Right, said **thread**…

Tim Anderson explores threads in Delphi 2.0, picks up snippets from the VBITS conference, and answers your VB queries.

Microsoft's theme for 1996 is the Internet, which featured strongly at the recent London VBITS conference for Visual Basic developers.

At one session, the presenter rashly asked how many delegates were actually developing for the Internet. A scattering of hands were raised. Okay, how many plan to develop for the Internet? A few more hands. The message is that while tools vendors steam ahead with Internet products, the actual developers are mostly stuck in the old world of databases, accounts and local networks.

Another key question is how many developers have switched to 32-bit Windows. Microhelp's VB or OLE tools is a new product, available in 16- and 32-bit versions, and distributor Contemporary Software, exhibiting at VBITS, reports that sales in the first quarter of 1996 were 57 percent in favour of the 32-bit product — a one-off statistic but an indication that the move to Windows 95 and NT is finally happening.

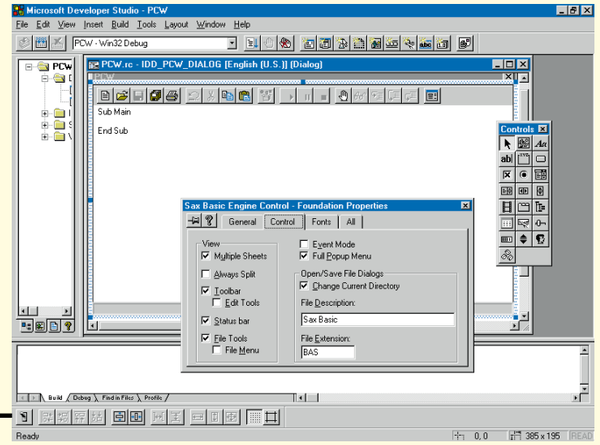
Those who did attend found a high standard of presentations, including API guru Daniel Appleman's demonstration of how to write a VB interface that runs as fast as C++. The answer is don't use controls, use VB's drawing methods instead. Of course, if you write VB applications like that you will be even less productive than your C++ counterpart. Even so, a point well made and a warning to go easy on controls, and especially VBX or OCX add-ons, if fast performance is a priority.

As expected, there are plenty of new Internet add-ons for Visual Basic and Visual C++. Microsoft's Internet Control Pack is a free download (beta at the time of writing) and contains OCX controls for integrating Web viewing, email, newsgroups and FTP file transfer into applications.

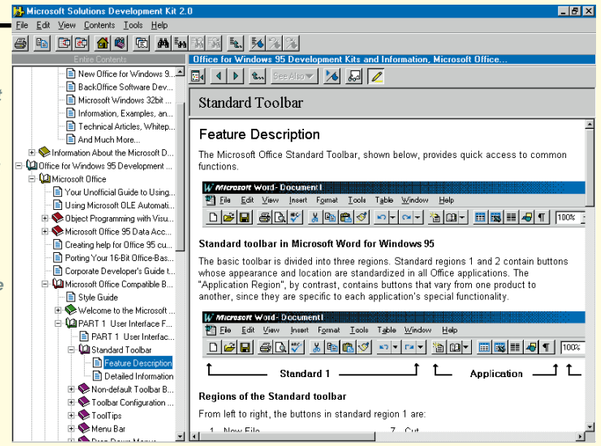
It's also been announced that Visual Basic 5.0, due out this year, will be able to create Active controls; another name for lightweight OCX components for Internet use.

The Internet again features in Visual C++ 4.1, an important update which adds MFC support for the Internet Information Server, Microsoft's Web server for Windows NT. A generous set of 12 third-party OLE controls has been added to Visual C++ 4.1, including Desaware's souped-up list box, the Sax Basic Engine for adding macro language support to your application, and Protoview's Interactive Diagramming Object for displaying data in the form of a diagram that can be visually modified by the user.

Finally, Microsoft has released the Solutions Development Kit, a CD which updates the Office Development Kit for those building applications with Office for Windows 95. For the latest news on Microsoft's tools, browse around the company's home page at http://www.microsoft.com.



Visual C++ version 4.1 comes with additional third-party controls including the Sax Basic Engine



The Solutions Development Kit is for high-level development with Microsoft Office. This page shows how to create an office-compatible toolbar

Books for Visual Programming

Wrox Press delivered one of the first books on Delphi 1.0, and now repeats the performance with *The Revolutionary Guide to Delphi 2.0*. Unfortunately, the trick is partly illusion, since much of the book covers 16-bit Delphi. This is a multi-author title aimed at those already competent with Delphi and attempts to cover every aspect of the package, making it a mixed bag. There are good chapters on debugging, component writing and the Windows API, along with skimpy coverage of the Borland Database Engine, ReportSmith, and issues specific to 32-bit Delphi. It would have been better to focus exclusively on Delphi 2.0 and cover fewer issues in greater depth. Nevertheless, the authors are knowledgeable and most Delphi developers will find plenty of valuable tips here.

