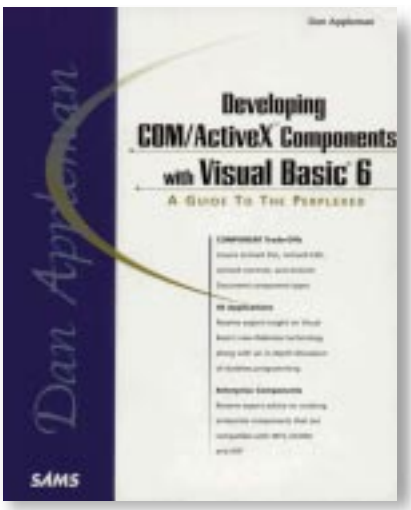
Do drop in

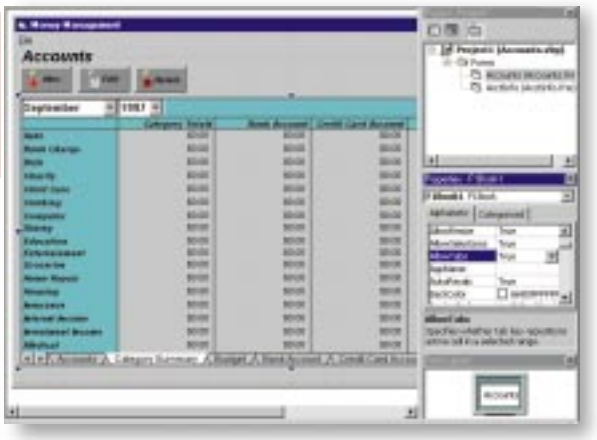
Tim Anderson has a light solution to embedding a spreadsheet.

Microsoft's Component Object Model (COM) is the underlying technology behind OLE, ActiveX, Automation, Data Access Objects and more besides. If you use Windows 95, 98 or Windows NT, you use COM. If you code in Visual Basic, you are a COM developer. There is no choice, it is deeply embedded into Windows and most of the time it works well.

There are a couple of interesting issues, though, the outcome of which will determine COM's long-term success. One is that COM development is hard once you go beyond merely using components or servers and start creating them. Visual Basic is meant to make it easy but only partly succeeds. The other issue is how well COM can make the transition from a handy way of working in Windows to being the foundation of large-scale applications which use distributed objects.

Every serious Visual Basic developer knows Dan Appleman's superb book on using the Windows API. His latest book is about this next challenge to VB programmers and is called *Developing COM/ActiveX Components with Visual Basic 6*. Topics include an explanation of COM interfaces, automation and binding, events, object lifetime, multithreading, building ActiveX controls, using collections and Microsoft Transaction Server (MTS) — a way of managing COM servers. There is even occasional humour. It is a worthwhile purchase but not as satisfying as his API volume. Web applications and MTS need more space and database issues are not covered.





▸Fig 1 For creating an application built around a spreadsheet, Formula One is hard to beat

It is an accessible book, though, and a useful supplement to the VB manuals. It also confirms the problem with VB and COM: for every ease-of-use feature, there is another one to trip you up. One example is the problem of controlling object lifetime in VB. It is all too easy to create objects that never terminate. If you want to create robust ActiveX controls or COM servers in VB, you have to learn a lot about COM itself and about how VB implements it. The benefits outweigh the cost, but do not expect an easy ride.

Drop-in spreadsheets

Embedding a spreadsheet into an application could be a smart move if you want to import or export spreadsheet data, say, or to present figures in a familiar and flexible format. One option is to use Excel as a control but if that heavyweight solution does not appeal you could try a spreadsheet control such as Tidestone's Formula One 6.0 or Farpoint's Spread 3.0.

