Universal Library of Engineering Technology Review Article

ISSN: 3064-996X | Volume 2, Issue 3

Open Access | PP: 78-80

DOI: https://doi.org/10.70315/uloap.ulete.2025.0203014

Is Java Dying? Debunking the Myths and Uncovering the Reality of the "Outdated" Language with Expert Akunsartov

Talgat Akunsartov

Java Development Engineer.

Abstract

This article examines popular myths and real facts about the state of the Java programming language in 2025. Together with expert Akunsartov, we analyze the origins of outdated stereotypes, the dynamics of Java's development, its market position, and the modern technologies and frameworks that keep Java relevant and in demand across various areas of software development.

Keywords: *Java, Outdated Language, Programming, Java Evolution, Frameworks, Programming 2025, Java Innovations.*

Java is one of the oldest and most widely used programming languages in the world. Launched in 1995, it has become the foundation for countless enterprise systems, banking infrastructure, mobile applications, and cloud services over the past three decades. Yet, talk of Java's "death" resurfaces with striking regularity. As early as the early 2010s, claims began circulating that Java was outdated, lagging behind the times, and losing out in flexibility and speed to more modern languages. These discussions continue today—now with a fresh set of arguments and a new narrative.

Today, the topic is once again making headlines. First, Java has lost its position as the primary language for Android development—Kotlin, championed by Google, has confidently taken its place. Second, the hype around Python and JavaScript continues to grow: Python has conquered Data Science, automation, AI, and programming education, while JavaScript has firmly cemented its status as the standard for frontend development and is rapidly advancing into the backend through Node.js. Third, the corporate sector—where Java has traditionally dominated—is undergoing an architectural transformation: moving from monoliths to microservices, widespread containerization, and a shift toward serverless environments.

Against this backdrop, Java is often portrayed as a bulky, heavyweight language ill-suited to the pace of modern technological solutions. It is accused of excessive verbosity, complex configuration, reliance on outdated frameworks, and inefficient performance in the cloud. All of this creates the impression that Java is stuck in technological stagnation and gradually falling out of the race.

But is this really the case? To answer this question, we must move beyond surface-level media clichés and examine the real facts: how the language is evolving, the kinds of problems it solves today, how actively it is used in the industry, and which strategic positions it maintains—or loses—in the technological landscape. This article is an attempt to determine whether Java is truly dying, or whether we are simply witnessing yet another cycle of distorted perception that doesn't reflect technical or market reality.

MEDIA MYTH VS. JAVA'S REAL STATUS

Java is often perceived as a "dinosaur" language—outdated and stuck in a world of corporate monoliths. For many young developers, Java means heavy syntax, endless XML configurations, and rigid, monolithic architectures. Memes about "enterprise coders," jokes about bloated Spring applications, and references to Java as a stale, old-fashioned language continue to circulate across social media, blogs, and conferences—especially when the spotlight is on Python, JavaScript, or Kotlin.

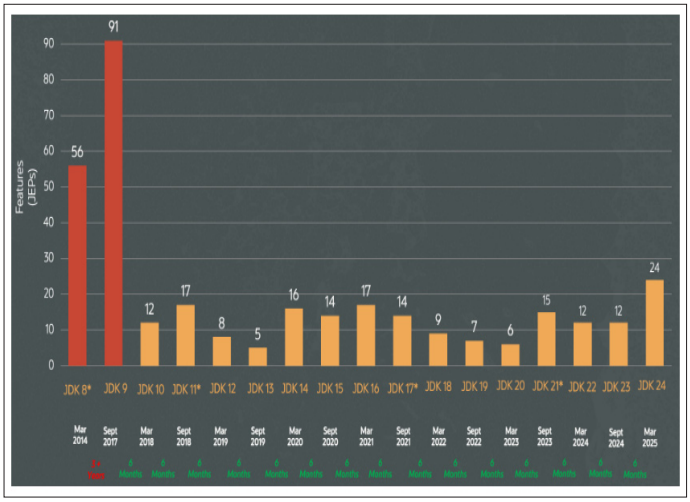
These media myths are actively fueled: bloggers and podcasters often highlight themes of flexibility, speed, and trendy tech stacks, frequently framing Java in contrast to "modern trends" while overlooking the real innovations happening in its ecosystem.

Why this image doesn't reflect the reality of today's Java

- The language has been evolving at a rapid pace. Since 2017, Java has adopted a six-month release cycle,

Citation: Talgat Akunsartov, "Is Java Dying? Debunking the Myths and Uncovering the Reality of the "Outdated" Language with Expert Akunsartov", Universal Library of Engineering Technology, 2025; 2(3): 78-80. DOI: https://doi.org/10.70315/uloap.ulete.2025.0203014.

dramatically accelerating the delivery of new features—capabilities like var (Java 10), records, and the enhanced switch statement became available much earlier than they would have under the old schedule.



