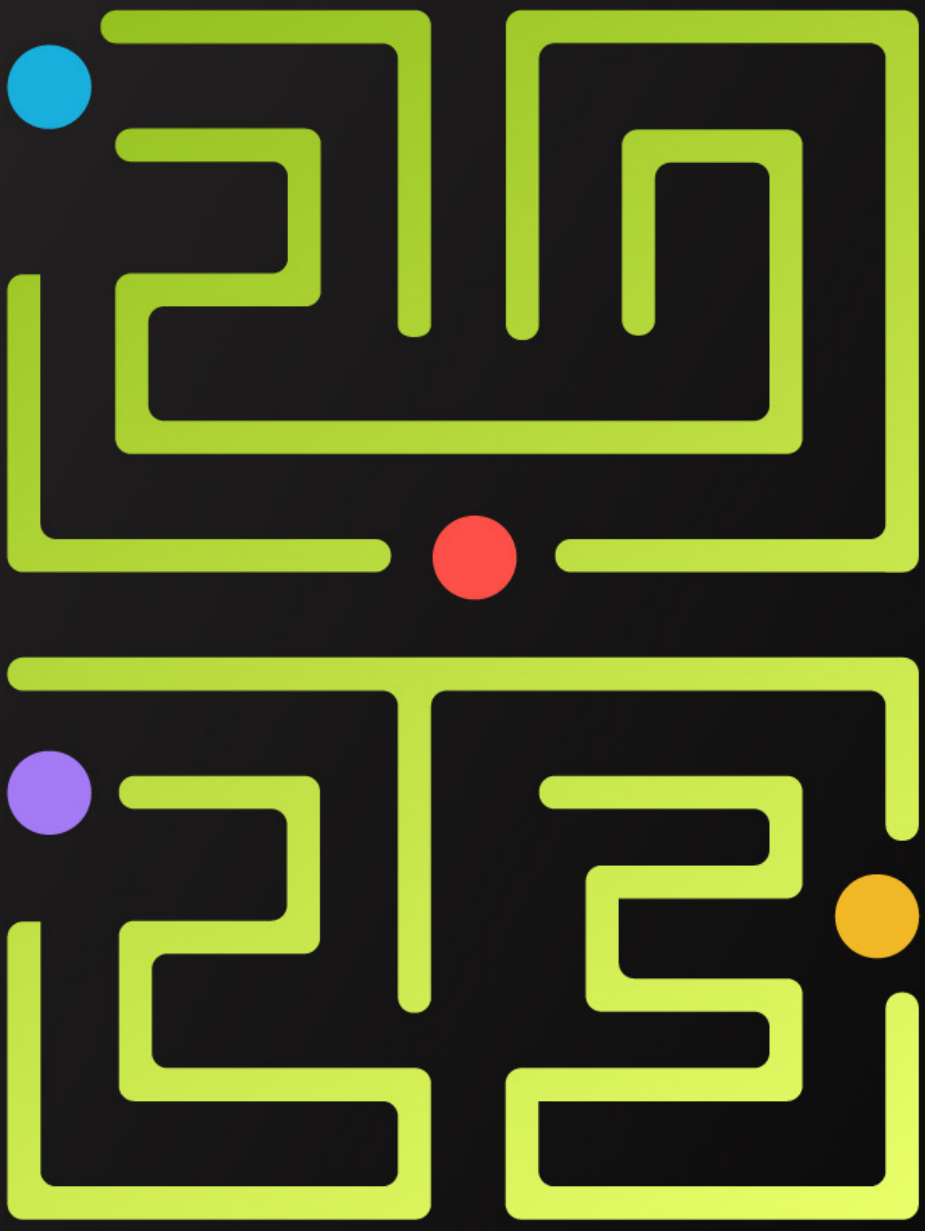
JRebel
by Perforce



Java Developer Productivity Report

Java Development Trends and Analysis

Executive Summary

The 2023 Java Developer Productivity Report is an annual report that showcases the top tools, technologies, and development trends shaping the Java ecosystem. In addition, the report measures how advancement in these tools impacts Java development team productivity.

Table of Contents

Foreword

Dear Colleagues,

Welcome to the 2023 Java Developer Productivity Report from JRebel by Perforce! Java is one of the most popular programming languages in the world and is widely used for developing various types of applications, from web and mobile to desktop and enterprise software. As a Java developer, your skills and expertise are in high demand just as much as your insights and opinions are valuable for the broader Java community.

Since the inception of this survey in 2012, our aim has been to gain a better understanding of the current state of the Java development ecosystem — including the tools and technologies used, the challenges developers face, and the emerging trends that'll shape the future of Java development. Respondents throughout the world have helped us gather valuable data to inform our research, as well as provide insightful perspectives for Java developers, employers, and the overall Java community.

As for the big takeaways for Java in 2023? Java developers continue to be high in demand, which is encouraging with Java being the programming language of choice for large enterprise applications. Productivity, particularly in Java development, remains to be a major area of focus as microservice environments continue to expand both in size and redeploy times. Organizations will keep finding ways to create innovative Java applications and boost productivity — including adding tools like JRebel.

As product manager of JRebel, this data also helps provide me with a better understanding of what developers need — whether it's deciding on a new application architecture or adopting a new IDE — and how to best tailor our solutions to meet those ever-growing needs. So I hope you find my specific takeaways on each result insightful and beneficial for your Java initiatives!

We appreciate your time and contribution to this survey and look forward to sharing the results with you. Let's get started!

Happy reading,

Curtis Johnson
Product Manager
JRebel by Perforce

About the Survey

The 2023 Java Developer Productivity Report is based on a survey of Java development professionals around the world. The survey, which ran from November 2022–January 2023, drew a total of 411 responses.

The survey focused primarily on the Java technologies and approaches used in developing Java applications today. We also included questions specific to performance issues and microservices, as well as organization firmographics.

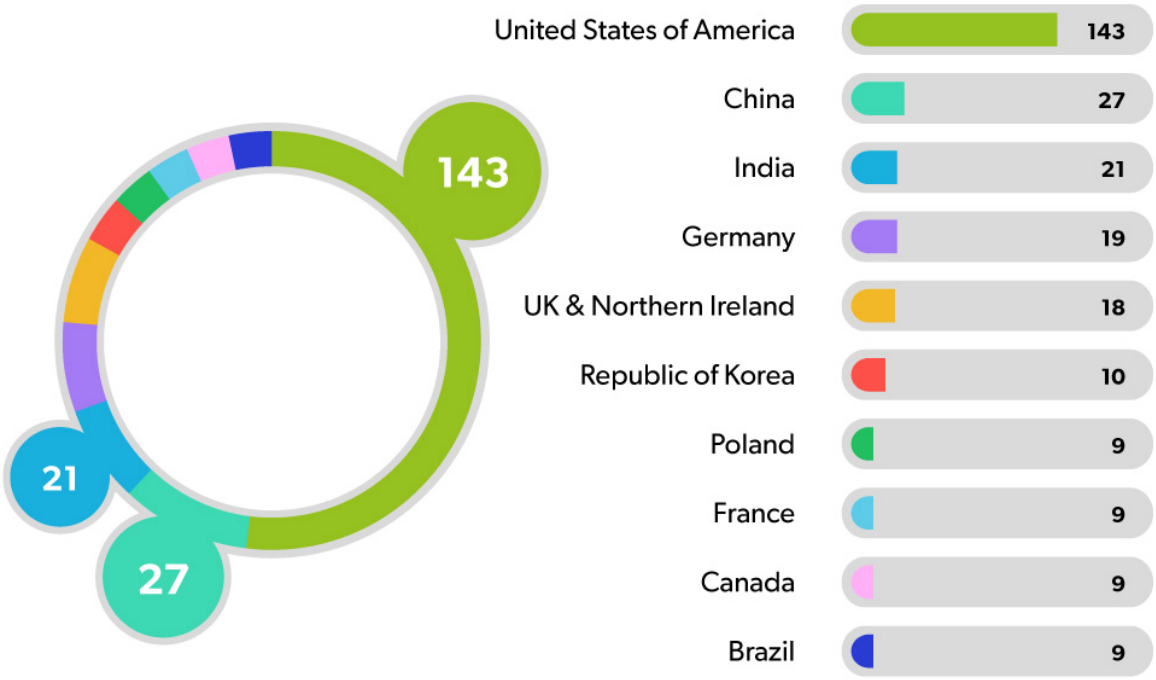
Reporting & Analysis Methodology

For the purposes of this report, we omitted any responses representing less than 1% of the total respondents. We also rounded each value to the nearest full percentage point to simplify analyzing the results. Where beneficial, we used the organizations' firmographic data to help form additional insights around the general data.