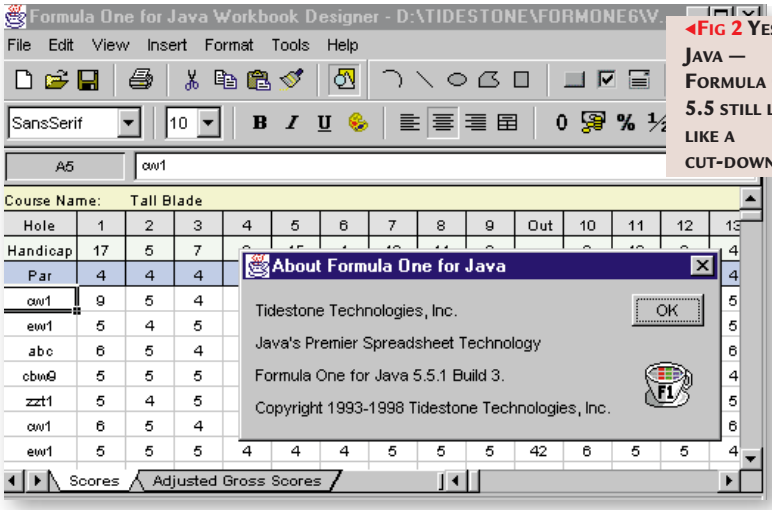
Although Formula One is well-known, Tidestone is an unfamiliar name. Formula One was first published by Visual Components, which was then acquired by Sybase. The company is now once more independent and has renamed itself Tidestone Technologies.

Formula One Professional is a bundle containing Formula One 6.0 for Windows, First Impression 6.0 for Windows, and Formula One 5.5 for Java. Two other products have been dropped: the dbComplete control could not compete against the data grids bundled with VB 6.0 and Visual Writer is now available from its original owner, DBS Software.

Formula One is an Excel-compatible spreadsheet control [Fig 1], and First Impression handles charts and graphs. Both are supplied as 32-bit ActiveX controls only. Drop a Formula One control on a VB form, right-click, and you can run the WorkBook Designer which is essentially a complete spreadsheet application that enables you to modify the embedded spreadsheet in the same way as you would normally work in Excel or Lotus 1-2-3. Formula One supports around 130 popular worksheet functions, a wide range of cell formats, drawing objects, buttons, checkboxes, drop-down lists and charts via First Impression. You can populate a spreadsheet from an ODBC database query and print preview is supported.

New features in version 6.0 include smarter recalculation, larger worksheet size (to 65,536 rows), cell merging and enhanced in-cell editing. Formula One is Excel-compatible and you can load and save spreadsheets in Excel 95 and 97 format. In my experience this works well. Formula One has the same tabbed worksheet style as Excel so multi-sheet workbooks import smoothly.

Bundled with Formula One 6.0 is a Java version [Fig 2] which offers a Formula One JavaBean. Although a little behind the ActiveX in features and



◂FIG 2 YES IT'S JAVA — FORMULA ONE 5.5 STILL LOOKS LIKE A CUT-DOWN EXCEL

compatible with Excel only up to Excel 7.0, this remains an impressive component. Much the same range of worksheet functions is supported and the user interface is almost as slick.

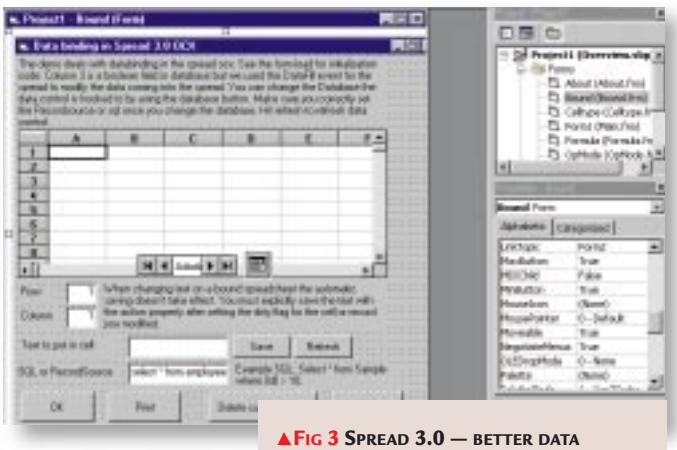
The fly in the ointment is that with effect from version 6.0, Tidestone no longer allows royalty-free deployment although version 5.0 is still available for this purpose. Otherwise, you have to obtain runtime licenses.

