The Largest Survey Ever of Java Developers

What 10,500 Java developers tell us about their projects, their tools, and themselves

by Andrew Binstock and Simon Maple

The data presented in the following report was taken from more than 10,500 questionnaires. If you were one of those survey-takers, many thanks for putting aside the time to share your experience for the benefit of others. Snyk and Java Magazine conducted the survey by publishing its availability to the Java community at large, to Java user groups (JUGs), and to subscribers of Java Magazine. As an inducement to complete the survey, Snyk and Java Magazine promised a contribution to Devoxx4kids.

Information regarding respondents, including geographical location, company size of their employer (if any), age, and experience with Java are presented at the end of this survey.

About Your JDK

1. Which Java vendor’s JDK do you use in production for your main applications?

We start the report with a core question. With so many vendors providing their own JDK implementations, which offerings are developers using in production for their applications? We can see the dominance that Oracle JDK and OpenJDK have over everyone else. With 7 in 10 developers opting to use the Oracle JDK and a further 2 in 10 opting for the OpenJDK, there isn’t much competition. However, future licensing and support changes might cause these numbers to differ in the future.

2. How do you plan to respond to Java’s new release cycle?

While the Java 9 release brought with it some major architectural changes, it also introduced a new release cadence in which Java SE versions ship every six months. Every two to three years, a long-term support (LTS) release offers longer-term support, such as security updates. This question asks how development teams will respond to the new release cadence. The responses were varied, suggesting there is still some uncertainty about how to proceed. In fact, almost 1 in 3 developers don’t yet know how they will respond to the new release cycle.

We expect that in the forthcoming year, best practices will emerge, and companies will settle into a preferred migration cycle, which likely will
3. What Java EE version do you use for your main applications?

Almost 4 in 10 respondents do not use enterprise Java. Of those who do, the majority are on version 7. Java EE 8 was released in September 2017, so it’s promising to see that within less than a year of release it is almost the most popular version. Unlike Java SE, a release of Java EE takes longer to be adopted, because it takes longer for implementations to become available. In addition, app server vendors require time to adopt and implement the specifications, so to see such user adoption after less than nine months is a promising sign.

We’ll be keen to see how these numbers shift as Jakarta EE 8 begins its rollout and adoption. We should spare a thought for the 2% who are struggling on J2EE—a version whose last major release was in 2003.

4. What is the principal JVM language you use for your main applications?

Exactly 9 in 10 JVM users are using Java for their main applications. While many projects today are defined as multilanguage or polyglot, the part on the JVM primarily runs Java.

Despite this strong preference for Java, JVM-based developers have consistently shown great interest in other JVM languages, as supported by the popularity of articles about them in Java Magazine and on major programming sites. For the last few years, the emerging and “hot” JVM language has been Kotlin, from JetBrains, which continues to make steady progress. It is now a supported language for development on Android, and it is the second major language for writing scripts for the build tool Gradle (behind Groovy). All this adoption has helped move Kotlin past Scala and just past Groovy in our survey.

The 3.0% figure for Clojure is remarkably high and signals—to us, at least—the continued interest in functional programming. The functional orientation of many Java 8 features shows the imprint of functional programming on Java itself.
5. Which IDE do you develop with?

The graph on the next page is consistent with other recent surveys: IntelliJ passing Eclipse in the last one to two years, and Apache NetBeans staying at around 10% of the market. The 45% total votes for IntelliJ consist of 32% IntelliJ IDEA Ultimate Edition (the paid version), 11% IntelliJ Community Edition (the free version), and 2% Android Studio users. The Eclipse category includes Eclipse STS, JBoss tools, Rational Application Developer, and other Eclipse-based tools.

The numbers of Apache NetBeans users haven’t changed much, which suggests that the migration from Oracle to the Apache Software Foundation hasn’t affected its user base. It’s also worth mentioning that Visual Studio Code has made an appearance, which, although it is at only 1%, shows it’s starting to make its mark in the Java community.

Also, a tip of the hat to the “vi/vim/emacs/etc.” group, who are probably reading this report on a tablet (carved out of stone).

