Moving Java Forward Faster

Sharat Chander
Senior Director, Java Product Management and Developer Relations
Java Platform Team

Twitter: @Sharat_Chander
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.
About me

– Director of Product Management for the Java Platform Group at Oracle
– In charge of managing the outbound product requirements for Oracle’s JDK since joining Oracle through the Sun acquisition in 2010 including go-to-market, sales enablement and partner support opportunities
– Also in charge of Java developer relations and community enablement
– Before joining Oracle, worked at Sun Microsystems
Communication  
Java Magazine  
250K+ subscribers

Community  
Java User Groups  
350+ worldwide

Collaboration  
Java Champions  
150+ worldwide

Contribution  
OpenJDK  
470 community participants
#1 Programming Language

12 Million Developers Run Java

38 Billion Active Virtual Machines

21 Billion Cloud Connected Virtual Machines
Java SE is #1 Runtime in the Cloud

- #1 Deployment runtime on AWS and Google App Engine and #3 on MS Azure
- Java Runtime is the foundation of the Cloud IaaS, PaaS and SaaS

Source: 2015 Vision Mobile
OpenJDK Platform Investments

• Security is our **#1 priority**
• Improving Java developer productivity and compatibility (Amber, Panama, Loom)
• Increasing density (Valhalla)
• Improving startup time (AOT, App CDS)
• Improving predictability (zGC, Shenandoah)
• Simplifying serviceability and profiling (JFR, JMC)
The New Release Model

No more limousines, think trains!
Previous JDK Release Model

Y1 Y2 Y3 Y4 Y5 Y6 Y7 Y8 Y9 Y10 Y11 Y12 Y13 Y14 Y15
Previous JDK Release Model
Previous JDK Release Model
Previous JDK Release Model

Y1  Y2  Y3  Y4  Y5  Y6  Y7
8    8u20  8u40  8u60
9   9.1  9.2  9.3
10  10.1 10.2 10.3
11  11.1 11.2 11.3
New JDK Release Model – Feature releases every 6 months
New JDK Release Model

Y1  Y2  Y3  Y4  Y5  Y6  Y7

9
10
11
12
13
New JDK Release Model - LTS Releases

Y1  Y2  Y3  Y4  Y5  Y6  Y7

9

10

11 (18.9 LTS)

12

13
New JDK Release Model - LTS Every 3 years

Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11, Y12, Y13, Y14, Y15

11 (18.9 LTS)

17 (21.9 LTS)
New JDK Release Model – Starting with JDK 9

- JDK 6
- JDK 7
- JDK 8
- JDK 9
- JDK 10
- JDK 11 (18.9 LTS)
- JDK 12
- JDK 13
- JDK 14
- JDK 15
- JDK 16
- JDK 17 (21.9 LTS)
Oracle JDK & OpenJDK


JDK 8

Oracle JDK - BCL

Oracle JDK - OTN

OpenJDK - GPL

JDK 6

JDK 7

JDK 8

Oracle JDK

OpenJDK

Java

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.
New OpenJDK binaries
Moving Java Forward Faster and more open! (Opener?)

Accelerating the JDK release cadence

mark.reinhold at oracle.com  mark.reinhold at oracle.com
Wed Sep 6 14:49:28 UTC 2017

Over on my blog today I’ve argued that Java needs to move forward faster. To achieve that I’ve proposed that the Java SE Platform and the JDK shift from the historical feature-driven release model to a strict, time-based model with a new feature release every six months, update releases every quarter, and a long-term support release every three years:

https://mreinhold.org/blog/forward-faster

Here are some initial thoughts on how we might implement this proposal here in the OpenJDK Community. Comments and questions about both the proposal and its implementation are welcome on this list.

- After JDK 9 we'll open-source the commercial features in order to make the OpenJDK builds more attractive to developers and to reduce the differences between those builds and the Oracle JDK. This will take some time, but the ultimate goal is to make OpenJDK and Oracle JDK builds completely interchangeable.

- Finally, for the long term we’ll work with other OpenJDK contributors to establish an open build-and-test infrastructure. This will make it easier to publish early-access builds for features in development, and eventually make it possible for the OpenJDK Community itself to publish authoritative builds of the JDK.

Oracle will now produce OpenJDK builds

The new OpenJDK builds will be licensed under GPL V2

GNU General Public License Version 2 with Class Path Exception (GPL 2 with CPE)

New Java feature release will be made every 6 months

Oracle will open source commercial features

Oracle will work with other OpenJDK contributors to make the community infrastructure complete, modern and accessible

URL: http://mail.openjdk.java.net/pipermail/discuss/2017-September/004281.html
From Oracle JDK to OpenJDK from Oracle
What Is Being Open-Sourced in Java

- **Java Mission Control**
  - Monitor and manage Java applications with minimal performance overhead

- **Java Flight Recorder**
  - Collects diagnostic and profiling data about a running Java application

- **Application Class Data Sharing**
  - Enables you to place classes from the standard extensions directories and the application class path in the shared archive

- **Java Usage Tracker**
  - Tracks how the JRE’s are being used in your systems

- **Infrastructure**
Late binding of features
**Oracle JDK Roadmap (as seen in Jul 2010)**

Proprietary

- Didn’t work out exactly as planned
- JDK 8 released on March 2014
- JDK 9 released Sep 2018
  - Jigsaw was the release driver

