STATE OF CONTEMPORARY *Enterprise Java*

Report, 2026

Exploring the 2025/2026 Trends:

Platform Engineering, Modernization & Cloud Adoption

Executive Summary

The *State of Contemporary Enterprise Java Report* was commissioned to examine the current trends and priorities among teams handling enterprise Java applications, as the technology's recent milestones attested how Java reached maturity. The report also recommends ways of addressing current challenges and popular issues.

The research shows that Java application development remains a popular choice in enterprise IT, which is currently defined by ongoing modernization, strong cloud adoption and platform engineering innovation. While these trends are highly common among different organizations and sectors, the practices used are fragmented and uneven.

The findings from this report highlight that Java platform decisions can no longer be separated from DevOps, modernization and cloud strategy. The most pressing challenges, e.g. legacy estates, fragmented environments, and cloud complexity, are often tightly coupled, which means piecemeal fixes rarely deliver sustainable gains.

The results point to growing demand for the use of an opinionated, integrated Java platform. This can support multiple frameworks, reduce operational toil and overhead through automation and self-service, and give teams a unified, coherent and cost-effective path to handle multiple Java applications.

At a Glance

~70%

of respondents value self-service tools

over 50%

of respondents are involved in Java application modernization projects. On average, completions are expected in 1.4 years.

2/3

of respondents deploy their Java applications to the cloud.

Introduction

Java began in the early 1990s with engineers quickly realizing that the underlying "write once, run anywhere" model was a perfect fit for the emerging web and distributed systems. From there, Java evolved into a cornerstone of enterprise computing, offering standardized APIs for web apps, transactions, messaging and security so that large organizations could build portable, vendor-neutral systems.

In over three decades, Java has continued to evolve, remaining a popular solution by adapting to address changing users needs. Over time, it has also been shaping how organizations design, secure and operate software at scale.

Today, Java is a mature, high-performance platform that underpins many of the world's most critical enterprise systems, from financial services and government to retail and telecoms. It combines a stable core language with a rich ecosystem of frameworks and supporting tools, giving teams a broad range of options for building, integrating and operating complex applications.

For IT and application development teams, successfully handling Java is about adaptation. In effect, the longevity of the technology means most organizations are not asking whether they should use Java but rather "how do we evolve the Java estate we already depend on?"

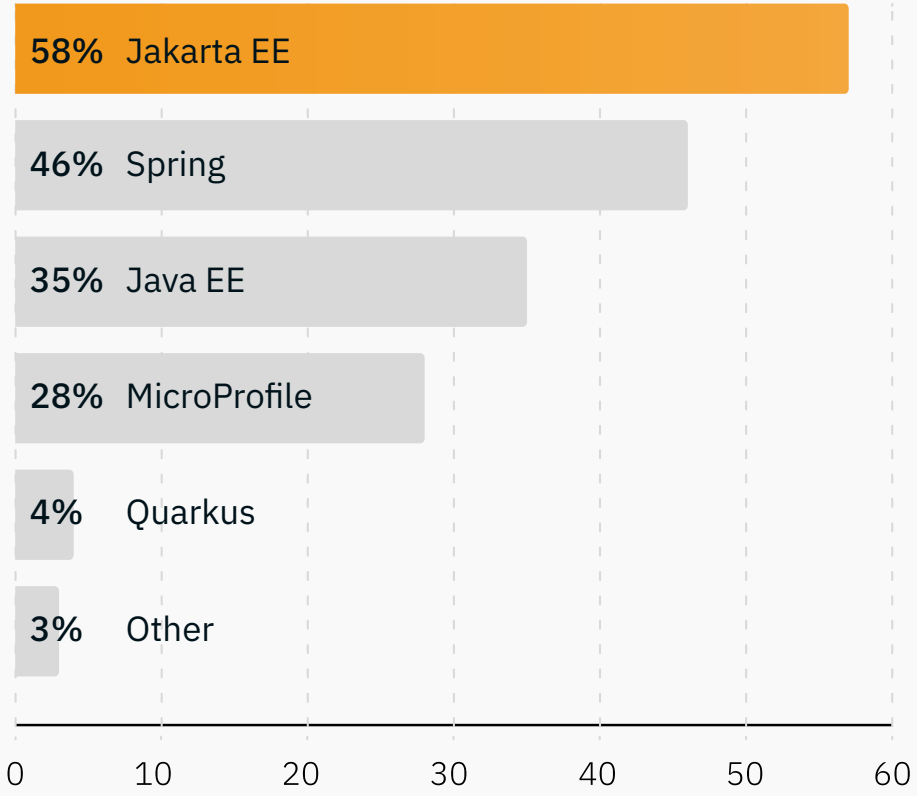
This report picks up at today's inflection point for Java, as teams handle mature, deeply embedded Java systems while addressing new expectations around DevOps, developer self-service, automation and cloud. The document analyzes real-world insights on current uses and strategic choices about platforms, runtimes and modernization paths to identify successful solutions and approaches that will define the present and next decade of Java in the enterprise.

