

The Future of Java & You

Heather VanCura
Director & Chair, JCP Program
Twitter @heathervc
July 2019



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.

Heather VanCura

- Chairperson & Director of the JCP Program
- International Speaker and Java Community Leader
- Open Source & Diversity Fan
- Californian - from San Diego
- Personal Interests: Travel, Fitness, Music, Fun
- Twitter @heathervc



Continuing Growth



#1
Programming
Language



12+
Million
Developers
Run Java



38 Billion
Active
Virtual Machines



21 Billion
Cloud Connected
Virtual Machines

Java Philosophies

- Platform Completeness
- Quality and Security
- Modernization and Innovation
- Open and Transparent Evolution
- Developer Productivity & Compatibility
- Active Ecosystem Involvement



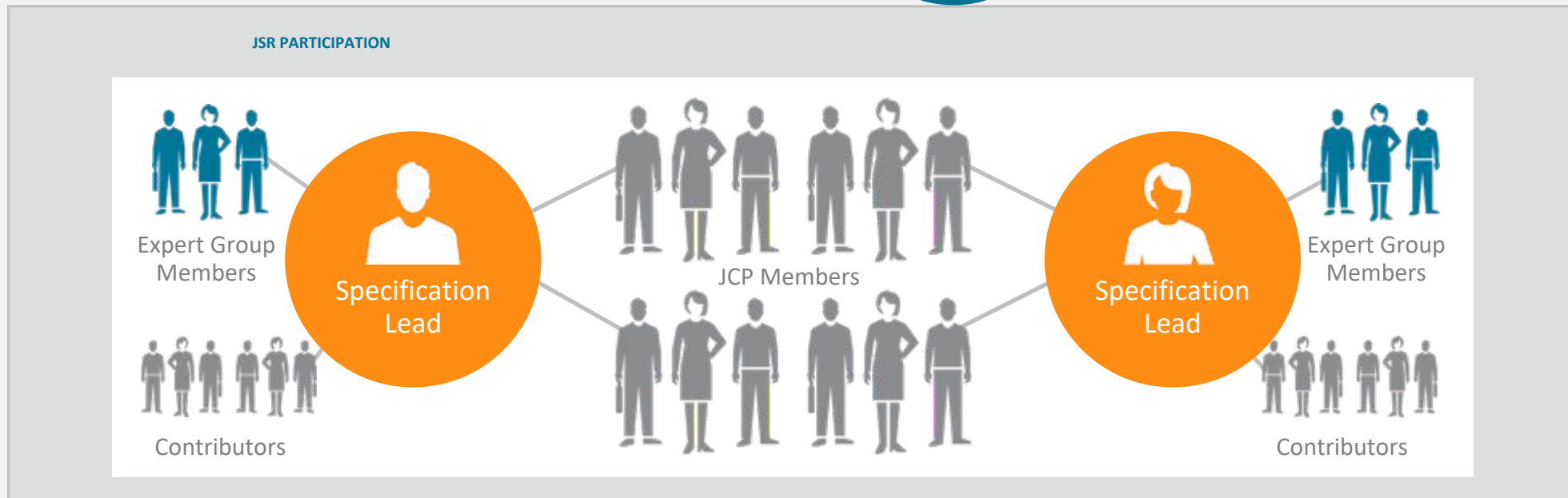
Celebrating 20 Years of the JCP Program!



Open Evolution

- 1995: Sun Microsystems develops Java.
- 1998: Sun opens up the development process to its competitors, creating the Java Community Process.
- 2006: Sun open-sources Java SE and Java EE.
- 2007: Sun goes into decline; Java stagnates.
- 2010: Oracle acquires Sun and becomes the steward of Java.
- 2011-2017: New Versions of Platforms Released.
- 2012—: Greater Engagement of Developer Community & User Groups.
- 2017—: Introduce Faster OpenJDK Release Cycle; Java EE migration to Eclipse.

Organization



Membership Quotes

- “Joining the JCP, especially being a member of several expert groups, has had a big impact on my career. Mostly by making it visible for the leadership and managers in my company that I am actually a part of forming the future of the platform. I also think, or hope, that it is an inspiration for my colleagues.”
- Ivar Grimstad
- “Joining the JCP is like being a Java citizen.”- Heinz Kabutz

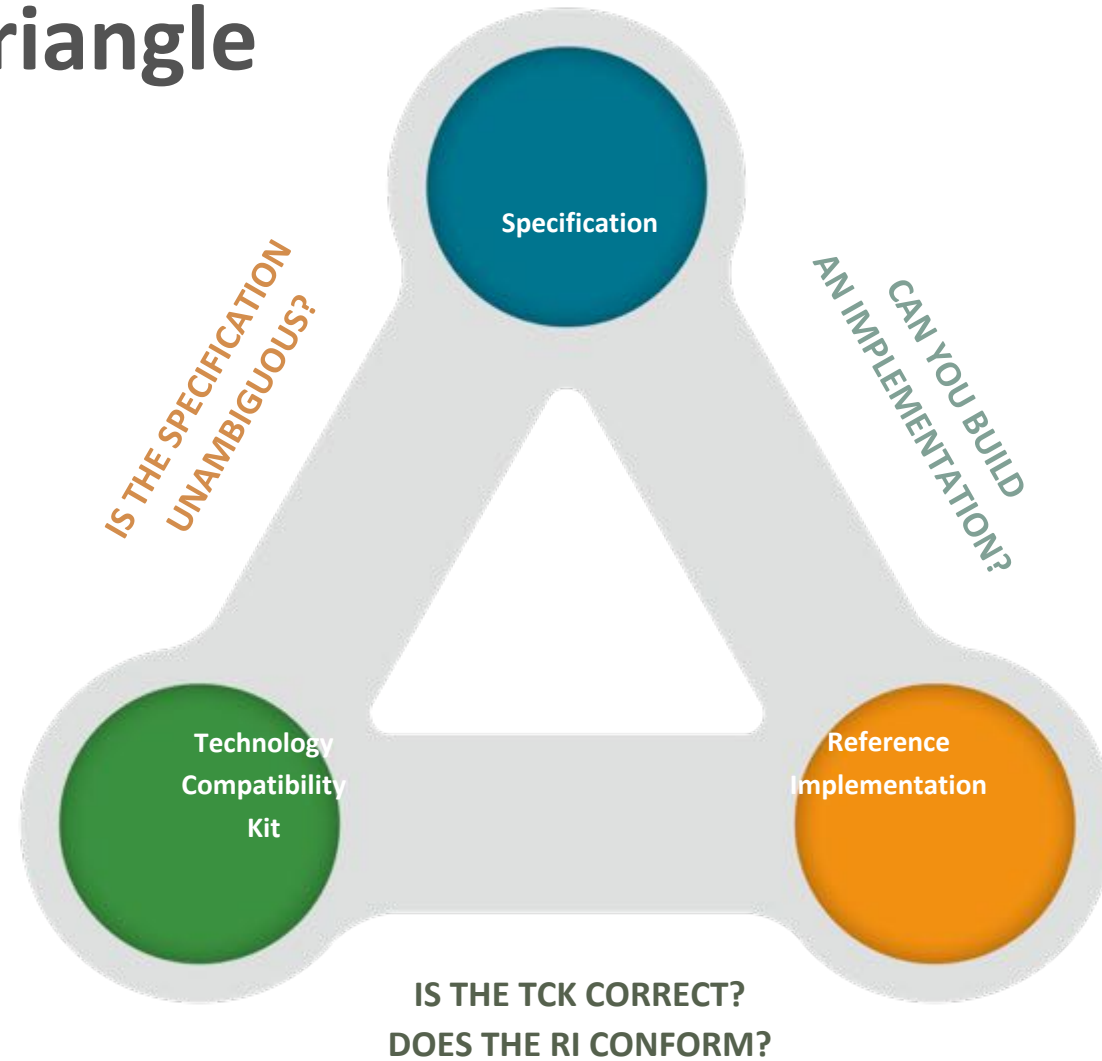
Collaborative Development - How does it work?

