

According to Hoyle...

Cross-Platform Software Development from a Macintosh Perspective: Java compilers

by Jonathan Hoyle

[jhoyle at maccompanion.com](http://www.maccompanion.com)

Last year, we began our series on cross-platform software development for the Macintosh, covering a number of development environments and frameworks:

- **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>
- **AMPC:** <http://www.maccompanion.com/archives/march2006/Columns/AccordingtoHoyle.htm>

In this month's column, we review the state of Java compilers for the Macintosh developer.

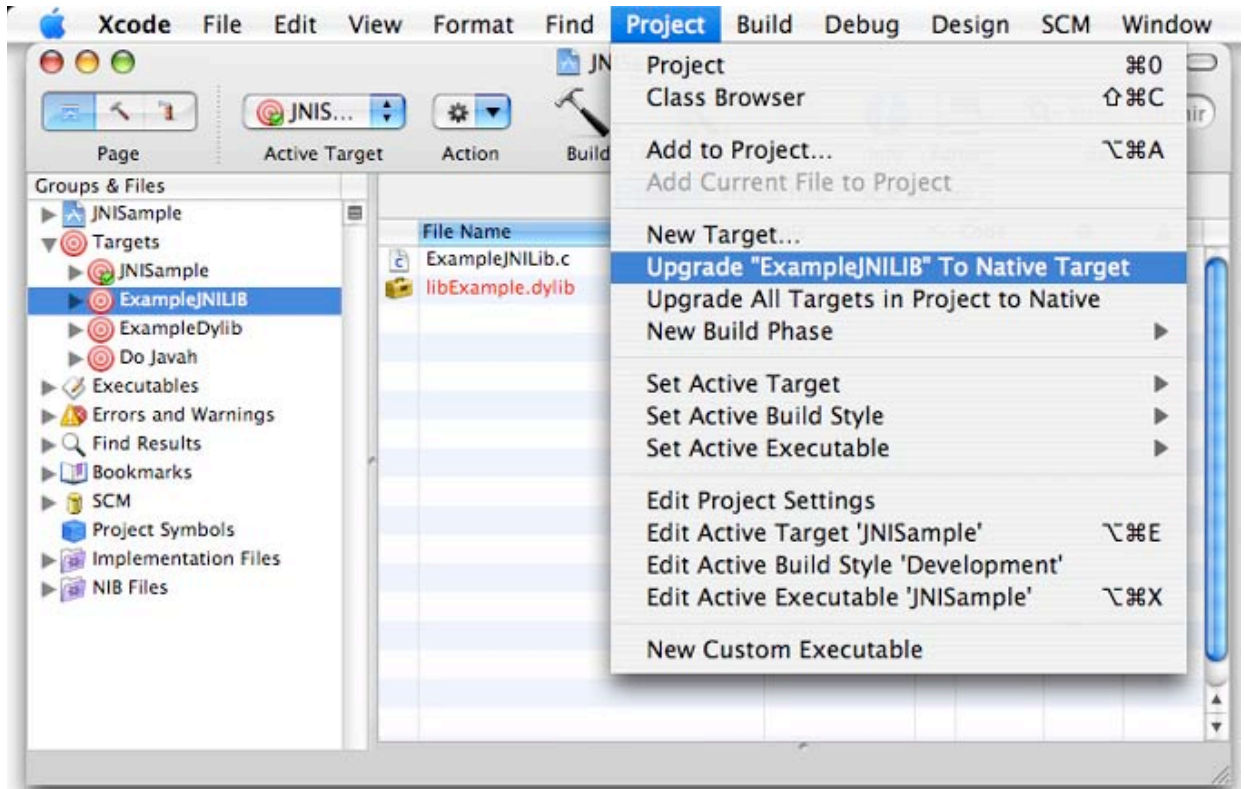
Java: The Preeminent Cross-Platform Development Platform

Unless you have spent the last 10 years as a prisoner of Azkaban, you have no doubt heard of Java. Developed by Sun Microsystems to be an object oriented language loosely based upon C++, Java has quickly grown to be one of the most popular programming languages. With built-in garbage collection, Java sheds C++'s pointer weaknesses of memory leaks and access errors. Most importantly, the designers created the Java Runtime Environment (JRE), allowing Java applications to run on any platform that supports the JRE.

