The developing world

You may think there's not much in it – four packages from two vendors – but, as Tim Anderson finds out, it's the subtle differences that make or break a development tool, especially in today's web-oriented world.

There are few choices when it comes to Windows development tools. To be different, you can go to the GNU C++ website – follow the links to the Win32 port from www.gnu.org – and download a free and highly competent compiler, but you will not get the visual tools and wizards that developers now take for granted. Database developers might use a tool such as PowerBuilder or Clarion, which are both good for building well-structured database applications. You can take the cross-platform route and build everything with Java, again starting with a download, from www.javasoft.com. The majority of Windows development work, though, is done with Microsoft's Visual Basic and Visual C++, and Inprise/Borland's Delphi and C++ Builder. Other options have fallen by the wayside. For example, PowerSoft's Power ++, an innovative RAD development tool for C++, has been discontinued, although the highly-regarded Watcom compiler lives on inside PowerBuilder.

This isn't the whole story. Java is big, and there's an array of visual development tools for Java, including Borland JBuilder, Symantec Visual Café and IBM VisualAge. There is little sense, though, in comparing a Java development tool with a Windows development tool, and *PCW* will look at Java tools in a future article.

The other change in application development is that multi-tier and Internet development is increasingly dominant. This is the half-way house between cross-platform development and applications that run only on Windows. Web development lets you run native code on the server, but supports any client that can use a browser. Even if all the clients happen to be Windows, it greatly simplifies deployment and maintenance. As a side-effect, development tools are focused increasingly on component-based development with web clients in mind. We have looked carefully at this aspect in the reviews that follow.

contents

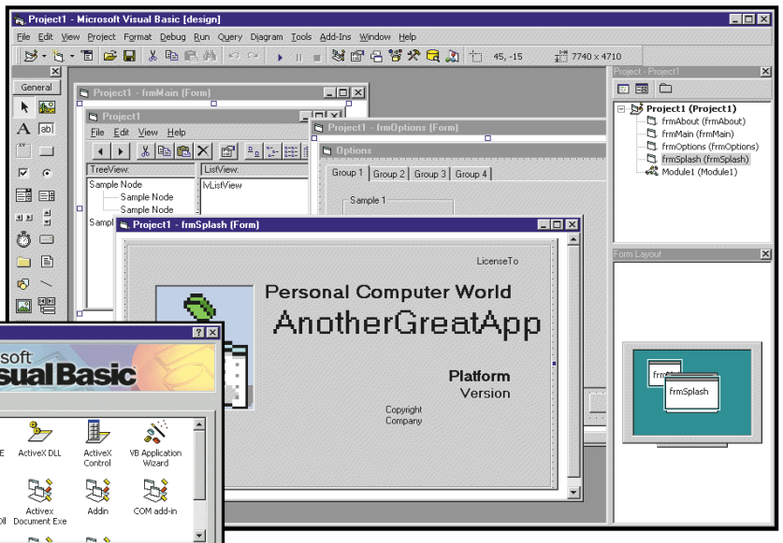
ILLUSTRATION RACHEL OXLEY


visual C++

Microsoft Visual Basic 6.0

IF YOU WORK with Windows, it is hard to avoid Visual Basic. The core language runtime is used by Microsoft Office as a macro language and by third-party applications such as Corel WordPerfect. The full version is said to be the most popular development tool for Windows, and is used by everyone from hobbyists to corporate development teams building enterprise systems.

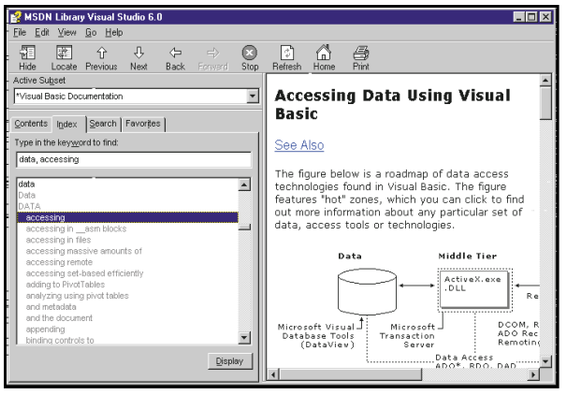
What you get is a visual form builder with a palette of widgets that can be extended by installing additional ActiveX controls. Database support is provided by the JET engine, which is the same one used by Microsoft Access. You can also access data through ADO (ActiveX Data Objects) including any ODBC data source. Most Visual Basic controls are data-aware, which means you can hook them up to a field in a database query so that the control automatically shows the current data.



Visual Basic is an old favourite and remains very capable

VB's new project wizard shows a rich range of project types

Visual Basic has strong advantages. It is easy to learn, and a novice can build a simple Windows application almost immediately after installing the product. The language is safe, and if you stay with native Basic code, you don't need to worry about memory allocation or pointers, as it is handled for you behind the scenes.



Visual Basic's online help is in HTML, packed with information but slow to search

Third-party support is great, and a web search will turn up a choice of ActiveX controls for all but the most obscure requirements. Documentation is good, and although the HTML help used to present it can be slow, the amount of content makes up for it.

