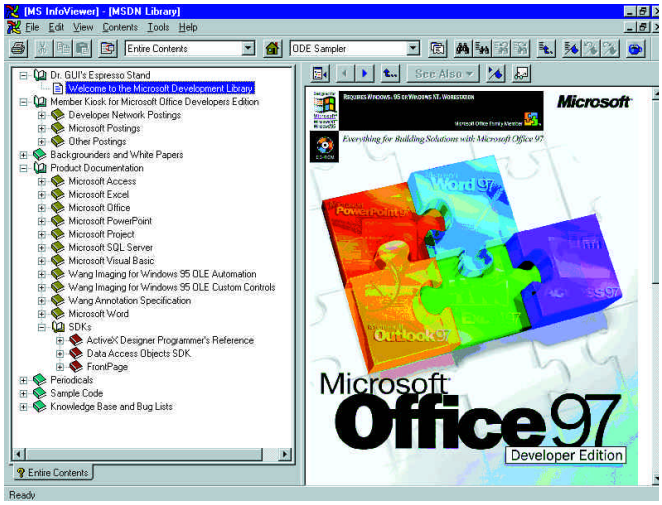
Open and shut case

Office 97… just more of the same and not worth the upgrade? To casual users, maybe; but for developers, it offers a more open programming environment. Tim Anderson explains.

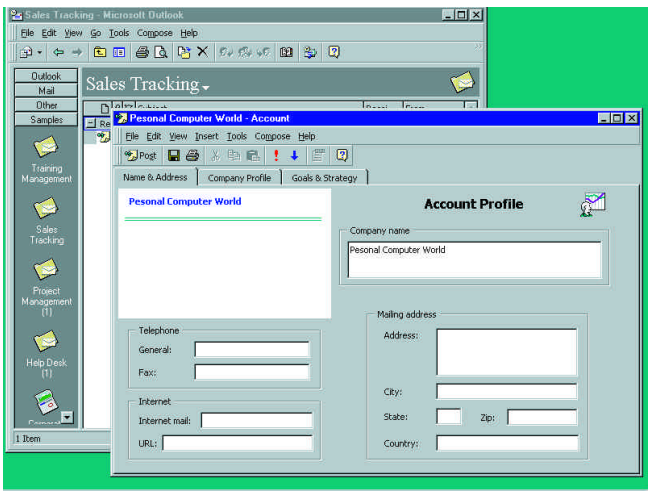
I have heard muttering to the effect that Office 97 is not much different from Office 95, or even Office 4.x. From some perspectives, that is correct. Word looks similar, Excel looks similar and all the casual user will notice at first is the Office Assistant (fantastic or horrific, according to taste) and a new, flatter, look to the toolbars.

But developers should welcome Office 97 with open arms. It is even worth explaining to the users why they really should upgrade, even if the animated Clippit is not their cup of paperclips. The reason is Visual Basic for Applications combined with the updated Office object model, all of which is exposed for programming. In most cases, equipping an Office user with suitable templates and macros soon pays for itself in increased productivity.



Left The Office 97 Developer edition is great if you need Access runtime, but otherwise it is not essential

Below, left You can develop applications in Microsoft Outlook by creating custom forms driven by VB Script. Nearly wonderful, but not quite there yet



To do these wonderful things, you need appropriate resources. This might or might not include the Office 97 Developer Edition, Microsoft's one-stop solution. What you get is the Office Professional CD, plus a further CD of developer tools. There are also two books: the *Office 97 Visual Basic Programmer's Guide* and *Building Applications with Access 97*, along with a booklet of Object Model charts.

Before getting too excited, though, it is worth recalling that the original Office Development Kit CD used to be more or less given away by Microsoft, presumably on the grounds that it pays to have people develop Office solutions, since in order to deploy them a copy of Office must be purchased for each installation.

Another noteworthy detail is that the books in the Developer Edition are available separately from Microsoft Press. A third observation is that not all the documentation you might need is actually included, neither on paper nor online. A notable example is *Building Microsoft Outlook 97 Applications*, which would be particularly valuable as Outlook is brand new. Another noteworthy example is the Office 97 Resource Kit: the version on the CD is for Office 95, or at least it is in my US shrinkwrap copy. The resource kit is aimed at network administrators but contains useful information for developers as well.