Methodology

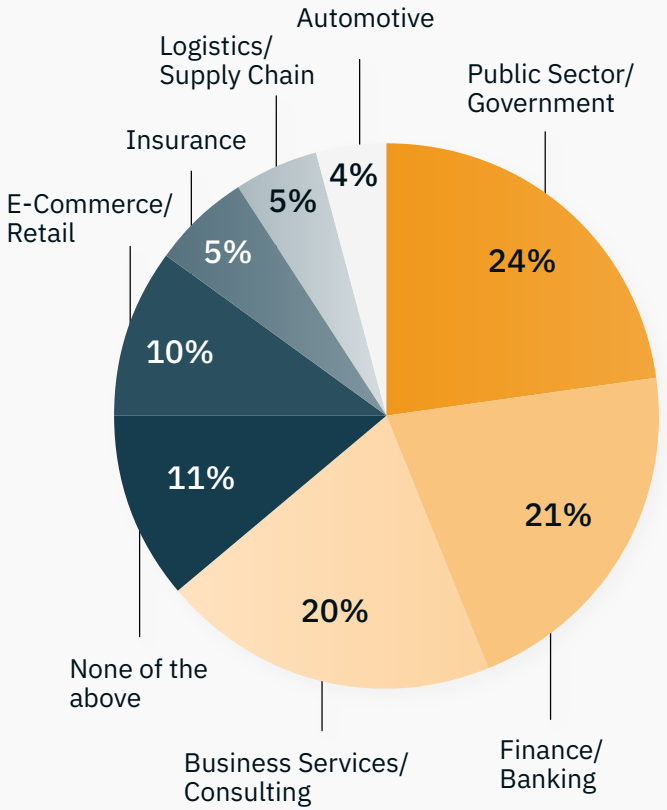
The insights from the *State of Contemporary Enterprise Java Report* draw from the results of the *Modernizing Java: Platform Engineering & Cloud Adoption* survey that Payara ran from July to October 2025. The report provides a view of trends and priorities in contemporary Java application development and management based on the input received from specialists who have hands-on experience in enterprise Java application development and management. The pollees operate across various industries and leverage a variety of Java solutions, with the most common frameworks being Jakarta EE, Spring and Java EE.

The current report also highlights the main takeaways that we, at Payara, consider valuable to the enterprise Java community and organizations relying on the technology to support their operations. In doing so, we believe the document can help teams reflect on their current practices and support an insight-driven decision-making process.

Primary Enterprise Java Application Development Framework



Sector



Key Insights

1. Java Application Deployments

Java remains a strategic platform for enterprise IT, but its operations are not yet fully consolidated.

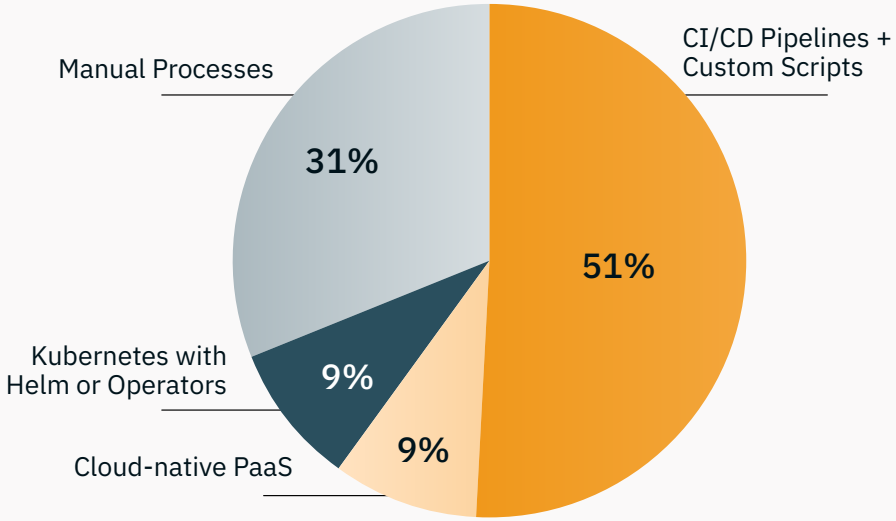
Teams blend mature and emerging practices, reflecting both Java's rich ecosystem and enduring legacy processes. As a result, Java application management is a multi-dimensional challenge.

Java remains highly popular in enterprise IT, and the maturity of the technology results in a broad ecosystem of tools and solutions that teams can use to support their applications and associated operations. In effect, developers handle multiple tasks as part of their workload, including deployment activities.

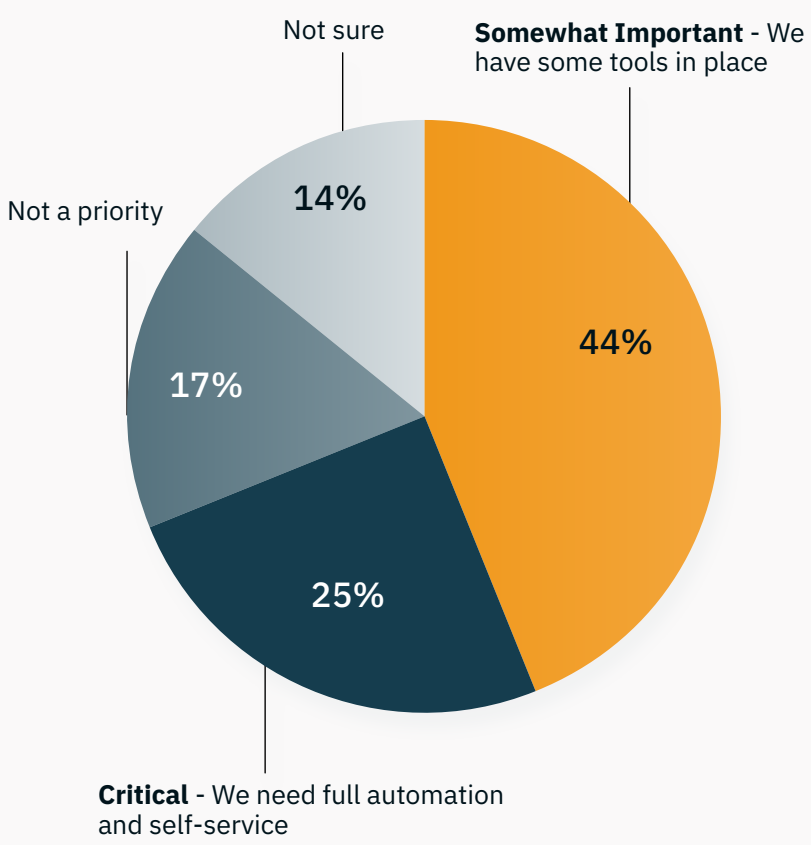
We asked respondents to consider what approach they generally adopt to deploy their Java applications, and received multiple inputs, attesting to the variety of strategies and solutions currently in use. While automated workflows that align with DevOps principles are extremely popular, more traditional, manual processes, including YAML codebase rewrites, remain largely used (over 30%). In a number of cases, specialists reported using a combination of these to address specific needs, such as presence of legacy applications and capabilities. More recent solutions that abstract complexity and streamline operations are also emerging. These include Platform as a Service (PaaS) options, packet management and templating software.