● Charles Petzold's *Programming Windows 95* is a major new edition of a book revered by developers for its clear description of how Windows hangs together. The emphasis is on understanding Windows internals, starting with the creation and control of windows themselves and going on to include text and graphics, resources, memory management, input devices and dynamic link libraries. There are brand new chapters on the user interface, multitasking and multithreading and OLE, two of which are by co-author, Paul Yao. Given the huge complexity of Windows, Petzold is remarkably clear and concise. There is nothing here about Microsoft Foundation Classes, Visual Basic or Delphi, but simply an explanation of the Windows API with examples in C and occasionally C++. Highly recommended.

documentation for Delphi's multithreading support is some sketchy online help and one sample application. It is just another example of Delphi's rough-and-ready documentation, but is particularly disappointing given that this is unfamiliar territory for many developers. As it happens, Delphi's Visual Component Library includes a TThread object that simplifies multithreaded programming. What follows is a quick look at how it works.

Under Windows 95 and NT, each 32-bit running application is described as a "process" and has its own space in memory. A thread is an execution path within a process, sharing its memory but able to execute independently, with processor time allocated by the operating system. This means you can create applications which are more responsive, performing lengthy background tasks while remaining available to the user. Unfortunately, multithreading makes program design even more difficult and introduces new possibilities for bugs and conflicts: that is no reason to ignore threads, but it does suggest caution.

The example application is designed to cycle through 140,000 colour combinations, displaying the results in a panel control. Real-world applications would not do this, but might be rendering an image or downloading a file from the Internet, to name two common background tasks.

In order to spin this off as a separate thread, we derive a new thread class from TThread, declared as follows:

```
TPCWThread = class(TThread)
  private
    iRed: integer;
    iGreen: integer;
    iBlue: integer;
    thispanel: TPanel;
  protected
  constructor Create(panel: TPanel);
  procedure Execute; override;
  procedure UpdateColour;
  end;
```

All classes based on TThread must override the Execute method as this is the procedure which runs when the thread object is created. If you create the new thread class by choosing Thread Object from Delphi's object repository, the skeleton declaration will do this for you.

Keeping in step

The essence of multithreading is that at any time the operating system may switch processing time between one thread and another. This is no problem if the threads are truly independent, but what if they interact?

For example, TPCWThread needs to update a panel on a form. Other threads, including the main application thread, also have every right to update that panel. This is the reason for the warning comment that appears when you create a new Thread Object: "Important: Methods and properties of objects in VCL can only be used in a method called using Synchronize."

"Synchronize" is a TThread method which performs a vital function, letting you safely call VCL components such as Delphi forms and controls without conflicts. Synchronize takes a method name as its parameter.

In this example, there is an UpdateColour procedure which updates the panel, called from the main Execute method via Synchronize.

Calling the thread

When the user clicks the "Start a thread" button, the following code executes:

Parent problems

Chris John is contemplating a move to Visual Basic 4. He asks: "I have been thinking of upgrading from VB 3 to VB 4 but have been concerned about the problems of converting existing applications. I have written to Microsoft which has given me some comfort regarding conversion, but the company was rather non-specific when referring to API calls. Your reference to the API call SetWindowPos [in the April issue] has encouraged me. However, I have also used the call:

```
Declare Function SetParent% Lib "User" (ByVal H%, ByVal J%)
```

to enable me to add and remove member frames to, or from, an array of frames together with their contents (other control arrays) at runtime. Can you tell me what modifications to this call might be needed, or does version 4 provide for the addition of members to an array of frames together with their contents, which would do away with the need for an API call?"

Visual Basic 3.0 code should run fine in VB 4.0 16-bit version, but the move to 32-bits is problematic. For a start, VBX add-ons are not supported and although OCX versions are generally available, the transition is not always smooth.

