REALbasic versus Visual Basic

By Jerry Lee Ford, Jr.

November 2006

When is comes to the development of Windows applications, REALbasic's main competitor it Microsoft's Visual Basic programming language. If you are a Windows programmer with a background in Visual Basic, you are going to find that REALbasic is remarkably similar to Visual Basic. As a result, you will be able to get up and running very quickly and will be able to leverage just about all of your Visual Basic programming experience. This is especially true for Visual Basic programmers with a Visual Basic 6 background because REALbasic very closely mirrors much of what is in Visual Basic 6. Programmers with a background based on Visual Basic .NET will also find the transition to REALbasic relatively smooth and will be able to bring over most of their Visual Basic programming experience, less any .NET specific features.

Because REALbasic is so similar to Visual Basic, you will find that it is relatively easy to port your Visual Basic applications over to REALbasic. In fact, you will probably find that you can port your applications over in an hour or a day instead of weeks or months, as would be the case if you moved from Visual Basic over to another programming language such as C++ or Java.

Visual Basic 6 is more similar to REALbasic than Visual Basic. NET. As a result, Visual Basic 6 applications will port over more easily, with less manual modification on your part. Porting over Visual Basic .NET applications will involved rewriting any Visual Basic .NET functionality with equivalent REALbasic statements.

Visual Basic 6

Microsoft initially introduced Visual Basic in 1991. It became instantly popular as a programming language that was easy to learn and quickly led to a whole new generation of Windows software. Over the years, Microsoft continued to enhance Visual Basic, adding support for databases, ActiveX, COM and so on, culminating in Visual Basic 6. The Component Object Model or COM gave Windows programmers access to various system resources. In addition, Visual Basic needs access to dynamic link library (DLL) files for some core functionality. Things often got ugly when Visual Basic programmers deployed their applications because the programmers also had to ensure that all the DLLs their applications needed were also installed on customer computers. This typically meant creating a deployment package that included copies of all required DLL files.

Unfortunately, problems sometimes occurred because the deployment packages often replaced DLL files already installed on customer computers with older versions of these DLL files. This sometimes broke other applications belonging to customers. This situation occurred so frequently that programmers nicknamed the problem *DLL hell*.

The .NET Framework

In the mid-1990s the Internet really took off. One fast rising star was the Java programming language, which provided programmers with the ability to create both server and client-platform software applications. Surprisingly, in many ways Microsoft almost missed the boat and was completely caught off guard by the popularity of Java.

Note: In order for Java to work, a collection of software known as the Java Virtual Machine has to be installed on any computer where Java is to run. While Java quickly became a major player in server-side application development, it never really took off on the client-side.

Microsoft took notice of Java's success and, realizing that the Internet was the future, Microsoft came up with the idea for its .NET Framework. The .NET Framework facilitates multi-language application development and provides support for the development of Windows, web and mobile applications.

The .NET Framework was now provides the core development services required for Windows application development environment. A strong understanding of .NET is now a requirement for all Visual Basic programmers. In order to ensure the widespread deployment of the .NET Framework, Microsoft made its installation a default component of Windows XP's installation. Microsoft also provides it as a free download at http://msdn.microsoft.com/netframework.

The .NET Framework serves as an interface that manages the interaction between the application and the Windows operating system. .NET also enables the development of applications using more than one .NET enabled programming language. .NET is also responsible for translating application code into a format that can run on the targeted platform.

Note: The .NET Framework resolved Visual Basic's DLL Hell issues by eliminating the need to rely on DLL files. Functionality formerly provided by DLL files is not provided by the .NET Framework. However, Visual Basic programmers now have to concern themselves with whether or not the .NET Framework is installed on customer computers.

Visual Studio

Whereas REALbasic is designed to provide cross-platform compatibility, Visual Basic .NET is designed to work as one of a number of member languages that make up Microsoft's Visual Studio software development suite. Languages included as part of the Visual Studio, include:

- Visual Basic
- Visual C# 2005 Express.
- Visual C++ 2005 Express.
- Visual J# Express.

Visual Basic .NET

In order to get it to work with the .NET Framework, Microsoft had to go back and redesign Visual Basic from the ground up. The end result was a totally new version of Visual Basic, termed Visual Basic .NET. Because Visual Basic .NET was now deeply integrated with the .NET Framework, it took longer to learn than previous versions of Visual Basic.

One of the results of Visual Basic .NET's new architecture was that Visual Basic programmers could no longer take for granted that the applications they created with previous versions of Visual Basic would be compatible with Visual Basic .NET. Backward compatibility was simply not a primary concern for Microsoft when it decided to redesign Visual Basic and make it .NET compatible.

