

Java Third Party Libraries

Impact on Java Applications

Aurelio Garcia-Ribeyro

Java Platform Group - Oracle





JUnit 5



JAXB

jackson



ASM



OpenJDK Wiki

Home | View | Edit

Dashboard > Quality > Main > Quality Outreach

- About
- Adoption
- Amber
- Build
- Client Libraries
- Code Tools
- Coin
- Compatibility & Specification Review
- Compiler
- CRaC
- Device I/O
- Duke
- Graal
- HotSpot
- IDE Tooling & Support
- JDK 8
- JDK 8u
- JDK Updates
- Kulla
- Lanai
- Lilliput
- Loom
- Memory Model Update
- Mission Control
- Multi-Language VM
- Nashorn

Quality Outreach

Created by Rory O'Donnell, last modified by David Delabassee on Oct 24, 2023

- [Quality Outreach](#)
- [Introduction](#)
- [How to join the Quality Outreach Program](#)
- [Status indicators](#)

Quality Outreach

Introduction

The Quality Group is promoting the testing of Free Open Source Software (FOSS) projects with OpenJDK builds, whether their own, or from someone else. We want to acknowledge projects who are actively testing, providing feedback and list any issues they have found during their testing. This is a great way to improve the quality of the various releases, adding new projects to the list is highly desirable.

How to join the Quality Outreach Program

If you would like to have your Project added to the list below please send an email to [quality-discuss](#) mailing lists including:

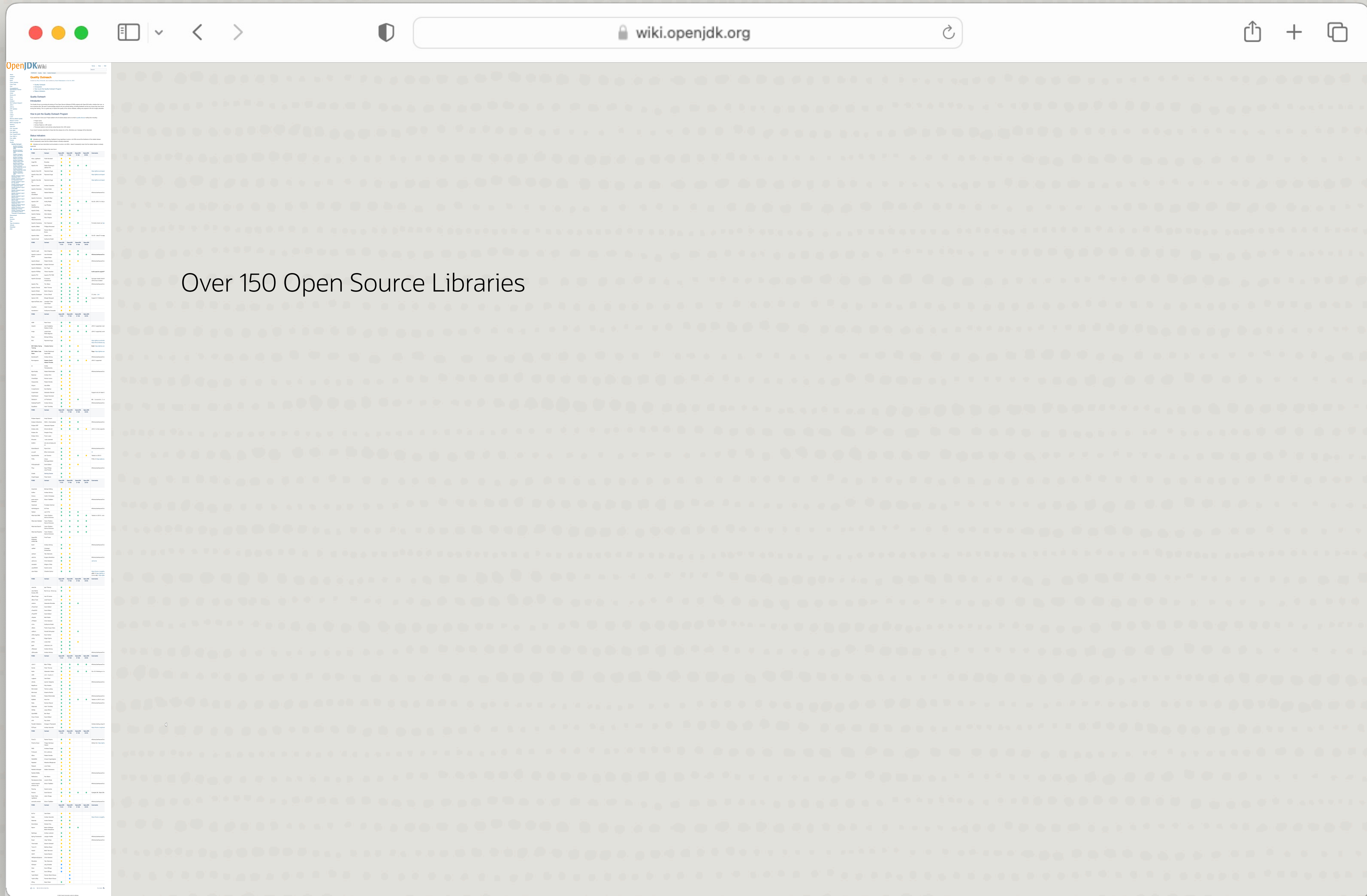
- Project name
- Project Contact
- Actively Tested on: JDK version



★ - Indicates will start testing in the near future

FOSS	Contact	OpenJDK 11.0.6	OpenJDK 17 GA	OpenJDK 21 GA	OpenJDK 22 EA	Comments
Akka, Lightbend	Patrik Nordwall	★	★			
Angel-ML	Brucetao	★	★			
Apache Ant	Stefan Bodewig & Jaikiran Pai	★	★	★	★	
Apache Aries CDI	Raymond Auge	★	★			https://github.com/apacl
Apache Aries JAX-RS	Raymond Auge	★	★			https://github.com/apacl
Apache Aries Spi Fly	Raymond Auge	★	★			https://github.com/apacl
Apache Camel	Andrea Cosentino	★	★			
Apache Chemistry	Florian Muller	★	★			
Apache CloudStack	Gabriel Bräscher	★	★			#WorksLikeHeavenOnJ
Apache Commons	Benedikt Ritter	★	★			
Apache CXF	Andriy Redko	★	★	★	★	Oct.23: JDK 21 is fully s
Apache DataSketches	Lee Rhodes	★	★			
Apache Derby	Rick Hillegas	★	★	★		





Over 150 Open Source Libraries



But so far this has been only about adopting new JDK versions

We also need to look at the support timelines....

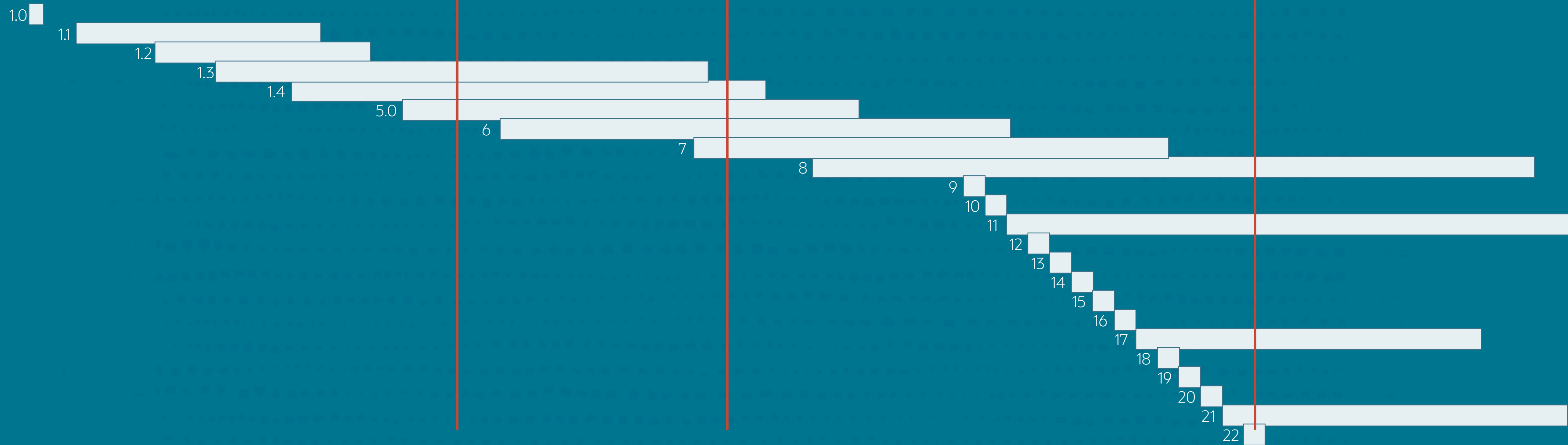
"Multiple release trains" model is well established