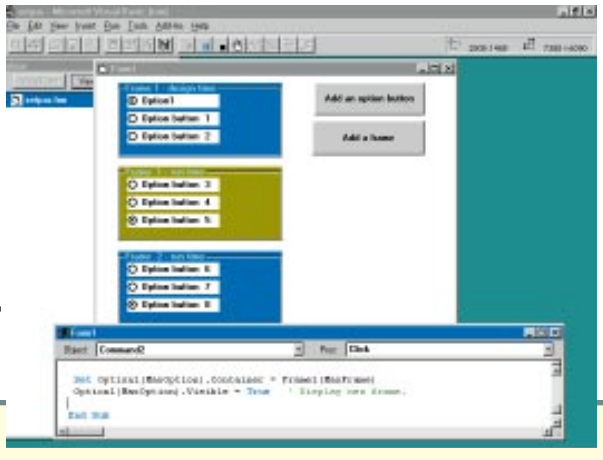
Next, although most API functions have a 32-bit equivalent, it is a different API and the declarations need changing. VB 4.0 comes with an API text viewer applet that lets you copy the declarations you need. In this case, the new declaration is:

```
Declare Function SetParent Lib "user32" (ByVal hWndChild
As Long, ByVal hWndNewParent As Long) As Long
```

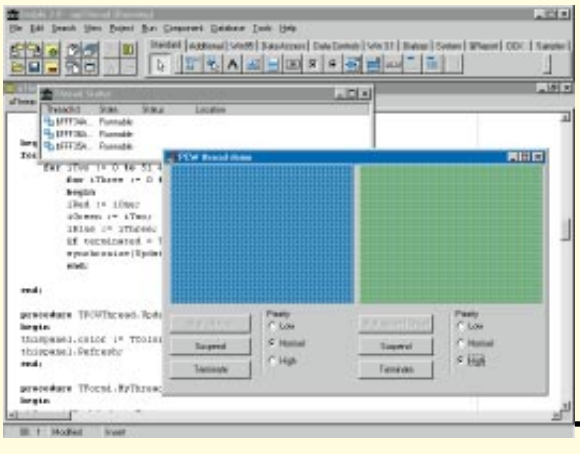
but the good news is that you probably don't need it.

In VB 3.0, although you can add new container controls like frames or picture boxes at runtime through a control array, the only way to add child controls to the new container is through the SetParent API call. VB 4.0 controls have a new Container property that overcomes the problem. For example, if you loaded a new frame at runtime, you could add an option button to it like this:

```
Load Option1(OptionIndex)
Set Option1(OptionIndex).Container = Frame1(FrameIndex)
Option1(OptionIndex).Visible = True   ' Display new button
```



This application shows how a control's Container property is used to add option buttons to a frame at runtime



You can check on Threads in a Delphi 2.0 application by showing the Thread Status window. In this demonstration, three are running, one for the main application thread and two TPCWThread objects. The user can control the priority of each thread while it is running. A thread can also be suspended or terminated

```
procedure TForm1.cbStart1Click(Sender: TObject);
begin
cbStart1.Enabled := False;
MyThread:= TPCWThread.create(Panel1);
Mythread.FreeOnTerminate := true;
Mythread.Priority := tpNormal;
rbNormal1.Checked := True;
Mythread.OnTerminate := MyThreadTerminate;
end;
```

To avoid creating two MyThread objects, the first line of code disables the button. Next, the thread object is created with the display panel passed to the constructor. The FreeOnTerminate property is set to true, which means the thread object is automatically destroyed when the thread stops running. Then, one of seven priority values is assigned, controlling how large a slice of processor action the thread receives. A corresponding radio button is checked.

Finally, a procedure is assigned to the Terminate event, so that the application can take appropriate action when the thread finishes its work. In this case, all the OnTerminate procedure has to do is re-enable the button.

There is no space here to print all the code but it can be found on our free, cover-mounted CD-ROM together with a compiled executable that anyone can run.

The finished application enables two TPCWThread objects to run side by side. Even with both running, the program remains responsive: the user can resize the window or move it around the screen. Each thread can be suspended and resumed, or heartlessly terminated before it finishes its task. The user can also control the priority that Windows gives to each thread.

All these are great benefits for certain types of application but this does not make it easy, so for serious multithreaded work, developers will need to look well beyond Delphi's sparse manuals.

PCW Contacts

Tim Anderson welcomes your Visual Programming comments and tips. He can be contacted at the usual *PCW* address, or at **freer@cix.compulink.co.uk** or **http://www.compulink.co.uk/~tim-anderson/**

Visual C++ 4.1 and the **Solutions Development Kit CD** are available as part of a Microsoft subscription. The **Solutions Development Kit** will also be sold separately, price not yet available. Phone **0800 960279**

Books
All books available from **Computer Manuals 0121 706 6000** (prices incl VAT).
The Revolutionary Guide to Delphi 2 (Wrox Press). Book and CD £46.99
Programming Windows 95 (Microsoft Press). Book and CD £46.99