

Creating Your Own Programs

MARK H. EBELL

So . . . you want to write your own programs for the Palm or Pocket PC. One of the thousands of programs written by professional programmers just won't do, eh? Your own custom database (Chapter 11) isn't powerful enough, hmmm? Well, aren't we special!

In this chapter, I'll survey some of the tools used to "roll your own" programs for the Palm and Pocket PC. No, you won't learn how to write software—my goal is to help you understand the strengths and weaknesses of each of your options. While there are dozens of minor languages, most are not well implemented or serve a particular niche. The programming environments discussed next are the most popular and useful.

What Is Programming?

Programming a computer is the process of creating a user interface (e.g., buttons, lists, and menus) and then writing a set of instructions that tell it in great detail what to do. For example, when a program starts, a set of instructions ("code") may tell the PDA where to find a database file and then load some initial data into a list for the user. When a button is pushed, another segment of code tells the program what to do.

Below is a very simple program for the Pocket PC that lets you choose a dose in mg/kg for amoxicillin, then calculates the proper dose for children of different weights. It is shown first in the programming environment, in this case embedded Visual Basic (Figure 18.1).

The developer chooses "controls" and draws them on a "form." Then, code is attached to the "Form_Load" event that adds doses to the drop-down box. Finally, a simple command figures out which dose was selected, calculates the appropriate dose for a child of this weight, and displays the answer. Some of this is pretty intuitive: the ".AddItem" method adds an item to a list. Some of it isn't: the "listIndex" tells me which item in the list was selected, and is 0 for 20mg/kg, 1 for 40mg/kg, and 2 for 10mg/kg.

It's appearance on a Pocket PC is shown in Figure 18.2.

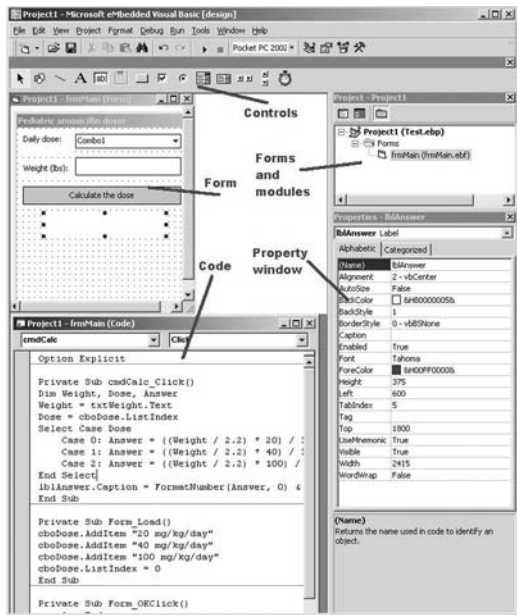


FIGURE 18.1. Embedded Visual Basic programming environment. (Reprinted with permission from Microsoft Corporation.)



FIGURE 18.2. Sample program created with embedded Visual Basic. (Reprinted with permission from Microsoft Corporation.)

Now that you are an expert programmer, let's consider some of the issues in choosing a programming environment.

What to Look for in a Programming Tool

There is a trade-off between power and ease of use in many products, just as there is between component stereos versus boomboxes and manual versus automatic transmissions. Programming tools are no exception. At one end of the spectrum are C++ compilers that require you to keep track of pointers and heaps and all sorts of arcane things. On the other hand, they create compact, very fast code that runs on a variety of devices. At the other end are programs like NSBasic and AppForge that provide a simpler interface but also create code that is slower and fatter.

Cost is another important issue—some tools are free, while some cost hundreds or even thousands of dollars. Ease of distribution is another. Although programs written in C++ can be distributed as a single file, those written with interpreted language require you to provide “run-time” files. These runtimes can conflict, leading to headaches for you and your users. Some runtimes even cost you money to distribute, a nonstarter for most developers. Table 18.1 summarizes the key characteristics of the programming tools that we discuss in this chapter.

TABLE 18.1. Comparison of programming tools for the Palm and Pocket PC.