Support Timelines

Java Platform

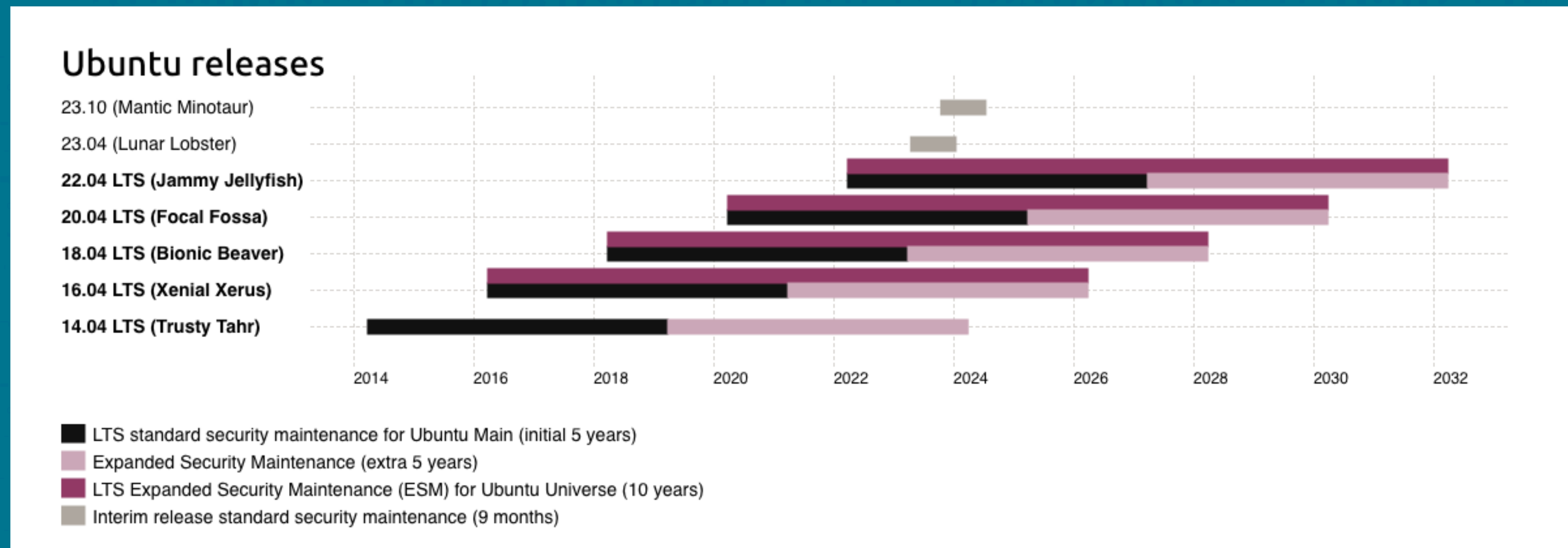
1995

2031



Support Timelines

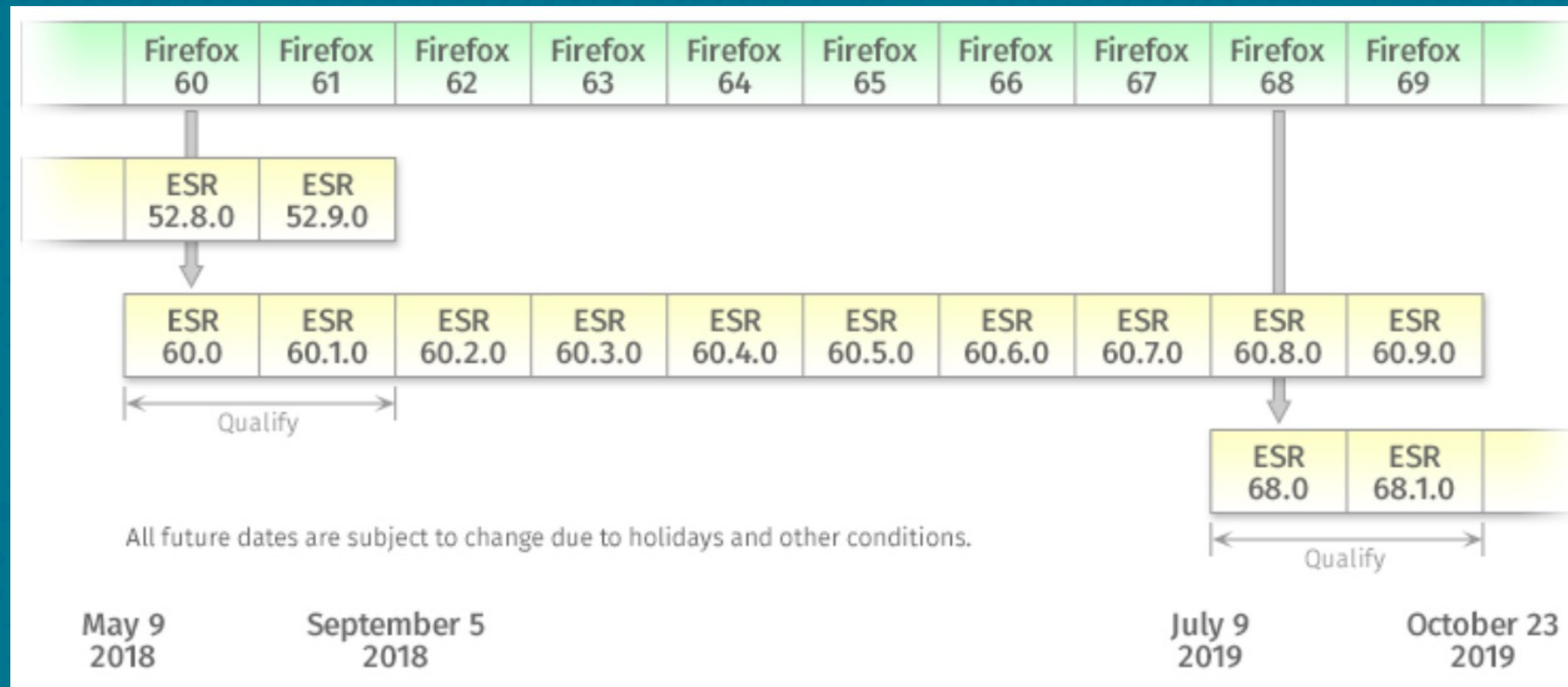
Ubuntu



<https://ubuntu.com/about/release-cycle> as of 2024/04/18

Support Timelines

Firefox: ESR = Extended Support Release



<https://support.mozilla.org/en-US/kb/firefox-esr-release-cycle> as of 2024/04/18

"Multiple release trains" model is well established... for the lower part of the stack: OS, Runtime, Infrastructure...

What about libraries...

The screenshot shows the Maven Repository website (mvnrepository.com) displaying a list of popular projects. The page includes a search bar, navigation links, and a sidebar with categories. The main content area lists the top projects with their names, logos, and usage statistics.

