nt main ()

Loosely speaking, the world of computer software can be divided into two groups: end-users and programmers. End-users are the people who utilize software for work and for play — they don't necessarily understand the intricacies of how their software (and yes, operating systems are software) works, they only care that the software functions (or doesn't). Programmers, on the other hand, are the people who develop the software everyone uses.

There's an aura of mystique surrounding programmers. Some end-users look on programmers the way mere mortals would look on demi-gods, with a mixture of awe, dread and confusion. To use an application is simple (or at least it should be) — to develop one must be near to impossibly difficult.

Actually, it doesn't have to be. It all depends on the tools you use. In this article, we'll be looking at three development tools from the standpoint of the novice programmer, someone who is interested in beginning the journey from end-user to developer.

o What Is Programming Anyway?

Developing a program for a Graphic User Interface like the Mac OS is essentially a three-part process. First you create visual resources, all the things the user sees: windows, menus, pictures, text, buttons, etc. These are the graphical parts of the GUI. Second, you write the source code to utilize those resources. Writing the code is the real meat of programming. This involves learning to use a specialized computer language, which can be as complicated as learning any foreign language. Finally, you compile and debug the application, which connects the visual resources to the code you've written and eliminates any errors that may appear.

In learning to program, you'll be using a development environment (essentially, software that creates software). Each of these development environments makes use of different computer languages. The remainder of this article will be looking at three of the leading development environments for the Macintosh.

Editor's Note: If you're hesitant to get into programming, don't be! Among the standard perks (you know, the money, the babes, the fame, the fast cars), programmers get to drink as much Cherry Coke as they want!

odeWarrior Pro

The most popular development environment for the Mac is CodeWarrior Pro (now at version 4) from Metrowerks. It has become the de facto standard of Macintosh development tools: most of the applications you use every day were probably developed in CodeWarrior. Creating software using CodeWarrior also involves using a resource editor like ResEdit or Resorcerer (see next month for a review of Resorcerer) to create the application's visual resources, then writing the source code using a high-level language like C, C++, Pascal, or Java. C and C++ are the most common languages for developing Mac applications, and at first, they can look quite formidable and confusing, as you can see here:

```
void   InitializeToolbox (void)

    // These eight calls to Toolbox functions initialize Toolbox
    // managers.
    InitGraf (&qd.thePort);
    InitFonts ();
    InitWindows ();
    InitMenus ();
    TEInit ();
    InitDialogs (0L);
    FlushEvents (everyEvent, 0);
    InitCursor ();
```

The above is just a very small portion of a program, used to initialize parts of the Macintosh "Toolbox." The Toolbox is the part of the Operating System which generally resides in the ROM (Read-Only Memory). It controls the routines governing how windows are displayed, how graphics and text are drawn on the screen, and other low-level management functions. The book

Macintosh Programming Techniques by Dan Parks Sydow is an excellent introduction to working with the Mac Toolbox. Alternatively, some developers also cherish Inside the Macintosh Toolbox. You can find these books (and lots of other cool stuff) at the Developer Depot. Drop by http://www.devdepot.com/ any time, any day.

As you can see from the code snippet, in order to develop in CodeWarrior, it is first necessary to learn one of the languages it supports. Many books about C and C++ (as well as Pascal and Java) are readily available, and the CodeWarrior CD-ROM includes electronic documentation in PDF format, including whole books like Dave Mark's Learn C On The Macintosh and Jim Trudeau's Programming Starter Kit for Macintosh.

Once you're more familiar with this kind of programming, application frameworks such as Metrowerks' PowerPlant and Apple's MacApp will simplify and speed up the development process. These application frameworks are software libraries which serve as templates. They handle the implementation of the behavior of a generic application, leaving you free to concentrate on the aspects of the application which make it unique. Using these application frameworks is conducive to rapid application development (RAD).

CodeWarrior Pro is fairly expensive — around $425 (though an educational discount brings the price down to a thrifty $119). If you're not sure whether programming is for you, a more reasonably-priced alternative is Metrowerks' CodeWarrior Discover Programming Edition (about $70). Although the Discover Programming Edition will not make PowerPC applications, and there are licensing limitations about distributing the work you produce with it, this is a good place for the newcomer to begin.

The Metrowerks website can be found at http://www.metrowerks.com/ .

EALbasic

Since C and C++ are so complex and their syntax so convoluted, RealSoftware's REALbasic (formerly CrossBasic) comes as a great relief. Developed by Andrew Barry as a Macintosh answer to Microsoft's Visual Basic (which is only available for Windows), REALbasic offers a truly integrated, visual development environment.

uilding your application's interface is as simple as dragging and dropping. Choosing from a palette of standard Macintosh controls — push buttons, progress bars, radio buttons, and so forth — you simply drag the controls where you want them and change their properties to suit your application's individual requirements. Then you use an object-oriented version of the BASIC programming language (Beginner's All-purpose Symbolic Instruction Code). It is object-oriented because you write code as discrete individual objects attached to the interface elements they control. REALbasic supports all of the commands of traditional BASIC.

