

WHAT IS JAVA?

Intro FAQ

by Jason English



It all started with a blunt letter.

In 1990, Sun Microsystems software engineer Patrick Naughton was fed up with trying to support the hundreds of different combinations of software APIs used within the company. When he told CEO and friend Scott McNealy of his plans to accept a job offer from NeXT, McNealy didn't take the news sitting down. He asked Naughton to create a list of his complaints and to suggest a solution "as if you were God."



When Naughton created his list, he didn't pull any punches about Sun's shortcomings. Naughton said the NeWS software architecture the company was working on should be scrapped, and he suggested that of the more than one hundred people working in the Window Systems Group at that time, most of them wouldn't be needed if Sun straightened out the technical mess. After Naughton sent the letter to McNealy, he figured it would be ignored. "Why should I care?" he asked himself. "I'm leaving anyway."

Much to Naughton's surprise, the letter did make a difference. Quietly, it was e-mailed to many of Sun's top software engineers. Naughton's e-mail box was flooded with messages from colleagues who agreed with his assessment of the company's situation. Bill Joy, a Sun founder, and James Gosling, Naughton's mentor at Sun, supported his views and added fuel to the fire by raising many of the same concerns to other senior executives.

The day Naughton was to leave for NeXT, Sun made him a counter offer. The company would create a team of top software developers and free them to do whatever they wanted. The only expected deliverable: make something cool.

The team of six, codenamed Green, went into a self-imposed exile, very much like the scientists on the Manhattan Project. The team stocked the refrigerator with Cokes and Dove bars; discussed what they liked and didn't like about the technologies that were out on the market; and took apart countless electronic devices, such as Nintendo Game Boys, TV set-top boxes and remote controls.

The reason for this free-form exploration of Nintendos and other consumer electronic devices was to find a way for the appliances to talk to each other. The team discovered early on that electronic devices such as VCR's, laser disc players, and stereos were all made with different CPU's. Thus if a manufacturer wanted to add functions or features to a TV or VCR, they were stuck because they were limited by what the hardware and its wired-in programming would allow them to do. This, coupled with the fact that the chips used by many of these devices were limited

in program space, suggested a fresh approach to software programming that might be a key to enabling innovation in this product space.

The team's efforts kicked off the development of a new object-oriented programming language that Gosling called Oak, after the tree outside his window. Loosely based on C++, the language was stripped down to a bare minimum in order to be compatible with the limited space the chips in handheld devices would offer, and was designed to allow programmers to more easily support dynamic, changeable hardware.

As work on Oak continued, the Green team conducted extensive research into how and why people were attracted to certain video games and how they interacted with various kinds of electronic equipment. After collecting their research data, the team developed a handheld, remote-control-like device with a tiny visual interface. The device, dubbed "*7", featured an animated character named "Duke" who helped guide users through the easy-to-use, image rich, graphical interface remote control. Central to the design of the *7 was the conviction that the interface must be engaging and fun to use, and that the device itself must be a small, personal artifact. "Duke," created by Joe Palrang, would go on to become Java's mascot.

Sun turned the Green team into a wholly owned company called First Person. The new Operating Company had an interesting concept but still no idea what to do with it. After struggling to come up with a marketable idea, the company decided to pursue the interactive television market that seemed to be emerging.

A deal with Time-Warner to create set-top boxes fell through at the last minute. Another potential deal with 3DO was scrapped when that company's chief executive wanted exclusive rights to the technology. Thus First Person's foray into creating set-top boxes for video-on-demand fizzled.

The company's fortune's changed in 1993 when the National Center for Supercomputing Applications introduced Mosaic, and the World Wide Web was born. More web technology soon followed, and the Internet, formerly a home only to computer scientists and educators, began to bustle with traffic.

In early 1994 the First Person team recommended focusing its limited resources on a software system for online multimedia. Bill Joy took that initiative further by positioning Oak as a "language based operating system" and took up Naughton's suggestion to give it away in source form on the Internet. The Oak language itself became the product, instead of part of a device. Arthur van Hoff wrote an Oak compiler entirely in Oak instead of in C. Naughton and Jonathan Payne built an Oak-ready browser called "WebRunner." The first applet -- Duke waving back at his parents over the Internet -- was born.

Sun backed the decision to give the language away, but not before renaming it Java. Much has been made of the now famous name but consider the fact that it could have been called Neon, Lyric, Pepper or Silk.

With Java in the hands of the Internet community at large, all that was needed was a way to run Java applets. "WebRunner" was renamed the HotJava browser because of a trademark conflict. Then, Netscape began supporting Java. Now millions are Java-ready, and Duke has never looked back.

So what exactly is Java?

It is commonly thought of as a way to make Web pages sexy -- incorporating stock tickers, sound or video into Web pages. It has evolved into much more. It is becoming known as a computing platform -- the base upon which software developers can build applications. Developers can build a variety of applications using Java -- traditional spreadsheets and word processors in addition to mission critical applications used by the biggest companies: accounting, asset management, databases, human resources and sales.

Java applications, or applets, are different from ordinary applications in that they reside on the network in centralized servers. The network delivers the applet to your system when you request them. For example, let's say that you want to check your personal financial portfolio. You'd dial in to your financial institution and use your Web browser to log into the bank's system. The portfolio data will be shipped to you along with the applet needed to view it. Let's assume that you're considering moving your money from one account to another. No need to perform a series of cut-and-paste exercises. The system will also send you an applet that will allow you to change the rate of interest and length of investment to perform a series of "what-if" scenarios.

From the corporations' point-of-view, Java will simplify the creation and deployment of applications thus saving money. Applications created in Java can be deployed without modification to any computing platform, thus saving the costs associated with developing software for multiple platforms. And because the applications are stored on centralized servers, there is no longer a need to have people insert disks or ship CD's to update software.

So what will the future hold for companies and their use of Java? Only time will tell, but one thing is certain -- it's unlikely that letters such as the one written by Patrick Naughton complaining about multiple and incompatible software APIs will ever need to be sent again.