Indexed Artifacts (35.3M)

Indexed Repositories (1991)

Popular Categories

- Testing Frameworks & Tools
- Android Packages
- Logging Frameworks
- Java Specifications
- JSON Libraries
- JVM Languages
- Language Runtime
- Core Utilities
- Mocking
- Web Assets
- Annotation Libraries
- HTTP Clients
- Logging Bridges
- Dependency Injection
- XML Processing
- Web Frameworks
- I/O Utilities

Top Projects

- JUnit** (126,102 usages, EPL)
JUnit is a unit testing framework to write and run repeatable automated tests on Java. It provides a robust environment to write, organize, and execute automated tests, ensuring code reliability and repeatability. Its user-friendly annotations and assert methods facilitate the development and running of test cases, making it a foundational tool for Java developers focusing on quality assurance and test-driven development.
Last Release on Feb 13, 2021
Relocated → [org.junit.jupiter](#) » [junit-jupiter-api](#)
- SLF4J API Module** (67,397 usages, MIT)
API for SLF4J (The Simple Logging Facade for Java) which serves as a simple facade or abstraction for various logging frameworks, allowing the end user to plug in the desired logging framework at deployment time.
Last Release on Jan 2, 2024
- Kotlin Stdlib** (39,872 usages, Apache)
Kotlin Standard Library
Last Release on Apr 9, 2024
- Kotlin Stdlib Common** (38,235 usages, Apache)
Last Release on Apr 9, 2024

Popular Tags

aar android apache api application arm assets build build-system bundle client clojure cloud commons config cran data database eclipse example extension framework github gradle groovy ios javascript kotlin library logging maven mobile module npm osgi plugin resources rlang sdk server service spring sql starter testing tools ui war web webapp



JUnit

MVN REPOSITORY Search for groups, artifacts, categories Search Categories | Popular | Contact Us

Home » org.junit.jupiter » junit-jupiter-api

JUnit Jupiter API

JUnit Jupiter is the API for writing tests using JUnit 5.

License EPL 2.0

Categories Testing Frameworks & Tools

Tags quality junit testing api

Ranking #22 in MvnRepository (See Top Artifacts)
#3 in Testing Frameworks & Tools

Used By 15,172 artifacts

Central (66) Redhat GA (3) Redhat EA (1) ICM (2)

	Version	Vulnerabilities	Repository	Usages	Date
5.10.x	5.10.2		Central	1,330	Feb 04, 2024
	5.10.1		Central	1,662	Nov 05, 2023
	5.10.0		Central	1,706	Jul 23, 2023
	5.10.0-RC2		Central	10	Jul 19, 2023
	5.10.0-RC1		Central	115	Jul 06, 2023
	5.10.0-M1		Central	101	May 13, 2023
	5.9.x	5.9.3		Central	1,828
5.9.2			Central	1,965	Jan 10, 2023
5.9.1			Central	1,868	Sep 20, 2022
5.9.0			Central	1,190	Jul 26, 2022
5.9.0-RC1			Central	79	Jul 04, 2022
5.9.0-M1			Central	108	May 15, 2022
	5.8.2		Central	2,716	Nov 28, 2021



JUnit

Android Packages #3 in Testing Frameworks & Tools mvnrepository.com

Central (66) Redhat GA (3) Redhat EA (1) ICM (2)

Version	Vulnerabilities	Repository	Usages	Date
5.10.x	5.10.2	Central	1,330	Feb 04, 2024
	5.10.1	Central	1,662	Nov 05, 2023
	5.10.0	Central	1,706	Jul 23, 2023
	5.10.0-RC2	Central	10	Jul 19, 2023
	5.10.0-RC1	Central	115	Jul 06, 2023
	5.10.0-M1	Central	101	May 13, 2023
5.9.x	5.9.3	Central	1,828	Apr 26, 2023
	5.9.2	Central	1,965	Jan 10, 2023
	5.9.1	Central	1,868	Sep 20, 2022
	5.9.0	Central	1,190	Jul 26, 2022
	5.9.0-RC1	Central	79	Jul 04, 2022
	5.9.0-M1	Central	108	May 15, 2022
5.8.x	5.8.2	Central	2,716	Nov 28, 2021
	5.8.1	Central	1,203	Sep 22, 2021
	5.8.0	Central	302	Sep 12, 2021
	5.8.0-RC1	Central	35	Aug 17, 2021
	5.8.0-M1	Central	288	Feb 11, 2021
5.7.x	5.7.2	Central	1,350	May 15, 2021
	5.7.1	Central	1,368	Feb 04, 2021
	5.7.0	Central	2,214	Sep 13, 2020
	5.7.0-RC1	Central	78	Aug 16, 2020
	5.7.0-M1	Central	118	Apr 19, 2020
5.6.x	5.6.3	Central	199	Oct 26, 2020
	5.6.2	Central	1,704	Apr 10, 2020
	5.6.1	Central	359	Mar 22, 2020

Indexed Repositories (1991)

- Central
- Atlassian
- Hortonworks
- JCenter
- Sonatype
- JBossEA
- KtorEAP
- Atlassian Public



SLF4J

mvnrepository.com

Central (102) | Redhat GA (42) | Redhat EA (7) | JBoss 3rd-party (2) | EmergyaPub (4) | Geomajas (1) | ICM (8)

	Version	Vulnerabilities	Repository	Usages	Date
2.1.x	2.1.0-alpha1		Central	23	Jan 02, 2024
	2.1.0-alpha0		Central	10	Dec 28, 2023
2.0.x	2.0.13		Central	218	Apr 12, 2024
	2.0.12		Central	2,299	Feb 06, 2024
	2.0.11		Central	1,332	Jan 08, 2024
	2.0.10		Central	415	Dec 29, 2023
	2.0.9		Central	4,044	Sep 03, 2023
	2.0.8		Central	16	Sep 03, 2023
	2.0.7		Central	4,345	Mar 17, 2023
	2.0.6		Central	3,342	Dec 13, 2022
	2.0.5		Central	950	Nov 25, 2022
	2.0.4		Central	476	Nov 17, 2022
	2.0.3		Central	1,254	Sep 28, 2022
	2.0.2		Central	346	Sep 20, 2022
	2.0.1		Central	383	Sep 14, 2022
	2.0.0		Central	759	Aug 20, 2022
	2.0.0-beta1		Central	19	Aug 06, 2022
	2.0.0-beta0		Central	10	Aug 05, 2022
	2.0.0-alpha7		Central	164	Mar 17, 2022
	2.0.0-alpha6		Central	172	Jan 13, 2022
	2.0.0-alpha5		Central	262	Aug 30, 2021
	2.0.0-alpha4		Central	30	Aug 12, 2021
2.0.0-alpha3		Central	10	Aug 10, 2021	
2.0.0-alpha2		Central	487	Jul 02, 2021	
2.0.0-alpha1		Central	1,004	Oct 01, 2019	

Indexed Repositories (1991)

- Central
- Atlassian
- Hortonworks
- JCenter
- Sonatype
- JBossEA
- KtorEAP
- Atlassian Public
- WSO2 Releases



Google Guava

The screenshot shows the mvnrepository.com website. The browser address bar displays 'mvnrepository.com'. The page features a navigation menu on the left with categories such as 'JSON Libraries', 'JVM Languages', 'Language Runtime', 'Core Utilities', 'Mocking', 'Web Assets', 'Annotation Libraries', 'HTTP Clients', 'Logging Bridges', 'Dependency Injection', 'XML Processing', 'Web Frameworks', 'I/O Utilities', 'Defect Detection Metadata', 'Code Generators', 'Configuration Libraries', 'OSGi Utilities', 'Concurrency Libraries', 'Reflection Libraries', 'Android Platform', 'JDBC Drivers', 'Date and Time Utilities', 'Assertion Libraries', 'Validation Libraries', 'Collections', 'Bytecode Libraries', 'Data Formats', and 'Aspect Oriented'. The main content area displays a table of versions for Google Guava, with tabs for different repositories: Central (125), Atlassian 3rd-P Old (2), Redhat GA (26), Redhat EA (11), JBoss 3rd-party (3), Mulesoft (1), ICM (17), and CUBA BT (1). The table has columns for Version, Vulnerabilities, Repository, Usages, and Date. The 'Vulnerabilities' column contains red text indicating the number of vulnerabilities for each version. The 'Repository' column shows 'Central' for all entries. The 'Usages' column shows the number of usages, and the 'Date' column shows the date of the version. The right sidebar shows 'Indexed Repositories (1991)' with a list of repositories including Central, Atlassian, Hortonworks, JCenter, Sonatype, JBossEA, KtorEAP, Atlassian Public, and WSQ2 Releases.