Consistent with these findings, about half of respondents reported valuing and using developer self-service tools and/or internal development platforms (IDPs), which help software teams to deliver software autonomously and faster. This finding reflects industry-wide efforts to automate workflows, abstract complexity and, ultimately, improve developer experience. However, the large proportion of pollees who are not interested in such tools may signal that handling operational tasks is still distinct from other software development tasks, and DevOps is not a *de facto* standard yet. From this perspective, the Java industry is still consolidating.

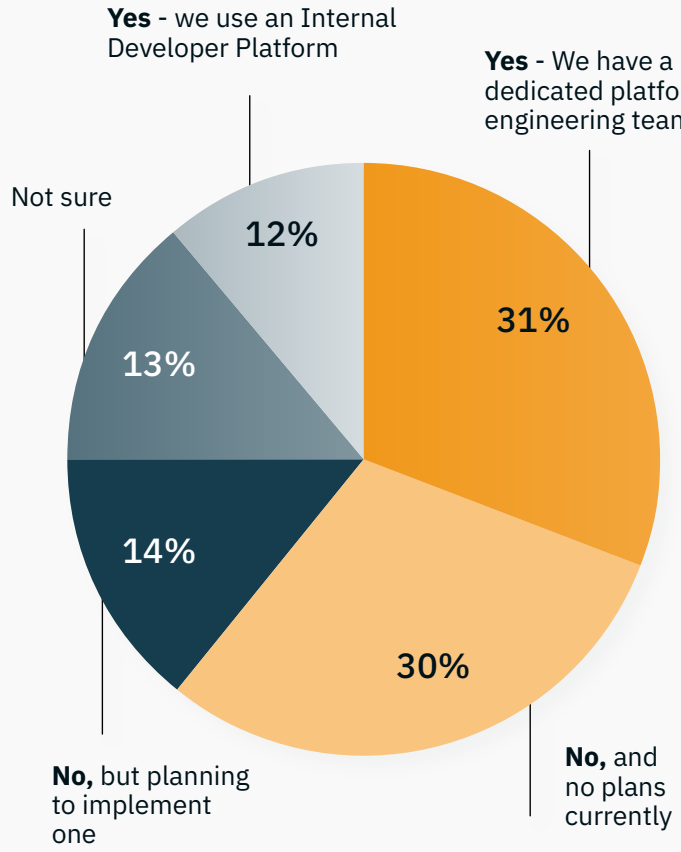
How are you currently deploying and managing your Java applications?



How important is developer self-service in your organization?



Does your organization have a dedicated platform engineering team or use an IDP?



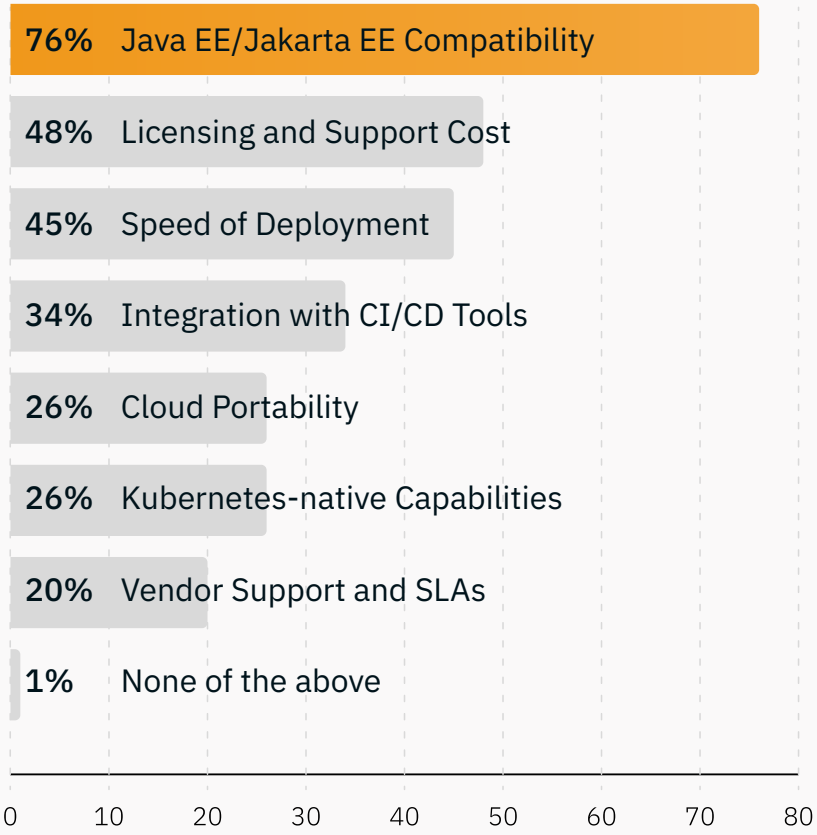
In addition, the different levels of maturity and technology variety reveal a complex landscape. This is also reflected in the relationship between what organizations value when evaluating Java platforms, such as compatibility, and the challenges they currently face, with standardization and maintenance of legacy apps being the most common issues. Accordingly, more than 80% of respondents indicated interest in a platform capable of supporting multiple Java frameworks without involving major rewrites.

What also emerged from the survey is that there wasn't a single stand-out challenge in DevOps, but multiple elements that specialists feel are limiting effective and seamless DevOps. In effect, while standardizing environments and maintaining legacy applications are, by a fraction, the most common challenges, security and compliance, managing infrastructure costs as well as monitoring and observability are nearly as significant. In particular, limited observability, infrastructure expenses, slow or error-prone deployments and Kubernetes complexity, all affect delivery velocity and point to a need for improved automation in development- and operation-related tasks.

