new relic

Status quo des Java-Ökosystems 2024

Trends und Daten rund um eine der populärsten Programmiersprachen

Bericht / April 2024

Inhalt

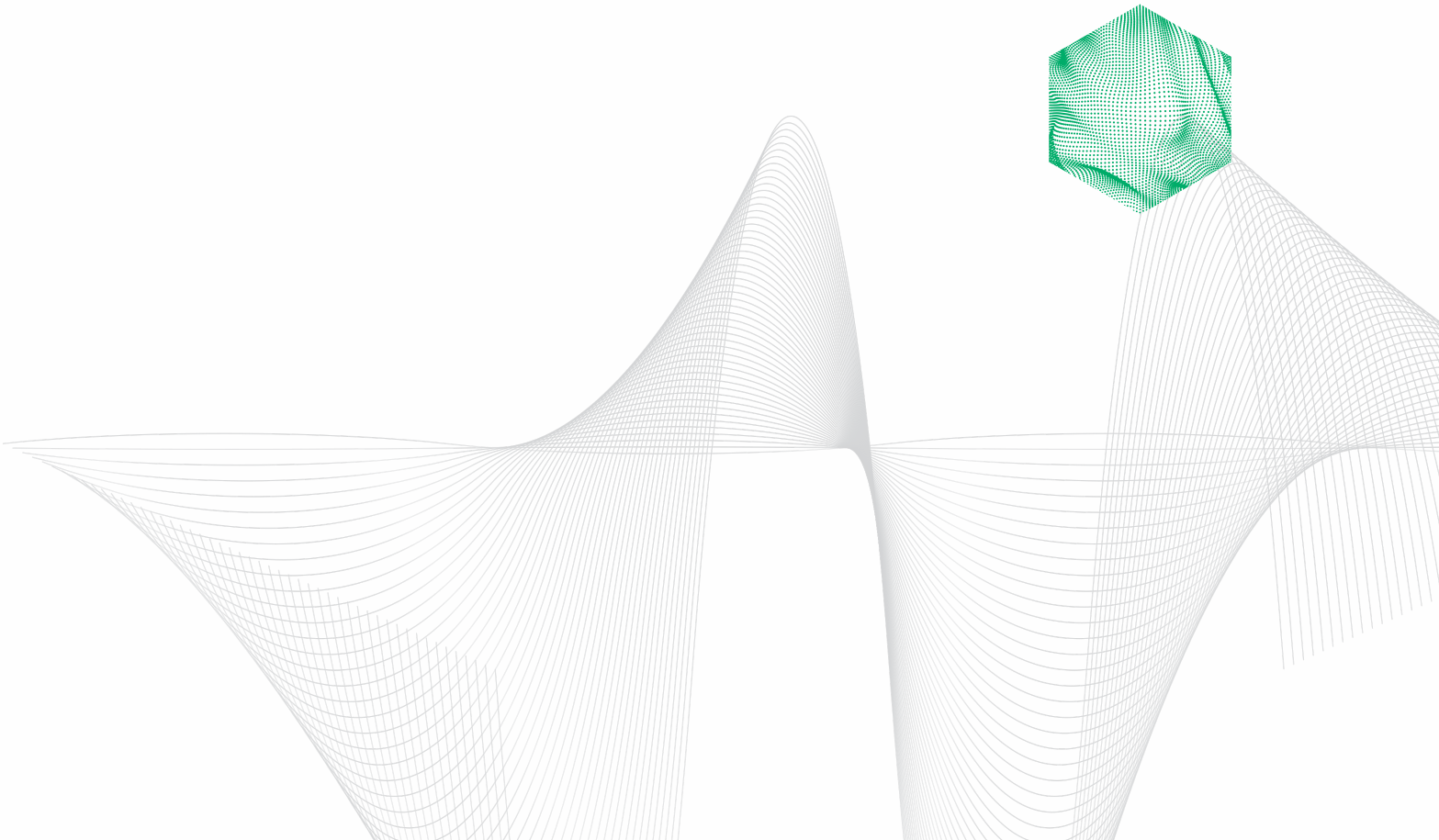
Überblick

New Relic hat das Java-Ökosystem in den letzten Jahren genau beobachtet, um herauszufinden, wie sich die Nutzung von Java durch Entwickler:innen verändert. Unter anderem hat New Relic untersucht, ob Java 21 schneller angenommen wird als andere Java-Versionen, ob es Trends bei den von Anbietern und der Community unterstützten Java Developer Kits (JDKs) gibt und welche Arten von Java-bezogenen Fragen von Entwickler:innen gestellt werden, die den GenAI-Assistenten für Observability von New Relic nutzen.

Dieser Bericht bietet Kontext und Einblicke in den Status quo des Java-Ökosystems, basierend auf den Daten von Hunderttausenden von Anwendungen, die jeden Monat an New Relic gemeldet werden.

Für den Jahresbericht 2024 wurden insbesondere folgende Bereiche untersucht:

- Die in der Produktion am häufigsten eingesetzten Java-Versionen
- Die beliebtesten JDK-Anbieter
- Rechen- und Speicherressourcen in Java-Anwendungen
- Gängige Java-Frameworks und -Bibliotheken für Logging, Verschlüsselung und Datenbank
- Häufige Fragen und Anliegen zu Java von Entwickler:innen



Neue Java-Versionen werden schneller angenommen

Werfen wir zunächst einen Blick auf die in der Produktion am häufigsten verwendeten Java-Versionen.

Früher hat Oracle neue Versionen nur bei größeren Updates und Änderungen im JDK veröffentlicht. Jetzt gibt Oracle alle sechs Monate eine neue Java-Version heraus – in der Regel im März und September – und jedes Release enthält neue Funktionen und Bugfixes. Alle zwei Jahre veröffentlicht Oracle eine neue LTS(Long-Term-Support)-Version von Java mit Updates zur Verbesserung der Stabilität, Sicherheit und Performance. Dies ist für Entwickler:innen häufig einer der wichtigsten Faktoren für ein Upgrade der Java-Version.

Im September 2023 brachte Oracle Java 21 auf den Markt. Dies war ein bedeutender Meilenstein für Java mit deutlichen Verbesserungen an Vorschaufunktionen in Form von virtuellen Threads und aktualisierten Bibliotheken sowie mit Neuerungen an der Syntax, die Java auf Augenhöhe mit vielen moderneren Sprachen bringen.

