
ORACLE®


Java™
ORACLE®

# JDK 9

## A Sneak Peek

Dalibor Topić
Principal Product Manager
OpenJDK Adoption Group Lead
March 16th, 2015

# Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

# Program Agenda

1. OpenJDK Community
2. JEP 2.0 Process
3. JDK 9 Status
4. Participation
5. Q & A

# OpenJDK Community

**OpenJDK.Java.net**

# OpenJDK Community

- "The place to collaborate on an open-source implementation of the Java Platform, Standard Edition, and related projects."

# OpenJDK Community – Lots Of Projects

- Annotations Pipeline 2.0
- Audio Engine
- Build Infrastructure
- Caciocavallo
- Closures
- Code Tools
- Coin
- Common VM Interface
- Compiler Grammar
- Device I/O
- Font Scaler
- Framebuffer Toolkit
- Graal
- Graphics Rasterizer
- HarfBuzz Integration
- IcedTea
- JDK 6
- JDK 7
- JDK 7 Updates
- JDK 8
- JDK 8 Updates
- JDK 9
- JavaDoc.Next
- Jigsaw
- Kulla
- Lambda
- Locale Enhancement
- Memory Model Update
- Modules
- Multi-Language VM
- Nashorn
- New I/O
- OpenJFX
- Panama
- Penrose
- Port: AArch64
- Port: BSD
- Port: Haiku
- Port: Mac OS X
- Port: MIPS
- Port: PowerPC/AIX
- SCTP
- Sumatra
- ThreeTen
- Tiered Attribution
- Type Annotations
- XRender Pipeline
- Valhalla
- VisualVM
- Zero

# OpenJDK Community – Lots Of Individuals

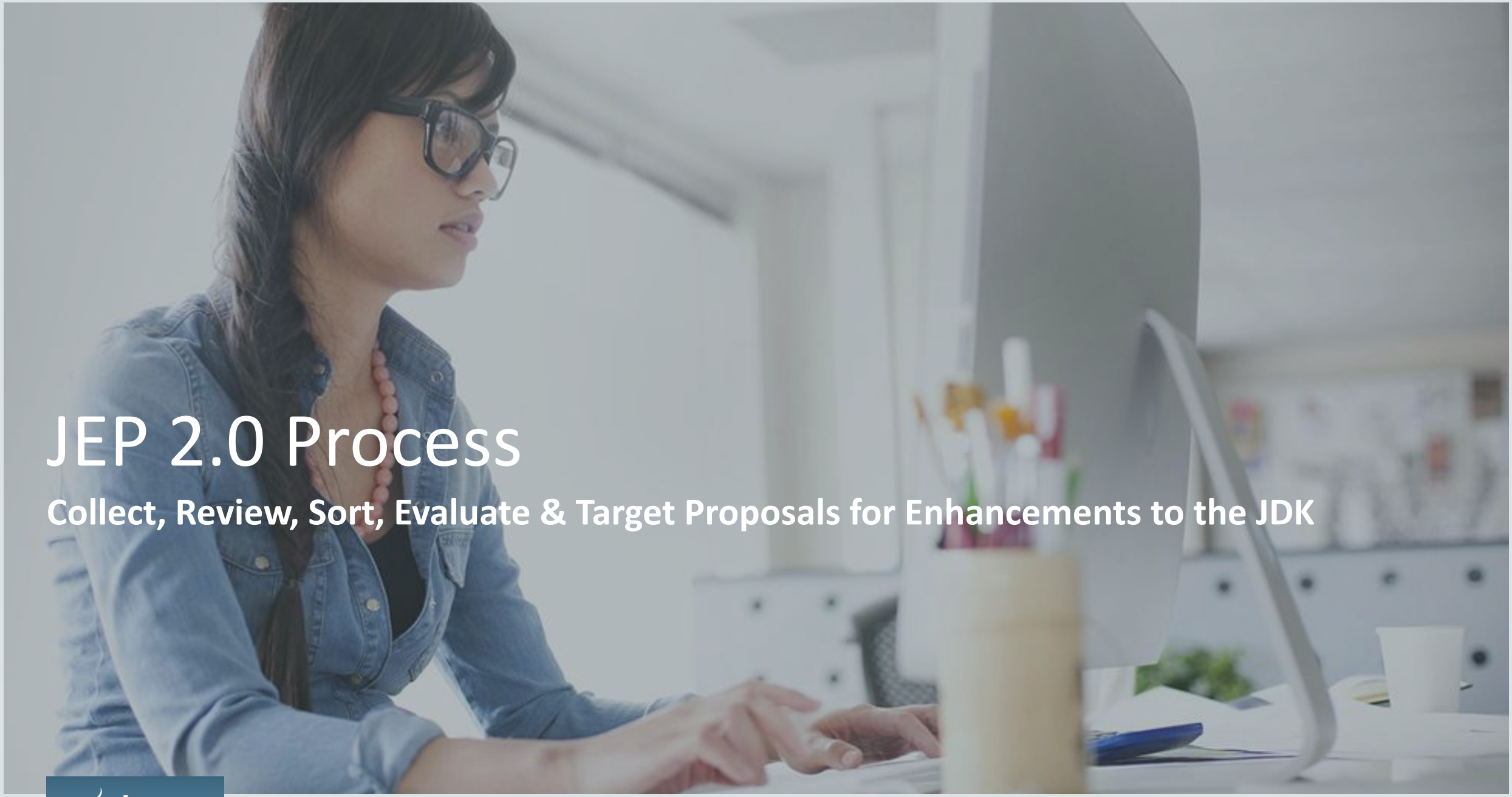
## As Authors, Committers, Reviewers, Leads, Members in different Projects & Groups