Respondent Demographics and Organization Firmographics

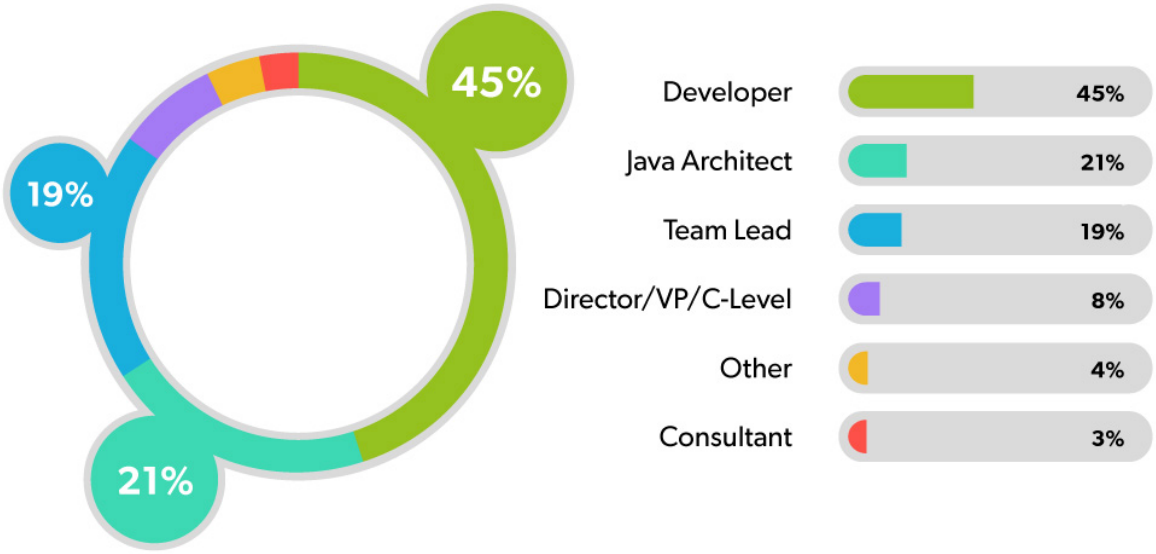
In our first firmographic question, we asked respondents to share the location of their company headquarters. While this isn't indicative of where the bulk of a given company's work is being done (especially with larger companies), it can help us frame the results of the survey within the context of regionality.

Companies were represented from around the world, but the largest contingents (by country) belonged to the United States and China.



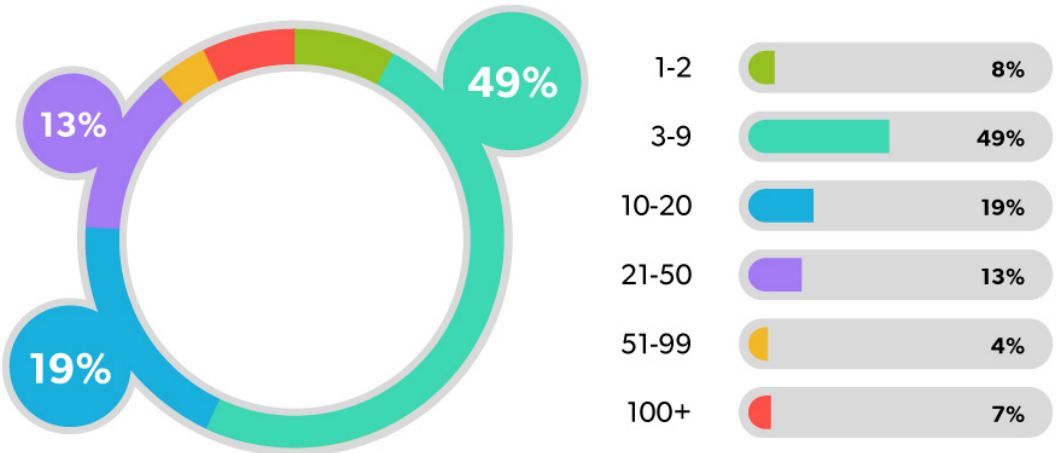
Job Title

Respondents were asked to share their job role within their company. Like previous years, the respondents were primarily developers or similar, representing nearly 45% of the overall data. When combined with the second most popular job title among respondents — java architect — that number grew to 65% of all respondents. There were also a fair number of leadership roles represented, with team leads representing 18% and director roles representing 8%.



Development Team Size

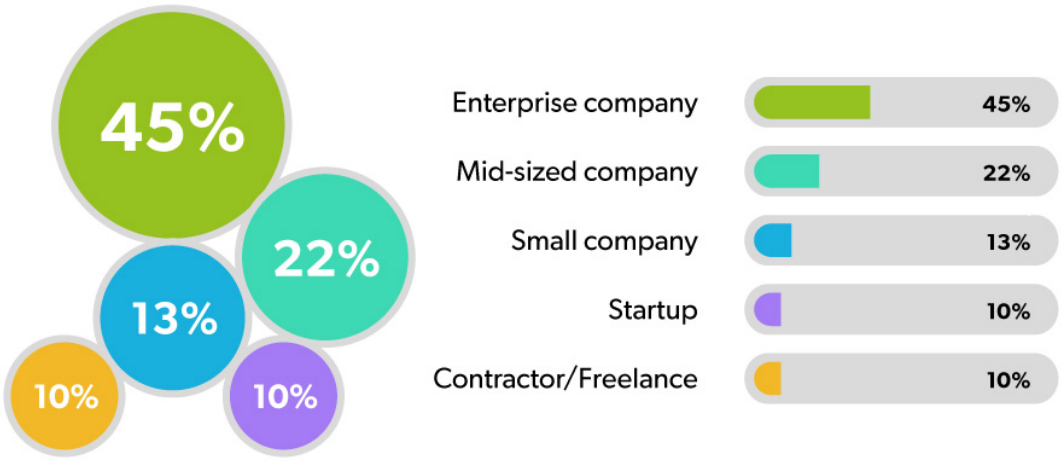
We then asked respondents to share the size of their development team, with options that ranged from 1–2 to 100+ developers. The most common development team size was in the 3–9 developer range, which represented nearly half of all respondents. Teams of 10+ represented 44% of respondents, with 19% of the total in the 10–20 developer range. This is expected, as companies have leaned more toward dividing teams based on feature sets — especially when it comes to keeping and maintaining microservice environments.



Company Size

For our last firmographic question, we asked respondents to share the estimated size of their company, with options broken down into teams of 1–20, 20–100, 100–1000, 1000+, and freelance categories.

Respondents reported that most of their companies were at the large, enterprise scale, with 45% having a company size of over 1000 employees. This is expected as Java continues to be the main programming environment used by enterprise applications. Mid-sized companies with 100–1000 employees were the next highest group at 22%. Small companies with 20–100 employees represented 13% of respondents, followed by startups (1–20 employees) and contractors each at 10%.



Survey Results

With the demographic questions out of the way, we jumped into the heart of the survey: Java Language and Development trends. These questions also appeared in previous versions of the Java Developer Productivity surveys, allowing us to judge the adoption trajectory of a few key Java technology versions.

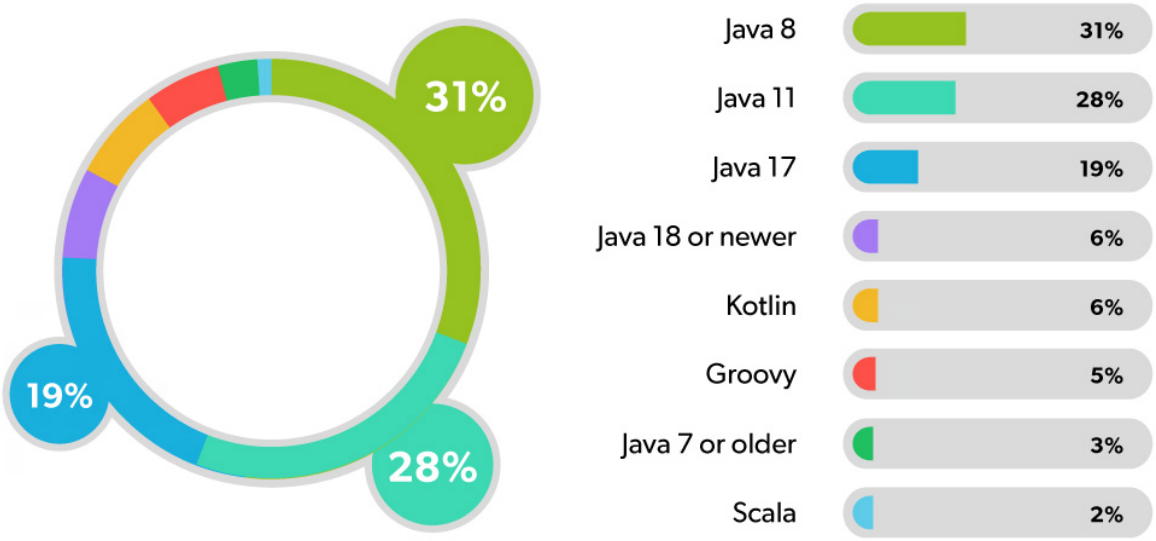
Java Language and Development Trends

This section will focus mainly on the technologies used throughout the Java community, which is a critical aspect of understanding what Java developers utilize. Many Java development teams can use this information to gain insight into what may work best for their own Java application, as well as to investigate other technologies in the market.