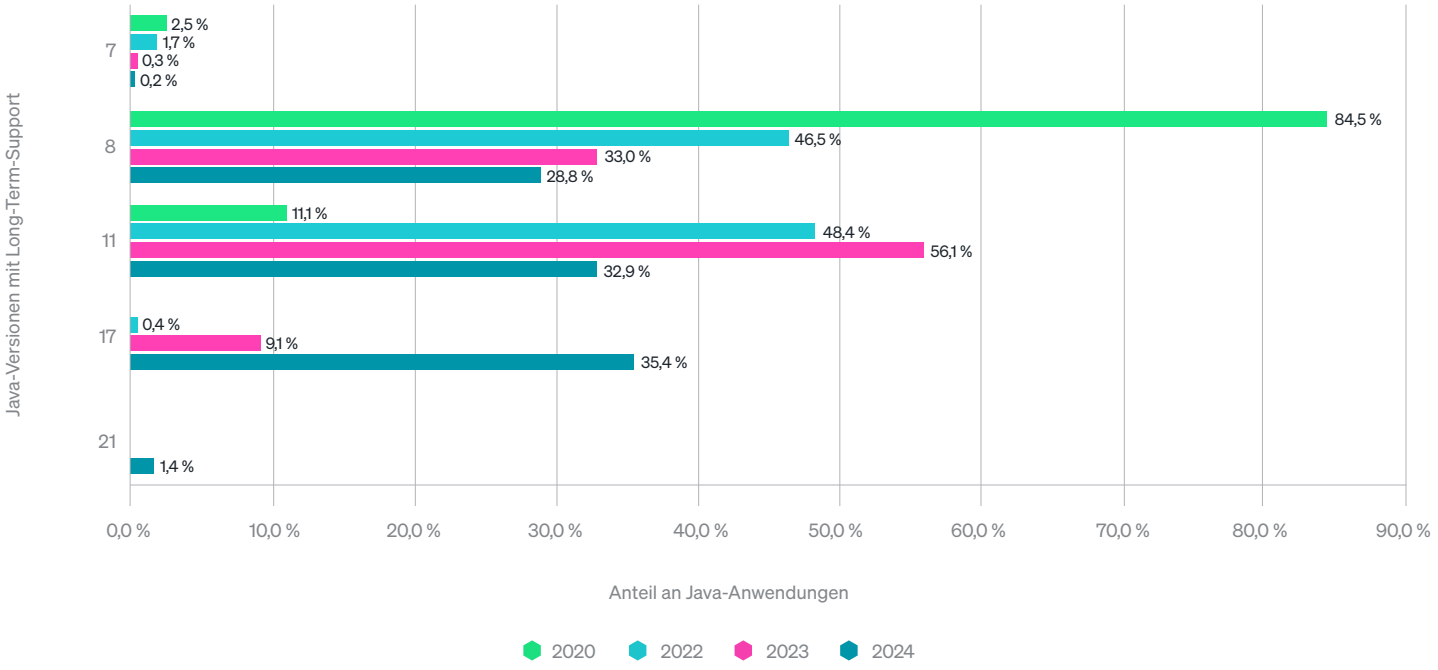
In den sechs Monaten nach der Veröffentlichung von Java 21 verwendeten 1,4 % der von New Relic überwachten Anwendungen Java 21. Zum Vergleich: In den sechs Monaten nach der Einführung von Java 17 nutzten nur 0,37 % der Anwendungen Java 17, also 287 % weniger.

35 %

aller Apps nutzen Java 17

Dabei übertraf die Nutzungsrate von Java 17 bei weitem das, was die Entwicklungs-Community bei der Einführung von Java 11 erlebte. Noch 2023 nutzten 9 % der Anwendungen Java 17 in der Produktion – mittlerweile sind es 35 %. Das entspricht einer Wachstumsrate von fast 300 % innerhalb eines Jahres. Es hat Jahre gedauert, bis Java 11 auch nur annähernd dieses Niveau erreichte.

Weniger als 2 % der Anwendungen nutzten Nicht-LTS-Versionen von Java, was verständlich ist, da diese in der Regel nicht in der Produktion eingesetzt werden.



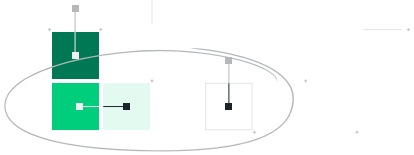
Nutzung von Java-Versionen mit Long-Term-Support nach Jahr

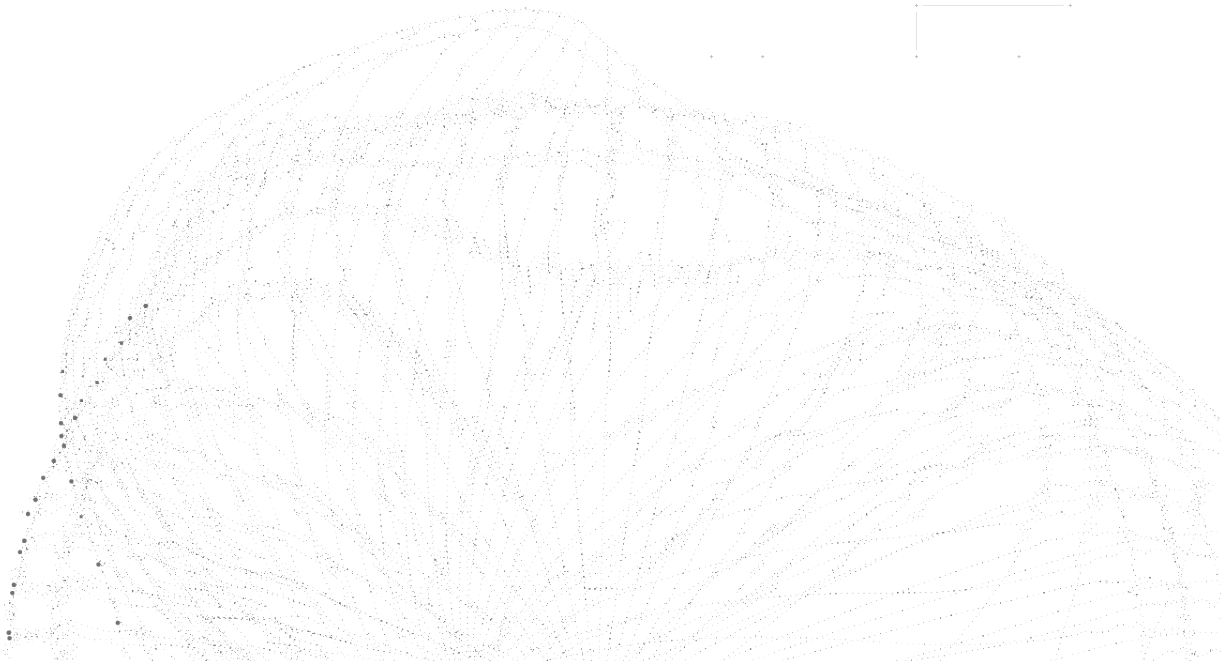
Steigende Popularität von Eclipse Adoptium unter JDK-Anbietern

Sehen wir uns nun die gängigen JDK-Anbieter an.

Java war früher Closed-Source oder proprietär, d. h. Entwickler:innen konnten das JDK nur direkt von Sun Microsystems herunterladen. Doch etwa zehn Jahre nachdem Sun Microsystems die erste Version von Java veröffentlicht hatte, brachte Oracle die erste Open-Source-Version von Java heraus. OpenJDK pflegt diese Version und gestattet Entwickler:innen sowie Anbietern wie Microsoft und Amazon die Pflege anderer Versionen des JDK.

2020 war Oracle der beliebteste JDK-Anbieter mit einem Anteil von etwa 75 % am Java-Markt. Nach der restriktiveren Lizenzierung von JDK 11 gab es eine spürbare Abkehr von Oracle-Binärdateien (bevor man mit Java 17 zu einer offeneren Haltung zurückkehrte), und seitdem ist ein stetiger Rückgang im Jahresvergleich zu verzeichnen. Während Oracle im Jahr 2022 seinen Spitzenplatz noch behaupten konnte (34 %), lag der Marktanteil 2023 bei 29 % und heute bei nur noch 21 % – das entspricht einem Rückgang um 28 % innerhalb eines Jahres.