A key feature in Visual Basic is that you can build ActiveX executables and DLLs. These are COM components that can be invoked by other applications such as web servers. This feature enables VB to compete as an enterprise development tool. Through DCOM (Distributed Component Object Model) these components can be executed remotely, so that applications can be partitioned, with developers controlling what runs on clients and what runs on servers. VB is the easiest way to build COM components. All you need to do is to start an ActiveX DLL project, add class modules for the objects you want to provide, and type in the required properties and methods. Most of the complexities of COM are hidden, although topics such as threading, instancing, and MTS (Microsoft Transaction Server) transactions do require some effort to learn. This is Visual Basic's most compelling feature.

Unfortunately, there are flaws in Visual Basic. Microsoft's efforts to simplify the language can backfire when power users dig deep into the product. Advanced use of the Windows API is easier using Visual C++. Few things are impossible in VB, but some can be convoluted. Another problem is that when you create an object defined by a VB class, you cannot pass it any parameters, making it hard to initialise it safely. VB doesn't support inheritance either, but in practice this is not a major issue. Advanced developers find VB too much of a black box – errors are hard to track down. Lastly, VB applications can be tricky to deploy because of the large number of files and COM registry entries needed by a typical application.

Visual Basic has special features for web development. DHTML projects create Dynamic HTML pages, hosted within an ActiveX container, in Internet Explorer. Application code is compiled into an ActiveX DLL that runs on the client. Overall, there is little advantage over a conventional Windows application. Webclasses are more interesting. Using Active Server Page technology, Webclasses are ActiveX DLLs that are called by Internet Information Server. The idea behind Webclasses is good, but the documentation is poor, and most ASP developers use other approaches.

Visual Basic is good for beginners, and for creating server-side COM components. For general application development, it's okay, but not the best.

DETAILS

★★★

PRICE Learning £91.65 (£78 ex VAT)
Professional £480.57 (£409 ex VAT)
Enterprise £1,128 (£960 ex VAT)
CONTACT Microsoft 0345 002 000
www.microsoft.com/uk
PROS Easy to learn, good database connectivity, strong COM support
CONS Frustrating for advanced developers, use of many runtime files makes deployment more difficult
OVERALL Powerful and easy to use, but for many projects Visual Basic is not the best choice

Microsoft **Visual C++ 6.0**

THIS IS MICROSOFT'S premier development tool and is used internally to create popular applications such as Word and Excel. Whereas the other products in this group test are designed around a visual form designer, in Visual C++ the code editor is king. What you get is a C++ compiler, the MFC (Microsoft Foundation Classes) library, the Active Template Library (ATL) for building lightweight COM objects, and an IDE (Integrated Development Environment) with numerous wizards and tools. Getting started with Visual C++ is made easier by the New Project wizard, which creates skeleton projects, including full Windows applications, dynamic link libraries, ISAPI extensions for Internet Information Server, database connections and ATL COM objects.

Once a project is started, Visual C++ has a high-productivity development environment, provided that you are familiar with both C++ and the MFC. A tabbed workspace window shows different views of the contents of a project, including a file list, class browser, resource list, and for database projects, a database explorer. The code editor has extra features such as List Members, which shows all the member variables or functions for the selected class or structure, and Parameter Info, which shows the parameter list for the selected function. IntelliSense pops up this information as you type, a valuable productivity benefit. The IDE also supports VB Script macros, complete with a macro recorder. It is also extensible by means of add-ins to give extra functionality.

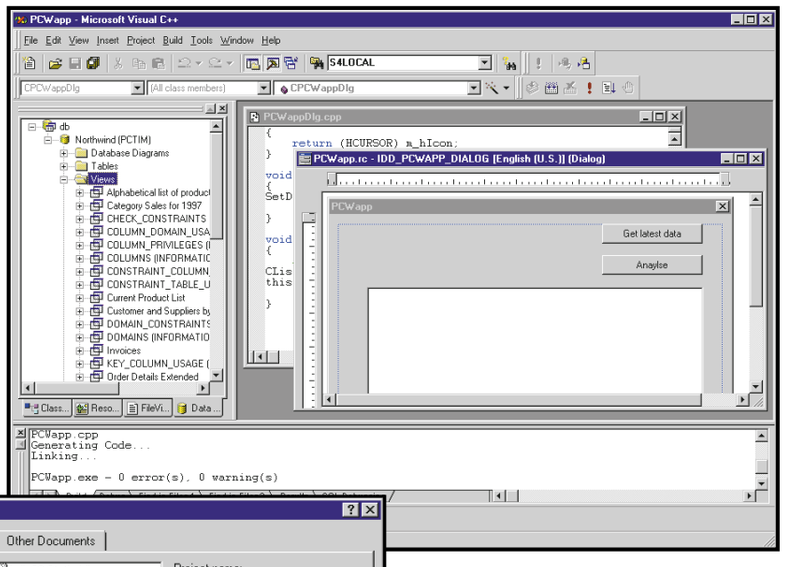
Class Wizard is a tool that automates much of the spadework when working with MFC applications. It will create event handler functions, add member variables to dialogs so that your application can read the user's selections, or edit the COM automation interface for appropriate classes. Working with Class Wizard and the dialog editor is almost as quick as working with Visual Basic or Delphi.