It is important to remember that there are two sides to Java: the front end Java programming language, and the back-end Java byte code that runs in the JRE. Although they are typically found together, there are commercial packages in which they are separated. For example, Microsoft's *Visual J#* development environment compiles the Java programming language into the .NET intermediate language instead of Java byte code; a reverse example is *Axiomatic Multi-Platform C* (reviewed last month) which compiles the C programming language into Java byte code. A more insidious example is the now defunct *Visual J++*, Microsoft's Java compiler, which failed to be fully cross-platform (many apps it created ran only on Windows). In this article, we shall review only fully compliant Java development environments, both front end and back end.

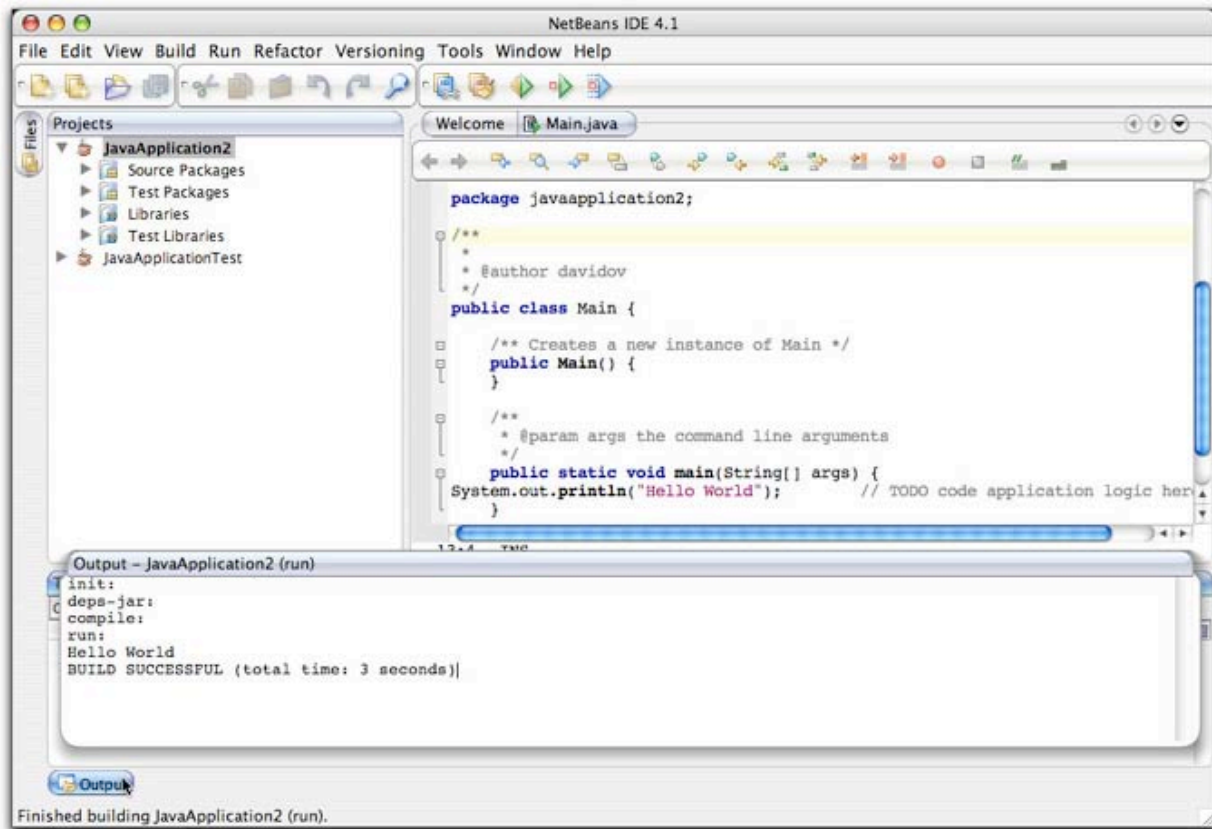
Free Java Compilers

Not all Java compilers are created equal, at least not from a Mac OS X perspective. Sun provides a free Java compiler called *javac*, and IBM provides a highly optimized one named *jikes*. Each of these, however, is a command line interface compiler and thus requires you to run it inside the *Terminal* application. Although they are both good compilers, console applications do not provide an acceptable Macintosh user experience. Fortunately, Apple comes to the rescue here with *Xcode*, the free IDE that comes with Mac OS X. *Xcode* can wrap either *javac* or *jikes*, allowing you to build Java applications <http://www.apple.com/xcode>. With its code completion and enhanced debugging capabilities, *Xcode* is an excellent platform for Java development on the Mac. However, because *Xcode* itself is Mac-only, the only way to debug Windows or Linux is through remote debugging. This is a less than optimal way to develop cross-platform code.



