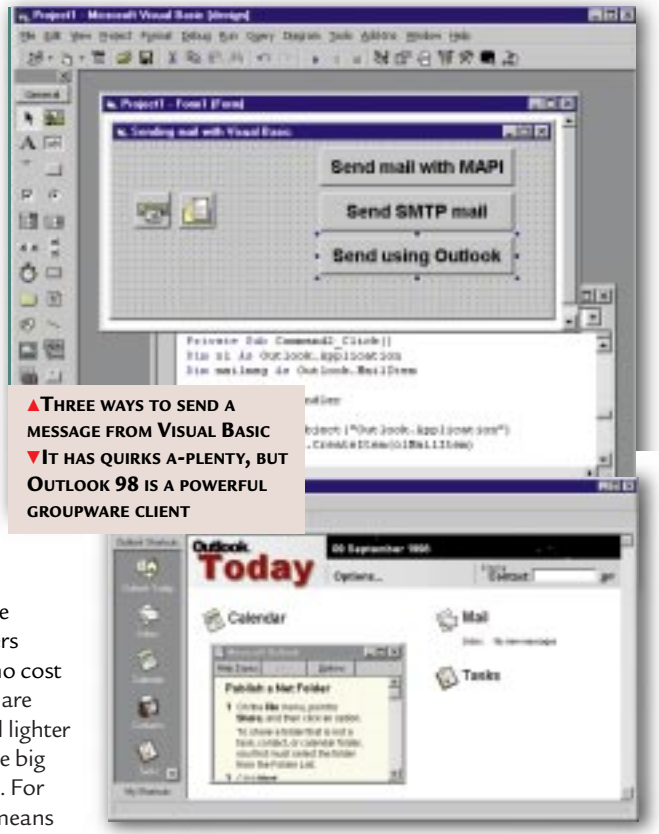
Mail message

Get mail-enabled fast, says Tim Anderson. Also, choosing between VB, Delphi and Access.

When you hear the word groupware, you probably think of monstrous applications like Lotus Notes or Microsoft Exchange, with hefty licence fees and heavy training needed before you can get up and running. These huge applications have their place, but a lot can be done with simple email using standard internet protocols.

Out and about

If you have Outlook 98, which was freely available for a time and is still a no-cost upgrade for Microsoft Office users, then you have in your hands an instant groupware client, able to share calendars, contacts and tasks, as well as standard messaging. Whether or not you use Outlook, most applications can benefit from some mail features. Applications that deal with documents or manage addresses should have a Send command, and business applications might automatically mail reports to appropriate contacts. The one component that does not come bundled with Windows or Office is a mail server, although you can send internet mail though a dial-up service provider. Windows does provide the workgroup post office, but in reality this is little more than a shared directory. Fortunately, there are numerous mail servers available at little or no cost (*see page 298*). These are cheaper, simpler and lighter on resources than the big groupware products. For Outlook users, this means you should use the Internet rather than the Workgroup version. In any case, this is a better choice if you do not need support for Exchange.



▲THREE WAYS TO SEND A MESSAGE FROM VISUAL BASIC
▼IT HAS QUIRKS A-PLENTY, BUT OUTLOOK 98 IS A POWERFUL GROUPWARE CLIENT

[FIG 1]

Minimal code for a MAPI session in Visual Basic

```
Private Sub Command1_Click()

MAPISession1.SignOn
If MAPISession1.SessionID <> 0 Then

With MAPIMessages1
.SessionID = MAPISession1.SessionID
.Compose ' start a new message
.RecipDisplayName = "Me old hearty"
.RecipAddress = →
"richard@myipdomain.com"
.MsgSubject = "Message from VB App"
.MsgNoteText = "We did it"
.Send False ' don't display a dialog
End With

MAPISession1.SignOff
End If
End Sub                    (→ continues on next line)
```

● How to send a message from Visual Basic

Once mail is installed successfully, you can build mail-enabled apps. There are many different approaches, and here are some possibilities.

☛ **First**, you can use MAPI (Mail API). MAPI is a horribly complex interface to the Windows messaging subsystem. The good news is that for most applications you can think of MAPI simply as a standard way of programming your mail client. Most email clients are MAPI-compliant. Visual Basic comes with two MAPI controls, MAPISession and MAPIMessages. Here's how you might send an email message with them:

1 **Place** the two MAPI controls on a form.

2 **Set** MAPISession's UserName and password properties to values that are valid on your system.