FarPoint Spread 3.0

No such limitation applies to FarPoint's Spread 3.0, another rather capable spreadsheet control. Although it may at first seem similar to Formula One, there are many differences. Spread 3.0 is supplied in a number of different guises. There is no Java version but there is a 16-bit VBX and DLL, a 32-bit DLL, and two ActiveX controls; one for Data Access Objects and the other for ActiveX Data Objects.

It also offers Excel compatibility, although only for the 97 version. However, when I tried to load my accounting spreadsheet it crashed both the Spread Designer and the entire Visual Basic IDE, suggesting that Formula One (which imported the same file perfectly) is the better choice in this respect. Another problem is that Spread 3.0 has no concept of workbooks and worksheets. If you import an Excel workbook, you have to specify which sheet you want.

With around only 70 worksheet functions, Spread 3.0 is a little behind Formula One, though you will probably still find the ones you need. Spread also lacks drawing objects, charts and graphs, although you can embed a picture in a cell. Nevertheless, Spread 3.0 [Fig 3] is a data-aware control. Whereas Formula One can only display the results of a query, Spread 3.0 can offer a dynamic, updatable view of your data. It also has a strong range of possibilities for cell formatting, including an owner-draw option which lifts all restrictions on what you can display in a cell. It is easy to put combo boxes, checkboxes, buttons or a popup calendar in a cell for easy data entry and a separate print preview control is supplied.



▲FIG 3 SPREAD 3.0 — BETTER DATA BINDING BUT NOT SO GOOD ON EXCEL FILES

The slicker of the two controls is Formula One as it is better for Excel support, and has an interesting migration path to Java. Spread 3.0 scores if you require data binding, royalty-free distribution, 16-bit or DLL versions. But both are products which add a huge range of features to your application at the drop of a control.

A different Outlook

I have received a pile of emails in response to the Outlook programming feature in the February column but reader Richard Deeming spotted an oversight: 'You stated that there is no accessible unique identifier for contact items in Outlook,' he writes. 'I use the EntryID property which exists for all Outlook items as the unique ID. This is a string value which is unique across all Outlook folders. It is suitable for use as a key in collections, ListView objects, TreeView objects and so on.

'In addition, the NameSpace object provides the GetItemFromID method, which takes the EntryID of the item and the StoreID of the folder it is stored in, and returns the item (if it exists). Finally, a more efficient method of testing for an invalid variable reference is to use the comparison VariableName Is Nothing, instead of VariableName.TypeName = "Nothing".'

This is excellent, although the EntryID is not quite as good as a typical primary key in a database. The reason is that it can change, for example if the item is moved to another folder. But for quick search and retrieval it is ideal. The EntryID and StoreID also provide a handy way to link to the CDO (Collaboration Data Objects) library using the GetFolder method of a CDO Session object.

Yes, you can also program Outlook through CDO or MAPI (Messaging API), giving extra features. *(For more on Outlook, see reader Ian Gordon's letter in the Question and Answer box, overleaf).*

Correction: *in February's column (page 294) three lines were omitted from Fig 1, 'Changing the MessageClass'. After the first three Set statements, you should insert this extra code:*

```
iMax = cts.Count
For iRecNo = 1 to iMax
set ct = cts(iRecNo)
```