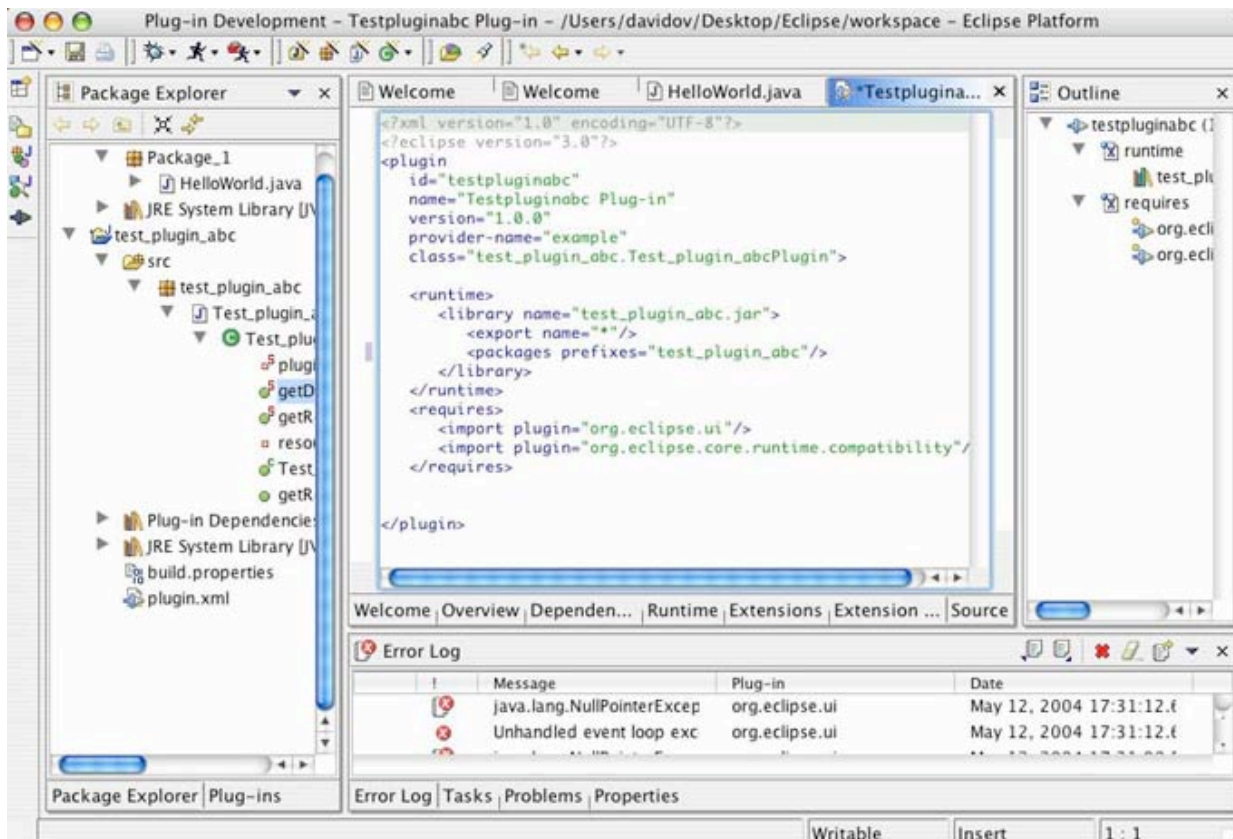
Xcode screenshot

There are two other free (and in fact Open Source) Java IDE's, which are themselves written in Java. One is *NetBeans* <http://www.netbeans.org>, a very popular open source Java programming environment. *NetBeans* has a built-in GUI builder, code completion, code folding and much more. Its feature set is rich and far more powerful than that found in *Xcode*. Unfortunately, its GUI is still a bit immature from a Mac perspective. The most obvious sign of this is its application menu bar being attached to the window, as opposed to being at the top of the desktop. This is a common problem found in applications written by Java developers who think of their Mac clients as an afterthought.



NetBeans screenshot

A better choice, and the one recommended for a free compiler, is *Eclipse* <http://www.eclipse.org>. *Eclipse* has most of the features that *NetBeans* has plus a cleaner Mac OS X user experience. *Eclipse* also has a great feature called *code refactoring*, the ability to make global changes to your code when changing the name of a method, variable or even modifying a function parameter list. Although *NetBeans* has recently added this to its feature set, code refactoring remains more mature in *Eclipse*. Furthermore, the user interface, besides being more Mac friendly, is also a lot more intuitive and easy to navigate. It has also begun to capture a huge chunk of the Java marketshare. By all accounts, *Eclipse* is the leader in the freeware Java arena.