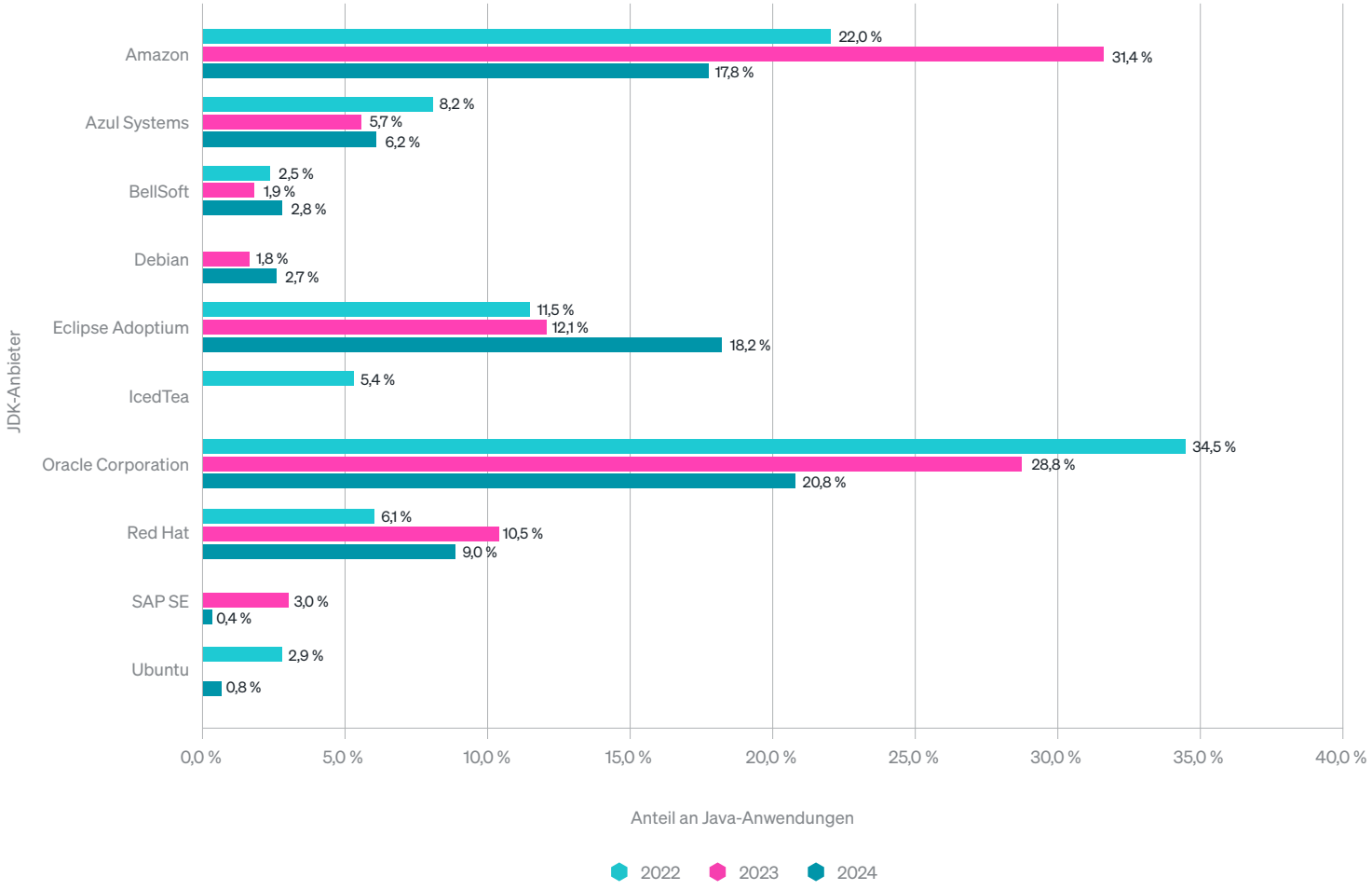


Amazon vergrößerte seinen Marktanteil 2023 auf 31 % (von 2,2 % im Jahr 2020 und 22 % im Jahr 2022). Dieser sank jedoch 2024 auf 18 % – ein Rückgang von 43 % im Vergleich zum Vorjahr.

Der aufsteigende Stern in diesem Jahr ist Eclipse Adoptium, dessen Verbreitung im Vergleich zum Vorjahr um 50 % von 12 % auf 18 % gestiegen ist. Da Eclipse Adoptium von der Community verwaltet wird, wird dieses JDK tendenziell häufiger aktualisiert als die JDKs von Oracle und Amazon.

18 %

verwenden das JDK von Eclipse Adoptium



Beliebteste JDK-Anbieter nach Jahr

Die häufigsten Java-Anwendungskonfigurationen

Nun möchten wir uns mit der Verwendung von Garbage Collectors, Rechenleistung und Speicherplatz in Java-Anwendungen befassen.

43 %

verwenden den G1 Garbage Collector

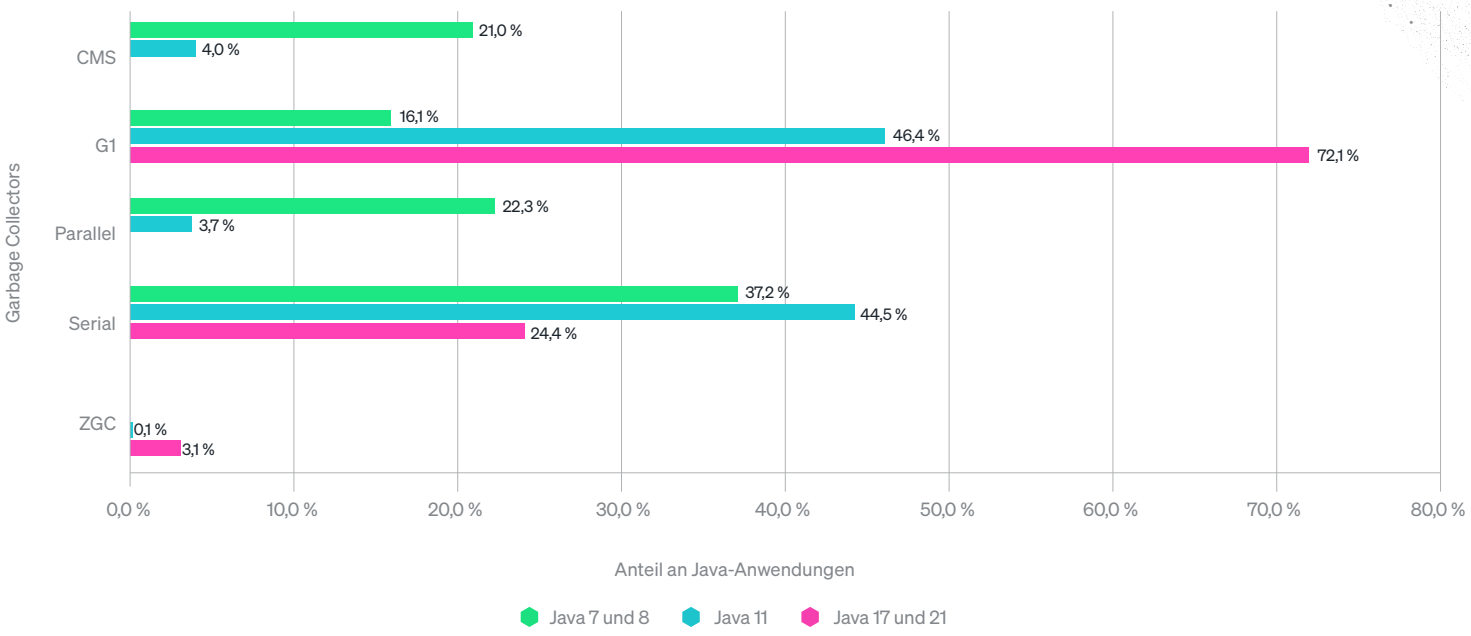
Garbage Collectors – wer entsorgt den Abfall?