- Java Specification Requests (JSRs)
 - A JSR is a single version of a Java specification.
- JSRs are led by a community member (the Spec Lead), with a group of interested members (the Expert Group) helping with the day-to-day decisions and work.
 - Any JCP member can submit and lead a JSR.
- Each Expert Group must deliver:
 - The Specification
 - A Reference Implementation (RI)
 - A Technology Compatibility Kit (TCK)

The JSR Development Cycle

- Every project developed through the JCP follows the JSR lifecycle
- It Includes formal public reviews and votes by the Executive Committee.
- Full Members can submit & lead JSRs, serve as Expert Group members on JSRs.
- Associate Members can participate as Contributors on JSRs.

Compatibility Triangle



An International Effort



Who Are The Members?

- Corporations
- Non-Profit & Open Source
- Java User Groups (JUG)
- Individual Developers



The Executive Committee

- Corporations
- Non-Profits/Open Source
- Java User Groups
- Individual Developers



How: Open Standards & Open Source

- We need both!



open source

Complementary

- Open source important requirement. So are Standards.
- Coop-etition. Agree on what to standardize (cooperation) & what to compete on (implementations).
- Open standards implemented in open source = easier to implement standard & for developers to understand technology.
- Results in more implementations and greater adoption of the standard.
- An effective way to develop a standard-start with an open source project that has demonstrated the need for standardization.

Organization Focus



- New revisions of platform.
- “JCP.Next” reforms
- Increased participation from Community.

Open Source Implementations

- The Reference Implementations of the Java platform are developed collaboratively and released under open-source license
- Java SE: OpenJDK