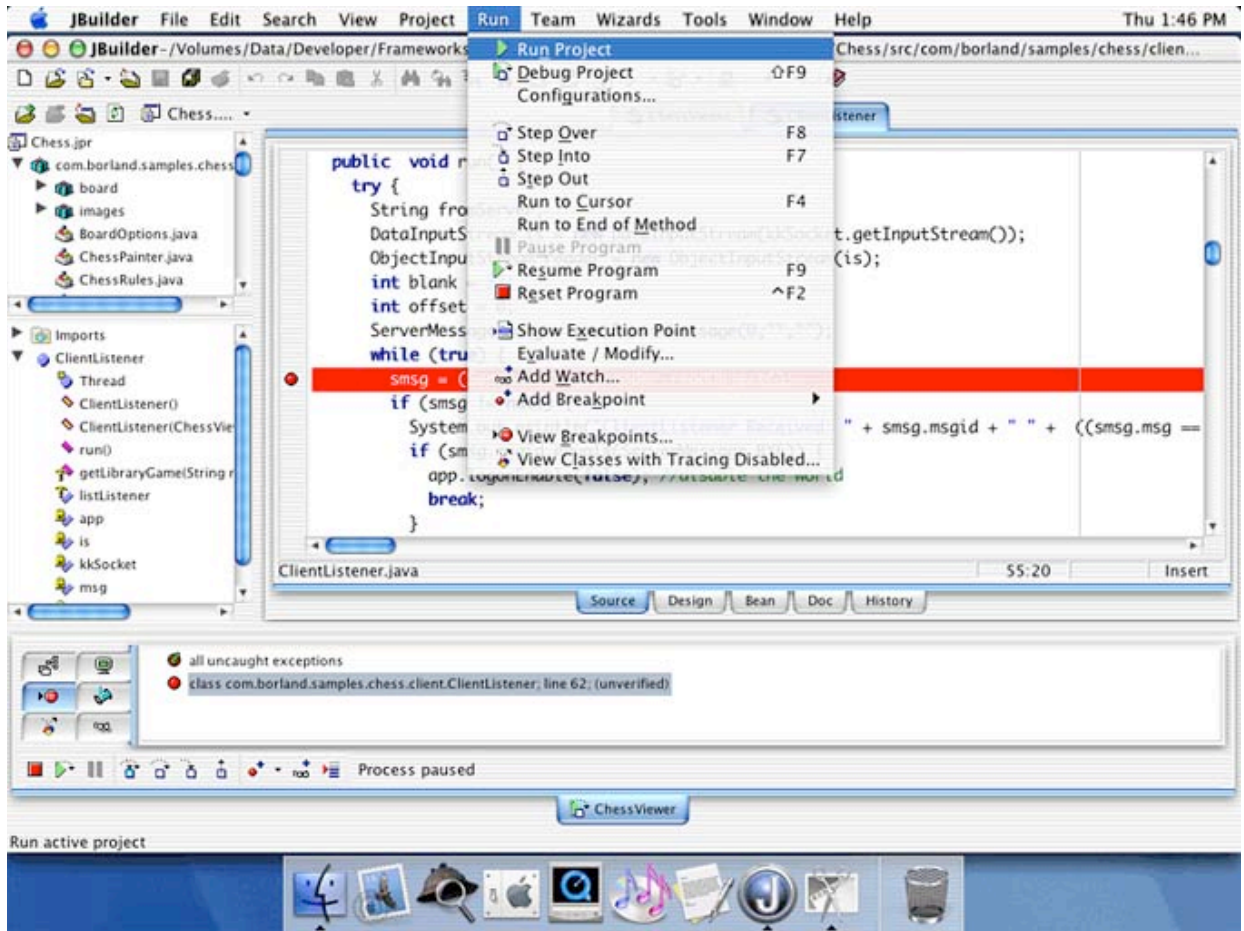


Eclipse screenshot

Commercial Java Compilers

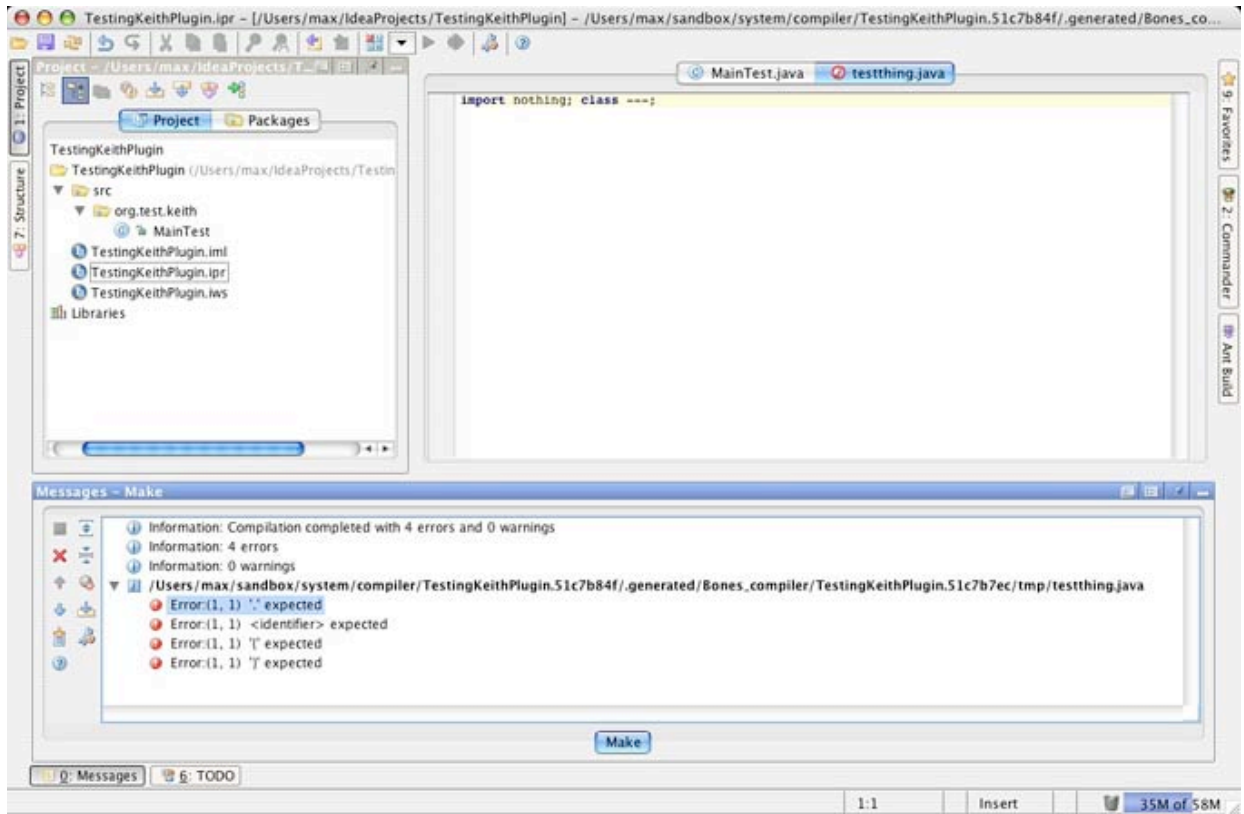
With the number of high quality freeware options for Java on the Mac, the need for a commercial Java package may be a bit limited. With CodeWarrior dropping out of the Mac and Windows development arena, coupled with Apple's big push to their own free tools, there are essentially two compilers left standing in the commercial world for the Mac: *JBuilder* by Borland <http://www.borland.com> and *IDEA* by JetBrains <http://www.jetbrains.com>. Each retails for \$499.

Shortly following the release of Mac OS X in 2001, Borland returned to the Mac development community with *JBuilder*, after an absence of nearly 15 years. Borland won a large number of awards for their *JBuilder* product, particularly for its ease of development. Supporting all of the latest standards, *JBuilder* quickly became