MFC itself has a mixed reputation. It was designed to be a thin wrapper over the Windows API, as opposed to Delphi's VCL, which is a high-level wrapper. The advantage is that you can easily access the whole API. The disadvantage is a steep learning curve and verbose code. The performance of the resulting applications is good, and Visual C++ applications generally run faster than those built with Visual Basic.

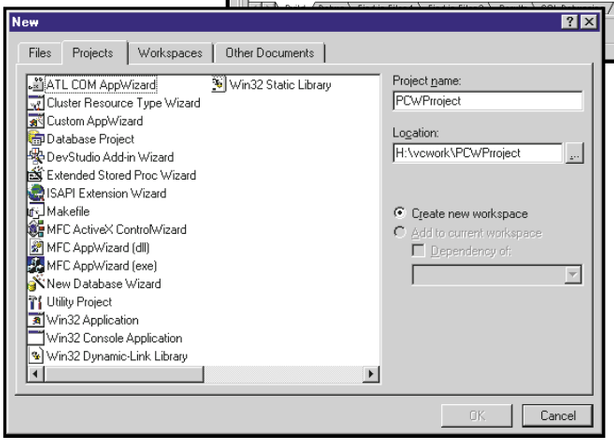
For online documentation, Visual C++ uses the MSDN library, as also found in Visual Basic and throughout Visual Studio, Microsoft's suite of development tools. Because it contains the latest version of the Windows platform SDK, this is a near-essential reference for any Windows developer, so to have it supplied with the product is a bonus. The catch with the MSDN library is that it is slow to load, and because it covers several different tools it can be confusing. Early versions of Visual C++, which integrated help as a docking window in the IDE, were easier to use.

One reason for writing code in C or C++ is that compilers exist for most platforms. In September 1998 the accepted standard for C++ was approved by an ANSI/ISO committee. Although Visual C++ has made some progress in supporting this standard, it is outclassed in this respect by a number of other compilers, including Borland C++, GNU C++ and Metrowerks Codewarrior. The cross-platform Standard Template Library (STL) is supplied with Visual C++, but it is not the best compiler for STL work. In other words, while Visual C++ is excellent for Windows development, it's less attractive if you want to create applications that compile on other platforms. Applications that call the Windows API cannot be cross-platform in any case, so this may not matter in many situations, although it does diminish the attraction of Visual C++ as a learning tool.

Despite these hesitations, Visual C++ is a delight to use. It has a good editor and the best environment for navigating code. The compiler itself is fast and efficient, and the debugging features are comprehensive. ATL is a breakthrough, enabling small, efficient COM components to be created easily.



The excellent Visual C++ IDE, showing the data explorer and dialog editor



Visual C++ has a fine range of project wizards to get you started

DETAILS

★★★★

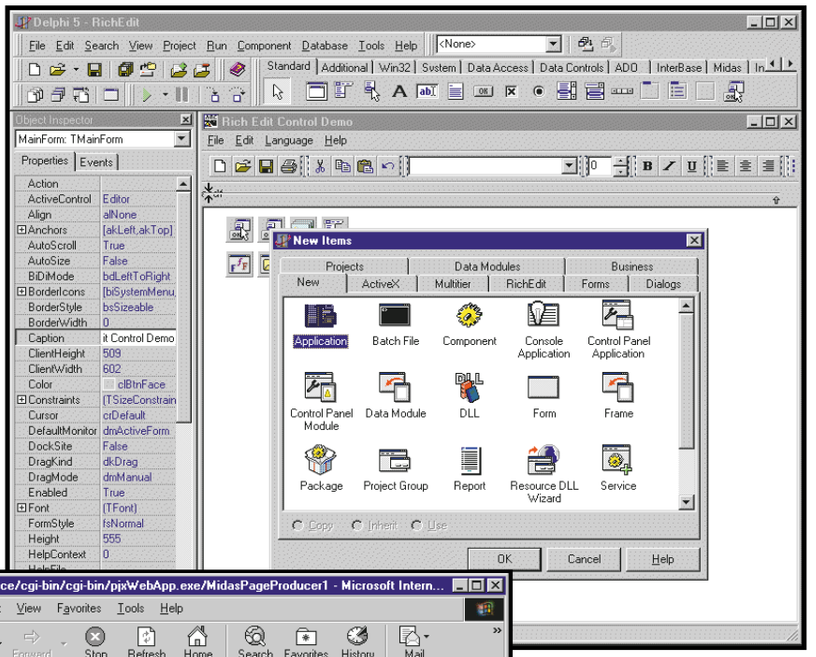
PRICE Standard £91.65 (£78 ex VAT)
Professional £478.23 (£407 ex VAT)
Enterprise £1,128 (£960 ex VAT)
CONTACT Microsoft 0345 002 000
www.microsoft.com/uk
PROS Productive IDE, ATL and MFC speed development, array of wizards and tools
CONS Not a RAD tool, weak standards compliance, steep learning curve
OVERALL The best tool for coding to the heart of Windows, but not recommended for novices

