

According to Hoyle...

jhoyle@maccompanion.com *macCompanion* March 2006

Cross-Platform Software Development from a Macintosh Perspective: AMPC: *Axiomatic Multi-Platform C*

by Jonathan Hoyle

Last year, we began our series on cross-platform software development for the Macintosh. Over these past many months we have examined many cross-platform approaches, from C++ frameworks to complete development environments. For those who are just joining us, below is a list of what we have just covered:

- **Intro:** http://www.maccompanion.com/archives/september2005/Columns/According_to_Hoyle_1.htm
- **Qt:** <http://www.maccompanion.com/archives/october2005/Columns/AccordingtoHoyle.htm>
- **wxWidgets:** <http://www.maccompanion.com/archives/november2005/Columns/AccordingtoHoyle.htm>
- **CPLAT:** <http://www.maccompanion.com/archives/december2005/Columns/AccordingtoHoyle.htm>
- **REALbasic:** <http://www.maccompanion.com/archives/january2006/Columns/AccordingtoHoyle.htm>
- **Runtime Revolution:**
<http://www.maccompanion.com/archives/february2006/Columns/AccordingtoHoyle.htm>

This month we examine *Axiomatic Multi-Platform C* (AMPC), an ANSI C compiler by *Axiomatic Solutions* which creates applications which can run in the Java runtime environment [<http://www.axiomsol.com>]. AMPC is available for Windows, Linux and Mac OS X.

But aren't C and Java were separate languages?

There are. Sort of. Unfortunately, the word "*Java*" can be used in one of two ways: as the programming language or as the resultant bytecode it's compiled into. Java the programming language, like C++ or Basic, is the front-end language the developer writes in, which the compiler turns into object code. Java the bytecode is the back-end which runs in the runtime environment, which is available on many different platforms. Typically, the Java programming language gets compiled into Java bytecode, so the distinction is typically not important. However, AMPC uses C as the front end programming language while preserving the Java bytecode back-end. In this way, AMPC finally gives C programmers the ability to create cross-platform applications right out of the box.

Is this a true ANSI C programming environment? Do it really create true Java applications?

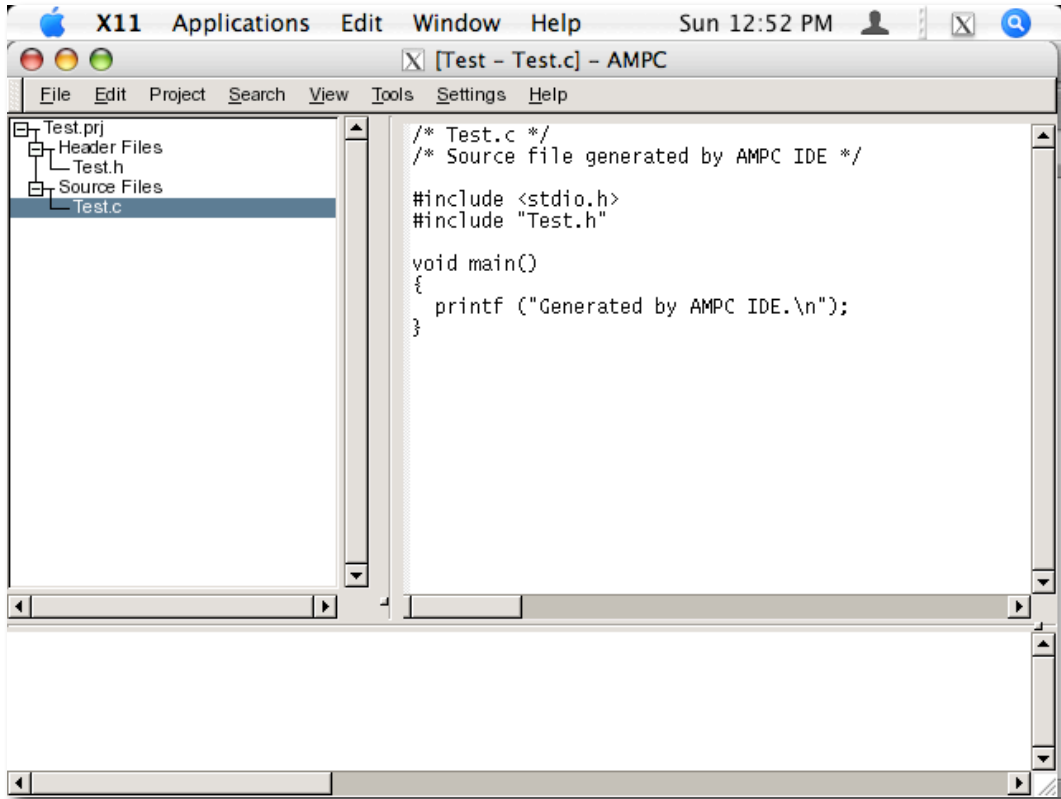
Surprisingly, yes to both. It is quite an impressive technology from that perspective. The Java runtime environment is fully object oriented with built-in garbage collection and Java-built applications are written to take advantage of it. On the other hand, the C programming language (AMPC does not do C++) is a procedural-based language using pointers and has no ability to access objects. Yet within AMPC, they coexist marvelously. Moreover, AMPC allows you to call Java functions within your C code as well, giving you even more flexibility.