Version	Vulnerabilities	Repository	Usages	Date
33.1.0-jre		Central	1,427	Mar 13, 2024
33.1.0-android		Central	12	Mar 13, 2024
33.0.0-jre		Central	1,557	Dec 19, 2023
33.0.0-android		Central	21	Dec 19, 2023
32.1.3-jre		Central	2,630	Oct 11, 2023
32.1.3-android		Central	47	Oct 11, 2023
32.1.2-jre		Central	3,133	Aug 01, 2023
32.1.2-android		Central	30	Aug 01, 2023
32.1.1-jre		Central	1,852	Jun 30, 2023
32.1.1-android		Central	13	Jun 30, 2023
32.1.0-jre		Central	49	Jun 29, 2023
32.1.0-android		Central	1	Jun 29, 2023
32.0.1-jre		Central	2,337	Jun 09, 2023
32.0.1-android		Central	44	Jun 09, 2023
32.0.0-jre	2 vulnerabilities	Central	850	May 27, 2023
32.0.0-android	2 vulnerabilities	Central	16	May 27, 2023
31.1-jre	2 vulnerabilities	Central	6,150	Mar 01, 2022
31.1-android	2 vulnerabilities	Central	144	Mar 01, 2022
31.0.1-jre	2 vulnerabilities	Central	3,417	Sep 27, 2021
31.0.1-android	2 vulnerabilities	Central	85	Sep 27, 2021
31.0-jre	2 vulnerabilities	Central	139	Sep 24, 2021
31.0-android	2 vulnerabilities	Central	7	Sep 24, 2021
30.1.1-jre	2 vulnerabilities	Central	4,017	Mar 19, 2021
30.1.1-android	2 vulnerabilities	Central	529	Mar 19, 2021



Apache Commons

mvnrepository.com

Tags: [apache](#) [commons](#)

Ranking: #10 in MvnRepository (See Top Artifacts)
#2 in Core Utilities

Used By: 28,345 artifacts

Central (20) Redhat GA (16) Redhat EA (2) RWJF (1) ICM (5)

Version	Vulnerabilities	Repository	Usages	Date
3.14.x 3.14.0		Central	2,070	Nov 22, 2023
3.13.x 3.13.0		Central	2,010	Jul 28, 2023
3.12.x 3.12.0		Central	8,547	Mar 02, 2021
3.11.x 3.11		Central	3,728	Jul 16, 2020
3.10.x 3.10		Central	2,340	Mar 27, 2020
3.9.x 3.9		Central	4,565	Apr 15, 2019
3.8.x	3.8.1	Central	4,022	Sep 19, 2018
	3.8	Central	1,282	Aug 16, 2018
3.7.x 3.7		Central	3,499	Nov 04, 2017
3.6.x 3.6		Central	2,460	Jun 09, 2017
3.5.x 3.5		Central	3,025	Oct 13, 2016
3.4.x	3.4	Central	5,207	Apr 03, 2015
	3.3.2	Central	2,904	Apr 06, 2014
	3.3.1	Central	243	Mar 15, 2014
3.3.x	3.3	Central	153	Feb 28, 2014
	3.2.1	Central	372	Jan 05, 2014
3.2.x	3.2	Central	147	Dec 28, 2013
	3.1	Central	2,408	Nov 15, 2011
3.1.x	3.0.1	Central	218	Aug 10, 2011
	3.0	Central	705	Jul 19, 2011

Indexed Repositories (1991)

- Central
- Atlassian
- Hortonworks
- JCenter

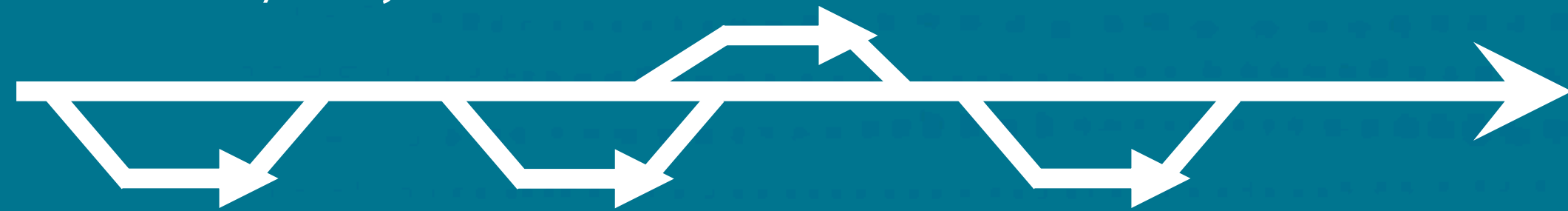


Open source libraries tend to offer a single "one-size fits all" version to cover everything

Obvious Cost vs Hidden Costs

Single version

Works well when everyone can use the latest/only version



<https://designer.microsoft.com/image-creator>

One version to serve them all

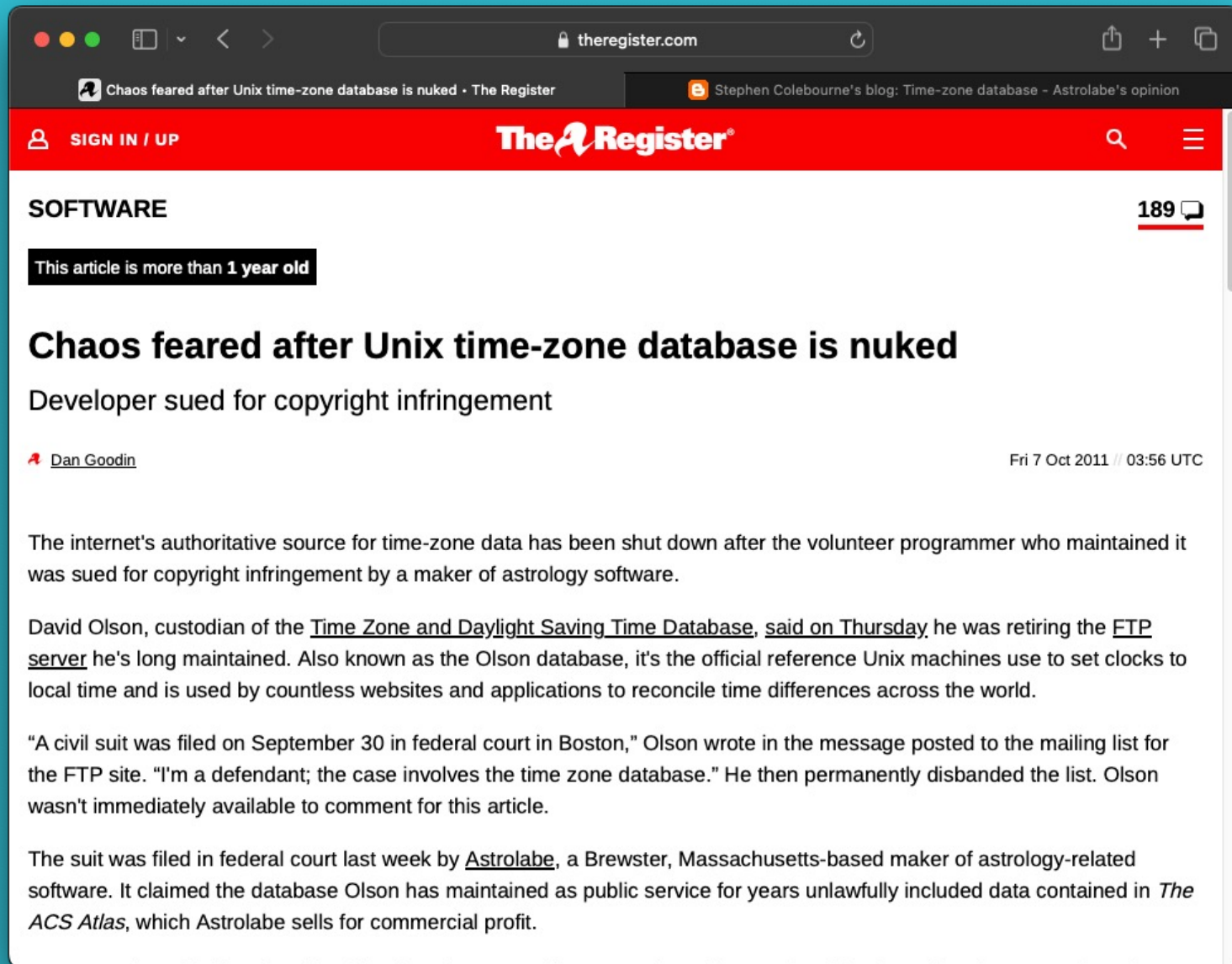
Can only use features/language from the oldest release