The lack of compatibility between Visual Basic 6 and Visual Basic .NET has frustrated a lot of Visual Basic programmers who had invested years of time and effort in creating their Visual Basic applications. The idea of going back and having to redesign their applications just to make them .NET compatible was a daunting challenge.

Note: In order to run a Visual Basic .NET application on a Windows computer, the .NET Framework must be installed. Microsoft has upgraded the .NET Framework a number of times since its initial introduction in 2002. Visual Basic .NET 2002 was designed to work with .NET Framework 1.0. Visual Basic .NET 2003 worked with .NET Framework 1.1. Visual Basic .NET 2005 worked with .NET Framework 2.0. While a Visual Basic .NET 2002 application can run on a computer running any version of .NET, things get ugly when, for example, you try running a Visual Basic .NET 2003 application on a computer running a version of the .NET Framework other than .NET Framework 2.0.

Ripples of Discontent in the Visual Basic Community

Visual Basic programmers were eagerly looking forward to hearing about what they expected to be Visual Basic 7 when Microsoft announced and end of the Visual Basic 6 line and the introduction of Visual Basic .NET. The end result was that Visual Basic programmers could no longer take for granted that their current applications would be automatically compatible with the next version of Visual Basic. In fact, when Microsoft redesigned Visual Basic .NET, it did so without regard to backward compatibility. This left many Visual Basic programmers unhappy.

On April 2005 Microsoft ended its free support for Visual Basic 6. In response, thousands of Visual Basic programmers signed a petition requesting that Microsoft continue providing free technical support and that it also add support for Visual Basic 6 to Visual Studio. However, Microsoft declined to change its direction, citing that paid support was still available for Visual Basic 6 through March 2008.

As a result of Microsoft's decision to drop free technical support for Visual Basic 6 and its refusal to include support for Visual Basic 6 in Visual Studio, many Visual Basic programmers feel that Microsoft is abandoning them. They feel that Microsoft's refusal to compromise is a direct result of a Microsoft plan to force Visual Basic programmers into the direction of application development for Windows XP and Microsoft's next operating system, Microsoft Vista.

As a response to the idea that Microsoft is more focused on forcing Windows programmers into developing applications for its newest operating system and thus to use the .NET Framework, REAL Software actively promotes REALbasic as a software development tool that is designed to meet the needs of its customers. Unlike Microsoft, REAL Software is not handicapped in the development of its programming language by having an operating system to promote.

A Head-to-Head Comparison Between REALbasic and Visual Basic

REALbasic and Visual Basic are very similar to one another. However, there are plenty of important differences. These similarities and differences are highlighted in the following sections.

The Development Environment

Both REALbasic and Visual Basic's IDE are very similar and their development process is nearly identical. You start by adding controls to a Window (called a Form in Visual Basic). Then you set Window and control properties and add code. Next, you test your application and when you are ready, you compile your stand-alone application.

Language Similarities

Both REALbasic and Visual Basic are object-orientated programming languages. Both share a common set of keywords (If, Then, Else, etc). Both have numerous functions in common.

Both also share a common syntax and both use dot notation to reference properties. The list of similarities goes on an on.

In many ways REALbasic can be looked at as being the next step in the evolution of Visual Basic 6 whereas Visual Basic .NET can be viewed more as being a new divergent form of the language. The end result is that it is usually easier and quicker to port a Visual Basic 6 application over to REALbasic than it is to modify it to run under Visual Basic .NET.

Platform Support

Visual Basic only supports Windows operating systems, specifically on those Windows operating systems running the .NET Framework. REALbasic, on the other hand, is designed to support cross-platform development on Macintosh, Windows and Linux. Of course, more platforms mean more customers and more sales.

Distribution Issues

Visual Basic 6 applications may required specific DLLs be installed on a computer for it to execute and is thus subject to DLL Hell. Visual Basic .NET applications require that the .NET Framework be installed in order for the applications to execute, thus requiring additional system overhead. REALbasic applications run natively on any supported operating systems without requiring the overhead of a framework or risking DLL Hell.

Windows Ready Applications

Visual Basic applications are, of course, automatically designed to work and act like any other Windows application. REALbasic applications that run on Windows are also automatically Windows theme ready, meaning that they will run and look like any other Windows application running on Windows 98, ME, NT, 2000, XP and Vista. REALbasic applications also support Windows technologies such as ActiveX and COM as well as the Windows registry.

Support for PDAs and Mobile Devices

One are where Visual Basic has capabilities not matched by REALbasic is Visual Basic's ability to support application development for PDAs and mobile devices such as cell phones.