Ooh, you **Saxy thang!**

Build point-and-click access into your applications with Sax Webster, says Tim Anderson. There's widgets and Windows, too.

FORGET LAPTOPS AND MOBILE phones; the fashion accessory of the moment must be the personal Web site. Web sites are of no use unless they are visited, so why not build point-and-click access into the applications you distribute?

You can do this by calling an external application like Netscape or Internet Explorer, but Sax Software lets you go one better by building a customised browser right into the application.

The Webster control is a 32-bit browser OCX that drops directly into any compatible development tool, such as Visual Basic 4.0 or Visual C++ 4.0. With the rampant growth of the Internet and increasing corporate usage of Intranet networks, Sax Webster has turned up at just the right moment.

For example, online help might now mean dynamic information on a Web site, rather than the static file shipped with an application. Another option is to direct the hapless user to a site offering further products and services. HTML pages can be loaded from disk as well as from the Internet, so you could also use Webster as a multimedia browser.

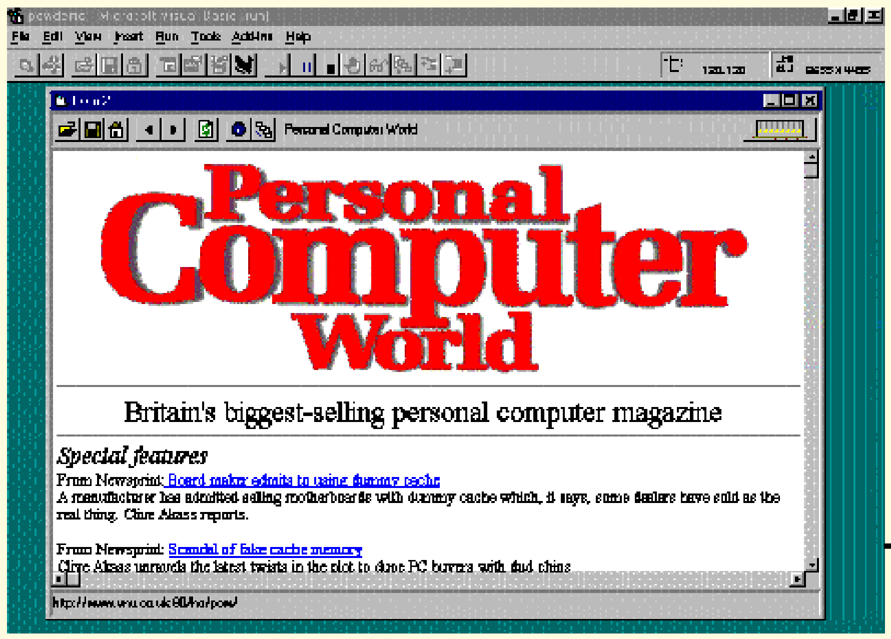
Sax Webster is a complete application wrapped in a control. You can create a browser simply by dropping the Webster control onto a form in VB or Delphi. It claims to support HTML version 3.0, but Sax notes that "because 3.0 is not yet defined as a standard, it may differ from what Netscape, or some other 3.0 browser, supports."

Here is the problem with Webster, and ultimately with the Web itself: lack of tightly defined standards resulting in compatibility problems. It may not matter too much, since it would be foolish to use a Webster application as a replacement for Netscape or Internet Explorer. Webster makes better sense as a tool for accessing specific Web sites that are linked to the container application, so you can ensure the compatibility of those particular pages.

Some problems can also be overcome by writing code to intercept Webster events: for example, Webster does not support the mailto command that HTML uses to initiate an email message. The VB 4.0 code in *Fig 1* will intercept mailto and call whatever application is associated with that command in the Windows 95 registry.

Fig 1 Intercepting mailto

```
Private Sub Webster1_DoClickURL(SelectedURL As String, Cancel As Boolean)
If Left$(LCase$(SelectedURL), 7) = "mailto:" Then
' run MS Exchange, using file association
ShellExecute 0, "open", SelectedURL, "", "", 0
SelectedURL = ""
' stop Webster attempting to act on this command
Cancel = True
End If
End Sub
```


pcwdemo - Microsoft Visual Basic [run]
File Edit View Insert Run Tools Add-Ins Help
Personal Computer World
Personal Computer World

Britain's biggest-selling personal computer magazine

Special features

From Newsprint: Board maker admits to using dummy cache

A manufacturer has admitted selling motherboards with dummy cache which, it says, some dealers have sold as the real thing. Clive Akass reports.

From Newsprint: Scandal of fake cache memory

Clive Akass unravels the latest twists in the plot to clone PC boxes with dud chips

http://www.vnu.co.uk:80/vnu/pcw/

All done with Webster: VB 4.0 visits the PCW home page

Another useful feature is the GetContent method, which lets you read all or part of an HTML page into a variable.