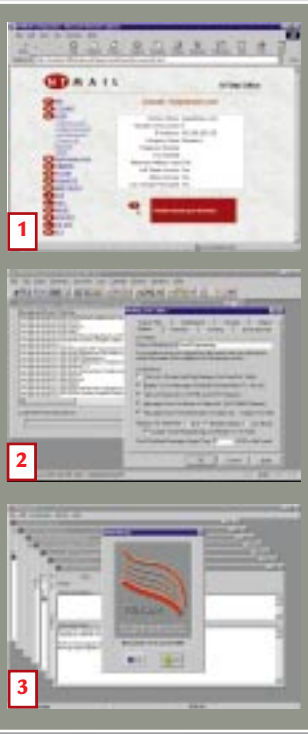
3 **Write** code like that in Fig 1. This first opens a MAPI session, and then makes use of it by calling the methods of MAPIMessages.

☛ **Second**, there are other ways to use MAPI. In the WINAPI directory of a VB installation you will find MAPI32.TXT which contains declarations for Simple MAPI, the essential API mail functions. There are also COM interfaces to MAPI. One is the CDO (Collaboration Data Objects) library which you can find through VB's Project -References dialog. Simple code for CDO looks similar to that using the MAPI controls. CDO used to be called Active Messaging.

FINDING A MAIL SERVER

It makes sense to run a mail server on a network of almost any size. There are numerous excellent products around, most of which let you download working evaluation versions. If you have a dial-up web connection, look for strong support for automatic sending, collection and distribution of mail. On Windows NT a popular choice is **NT Mail** from **Internet Shopper** (1). The new version 4.0 includes a built-in web proxy, so that you can browse the web from anywhere on the network. There is also a list server, anti-spam filter, dial-up support, and flexible configuration options. The interface for administration is entirely browser-based, which is good if you need access anywhere on the network, but performs poorly compared to a conventional Windows interface.

Despite its modest starting price I feel NT Mail is best suited to large networks. Another possibility is **mDaemon** from **Alt-N Technologies** (2). This is a great choice for small networks and runs on Windows 95, 98 or NT. The dial-up support is especially well thought-out, with a multi-POP option that lets you collect mail from several POP3 mailboxes on each connection. The administration interface is over-busy, but flexible and responsive. Remote configuration is also possible. A list server is built-in, but not a proxy server, although the Wingate proxy comes from the same stable. For those on a tight budget, **Mercury/32** (3) is a solid mail server whose author permits unlimited use without fee. Its only real disadvantage is that advanced configuration is by means of editing text files rather than through a graphical interface. With Mercury/32 plus Outlook 98 ,you have all you need for some clever groupware at minimal cost.



[FIG 2]

Minimal code to send SMTP mail from Visual Basic

```
Private Sub Command1_Click()
Dim sText As String

With SMTP1
If .State <> prcDisconnected Then
Exit Sub
.RemoteHost = "192.168.255.126"
.DocInput.headers.Add "From", ➔
      "tima@myipdomain.com"
.DocInput.headers.Add "To", ➔
      "richard@myipdomain.com"
.DocInput.headers.Add "Subject", ➔
      "Straight from VB"
sText = "Look - no MAPI!"
.SendDoc , , sText
End With

End Sub
```

(➔ continues on next line)

Third, you can use standard internet functions. This is a great way to add simple messaging to an application, if you are not using Exchange. One approach is to use the Netmanage ActiveX controls. Of course, you can send mail via SMTP (Simple Mail Transfer Protocol) using MAPI or CDO, but in that case you are programming an external mail client as opposed to connecting directly. Fig 2 shows how to send a message using the Netmanage SMTP control.

Note that this example code is only a start. The secret of success with the Netmanage controls is to monitor the State property along with normal error-checking. The best approach is to use a timer control and/or make use of the StateChanged event in order to avoid calling methods that are invalid in the current state, for example if you are not yet connected. It is not as easy as it should be, because Microsoft has concentrated on promoting Exchange and CDO. Incidentally, the same controls come with Delphi, unless you get hold of a high-end edition of Delphi 4.0 that has the much better NetMasters SMTP control.

Fourth, you can automate Outlook using its automation object model. Fig 3 shows how to send a message using this technique. First, you need to open Project-References and check the Microsoft Outlook 98 object model. The advantage of using Outlook is that you get more than just mail. If you have installed a mail server along with Outlook's Net folders option, you get the ability to share contacts and calendars, for example, through email-based replication.