REALbasic offers excellent support for sprite-based animation (in which individual elements move independently and react to user input), TCP/IP networking, QuickTime media, music, and even contextual menus! The environment is extensible with third-party plug-ins, and it can utilize external functions and external commands (XFCNs and XCMDs) which were developed for HyperCard. XFCNs and XCMDs are small pieces of software written in a high-level language like C or Pascal and compiled so that they can add new functionality over and above what was originally built into the environment.

While easier to learn than C, REALbasic's programming language can still be quite confusing at first. Consider this example, in which the File menu item Open is handled:

```
Function Action As Boolean
   dim f as folderitem
   dim i as textinputstream
   dim s as string
   dim w as window1
   f=getopenFolderItem("any")
   if f <> nil then
      i=f.openAsTextFile
      while not i.eof
      s=s+i.readline
      wend
      i.close
      w=new window1
      w.editfield1.text=s
      w.document=f
      w.title=f.name
   end if
End Function
```

In REALbasic, you define variables (containers which hold different values) using the "dim" statement. "Dim" is short for dimension, and it means making space for the variable. Variables can be defined as a variety of data types, including strings (series of characters), integers (whole numbers), and Booleans (which means either true or false).

If you're willing to surmount its learning curve, REALbasic's power and flexibility make it an exciting solution for software developers of all skill levels.

REALbasic 1.0 only costs $100 (or $60 at academic prices). For such a robust development environment which is constantly being enhanced and improved, this is a REAL bargain.

REALbasic is available from Real Software's website at http://www.realsoftware.com/ .

uperCard

My development environment of choice is SuperCard from IncWell DMG. SuperCard is similar to Apple's own HyperCard, but in addition to native support for color, SuperCard offers a host of improvements over HC.

SuperCard uses the "card and stack" metaphor — think of a SuperCard project as a stack of 3 x 5 cards, each of which contains different information. SuperCard projects can contain text, pictures, sounds, animations, music, and QuickTime movies. Like REALBasic, SuperCard has palettes of standard interface elements (buttons, text fields, popup menus, etc.) which you can easily add to your project, and you can utilize HyperCard XCMDs and XFCNs.

Navigating through the stack of cards and adding other kinds of interactivity is done by means of the SuperTalk scripting language. SuperTalk is easy to learn because it's so much like English. For instance, let's say the user clicks a button. When the mouse button is let up (indicating that the click has been completed), the "mouseUp" message is sent to the button, and a handler attached to the button acts on it. Here's what it looks like:

```
on mouseUp
  go to next card
end mouseUp
```

That's pretty easy to understand: pushing this button causes you to go to the next card in the stack.

Another of SuperCard's strengths is a built-in utility called ClickScript.

ClickScript enables novice users to add interactivity to their projects without actually learning the SuperTalk language beforehand. ClickScript presents a series of options and actions to choose from, then generates the SuperTalk scripts automatically. Using ClickScript is a good way to learn the language while gaining hands-on experience with SuperCard.

I've found SuperCard to be an excellent solution for creating multimedia applications, presenting information in a visually interesting manner. I've used it to develop two trivia games, as well as a version of the I Ching. However, SC does have some limitations. It's not ideal for sprite-based animation, so you probably wouldn't want to use it to develop a shoot-em-up arcade game. Additionally, SuperCard applications are a bit on the slow side when compared to applications developed in C or with REALbasic. Also, there's a file-size overhead when making standalone applications with SC; even small projects become considerably larger because the standalones need to have the SuperCard player application built directly into them.

Despite its limitations, SuperCard is probably the easiest way to develop applications quickly, and it has a very shallow learning curve, making it an excellent choice for the beginner. It's an outstanding RAD tool and a great way to learn the fundamentals of programming.

The newest version, 3.5, is priced at $145, with discounts for upgrading from earlier versions.

SuperCard is available from IncWell DMG Ltd. on the web at http://www.incwell.com/ .

et Info

Apple provides a wealth of programming resources through their developer website at http://www.apple.com/developer/ . There you'll find information about development tools (links to compilers, editors, and debuggers), documentation (including the entire series of Inside Macintosh books in downloadable PDF format), sample code, and software development kits (SDKs). You'll also find information about joining the Apple Developer Connection support program, which offers a variety of ways to receive technical and business resources from Apple: online, on monthly CD-ROMs, and with live, human tech support.

AllDone = true;   // The Conclusion

Obviously, this is not a complete look at every aspect of programming on the Macintosh — that would fill many volumes. Consider this a teaser, a preview of what's in store. The journey from end-user to programmer can be a long and challenging one, and as someone who is just beginning that journey myself, I hope this article can encourage others in taking those first tentative steps on the road to creating their own software.

Brian Kelley
brian@applewizards.net