Java Garbage Collectors (GCs) sind Speicherverwaltungskomponenten, die dazu dienen, Speicherlecks zu verhindern, die Speichernutzung zu optimieren und insgesamt die Performance und Stabilität von Java-Anwendungen zu gewährleisten.

Einige Java-Versionen unterstützen nur bestimmte GCs. Welchen GC Entwickler:innen verwenden, hängt also teilweise von ihrer Java-Version ab.

Seit Java 11 ist der Garbage-First (G1) GC der Standard. Das könnte erklären, warum 43 % der Kund:innen ihn verwenden und warum es einen so großen Sprung in der Nutzung von Java 11, 17 und 21 im Vergleich zu Java 7 und 8 gab. Einer der Hauptvorteile von G1 ist außerdem, dass er kleinere Bereiche löscht, anstatt große Bereiche auf einmal zu löschen. Das optimiert den Prozess. Außerdem bleibt die Ausführung nur selten hängen und es können sowohl die neue als auch die alte Generation gleichzeitig bearbeitet werden, was es zu einer hervorragenden Wahl für Entwickler:innen macht.

An zweiter Stelle folgt der Serial GC (37 %), der sich ideal für Anwendungen oder Systeme eignet, die auf einem einzigen Prozessor laufen oder bei denen eine große Anzahl virtueller Java-Maschinen (JVMs) auf demselben Rechner ausgeführt werden. Außerdem ist der CPU- und Speicher-Overhead im Vergleich zu komplexeren GCs geringer, sodass er sich für ressourcenbeschränkte Umgebungen eignet.