C pointers are done particularly well. Remember that in a traditional C compiler, pointers are references to raw memory locations. In the Java Runtime Environment, raw memory locations are not available. To solve this problem, AMPC implements pointers as opaque handles referring to Java objects; all of this happens under the hood and is transparent to the user. There are however some unusual aspects of this C compiler. For example, all scalar types: short, int, long, float, double, and even char, are each the same size: 32-bits. AMPC does provide its own proprietary DOUBLE (all caps) data type for 64-bit floating point numbers (implemented as a struct of two 32-bit values). Also, certain language features are not yet fully implemented, such as bitfields and limited goto's.

Creating Applications with AMPC

As you might expect, there isn't any built-in facility to make native calls to the Mac OS X API. For a cross-platform development environment, that isn't terribly surprising. AMPC does provide C-equivalents to the Java Swing API, ensuring that the applications you build with it are truly cross-platform. Standard console applications can also be compiled, using traditional `scanf()` and `printf()`. C source and header files are handled identically as they would in any other ANSI C compiler, except AMPC projects are compiled into `.jar` files (Java equivalents to libraries and executables) rather than traditional applications. `.jar` files can be launched in Mac OS X simply by double-clicking them.

For those interested in obtaining more technical information about AMPC, download the user manual at: http://www.axiomsol.com/hedesu/fm/download_file.php?id=180. There is a free demo that can be downloaded as well, but strangely, only the Windows version is available from their web site. I emailed *Axiomatic Solutions*, and they told me that to obtain the Mac demo, I had to retrieve it from <http://www.download.com>.



The Macintosh User Experience

Unfortunately, this is where AMPC comes up a bit short. Unlike traditional Mac OS X applications, AMPC runs inside the X11 runtime environment. X11 is optional and does not come as part of the default installation of Mac OS X. Therefore, to run AMPC, you will have to break out your Mac OS X Install DVD and install X11 first (AMPC requires 10.4 Tiger).

After the installations of X11 and AMPC, its poor Macintosh user experience first presents itself with a double menu bar: the X11 menu bar where the application menu should be, and the application menu attached to the window. When launched, AMPC sizes itself to fill the entire screen; although this may be appropriate behavior on some operating systems, it is not so on the Mac. The first great difficulty in usability came in attempting to resize this window. The window's lower right corner thumb did not easily respond to my mouse. After several attempts, I was finally able to resize the window, but once I let go of the mouse, the window's title bar is suddenly moved beneath the X11 menu bar, preventing any further attempt to access it. I had no further choice than to quit and relaunch the application.

Accessing the application menu and opening files behaved somewhat as expected. The file picker however is not from Navigation Services found with native Mac OS X application, but rather a Java Swing picker which looked awkward. A Unix directory listing is displayed, including directories such as "." and "..". I then clicked a little too far to the left of the File menu and unintentionally tore this menu off.

Conclusion

AMPC is priced well at only \$99. The technology is quite intriguing, and it opens up a new realm for creating cross-platform applications. Unfortunately, the environment (particularly the Mac version) is still relatively immature. Sadly, *Axiomatic Solutions* views their Mac OS X target is merely another version of their Linux target. Even some of the most basic usability tests seem to be lacking, as it is painful to use for more than just a few minutes at a time. AMPC shows a great deal of promise, and I hope that a future version will be forthcoming which is more Macintosh friendly. Until then, as much as I'd like to, I simply cannot recommend AMPC for Mac users at this time.