The most essential developer reference, the VBA reference for each application, is actually on the Office 97 Professional CD so it is questionable whether the Developer Edition is worth having. Most of the material can also be found on the Microsoft web site, often in updated form. There is only one convincing reason for buying this product: to obtain the runtime version of Access 97. This gives you a licence to deploy Access applications royalty-free, and could soon pay for itself. If you don't need it, just subscribe to MSDN (or buy a library

p302 ➢

Fig 1 Using the clipboard

```
Dim cr As String
cr = Chr$(13) & Chr$(10)
RichTextBox1.Text = "This is a picture" + cr
RichTextBox1.SelStart = Len(RichTextBox1.Text)
RichTextBox1.SetFocus
Clipboard.Clear
Clipboard.SetData Image1.Picture, vbCFBitmap
SendKeys "^v"
```

CD from time to time) and buy the books you really need from a bookshop.

Outlook: nearly great

Is it a Personal Information Manager, or an email client, or maybe groupware? Outlook is ambitious, and nearly the foundation of a complete Office solution. For example, it should be possible, with a bit of customisation, to right-click a contact name and open up a customer's order history or an index of previous correspondence.

There are two snags, though. One is that Outlook has VB Script but not yet Visual Basic for Applications. The second is that you need Exchange Server to do anything serious with Outlook over a network, like sharing an address book or viewing other people's calendars. In fact, Outlook without Exchange Server is less capable than the old Schedule, a fact which has not gone down well with small businesses running peer-to-peer networks. Exchange Server needs Windows NT, is priced for the Enterprise market and needs client licences, too, which makes Outlook far less attractive. Incidentally, if you decide to get going with Outlook development, a trip to Microsoft's web site is essential. Documentation and numerous sample applications are available for free download.

Sheridan's Active Threed

Sheridan products now come on a Toolkit CD containing all the company's developer tools. You can install demonstration versions of any tool, or full versions where you have the right key code. For instance, if you purchase Active Threed you get the code for this product along with the CD. There is no manual, the lame excuse being that Sheridan wanted to check the printed manual against the release code. A voucher lets you obtain it at nominal cost. But the manual aside, the all-in-one CD is a great idea. Another plus is that the ActiveX controls come digitally signed with CAB versions included for web distribution.

Active Threed offers seven controls which are intended as plug-in replacements for the standard Windows items like command buttons and check boxes, but with extra features. These include marquee captions which blink, scroll, slide and bounce, plus animated pictures created with a sequence of bitmaps. There is also a splitter control which lets you create windows with resizable panes. The controls are 32-bit only; not even Visual Basic 4.0 16-bit is supported. Packages like this cause me to hesitate since many VB applications are slow enough without the additional weight of controls which aren't strictly necessary.

Two things make ActiveThreed worth a second look, though. Firstly, the SSplitter control is well implemented and provides a feature which has become something of a Windows standard. Secondly, the SSRibbon control allows you to create toolbar icons with an active border, as seen in Office 97 and Internet Explorer 3.0. I was also glad to find that Delphi samples had been included.

Rich Text in Delphi and VB

Dr Francis Burton asks: "*I want to be able to alternate graphics and text in a scrollable window, with the ability to cut and paste text (and possibly bitmaps/metafiles, too). New text and graphics are appended to the end of the window. Do you know of any controls, either Visual Basic or Borland Delphi, which implement scrollable graphics/text windows?*"

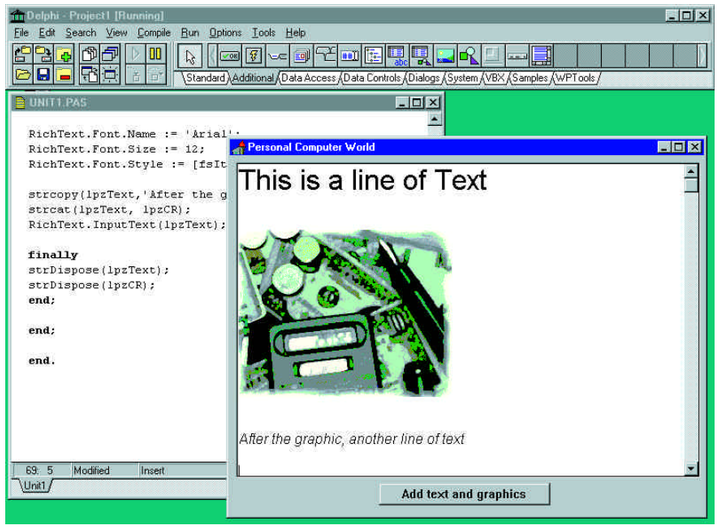
If you are working in Windows 95 or NT, the standard rich text control can display formatted text and graphics. It is easy to miss this functionality in Visual Basic, since there is no InsertPicture method. You can insert OLE objects, though, using the OLEObjects collection. For example, this code inserts a line of text and a picture:

```
RichTextBox1.Text = "This is a
picture"
RichTextBox1.OLEObjects.Add , ,
"c:\test.bmp"
```