Von Java-LTS-Versionen verwendete Garbage Collectors

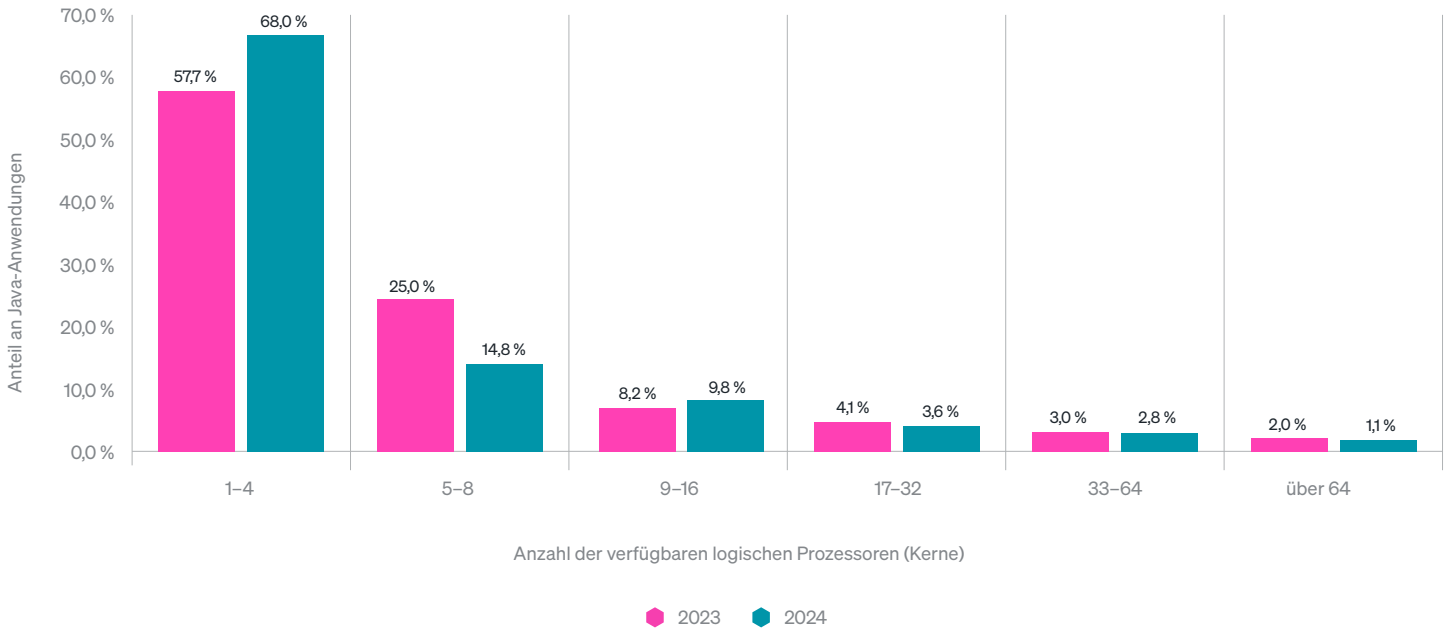
Rechen- und Speichereinstellungen

68 %

nutzen 1–4 Kerne

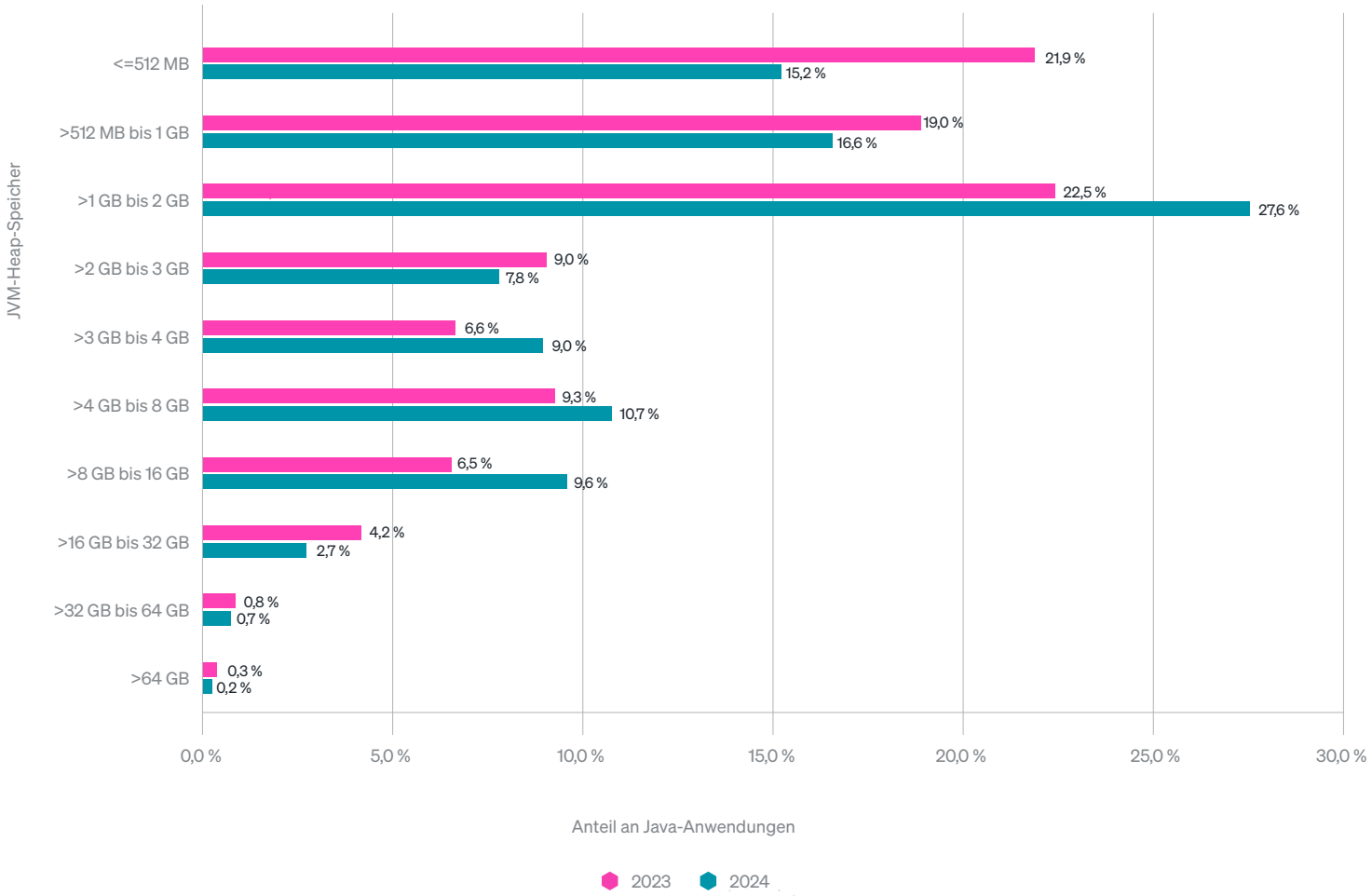
Die Daten von New Relic zeigen einen Anstieg um 18 % im Vergleich zum Vorjahr bei Anwendungen, die mit vier oder weniger Prozessorkernen laufen.

Gerade in Cloud-Umgebungen, in denen Container häufig zum Einsatz kommen, ist es durchaus sinnvoll, die Ressourcennutzung sparsamer zu halten. Bei bestimmten Anwendungen können damit allerdings auch unerwartete Probleme einhergehen. Dies gilt insbesondere für viele der Vorteile der parallelen („concurrent“) Ausführungsmethodik des standardmäßigen G1 GC auf aktuellen JVMs: Bei Zuweisung von weniger als zwei Kernen sind diese nicht mehr realisierbar. Denn sämtliche mit nur einem Prozessorkern ausgeführten Instanzen könnten genauso gut den seriellen Collector nutzen, wofür dann auch die ihm zugehörigen Performance-Kosten anfallen.



Verfügbare logische Prozessoren (Kerne) nach Java-Anwendungen 2023 und 2024

Was die JVM-Speichereinstellungen angeht, nehmen 32 % der Java-Anwendungen 1 GB oder weniger in Anspruch, 68 % brauchen mehr als 1 GB. Im Vergleich zum Vorjahr ist der Anteil der Anwendungen, die mehr als 1 GB Speicherplatz beanspruchen, um 15 % gestiegen.



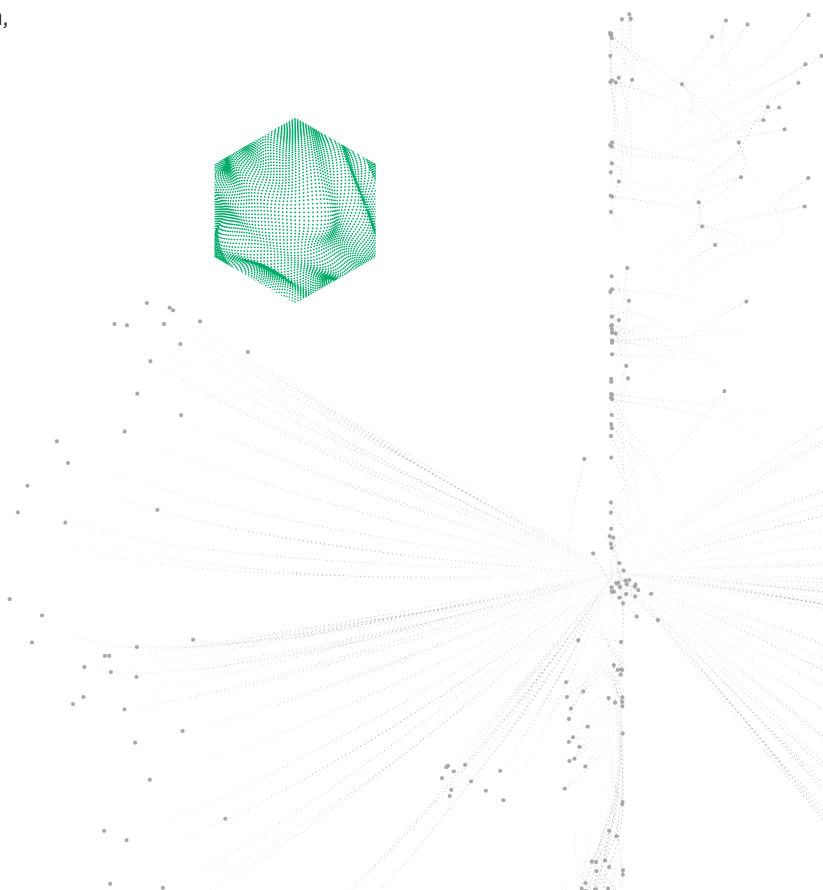
JVM-Heap-Speicher nach Jahr

Gängige Java-Frameworks und -Bibliotheken für Logging, Verschlüsselung und Datenbank

Java-Entwickler:innen verwenden Frameworks und Bibliotheken, um die Anwendungsentwicklung zu optimieren, indem sie vorgefertigten Code für gängige Aufgaben wie Webentwicklung und Datenbankkonnektivität verwenden. Mit Bibliotheken wird der Funktionsumfang einer Anwendung erweitert. Im Folgenden untersuchen wir die beliebtesten Frameworks und Bibliotheken für Logging, Verschlüsselung und Datenbankanbindung.

Log4j ist das beliebteste Logging-Framework für Java-Anwendungen

Bei allen Softwareanwendungen und Systemen kann es in der Test- oder Produktionsumgebung zu Fehlern und Problemen kommen. Um solche Probleme zu finden und zu beheben, verwenden Entwickler:innen Logging-Tools. Logging ist jedoch nur dann sinnvoll, wenn es die erforderlichen Informationen aus den Lognachrichten liefert, ohne die Performance der Anwendung zu beeinträchtigen. Software-Entwickler:innen verwenden verschiedene Logging-Frameworks, um diese Probleme zu lösen. So nutzen 91 % aller Java-Anwendungen, die Daten an New Relic übermitteln, Logging-Frameworks.