aaivanov aastrand abuckley acorn adinn adlertz aefimov aeremeev aeriksso afedorch agabrielsson aharlap ahe ahgross aivanov akasko akhil alanb albertz alevasil alexp alexsch alin alitvinov alkonsta allwin almatvee alundblad amenkov amironov amlu amurillo anashaty anazarov andreas andrew anoll ant anthony apangin apetrusenko apetushkov aph aredko arieber art asaha ascarpino aschenkm asemenyuk asiebenborn asmotrak asmundak astrange attila avstepan avu aw aywang azakharov azeemj azhebel azvegint bachmann bae bagiras batsatt bcbeck bchristi bdelsart behdad bharadwaj billyh bino bkurotsu blacklion bmoloden bobv bpassani bpatel bpb bpittore briand briangoetz bristor brutisso burban bvaidya campbell cbasha cbensen ccenotti ccheung ceisserer cfang cgruszka chaeubl chegar chrisphi christos chumer chunma cjplummer ckyang cl clucasius coffeys coleenp collins courington cquinn ctornqvi cwimmer cwirth daguangzhu darcy dav david davidb dbhole dblaukop dbryant dbuck dcherepanov dcubed ddehaven ddhill ddurrence denis dermashov dfazunen dfuchs dgollapudi dgrieve dgu dholmes dhorak dice dingxmin dl dli dlila dlong dlsmith dmalav dmarkov dmasada dmeetry dmocek dnsimon dougfelt dpochepk drchase drwhite dsamersoff dsimms dtitov duke dwanvik dxu ecaspole egahlin ehelin eistepan ejunberg ekleyman ekrejcir elandau eleponn emc emcmanus enevill erikj ersh ewang ewendeli ewong farvidsson faryad fheidric fjennings flar forax fparain fweimer fyuan fzhinkin gadams gafter gbenson gdub georges.saab gfrost gkbrown glewis gls goetz gsm gthornbr gtriantafill gziemski hamish hannesw hawtin hdong headius hedberg heikki henryjen herrick hinkmond hiroshi hirt hloef hosterda hseigel hualan huntch idk igerasim igor ihse iignatyev iklam ikrylov iris ivant iveresov jandrese janvalenta jasper jbachorik jcambon jccollet jchen jcoomes jdinga jdn jeff jeisl jenda jennyh jfdenise jferaud jfranck jgiles jgish jgodinez jgordon jhollida jiangli jjb jjg jjh jlahoda jlaskey jld jmanson jmasa jmcglynn jmelvin jnimeh joehw john.duimovich johnc jonas jonathan jpaetzold jprovino jqzuo jrose jrufus jtristan jtroyal jtusla juh jukim jviswana jwha jwilhelm jyoon jysong jzavgren kalli kamg karianna karunakar katakai katleman kbarrett kbr kcr kevinw khazra kizune klooney klward kmiller kmo kselle kshefov ksrini kurosaki kurt kvn kwwong lagergren lana lancea landonf langel ldayne leifs leonidr lfoltan lhowes littlee lmalvent lmesnik lnerad loefty loneid lowasser lpriima lstadler luchsh lucyzhao lvjing malenkov mali martin maxelsso mbankal mbykov mcastegr mcherkas mchicharro mchung mcimadamore mdoerr mduigou mernst mfang mgerdin mgoldyn mgrebac mgrimmer mgronlun mhaupt mheinrichs mhowe miauno michaelm miflemi mikael mikejwre minqi miroslawzn misterm mjjordan mknapp mkos mlapshin mlarsson mluyao mlvande mnunez mnutter mo morris mpersson mr mrigger mrkam msarkar mschoene mseledtsov msheppar msladecek msoch mtobiass mtvo mullan mvfranz mwong nadezhin nam naoto neliasso neugens never newell ngmr ngthomas nikgor nloodin npoore ntoda ogb ogino ohair ohrstrom okutsu olagneau olof omajid omazurov parrt paulk pbhat pbk pchelko pchistyakov pchopra peterjones peterz peytoia pgovereau phejl phh pjr plevart pliden pnauman poonam ppantele ppisl ppunegov prappo prr psafrata psandoz psomashe psoper psun ptbrunet ptisnovs pzhang qgong radko raginip raimandi ramap rasbold ratayar rbackman rbair rdurbin redestad resii rfield rgallard rgoyal rgurma rhalade rhoover rinaldo rkennke rlamothe rlewis robilad robm rodonnel roland rottenha rratta rriggs rschatz rupashka rwarburton sadayapalam sangheki santhoshla sbhusarapu sbirger sbohne schien scolebourne serb sergei sflores sfriberg sgabdura sgehwolf sgoldman shade sharonz shemnon sherman shinyafox shirishk shommel shurailine simonis sjiang sjohanss skannan skoivu skoppar skovalev skovatch skuksenko sla slions slugovoy smarks snazarki snorthov sogoel son spoole srikchan srl ssides sspitsyn stanleyh stayer stefank stephenh strarup stsmirno stuefe sundar swamyv swingler tamao tbell tdeneau tdv tellison tgranat thartmann thomaswue thor thurka tmarble tonyp tonywyant tpivovarova trims tschatzl twisti tyan ulfzibis uta vadim vaibhav valeriep van vasya vbaranov vdrozdov vikram vinnie vita vkarnauk vkempik vladidan vlivanov volk vrao vromero vshubov vvenkat walles weijun wenjyang wetmore wmdietl wmeissner wrockett xdono xiaofeya xlu xudwu xuelei xwang yan yangjiang yanqchen yaow ygaevsky yhuang yjoan ykantser youdwei ysr ysuenaga zailiu zgu zhangshj zhouyx zmajo