Unfortunately, this can have unpredictable results depending on how OLE file associations are set up in the registry. There is also an overhead involved with OLE which makes the rich text box update rather slowly when an object is inserted. A safer approach would be to use the clipboard. The example in Fig 1 and the Delphi one (Fig 2) assume you have placed the picture you want to insert into an invisible image control on a form. The final

Fig 2 Using WPTools

```
var
lpzCR: pchar;
lpzText: pchar;

begin
lpzText := stralloc(256);
lpzCR := stralloc(3);

try
strcopy(lpzCR,chr(13));
strcat(lpzCR, chr(10));

RichText.clear;
RichText.Font.Name := 'Arial';
RichText.Font.Size := 24;

strcopy(lpzText,'This is a line of Text');
strcat(lpzText, lpzCR);

RichText.InputText(lpzText);

RichText.PicInsert( image1.picture, 0,0 );
RichText.InputText(lpzCR);
RichText.Font.Name := 'Arial';
RichText.Font.Size := 12;
RichText.Font.Style := [fsItalic];

strcopy(lpzText,'After the graphic,
another line of text');
strcat(lpzText, lpzCR);
RichText.InputText(lpzText);

finally
strDispose(lpzText);
strDispose(lpzCR);
end;
```

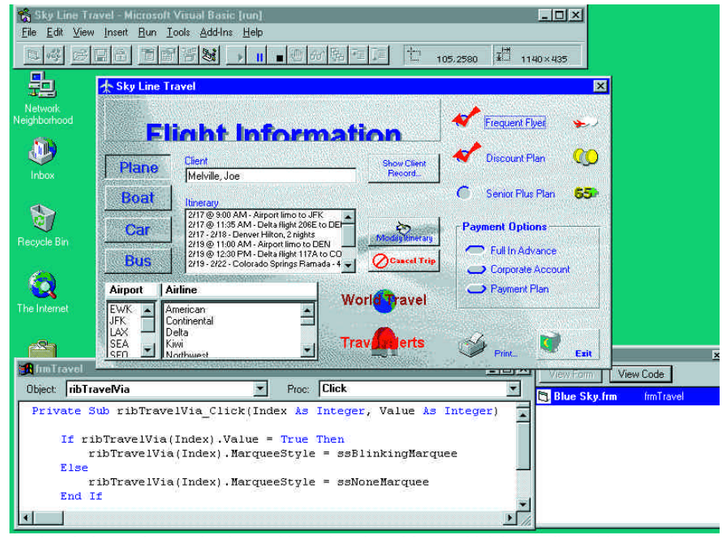
Even Delphi 1.0 is able to display text and graphics within a scrolling document using WPTools (illustrated here), or a component such as Visual Writer

SendKeys statement simulates the CTRL-V keypress which pastes from the clipboard at the insertion point.

For some reason, Delphi's equivalent rich text control does not support graphics. It is odd, since both use the same underlying common control, and it is likely that careful investigation of the Visual Component Library would reveal a way to overcome the problem by creating a new component which exposes more of the features in the rich text control. Or, you could use VisualWriter, an OCX and VBX control which does support graphics. Better still, use a native Delphi component like WPTools (it's shareware but works well). The WPRichText control, part of WPTools, has a PicInsert method which lets you insert a picture. It also support fonts and styles so you could implement scrollable text and graphics as required. The main snags with WPTools are its uneven documentation, and extensive use of pointers which can be error-prone. Fig 2 shows example code using WPTools.