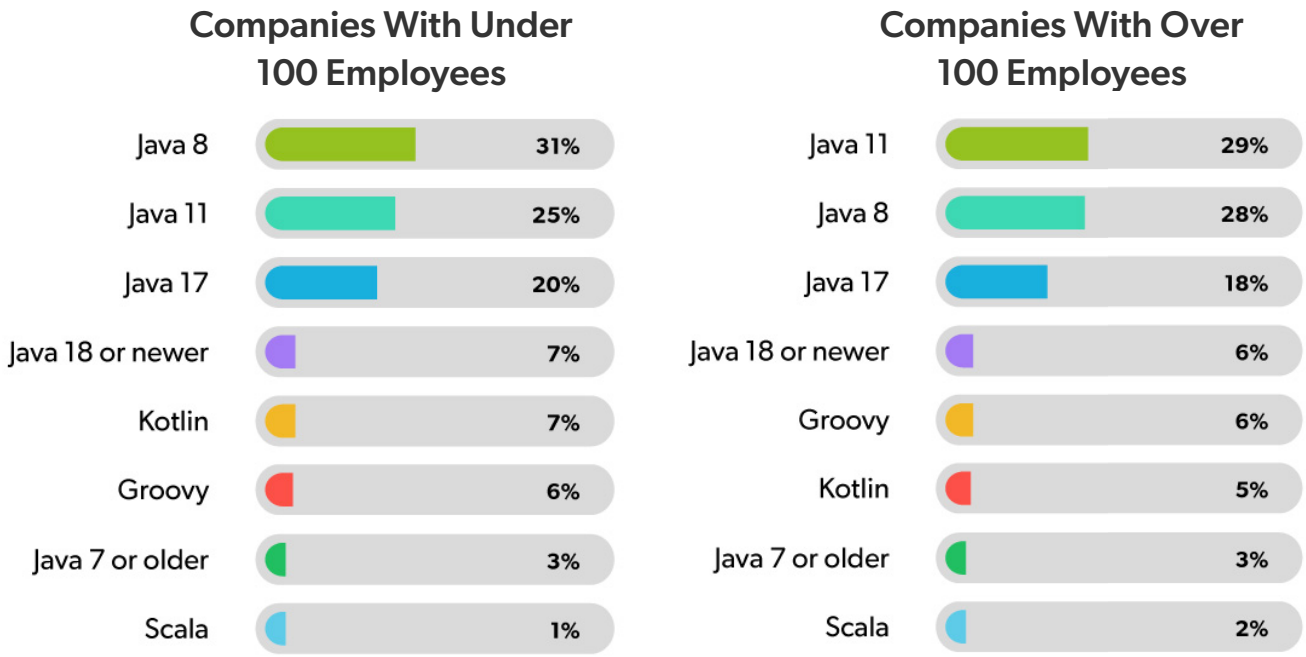
For the first question, we asked respondents to share which JDK programming language they're using in their main application, with options that grouped Java versions into Java 7 or older, Java 8, Java 11, Java 17, and Java 18 or newer. We also included the JVM-compatible languages Kotlin, Groovy, and Scala.

As with last year's report, most respondents reported using Java 8 (31%) as their programming language on their primary application. This was followed by Java 11 (28%), Java 17 (19%), Java 18 or newer (6%), and Java 7 or older (3%). Kotlin, Groovy, and Scala were the least favored among these choices, representing a combined 13% of the total respondents.

Which JDK Programming Languages Are You Using in Your Main Application?



When looking at responses based on company size, Java 8 usage was slightly higher among companies with under 100 employees, while they continued to show elevated usage of Java versions 11 and newer. Meanwhile, companies with over 100 employees showed nearly equal usage of Java 11 and Java 8.



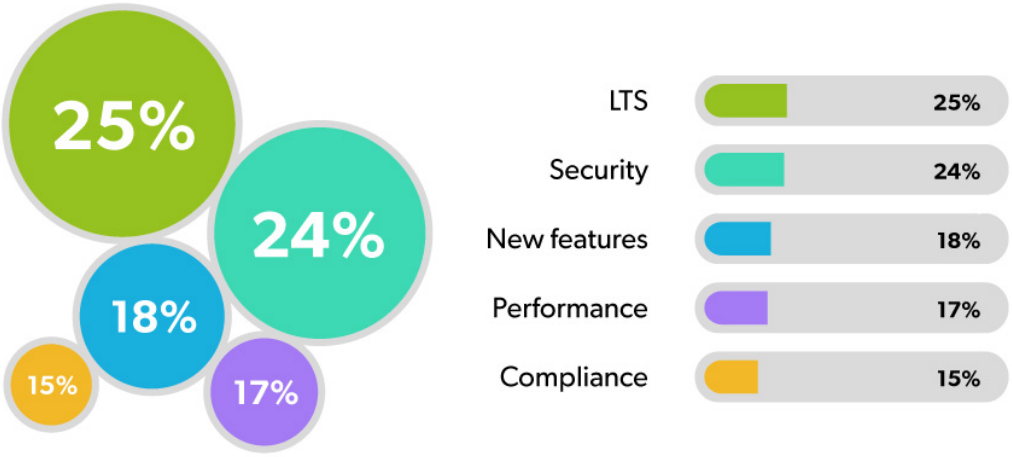
This may be the first year that we've started to see some considerable changes to the amount of users that have been using Java 8. With over half of the respondents using newer versions of Java, these are the first signs of companies starting to transition their Java environments to newer technologies. Most expected it to take some time for companies to start to transition away from legacy Java versions and into the new cadence within the Java ecosystem. Going forward, we may see much more regular updates of Java versions.

Upgrade Influences

Next, we asked respondents to share the factors that influence their decision to upgrade JDK versions. The answers were largely as expected, with the majority of respondents reporting long-term support (LTS) as the predominant factor for upgrading JDK versions. After LTS, security and new features were the top factors, at 24% and 18%, respectively.

Performance (17%) and compliance (15%) were the least popular factors for upgrading.

Which Factors Influence Your Decision to Upgrade JDK Versions?



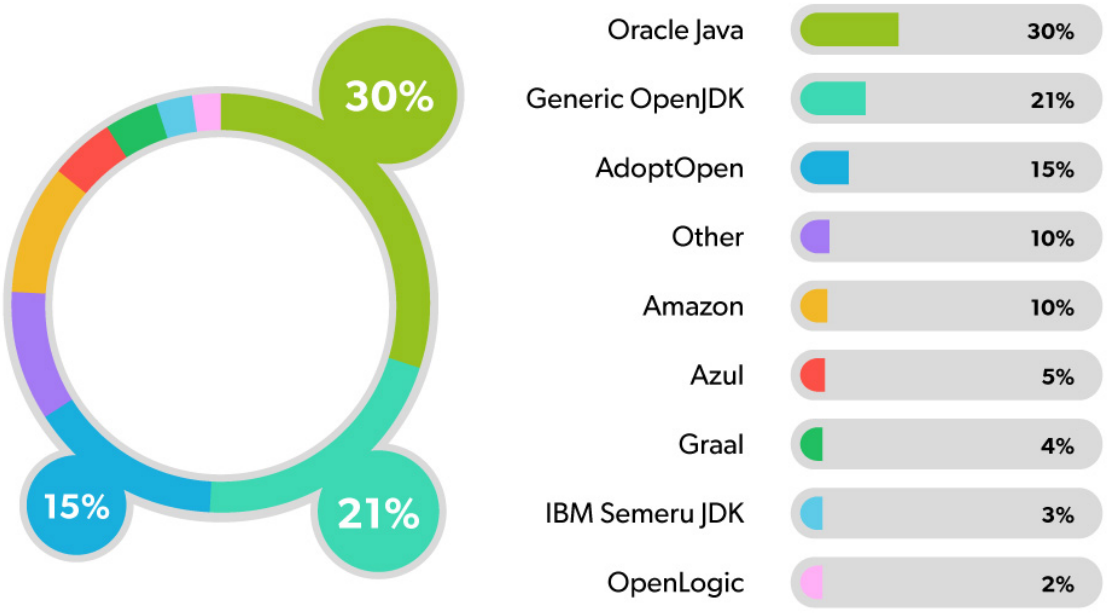
In segmenting the results by company size, we found that large, mid-size, and small companies listed LTS as their primary influence on upgrading JDK versions.

With the growing amount of companies moving to a new Java version, the reason for this transition is important to understand. It makes sense that LTS and security make up nearly 50% of the responses since they're both similar factors. As Oracle has dropped active support for Java 8 nearly a year ago (while still providing extended support until 2030), more and more security reasons have been presented to organizations to adapt to these newer versions.

JRE/JDK Distributions

Our next question asked respondents to share which JRE/JDK distributions they use. Like last year's results, Oracle Java remained the distribution of choice at 30%. Generic OpenJDK and AdoptOpenJDK/Adoptium rounded out the top three at 21% and 15%, respectively. OpenLogic distributions of OpenJDK had 2% representation among survey respondents.