Borland **Delphi 5.0**

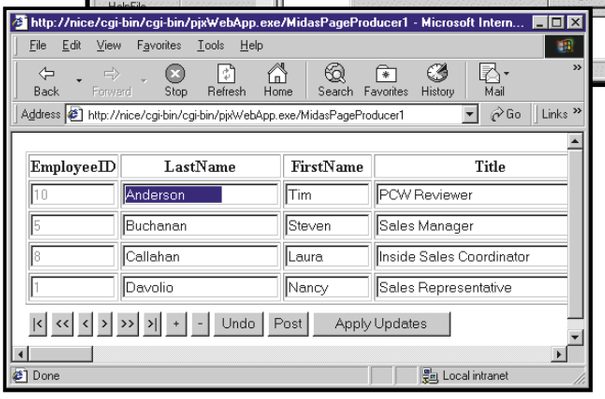
BORLAND DELPHI is a product that has many characteristics of the ideal development tool. The IDE is modelled on Visual Basic, and lets you create applications by placing visual components on a form and setting their properties. The IDE automatically creates event handlers such as ButtonClick procedures, where you can write code to further control the application's components.

So far Delphi sounds much like Visual Basic, but there are important differences. Although the IDE is VB-like, Delphi looks more like a C++ product underneath. It is a full, native code compiler, and you can build applications that do not require any runtime libraries. Most of the source code to Delphi's class library is available, making it easier to track down bugs or to write your own modified components.

Delphi's language is Pascal, modified by Borland to provide full object orientation, including inheritance. The language supports pointers, making it easier to work with the Windows API or other C or C++ libraries. You can build ordinary DLLs in Delphi, whereas VB can only create ActiveX DLLs. Unlike VB, Delphi supports multi-threading. And whereas VB does error handling through a crude On Error Goto statement, Delphi provides exception handling and a 'try ... finally construct' which makes it easier to write safe code. The list goes on, and in essence Delphi has most of the productivity of VB, but without its irritations and limitations.



The Delphi environment looks similar to Visual Basic, but packs a greater punch



This web database application is powered by a server-side Delphi web application

Visual Basic remains more popular than Delphi, and the reasons are not entirely Microsoft's marketing. Because of its flexibility, Delphi is a little more demanding than VB, and does not provide as much protection from coding errors. There is less third-party support, although some ActiveX components intended primarily for VB will also work in Delphi. Support for COM in Delphi is actually very good, although it does not have the quick and easy route to building COM components that Visual Basic offers. You can do it in Delphi, but there is more to learn.

Another problem area in Delphi has been its database support. Visual Basic comes with the JET database engine and integrates easily with Access and ODBC data sources. Delphi's equivalent, the Borland Database Engine (BDE), works primarily with Paradox or dBase data, and, although ODBC is supported, it is not the best ODBC client. To get SQL links for popular server databases such as Oracle and SQL Server, you need to purchase the high-end client-server or Enterprise version of Delphi. The situation has improved in Delphi 5.0, which now supports ActiveX Data Objects without use of the BDE, although again this is not supplied with the Standard or Professional edition, but has to be purchased as an add-on.

The core of Delphi is exactly what it should be, but the way Delphi has evolved is, in some ways, disappointing. Borland has put its main effort into pitching for the Enterprise market, by giving Delphi the ability to build CORBA objects, integrating with the Visibroker ORB (Object Request Broker), another product from Inprise/Borland. You can build multi-tier database applications using MIDAS (Multi-tier distributed application services suite), a sophisticated mechanism for sending datasets across a network that supports both COM and CORBA. Good though they are, few Delphi developers need these features, which require expensive runtime licences.

What Delphi users want is an outstanding RAD tool for Windows, and here Delphi has been slow in keeping up with the latest features. For example, ADO was supported in Visual Basic for some time before it arrived in Delphi. Delphi's Rich Text Edit control doesn't support the latest features of the Windows common control, on which it is based. Such niggles spoil Delphi, but determined developers, with energetic third-party support, always find a way round problems. The latest news is that Delphi for Linux is under development – exciting for those thinking of porting applications.

DETAILS

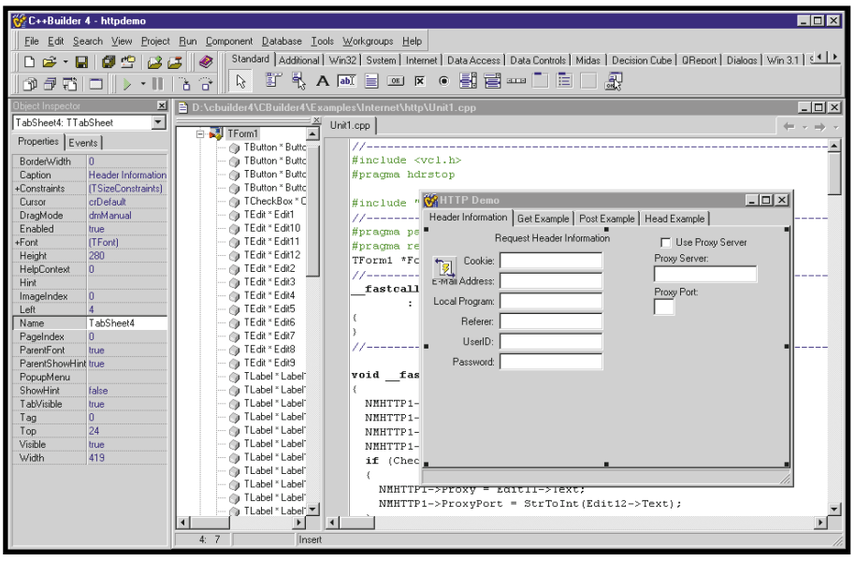
★★★★

PRICE Standard £81.01 (£69 ex VAT)
Professional £527.58 (£449 ex VAT)
Enterprise £1,996.33 (£,1699 ex VAT)
CONTACT Borland 0118 932 0022
www.borland.com/delphi
PROS RAD without compromise: the power of C++ and the productivity of Visual Basic
CONS A little slow to support new Windows features. Expensive Enterprise features irrelevant to many developers
OVERALL Any Windows developer should at least consider using Delphi