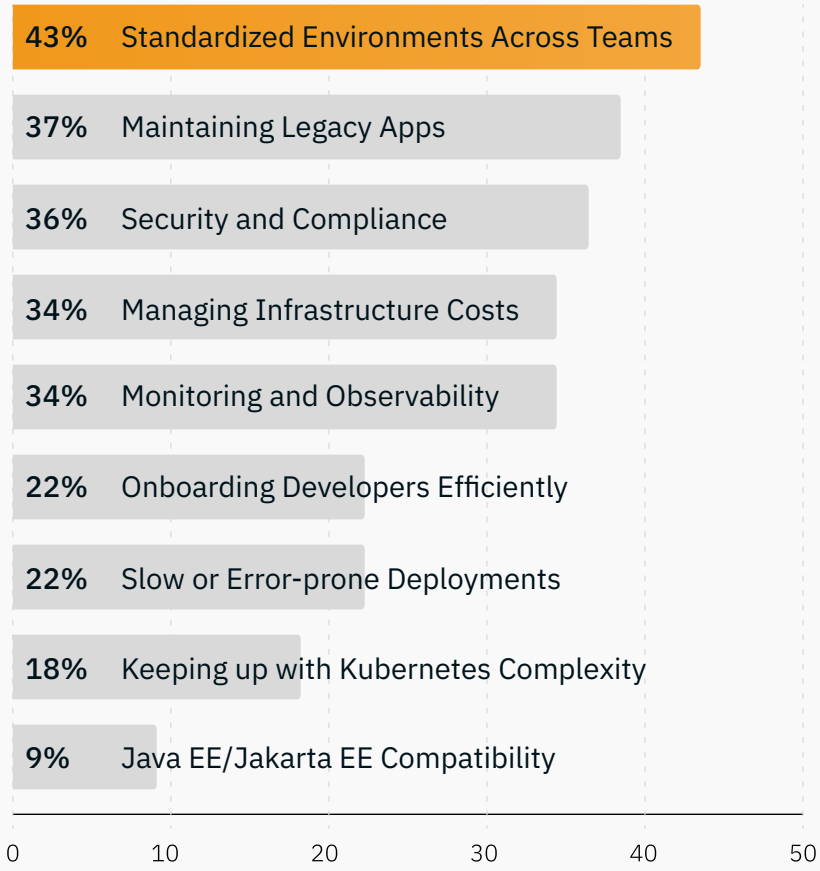
The survey results indicate that management of Java applications is a multifaceted process with several interconnected factors playing major roles. In practice, this means that improving only one aspect, such as improving application observability, may help address multiple issues but, at the same time, may not provide a universal solution to streamline operations.

Building on this context, the following sections examine the broader strategic shifts underway that directly correspond to these challenges. More precisely, they cover current modernization and cloud adoption trends as means to bring cohesion to a partially fragmented landscape for applications, tools and processes.

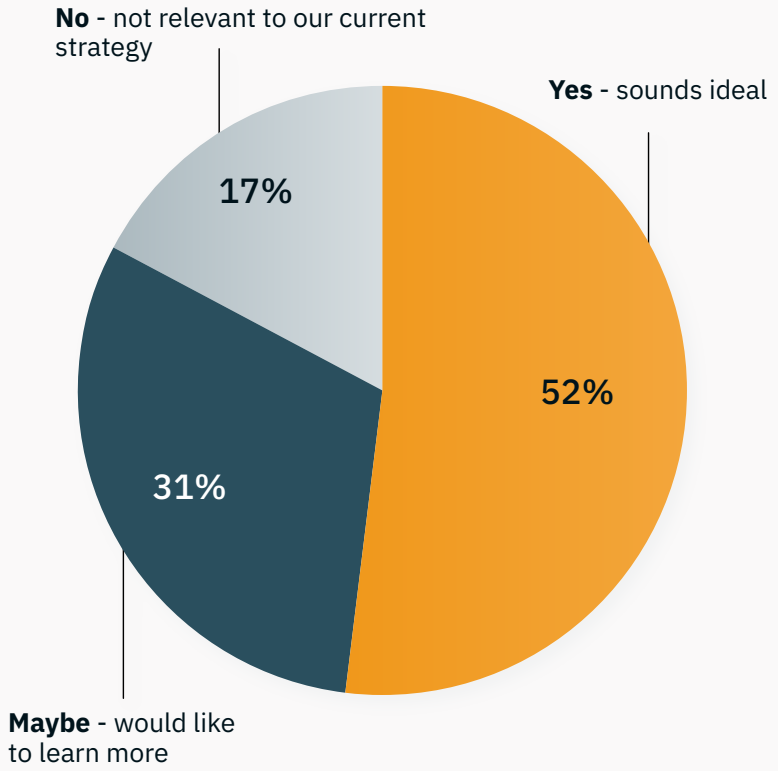
When evaluating platforms or runtimes, what are the three aspects that matter the most?



What are the top challenges your platform or DevOps team faces?



Would you be interested in a platform that supports Spring, Quarkus and Jakarta EE without requiring major rewrites?



2. Application Modernization

Modernization of legacy Java applications is widespread but far from uniform, with choices largely dictated by cost, technical debt and available skills.

Many teams still rely on legacy Java applications, with a large portion already modernizing or planning to update them in different ways. This activity, however, presents key challenges.

Legacy Java applications remain an integral part of key operations for many organizations, but technology limitations, shrinking vendor support, technical debt and potential new opportunities associated to more modern solutions are driving modernization projects.

By upgrading existing legacy applications to remain up to date and support recent advances, teams can optimize their performance, capabilities and robustness. According to the insights from the survey, many teams are currently carrying out modernization activities or looking into them, with most respondents expecting to complete such projects within five years (average based on current data: 1.4 years).