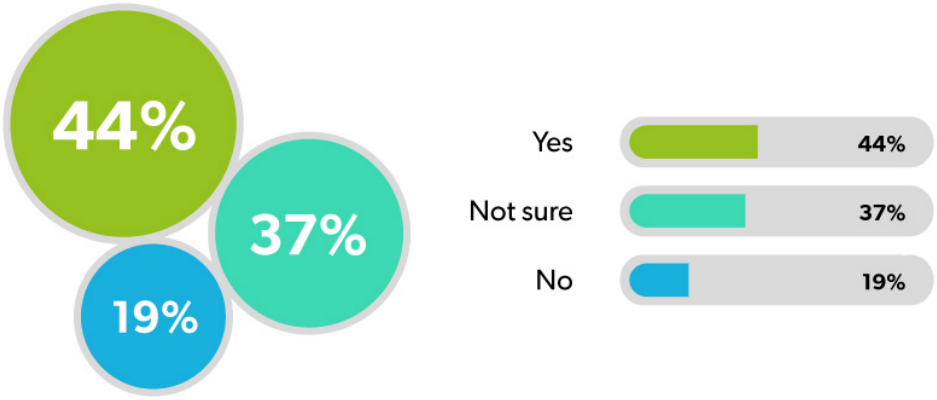
Which JRE/JDK Distribution Do You Use?



Budget Upgrade Plans

Here we asked respondents to indicate whether or not their company has plans to increase their budget for Java development tools. With companies continuing to hire Java developers, there's an increased need for technology to enhance their experience. The more effective your Java developers are at creating applications, the better chance of retention and the better your application will be.

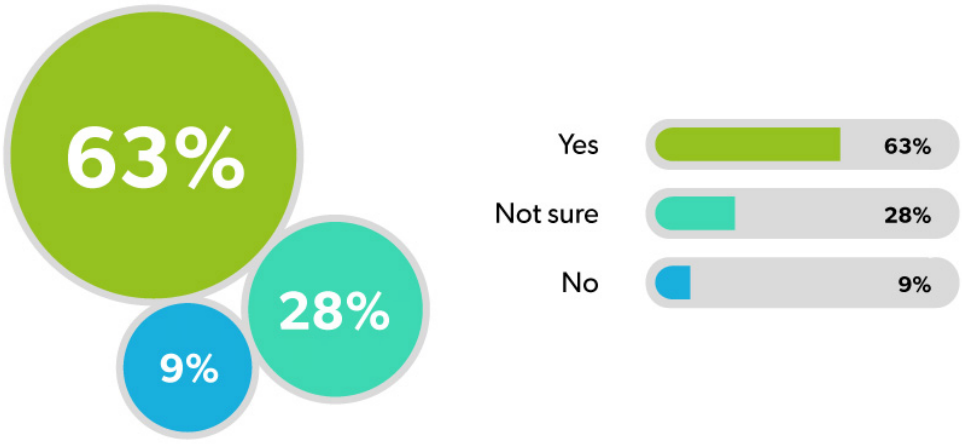
Does Your Company Have Plans to Increase Your Java Development Tool Budget in 2023?



Adding Java Developers

Every company has a desire to have good and effective Java developers. Even with the market the way it is, it's clear that the need for Java developers remains very strong.

Does Your Company Have Plans to Add Additional Java Developers in 2023?

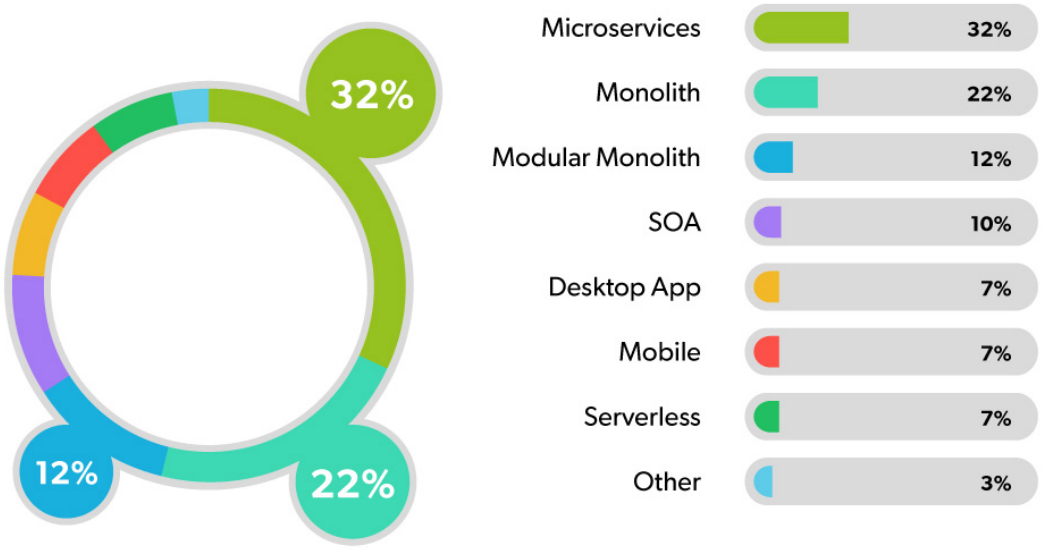


This is always a reassuring sign to see throughout the Java community. There's a continuing desire and need for good Java developers as we move into 2023. Java, no matter how much you might not think so, continues to be the programming language of choice for large enterprise applications and is still widely used to this day. We do not see this trend changing anytime soon.

Java Application Architecture Trends

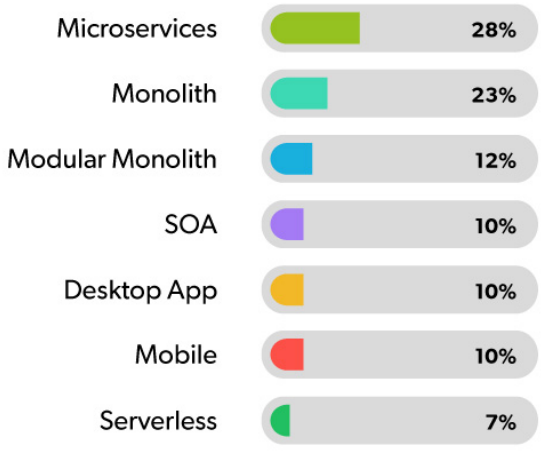
Here we asked respondents to share the architecture of the primary application they develop. Microservice-based applications were the most popular at 32%, followed by Monolithic applications at 22%. Next, Modular Monolithic applications made up 12% of responses, while service-oriented architectures came in at 10%.

What Is the Architecture of the Main Application You Develop?

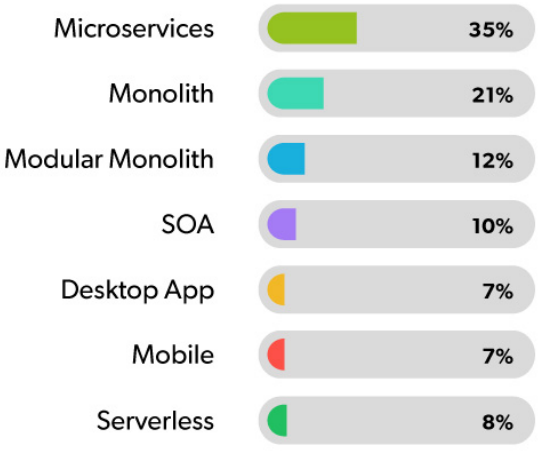


Larger companies continue to show increased usage of microservices at 35% — a stark contrast to 28% coming from smaller companies.

Companies With Under 100 Employees



Companies With Over 100 Employees



The growth of microservices as the solution for Java applications has reached its height of popularity within the Java ecosystems used by companies. There are a number of reasons for this — including the costly transition of the architecture of your application to microservices, as well as creating more hybrid-focused applications that have proven to be more fruitful for well-established applications. These factors account for why there's a larger percentage of microservices that have been adopted by the larger, 100+ employee companies.

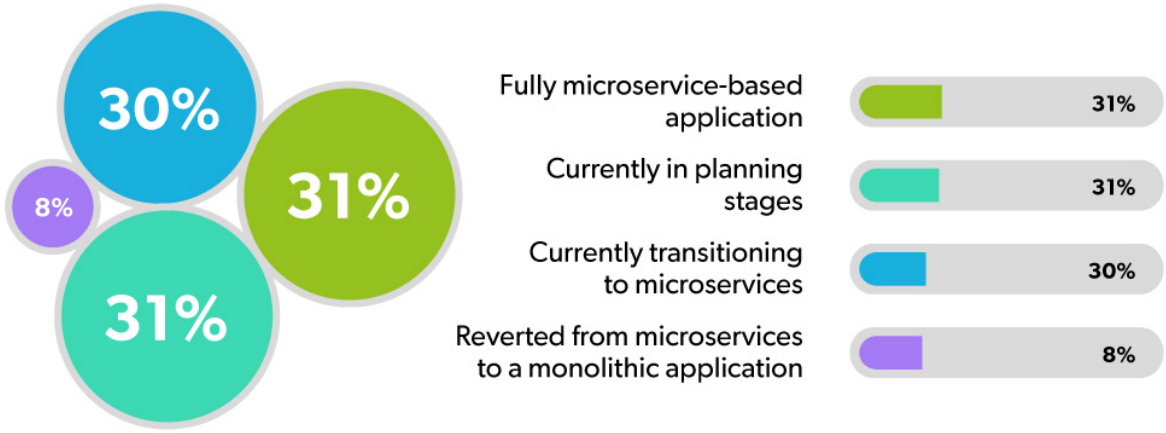
Microservices Trends

For respondents who reported using microservices, we asked a series of questions related to their status in adopting microservices, how they're using microservices, the number of microservices in their application, and their choice in microservice-based applications.

