The History and Future of the Java Programming Language | @DevOpsSummit #Java #DevOps

The History and Future of the Java Programming Language
By Omed Habib

As the internet's renowned programming language, Java has had a profound impact on how people navigate the digital world. Much of what users expect in terms of performance from their devices that access the internet has been set by Java functionality. You don't have to be a developer, however, to recognize its influence.

The story of Java goes back more than two decades and has evolved along with the digital transformation of the world. As consumer and business demands on scalability increases, Java is forced to grow and adapt in order to stay relevant. Stakeholders are approaching their work armed with a primer on Java's history, current use, and future direction.

The History of Java: A Timeline
Early Development
Java is the brainchild of Java pioneer James Gosling, who traces Java's core idea of, "Write Once, Run Anywhere" back to work he did in graduate school.

After spending time at IBM, Gosling joined Sun Microsystems in 1984. In 1991, Gosling partnered with Sun colleagues, Michael Sheridan and Patrick Naughton on Project Green, to develop new technology for programming next-generation smart appliances.

Gosling, Naughton, and Sheridan set out to develop the project based on certain rules. They were specifically tied to performance, security, and functionality. Those rules were that Java must be:

1. Secure and robust
2. High performance
3. Portable and architecture-neutral, which means it can run on any combination of software and hardware
4. Threaded, interpreted, and dynamic
5. Object-oriented

Over time, the team added features and refinements that extended the heirloom of C++ and C, resulting in a new language called Oak, named after a tree outside Gosling's office.

After efforts to use Oak for interactive television failed to materialize, the technology was re-targeted for the world wide web. The team also began working on a web browser as a demonstration platform.

Because of a trademark conflict, Oak was renamed, Java, and in 1995, Java 1.0a2, along with the browser, name HotJava, was released.

Developer Reception
Java was well-received by developers in the software community, in particular because it was created based on the "Write Once, Run Anywhere" (WORA) philosophy. This flexibility is rooted in Java's Bytecode compilation capabilities, which bypass the potential barrier of different system infrastructure. Java was a unique programming language, because it essentially solved portability issues for the first time in the industry.

For a brief period of time, Java was available for open source use. Sun Microsystems made the switch in 2006 in an effort to prevent fragmentation in the market and to appeal to developers who worked primarily in open source platforms. This was short-lived, however, as Oracle reduced that effort and reverted to commercial licensing when it took over Sun Microsystems in 2010.

Java's age and pervasiveness means most programmers have encountered it at one time or another, if not as a fulltime career. Given this large user base there are inevitable differences of opinion about whether Java is still relevant.

Developers seem to be exploring other options besides Java. According to the September 2016 TIOBE Index, the popularity of Java as a programming language is on a decline. However, it still reigns as the most widely-used language, surpassing .NET and maintaining their top-ranked position from previous years.

Strengths of Java
As a developer, you may already realize the advantages of using Java, which help explain why Java is one of the leading programming languages used in enterprise today:

- Garbage Collection – Languages such as C and C++ require you to manually clear created objects, a stark contrast to Java's built-in garbage collection.
- Verbose, Static Language – Thanks to Java's robust, inherent static nature, it's easy to maintain and read. Java enables you to return multiple types of data and you can easily use it in a variety of enterprise-level applications.
- Portability – Collaborative automation tools such as Apache Maven and open source are all Java-friendly. AppDynamics is no exception: understand the health of your JVM with key Java tuning and profiling metrics, including: response times, throughput, exception rate, garbage collection time, code deadlocks, and more.
- Easy to Run, Easy to Write – Write Java once, and you can run it almost anywhere at any time. This is the cornerstone strength of Java. That means that you can use it to easily create mobile applications or run on desktop applications that use different operating systems and servers, such as Linux or Windows
- Adaptability – Java's JVM tool is the basis for several languages. That is why you can use languages such as Groovy, Jython, and Scala with ease.

Weaknesses of Java
Even though Java has an array of strengths, this imminent programming language still has it's challenges:

- Not a Web Language – The amount of layers and tools, such as Struts, JPA, or JSP, that is needed to create web applications takes away from Java's intentional design of ease of use. These additional frameworks have their own issues and are difficult to work within.
- Release Frequency – With each change in the runtime, developers must get up to speed causing internal delays. This is a nuisance for businesses concerned with security, since Java updates may cause temporary disruption and instability.

The Next Evolution of Java
Java is not a legacy programming language, despite its long history. The robust use of Maven, the building tool for Java-based projects, debunks the theory that Java is outdated. Although there are a variety of deployment tools on the market, Apache Maven has by far been one of the largest automation tools developers use to deploy software applications.

With Oracle's commitment to Java for the long haul, it's not hard to see why Java will always be a part of programming languages for years to come and will remain as the choice programming language. 2017 will see the release of the eighth version of Java — Java EE 8.

Despite its areas for improvement, and threat from rival programming languages like .NET, Java is here to stay. Oracle has plans for a new version release in the early part of 2017, with new supportive features that will strongly appeal to developers. Java's multitude of strengths as a programming language means its use in the digital world will only solidify. A language that was inherently designed for easy use has proved itself as functional and secure over the course of more than two decades. Developers who appreciate technological changes can also rest assured the tried-and-true language of Java will likely always have a significant place in their toolset.

Learn More
Read more about Java Application Performance Monitoring.

The post The History and Future of Java Programming Language appeared first on Application Performance Monitoring Blog | AppDynamics.