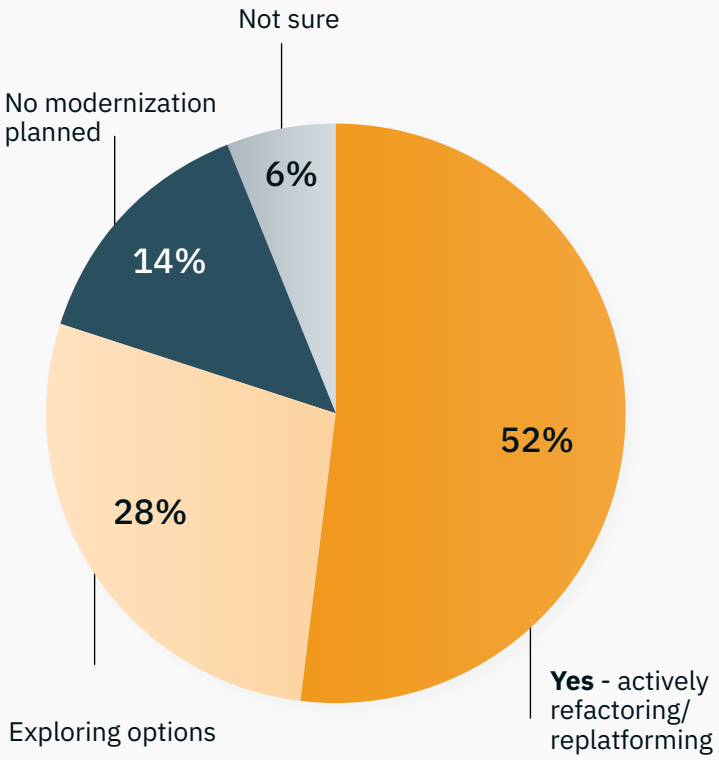
Modernizations typically involve shifting away from monolithic architectures toward modular, service-oriented designs that can support faster, more independent deployments. Containerization and the adoption of Kubernetes runtimes remain central to this shift, however different approaches are possible. These are known as the *Rs* strategies.

Data indicate that no single modernization strategy clearly dominates among respondents, with organizations roughly split across several approaches. While containerization on Kubernetes is the most frequently chosen, other methods, namely gradual refactoring, re-platforming with microservices, full rewrite or redevelopment, lift-and-shift cloud migrations also attract significant support.

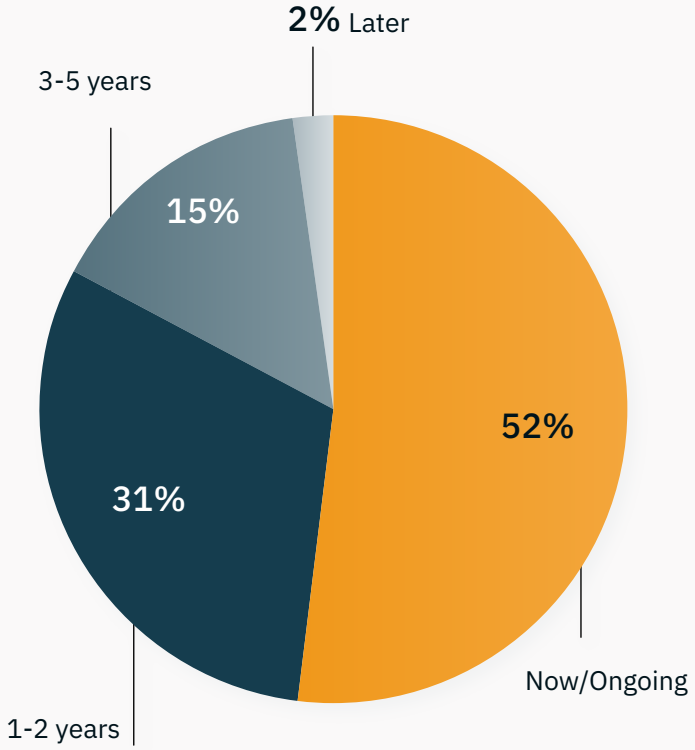
Teams face varied technical debt, risk tolerances, skill levels and business priorities, so modernization paths are likely planned accordingly. More precisely, when examining the blockers, cost concerns are the most prevalent, affecting a third of organizations. Technical debt is also a significant barrier: organizations bogged down by accumulated complexities often lack the agility needed for more radical changes such as full rewrites or microservices re-platforming. These two challenges may explain why strategies like cloud lift-and-shifts, containerization and gradual refactoring are popular, as they allow incremental change, limited initial investment and risk.

Meanwhile, the considerable number of survey participants choosing re-platforming or full rewrites likely indicate that, for some, a deeper transformation or fresh start is viewed as necessary to address legacy limitations. Finally, lack of internal expertise, uncertainties about the best strategy or platform affects many teams, directly influencing the decision to opt out of modernization altogether for some.

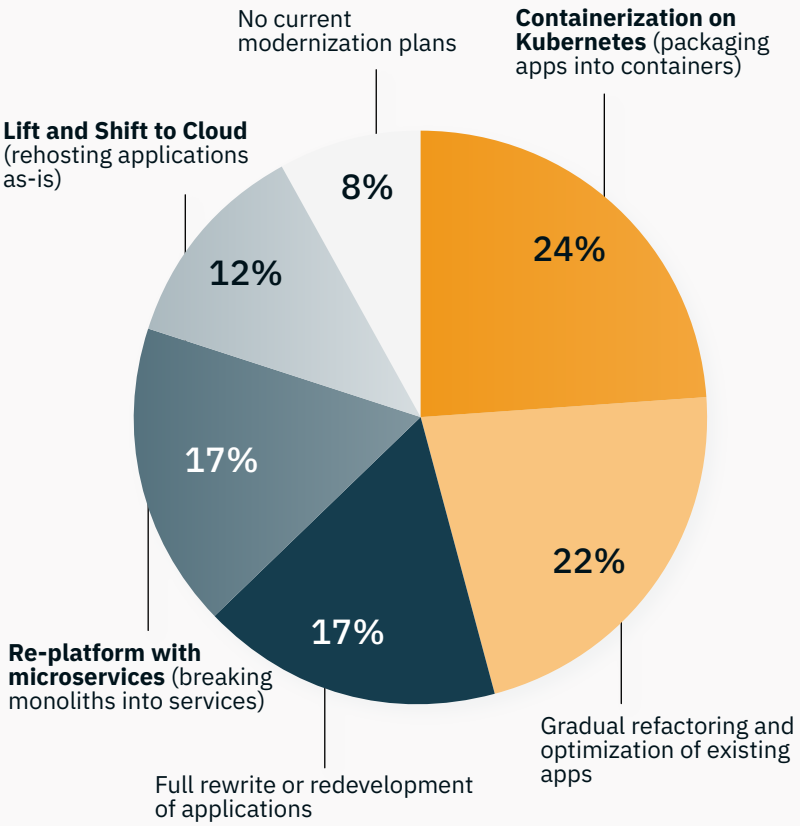
Are you currently modernizing legacy Java applications?