Borland C++ Builder 4.0

WHEN DELPHI WAS first released, some developers were put off by its language, Pascal, when the dominant language for Windows development is C++. This product is Borland's answer. C++ Builder and Delphi share the same back-end compiler, but Borland has made the Pascal Visual Component Library look like a C++ class library. The developer can work entirely in C++, while benefiting from the same rapid, visual development that Delphi offers.

Most of what applies to Delphi also applies to C++ Builder. The IDE looks almost the same, until you open the code editor and find C++ in place of Pascal. Like Delphi, this product comes with a full range of visual and non-visual components, database access, based on the Borland Database Engine and SQL Links client-server drivers, and support for multi-tier applications, using either COM or CORBA components. Since Delphi add-ons can easily be converted for C++ Builder, third-party support is similar to that of Delphi.



C++ Builder 4.0 looks very like Delphi, because it shares the same component library

Borland's earlier C++ product was simply called Borland C++ and was a direct competitor to Microsoft's Visual C++. It had its own class library, the Object Windows Library (OWL). Many developers preferred OWL to MFC, as it does a better job of shielding developers from the intricacies of the Windows API, but MFC dominated because of Microsoft's support. C++ Builder combines Delphi with Borland C++. Support for OWL is still included, although there is no documentation or tools.

Borland C++ Builder is a hybrid Pascal/C++ product that in practice works well

Although Borland C++ has been discontinued, not every feature has been retained. For example, Borland C++ had a macro language for automating the editor and IDE, and a code profiler for optimising performance. Borland C++ was a better product for non-visual C++ development. That doesn't mean C++ Builder is poor for general C++ work. It has good support for the Standard Template Library, and Borland is committed to support for ANSI/ISO C++.

C++ Builder will compile MFC and comes with a version of Microsoft's library, but not the latest. You can convert projects from Visual C++ 5.0. Note the visual designers in C++ Builder are for VCL projects, and not applicable to MFC. So Visual C++ is a better choice for MFC work, but the compatibility might be useful to a Visual C++ migrant, or for third-party libraries designed with MFC in mind.

The most compelling reason to use C++ Builder rather than Delphi is that you prefer C++ to Pascal. The Windows Platform SDK tends to assume use of C++, which is another reason to use this product. An example benefit is that C++ Builder creates COM server components using Microsoft's ActiveX Template Library, whereas Delphi cannot use this. C++ Builder is more flexible than Delphi – you can compile both Pascal and C++ code, but Delphi only compiles Pascal.

Despite these strengths, C++ Builder is a compromise. Because the VCL is Pascal, C++ developers will find they have to become familiar with both languages to learn the product in depth. It also makes working with the VCL less intuitive for a C++ developer than if it were pure C++. Another irritation is that new versions of C++ Builder tend to lag behind Delphi. At the time of writing Delphi 5.0 is available but C++ Builder is still at version 4.0, although version 5.0 is in the pipeline. On balance, developers who don't care which language they use are probably better off with Delphi.

Another interesting choice is between Visual C++ and C++ Builder. For a code-intensive project with a simple user interface, Visual C++ is preferable. Using Microsoft's compiler with MFC is also a safe route for those who want to keep up-to-date with Windows. However, using C++ Builder you can snap together a rich graphical interface with database access in a fraction of the time it would take in Visual C++.

DETAILS

★★★★

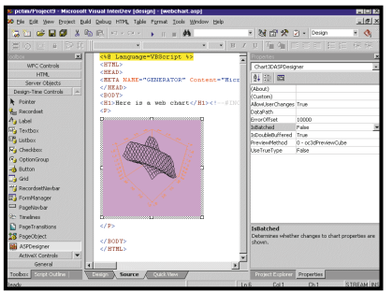
PRICE Standard £99 (£84.26 ex VAT)
Professional £527.58 (£449 ex VAT)
Enterprise £1,996.33 (£1,699 ex VAT)
CONTACT Borland 0118 932 0022
www.borland.com
PROS RAD for C++ developers, commitment to ANSI/ISO standards
CONS Uses Pascal class library, lags behind Delphi
OVERALL An excellent application builder for C++ developers

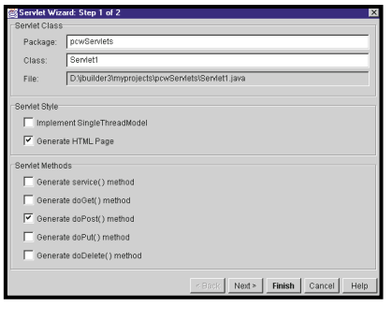
Developing for the web

The approach to take with web development is to choose the tool that speaks your language.