Multiple Parallel Versions

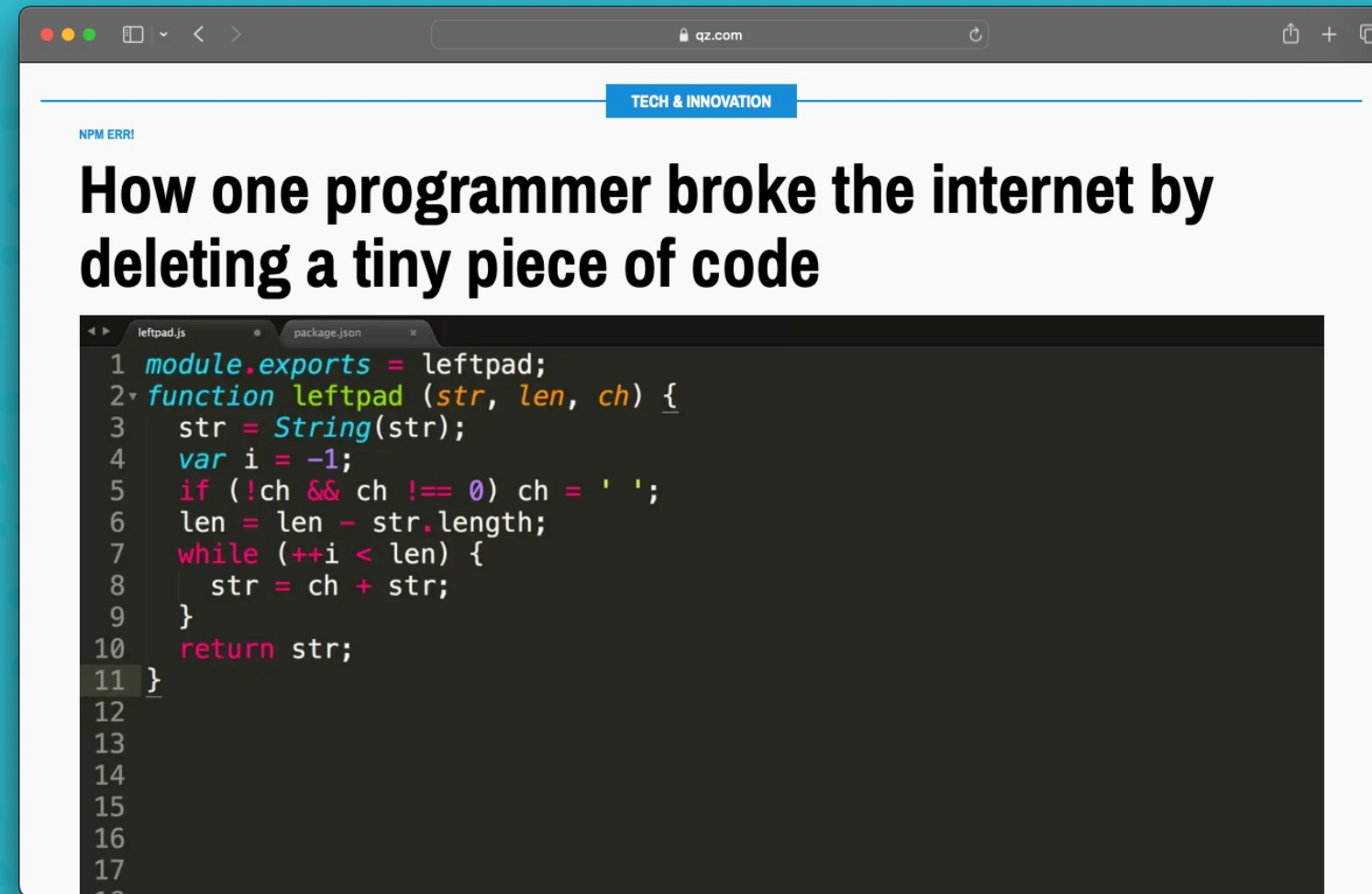


More branches = more resources

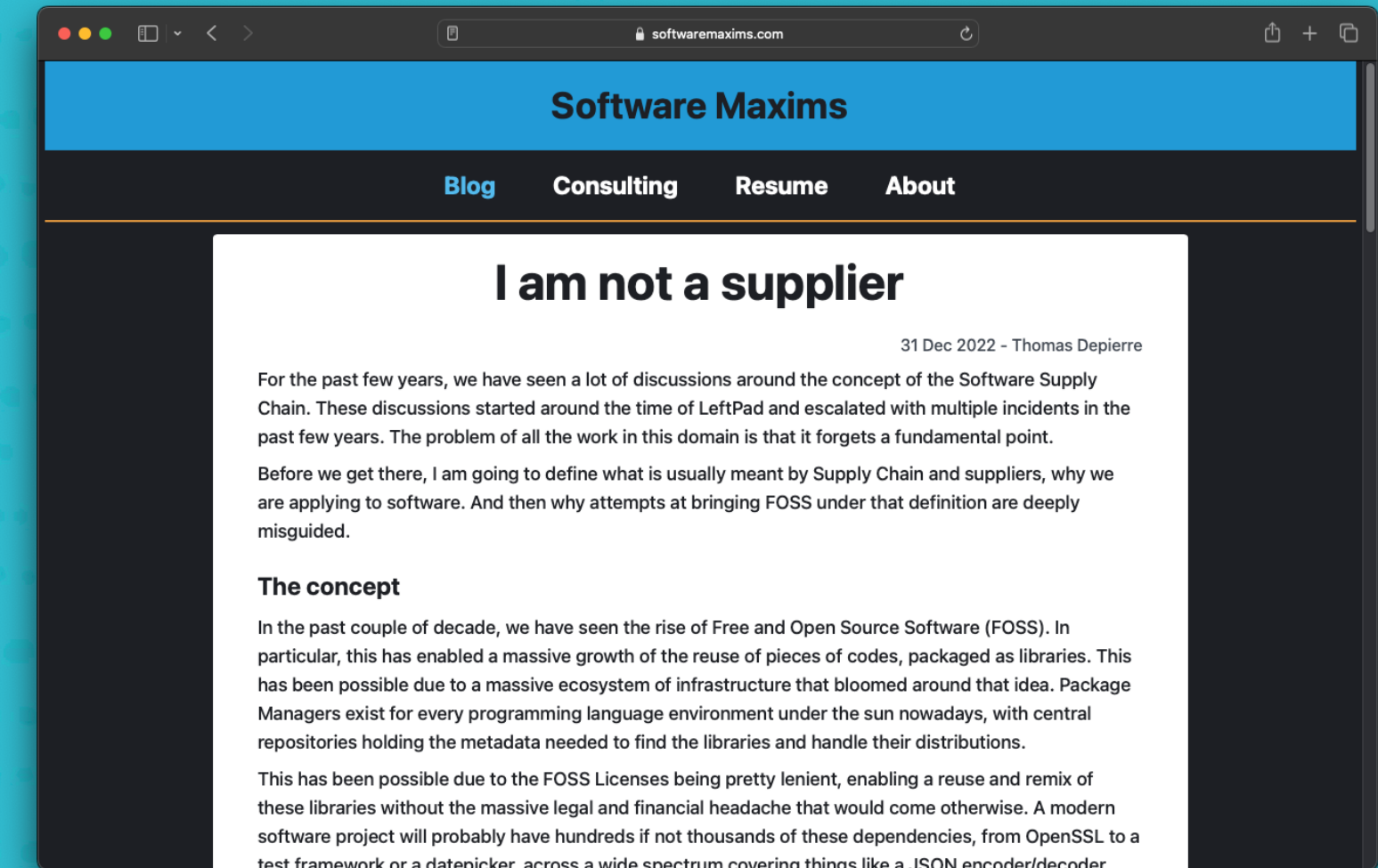
As differences between tip and branch(tail) increase it gets pricier to maintain the branch