# OpenJDK Community – Lots Of Code

Let‘s look at OpenHUB statistics for JDK 9

Source: https://www.openhub.net/p/openjdk



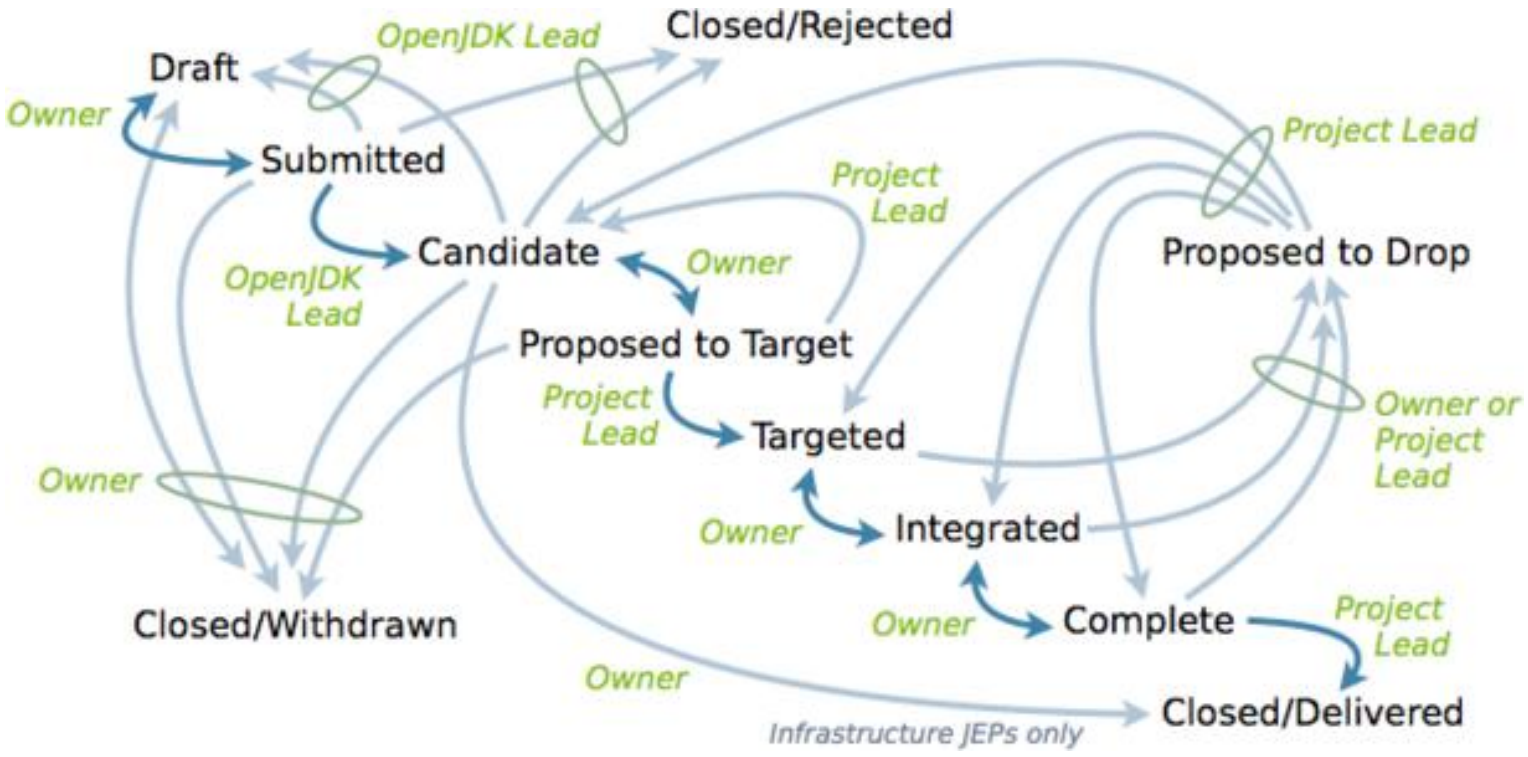
# JEP 2.0 Process

## Collect, Review, Sort, Evaluate & Target Proposals for Enhancements to the JDK

# JEP 2.0 Process

- Any Committer to a Project may propose to target a Feature JEP to a release of that Project after documenting a realistic engineering plan.
- The owner of a targeted JEP may later propose to drop that JEP from its targeted release.
- The Project's Lead may propose an initial release schedule, and thereafter propose changes to that schedule.
- Proposed changes to the feature set and schedule of a release are approved by rough consensus of the Committers to the Project, as determined by the Project Lead.

# JEP 2.0 Process



# JDK 9 Status

# JDK 9 Status

## http://openjdk.java.net/projects/jdk9/

### JDK 9

The goal of this Project is to produce an open-source reference implementation of the Java SE 9 Platform, to be defined by a forthcoming JSR in the Java Community Process.