In contrast to the Windows development arena, there is much activity in providing web development tools. Here, the platform is a web server and the client a browser. The complexity of web development is that the different application layers require different languages and tools.

For client-side scripting the de facto language is Javascript, also known as Jscript and Ecmascript. Internet Explorer also supports VB Script, but this has not been implemented by other browsers. JavaScript is not difficult to learn, but the challenge of scripting is programming the API exposed by the browser, which takes the form of an object model. Each browser version presents a different object model, and many developers use Notepad or a programmer's editor to code JavaScript, with a browser reference handy, to get round the incompatibilities. Tools such as Microsoft's Visual InterDev do syntax highlighting, code completion and script debugging for JavaScript, but compatibility can still cause headaches.





TOP Visual InterDev is perplexing at first, but offers a comfortable environment for scripting and ASP development BOTTOM Java Servlets are a great choice for server-side web components that are not limited to Windows

Macromedia's Dreamweaver is great for web development, particularly since it comes bundled with Allaire Homesite for text-based editing and scripting

Some web-authoring tools provide libraries of JavaScript functions written with cross-browser compatibility in mind. With Macromedia's Dreamweaver, you can create dynamic web pages while hardly being aware that pre-authored scripts are being used.

The other options for client-side browser development are Java applets and ActiveX controls. There are many Java tools available, such as Borland JBuilder, IBM VisualAge for Java, and Symantec Visual Café, or you can work with Java using a simple text editor and the Java Development Kit (JDK) downloaded from Sun. Java applets are powerful, but not that popular, thanks to variable browser support and slow loading times. ActiveX controls are even less popular, as Windows-only support loses a key feature of web development, and there are security concerns.

At the server end there are several technologies. Microsoft's Active Server Pages is a solution for Internet Information Server that involves server-side scripting in JavaScript or VB Script, and COM components running on the server. Through Distributed COM (DCOM), components can interact with others running elsewhere on the network.

The premier tool for ASP is Visual InterDev. This provides a visual HTML designer, script editor and debugger, and a clever technology called design-time controls, which partially automates script writing for data access and server-side components. There's a range of tools, including a database explorer, site diagram, style-sheet editor, and a deployment wizard.

A favourite on non-Windows platforms, with similar features to ASP but without COM-component support, is PHP (Hypertext Preprocessor). Its strength is its database support. Functions for accessing numerous database servers including ODBC are available for PHP.

Another web technology is Java Servlets and Java Server Pages (JSP). For these you need an add-on such as Allaire's JRun, or a web server such as Sun's Java Web Server that has integrated support. Java is outstanding for server-side component development. It was designed with the Internet in mind, with support both for databases, through JDBC, and distributed applications through Remote Method Invocation (RMI) or CORBA.

If you want a web development tool, then for ASP go with Visual InterDev. For more general work, Macromedia's Dreamweaver comes bundled with Homesite 4.5 – the best of both worlds.

COM and ActiveX

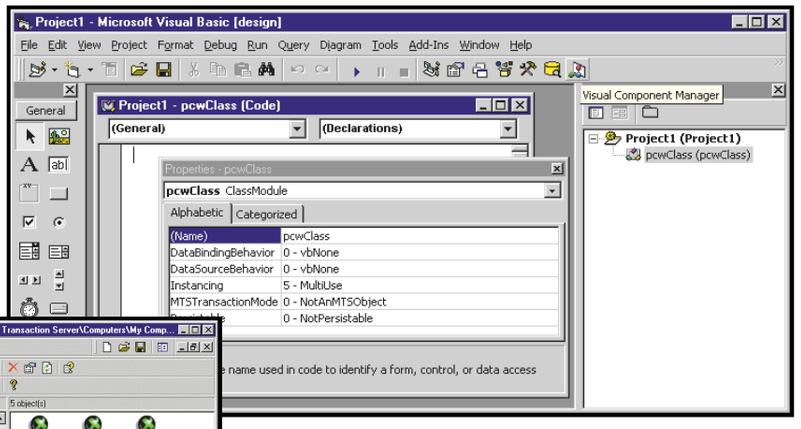
While Windows remains the mainstream OS, COM will continue to be the developers' choice.

COM stands for Component Object Model, and it describes the technology whereby software components can communicate with each other, even across process and network boundaries. Windows features that depend on COM include ActiveX controls, OLE (Object Linking and Embedding, for example inserting an Excel spreadsheet into a Word document), and ActiveX automation where one application programmatically controls another.

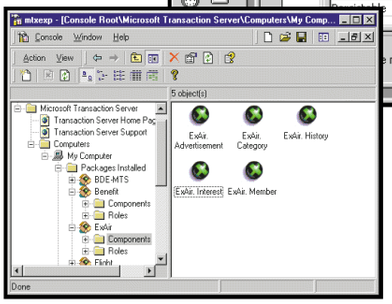
The power of COM is that it lets you break down an application into components. These components can then be deployed flexibly. Windows DNA (Distributed Internet Architecture) is Microsoft's name for a multi-tier architecture that separates presentation, business logic, and database services. In a web application, for instance, the browser handles presentation, business logic components run on the web server, and data is accessed from a third machine.