Microservice Adoption Status

In this question, we asked respondents to share their status for microservice adoption. The responses showed that most organizations either had fully microservice-based applications, were currently in planning stages, or were currently transitioning to a microservices architecture.

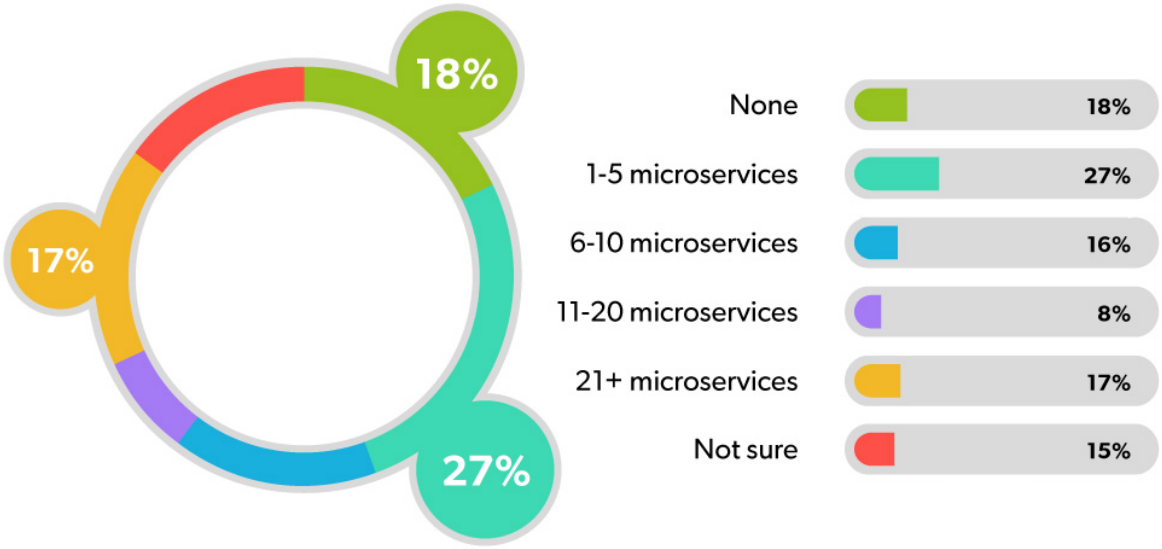
What Is Your Status for Microservice Adoption?



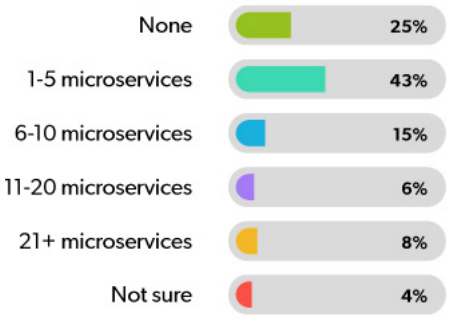
Microservices Per Application

We then asked about the number of microservices that comprise the respondents' microservice-based application. 27% reported having 1–5 microservices, while 16% reported having 6–10 microservices. At the higher end of the scale, 8% indicated having 11–20 microservices, and 17% indicated having more than 21 microservices. Those who reported not having insight into the number of microservices used in their application accounted for 15% of the total, while 18% reported having none.

How Many Microservices Do You Have in Your Primary Application?



Companies With Under 100 Employees



Companies With Over 100 Employees

Companies With Over 1000 Employees

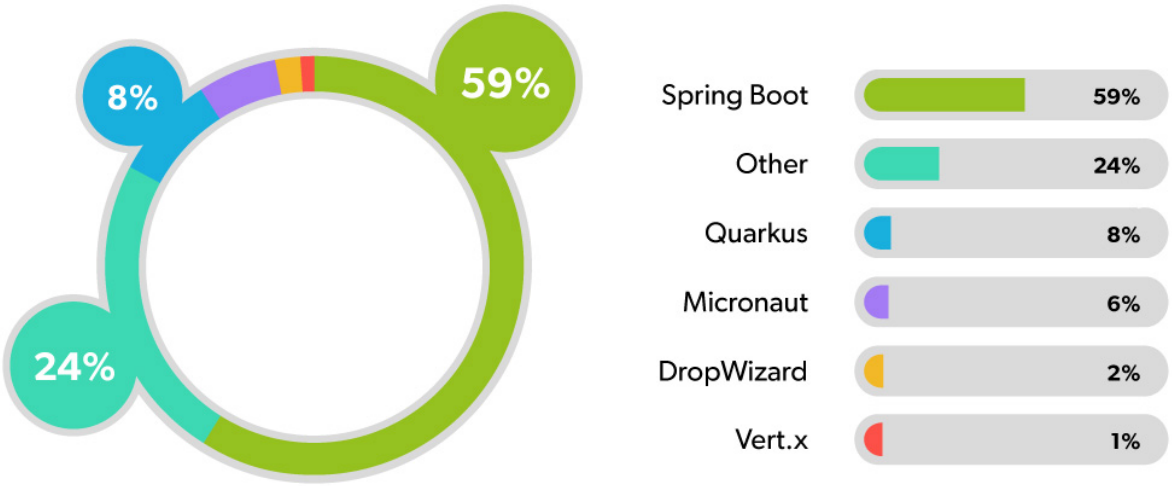
This was an interesting segmentation showing how the size of the company considerably impacted the number of microservice applications used. It makes sense that the larger the company, the larger the amount of microservices utilized. The reality is, microservices have been around for a while, which appears to have allowed larger companies to implement an extensive number of microservices in their applications.

Microservices Framework Usage

In our next question, we asked respondents to share their top choice in microservice application framework. Like last year, Spring Boot remained the top microservice application framework at 59%. Quarkus, Micronaut, DropWizard, and Vert.x rounded out the bottom four at 8%, 6%, 2%, and 1%, respectively.

Common frameworks in the "Other" category included enterprise application specifications like MicroProfile, as well as the Spring framework. However, the most common "Other" response was that they weren't using a microservice framework at all.

What Microservice Application Framework Are You Using on Your Main Project?

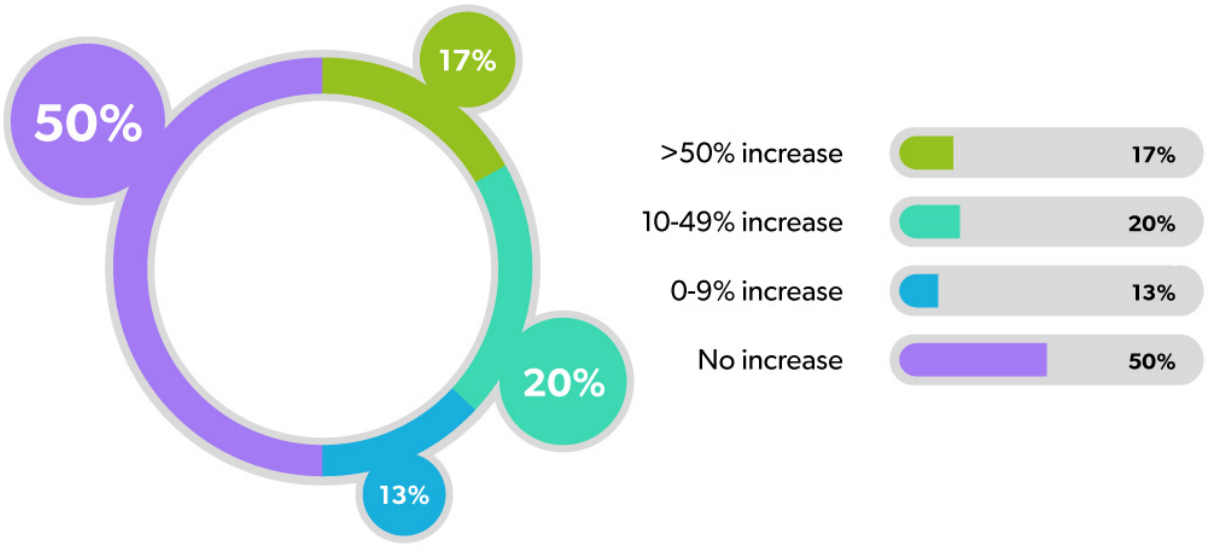


Microservice Application Start-Up Times

In this question, we asked respondents to estimate the percentage increase in start-up time that their microservice-based application has experienced since the time it was created. Our survey found that a combined 50% had experienced an increase, while 50% had not.