Programming tool	Device	Cost	Core language	Compiled ^a	Runtime required
CodeWarrior www.metrowerks.com	Palm	\$499	C/C++	Yes	No
embedded Visual C www.microsoft.com/windowsmobile/information/ deprograms/default.mspy	Pocket PC	Free	C	Yes	No
embedded Visual Basic www.microsoft.com/windowsmobile/information/ deprograms/default.mspy	Pocket PC	Free	Basic	No	Yes
NSBasic/Palm www.nsbasic.com	Palm	\$149.95	Basic	No	Yes
NSBasic/PocketPC www.nsbasic.com	Pocket PC	\$149.95	Basic	No	Yes
.Net Compact Framework msdn.microsoft.com/vstudio/device/compact.asp	Pocket PC	\$1079 ^b	C# or Basic		No
AppForge www.appforge.com	Palm, Pocket PC, Symbian/Nokia	\$899 (\$129 for Palm only)	Basic	No	Yes ^c

^aCompiled languages are faster and more compact than interpreted.^b\$579 for upgrade from any other Microsoft programming language.^c\$10 fee for Pocket PC, free for Palm.

There are lots of options, as you can see, and quite a range in price. Let's look at each in a bit more detail.

CodeWarrior for Palm (Metrowerks)

CodeWarrior is an industrial-strength programming tool primarily aimed at professional developers that uses the C or C++ language. Like the .Net Framework (discussed below), it is not for the faint of heart and has a price to match. On the other hand, this is what they used to write the Palm operating system itself, if that gives you an idea of its power. For creating tight, fast, professional code for the Palm, CodeWarrior is the best choice. If you're a hobbyist or part-time developer, and especially if you don't know C, keep reading.

Embedded Visual C and Embedded Visual Basic (Microsoft)

These sister applications from Microsoft are a great deal—they're free! They don't require any other programming languages, and run in Windows 98, ME, 2000, NT, and XP. Note that the emulation features, which simulate a little Pocket PC on your screen for testing purposes, do not function in Windows 98 or ME.

Most professional software (i.e., the kind you have to pay for) has been created with one of these languages, the majority with embedded Visual C (eVC). eVC has all the strengths and weaknesses of any C compiler: terrific power, and a very steep learning curve. It is only recommended for those with 2 or 3 months to spare just getting up to speed.

Embedded Visual Basic (eVB), on the other hand, is easy to pick up for anyone who has done previous programming, even those simple Basic language programs in high school that made the computer say "hello world" when you pressed a certain key. This is especially true if you are familiar with its big brother Visual Basic for Windows. I showed you a very simple application created in eVB earlier in the chapter 3. It took all of 5 minutes to write and download to my Pocket PC. This is the strength of eVB: it's cheap, it's easy, and it's fast.

But (you knew this was coming) . . . it is also slow, a little buggy, and creates bloated installation packages. That simple program that I created would be about 3 kilobytes (KB) written in eVC, but would weigh in at well over 1000 KB written in eVB. Why? Its those pesky run-time files. While in theory they come preinstalled on every Pocket PC, you can't count on it, so you end up bundling a half dozen or more of these files with your tiny little program. This is less of an issue when your program is 3 or 4 MB (or 37 MB, as in our InfoRetriever software), but is a real disincentive for the hobby programmer who just wants to create a quick little program.

Programs written in eVB are slower than eVC programs, but comparable in speed to NSBasic and AppForge applications. There is a very active

user community that has developed work-arounds for many of the language's shortcomings, and several dozen add-in programming controls from third-party vendors. Visit www.devbuzz.com or www.pocketpcdn.com for more information, as well as Microsoft's own developer site (<http://www.microsoft.com/mobile/developer/default.asp>). You can easily and quickly build links to the address book and calendar, wireless networks, databases, and the Internet. You can build in a Web browser, manipulate XML documents, and much more.

.Net Compact Framework (Microsoft)

This is the future of programming, according to Microsoft. The .Net Framework includes a half-dozen programming languages for desktop computers (e.g., C++, Visual Basic, C#) and what it calls the "Compact Framework" (CF) that lets you create programs for Pocket PCs and other mobile devices that run the Windows CE operating system. The Compact Framework has just been released in its final form in mid-2003. Because Microsoft does not plan any further support for the eVB and eVC programming tools, most serious programmers will probably migrate to .Net over the next few years.

.Net CF promises to allow reuse of code from programs originally written for the desktop computer. Of course, although code can be reused, the user interface would still have to be designed from the ground up for mobile devices. They also promise that the language will create software that is faster and less prone to "dll hell," the bugs created when conflicting run-time libraries exist on the same device.

Don't even try to run .Net CF unless you have 256MB of RAM on your desktop computer and Windows 2000 or Windows XP, not to mention a very big hard drive. This is a serious programming environment for serious programmers, and is not well suited to the casual or hobbyist programmer. If you are reading this chapter, this probably isn't the programming tool for you.