OpenJDK

Java SE Platform



**Last Major
Release**

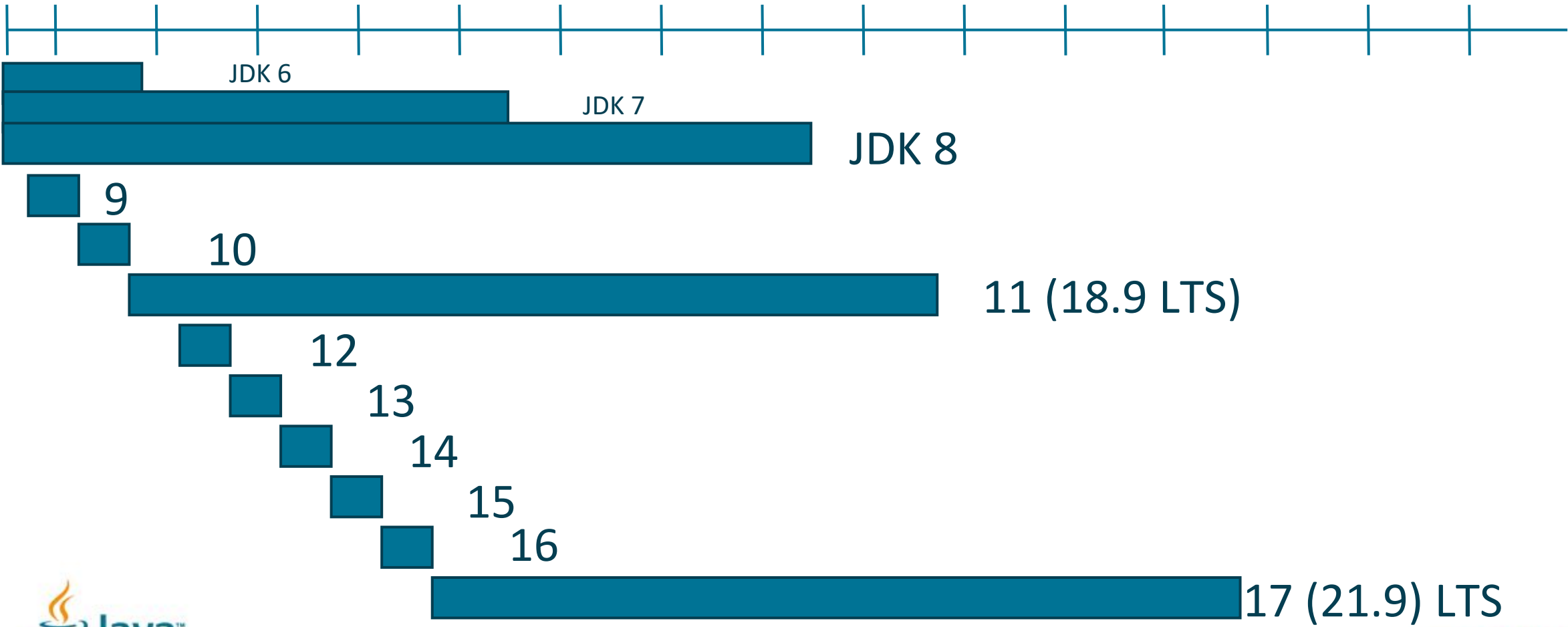


Modularity

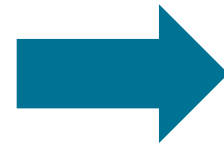


Cadence

New JDK Release Model – Starting with JDK 9, LTS every 3 yrs



From Oracle JDK to OpenJDK from Oracle



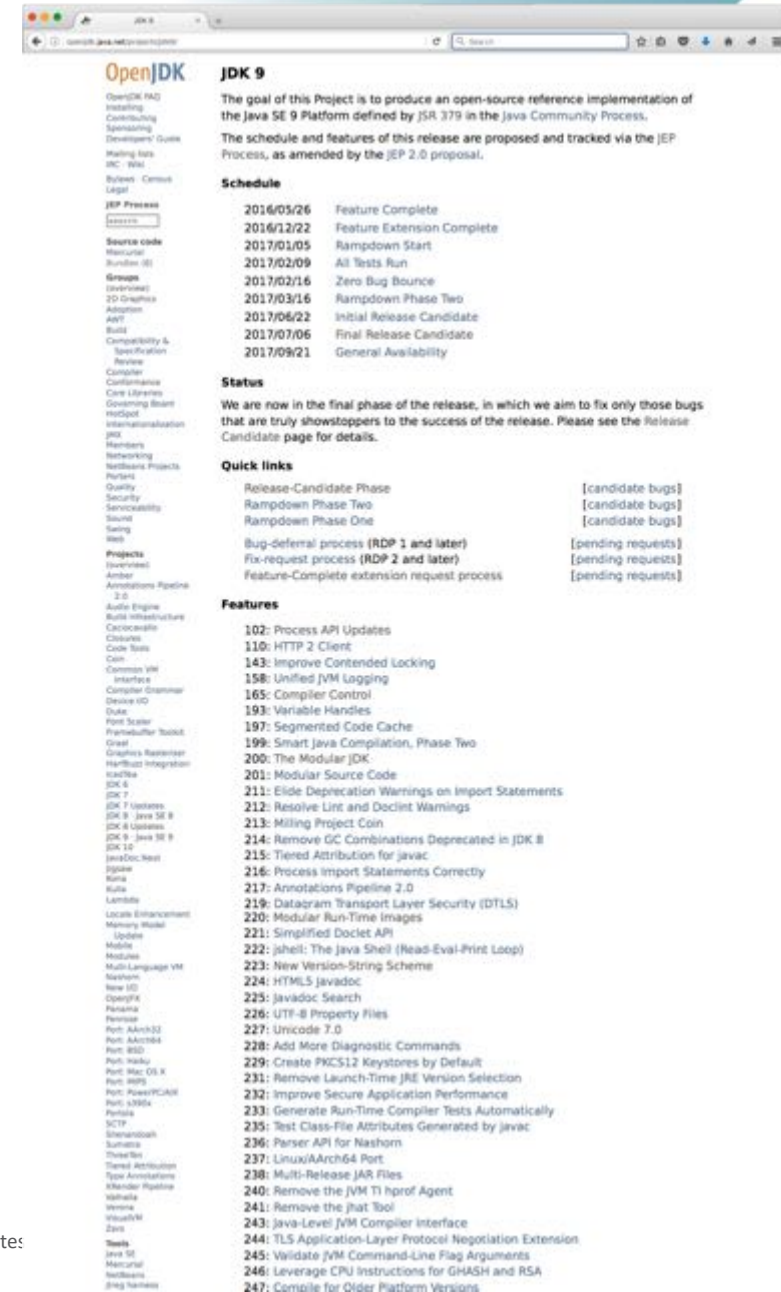
OpenJDK

Features Open-Sourced in Java

- **Application Class Data Sharing (Now available in OpenJDK 10)**
 - Enables you to place classes from the standard extensions directories and the application class path in the shared archive
- **Project ZGC (Now available in OpenJDK 11)**
 - Low latency garbage collector to support multi-terabyte heaps
- **Flight Recorder (Now available in OpenJDK 11)**
 - Collects diagnostic and profiling data about a running Java application
- **Mission Control (Now available in OpenJDK11)**
 - Monitor and manage Java applications with minimal performance overhead
- **Usage Logger (Now available in OpenJDK11)**
 - Logs how the JRE's are being used in your systems

JDK 9

- Released September 2017
- #WorksFineOnJDK9
- Last Major Release
– 100+ features



OpenJDK

The goal of this Project is to produce an open-source reference implementation of the Java SE 9 Platform defined by JSR 339 in the Java Community Process. The schedule and features of this release are proposed and tracked via the JEP Process, as amended by the JEP 2.0 proposal.

Schedule

2016/05/26	Feature Complete
2016/12/22	Feature Extension Complete
2017/01/05	Rampdown Start
2017/02/09	All Tests Run
2017/02/16	Zero Bug Bounce
2017/03/16	Rampdown Phase Two
2017/06/22	Initial Release Candidate
2017/07/06	Final Release Candidate
2017/09/21	General Availability

Status

We are now in the final phase of the release, in which we aim to fix only those bugs that are truly showstoppers to the success of the release. Please see the Release Candidate page for details.

Quick links

Release-Candidate Phase	[candidate bugs]
Rampdown Phase Two	[candidate bugs]
Rampdown Phase One	[candidate bugs]
Bug-deferral process (RDP 1 and later)	[pending requests]
Fix-request process (RDP 2 and later)	[pending requests]
Feature-Complete extension request process	[pending requests]

Features

- 102: Process API Updates
- 110: HTTP 2 Client
- 143: Improve Contended Locking
- 158: Unified JVM Logging
- 165: Compiler Control
- 193: Variable Handles
- 197: Segmented Code Cache
- 199: Smart Java Compilation, Phase Two
- 200: The Modular JDK
- 201: Modular Source Code
- 211: Elide Deprecation Warnings on Import Statements
- 212: Resolve Lint and Doctlet Warnings
- 213: Milling Project Coin
- 214: Remove GC Combinations Depreciated in JDK 8
- 215: Tiered Attribution for javac
- 216: Process Import Statements Correctly
- 217: Annotations Pipeline 2.0
- 219: Datagram Transport Layer Security (DTLS)
- 220: Modular Run-Time Images
- 221: Simplified Doctlet API
- 222: jshell: The Java Shell (Read-Eval-Print Loop)
- 223: New Version-String Scheme
- 224: HTML5 Javadoc
- 225: Javadoc Search
- 226: UTF-8 Property Files
- 227: Unicode 7.0
- 228: Add More Diagnostic Commands
- 229: Create PKCS12 Keystores by Default
- 231: Remove Launch-Time JRE Version Selection
- 232: Improve Secure Application Performance
- 233: Generate Run-Time Compiler Tests Automatically
- 235: Test Class-File Attributes Generated by javac
- 236: Parser API for Nashorn
- 237: Linux/AArch64 Port
- 238: Multi-Release JAR Files
- 240: Remove the JVM TI hprof Agent
- 241: Remove the jhat Tool
- 243: Java-Level JVM Compiler Interface
- 244: TLS Application-Layer Protocol Negotiation Extension
- 245: Validate JVM Command-Line Flag Arguments
- 246: Leverage CPU Instructions for GHASH and RSA
- 247: Compile for Older Platform Versions



JDK 10

- Released March 2018
- First feature release
- 12 JEPs
(Java Enhancement Proposals)

OpenJDK

JDK 10

This release will be the Reference Implementation of the next version of the Java SE Platform, as specified by JSR 383 in the Java Community Process.

