OpenJDK

Installing
Contributing
Sponsoring
Developers' Guide
Vulnerabilities
JDK GA/EA Builds

Mailing lists
Wiki · IRC

Bylaws · Census
Legal

**Workshop**

**JEP Process**

**Source code**
Mercurial
GitHub

**Tools**
Git
jtreg harness

**Groups**
(overview)
Adoption
Build
Client Libraries
Compatibility & Specification Review
Compiler
Conformance
Core Libraries
Governing Board
HotSpot
IDE Tooling & Support
Internationalization
JMX
Members
Networking
Porters
Quality
Security
Serviceability
Vulnerability
Web

**Projects**
(overview, archive)
Amber
Audio Engine
CRaC
Caciocavallo
Closures
Code Tools
Coin
Common VM Interface
Compiler Grammar
Detroit
Developers' Guide
Device I/O
Duke
Font Scaler
Galahad
Graal
Graphics Rasterizer
IcedTea
JDK 7
JDK 8
JDK 8 Updates
JDK 9
JDK (..., 21, 22)
JDK Updates
JavaDoc.Next
Jigsaw
Kona
Kulla
Lambda
Lanai
Leyden
Lilliput
Locale Enhancement
Loom
Memory Model Update
Metropolis
Mission Control
Modules
Multi-Language VM
Nashorn
New I/O
OpenJFX
Panama
Penrose
Port: AArch32
Port: AArch64
Port: BSD
Port: Haiku
Port: Mac OS X
Port: MIPS
Port: Mobile
Port: PowerPC/AIX
Port: RISC-V
Port: s390x
Portola
SCTP
Shenandoah
Skara
Sumatra
Tiered Attribution
Tsan
Type Annotations
Valhalla
Verona
VisualVM
Wakefield
Zero
ZGC

ORACLE

# JEP 391: macOS/AArch64 Port

| | |
|---|---|
| *Authors* | Anton Kozlov, Vladimir Kempik |
| *Owner* | Vladimir Kempik |
| *Type* | Feature |
| *Scope* | JDK |
| *Status* | Closed / Delivered |
| *Release* | 17 |
| *Component* | hotspot |
| *Discussion* | aarch64 dash port dash dev at openjdk dot java dot net |
| *Effort* | M |
| *Duration* | M |
| *Depends* | JEP 388: Windows/AArch64 Port |
| *Reviewed by* | Andrew Haley, Vladimir Kozlov |
| *Endorsed by* | Vladimir Kozlov |
| *Created* | 2020/08/07 07:08 |
| *Updated* | 2022/11/23 04:45 |
| *Issue* | 8251280 |

## Summary

Port the JDK to macOS/AArch64.

## Non-Goals

- It is not a goal to implement all optional components (e.g., compiler intrinsics), even if they are implemented in other AArch64 ports.
- It is not a goal to support the write-xor-execute (W^X) memory-protection policy for targets other than macOS/AArch64.

## Motivation

Apple has announced a long-term plan to transition their line of Macintosh computers from x64 to AArch64. We therefore expect to see broad demand for a macOS/AArch64 port of the JDK.

Although it will be possible to run a macOS/x64 build of the JDK on AArch64-based systems via macOS's built-in Rosetta 2 translator, the translation will almost certainly introduce a significant performance penalty.

## Description

An AArch64 port already exists for Linux (JEP 237), and work is underway on an AArch64 port for Windows (JEP 388). We expect to reuse existing AArch64 code from these ports by employing conditional compilation — as is usual in ports of the JDK — to accommodate differences in low-level conventions such as the application binary interface (ABI) and the set of reserved processor registers.

macOS/AArch64 forbids memory segments from being executable and writeable at the same time, a policy known as *write-xor-execute* (W^X). The HotSpot VM routinely creates and modifies executable code, so this JEP will implement W^X support in HotSpot for macOS/AArch64.

## Testing

Testing will include, but not be limited to, compatibility testing with the TCK, regression testing with jtreg, and validation with applications. The execution environment will include development platforms available from Apple as well as consumer hardware, once it becomes available.

## Risks and Assumptions

- The changes for macOS/AArch64 risk breaking the existing Linux/AArch64, Windows/AArch64, and macOS/x64 ports. This risk will be reduced via extensive pre-integration testing.
- We expect to be able to implement the new ABI convention with reasonably small changes in the shared AArch64 code. We expect the footprint of the macOS-specific code to be small.
- We expect the macOS/AArch64 and Windows/AArch64 ports to be similar in some ways, allowing some code to be shared across these ports and further reducing the macOS-specific AArch64 code.
- We assume that the new version of macOS will not differ substantially from past versions, so that the amount of code change required for the new version will be small.
- We expect that supporting the W^X policy will be aided by operating-system services such as the `pthread_jit_write_protect_np` system call. If not, we will develop alternative approaches. The first implementation will target correctness with a possible performance penalty in uncommon cases, such as deoptimizations.

## Dependencies

The macOS/AArch64 port and the Windows/AArch64 port (JEP 388) will likely share some code. Some parts of this JEP will depend upon the integration of JEP 388, while other parts can be developed in parallel.