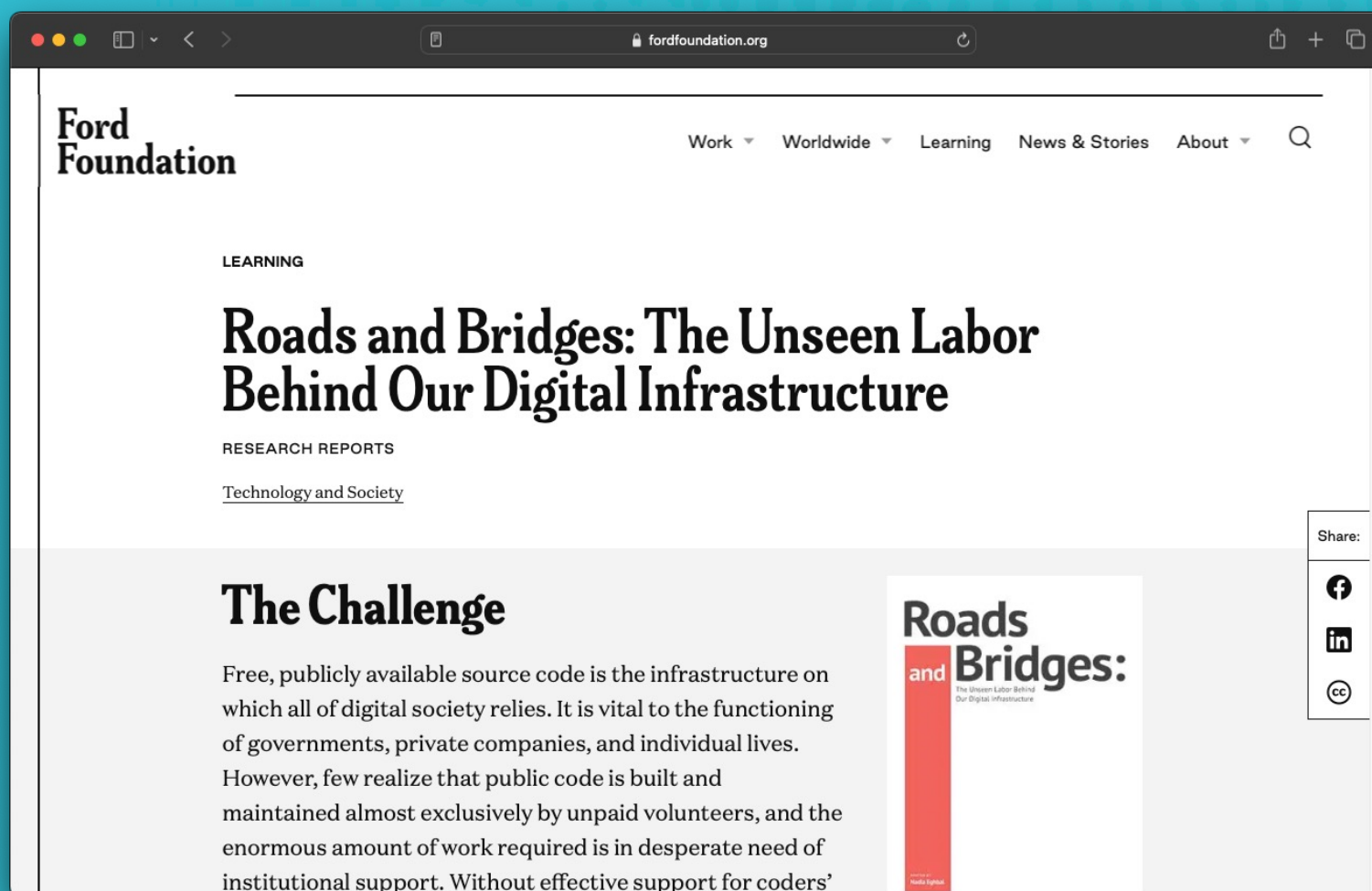
2011: https://www.theregister.com/2011/10/07/unix_time_zone_database_destroyed/



2016 Re Left Pad: <https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code>



2022: <https://www.softwaremaxims.com/blog/not-a-supplier>



2016: <https://www.fordfoundation.org/work/learning/research-reports/roads-and-bridges-the-unseen-labor-behind-our-digital-infrastructure/>



2020: <https://xkcd.com/2347/>



2024 (re Xz backdoor)
<https://mastodon.social/@glyph/112180922900094371>



Can we all just move to only latest?

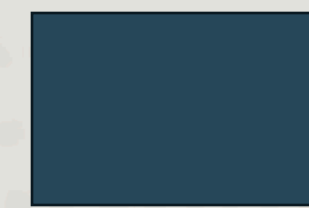
If we could create the right migration tool...

Can it be done?

- Third party components and tools
- Deprecated/removed/encapsulated features without a clear path: Windows-32, Web Start (and even applets), use of internal APIs, security manager, CORBA, configuration changes, and others
- Non-trivial changes: Reactive to virtual threads, JNI to FFM API

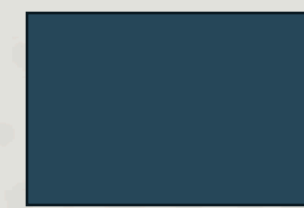
How many users would hold out?

Typical JDK corporate adoption (so far)



JDK X

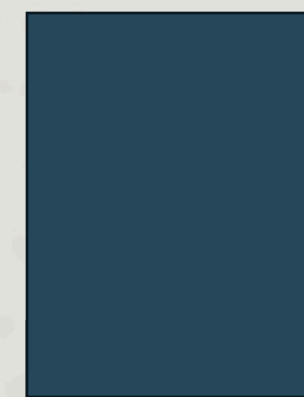
Typical JDK corporate adoption (so far)



JDK X

JDK X+1

Typical JDK corporate adoption (so far)



JDK X

JDK X+1

Typical JDK corporate adoption (so far)



Typical JDK corporate adoption (so far)



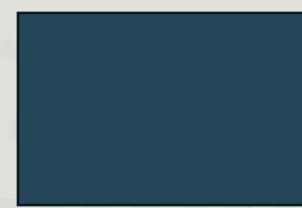
JDK X



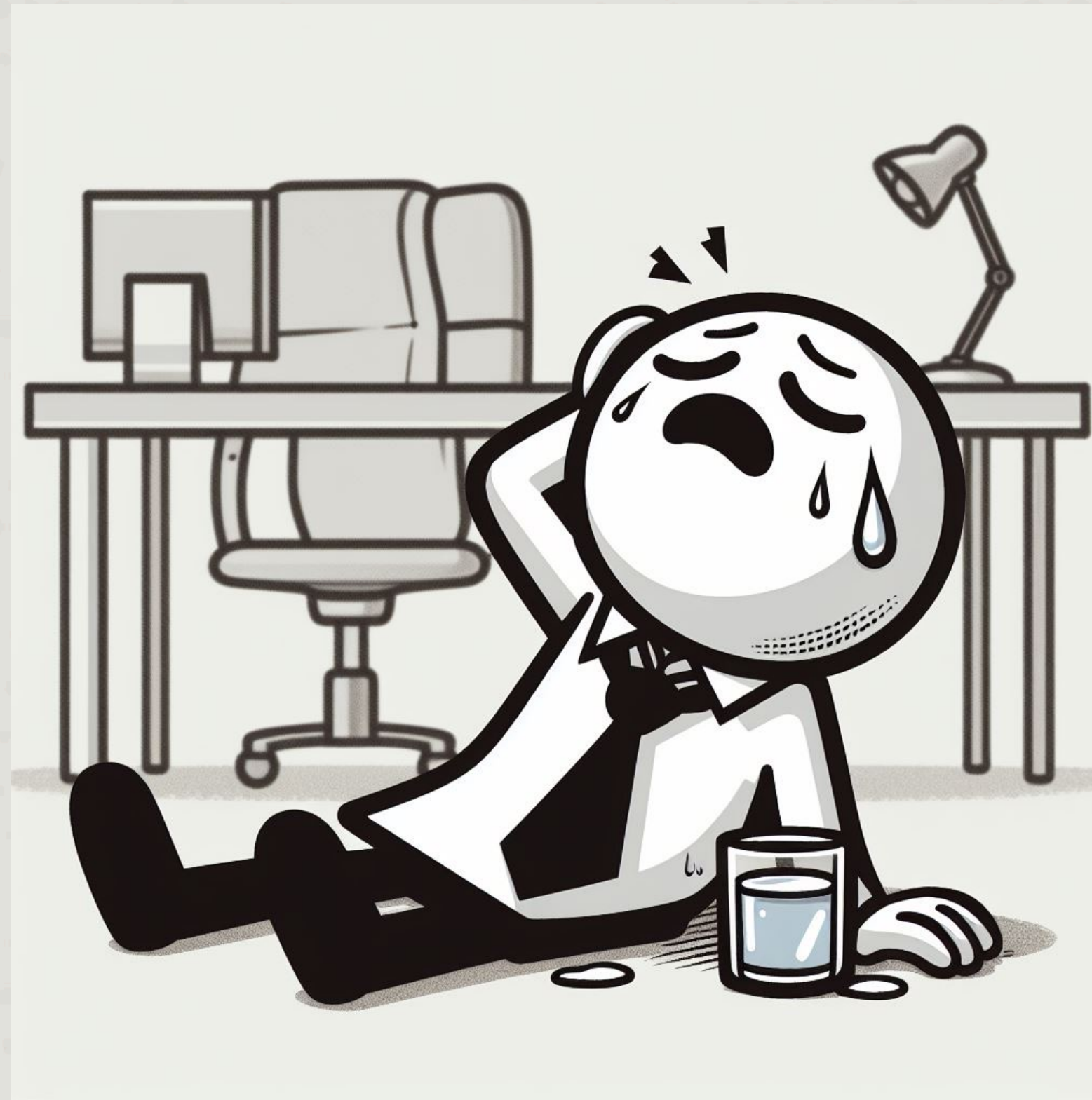
<https://designer.microsoft.com/image-creator>