Status

JDK 10 is in the [Release-Candidate Phase](#), in which we're fixing only those bugs that are absolutely critical to the success of the release. Please see the [Release-Candidate Phase](#) page for process details.

Schedule

2017/12/14	Rampdown Phase One
2018/01/11	All Tests Run
2018/01/18	Rampdown Phase Two
2018/02/08	Initial Release Candidate
2018/02/22	Final Release Candidate
2018/03/20	General Availability

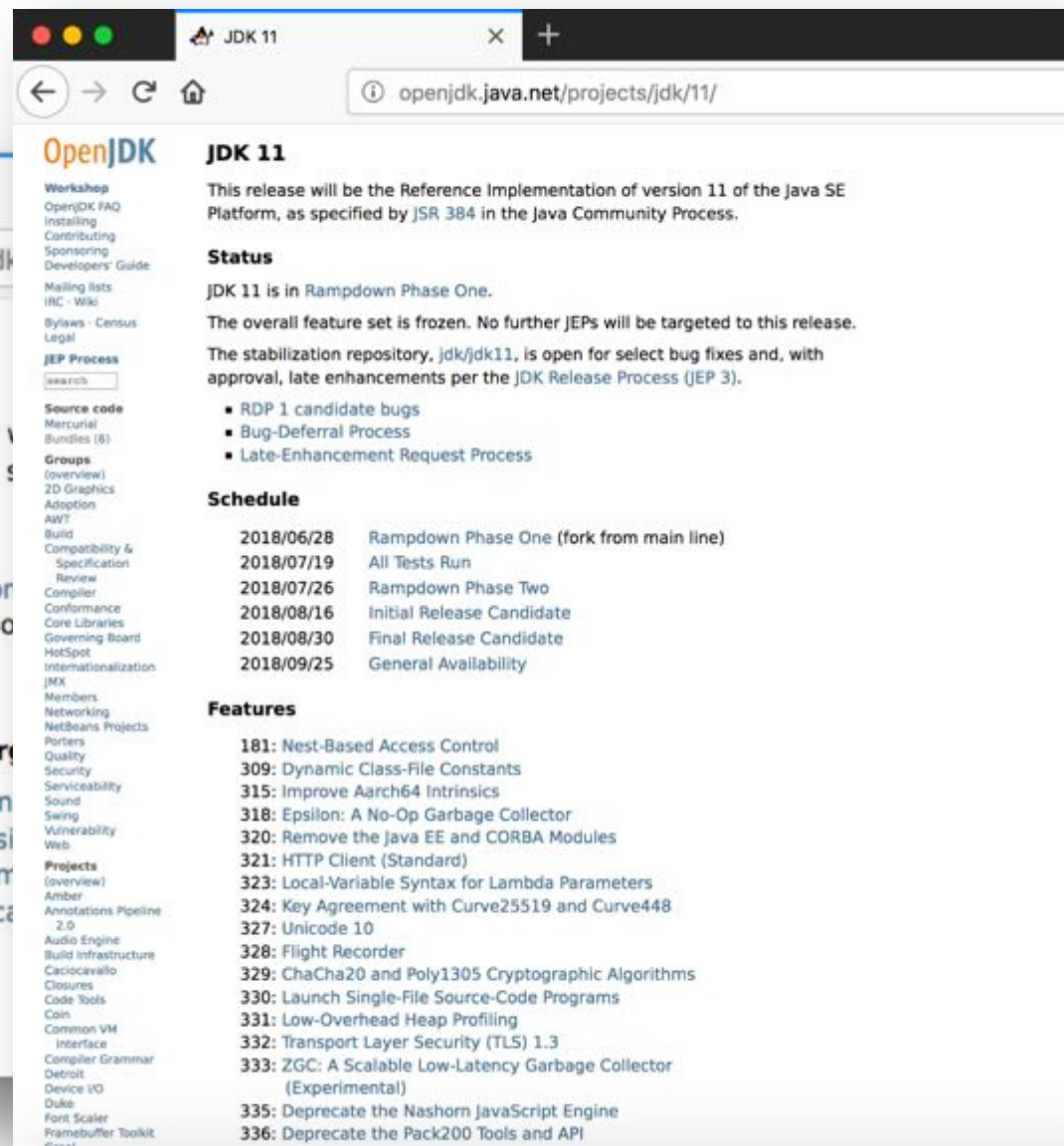
Features

- 286: Local-Variable Type Inference
- 296: Consolidate the JDK Forest into a Single Repository
- 304: Garbage-Collector Interface
- 307: Parallel Full GC for G1
- 310: Application Class-Data Sharing
- 312: Thread-Local Handshakes
- 313: Remove the Native-Header Generation Tool (javah)
- 314: Additional Unicode Language-Tag Extensions
- 316: Heap Allocation on Alternative Memory Devices
- 317: Experimental Java-Based JIT Compiler
- 319: Root Certificates
- 322: Time-Based Release Versioning

Last update: 2018/2/9 18:21 UTC

JDK 11 – Sep 2018

- 17 JEPs
- 4 JEPs targeted initially
- #WorksLikeHeavenonJDK11
- <http://jdk.java.net/11/>



JDK 12 – JSR 386

- March 2019 – available today
- 8 JEPs
- New model calls for JEPs to be targeted only when ready
- <http://openjdk.java.net/projects/jdk/12/>

JDK 12

This release will be the Reference Implementation of version 12 of the Java SE Platform, as specified by JSR 386 in the Java Community Process.

Status

JDK 12 is in the [Release-Candidate Phase](#).

The overall feature set is frozen. No further JEPs will be targeted to this release.

The stabilization repository, `jdk/jdk12`, is open only for P1 bug fixes, with approval, per the [JDK Release Process \(JEP 3\)](#).

- [Release-Candidate bugs](#)
- [Bug-Deferral Process](#)

Schedule

2018/12/13	Rampdown Phase One (fork from main line)
2019/01/17	Rampdown Phase Two
2019/02/07	Release-Candidate Phase
2019/03/19	General Availability

Features

- 189: [Shenandoah: A Low-Pause-Time Garbage Collector \(Experimental\)](#)
- 230: [Microbenchmark Suite](#)
- 325: [Switch Expressions \(Preview\)](#)
- 334: [JVM Constants API](#)
- 340: [One AArch64 Port, Not Two](#)
- 341: [Default CDS Archives](#)
- 344: [Abortable Mixed Collections for G1](#)
- 346: [Promptly Return Unused Committed Memory from G1](#)

JDK 13 – JSR 388

- Planned Release
September 2019
- 5 JEPS
- In ramp down
phase
- Early Access
Builds are out

JDK 13

This release will be the Reference Implementation of version 13 of the Java SE Platform, as specified by JSR 388 in the Java Community Process.

