# A break from the old routine

**Tim Anderson** **makes a splash with Visual Basic, and studies a slimline alternative to the Microsoft or Borland database engines.**

One of the keys to developing efficient, robust software, especially if you want to do so quickly, is to re-use code. Ways to do this include creating Delphi components, C++ classes, or using VBX or OCX controls in Visual Basic.

Dynamic Link Libraries (DLLs) are the foundation of Windows, and a great way to create functions that you can call from any programming language. You cannot create old-style DLLs with VB, but version 4.0 introduced OLE DLLs, allowing VB code to be called from other applications via OLE automation.

These are good ways to re-use code, but there is still a place for the oldest and crudest technique, which is cutting and pasting routines from one application to another. Programmers are lazy and will happily ransack old but working code to save time and avoid errors.

For example, a common requirement in VB database applications is to export a query as a .DBF table, the most universal format for mail merge, or transfer to other applications. JET's SQL supports a SELECT ... INTO clause that creates a new table from a query. If the database in question is an Access MDB, this only creates a new table in .MDB format. To get round this, I use this technique:

1. Copy the query to a temporary table.
2. Copy the structure of the table to a new .DBF.
3. Copy the records in the temporary table to the .DBF.

This works well, and I have no intention of rewriting the code, which gets popped into applications as required. Only the second step takes more than a single SQL command, so this is wrapped in a re-usable function declared like this:

```
Sub CopyStructureToDBF(MDBName As
String, TableName As String, DBFPath
As String, DBFName As String)
```

It is vital that no paths or field names are hard-coded into the routine as this would wreck its re-usability.

## The DIY solution

Once you have built up a library of routines, the next question is where to store them. Simplest is to have a directory full of .BAS files, but this is awkward to manage. It can also lead to the inefficient and unsafe strategy of including many unused routines in your project, for the sake of one or two that happen to be in the same module. A better solution is to write your own database application, storing each procedure in its own memo field.

Alternatively, there are utilities that aim to make it easier to manage your code library. One is Sheridan's VB Assist, now at version 4.0a. VB Assist loads as an add-in, and includes Code Assistant. Code Assistant has two main elements. One is a visual clipboard, a text window to which clipboard output can be redirected. The other is a code database, called Code Librarian, which is actually a VB front-end to an MDB. You can create groups within which to store your routines, and add keywords for easy search and retrieval.

Code Librarian is a good idea, but it's not as well implemented as it should be. The way the database is structured suggests an outline tree for navigation, rather than the drop-down combos actually used. It is silly that keywords can be no more then ten characters long. You can edit code within the Assistant or Librarian, but it's not a good environment for coding, with no syntax highlighting or search-and-replace facility. But it's better than nothing.

## CodeBank

Unlike Code Assistant, which is part of VB Assist, CodeBank is a separate product from Visual Components, best known for the Visual Developers Suite. Again it is a VB database application, but is a standalone program rather than an add-in. The idea is that all your re-usable routines are stored in the CodeBank database. When you want to make use of them, you ask CodeBank to create a new, empty .BAS module, and add the required routines. Finally, the generated module is added to your Visual Basic project in the normal way.

CodeBank has a tidy, effective interface. Procedures are shown in a tree, which can be sorted by category, author, name or type (procedure or function). Each routine can have substantial information stored with it, including short and long descriptions, an example of use, maintenance history, and links to any declarations or other routines that are required. Codebank is intelligent about these links: if a particular procedure makes use of a user-defined Type, the generated Basic module will include the declaration as well as the procedure itself.

The bonus is that CodeBank includes a library of 160 routines, with the emphasis on economy and performance. Many of them use VB code to emulate what might normally be done with a VBX or OCX control: for example, the outline used by CodeBank is drawn entirely using VB code. Another example is a procedure which shows text next to a control by printing directly to the controls' container, avoiding the need for a conventional label control. These routines are impressive, letting you create sophisticated graphic effects without the performance and size penalty of adding lots of components. Anyone interested in efficient VB coding will enjoy them.

## CodeBase 6.0

If the idea of distributing applications on a single floppy disk appeals to you, you will like CodeBase. Very small executables can be built in C, while VB or Delphi applications require a 500Kb runtime DLL, much smaller than either JET or the Borland Database Engine. Well-established in the xBase community, CodeBase is a C library for handling database tables in .DBF format, which are either FoxPro, Clipper or dBase IV compatible. Sequiter now provides versions for C++, Visual Basic and Delphi. The new version bundles the lot onto a single CD-ROM, which is convenient if you use more than one of these languages.

Other significant changes in version 6.0 are limited 32-bit support, the addition of client-server support via a new CodeBase database service application, and new transaction processing functions that will be useful in both standalone and client-server applications.