Slightly higher than last year's results, 17% of respondents (compared to 13% in 2022) reported over a 50% increase in start-up times for their microservice-based application since its creation.

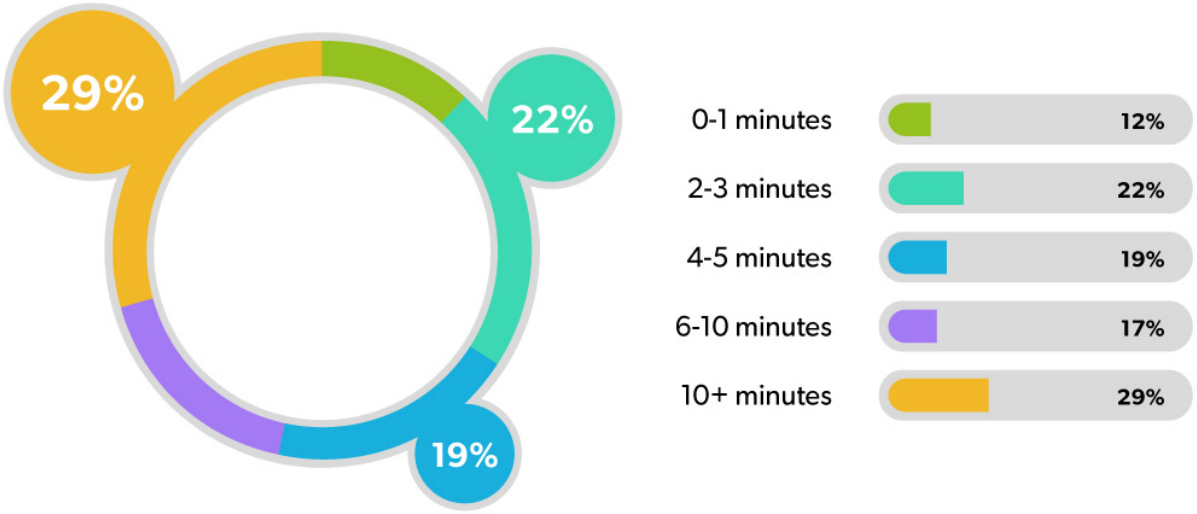
Have You Experienced an Increase in the Time It Takes to Start Up the Services in Your Microservice Application Since the Original Transition/Creation of the Microservice?



Microservice Application Redeployment Times

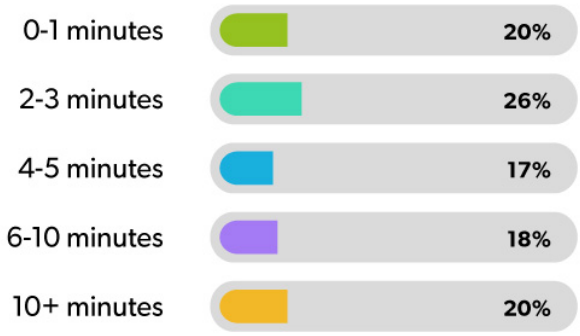
We then asked respondents to weigh in on the time it takes to remotely deploy their containerized environment. Our survey found 46% of respondents reporting a redeployment time of over five minutes, with 29% accounting for 10+ minutes.

How Long Does It Take You to Remotely Deploy Your Containerized Environment?

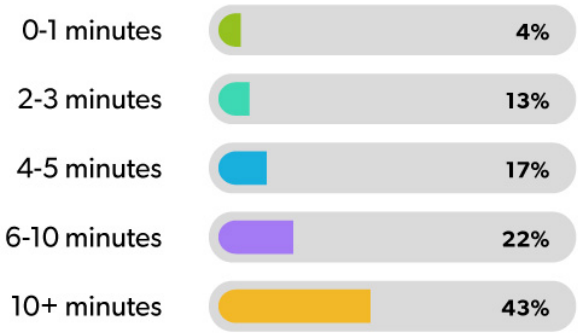


We found larger companies experiencing higher redeploy times, with 65% reporting times of over five minutes compared to 38% of smaller companies.

Companies With Under 100 Employees



Companies With Over 100 Employees



This is always an interesting question, considering the large number of microservices in the Java community that face considerable pain when it comes to the application start-up time and the actual deployment of that code. The complex and detailed architecture to manage these large microservice applications makes it challenging for the average developer writing code every day.

Java Technology Trends

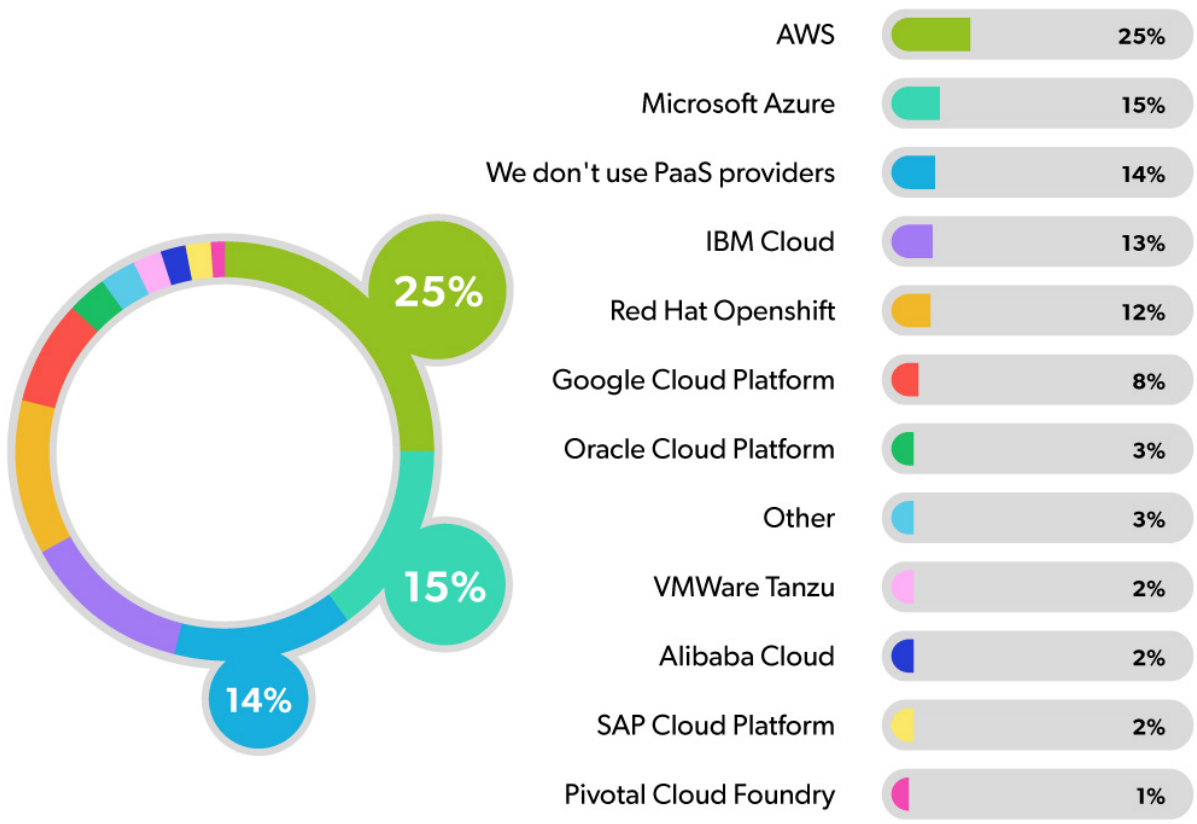
At JRebel, we have a vested interest in staying current with the latest trends in Java development technologies. Every year, we ask Java users to share information about the tools they use to develop Java applications.

This year, we asked teams to share their tool usage for cloud platforms, application servers, and IDEs.

Java PaaS Providers

First, we asked respondents to share which cloud platform or platform as a service (PaaS) they use with their Java applications. Like last year, results showed AWS as the most popular choice at 25%. Azure came in second at 15%, followed by those who don't use PaaS providers at 14%. We saw a rise in IBM Cloud usage over last year's results at 13%, with Red Hat coming in next at 12%.

If You Use a Platform, Who Is Your PaaS Provider?