JDK X+many

Typical JDK corporate adoption (so far)



JDK X

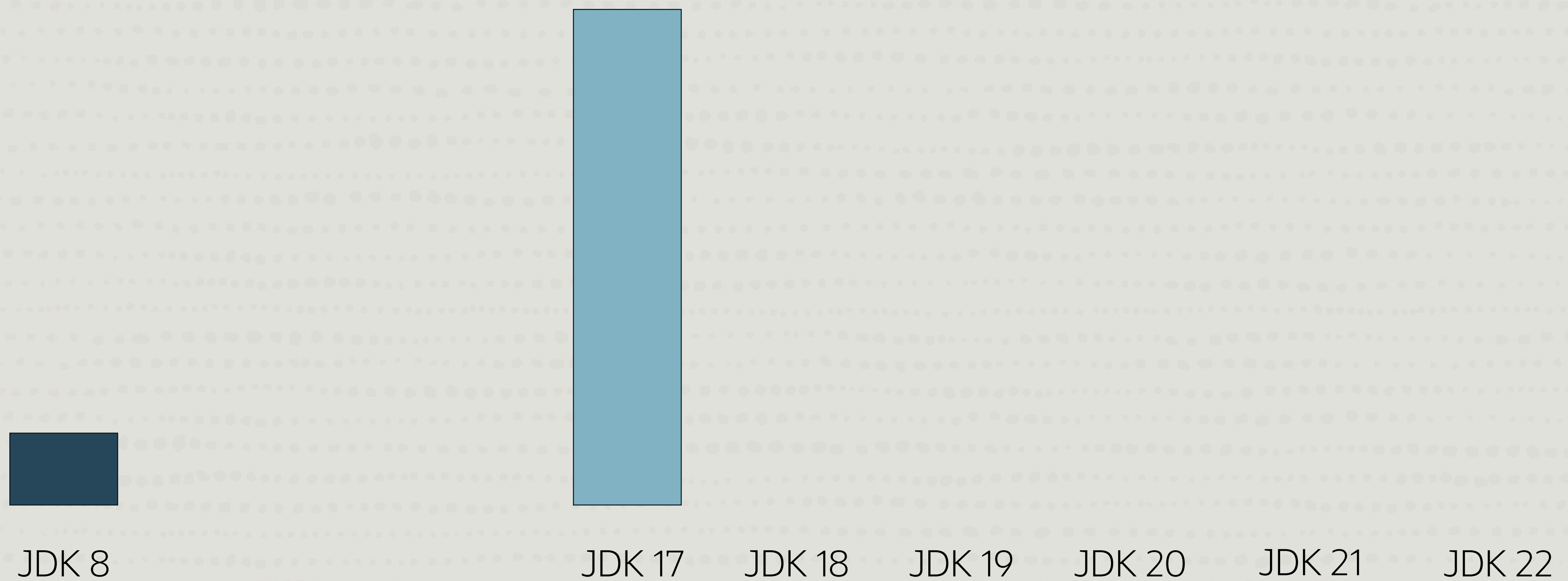


<https://designer.microsoft.com/image-creator>

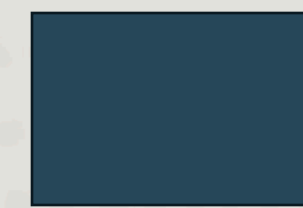


JDK X+many

Typical JDK corporate adoption (so far)