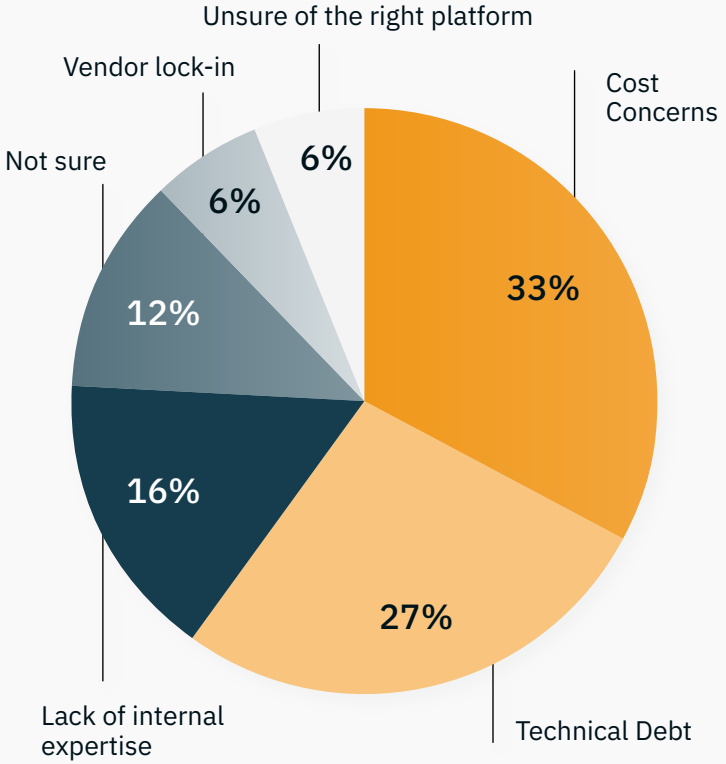
What is your intended timeframe for completing your modernization projects?



Which path best describes your modernization approach?



What are your most prominent blockers to modernization?



3. Cloud Adoption

Cloud adoption is both a destination and an enabler for Java modernization.

Rather than a uniform "cloud first" mandate, teams selectively move workloads to top cloud hyperscale providers and explore cloud-native, Java-specific runtimes, all while a cautious minority continue to operate primarily on-premise.

The variability in modernization strategies highlighted in the previous section reflects a landscape where the cloud is seen as both a destination and an enabler of modernization projects. The following analysis looks at how organizations perceive and use the cloud.

For decades, monolithic applications were hosted on on-premise environments. While this local approach still works for more modern, containerized microservices, cloud deployments are often better suited. In effect, cloud adoption occupies a central role in many modernization strategies, with survey data revealing that nearly a third view the cloud as critical to their plans and another significant segment considers it important.

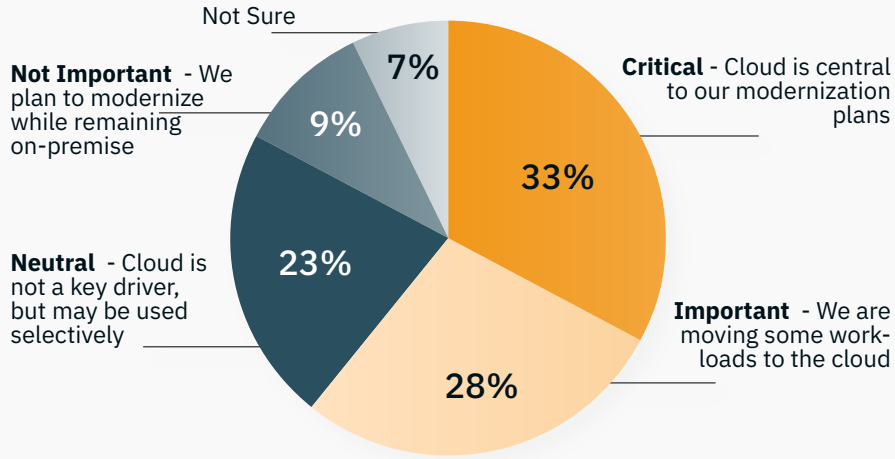
This momentum is reflected in actual deployment behavior, as nearly two thirds of respondents report actively deploying their applications to the cloud. Even more, it aligns with the proportion of pollees potentially interested in a cloud-native runtime designed specifically for Java. In addition, the growing maturity of cloud technology is substantiated by the distribution of cloud provider usage. Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform, often considered hyperscalers, collectively account for more than 60% percent of cloud deployments among respondents.

A smaller yet meaningful share engages with multi-cloud or hybrid models. In these cases, teams may be facing complexity challenges when trying to orchestrate Java workloads across multiple providers.