Status

JDK 13 is in [Rampdown Phase One](#).

The overall feature set is frozen. No further JEPs will be targeted to this release.

The stabilization repository, [jdk/jdk13](#), is open for select bug fixes and, with approval, late enhancements per the [JDK Release Process \(JEP 3\)](#).

- [RDP 1 candidate bugs](#)
- [Bug-Deferral Process](#)
- [Late-Enhancement Request Process](#)

Schedule

2019/06/13	Rampdown Phase One (fork from main line)
2019/07/18	Rampdown Phase Two
2019/08/08	Initial Release Candidate
2019/08/22	Final Release Candidate
2019/09/17	General Availability

Features

- 350: [Dynamic CDS Archives](#)
- 351: [ZGC: Uncommit Unused Memory](#)
- 353: [Reimplement the Legacy Socket API](#)
- 354: [Switch Expressions \(Preview\)](#)
- 355: [Text Blocks \(Preview\)](#)

JDK 14 – JSR 389

- JSR filed and posted on jcp.org
- Planned Release March 2020

JDK 14

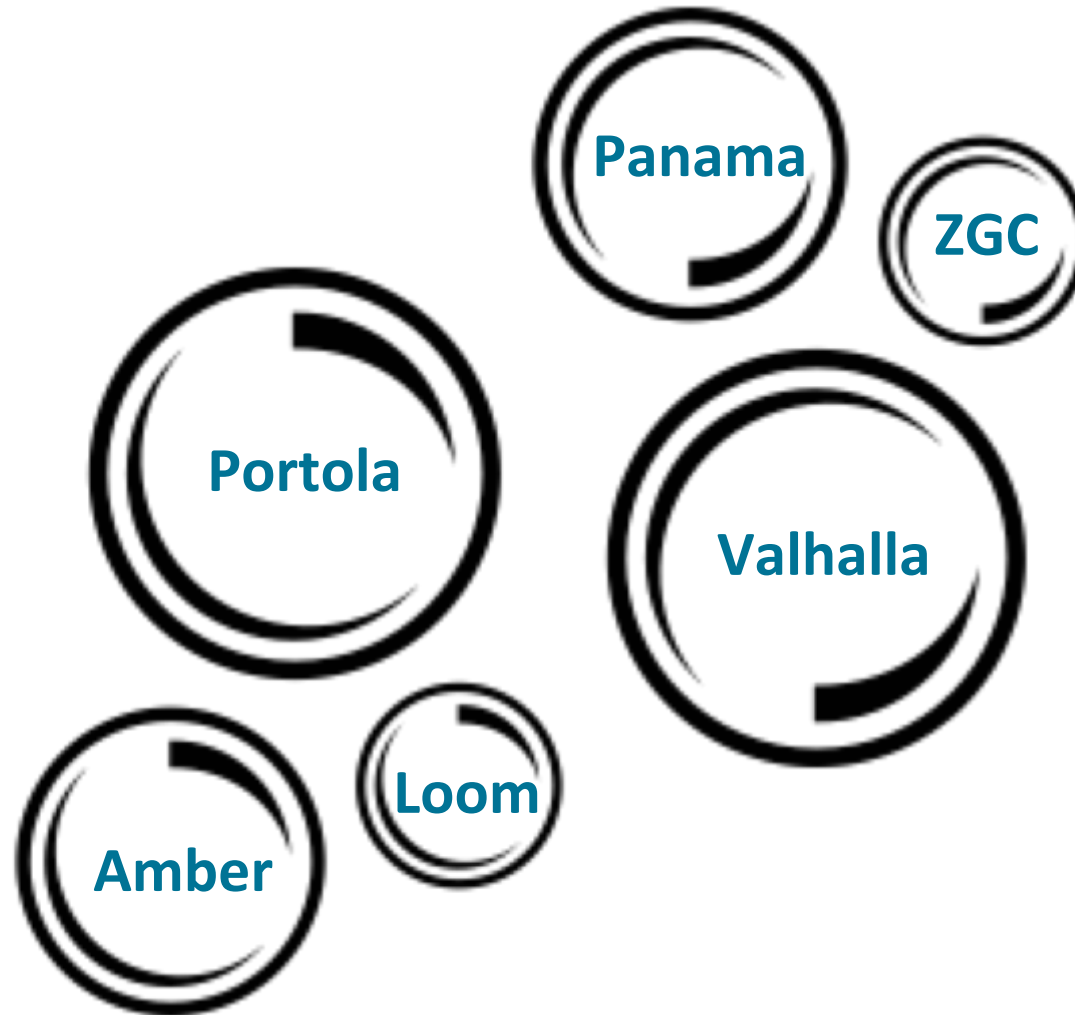
This release will be the Reference Implementation of version 14 of the Java SE Platform, as specified by [JSR 389](#) in the Java Community Process.

Status

The development repositories are open for bug fixes, small enhancements, and JEPs as proposed and tracked via the JEP Process.

Last update: 2019/6/18 20:55 UTC

And Beyond



Java SE Platform Investments

- Security is **#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)

Project Valhalla : 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

Project Portola

- Java in a world of Containers
- Java's characteristics make it ideal for container deployment
 - Safe & Secure, Performant, Reliable, Rich Eco System
- We are committed to having Java remain the first choice for deployments in the cloud

Project ZGC : A Scalable Low-Latency Garbage Collector

- Open sourced earlier this year
- <http://wiki.openjdk.java.net/display/zgc/>
- Early Access builds available: <http://jdk.java.net/zgc/>
 - Enable ZGC: **-XX:+UseZGC**

Project Panama

- Foreign Functions & Data
- Improved Java/Native Interoperability
- Simple, safe, and performant replacement for JNI
- Access to low-level hardware functionality through normal Java code
 - Vector instructions, special memory types (for example non-volatile memory)
- Big Data, Machine Learning

Project Loom

- Easier and more scalable concurrency model
 - Making blocking calls virtually free
- “Fibers” (lightweight threads) and continuations
 - Millions of fibers can be spawned in a single JVM instance

Project Amber

- Language improvements
 - Dynamic Class-File Constants
 - Raw string literals
 - Switch expressions
 - Pattern matching