Initially only available as a 32-bit OCX, Sax has now released a 16-bit OCX as well, but nothing yet for VB 3.0 or Delphi 1.0 diehards.

New-look Data Widgets

Sheridan's Data Widgets has long been one of the most popular Visual Basic add-ons, particularly since the VB 3.0 grid is so poor. The data-bound controls in VB 4.0 are better but still leave room for third-party enhancements. Version 2.0 brings the expected conversion to 16- and 32-bit OCX format, but with enhancements. Sheridan has taken the opportunity to restructure the Data Widgets using objects and collections, bringing it into line with other programmable OLE objects. This makes for more logical code and increases the programmer's control, the disadvantage being that code which worked with Data Widgets 1.0 will have to be extensively rewritten (*Fig 2*).

The actual Data Widgets controls are the same six as before: Data Grid, Data Combo, Data Dropdown, Data OptionSet, Data Command and the Enhanced Data Control. All are useful, but the Data Grid is the reason people buy this package. Its neatest trick is to link with a Data DropDown so that users can click on a grid cell and select values from a dropdown list bound to a field in another table.

Do you need Data Widgets? It depends on how you prefer to program. If you make extensive use of bound controls this bundle is all but indispensable, particularly if a data grid is a key part of the user interface. The data control in VB 4.0 is not compromised in the same way as VB 3.0's effort, so this is a perfectly sound approach. The cautionary note is that large OCX controls like these cause substantially slower loading of your VB application, and grids are often not the best way to present data to the user. The Data Grid also works well as an unbound virtual list control — a further enticement which may sway doubters.

Fig 2 On the button

To put a button in a DataGrid cell in version 1.0 of Data Widgets, use a ColBtn property:

```
SSDbGrid1.ColBtn(2) = True
```

which in Version 2.0 becomes

```
SSDbGrid1.Columns(2).Style = 1 ' edit button
```

Tools of the trade

Microhelp's OLE tools may have up-to-date OCX technology, yet the package conveys a dated impression: the main reason being that apart from their OCX conversion, many of the controls are little changed from earlier versions, right down to their description in the manual and the clunky example applications. OLE tools also slipped up during review when one of the genuinely new items, MhSubClass, failed to deliver. This is a message-trapping control that can catch Windows API messages and either kill them, or respond with a custom event and pass them on.

MhSubClass is fine for some purposes; for example, if you want to inspect WM_MENUSELECT messages in order to provide a help text as the mouse runs down a menu. But a common requirement is to trap a message and then write code to determine whether to kill it or pass it on. MhSubClass cannot do this, since the fate of the message has to be determined before the VB event is triggered. Rivals such as the MessageBlaster OCX have no such handicap.

Never mind the quality — with 54 separate controls, the bundle still rates as good value. MhCalendar is a data-aware

Top Tips: slim, dim and begin

● Speed VB's load time and slim your applications by stripping down AUTOLOAD.MAK (VB3) or AUTO32LD.VBP (VB4) to include only controls and references essential to every project.

● Avoid "Dim iA, iB as Integer". This code declares iA as a variant. Instead, use "Dim 1A as Integer, IB as Integer".

● In VB4, disable Compile on Demand (in Tools - Options - Advanced) to have the compiler check for syntax errors before a project runs.

● Your Delphi application can easily check for command-line parameters. ParamCount returns the number of parameters; ParamStr(0) returns the path and filename of the application; and ParamStr(n) returns the nth parameter up to ParamCount.
For example:

```
procedure TForm1.Button1Click(Sender: TObject);
var
i: integer;

begin
for i := 0 to ParamCount do
  MessageDlg(ParamStr(i), mtInformation,
   [mbOk], 0);
end;
```

● If you are adding lines to a string control like a listbox or memo, or an outline component, use BeginUpdate to increase performance by preventing screen updates. For example,

```
procedure TForm1.Button2Click(Sender: TObject);

begin
listbox1.items.beginupdate;
listbox1.items.add('One item');
listbox1.items.add('another item');
listbox1.items.endupdate;
end;
```

Fig 3 A little application to toggle with

Here are two functions to detect the status of the screensaver and to check its state. Note that to work in Windows 3.1, the declarations will need to be adapted.