**Status**

Features for the release are proposed and tracked via the JEP Process, as amended by the JEP 2.0 proposal.

**JEPs targeted to JDK 9, so far**

102: Process API Updates
143: Improve Contended Locking
158: Unified JVM Logging
165: Compiler Control
197: Segmented Code Cache
199: Smart Java Compilation, Phase Two
201: Modular Source Code
211: Elide Deprecation Warnings on Import Statements
212: Resolve Lint and Doclint Warnings
213: Milling Project Coin
214: Remove GC Combinations Deprecated in JDK 8
216: Process Import Statements Correctly
217: Annotations Pipeline 2.0
219: Datagram Transport Layer Security (DTLS)
220: Modular Run-Time Images
224: HTML5 Javadoc
228: Add More Diagnostic Commands
229: Create PKCS12 Keystores by Default
230: Microbenchmark Suite
231: Remove Launch-Time JRE Version Selection
235: Test Class-File Attributes Generated by javac
236: Parser API for Nashorn
237: Linux/AArch64 Port

Last update: 2015/2/5 04:29 -0800

# JEP 102: Process API Updates

**Improve the API for controlling and managing operating-system processes**

- Many enterprise applications and containers involve several Java virtual machines and processes and have long-standing needs which include:
  - The ability to get the pid (or equivalent) of the current Java virtual machine and the pid of processes created with the existing API.
  - The ability to get/set the process name of the current Java virtual machine and processes created with the existing API (where possible).
  - The ability to enumerate Java virtual machines and processes on the system.
  - The ability to deal with process trees, in particular some means to destroy a process tree.
  - The ability to deal with hundreds of sub-processes, perhaps multiplexing the output or error streams to avoid creating a thread per sub-process.

# JEP 212: Resolve Lint and Doclint Warnings

**Fix numerous lint and doclint errors as reported by javac in JDK code base**

- This JEP proposes to complete efforts to fix warnings that have been underway in JDK 8 and JDK 9 as well as to formalize a subset of source-code improvements previously proposed to jdk9-dev.
  - Most of the warnings are resolved by modifying the interior of method bodies.
  - Resolving some of rawtypes warnings involves changing method signatures, such as changing a parameter type from a raw java.lang.Class to a java.lang.Class<?> or some more specific type.
  - Any API changes will stay within the general evolution policy of the JDK.

# JEP 214: Remove GC Combinations Deprecated in JDK 8

**Remove the GC combinations that were previously deprecated in JDK 8 via JEP 173.**

- As described in JEP 173 there is a large maintenance cost associated with keeping all of the existing GC combinations around.
  - Removing the deprecated GC combinations will allow for many simplifications in the HotSpot GC code, which in turn will reduce the number of bugs and allow for more rapid development of the remaining GC combinations.
  - See JEP 214 for a detailed summary of the flags and flag combinations that will stop working.

# JEP 201: Modular Source Code

## Reorganize the JDK source code into modules

- This JEP is part of the first step of Project Jigsaw; later JEPs will modularize the JRE and JDK images (JEP 220) and then introduce a module system.
- The motivations to reorganize the source code at this early stage are to:
  - Give JDK developers the opportunity to become familiar with the modular structure of the system;
  - Preserve that structure going forward by enforcing module boundaries in the build, even prior to the introduction of a module system; and
  - Enable further development of Project Jigsaw to proceed without always having to "shuffle" the present non-modular source code into modular form.