Download Early Access for Panama and Valhalla

jdk.java.net

Java Development Kit builds, from Oracle

Ready for use: JDK 11

Early access: JDK 13, JDK 12, Jpackage, OpenJFX, Panama, Valhalla, & JMC

Reference implementations: Java SE 12, Java SE 11, 10, 9, 8, & 7

Get Involved

Follow on Twitter
@OpenJDK

OpenJDK

Join and become an OpenJDK contributor
<https://openjdk.java.net>

Updated Java Training & Certifications

https://education.oracle.com/learn/java/pPillar_80

JAVA SE 11 Oracle Training and Certification

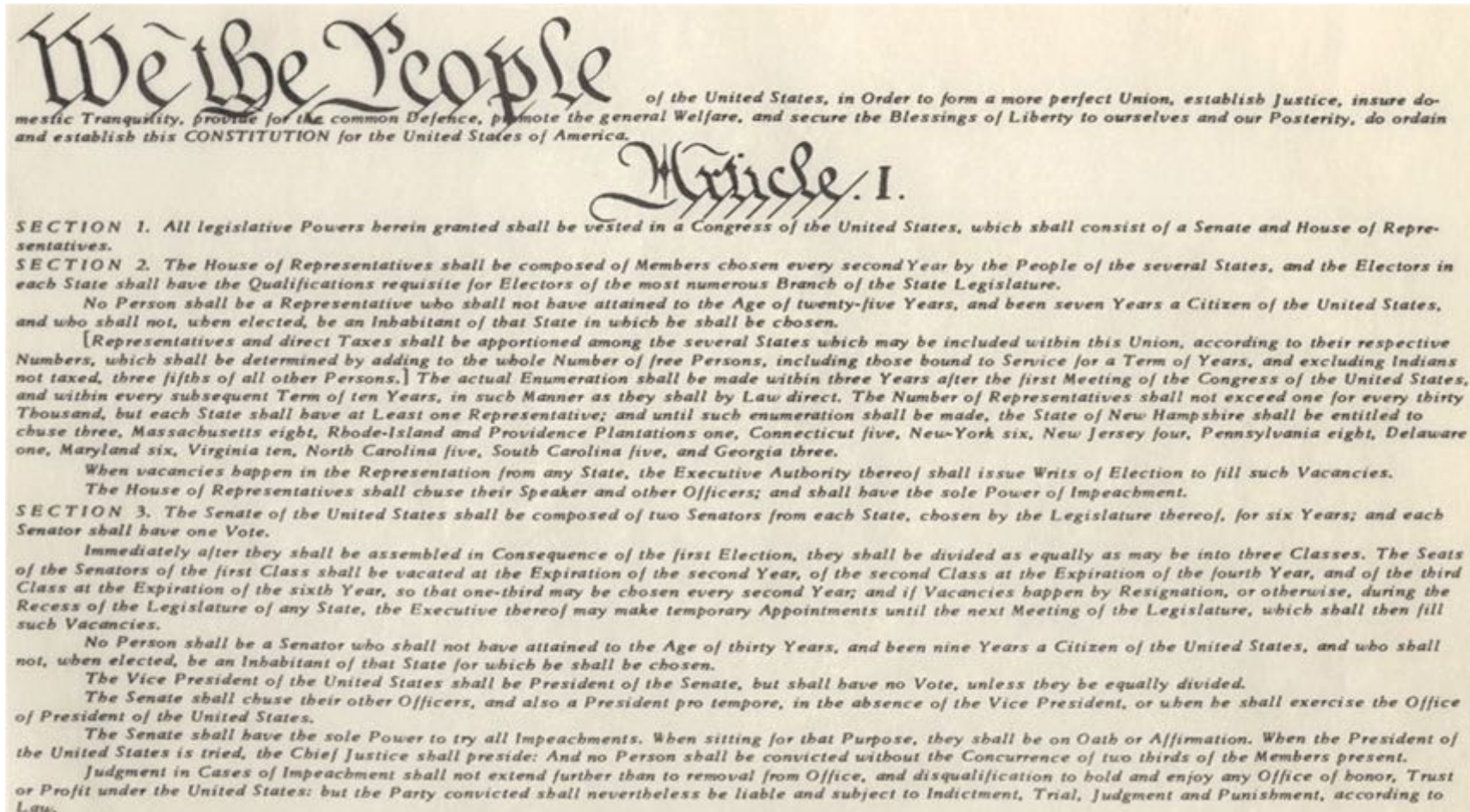
I'm a student Take this training	I'm a Java Oracle Certified Professional (OCP) Take this training	I'm a Java Oracle Certified Associate (OCA) Take this training	I'm new to Java Take this training	My Java Certification is on an older version Take this training
Choose one course to get started	Java SE 6, 7 OCPs	Java SE 6, 7 or 8 OCAs	Get Started	For any version before SE 6 Java OCA or OCPs
Java Foundations - Oracle Academy course Java SE: Programming I	Java SE: Programming II Java 11: New Features (free video)	Java SE: Programming II Java 11: New Features (free video)	Java SE: Programming I Java SE: Programming II	
	Java SE 8 OCPs			
	Java SE: Exploiting Modularity and Other New Features Java 11: New Features (free video)			
Take this exam	Take this exam	Take this exam	Take this exam	
Java Foundations 1Z0-811	Upgrade OCP Java 6, 7 & 8 to Java SE 11 Developer 1Z0-817	Java SE 11 Programmer II 1Z0-816	Java SE 11 Programmer I 1Z0-815 Java SE 11 Programmer II 1Z0-816	