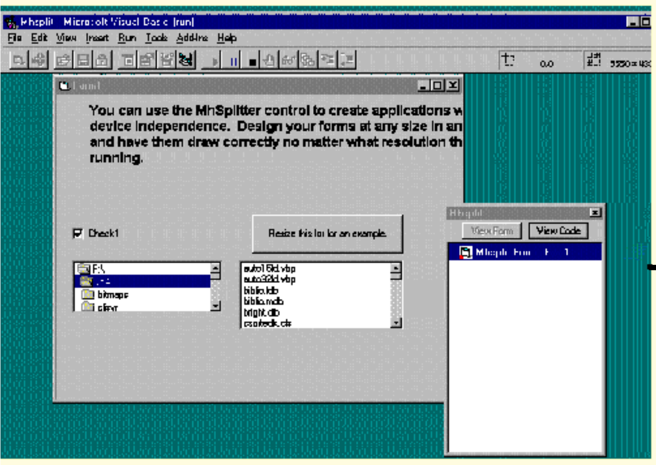
```
Option Explicit
Declare Function SystemParametersInfo Lib “user32” Alias ➞
“SystemParametersInfoA” (ByVal uAction As Long, ➞
ByVal uParam As Long, lpvParam As Long, ByVal fuWinIni As Long) As Long
Public Const SPI_GETSCREENSAVEACTIVE = 16
Public Const SPI_SETSCREENSAVEACTIVE = 17

Function isActive() As Boolean
Dim lRetVal As Long
Dim pvParam As Long

lRetVal = SystemParametersInfo (SPI_GETSCREENSAVEACTIVE, 0, pvParam, 0)
If lRetVal = False Then
MsgBox “Call to SystemParametersInfo failed”
isActive = False
Exit Function
End If
If pvParam = False Then
isActive = False
Else
isActive = True
End If
End Function

Sub SetActive(bActive As Boolean)
Dim lRetVal As Long
Dim pvParam As Long
lRetVal = SystemParametersInfo ➞
(SPI_SETSCREENSAVEACTIVE, bActive, ByVal pvParam, 0)
If lRetVal = False Then
MsgBox “Call to SystemParametersInfo failed”
End If
End Sub
```

(Note: ➞ this symbol has been used where the code shown on the following line is a continuation, and should be entered as such. You’ll find the complete code on this month’s cover-mounted CD-ROM, together with versions for VB3 and Delphi).


You can use the MhSplitter control to create applications w device independence. Design your forms at any size in an and have them draw correctly no matter what resolution th running.
Mhsplit Microsoft Visual Basic [run]
File Edit View Insert Run Tools Add-Ins Help
Check1
Resize this box for an example.
F:\
bitmaps
auto16ld.vbp
auto32ld.vbp
biblio.ldb
biblio.mdb
bright.dib
View Form
View Code

The MhSplitter control from OLE Tools attempting resolution independence. Unfortunately, this text box does not always get correctly resized

calendar control. MhSplitter allows you to build resolution-independence into interfaces by automatically resizing controls within the container, albeit rather slowly. MhRealInput is a text box that improves on VB’s masked edit control for working with real or currency values. And so it goes on, providing something of value for most VB projects.

Microhelp supplies two versions of these tools: OLE tools has 16- and 32-bit OCXs, while VB tools stays with the old VBX format. There are differences between the two. For example, the inadequate MhSubClass is OCX-only, while the clever MhOutOfBounds universal data binding control is VBX-only.

Finally, VB tools used to come with a version of Farpoint’s Grid control, but that has now been dropped.

Screensaver settings: hacking into the Win95 API

“I’ve bought a new system with Windows 95 and VB 4.0. My computer has a WIn/TV card, and I wanted to write a program that would turn the screensaver off and on without having to go into the display properties tab.

How or where can I find out about the API calls necessary to change the screensaver settings? Is there a book which describes all the Win32 (and/or Win16) API calls?”

Mark Horton

Windows 3.1 introduced a handy function called SystemParametersInfo. This reads or sets numerous system parameters including the screensaver settings.

Fig 3 shows a small VB application for Windows 95 which toggles the screensaver on and off. The two key functions, IsActive and SetActive, work by calling SystemParametersInfo. The application checks the current state of the screensaver on loading, so that it can be restored on exit.

Another possibility is for your application to disable the screensaver whenever it has the focus. Windows activates the screensaver by sending a WM_SYSCOMMAND message with wParam set to SC_SCREENSAVE. By intercepting and killing this message, you prevent the screensaver from kicking in. Delphi programmers can trap messages easily but VB users will need an add-on like the Message-Blaster OCX.

Many problems such as this can only be solved using the Windows API. That, in turn, means having a good API reference and the starting point is the Windows SDK help file called WIN31WH.HLP for Windows 3.1, and WIN32.HLP for 32-bit Windows. Surprisingly, Visual Basic 4.0 comes with declarations for the 32-bit API but not the 20Mb help file.

An alternative is Daniel Appleman’s book, *VB Programmer’s Guide to the Windows API*, which provides what’s needed for Windows 3.1 and is to be updated for Win32. ▣

PCW Contacts

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

Contemporary Software supplies Sax Webster £110; Data Widgets 2.0 £99; OLE Tools £149; and VB Tools £99.
Tel **01727 811999**