There are rough edges in this product. Although a 32-bit DLL is supplied for Visual Basic, the data-aware CodeControls are VBX only. An error in one of the main VB examples prevents it from running. Delphi support is currently only 16-bit, although a 32-bit DLL is available on request. Pascal documentation is more complete than in previous releases, but examples are in the form of short routines rather than a full demonstration application.

No effort has been made to create Delphi units or components to simplify use of CodeBase, which is a missed opportunity, bearing in mind the large number of migrants from Clipper, dBase and FoxPro now using Delphi. Successware, with its xBase product called Apollo, has done more to appeal to the Delphi community.

It is worth persevering, for the sake of fast performance on modest hardware, as long as you are willing to get your hands dirty with mysterious functions like "relate4createslave" and "code4initundo." While it is fine for both single and multi-user databases, it is harder to see the benefits for client-server work, unless you have an existing CodeBase system to upgrade. It is competing with many other advanced SQL-based systems, as well as another Sequiter product, the ODBC-compatible CodeSQL.

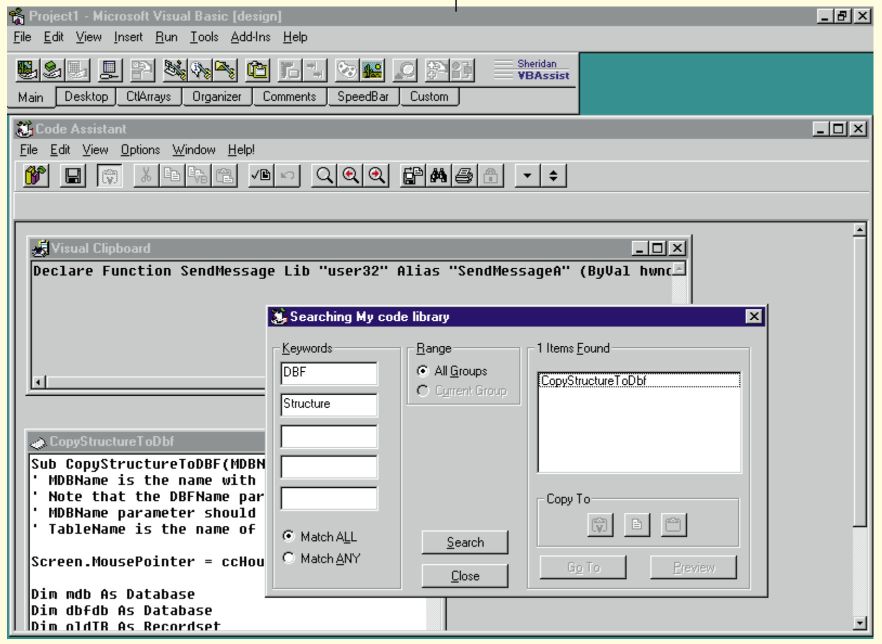
## Code Complete makes a splash

Seasoned VB developers will know the story. A bemused user calls and says, "I tried to run your application. A message came up saying, 'Wrong version of SOMESTUFF.VBX', and then it quit." Windows is highly vulnerable to this kind of problem, and increasing use of OLE, which has its own myriad support libraries, will only make things worse.

Microhelp has a solution in the form of the Splash Wizard. From its name, you would think this is just a way of creating

➡

*Part of Sheridan's VB Assist, Code Assistant lets you create libraries of code, and copy routines either direct to your application or to an intermediate clipboard*


Project1 - Microsoft Visual Basic [design]
File Edit View Insert Run Tools Add-Ins Help
Sheridan VBAssist
Main Desktop CtlArrays Organizer Comments SpeedBar Custom
Code Assistant
File Edit View Options Window Help!
Visual Clipboard
Searching My code library
Keywords
Range
All Groups
Current Group
1 Items Found
Match ALL
Match ANY
Search
Close
Copy To
Go To
Preview
CopyStructureToDbf