# JEP 220: Modular Run-Time Images

## Restructure the JDK and JRE run-time images to accommodate modules

- **Goals**
  - Adopt a run-time format for stored class and resource files that:
    - Is more time- and space-efficient than the legacy jar format, which in turn is based on the ancient zip format;
    - Can locate and load class and resource files on a per-module basis;
    - Can store class and resource files from JDK modules and from library and application modules; and
    - Can be extended to accommodate additional kinds of data going forward, such as precomputed JVM data structures and precompiled native code for Java classes.
  - Restructure the JDK and JRE run-time images.
  - Provide supported ways to perform common operations such as, *e.g.*, enumerating all of the classes present in an image.
  - Enable the selective *de-privileging* of JDK classes

# JEP 220: Modular Run-Time Images

**Restructure the JDK and JRE run-time images to accommodate modules**

- A modular run-time image will contain the following directories:
  - The bin directory will contain any command-line launchers defined by the modules linked into the image.
  - The conf directory will contain the .properties, .policy, and other kinds of files intended to be edited by developers, deployers, and end users, which were formerly found in the lib directory or subdirectories thereof.
  - The lib directory on Mac OS, or the lib/$ARCH directory on Linux and Solaris, will contain the run-time system's dynamically-linked native libraries, as it does today.
  - All other files and directories in the lib directory must be treated as private implementation details of the run-time system. They are not intended for external use and their names, format, and content will be subject to change without notice.