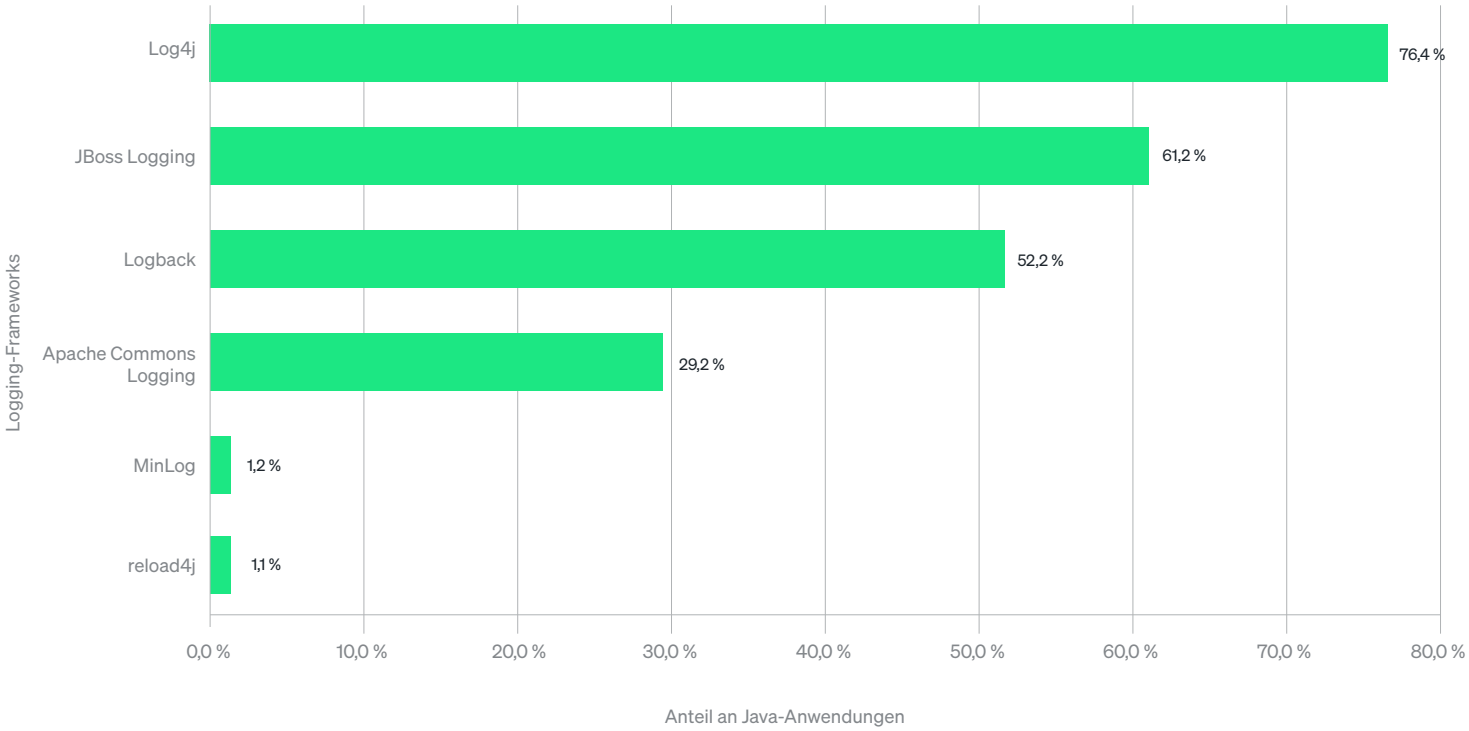


Das am häufigsten verwendete Logging-Framework war Log4j, das 76 % der Java-Anwendungen nutzen, gefolgt von JBoss Logging (61 %) und Logback (52 %).

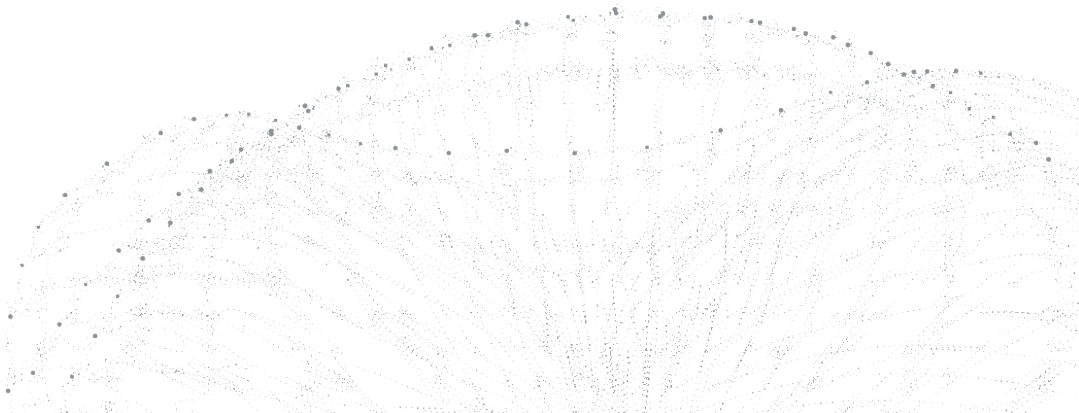
Die meisten Java-Entwickler:innen (83 %) verlassen sich auf SLF4j, ein Framework, das als Abstraktion für andere Java-Logging-Frameworks dient. SLF4j ermöglicht es Softwareentwickler:innen, das Logging-Framework ihrer Wahl zu verwenden, und lässt zu, dass Anwendungen zu jedem beliebigen Java-Logging-Framework wechseln können, ohne die Implementierungen zu beeinflussen oder Änderungen vorzunehmen. Damit macht SLF4j Anwendungen unabhängig von Logging-Frameworks und bietet mehr Flexibilität und Portabilität für das Logging in jedem Systembereich. Außerdem bedeutet dies, dass Java-Anwendungen mehr als ein Logging-Framework verwenden können.