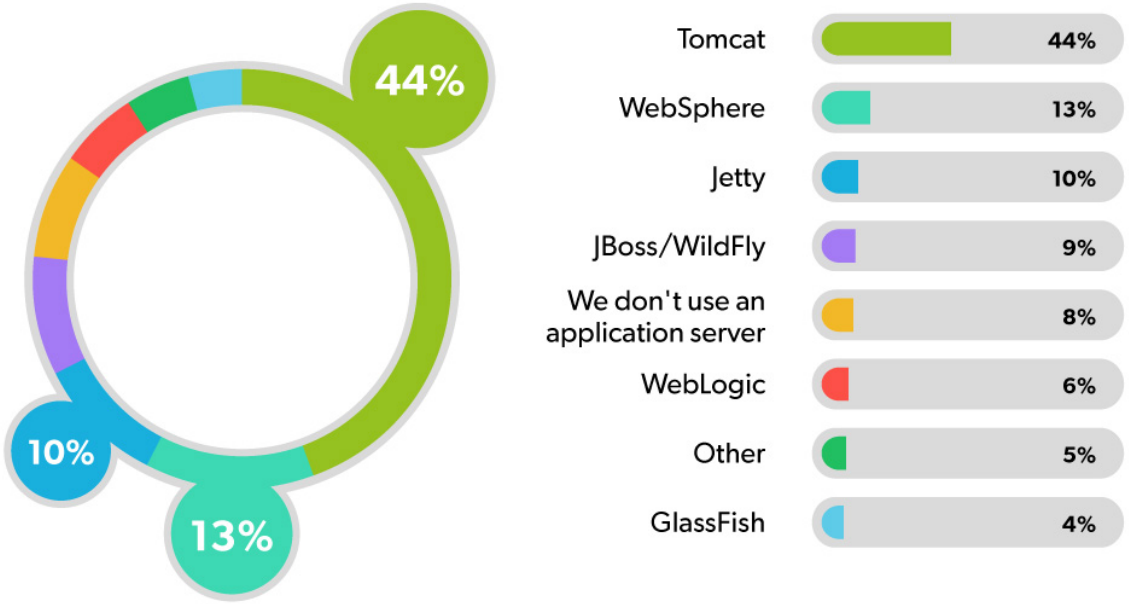


Application Servers

We then asked surveyors to share details on the application server they use in developing Java applications. As with previous years, Apache Tomcat came in at the top — accounting for 44% of all responses. Tomcat was followed by WebSphere (13%), Jetty (10%), JBoss/WildFly (9%), None (8%), WebLogic (6%), and GlassFish (4%).

Among respondents who selected "Other," Payara continued to be the most common choice.

How Do You Configure Most of Your Frameworks?

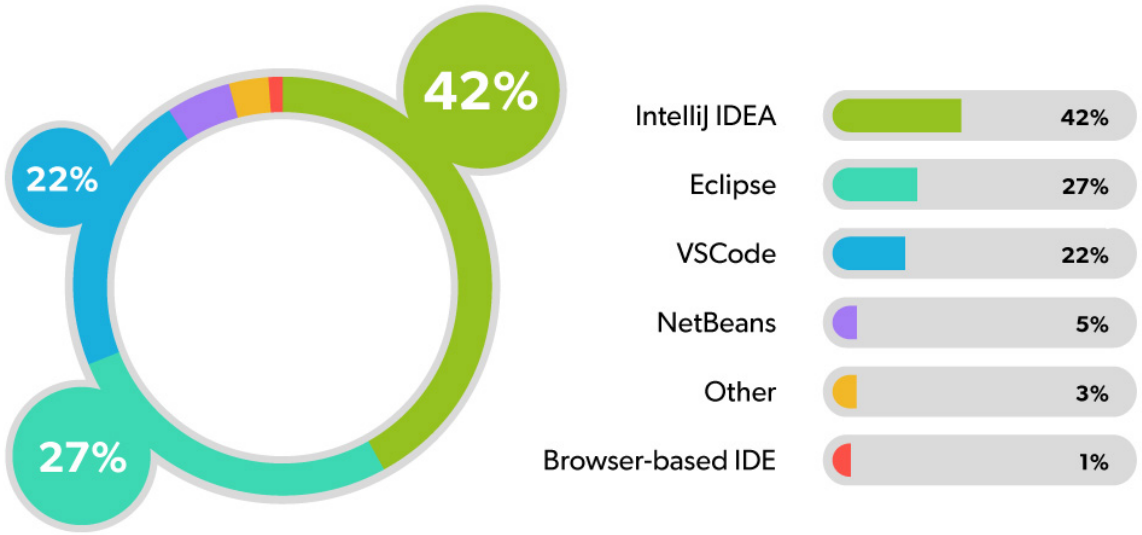


Tomcat once again dominating the application servers used by developers is expected. A surprising element is the amount of WebSphere users. Most of our surveys over the years have kept WebSphere and WebLogic usage around the same result, ranging between 5–8% of respondents. While the growth for WebSphere is unclear, it will be interesting to watch in the future.

Java IDEs

Every year, we also ask respondents to share their preferred IDE. Accounting for 42% of total respondents, Intellij IDEA was once again the most popular choice. Intellij was followed by Eclipse (27%), VSCode (22%), and NetBeans (5%). When it came to combining IDEs (as we allowed respondents to select more than one option), most respondents utilized either IntelliJ and VSCode or IntelliJ and Eclipse.

What Developer IDE Do You Use Professionally?



VSCode has clearly become more relevant in the Java community with its increasing usage amongst Java developers. With more responsibilities falling on the plate for Java developers, they're being forced to use additional IDEs to meet the needs of different aspects of their development practices. VSCode, while showing considerable growth in usage, was mainly used alongside another more established IDE, like IntelliJ or Eclipse.

Developer Productivity Trends

One of our favorite parts of our survey each year is asking about redeploy times. For those that aren't familiar with JRebel, we ask about this because JRebel helps Java developers completely bypass the redeployment process during Java development.

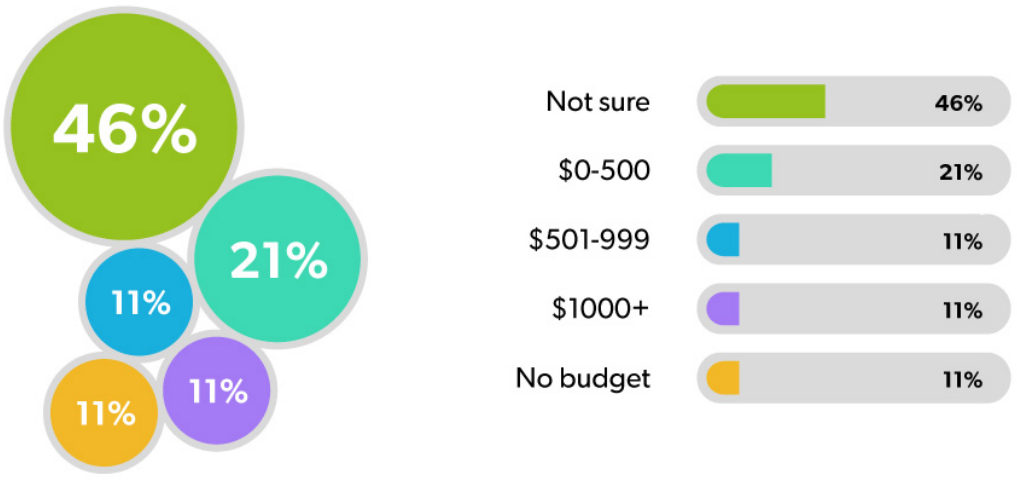
When you consider the time wasted per redeploy, and the number of times the average Java developer redeploys their application every day, this time spent can be a costly impediment to efficient Java development.

In the next questions, we dive in on those numbers — asking developers to share their experiences with redeploy times, their largest obstacles, and what teams would do if they could save the time wasted. We also looked into the estimated budget put into development tools, per developer.

Developer Tool Budget

Developers spend money every single year one way or the other. Some companies will give their developers a say over what tools they use, while others set standards that are passed on to all the developers in the company. You can see how that disparity plays out from our survey — with nearly 45% of the users having some level of a budget, and the rest either don't know or don't have a budget to work with.

What Is Your Estimated Development Tool Budget Per Developer, Per Year?

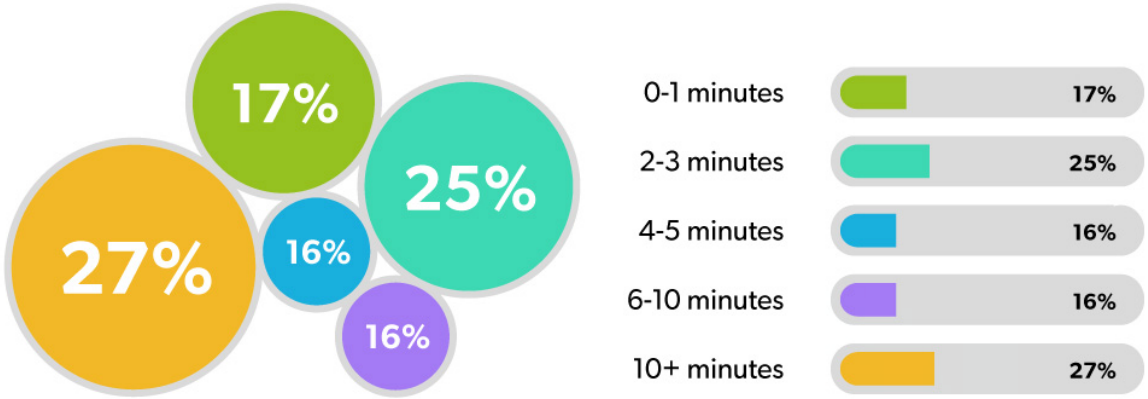


Redeploy Times

Earlier in the report, we shared the redeploy times for teams using microservice-based applications. For this question, we asked respondents to share the average redeploy time for their Java application.