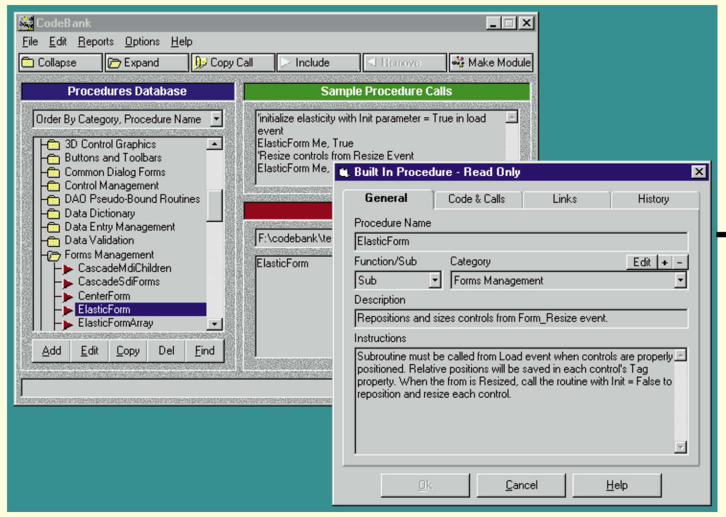
Keywords: DBF, Structure
Range: (●) All Groups ( ) Current Group
(●) Match ALL ( ) Match ANY
1 Items Found: CopyStructureToDbf

```
Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hwnc
```

```
Sub CopyStructureToDBF(MDBN
' MDBName is the name with
' Note that the DBFName par
' MDBName parameter should
' TableName is the name of

Screen.MousePointer = ccHou

Dim mdb As Database
Dim dbfdb As Database
Dim oldTB As Recordset
```



Order By Category, Procedure Name

```
'initialize elasticity with Init parameter = True in load
event
ElasticForm Me, True
'Resize controls from Resize Event
ElasticForm Me,
```

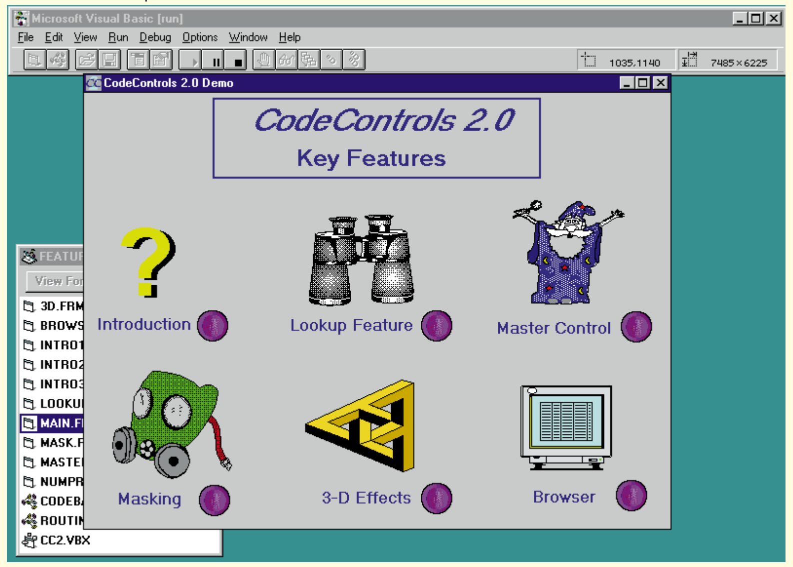
F:\codebank\te

ElasticForm

Procedure Name: ElasticForm
Function/Sub: Sub
Category: Forms Management
Description: Repositions and sizes controls from Form_Resize event.
Instructions: Subroutine must be called from Load event when controls are properly positioned. Relative positions will be saved in each control's Tag property. When the from is Resized, call the routine with Init = False to reposition and resize each control.

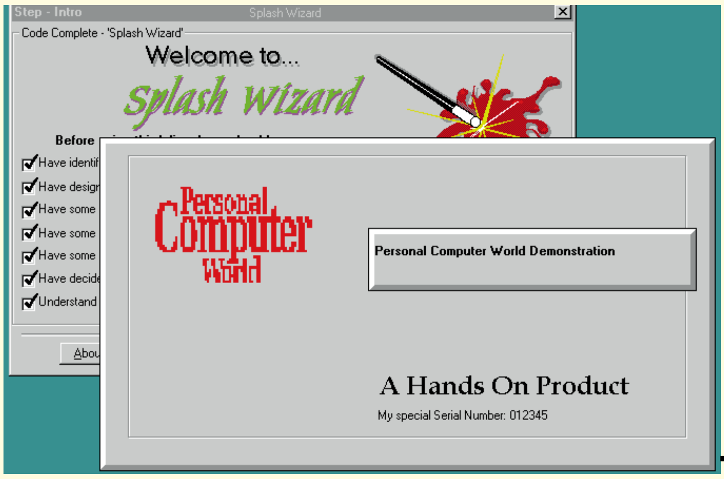
*Codebank comes with a generous library of routines for slimline VB programming. No, there is not a tab control on this dialogue — it's all done with Basic*

*CodeBase can be integrated with your preferred visual tools, but not without some nitty-gritty coding*



fancy welcome screens, but this is secondary. The Splash Wizard creates a new executable which does comprehensive version-checking before launching your application. That way, problems can be identified before your application tries to load. Another possibility is to check for a valid user name and serial number. You can configure things so that your application can only run after the splash executable gives the OK.