ActiveX controls are COM components, designed to be hosted by a container, such as a web page. Once installed on a user's machine, they are fast, as the web page only needs to contain the data for the control. The control itself gives the same performance as any other compiled Windows application. Web users are wary of ActiveX controls, since they run as Windows executables with full access to the client computer, posing a security risk, although controls can be signed for safe identification.

In its early days COM was unreliable but the system has matured and is now highly usable. ActiveX controls are not popular on the web, but are widely used in Visual Basic applications. COM server components have great potential in web applications. Component-based, multi-tier applications are the way forward, although COM itself has rivals. Java supports RMI (Remote Method Invocation) for distributed applications, while Java servlets perform a similar role to COM servers in web applications. CORBA (Common Object Request Broker Architecture) is another approach to distributed applications, more advanced technically, but complex and expensive to implement. In a Windows environment, it's COM that remains the natural choice.



Creating a COM automation server in Visual Basic takes just a few simple steps



Microsoft Transaction Server plays a key role in COM distributed and web applications. In Windows 2000 it is built into the operating system and renamed Component Services

Understanding the Windows API

Windows API (Application Programming Interface) is the programmatic way to control the operating system. When a Windows application runs, it makes thousands of calls to the core Windows runtime libraries, such as kernel32.Dll, user32.dll and gdi32.dll. In the early days of Windows, each call had to be coded manually. Successive development tools have found ways of wrapping the Windows API as higher-level objects, called a class library. In Delphi, the following code creates and displays a window:

```
f := TForm.create↙
(Application);
f.show;
```

(*Key:* ↙ *code string continues*)

Some versions of Delphi supply the source code for forms.pas, where the TForm class is defined. Open that, and you can see the extensive code which executes behind the scenes, including calls to the CreateWindowEx API function that provide the new window.

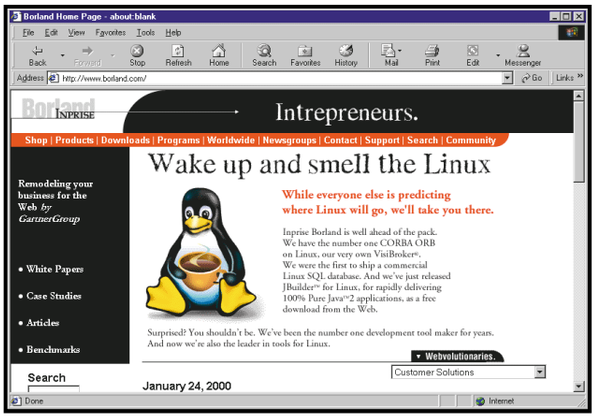
What this means is that when you choose a Windows development tool you are also choosing a class library. For Visual C++ it is the Microsoft Foundation Classes (MFC). For Delphi and C++Builder it is the Visual Component Library (VCL). For Visual Basic it is hidden from view, but VB also wraps the Windows API in a high-level, object-based library. These libraries are sub-sets of the full API, so you will need to extend or bypass them occasionally to get at hidden features. This is harder in VB than in the MFC or VCL – a source of frustration for VB developers.

Development futures

While Microsoft is busy with Visual Studio 7.0, Borland is taking the Linux road.

During 2000 there will be new releases of both Visual Studio and C++ Builder, and at least an update for Delphi 5.0. Clearly, the priority for both product families is that Windows 2000 is fully supported. Microsoft has produced a certification scheme for the new operating system, which specifies the requirements for well-behaved applications. There are 40 key points in seven categories, and not all of them are trivial to implement. They include installation using the Windows Installer service, proper support for roaming users, careful interface design to support multiple monitors, co-operation with power management schemes, and for distributed applications, Active Directory support.

Microsoft has given some snippets of information about features that will appear in Visual Studio 7.0. Two of the most important additions are web-related. First comes a new kind of form, called a web form. A web form will be an ASP page that calls compiled COM-server components to achieve its functionality. This is ideal for intranet or Internet applications that work cross-platform and without the need to install anything on the client. You can do this with the current Visual Studio, but the difference in Visual Studio 7.0 is that working with a web form is like working with a traditional Visual Basic form. You click to open the code for an event handler, and click the Run icon to execute the application. As VB developers are used to this model, the transition to ASP applications will be easier.



Borland embraces Linux: Shrewd marketing, or cashing in?

The second new feature is that COM components will be able to communicate via XML, using the Simple Object Access Protocol (SOAP). Provided you have an HTTP connection, you will be able to call the methods and set the properties of remote objects. With XML there is nothing to configure and the quality of the network connection is less critical, provided that the data is delivered. According to Microsoft, Visual Studio 7.0 will make working with XML interfaces as easy as programming COM objects.

Inprise/Borland aims to reduce dependence on Windows and to support cross-platform development. Kylix is the codename for new Linux versions of Delphi and C++ Builder. How easy it will be to port existing applications will depend both on Kylix and the application. The Windows API doesn't exist on Linux, and the graphical widgets are different.