6. Which build tool do you use for your main project?

In examining the numbers, it’s important to note that we asked for the principal build tool used. We want to know which build tool you rely on for your main project. It can also be interesting to know which build tools teams use across all their projects, but the plethora of answers tends to dilute the usefulness of the responses. As we can see here, Maven clearly dominates with a 3:1 ratio over its closest rival, Gradle. Ant is still in use by 1 in 10 developers, whereas 1 in 20 use nothing!

In a 2016 version of a similar survey by RebelLabs (2,000 respondents), Maven stood at 68% and Gradle at 16%. Gradle’s improved adoption rate is due perhaps to the addition of support for Kotlin as the scripting language. Gradle is also the default build engine for Android projects. However, its progress against Maven has been slow.

7. Which static quality tools do you use?

Before you send that email or tweet trying to fix the internet, note that this is a multiple-choice question, so numbers here do not add up to 100%. One takeaway from this data is that there are only a handful of static quality tools in popular use, with no real surprises at the top: SonarQube, FindBugs, and Checkstyle dominate. We were surprised to see that 36% of respondents don’t use any static quality tool whatsoever. We assumed that the use of these tools was the norm.
8. Which static security tools do you use?

Most sites don’t use static security tools. It’s a surprise to see how low overall adoption is, considering the widely publicized costs of security issues. For sites that do use them, the older tools dominate: Sonatype and Fortify lead the market. Snyk makes its first appearance on our surveys and provides full automated remediation across many ecosystems. We hope a wider adoption of security tools will appear in future surveys.

![Static Security Tools Usage Chart]

9. Which CI server do you use?

As most developers would expect, Jenkins wins the CI server race with a whopping 57% market share. Its closest competitor is “None” at 21% of the vote, which almost matches the rest of the competition combined (at 22%). The remaining CI servers have less than 5% of the market share each, with Hudson, the elderly relative of Jenkins, weighing in at 2%. It’s worth mentioning VSTS (Microsoft Visual Studio Team Server), which is not usually thought of in the Java/JVM space, clocks in at 2%.

Most developers expect that nearly all sites today use continuous integration. So, it’s startling to see that 1 in 5 applications rely on none at all. Even personal projects today use CI (such as Travis CI and CircleCI), which are made available on public project-hosting sites such as Bitbucket and GitHub. If you’re one of the 21% who don’t use CI on your projects, we’d love to hear why.

![CI Server Usage Chart]

10. Which source code management platform does your team use for your main project?

As expected, Git has convincingly won the horse race in source code management. If you were in any doubt as to the extent of its dominance, almost 3 in 4 of our respondents work in teams that use Git to manage their codebases. Subversion now covers the majority of the remaining respondents; and somehow in 2018, 3% of people still don’t use source code management whatsoever. Sometimes, there are no words.

![Source Code Management Platform Usage Chart]

11. Which code repository do you use for your main project?

With code repositories, the story is quite different from source code management: much more spread out, with GitHub and Bitbucket neck and neck at 25% each and GitLab close behind at 20%. We could call those the “big three” of project hosting. Note that this question is not just for
public projects (in which we expect GitHub would have a more significant lead), but also for public and private project hosting.

Microsoft’s recent acquisition of GitHub might affect its future adoption rate, and we’ll know more in future surveys. Of the 25% share that GitHub has, just over half (52%) of those respondents are using the public version, whereas the remainder (48%) are using the private GitHub Enterprise on-premises offering. VSTS makes up part of the “Other” bucket with 2%.

12. Which private binary/artifact repository do you use?

Most sites don’t use a packaged artifact repository—in theory because they don’t have the need. Those that like the convenience it offers choose the well-established Nexus, which is heavily focused on the JVM ecosystem, followed by JFrog’s Artifactory, which is somewhat more popular across polyglot ecosystems.

13. Which testing technologies do you use?

With an amazing (almost) 4 in 5 people using JUnit and TestNG used by 10% more, it’s clear that unit testing is by far the most dominant testing practice in the JVM ecosystem. (Respondents could choose multiple answers, so totals exceed 100%.) As to mocking, it’s now clear that Mockito has emerged as the preferred mocking framework.

With JMeter being used by almost 1 in 4 respondents and Gatling by 5%, we can see that the need for performance testing is becoming better appreciated. Selenium takes an impressive 29%. Unlike the results for static quality tools, only 10% of respondents say they don’t use any testing tool whatsoever. Hang on…1 in 10 people don’t use a testing tool? We should move on before we lose faith in humanity.

