

Contributing Sponsoring Developers' Guide **Vulnerabilities** JDK GA/EA Builds Mailing lists

Wiki · IRC Bylaws · Census

Workshop

Legal

JEP Process Source code

Mercurial GitHub Tools

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 Babylon CRaC Caciocavallo Closures Code Tools Coin Common VM Interface Compiler Grammar Detroit Developers' Guide Device I/O Duke Galahad Graal

IcedTea IDK 7 JDK 8 JDK 8 Updates IDK 9 JDK (..., 21, 22, 23) JDK Updates JavaDoc.Next Jigsaw Kona Kulla Lambda Lanai Leyden Lilliput Locale Enhancement Loom Memory Model

Metropolis Mission Control Multi-Language VM Nashorn New I/O OpenJFX Panama Penrose Port: AArch32 Port: AArch64 Port: BSD Port: Haiku Port: Mac OS X

Update

Port: MIPS Port: Mobile Port: PowerPC/AIX Port: RISC-V Port: s390x Portola **SCTP** Shenandoah Skara Sumatra **Tiered Attribution**

Type Annotations Valhalla Verona VisualVM Wakefield Zero ZGC

Tsan

ORACLE

JEP 388: Windows/AArch64 Port

Authors Monica Beckwith, Ludovic Henry, Bernhard Urban-Forster

Owner Vladimir Kozlov Type Feature

Scope Implementation

Status Closed / Delivered Release 16

Component hotspot Discussion aarch64 dash port dash dev at openjdk dot java dot net

Effort M Duration S

Blocks | IEP 391: macOS/AArch64 Port

Reviewed by Vladimir Kozlov

Endorsed by Mark Reinhold, Vladimir Kozlov

Created 2020/06/29 19:13 Updated 2021/08/28 00:28

Issue 8248496

Summary

Port the JDK to Windows/AArch64.

Motivation

With the release of new consumer and server-class AArch64 (ARM64) hardware, Windows/AArch64 has become an important platform due to end-user demand.

Description

We have ported the JDK to Windows/AArch64, by extending the work previously done for the Linux/AArch64 port (JEP 237). This port includes the template interpreter, the C1 and C2 JIT compilers, and garbage collectors (serial, parallel, G1, Z and Shenandoah). It supports both the Windows 10 and Windows Server 2016 operating systems.

The focus of this JEP is not the porting effort itself, which is mostly complete, but rather the integration of the port into the JDK main-line repository.

Currently, we have a little over a dozen changesets. We have kept the changes to the shared code to a minimum. Our changes extend the support of the AArch64 memory model to Windows, address some MSVC issues, add LLP64 support to the AArch64 port, and perform CPU feature detection on Windows. We have also modified the build scripts to better support cross-compilation and the Windows toolchain.

The new platform code by itself is confined to 15 (+4) files and 1222 lines (+322). Early-access binaries are available here.

Testing

We have done thorough functional, performance, and regression testing on AArch64 test systems including Ampere eMAG1s, Marvell ThunderX2s (with SMT both on and off), and Surface Pro Xs. Our test suites include JTReg, JCStress, jmhjdk-microbenchmarks, JFC applications and applets (demo/jfc), SPECJBB2005, SPECIBB2015, SPEC SERT, SPECIVM2008, DaCapo, and a few smaller benchmarks. Two of the test matrices are documented in detail: |TReg and SPEC|BB2015.

Apart from the AArch64 systems, we have also tested our patches on x64 systems (Intel Skylakes). Our test coverage includes Linux/AArch64, Windows/AArch64, Windows/x64, and, only for functional testing, Linux/x64.

We have a robust CI for functional tests and will continue running our patches through the CI as we make further changes.

Risks and Assumptions

- We have made changes to shared code in the Linux/AArch64 backend to extend LP64 (64-bit long ints and pointers) to accommodate the long-long ints and pointers (LLP64) needed by the 64-bit Windows platform.
- We have also modified shared code on Windows, as mentioned above.
- We need to rework the usage of register x18, which on Windows/AArch64 is used for the Windows Thread Environment Block (TEB). We are working together with the Lead of the AArch64 Port project to find a graceful solution.