

Introduction to REALbasic

“Part 1: The Roots of REALbasic”

by *Wally Wang*

When Microsoft first introduced Windows, it was a real pain for developers. All the familiar tools that programmers used with MS-DOS wouldn't work to create Windows programs. If you wanted to write a Windows program, you had to learn C and learn all the intricacies of the Windows API (Application Programming Interface). This wasn't necessarily difficult, just time-consuming. If you didn't want to learn C and you wanted to write Windows programs, you were pretty much out of luck.

Then Microsoft introduced a revolutionary programming tool called Visual Basic, which made writing Windows programs easy in two ways. First, instead of learning the cryptic C programming language, you could use the much friendlier and easier BASIC programming language.

Second, writing Windows programs meant creating three parts of a program. First, you had to write commands to create all the user interface—all the windows, dialog boxes and buttons that you see in every program. Next, you had to write commands that made that user interface actually work when the user clicked, dragged, or tried to manipulate it. Finally, you had to write commands that actually made your program do something.

Creating these three parts of a program meant that programming was tedious, time-consuming and error-prone. If any of those three parts of your program didn't work, your program wouldn't work. Programming Windows with C meant writing, debugging and testing all three parts of your program.

In one swoop, Visual Basic eliminated these barriers to Windows programming. Instead of writing commands to create a user interface and then a second set of commands to make that user interface work, Visual Basic just let you draw your user interface without writing a single command at all. After you drew your user interface, it would work perfectly every time. With Visual Basic, you could create a working user interface in a fraction of the time required to create that same user interface using C.

After drawing your user interface in Visual Basic, you just had to write simple BASIC commands to make your program work. Where C made you write commands to create your user interface, commands to make your user interface work, and commands to make your program do something worthwhile, Visual Basic just let you focus on writing commands to make your program work, essentially cutting your programming time by two-thirds.

This new programming paradigm, dubbed rapid-application development (RAD), made Visual Basic the best programming tool around. If you wanted to write Windows programs quickly with less code, the best choice was clearly Visual Basic.

Visual Basic took the programming world by storm and went through six different versions before Microsoft made a fatal mistake. First, Visual Basic was great for creating Windows programs, but it could never create cross-platform programs that ran on other operating systems. While C programmers could transplant their programming skills to program practically anything, Visual

Basic programmers were stuck writing Windows programs. That in itself wasn't so bad while Windows remained the dominant operating system on the planet. The problem began when Windows began to slip with the growing popularity of both Linux and Mac OS X.

A second, more crucial mistake that Microsoft made was changing the Visual Basic language. Microsoft developed a new programming framework dubbed .NET. The idea behind .NET was to allow multiple programming languages to work seamlessly together. It worked, but the cost of making Visual Basic adapt to this new .NET framework meant that Microsoft had to change the Visual Basic language.

Suddenly, Visual Basic programs created with versions 1.0 through 6.0 wouldn't work at all with the latest version of Visual Basic dubbed VB.NET without extensive modification. Even worse, the changes Microsoft made to the new VB.NET language made it nearly as difficult to learn as C#, another language that Microsoft developed based on the C and C++ programming languages.

With the Visual Basic language fractured and mutated from a friendly, easy-to-learn language to a convoluted, difficult-to-learn language, Microsoft effectively killed every advantage Visual Basic held for writing programs quickly and easily.

Then REALbasic appeared. REALbasic essentially took the same RAD programming paradigm that made Visual Basic popular and brought this programming tool to the Macintosh. For the first time, programming the Macintosh was suddenly easy, quick and fun.

REALbasic then went one step further and created versions capable of running on Windows, Linux and Mac OS X. Unlike Visual Basic, which limited you to writing Windows programs, REALbasic allowed you to write a program once and create three different versions that could run on Windows, Linux and Mac OS X with minimal modifications.

More importantly, REALbasic was so close to the original Visual Basic language that many Visual Basic programmers found they could take their own Visual Basic programs, recompile them under REALbasic, and essentially turn their previous Windows-only program into a Windows, Linux and Mac OS X program, tripling their potential market in an instant.

While Visual Basic still remains popular, its popularity is nowhere near its previous high and programmers are defecting from Visual Basic every day. In contrast, REALbasic continues to get better, and with the ability to run on the three most popular operating systems in the world, REALbasic is only gaining a larger, loyal following. For many programmers, their secret weapon for creating programs quickly is REALbasic.

If you're discovering REALbasic for the first time, get ready to experience a level of programming productivity that few other languages can match. Whether you're using Windows, Linux, or Mac OS X, you can create programs for yourself, for your job, or for the general public. Whether you want to program for fun, for a job, or for a career, you'll find that REALbasic finally makes programming available for the rest of us.

To grab a trial version of REALbasic, visit REAL Software (www.realsoftware.com). REALbasic runs on Windows, Linux and Mac OS X, so grab a copy for your computer and you can start

learning programming today.

Next week, "Getting to Know REALbasic".

In the early days, before Wally became an Internationally renowned comedian, computer book writer, and generally cool guy, Wally Wang used to hang around The Byte Buyer dangling participles with Jack Dunning and go to the gym to pump iron with Dan Gookin.

Wally is responsible for Microsoft Office 2007 for Dummies, Breaking Into Acting for Dummies, Beginning Programming All-in-One Reference for Dummies, and Mac All-in-One Reference for Dummies from www.dummies.com, as well as, Steal This Computer Book 4.0, Visual Basic Express 2005: Now Playing, and My New Mac from www.nostarch.com. He is also the co-author of Strategic Entrepreneurism from www.selectbooks.com.

Every Saturday morning from 9:00 am - 10:00 am in San Diego, you can hear Wally with fellow co-hosts Dane Henderson and Candace Lee, on the radio show CyberSports Today (www.cybersportstoday.com), which covers the video gaming industry on ESPN Radio 800 AM. Wally covers the military history side of the video game industry.

When not performing stand-up comedy or writing computer books, he likes to paper trade stocks with the video game Stock Reflex (www.plimus.com/jsp/download_trial.jsp?contractId=1722712&referrer=wwang).

Wally can be reached at wally@computoredge.com.

Send mail to ceeditor@computoredge.com with questions about editorial content.

Send mail to cwebmaster@computoredge.com with questions or comments about this Web site.

Copyright © 1997-2009 The Byte Buyer, Inc.

ComputerEdge Magazine, P.O. Box 83086, San Diego, CA 92138. (858) 573-0315

COMPUTOREDGE[®] ONLINE

www.computoredge.com

04/03/2009