Better JDK corporate adoption

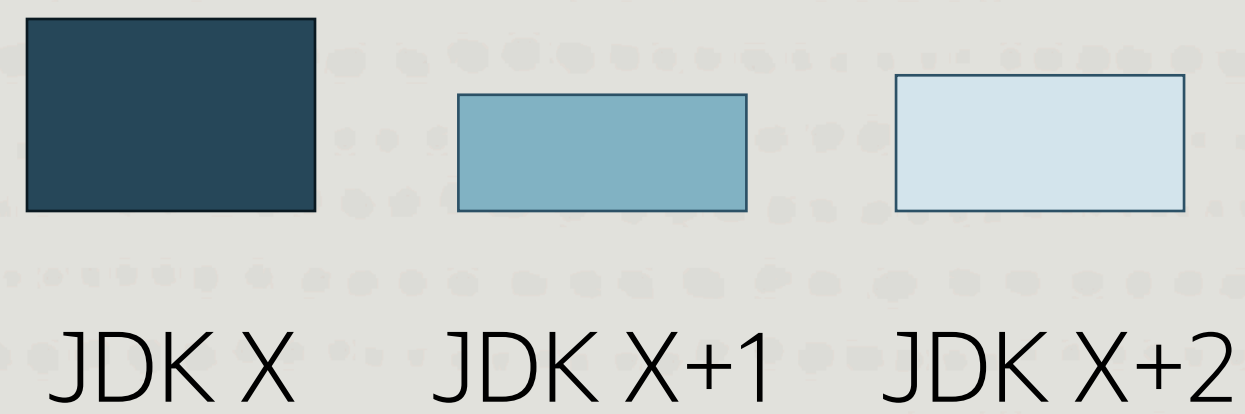


JDK X

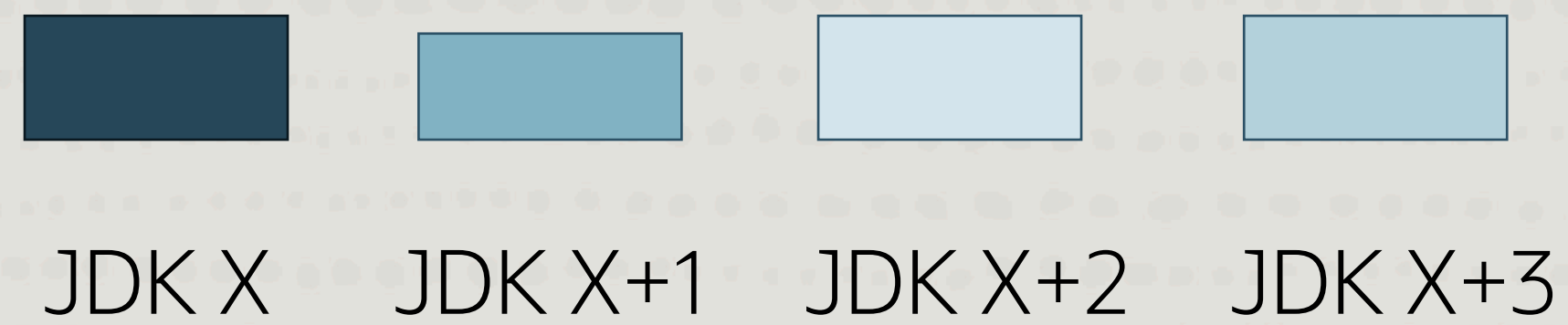
Better JDK corporate adoption



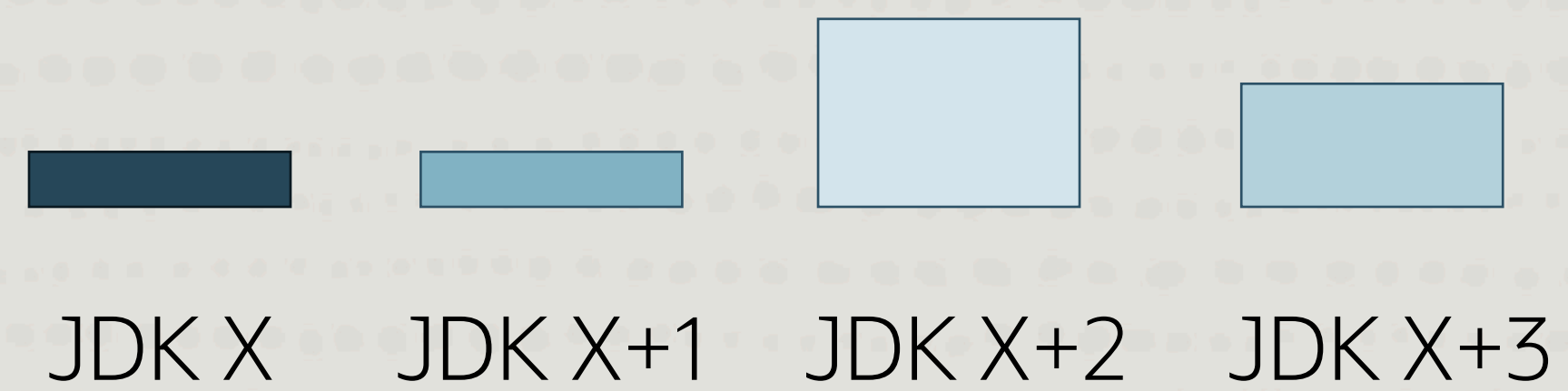
Better JDK corporate adoption



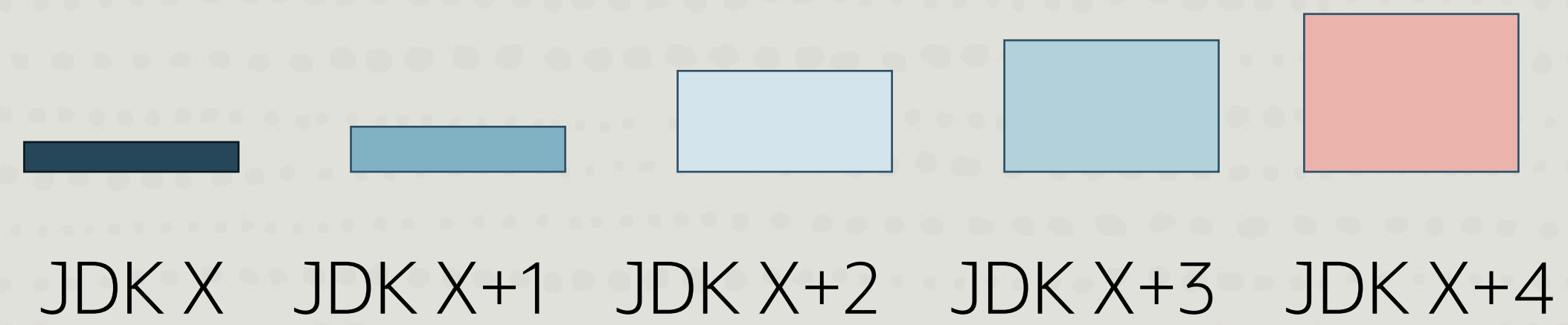
Better JDK corporate adoption



Better JDK corporate adoption



Better JDK corporate adoption

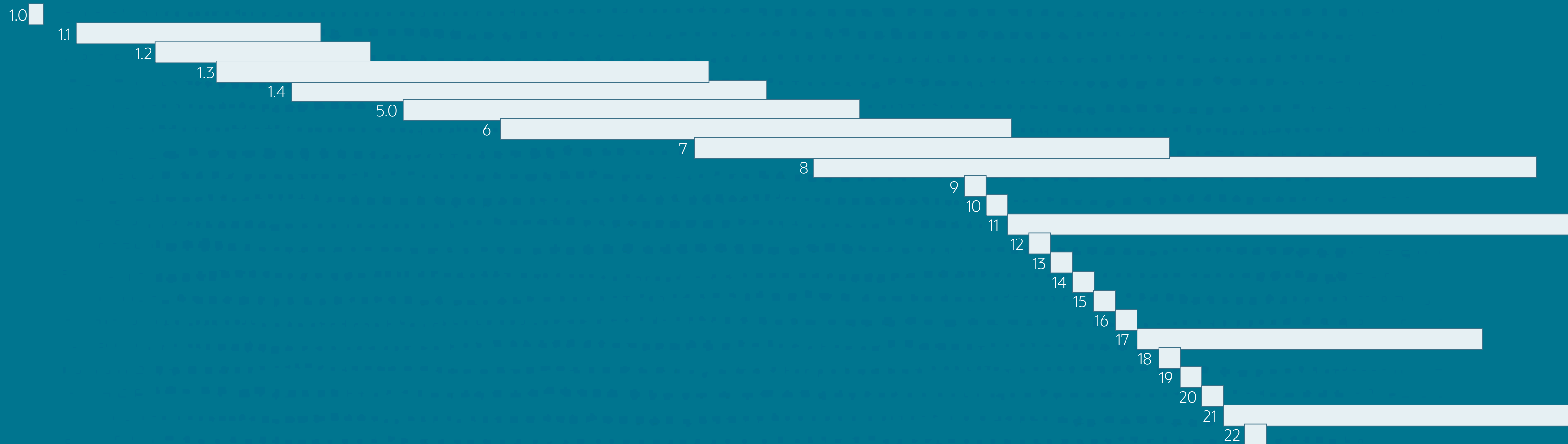


Should we use all releases?

For this model think of non-LTS releases as "part" of the **next** LTS

1995

2031

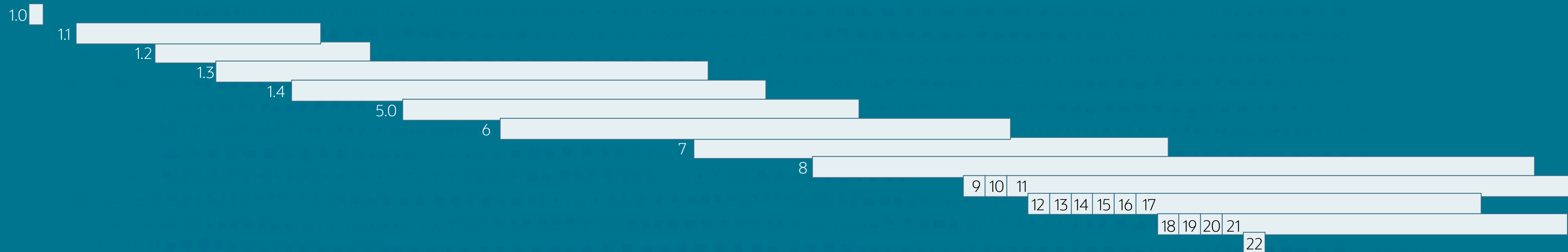


Should we use all releases? YES

For this model think of non-LTS releases as "part" of the **next** LTS

1995

2031



What do we need to make this model more common

Tools and libraries adopting/supporting new JDK versions quickly (ideally at GA)

- And supporting older Java versions long enough to allow applications to stay put for most/all their lifetime [with only critical fixes]

Development "best practices" and corporate culture to accept multiple runtime versions coexisting

- The right Java version depends on the type of application and where it is on its own lifecycle; not on what every other applications runs on