the dominant IDE for Java, both on Windows and Mac OS X. Borland began making appearances at Apple's Worldwide Developer's Conference and quickly made inroads into the Macintosh. However, the tide began to turn in 2003 and 2004 when the popular (and free) *Eclipse* began to erode *JBuilder's* marketshare. About a year ago, Borland finally gave into the competition by announcing that its future roadmap will include integrating *JBuilder* with *Eclipse*: http://www.theserverside.com/news/thread.tss?thread_id=34246. As of this writing, *JBuilder 2006's* system requirements no longer include the Macintosh.



JBuilder screenshot

For the best Java compiler available today, the hands-down winner is *IDEA*. *IDEA* is similar to *Eclipse* from a feature checklist perspective, but *IDEA* does it more easily and more intuitively. Despite how nice *Eclipse* is, it appears clunky when compared with *IDEA*. *IDEA* takes fewer keystrokes to do most of the same tasks, and the workflow appears smoother. Its interface is more easily discoverable as well. Its GUI builder is also very intuitive and very powerful. The benefits of *IDEA* are harder to describe than they are to show, so I would encourage anyone interested should download the demo from their web site at: <http://www.jetbrains.com/idea/download>.



IDEA screenshot

Conclusion

Each of the packages discussed in this article are very good Java compilers, so from that perspective, you really can't go wrong. However, *Eclipse* has taken on the Java world by storm and is likely to come out the clear winner in marketshare. It is one of the best freeware IDE's ever made available, and is better than most commercial ones. The only product better is *IDEA*, which excels above and beyond and Java IDE before it. *IDEA* is the best of the best and is this author's recommendation for Java development.