String along with SQL

Michael O'Reilly writes: "*I have been following your excellent VB tutorial, and while using some of the code given in the February issue article I encountered a problem I can`t solve. In the example given, you take a string from a text box and use it in an SQL query. How do you do the same with a number from a text box?*"



A printed picture does no justice to this Active Threed form, which is crawling with animation. Note the split window at bottom left — a genuinely useful feature

When you create an SQL string for querying a database, you use a different technique according to whether the field is character or numeric. If it is the former, the value must be in single quotation marks, as in:

```
sSql = "select * from
members where
members.surname = '" &
mySurname & "'"
```

Admittedly this looks ugly, but it works well. If the field is numeric, then the single quotation marks must not be used. All you need to do is convert the numeric value to a string like this:

```
sSql = "select * from members where
members.surname = " &
str$(myID)
```

The following question comes from Andrew Shaw: "*I need a lot more input boxes than the PCW Sports Club uses and want to implement some kind of counter/loop to run through the DisplayPerson code. I want to avoid:*

```
txtForename = CurrPerson.Forename
txtSurname = CurrPerson.Surname
```

"*I tried an array of text boxes and a laborious trawl through the manual, with no success. I want to try something like:*

```
txtInput(Counter) =
CurrPerson.Counter
```

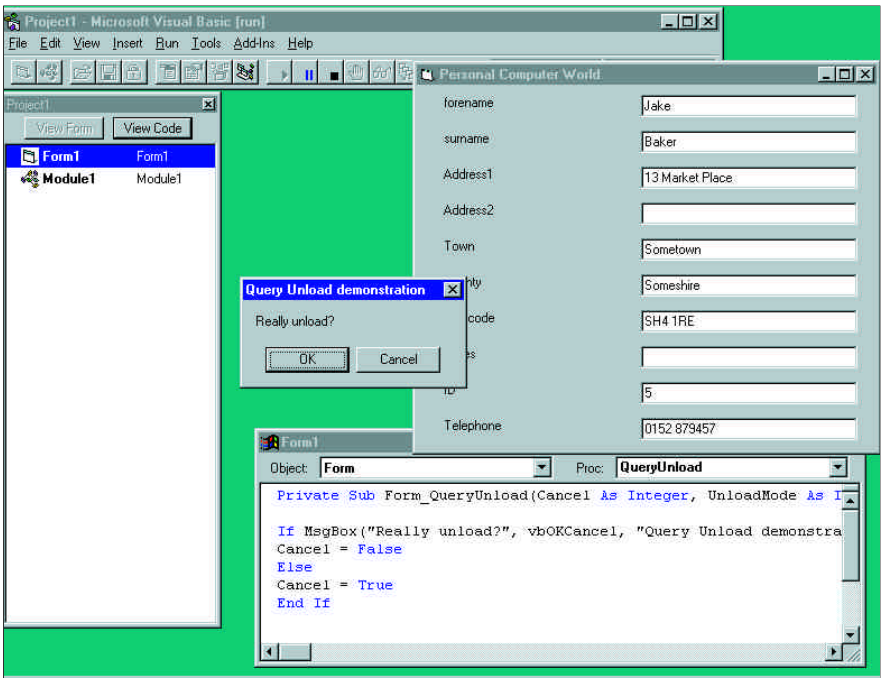
"*What should the CurrPerson.Counter part look like?*"

Andrew's idea is to write a loop that iterates through all the fields of a particular record, filling text boxes with the values along the way. This can be done as follows:

```
Dim iCountvar As Integer
For iCountvar = 0 To
 (ds.Fields.Count - 1)
Label1(iCountvar).Caption =
 ds.Fields(iCountvar).Name
Text1(iCountvar).Text = "" &
 ds.Fields(iCountvar).Value
Next
```

The trick is to get at the Fields collection of a Recordset object. You could make the routine even more flexible by creating the necessary labels and text boxes at runtime.