The company is also promoting some products by making them free. Alongside the Interbase client-server database, the Borland C++ compiler is to be free for download. This will be command-line tools only, as C++ Builder will remain a commercial product. Another recent development is JBuilder Foundation, a free version of a RAD tool for Java that runs on both Windows and Linux. Clearly, the intention is to make a splash on Linux and obtain a larger user-base, benefiting by selling related products and services.

Visual Studio: The ultimate developer bundle

Microsoft now bundles all its tools into a single suite, although they are still available separately. If you need more than one, the suite is better value than buying separate products. The core products are Visual Basic, Visual C++, Visual InterDev, Visual J++, Visual FoxPro, and in the high-end edition, Visual SourceSafe for source management and version control. The suite is ideal for developers who want to build a front-end in Visual Basic, calling DLLs created with Visual C++. Another benefit is Microsoft is forced to ensure that the products work together – a tough task, given the number of shared components.

The venerable Visual FoxPro is a database development product whose language is based on the original dBase. It has acquired full object orientation, and FoxPro is now equally useful as a client to a server database as it is working with its native dBase-like data format. While in some ways better than Visual Basic, FoxPro never managed to lose its image as a legacy product, and has acquired few new developers. It also suffers from large runtime files.

Visual J++ 6.0 is a brilliant product with an uncertain future. It combines Java with a Windows-specific component library, and is the closest Microsoft has come to rivalling Delphi for productivity. Legal disputes with Sun and the preference of Java developers for cross-platform products may have killed Visual J++, putting more pressure on Microsoft to improve the RAD capabilities of Visual C++.

Editor's **Choice**

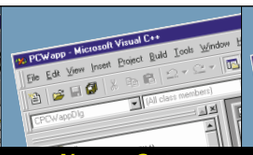
There are several key issues in choosing a development tool. First, power. Visual C++ and C++ Builder are fractionally ahead of Delphi, but not by much. Visual Basic comes in last. Despite being good enough for most tasks, there are still things that you cannot do or cannot easily do in VB. Second, productivity. For the simplest projects, Visual Basic has the advantage, but for projects of even moderate complexity, Delphi is the winner, followed by C++ Builder, Visual Basic and finally Visual C++.

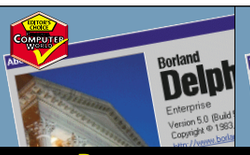
The third factor is ease of use, taking into account the design of the IDE and the ease of learning how to accomplish new tasks. Again, although Visual Basic is easier to begin with, Delphi is the long-term winner. Fourth comes database connectivity. Visual Basic and Visual C++ are better ODBC clients, but now that Delphi properly supports ADO, it is a close call. Overall, Delphi easily achieves the **Editor's Choice** award.

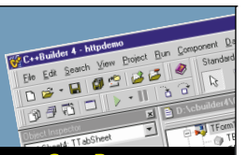
This has been true for several years now. Delphi's VCL is a generation ahead of both MFC and the black-box approach of Visual Basic. Even though Borland's development of Delphi has been unspectacular, Microsoft has not yet bridged this fundamental gap. The closest it has come is with the ill-fated Visual J++ 6.0, which now appears to be a false trail.

One reason Microsoft has not worried much is that its tools are still more popular. Visual Basic is hard to avoid, being embedded into Microsoft Office, Internet Explorer and, through the Windows Scripting Host, everywhere in Windows. Visual C++ has a superb IDE and is a comfortable choice if you are using Microsoft's documentation, since it is assumed that you use it. Nevertheless, most tasks can be done more quickly in Delphi or C++ Builder, with very little sacrificed in terms of performance or flexibility.

Table of features







Product	Visual Basic	Visual C++	Delphi	C++ Builder
Version	6.0	6.0	5.0	4.0
Supplier	Microsoft	Microsoft	Borland	Borland
Tel	0345 002 000	0345 002 000	0118 932 0022	0118 932 0022
URL	www.microsoft.com/uk	www.microsoft.com/uk	www.borland.com/delphi	www.borland.com
Prices inc VAT	£91.65/£480.57/£1,128	£91.65/£478.23/£1,128	£81.01/£527.58/£1,996.33	£99/£527.58/£1,996.33
Prices ex VAT	£78/£409/£960	£78/£407/£960	£69/£449/£1,699	£84.26/£449/£1,699
Language	Basic	C++	Pascal	C++
Native code compilation	✓ with runtime	✓	✓	✓
Native/bundled database engine	JET	JET	BDE	BDE
Report designer	Crystal Reports	None	QuickReport	QuickReport
ODBC support	✓	✓	✓	✓
ADO support	✓	✓	✓ (add-on)	✘
Host ActiveX controls	✓	✓	✓	✓
Easily create ActiveX controls	✓	✓	✓	✘
Supports Active Template Library	N/A	✓	N/A	✓
Create COM servers	✓	✓	✓	✓
Create web server applications	✓	✓	✓	✓
Setup toolkit	Wizard	InstallShield	InstallShield	InstallShield (Not Standard edition)
Windows Installer support	✓	✓	✘	✘
Class/Component library	Built-in	MFC	VCL	VCL/MFC/OWL
Component library source supplied	N/A	✓	✓	✓
Profiler	✘	✓ (Not learning edition)	✘	✘
Scriptable IDE	✘	✓	✘	✘