ORACLE Certified Junior Associate

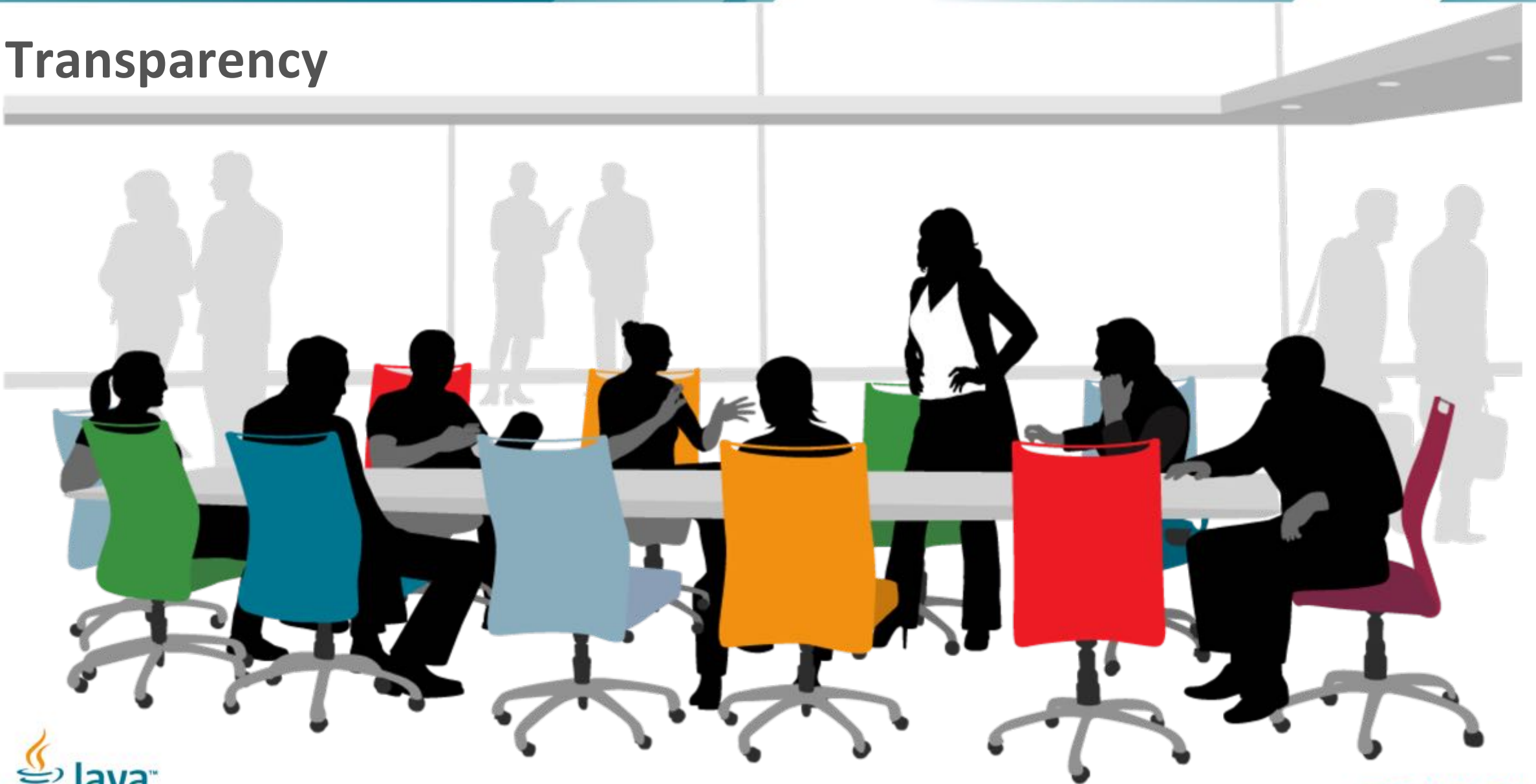
2019 ORACLE Certified Professional

ORACLE University

JCP.next: Changing the Constitution



Transparency

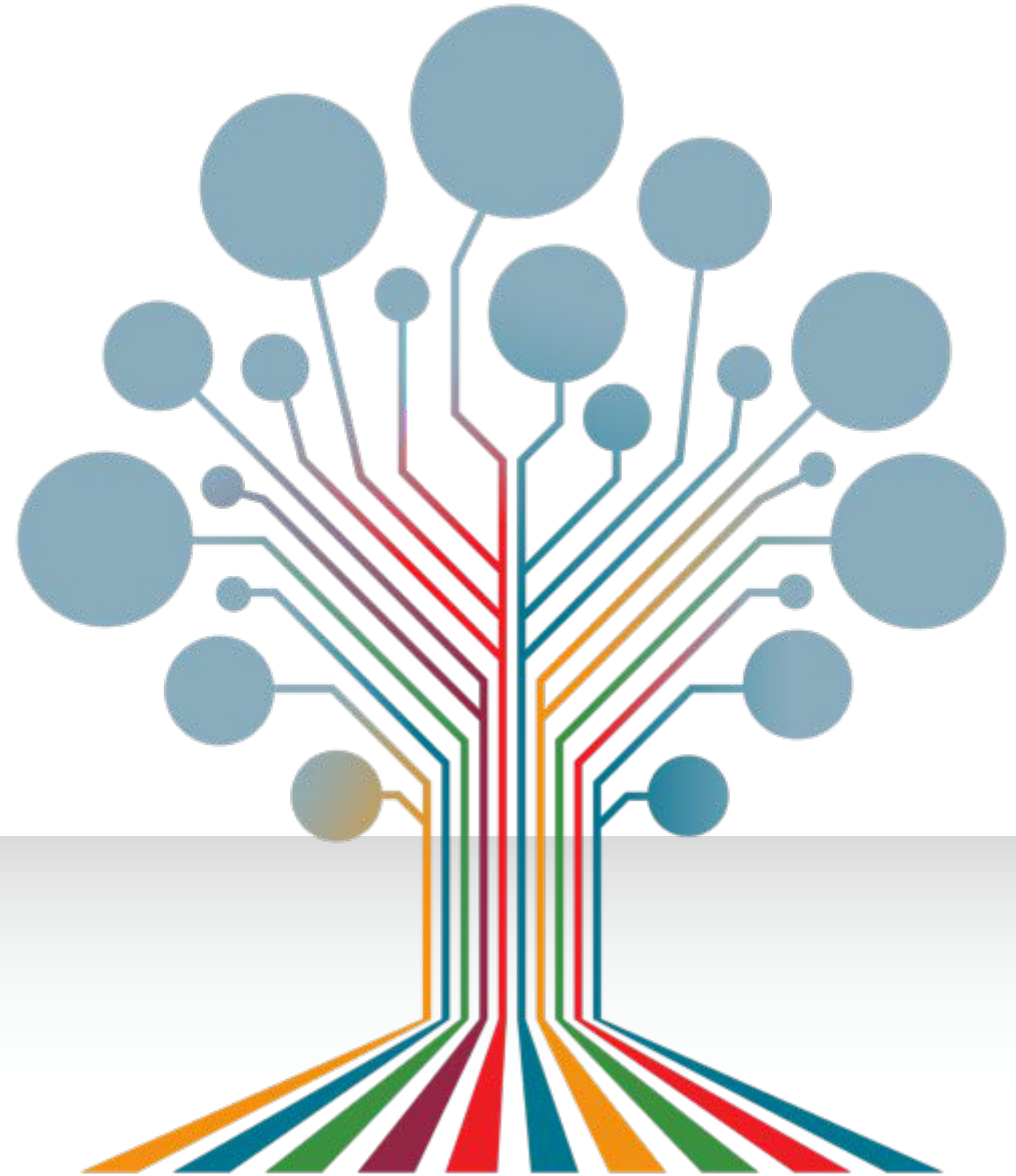


JCP Executive Committee (EC)

- One Java, One EC - merged ECs
- Resize the EC - 2019



Participation Move Faster



Broadening JCP Membership

- Eliminate Barriers to participation.
- Introduce new Membership levels.
- No Membership Fees.
- Electronic Signatures.
- Add Contributors for JSR Expert Groups.
- Add Associate Seats on Executive Committee.

Membership Levels

- Associate Members are individuals who can be listed as contributors to JSRs and vote for the Executive Committee.
- Partner Members are Java User Groups and other non-profit organizations that can serve on and vote for the Executive Committee.
- Full Members can serve on Expert Groups, lead JSRs, serve on and vote for the Executive Committee

The Streamline the JSR Development Cycle

Includes formal public reviews and votes by the Executive Committee.

See the Process Document for the details.



Evolution Continues...What's Next?