p305 ➢



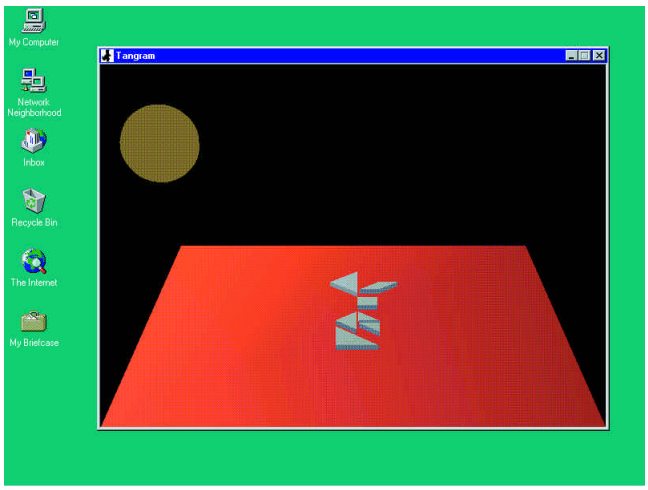


Fig 3 (above) Using the QueryUnload event to confirm a close decision

Left (see "Inside Com") This application from Inside COM demonstrates agregation, containment, and interchangeable components. The Tangram pieces look like a rabbit… allegedly

The main value of a routine like this is in applications where the number and type of fields in the recordset may vary at runtime. For example, you could let the user choose which fields they wanted to view, build an SQL string to return just those fields, and display them using the procedure described.

Where's the tab strip?

Phil Richard asks: "*In a recent column, you mentioned a tab strip option in VB4 which I would like to use; either TabStrip or SSTab. Neither seem to be included in my installation of Standard Edition VB4, which I purchased as an upgrade. I have, however, found TABCTL32.OCX from the Sheridan web site. Is there a way of registering it, as it appears as not found when trying to load the PCWClub project.*"

Unfortunately Phil is correct, and the Tab custom controls are only present in the Professional edition of Visual Basic. While a lot can be done with the Standard version, it is severely restricted both in its use of the JET database and in the number of custom controls supplied. It is also cheap, and my guess is that Microsoft views it as an introductory, learning product rather than a real development tool.

With TABCTL32.OCX, most third-party OCX vendors allow free distribution of its controls. So in order to make some sales, use of an OCX in developing an application is allowed only if you have purchased the control. When you do buy it, you get either a .LIC file or special registry entries that allow you to use it for development.

How do I cancel?

Ammar EL-Hassan has this query: "*I am trying to add a facility to enable the user of my VB application to CANCEL the operation which Closes his form. The user clicks the control box in the top left corner of the form. They then click the Close option, which unloads the form (application dead!). How can I interrupt this to enable the user to cancel after selecting Close?*"

VB forms have a QueryUnload event for this purpose (Fig 3). Use code like this:

```
Private Sub Form_QueryUnload(Cancel
 As Integer, UnloadMode As Integer)
If MsgBox("Really unload",
vbOKCancel) = vbOK Then
Cancel = False
Else
Cancel = True
End If          End Sub
```

You can even discover why the form is trying to close, by inspecting the UnloadMode parameter. If it is vbAppWindows, then the user is trying to close down Windows.

Inside COM by Dale Rogerson

Inside COM is a book that takes you step by step through the mysteries of COM interfaces, reference counting, globally unique identifiers, containment, aggregation and automation. All the examples are in C++ but the author has avoided Windows-specific code where possible. The strength of the book is that it is about COM rather than OLE or ActiveX technologies which are based on COM, so it does a good job of explaining what COM is and how it works. Of course, the impressive thing about tools like Visual Basic and Delphi is that you can use COM without needing to understand much about it. When it comes to advanced development or troubleshooting, though, a book like this provides an invaluable background.

PCW Contacts

Tim Anderson welcomes your Visual Programming comments, queries and tips. Contact him at the usual *PCW* address or email visual@pcw.vnu.co.uk.

Office Developer Edition is £639 (ex VAT). Upgrades from Office 97 Professional are £215 (ex VAT). Contact Microsoft 0345 002000
Office 97 Visual Basic Programmer's Guide available separately at £32.49 from Computer Manuals 0121 706 6000
Sheridan Active Threed £99 (ex VAT) from Contemporary Software 01344 873434
Inside COM (Microsoft Press) is £32.99 from Computer Manuals 0121 706 6000.
Visual Writer is £195 (ex VAT) from Visual Components 01892 834343
WP Tools is shareware. Contact Julian Ziersch 100744.2101@compuserve.com