[FIG 3]

Automating Outlook 98

```
Private Sub Command1_Click()
Dim ol As Outlook.Application
Dim mailmsg As Outlook.MailItem

On Error GoTo handler

Set ol =
CreateObject("Outlook.Application")
Set mailmsg =
ol.CreateItem(olMailItem)

With mailmsg
.Subject = "Outlook at your
fingertips"
.Body = "This is send automatically
from VB"
.To = "Richard"
' name must exist in address book
.Send
End With
Exit Sub
handler:
MsgBox Err.Description
End Sub
```

Questions & answers

Q My main development tool is Access. How would you summarise the differences between a database development tool like Access and other packages like VB and Delphi? I don't use the latter and would like to know what advantages they can offer over Access.

PHILIP LOPORTO

a The short answer is that if you have a good route to creating applications that work, there is no reason to change. Many typical business applications can be done either way. If you are undecided, the main advantage of applications like Access is the large amount of built-in functionality they provide, such as query builders, report writers and mail-merge. These require extra work in Visual Basic or Delphi. On the other hand, if you want to lock-down the application so that users are presented with a simple interface and few options, then these extra features can get in the way and you will be working to disable or hide them. By contrast, if you use a language product you start with a blank sheet. It is more work, but you have full control. Some other advantages are:

- **Language products** are more flexible if you want a highly customised user interface.
- **Royalty-free deployment** comes automatically with VB or Delphi, whereas Access require you to purchase an expensive developer version, or to have the full version installed on each user's machine.
- **You can create** smaller apps, particularly with Delphi or C++.
- **A language product** will scale better if you want to move to a distributed or web-based application.

Q How do you pass variables to and from a form? Some of my applications have options that are password protected. I have a form which displays the appropriate message and password, gets input from a user and returns a flag acording to validation. Currently I am declaring three public variables: strPUBMessage_To_User, strPUBPassword_Expected, and intPUBResult. This seems clumsy. Is there a way of calling a form such as:

```
PasswordCorrect =
frm
GETPASSWORD
("YOU NEED A
PASSWORD","LETMEIN",Result)
```

NIGEL THOMAS

a There is nothing to stop you writing code to show the form but forgetting to set the password variable first. Good programming protects you to some extent from your own mistakes. By the way, VB has a wizard-generated log-in dialog but it's poor. Wrapping the dialog display in a function is a start, but here is what I consider the best technique.

- **First**, use property procedures. That means using Visual Basic 4.0 or higher, where forms are a kind of class module. Make the necessary variables private to the form, by declaring them with the Private keyword. Then add property procedures, using the Tools - Add Procedure option and checking Property. Think about whether a property should be read-only or write-only. If you wanted to make the password property write-only, for example, delete the Property Get procedure. You need to add code linking the property procedure with the private variable.
- **Second**, initialise the variables in the form's Initialise event. For example, you might set m_password and m_username to the empty string.
- **Third**, add validation to the form's Load event. You could check for an empty password and raise an error or unload the form in response.
- **Fourth**, when you use the dialog, do not use the pre-declared form object. Instead, use the New keyword to create a new instance. That way, you can be sure that the Initialise event will fire. The only disadvantage is a longer load-time for a complex form, but this is not an issue for a simple dialog.

[FIG 4]

A possible login routine in Visual Basic

```
Function frmGETPASSWORD(sMessage
As String, sUserName as string,
sPassword As String) As Boolean
Dim dlg As frmLogin ' or the name →
of your login form
On Error GoTo handler
Set dlg = New frmLogin
dlg.Message = sMessage
dlg.username = sUserName
dlg.Password = sPassword
dlg.Show vbModal
 If dlg.bLoginSucceeded Then
 frmGETPASSWORD = True
 Else
 frmGETPASSWORD = False
 End If
Unload dlg
Set dlg = Nothing
Exit Function
handler:
MsgBox Err.Description
Resume Next
End Function
```

(→ continues on next line)

PCW CONTACTS

Tim Anderson *welcomes your questions and tips. He can be contacted c/o the PCW editorial office (address, p10) or at* **visual@pcw.co.uk.**

To explore what mail servers are available, a good place to start is the Tucows site, for example **http://tucows.cix.co.uk.**

For NTMail contact Internet Shopper 01275 340333 or visit **www.ntmail.co.uk.**

Prices start from £49.00 (£57.58 inc VAT) for five mail accounts.

For mDaemon contact Grey Matter 01364 654100 or visit **www.mDaemon.com.**

Priced £71.00 (£83.43 inc VAT) for five mail accounts.

Mercury/32 is at **www.pegasus.usa.com.** *It is free to use but detailed manuals must be purchased. Information on programming Outlook can be found in Building Applications with Outlook 98(ISBN 1-57231-718-3), Microsoft Press, £37.49 with CD. Contact Computer Manuals 0121 706 6000.*