<table>
<thead>
<tr>
<th>JDK 7 – July 2011</th>
<th>JDK 8 – H2 2012</th>
<th>JDK 9 - 2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Coin – Small language enhancements (JSR 334)</td>
<td>• Jigsaw (Modularization)</td>
<td>• Data Bindings</td>
</tr>
<tr>
<td>• Dynamic Language Support (JSR 292)</td>
<td>• Lambda (Closures)</td>
<td>• Language Interop</td>
</tr>
<tr>
<td>• Concurrency and Collections updates (JSR 166y)</td>
<td>• Annotations on Java types (JSR 308)</td>
<td>• JNI Replacement</td>
</tr>
<tr>
<td>• Network and File System (JSR 203)</td>
<td>• More Small Language Enhancements</td>
<td>• Type Reification</td>
</tr>
<tr>
<td>• Security</td>
<td>• Complete JVM Convergence</td>
<td>• (TBD)</td>
</tr>
<tr>
<td>• Internationalization</td>
<td>• JavaScript Interop</td>
<td></td>
</tr>
<tr>
<td>• Other Miscellaneous Enhancements</td>
<td>• New Data Serialization Formats (JSON, YAML)</td>
<td></td>
</tr>
<tr>
<td>• HotSpot/JRockit Convergence</td>
<td>• Integrated JavaFX</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Improved Device Support (Location, Multi Touch, ...)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Misc Enhancements to Security, Networking, Internationalization</td>
<td></td>
</tr>
</tbody>
</table>

JCP Approval required for all Java SE features. Confirmed. **Unconfirmed.**
Java 9
JDK 9

- Released September 2017
- Last Major Release – 100+ features

More information on any JEP: http://openjdk.java.net/jeps/{JEP#}
JDK 9

- Released September 2017
- Last Major Release
  - 100+ features

More information on any JEP:
http://openjdk.java.net/jeps/{JEP#}
New world, new deployment option
Java on the Browser: 3 Way-conversation
Bring-your-own-Java: More control, less surprises

Java™ 8u141
JAR

Java™ 8u151
JAR

Java™ 9
JAR
Bring-your-own-Java: More control, less surprises

8u141

8u151

9
JEP 282: jlink: The Java Linker

tools / jlink

- Create a tool that can assemble and optimize a set of modules and their dependencies into a custom run-time image as defined in JEP 220. Define a plugin mechanism for transformation and optimization during the assembly process, and for the generation of alternative image formats

- Create a custom runtime optimized for a single program

- JEP 261 defines link time as an optional phase between the phases of compile time and run time. Link time requires a linking tool that will assemble and optimize a set of modules and their transitive dependencies to create a run-time image or executable
Project Jigsaw

Modularize the Java Platform

• JEP 261: Module System
• JEP 200: The Modular JDK
• JEP 201: Modular Source Code
• JEP 220: Modular Run-Time Images
• Plus
  – JEP 260: Encapsulate Most Internal APIs
  – JEP 282: jlink: The Java Linker
Java SE Modules

• Modularize your application
Java SE Modules
Java SE Modules
Java Custom Runtime

• Includes the Modular Application

Custom Image
~ 40 Mb
Using Jlink

8u141  8u151  9
Using Jlink

8u141

8u151

9
JDK 9 Jigsaw Security

Module boundaries enforced by the JVM

- Encapsulate implementation—internal classes inside modules
  - Share them with other implementation modules only as needed
- Massive maintainability improvement
- Simpler compatibility upgrade path
  - **We and You** can now hide and preclude access to unsupported internal APIs and implementation
- Will also significantly improve Security
  - Enable developers to create customized runtime that removed unused security sensitive APIs
Java 9 Screencasts Available

• Covering the most important features from JDK 9

https://www.oracle.com/java/java9-screencasts.html
Java 9 Screencasts Available

• Covering the most important features from JDK 9

https://www.oracle.com/java/java9-screencasts.html
Java 10
JDK 10 – Mar 2018

- First feature release
- 12 JEPs
  (Java Enhancement Proposals)
Java 11
JDK 11 – Sep 2018

• 4 JEPs targeted so far...
  – New model calls for JEPS to be targeted only when ready
OpenJDK EA releases

– Upcoming Feature Releases
– Special Projects EAs
Beyond Java 11
The Next Big Challenge: Object Data layout

- Java is very good at optimizing code, less so at optimizing data
- Java’s type system gives us primitives, objects, and arrays
- But flexibility is not exactly where we need it
- The big problem: object identity
- Project Valhalla – Value Types
Improved Java/Native Interoperability

- Big Data Hadoop and Spark are highly dependent on native libraries
- Meanwhile, Java has significant technical debts in support of foreign calls
- Project Panama - provide an easier, safer and faster JNI
- Project Loom – Lightweight thread and continuation
Summary

• The Java platform development on OpenJDK is becoming more open
  – Contributing all commercial features (zGC, JFR, AppCDS, etc)
  – GPL+CPE build

• The cloud is demanding a faster pace and continuous delivery
  – Uptake new Java releases every 6-months!

• Beyond 10, we have a solid technical roadmap

• Let’s continue to innovate and advance the Java SE Platform on OpenJDK together!

Join and become an OpenJDK contributor

https://openjdk.java.net
GREAT INDIAN DEVELOPER SUMMIT 2019
Conference: April 23-26, Bangalore

Register early and get the best discounts!

www.developersummit.com  @greatindiaandev  bit.ly/gidslinkedin