About Your Cloud

14. Which cloud approaches do you use?

Containers lead the way at 43%, while VMs stay in the game at 33%. As a relatively new technology, serverless/FaaS comes in strongly, with almost 1 in 10 respondents adopting this approach. PaaS, which has been around a lot longer, sits on a similar split at 10%. 1 in 3 respondents don’t use any cloud approach at all.
15. Which continuous deployment or release automation tools do you use?

Nearly half the respondents don’t use any continuous deployment or release automation tools whatsoever. This might even be higher, because almost 1 in 5 don’t have any idea which tools are used in CD or release automation. It’s somewhat surprising to see bash more popular than Chef and Puppet. Actually, who are we kidding? Everyone loves bash! Ansible is the leading CD tool at 16%.

About Your Application

16. Which other (non-JVM) languages does your application use?

In today’s polyglot world, it would be naive to assume that JVM languages are the only languages used in JVM apps. In fact, more than half of JVM applications use front-end JavaScript, 1 in 5 use Python, and almost 1 in 4 use Node.js. As you’d expect, many projects use SQL as well.

17. Which web frameworks do you use?

Few words can better express the Spring domination in the Java ecosystem than this graph. With 4 in 10 developers using Spring Boot in their applications, it’s interesting to see it has overtaken the Spring MVC framework for the first time. JSF is the closest entrant with a respectable 19% and Struts, despite a constant stream of remote code-execution vulnerabilities in the news, is a strong fourth with almost 1 in 10 developers adopting it. More than 1 in 5 developers likely boast about how small their applications are, not needing a web framework at all.

18. Which ORM frameworks do you use?
More than 1 in 2 developers use Hibernate in their applications. Almost 1 in 4 developers are happy with plain old JDBC, and Spring developers of course have the option of using Spring JdbcTemplate, which is used by 23%. 1 in 5 developers don’t use any ORM framework whatsoever to access their data. (Developers could choose more than one answer, so totals do not equal 100%.)

19. Which database do you use in production?

Once again, Oracle Database takes the top spot, with almost 3 in 10 applications using it in production. MySQL and PostgreSQL are strong competitors, taking 21% and 20%, respectively. MongoDB is the highest NoSQL database in use, with 5%.

20. Which application server do you use in production for your main application?

More than 4 in 10 respondents use Apache Tomcat as their application server of choice. The fast, lightweight, open source, community favorite has led the pack for a long time now, and it doesn’t look like that’s going to change any time soon. JBoss and WildFly are not too far behind at 15%. In the larger enterprise app server category, Oracle WebLogic has a slight lead over WebSphere. The “Other” category contains Apache TomEE and Liberty Profile at 1% each, which lead that group.

21. Do you develop on the same application server you use in production?

Despite the obvious dangers, more than one-third of respondents develop on a different server from the one they use in production—trading the possible cost of failures for the convenience. Surprisingly, those who state they use different application servers (or none) in development actually have a wide variety of apps and servers in production. We were expecting mostly the larger monolith-suited app servers that could cause developers pain to use locally, but the ratios were comparable.
22. How many open source (direct) dependencies does your main application have?

It would be interesting to know how many people had to check to see how many direct dependencies their application has. We’d bet it was the vast majority of you. It’s a good thing we didn’t ask for direct and transitive dependencies too! In fact, almost 1 in 4 respondents openly state they don’t know how many dependencies they have. This might be because of the way the application is distributed across a more complex build system. We can see from the results that fewer than 1 in 20 respondents don’t use any open source dependencies, whereas the overwhelming 72% do. If we remove those who don’t know, we can see that 95%, or 19 of 20 respondents, use open source dependencies in their applications. This shows how far open source adoption has come, as well as the need for us to ensure these third-party libraries provide the security, quality, and availability required from an application.

![Chart showing the distribution of open source dependencies]

About Your Processes

23. How often do you release new versions of your code?

Not shown in this graph is that almost 1 in 4 respondents (24%) don’t know how often their code is released. This again might be due to the complexity of the application, and perhaps because different services are being released at various times.