76 %

nutzen das Logging-Framework Log4j



Die beliebtesten Logging-Frameworks für Java-Anwendungen



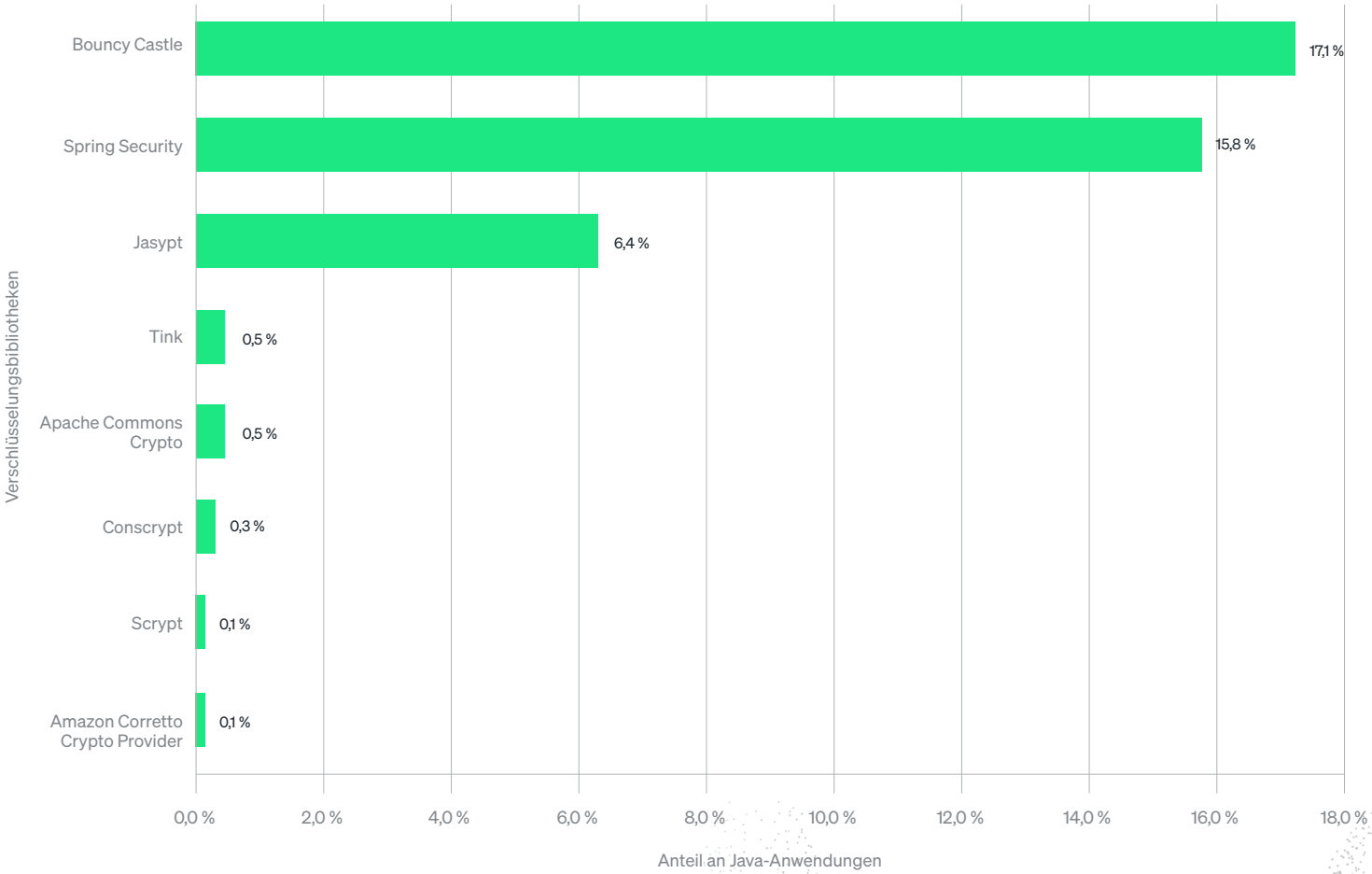
Bouncy Castle ist die am häufigsten verwendete Verschlüsselungsbibliothek für Java-Anwendungen

17 %

nutzen die Verschlüsselungs-bibliothek Bouncy Castle

Mehr als ein Drittel (41 %) der Java-Anwendungen, die Daten an New Relic übermitteln, verwenden Verschlüsselungsbibliotheken: 17 % nutzen Bouncy Castle, 16 % verwenden Spring Security und 6 % verwenden Jasypt. Bouncy Castle ist seit vielen Jahren dank seiner Cipher Suites und Dienstprogramme eine beliebte Verschlüsselungsbibliothek.

Obwohl bislang nur 0,09 % der Entwickler:innen Amazon Corretto Crypto Provider (ACCP) verwenden, erwarten wir, dass diese Bibliothek in naher Zukunft für mehr Anwendungen genutzt wird, da Unternehmen und Entwickler:innen versuchen, Anbieter zu konsolidieren – und weil sie in der Regel eine bessere Leistung bietet.



Die populärsten Verschlüsselungsbibliotheken für Java-Anwendungen

Oracle ist das am häufigsten genutzte Datenbanksystem für Java-Anwendungen

17 %

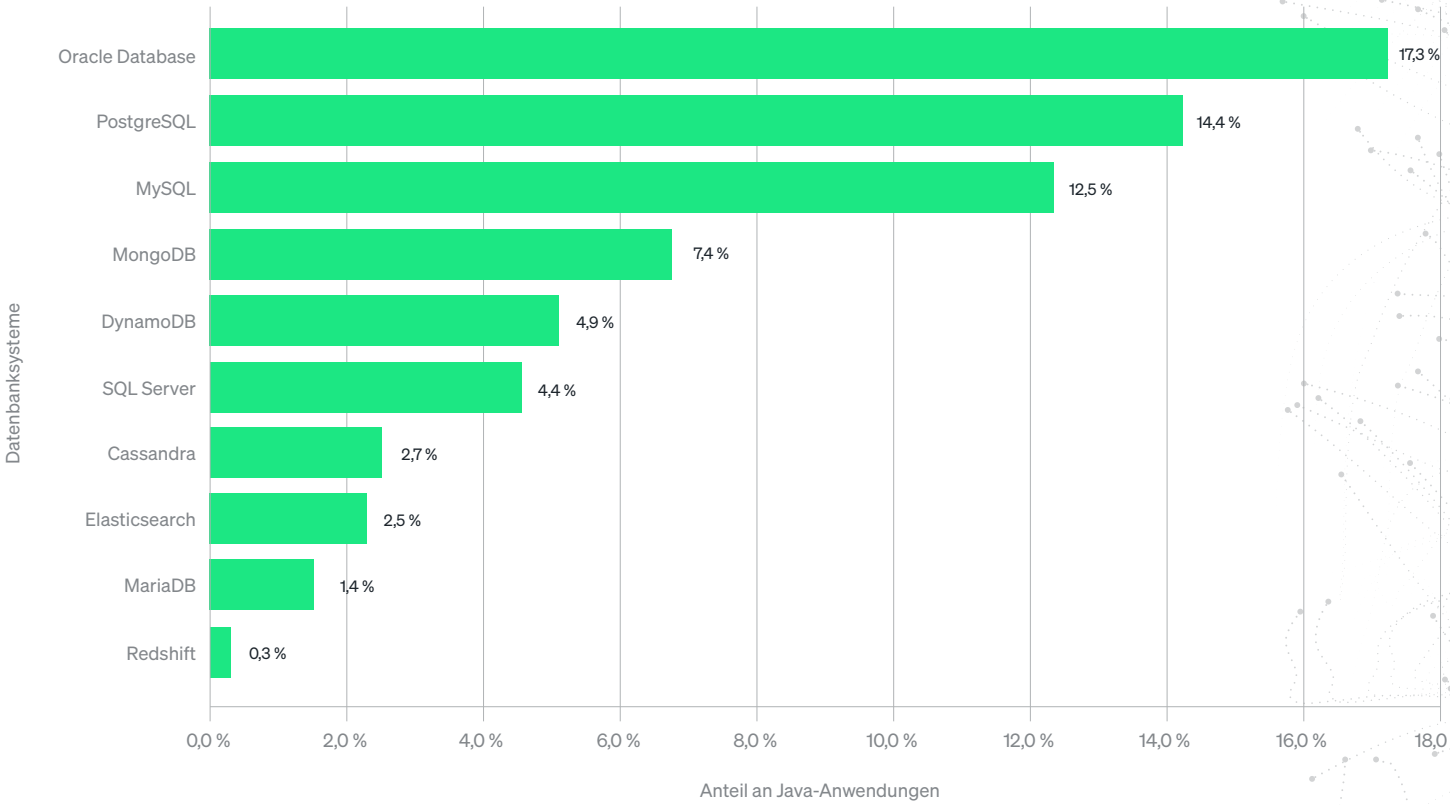
verwenden Oracle Database

Bei den Datenbanken ist Oracle Database am weitesten verbreitet. 17 % der Java-Anwendungen, die Daten an New Relic melden, nutzen diese Datenbank. Oracle Database ist bekannt für seine Skalierbarkeit und die Fähigkeit, große Datenmengen schnell und effizient zu verwalten. Daher ist es in der Regel das bevorzugte Datenbanksystem für Unternehmen. Darüber hinaus bietet es Kundensupport und eine Reihe zuverlässiger Tools.