NSBasic (NSBasic)

NSBasic comes in three flavors, one each for Palm, Windows CE (Pocket PC), and Newton. Newton? That was the grandpapa of handheld computers, and had a terrific operating system . . . unfortunately, it was too bulky and arrived a bit before its time. The Palm and Windows CE versions of NSBasic are now in version 3.0 and 4.1, respectively, making it a mature language. NSBasic is a great language for clinicians who want to create programs for their own use or to share with colleagues, but has enough power that it can be considered for commercial applications as well.

As with other interpreted languages, the major drawbacks are that it requires a run-time file (although this can be built into the program) and is slower than compiled programs. Strengths of NSBasic include ease of use,

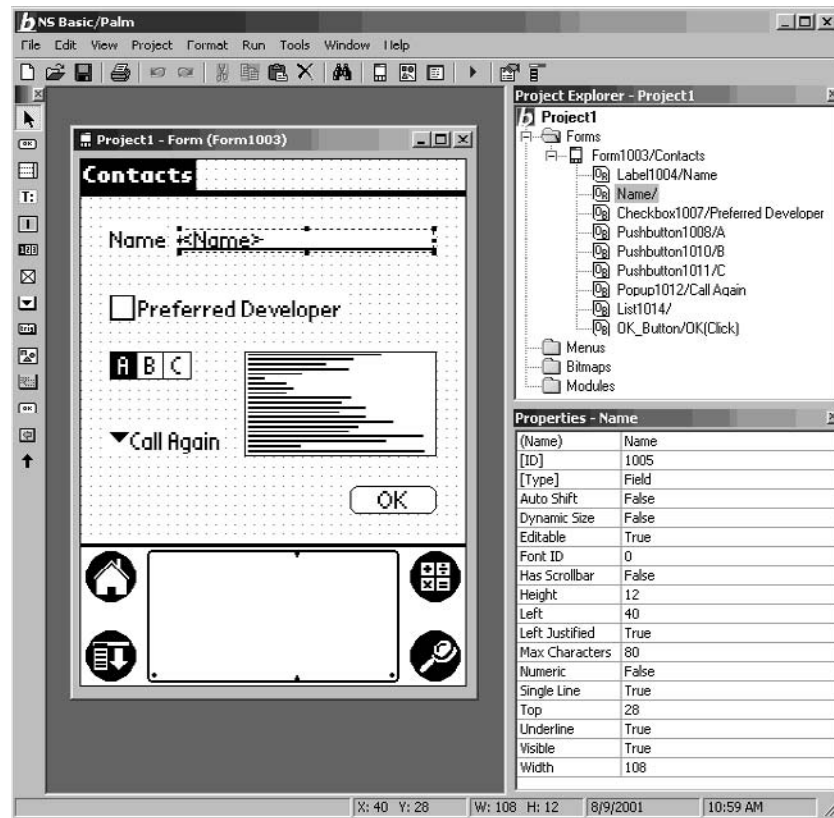


FIGURE 18.3. The NSBasic/Palm programming environment. (Reprinted with permission from NSBasic.)

low cost, and great technical support from the software's author (George Henne) and an enthusiastic community of users. Its chief competitor is AppForge. While AppForge is a bit slicker and has a few more features, NSBasic has the advantage of being much less expensive and not requiring that you own Visual Basic 6.0. The disadvantage is that it takes more effort to move or "port" code from Visual Basic 6.0 to NSBasic than to AppForge.

A screenshot of NSBasic/Palm is shown in Figure 18.3.

AppForge/MobileVB (AppForge)

AppForge, which has recently changed its name to "MobileVB," has several unique features that make it worth considering. If you are already familiar with Visual Basic 6.0, perhaps the most widely used programming language for desktop computers, then it is an easy transition to learn AppForge. It is also the only programming tool that lets you use a single code base for



FIGURE 18.4. Selecting a type of project with AppForge in Visual Basic 6.0. (Reprinted with permission from Microsoft Corporation.)



FIGURE 18.5. Selecting a device target within AppForge. (Reprinted with permission from AppForge.)

