


State of Java on RISC-V

JCP - April 2024

About

Software Engineer & Team Lead at  Rivos

- Managed Runtimes, System Libraries, Profiling

Contributor to 

- Windows-AArch64, macOS-AArch64, Linux-RISC-V ports

Language Runtimes WG at  RISE
RISC-V Software Ecosystem

- “collaborative effort [...] to accelerate the development of open source software for the RISC-V architecture”
- OpenJDK, Go, Python, .NET, ART, V8
- Compilers, Runtimes, and Ecosystem (libraries, tools)

Adoptium WG  ADOPTIUM

- Distributing LTS versions (11, 17, 21, 22)

What is RISC-V

- Open Standard ISA, introduced in 2014
 - Royalty free, open source licenses
 - Anyone can implement a RISC-V compliant processor
 - Community-led development, on [GitHub](#)
 - Billions of chips shipped annually, and growing rapidly
- [RISC-V International](#)
 - Foundation, not for profit
- Healthy ecosystem of Hardware and Software providers
 - Working together to define the specification
 - Targeting wide set of workloads: from microcontrollers to servers
 - Based on extensions (ex: V, C, Zba, etc.); simplified with [profiles](#)

What is RISC-V

- From embedded to servers
 - Companies announcing server-class hardware
 - Alibaba, Ventana, Rivos, SiFive
 - Mostly announcements, expecting some deliveries by late-2024 - mid-2025
 - Vendor-specific extensions
- Targeting AI workloads
 - Rivos with a Data Parallel Accelerator
 - Tenstorent with an Inference PCIe card
- International Hardware Market
 - Not tied to 1-3 companies for IP
 - More openly accessible market

Compilers / Runtimes / Libraries

- Support in many compilers/runtimes
 - GCC, LLVM, OpenJDK, Go, Python, .NET, V8, ART, and many more
 - Various degrees of quality and support
 - Rapidly evolving
 - Importance of latest and greatest
- Support in more and more libraries
 - Most of the upcoming work

Compilers / Runtimes / Libraries

- <https://landscape.riscv.org>



MySQL
Nginx
OpenBLAS
PyTorch
Android
Linux
Redis
Docker
Go
FreeBSD
...

OpenJDK: versions supported

- [JEP 422: Linux/RISC-V Port](#)
- Integrated in Java 19
- Backported to Java [11](#) and [17](#)

- Vendors
 - Eclipse Temurin: 21, 22, tip
 - 11 and 17 are work in progress
 - Bellsoft Liberica: 21 and 22
 - Ubuntu/Fedora: 11*, 17, 21, tip

[*] Based on ZeroVM, Full-interpreter, platform-agnostic mode of OpenJDK

OpenJDK: features supported

- Everything
 - Code Generation: Interpreter, C1, C2
 - Garbage Collectors: Epsilon, Serial, Parallel, G1, ZGC, Shenandoah
 - Serviceability: JVMTI, JFR, etc.
 - Desktop: AWT, Swing
 - And many more