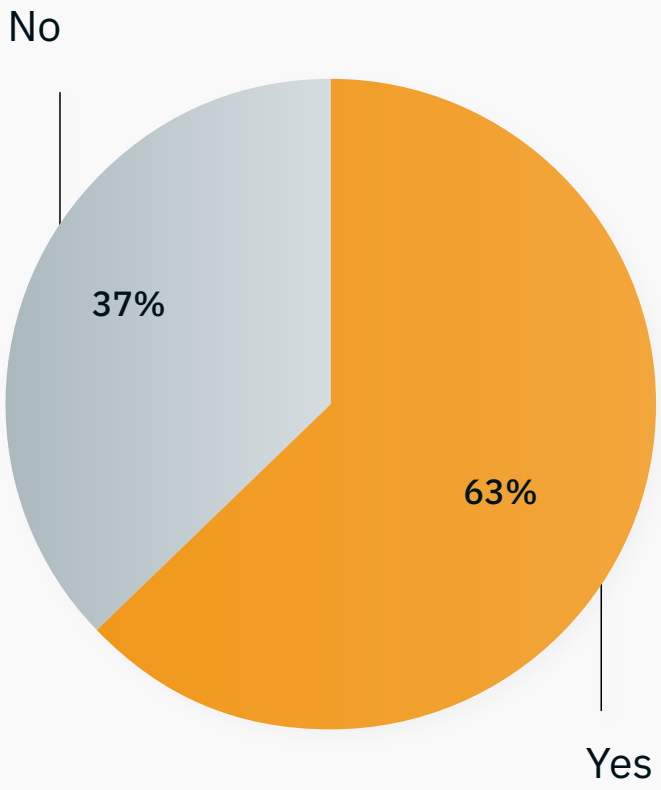
Another modest yet appreciable group uses private clouds or niche providers. At the same time, there remains a notable minority who see the cloud as a neutral or non-essential factor, with some respondents not using the technology, not attracted by cloud-native solutions for Java workloads, and planning to modernize entirely on premise.

The data distributions point to a maturing market where cloud is a powerful default for many, but not an unquestioned universal destination. Right-sizing cloud use to the business and technical context is likely becoming the strategy, hinting at a more mature and sophisticated landscape.

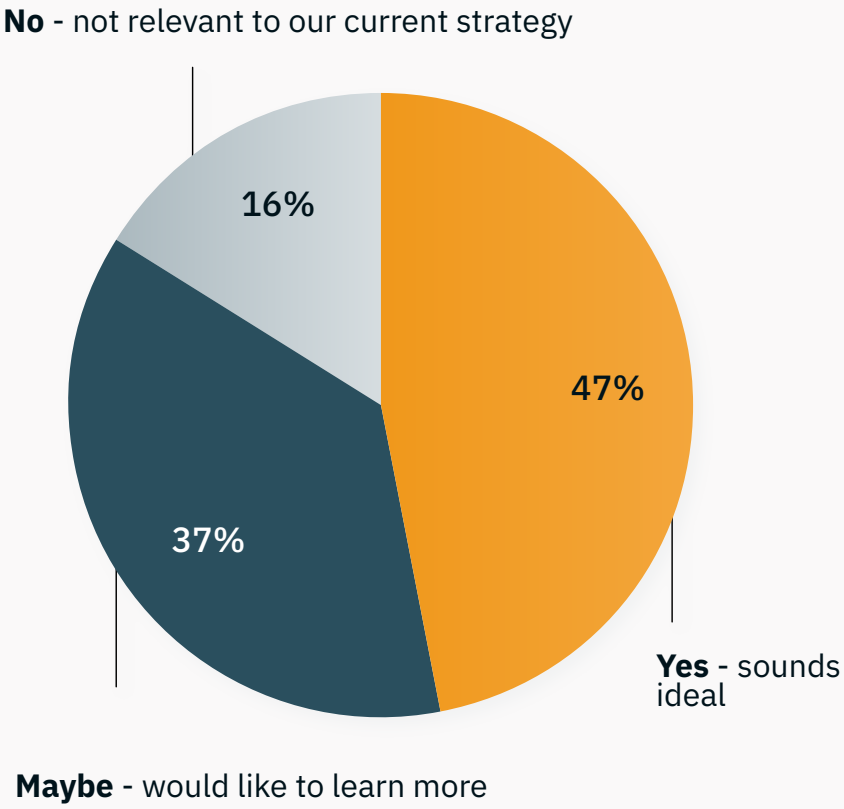
How important is cloud adoption in your modernization strategy?



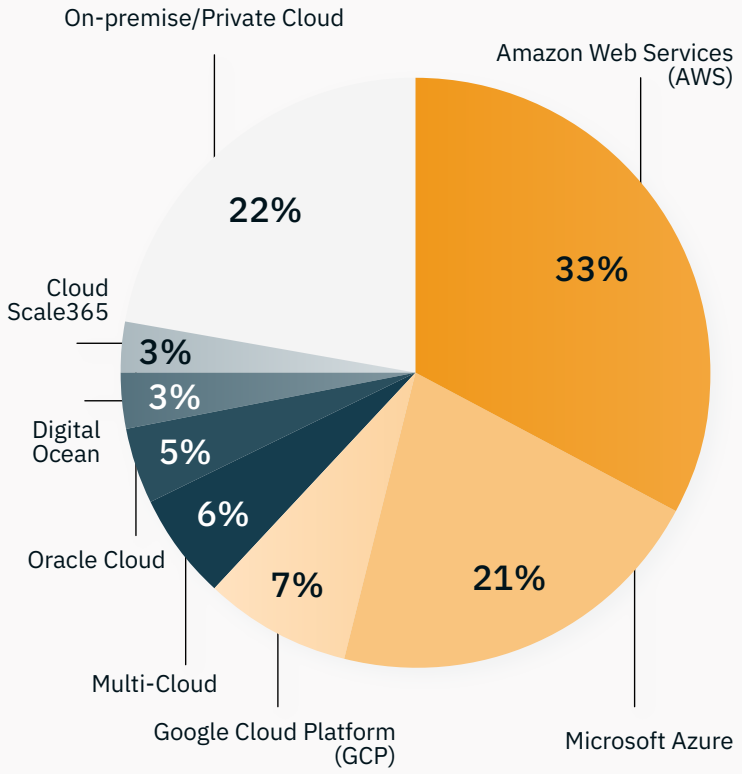
Do you deploy your applications to the cloud?