# JEP 220: Modular Run-Time Images

**Restructure the JDK and JRE run-time images to accommodate modules**

- **Removed: The endorsed-standards override mechanism**
  - A modular image is composed of modules rather than jar files. Going forward we expect to support endorsed standards and standalone APIs in modular form only, via the concept of *upgradeable modules*.
- **Removed: The extension mechanism**
  - To simplify both the Java SE Platform and the JDK we propose to remove the java.ext.dirs system property, the lib/ext directory, and the code that implements this mechanism.
  - The compiler and the launcher will ignore the platform-specific system-wide extension directory by default, but if the -XX:+CheckEndorsedAndExtDirs command-line option is specified then they will fail if that directory exists and is not empty.

# JEP 220: Modular Run-Time Images

**Restructure the JDK and JRE run-time images to accommodate modules**

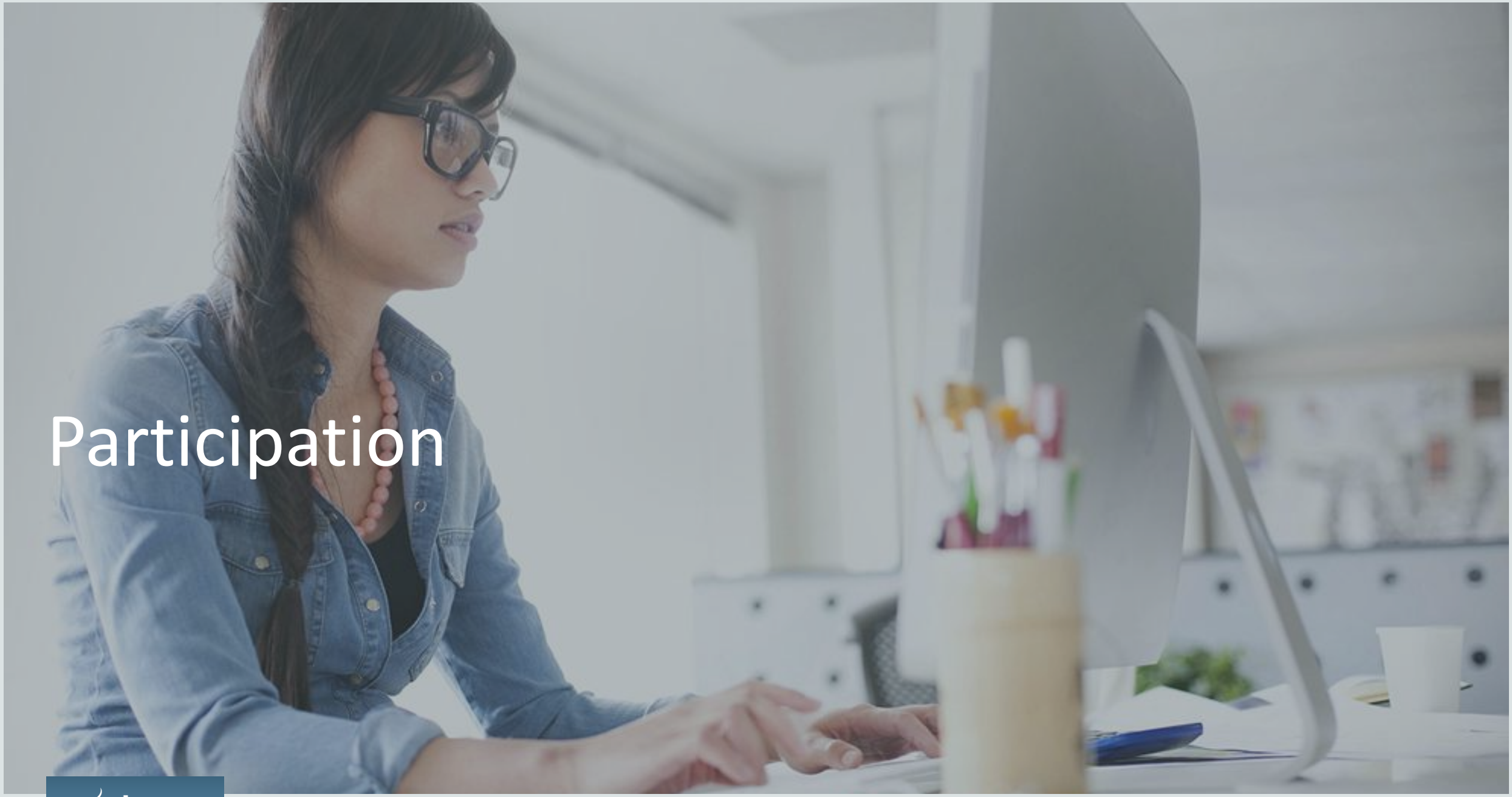
- **Removed: rt.jar and tools.jar**
  - The class and resource files previously stored in lib/rt.jar, lib/tools.jar, lib/dt.jar, and various other internal jar files will now be stored in a more efficient format in implementation-specific files in the lib directory. The format of these files will not be specified and is subject to change without notice.
- **New URI scheme for naming stored modules, classes, and resources**
  - we propose to define a new URL scheme, jrt, for naming the modules, classes, and resources stored in a run-time image without revealing the internal structure or format of the image.
  - Example: jrt:/java.base/java/lang/Class.class