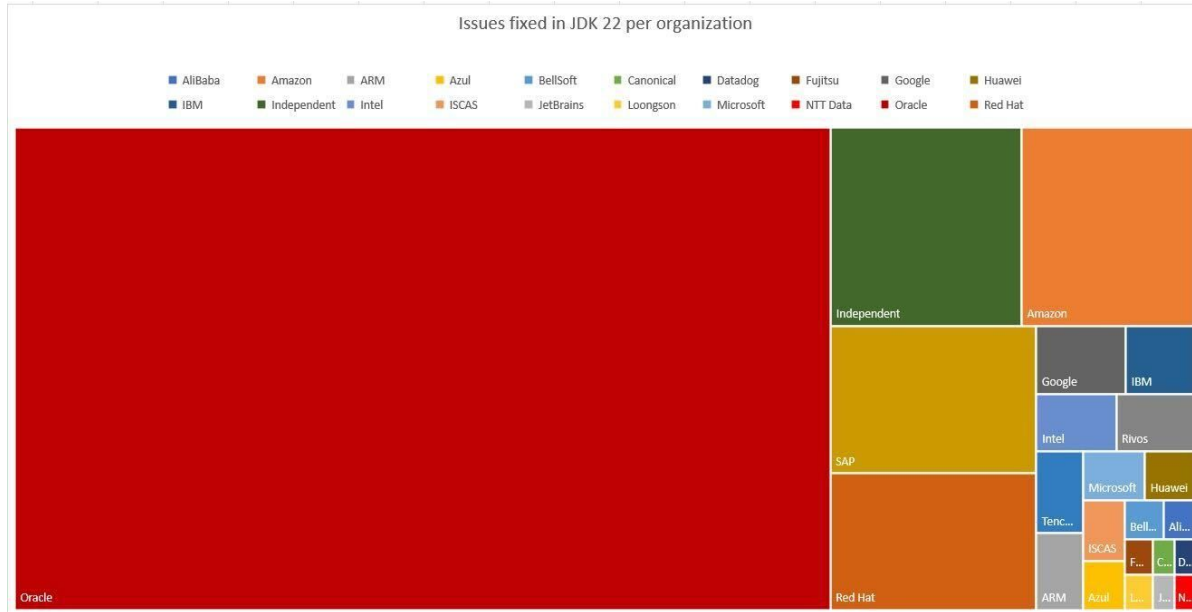
- TCK is passing on RISC-V on Java 17, 21, 22 and 23

OpenJDK: features supported

- Lots of work to accelerate OpenJDK
 - Intrinsic:
 - Cryptography: AES, SHA, MD5, etc.
 - Memory copy/zeroing
 - Math: montgomery multiply, rounding, etc.
 - Vector API
 - Code lowering
 - Bitmanip extensions
 - Floating points
 - Compiler optimizations
 - Memory allocation/zeroing
 - Autovectorization
- Taking advantage of all the hardware offers
 - Specifying new extensions (ex: for Garbage Collection)

OpenJDK: contributors

- Original port from Huawei; Reviewed by Red Hat
- Regular contributions from: Huawei, Alibaba, Rivos, ISCAS, Syntacore



Ecosystem

- Maven Central: more than 2M packages
 - Many are platform-agnostic, but some have platform-specific code
 - Some examples
 - [Netty](#): networking library
 - [RocksDB](#): embeddable, persistent key-value store
 - [Apache Commons Crypto](#)
 - [Snappy](#): compression algorithm
- Many transitive dependencies; what to prioritize?
 - We need the community's input

Ecosystem: contributors

- [RISE](#)
 - Accelerate the development of open source software for the RISC-V architecture
 - https://wiki.riseproject.dev/display/HOME/LR_00%3A+Java
 - Members are Google, Red Hat, Rivos, Alibaba/T-HEAD, Intel, Canonical, ISCAS, and more
 - [RISC-V Optimization Guide](#)
- [Adoptium](#)
 - <https://adoptium.net/blog/2024/04/eclipse-temurin-21-and-22-available-on-riscv/>
- RISC-V International
 - J-extension WG, Managed Runtimes SIG
 - Specify RISC-V extensions to accelerate Runtimes like OpenJDK, Go, and more

Challenges

- Lack of high-performance hardware
 - Developer boards are available: [VisionFive 2](#), [LicheePi 4](#), [HiFive Unmatched](#)
 - Risk of over-optimizing for Raspberry-Pi-sized hardware
 - Some performance accurate models, but all are closed-source
- Need to rely on emulation for most testing
 - Easily accessible, great for most testing, but not fast
- Need for more documentation
 - How to run emulation on your CI?
 - Which compilers/runtimes to use?
 - Which libraries/packages support RISC-V?
 - Looking for feedback and contributions

What's next?

- Continue investing in OpenJDK
 - More optimizations, more features
- Continue porting ecosystem libraries to RISC-V
 - Identify and Prioritize
 - Document, Support, and Sponsor
 - Machines
 - Contributions

Questions

Contact

ludovic@rivosinc.com

<https://github.com/luhenry>

https://mastodon.social/@ludovic_dev

OpenJDK

<https://mail.openjdk.org/pipermail/riscv-port-dev/>

Rivos

<https://www.rivosinc.com/>

<https://www.linkedin.com/company/rivos-inc/>

RISE

<https://riseproject.dev/>

<https://www.linkedin.com/company/risc-v-software-ecosystem-rise/>