Splash Wizard is a good idea, but I was not convinced by its implementation. It is fiddly to use, particularly since the wizard only operates from scratch. If you want to amend an existing splash executable, you have to tweak its resource file by hand, or by using a resource editor. Finally, in a simple test run, I tried out the Splash Wizard by deliberately deleting a .DLL needed by a VB application. The Wizard took a finger-tapping fifteen seconds to report the problem; the VB application on its own took two or three seconds.



*Splash Wizard is an expert version checker, but can be slow*

Code Complete comes with three other components. The Assistants automate the creation of common dialogues, message boxes, and allocating help IDs to VB controls, this last one being the most useful. Code Analyst will analyse and cross-reference Visual Basic projects, commenting on unused code and identifying deviations from standards you specify. For example, you can check that all modules include Option Explicit, or that error handling is enabled in all procedures.

If you have problems, the fourth component, AutoCoder, may help you out. A template-based system, it can automatically edit your code by adding error handlers for example. Another useful trick is to add temporary timing functions so that you can profile the application, discovering which routines are slowing down your software and need tweaking.

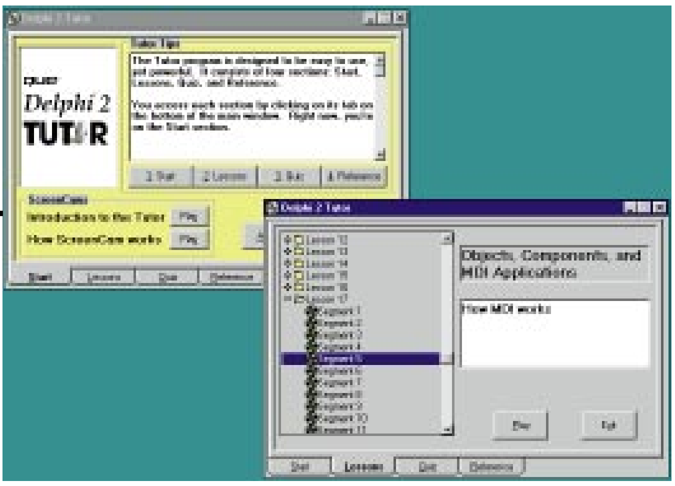
## And finally...

Keep honing those Visual Basic skills. Microsoft is licensing the next version of the VBA engine to third-parties, so expect to see it in new versions of applications including Photoshop, AutoCAD and Visio. ▣

## Visual Programming: read all about it

**Delphi 2 Tutor, by Mike McKelvy**
Ironically, the software which runs this Delphi tutor is written in Visual Basic 4.0, assisted by Lotus ScreenCam. It is the opposite to *Delphi Unleashed*. Introductory and shallow, the excuse is that it is for complete beginners. The special feature is that each lesson has several screen demonstrations with explanatory voiceovers; seeing something done is certainly a help, but in this case it is not well implemented. The interface for the tutorial application is poor, a shame in a teaching tool, and the reference section is skimpy and inadequate. While Mike McKelvy's accompanying book has clear explanations of basic programming concepts, there is not enough information here to build real applications of any substance. A better approach would be to take the reader step-by-step through creating an example project. Video demonstrations are counter-productive unless they encourage hands-on experience as well.

**Delphi 2 Unleashed, by Charles Calvert**
The first edition of *Delphi Unleashed* established itself as one of the best titles for serious Delphi developers. The author works for Borland and is well placed to uncover Delphi's inner workings. This is no cosmetic rewrite: the new edition has over 1,400 pages, and more than half of this bulky volume is completely new. For example, you get 50 pages on multithreading, 250 pages on databases, 150 pages on OLE, and 200 pages on multimedia development. It is an enormously useful resource, clearly written, with sound explanations of both Object Pascal and the Windows API. The sheer amount of material makes it an intimidating volume, both physically and otherwise. Some will be glad to know how to create windows without using Delphi's Visual Component Library; others will wonder why we need to be told. Overall, not for the faint-hearted or beginners, but still a great companion to Delphi's inadequate manuals and online help.



*Delphi 2 Tutor includes plenty of video sequences, but neither the presentation nor the content is inspiring*

## PCW Details

**Contemporary Software**
07000 422224
(VB Assist 4.0a, £135; Code Complete, £175) **Visual Components**
01892 834343 (Codebank, £99)
**Highlander Software** 0181 316 5001
(CodeBase 6.0, £225)

**Books**
Books from **Computer Manuals**
0121 706 6000
**Delphi 2 Unleashed** (Sams). Book and CD, £54.95 (inc VAT)
**Delphi 2 Tutor** (Que). Book and CD £46.99 (inc VAT)