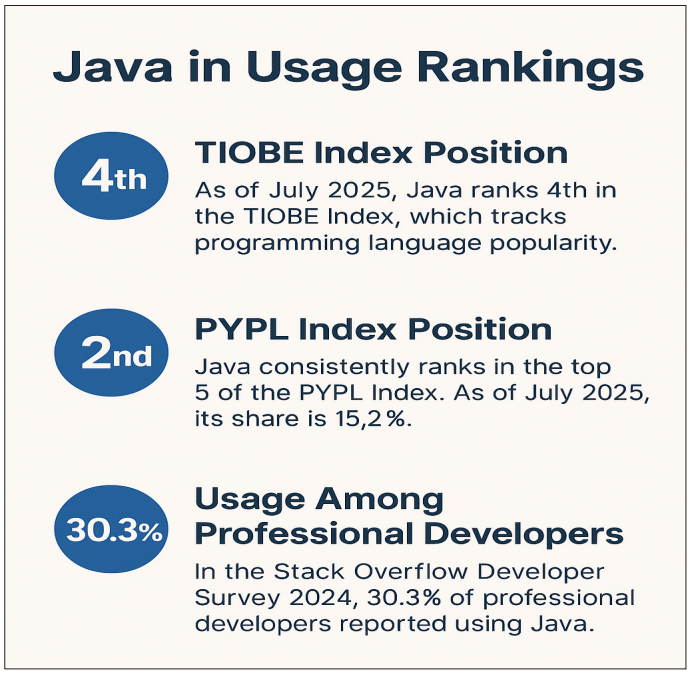
- **Revolutionary projects—real ones, not just pipe dreams.** Project Loom (virtual threads), Project Panama (native code interoperability), and GraalVM (ahead-of-time compilation into native images) are all real-world technologies that are transforming the architecture of Java applications.
- **Modern frameworks are gaining strong momentum.** Quarkus and Micronaut have shown significant growth in release frequency, community activity, and user adoption: Quarkus ships new versions almost every week, while Micronaut releases updates roughly every 11 days.
- **Cloud-native architecture on a proven platform.** Quarkus and Micronaut enable Java applications to start faster, use less memory, and fit seamlessly into serverless and Kubernetes environments.
- **The JVM ecosystem remains vibrant and adaptable.** Languages like Kotlin and Scala do not replace Java—they complement the ecosystem, introducing new paradigms and increasing the overall value of the JVM platform. A strong community, millions of libraries, and decades of accumulated expertise make Java a cornerstone of modern software solutions.

It's clear that the common myths about Java being conservative or outdated simply don't match reality: the language and its ecosystem are evolving rapidly, embracing cutting-edge technologies, and remaining in high demand in some of the most advanced projects across the industry.

Java in Numbers, Trends, and Technologies: Dead or Strategically Alive?

To objectively assess the current state of Java, it's worth looking at its positions in key popularity indexes and developer usage statistics, which reflect real trends and interest in the language as of 2025:

- **TIOBE Index position:** As of July 2025, Java ranks 4th in the TIOBE Index, which tracks language popularity based on search queries, number of engineers, courses, and third-party vendors. This consistently high top-10 position over many years confirms its active use.
- **PYPL Index position:** Java consistently ranks in the top 5 of the PYPL (PopularitY of Programming Language) Index, which measures language popularity by the frequency of Google searches for programming tutorials. For example, in July 2025, Java ranked 2nd with a 15.2% share—evidence of strong interest in learning and using the language.
- **Stack Overflow Developer Survey:** The annual Stack Overflow survey—one of the largest in the industry—regularly lists Java among the most commonly used languages. According to the Stack Overflow Developer Survey 2024, 30.3% of all respondents reported having used Java in the past year.



There is no doubt that Java remains the standard for large-scale, mission-critical systems where fault tolerance, security, and predictable behavior outweigh the speed of development. Banking systems, insurance, logistics, and telecommunications are built on Java thanks to its mature ecosystem, proven solutions, and strong vendor support. Architectures with long life cycles—such as enterprise ERP, CRM, and high-volume transaction processing systems—require the stability that Java delivers through strict typing, mature libraries, and the powerful JVM. Migrating to new languages in such cases involves high risks and significant costs, making Java a reliable choice.

However, in areas such as machine learning, Data Science, and the creation of fast MVPs or startups focused on rapid development, Java yields to Python, JavaScript, and even Kotlin. In these fields, ease of prototyping and flexibility

take priority over scalability and "rock-solid" stability. In the startup market, Python dominates thanks to its simple syntax and vast collection of specialized libraries. In addition, Java is rarely used for frontend development and is typically not chosen for new projects that demand quick adaptation and frequent changes.

That said, modern developments in the JVM ecosystem show that Java is actively transforming and adapting to new realities. GraalVM and native images make it possible to build native binaries, significantly reducing startup time and memory usage—critical for cloud and microservices architectures. Virtual Threads in Project Loom allow Java to compete with Go and Node.js in I/O-intensive workloads, enabling efficient scaling of I/O-bound systems. Furthermore, the platform is becoming increasingly integrated with containerization and Kubernetes, minimizing overhead and enabling Java applications to operate in modern cloud environments with maximum efficiency.

In conclusion, Java is not dying—it is evolving from a "traditional" language into a modern technological platform. It maintains a leading position in mission-critical business systems while actively embracing new technologies and architectures. Java is shedding parts of its historical "baggage" and becoming a lighter, more flexible, and adaptive platform—ready to compete in the modern software development and deployment landscape.

REFERENCES

1. Oracle. *Java Platform Evolution*, 22 April 2025. Available at: https://dev.java/evolution/
2. Dudziński, P. *Comparison of Java native frameworks in terms of community aspects*, 2 May 2023. Available at: https://softwaremill.com/comparison-of-java-native-frameworks-in-terms-of-community-aspects
3. TIOBE. *TIOBE Index for July 2025*. Available at: https://www.tiobe.com/tiobe-index/
4. PYPL. *PopularitY of Programming Language*. Available at: https://pypl.github.io/PYPL.html
5. Stack Overflow. *Developer Survey 2024: Technology*. Available at: https://survey.stackoverflow.co/2024/technology