# JEP 237: Linux/AArch64 Port

## Port JDK 9 to Linux/AArch64

- AArch64 is the new processor architecture from ARM Holdings plc. It is a departure from the 32-bit ARM processor architecture, and is effectively a complete redesign.
- We (the AArch64 Porting Project) have ported the JDK to a new platform: Linux/AArch64. We have implemented the template interpreter, the C1 (client) and the C2 (server) JIT compilers.
- Red Hat and Linaro regularly build and test the ports on the porting platforms as well as on the currently-supported JDK 8 platforms to ensure that no regressions are being introduced which break the existing platforms.

# Participation



# Participation

- Download and test JDK 9 Early Access (EA) builds
  - See https://jdk9.java.net/ for downloads
  - If you‘re a core developer on a FOSS Project, join Quality Outreach effort
    - https://wiki.openjdk.java.net/display/Adoption/Quality+Outreach
    - Let us know about regressions & showstoppers you find testing your project against EA builds
- Provide feedback on Draft JEPs on their OpenJDK mailing lists
  - See http://openjdk.java.net/jeps/0 for list and details
- If you want to contribute changes, see http://openjdk.java.net/contribute/

# Preparation

- Remove invalid VM options from launcher scripts
  - Example: "-XX:PermSize" and "-XX:MaxPermSize“
    - All VM options related to PermGen were deprecated in JDK 8 and removed in JDK 9, on all platforms.
- Run JDK 8 jdeps tool on your code with ***-jdkinternals*** option
  - With jdeps you can find *static* dependencies to any JDK internal APIs that are
    - unsupported and
    - private to JDK implementation
  - See https://wiki.openjdk.java.net/display/JDK8/Java+Dependency+Analysis+Tool

# Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

# Hardware and Software
## Engineered to Work Together


Java™
ORACLE®


ORACLE®