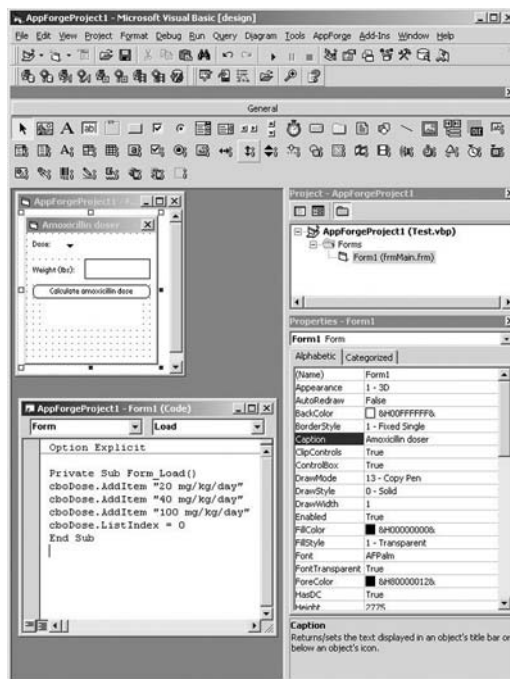


FIGURE 18.6. The AppForge/MobileVB programming environment, running within Visual Basic 6.0. (Reprinted with permission from AppForge and Microsoft Corporation.)

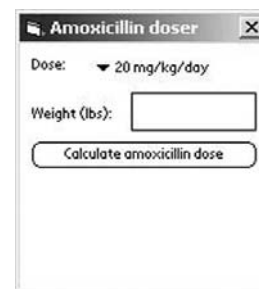


FIGURE 18.7. Example of a program created with AppForge, running in the “emulator” on the desktop computer. (Reprinted with permission from AppForge.)

TABLE 18.2. Resources for handheld device programmers.

Web sites

- Palm/CodeWarrior
 - MetroWerks (<http://www.metrowerks.com/MW/Develop/Desktop/PalmOS/Default.htm>)
 - Palm home page for developers (<http://www.palm.com/developers/>)
- eVB/eVC
 - DevBuzz (www.devbuzz.com)
 - Microsoft (<http://www.microsoft.com/mobile/developer/default.asp>)
- AppForge
 - AppForge home page (www.appforge.com)
- General Pocket PC
 - Pocket PC developer network (www.pocketpcdn.com)
- .Net Compact Framework
 - GotDotNet community (<http://www.gotdotnet.com/team/netcf/>)
 - Microsoft (<http://msdn.microsoft.com/vstudio/device/compactfx.asp>)
 - DevBuzz (www.devbuzz.com)

Books

- Wigley A. Microsoft .NET Compact Framework (core reference). Redmond Washington: Microsoft Press, 2002.
- rattan N. Pocket PC: Handheld PC Developer's Guide with Microsoft Embedded Visual Basic. Upper Saddle River, N.J.: Prentice Hall, 2002.
- Jamsa KA, Jamsa K. Instant Palm OS Applications. New York: McGraw-Hill Osborne Media, 2001.
- Foster LR. Palm OS Programming Bible, 2nd Ed. New York: Wiley, 2002.

creating programs for the Palm, Pocket PC, and even Nokia mobile phones that use the Symbian operating system.

After installing AppForge/MobileVB, you now have a new kind of project when you first launch VB 6.0 (Figure 18.4).

Select an AppForge project, and you are prompted to choose Palm OS or Pocket PC as your target device (Figure 18.5).

If you select Palm OS, this is what your programming environment looks like. Note the new controls in the toolbar and a new menu (Figure 18.6).

Each of the new controls has a little “i” in the lower-right corner that stands for “ingot,” which is what the AppForge calls their controls. You can't use the built-in VB controls, which remain in the toolbar but are not functional. In the screen above, I have quickly drawn a few controls. Pressing the right-hand arrow or selecting “Start” from the Run menu quickly compiles the program and runs it in a simulator windows, as shown in Figure 18.7.

AppForge/MobileVB is a comprehensive and easy-to-use tool for programming. Compared with NSBasic, its programs run at about the same speed, but it is more expensive, requires Visual Basic 6.0, and its run-time file is quite a bit larger (350 KB vs. 88 KB for NSBasic). On the other hand, I found that it was a more efficient programming tool, saving me time and

allowing me to reuse code from Visual Basic projects. If your only interest is in Palm programming, check out the Palm-only version for \$129.

Summary

Programming isn't just for teenagers, nerds, and bearded computer gurus. New tools like NSBasic and AppForge allow you to quickly and easily create programs for either Palms or Pocket PC's, and you can't beat the price of embedded Visual Basic and embedded Visual C for creating Pocket PC software. See Table 18.2 for a list of resources, then start cranking out the code!