An zweiter Stelle folgt PostgreSQL, das von 14 % der Java-Anwendungen, die Daten an New Relic übermitteln, verwendet wird. Während Oracle Database direkt von Oracle verwaltet wird und über eine Lizenz verfügbar ist, ist PostgreSQL eine Open-Source-Datenbank, die kostenlos genutzt werden kann und bevorzugt für die Verwaltung von Lese- und Schreibvorgängen sowie komplexen Abfragen eingesetzt wird.

An dritter Stelle steht MySQL, das 13 % der Java-Anwendungen verwenden. MySQL ist ebenfalls eine Open-Source-Datenbank. Es bietet weniger Funktionen als Oracle Database und PostgreSQL, was es stabiler und schneller bei der Verarbeitung macht, insbesondere bei schreibgeschützten Abfragen.

PostgreSQL hat im letzten Jahr an Popularität gewonnen, da es die SQL-Spezifikationen besser erfüllt und mehr Funktionen als MySQL unterstützt. MySQL ist zwar nach wie vor eines der führenden Datenbanksysteme, doch es ist nicht zu verkennen, dass PostgreSQL aufholt.



Die populärsten Datenbanksysteme für Java-Anwendungen

Häufige Fragen und Anliegen zu Java von Entwickler:innen

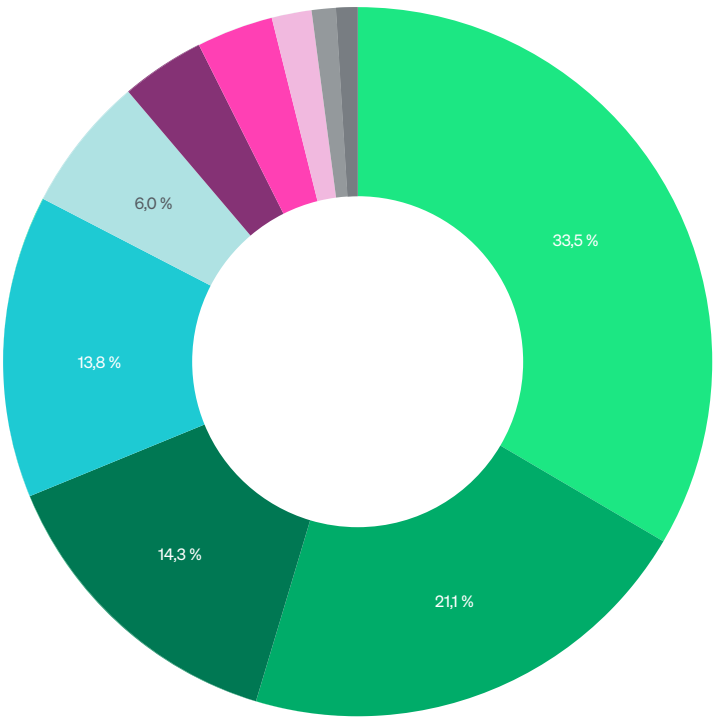
34 %

aller Java-bezogenen Fragen waren Bitten um eine Anleitung

Wir haben außerdem untersucht, welche Java-bezogenen Fragen Entwickler:innen dem GenAI-Assistenten für Observability New Relic AI stellen. Die Daten zeigen, dass es seit Januar 2024 bei 34 % der insgesamt 483 Java-bezogenen Fragen um die Suche nach Anweisungen ging, 21 % waren Fragen nach bestimmten Metriken, 14 % Fragen zur Konfiguration und bei 14 % der Fragen drehte es sich um Fehlerbehebung.

Einige beispielhafte Fragen und Prompts:

- Wie kann ich API-Aufrufe in meinem Java Spring Boot loggen?
- Muss ich den Agent neu konfigurieren, wenn ich das Java-Framework von Tomcat auf JBoss umstelle?
- Erstell mir ein Dashboard für die Container-Speicherauslastung im Vergleich zur JVM-Heap-Auslastung.
- Diese überwachte Anwendung verwendet das Vaadin-Framework in Java. Es hat eine hohe Speichernutzung. Woran könnte das liegen?
- Was bedeutet in der JVM die Heap-Nutzung von G1 Eden Space?



Die häufigsten Java-bezogenen Fragen von Entwickler:innen an New Relic AI

Methodik

Dieser Bericht basiert auf Daten aus Hunderttausenden von Anwendungen, die Performance-Informationen an New Relic liefern. Er bietet daher kein umfassendes Bild der Java-Nutzung. Alle Daten wurden im Jahr 2024 erfasst.

Die Daten wurden anonymisiert und um Details bereinigt, die nicht zur allgemeinen Beurteilung des Java-Ökosystems notwendig sind. Sämtliche Informationen, die für Cyber-Angriffe oder andere böswillige Aktivitäten von Nutzen sein könnten, haben wir aus diesem Report absichtlich komplett ausgeklammert.

Überwachen Sie Ihre Java-Daten jetzt mit New Relic.

Java Quickstart installieren

Über New Relic

Als führender Anbieter von Observability-Technologien unterstützt New Relic die globale Engineering-Community mit einer datenfundierten Methodik für den gesamten Software-Lifecycle – von der Konzeptphase bis zur operativen Umsetzung. In New Relic erhalten Entwickler:innen die einzige Plattform zur Erfassung sämtlicher Telemetriedaten: Metriken, Events, Logs und Traces. Im Verbund mit umfassenden Analysetools für den gesamten Stack führt New Relic in kürzester Zeit von einer grundlegenden Situationsanalyse zur genauen Problemursache.

Mit dem branchenweit ersten klar planbaren verbrauchsbasierten Kostenmodell setzt New Relic auf absolute Transparenz in jeder Hinsicht und liefert der Engineering-Community so diverse Vorteile, von optimierter Cycle-Planung bis hin zu besseren Ergebnissen bei der Rate änderungsbedingter Ausfälle, der Release-Frequenzen und den mittleren Lösungszeiten (MTTR). Diese Möglichkeiten nutzen branchenführende Weltmarken wie auch Hypergrowth-Start-ups für bessere System-Uptime und -Stabilität, mehr Effizienz und optimale UX für ihre Endkund:innen – und für mehr Innovationschancen und Wachstum für sich.

new relic.

© Copyright 2024, New Relic, Inc. Alle Rechte vorbehalten. Alle hier erwähnten Marken, Handelsnamen, Dienstleistungsmarken und Logos sind Eigentum der jeweiligen Unternehmen. 04.2024