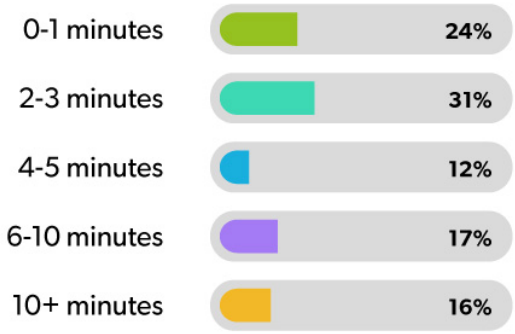
Unlike last year's report in which respondents reported redeployment times between 2–3 minutes as the most common, this year's results showed 10+ minutes as the most common — coming in at 27% in 2023 vs. 21% in 2022. Teams who experienced under four minutes per redeploy represented 42% of all responses, while those who indicated over three minutes accounted for 59%.

After a Code Change in Your Application, How Long Does It Take to Compile, Package, and Redeploy Your Application to a Visibly-Changed State at Runtime?

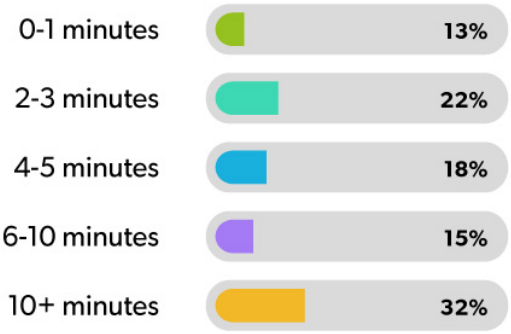


When comparing redeploy times across organizations of different sizes, we found that 67% of smaller organizations (<100 employees) reported redeploy times of under five minutes, compared to 53% of larger organizations (100+ employees).

Companies With Under 100 Employees



Companies With Over 100 Employees

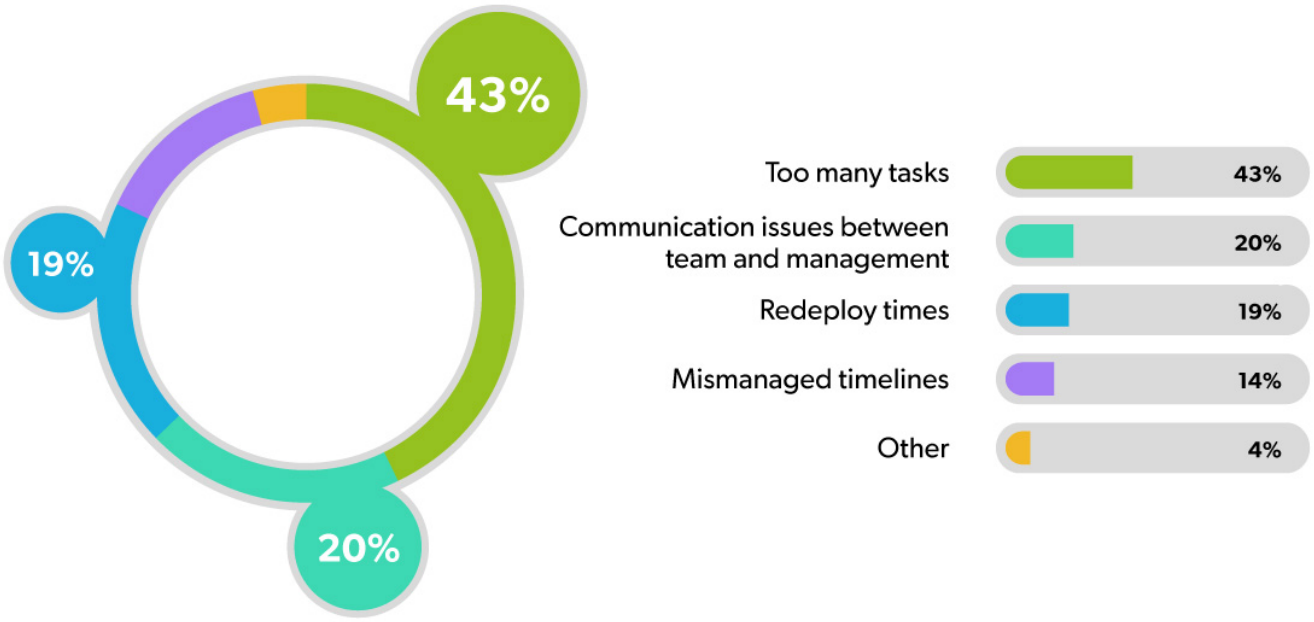


Java developers continue to see large redeploy times throughout their applications. While we expect this pattern to continue, we believe that tools like JRebel will help address these pain points to improve Java developer productivity.

Biggest Obstacles

Every Java developer deals with issues that make their ability to efficiently perform their tasks more difficult. In this question, we dove into what those major obstacles are.

What Is the Biggest Obstacle Your Team Faces to Increasing Productivity?



Java developers clearly feel that there are too many tasks put on their plate, which appears to be impacting their overall productivity. This makes a lot of sense and is also very telling for team leads on what's impacting the overall coding experience.

What Teams Would Do Without Redeploys

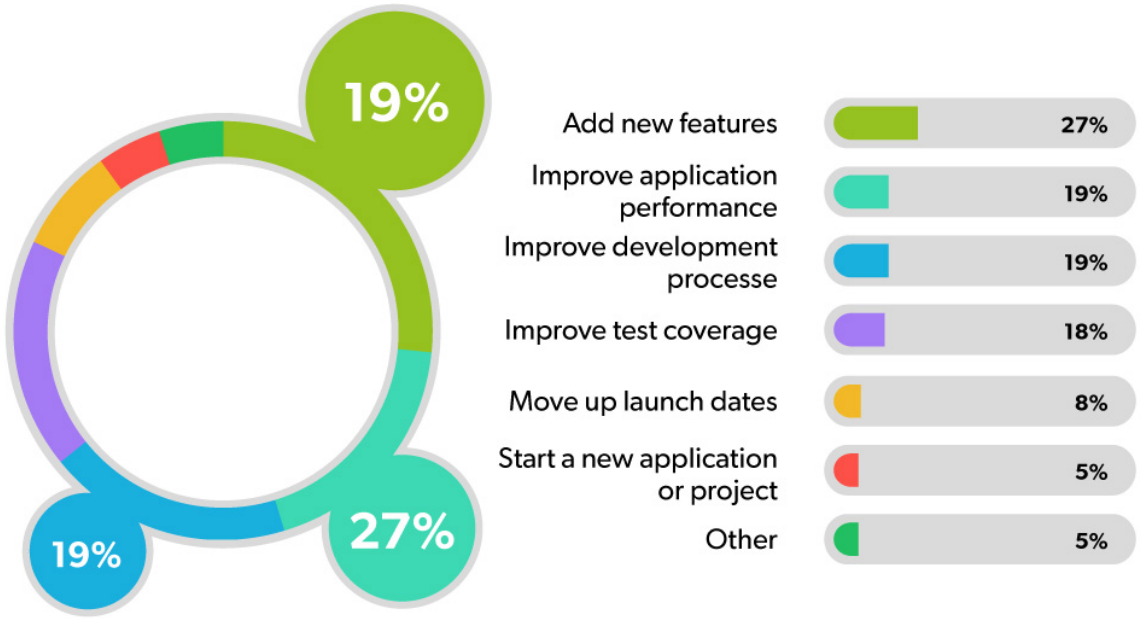
As in previous years' surveys, we wrapped up our survey by asking respondents what they would do if they could save 10% of their development time each workday.

While most people pointed to practical uses, like adding new features (27%) or improving their development process (19%), we're inclined to point out some of the "Other" responses. In all fairness, taking regular breaks and maintaining a healthy work-life balance are also important contributors to overall productivity.

Highlights included:

- Coffee break
- Eat lunch
- Leave early
- Watch cats on the internet

If Your Team Could Save 10% More Time During the Workday, What Would They Do With That Extra Time?



Closing Thoughts

In the 11 years that JRebel has produced the Java Developer Productivity Report, there has been a constant influx of new innovations, technologies, and trends. It's also expected that with the steady march of innovation comes both new and familiar challenges.

Redeploys remain a constant dilemma that many developers face on a regular basis. Eliminating those redeploys will continue to be an area of focus we hope to meet the need of in 2023 and beyond. With the continued overall growth in the Java community, we look forward to monitoring these results — and more.

A special thanks to all the developers, team leads, and management that took the time to fill out our survey and help make this report a success. We hope you enjoyed reading the report as much as we enjoyed making it. If you have any feedback, please feel free to reach out directly to JRebel's product manager, Curtis.

About JRebel

JRebel is a Java developer productivity tool that allows developers to skip redeploys while maintaining application state.

With over 3000 customers around the world, JRebel is trusted by leading brands to improve their Java development productivity.

Interested in seeing how JRebel works on your projects? Try it free for 14 days with no commitment.

TRY JREBEL FOR FREE

www.jrebel.com/products/jrebel/free-trial