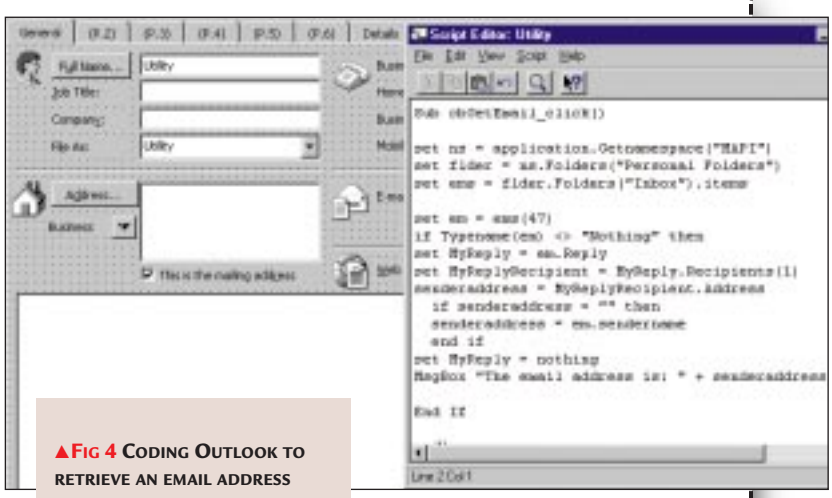
Questions & answers

Q I am working with Delphi 3 and I wish to query against date fields in Paradox tables. When I enter the SQL I have to use a date format of mm/dd/yy and not dd/mm/yy. All my dates are displayed correctly and I assume that is due to the regional settings in Windows (I'm using '98 and NT4). I have had a root around in the BDE Administrator and checked the Date settings on the Configuration tab.

KEVIN PARSONS

A *If you set the BDE's date format to dd/mm/yy, which corresponds to a MODE of 1 in the Date setting in the BDE Administrator, you can use that format in some contexts, such as in the Query By Example grid in the database desktop or with the SetRange method of a TTable but not in Local SQL which is the version used by the BDE when querying Paradox tables. Local SQL requires either mm/dd/yy or mm/dd/yyyy. It is annoying, although at least you know for certain what format you have to target.*

Applications which fail to check data formats and issue wrong dates if the user has an unexpected Control Panel setting are a liability. The worst aspect is that the first 12 days of the month produce a valid but different date depending on which format is used. You can make an application bullet-proof but it is easy to slip up.



▲FIG 4 CODING OUTLOOK TO RETRIEVE AN EMAIL ADDRESS

On our cover disc this month, there is Delphi 2 Developer from Borland. See Cover Disc Notes, starting on page18) for more information

Q Using Visual Basic 5, I want to be able to pass command line parameters to my application. The help files and Books online hasn't helped.

'DAVE'

A *Funnily enough, the Command function returns exactly what Dave wants; the command line parameters. The real issue here is why the information is hard to find. It is probably because Microsoft calls them arguments, not parameters, so searching for parameters gets you nowhere. If you are sure the answer is in there somewhere, try some lateral thinking!*

Q How can I access the sender's email address from a message? The name is easy but the address is impossible!

IAN GORDON

A *This question relates to my February column which dealt with Outlook programming and it is only nearly impossible. See Fig 4. Bizarre though it may seem, the Outlook object model does not include the email address of the sender of a message. The best way I can find using Outlook alone is shown in Fig 5. This involves pretending to reply and then inspecting the recipient of the reply. The problem is that the SenderName property holds the friendly name for the sender, not the email address — unless no friendly name is available. A better approach is to create a CDO Session object, find the message and inspect the Sender.Address property. This technique is illustrated on the web at www.slipstick.com/exchange/olforms/bulkreply.htm along with many other Outlook tips.*

• For more comment on Outlook, see also 'A Different Outlook' in the main text.

[FIG 5] Finding the email address of a message

```
Set ns = Application.GetNamespace("MAPI")
Set flder = ns.Folders("Personal Folders")
Set ems = flder.Folders("Inbox").Items
Set em = ems(1) ' or use Find method to get a specific message

If Not (em Is Nothing) Then
Set MyReply = em.Reply
Set MyReplyRecipient = MyReply.Recipients(1)
senderaddress = MyReplyRecipient.Address
  If senderaddress = "" Then
  senderaddress = em.SenderName
  End If
Set MyReply = Nothing
MsgBox "The email address is: " + senderaddress
End If
```

PCW CONTACTS

Tim Anderson can be contacted via the PCW editorial office (address, p14) or email visual@pcw.co.uk

Developing COM/ActiveX (Dan Appleman) £46.95 incl book and CD (£39.96 ex VAT), ISBN 1-56276-576-0 from Computer Manuals 0121 706 6000 www.computer-manuals.com

Formula One Professional £69 (£81.08 inc VAT) from Tidestone 01892 834343 www.tidestone.com. Runtime licenses from £57.50 (£67.56 inc VAT) per user

Spread 3.0 is £249 (£292.58 inc VAT), royalty-free deployment, from Contemporary Software 01344 873434 www.contemporary.co.uk