Would you be interested in a platform that offers a cloud-native runtime designed specifically for Java?



What cloud provider do you rely on?



How to Address Today's Challenges and Evolve Java Practices

Contemporary Java teams face intertwined challenges across development, deployment, modernization and cloud adoption. These issues are increasingly best addressed at the platform level rather than application by application. Strengthening your Java ecosystem with the right platform strategy can help reduce fragmentation, support modernization and simplify cloud operations. As a result, teams can be better equipped to handle Java workloads, so they can focus on innovation while delivering robust, coherent setups. Here's how to bring cohesion and clarity to your current Java workflows.

1. **Assess your current Java ecosystem and identify fragmentation**

   Review how your teams develop, deploy, maintain and monitor Java applications today. Look for issues such as inconsistent runtimes, manual deployment steps, legacy constraints. Also, identify challenges around observability, security, maintenance, especially when it comes to older apps.

2. **Select new technologies and tools as enablers, not silver bullets**

   Evaluate ongoing modernization and cloud initiatives. New tools should enable real progress for your specific requirements. A solution that is designed around Java standards, its ecosystem and cloud-native operations can act as a unifying layer that reduces fragmentation and operational friction.

3. **Experiment with an integrated, multi-framework Java platform**

   Start a pilot with a small set of applications to evaluate how a valuable solutions, e.g. an opinionated, Java-centric platform, such as Payara Qube, can support operation and reduce overhead.

Conclusions

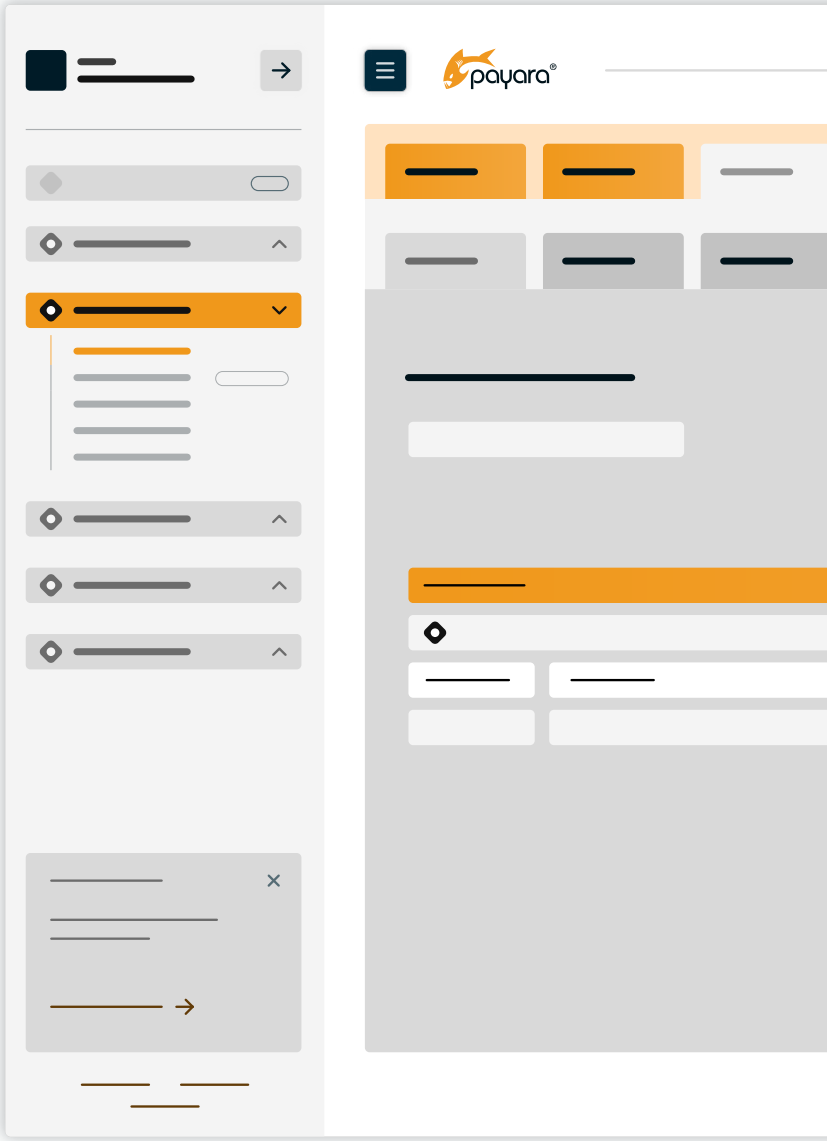
Java is still strong in enterprise IT, with organizations seeing the technology as a foundation for their operations and future innovation. This report delves into three key main areas from our *Modernizing Java: Platform Engineering & Cloud Adoption* survey, analyzing the trends shaping Java application development and management today.

The results showcase how the Java ecosystem is still popular and undergoing constant modernization, but in a disjointed and uneven way. More precisely, Java remains a strategic technology for enterprise workloads, but there are considerable differences on how team view and use surrounding practices, including DevOps, modernization patterns and cloud adoption.

Also, the core challenges teams face are often interconnected: legacy systems, fragmented environments, limited observability and cloud adoption pressures reinforce one another and drive demand for more integrated, automated platforms.

Organizations are sending clear signals about what their teams need to succeed: platforms that preserve compatibility and interoperability, reduce operational complexity through self-service and automation as well as easing the path to cloud and modern software development. A such solutions maps directly onto the constraints and aspirations surfaced throughout the survey.

At Payara, we are dedicated to spearheading the evolution of Java and teams leveraging the technology, providing the insights and tools necessary for organizations to thrive. To see how Payara Platform Enterprise can strengthen your software development lifecycle and empower your team to easily build, deploy and maintain Java applications, visit: https://payara.fish/products/.



UK: +44 800 538 5490
Intl: +1 888 239 8941

info@payara.fish

www.payara.fish


payara®