Almost 1 in 10 respondents are brave enough to release multiple times a day, but then again, when you release that often, bravery is replaced with consistency. The majority of respondents release once every couple of weeks to once a month. Fewer than 1 in 10 respondents release once every six months or less.

![Chart showing the frequency of code releases]

24. How often do you audit your code?

About half the sites audit their code. Only 1 in 4 do so more than once a quarter. Whether the audit is for security, performance, or quality, it’s good to have a clear out. Half of the respondents do not audit their code whatsoever. Imagine the gremlins that could exist in those codebases!
About You

25. Where do you do your development work?

Not much to say here, other than we expected more respondents to come from North America.

26. How would you describe yourself?

We can see the vast majority of respondents are technical, with 87% being either developers, team leaders, or architects. More than half state they are software developers, and 2% of respondents were C-level who took the time to fill out our survey.

27. How do you rank your security expertise?

Security is often considered one of the dark arts. Many developers learn only “just enough” to get by and deliver their feature work, meaning the real experts are those who have a dedicated security career. As developers are owning more and more application security responsibility, this is becoming a hot topic. In our survey, 1% of respondents state they have zilch, none, zero security knowledge and are just happily writing their applications. The same number state they are true security experts and gave themselves 10 out of 10. The majority sit around the 5 to 7 mark, with 6 in 10 respondents stating they’re no expert, but certainly not novices.

28. How old are you?

Programming remains a profession associated with the young and early middle-aged. 37% of survey respondents are younger than 35, 35% are between 35 and 45, and only 25% are older than that. Survey data shows a correlation between positions of greater responsibility and age, suggesting that the lower numbers after middle age are in part due to programmers moving into management positions.
29. How many years of paid professional experience with Java do you have?

There was quite a range of experience among our respondents, as you can see from the graph, but we wanted to also look at the median ages of job roles back from question 26, to see when people typically receive promotions. The results are very interesting, with the median developer having 10 years of experience, team leaders 11 years, architects 14 years, and finally C-levels with 12 years.

30. What is the size of your company?

With almost 40% of respondents working for companies that have fewer than 100 employees, we see that Java continues to have a significant role in startups and in small-to-medium businesses. This finding is at odds with the perception of Java being the language for enterprise apps. It is that, certainly, but definitely more than that.

31. Where do you principally get information about Java online?

Stack Overflow remains the preferred forum for asking one-off questions and grepping replies to previous answers for useful information. Oracle’s excellent documentation is a natural place for reference. Java Magazine presents long-form, in-depth articles for readers wishing to fully understand a topic, and YouTube does the equivalent in video form. DZone and to a lesser extent InfoQ overlap all these areas. (Respondents could choose multiple answers.)

32. Are you a member of a Java user group (JUG)?

JUGs remain an underused resource in the community, with only 1 in 5 developers attending one meeting a year. The overwhelming two-thirds of respondents are not members of any JUG whatsoever. If geography is a factor for those developers, the Virtual Java User Group is an excellent solution.
33. How much do you contribute to open source?

Open source remains what it has primarily been from the start: the domain of a small minority of dedicated developers. The existence of GitHub and other easily accessible code repositories has helped developers to contribute their code, but with more than 1 in 2 developers never having contributed to an open source project, there’s still a lot of work to be done. You should want to contribute back so that rather than just being one of the 19 in 20 developers using open source, you can one day be one of the 19 in 20 developers contributing to open source.

These survey results were originally published in the November/December 2018 issue of Java Magazine.

Andrew Binstock

Andrew Binstock (javamag_us@oracle.com, @platypusguy) is the editor in chief of Java Magazine. Previously, he was the editor of Dr. Dobb’s Journal. He co-founded the company behind the open-source iText PDF library, which was acquired in 2015. His book on algorithm implementation in C went through 16 printings before joining the long tail. Previously, he was the editor in chief of UNIX Review and, earlier, the founding editor of the C Gazette. He lives in Silicon Valley with his wife. When not coding or editing, he studies piano.

Simon Maple

Simon Maple (@sjmaple) is a Java Champion and director of developer advocacy for Snyk, a company whose products find and fix vulnerabilities in project dependencies. He is also the founder of the Virtual Java User Group (vJUG). Earlier, he was director of developer relations at ZeroTurnaround. Previous to that, he held a similar role at IBM.