- JSR 387, Streamlining the JCP Program
 - JCP.Next Working Group
 - Final Release December 2018
- Follow the EC Summaries and discussion:
 - https://jcp.org/en/resources/EC_summaries

More Open Than Ever Before



How will you Participate?

- As an Individual – OK
- As part of a team – better
- Work through JUG or employer
- Help each other
- Teach other
- Work with each other



Working Together – We Achieve More



JUG Members & Adopt-a-JSR Global Adoption Efforts

Abdijan JUG (Ivory Coast)	Detroit JUG (USA)	Japan JUG	JUG-EG (Egypt)	Madras JUG (India)
Alpes JUG (France)	Duchess (Women)	Java Hellenic User Group (Greece)	JUG Frankfurt [JUGF] (Germany)	Nashville JUG (United States)
Austin JUG (USA)	ESPRIT Tunisian JUG (Tunisia)	Java Student User Group (Vienna)	JUG Indonesia	Oklahoma City JUG (USA)
BeJUG (Belgium)	FASOJUG (Burkina Faso)	Java Web User Group (London)	JUG JogLoSemar (Indonesia)	Philly JUG (USA)
BreizhJUG (Brittany)	Guatemala Java User Group	Jozi JUG (South Africa)	JUG-MK (Macedonia)	Rio JUG (Brazil)
CEJUG (Brazil)	Green Tea JUG (China)	JUG-AFRICA	JUG-RU (Russia)	Riviera JUG (France)
Central Ohio JUG (USA)	Houston JUG (USA)	JUG Chennai (India)	JUG-USA	Ruhrjug (Germany)
Chicago JUG	iJUG e.V. (Germany)	JUG-Cologne (Germany)	Lagos JUG (Nigeria)	Silicon Valley JavaFX User Group
ChinaNanjingJUG	IndiJava (India)	JUG Dortmund (Germany)	London Java Community (UK)	SeneJUG (Senegal)
Connecticut JUG (USA)	ITP_JAVA (Peru)		Malaysia-JUG	SouJava (Brazil)
	Istanbul JUG		MoroccoJUG	Utah JUG (USA)
				Vancouver Island JUG (Canada)

JUGs around the World - Driving Adoption

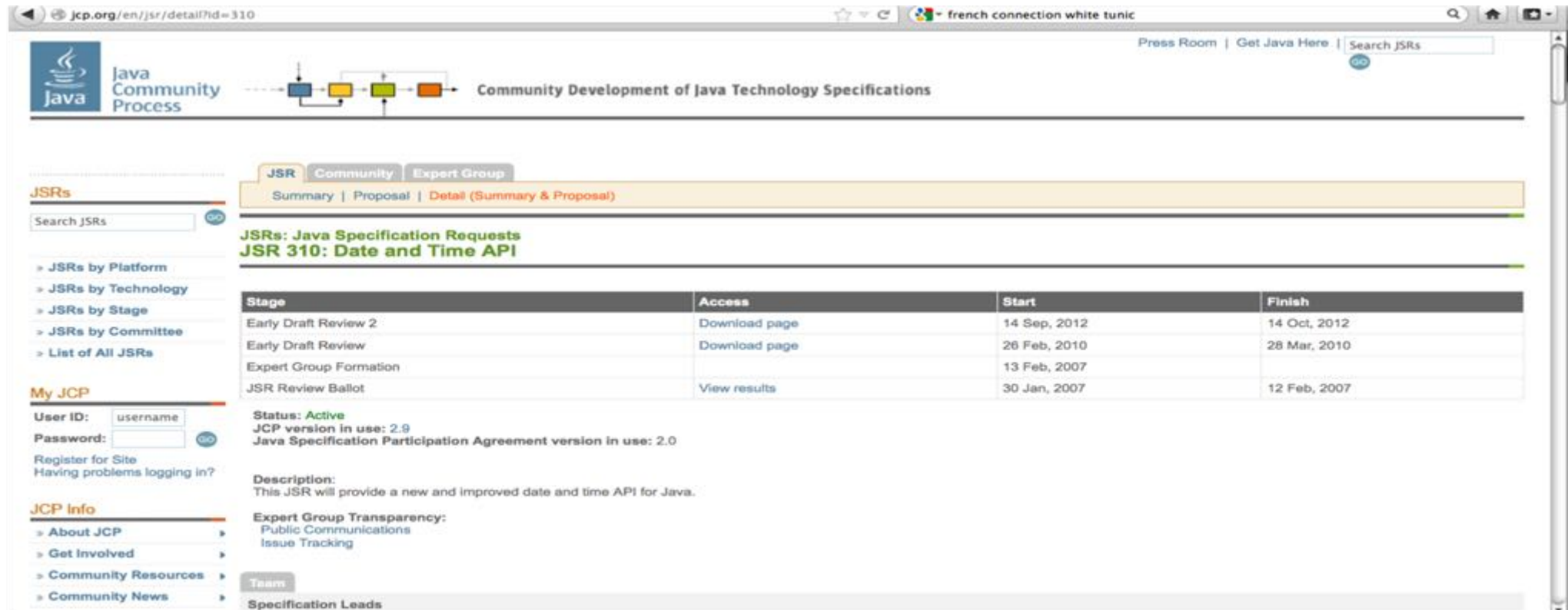


1) Pick JSR - New JSRs

- Recently Submitted:
 - MVC 1.0 (JSR 371)
 - Visual Rec (JSR 381)
 - Desktop Application API (JSR 377)
 - Portlet 3.0 Bridge for JSF 2.2 (JSR 378)
 - Units of Measurement 2.0 (JSR 385)
 - Java SE 13 (JSR 388)

<https://jcp.org/en/jsr/stage?listBy=active>

JSR example page



The screenshot shows the JSR 310 page on the Java Community Process website. The page title is "JSR 310: Date and Time API". The page is divided into several sections: a navigation bar with "JSR", "Community", and "Expert Group" tabs; a search bar for JSRs; a sidebar with navigation links; a main content area with a table of stages and a description; and a footer with "Team" and "Specification Leads" links.

JSRs: Java Specification Requests
JSR 310: Date and Time API

Stage	Access	Start	Finish
Early Draft Review 2	Download page	14 Sep, 2012	14 Oct, 2012
Early Draft Review	Download page	26 Feb, 2010	28 Mar, 2010
Expert Group Formation		13 Feb, 2007	
JSR Review Ballot	View results	30 Jan, 2007	12 Feb, 2007

Status: Active
JCP version in use: 2.9
Java Specification Participation Agreement version in use: 2.0

Description:
This JSR will provide a new and improved date and time API for Java.

Expert Group Transparency:
Public Communications
Issue Tracking

Team
Specification Leads

List of Active JSRs (posting in last 12 months):

<http://jcp.org/en/jsr/all?status=Active&activeMonths=12>

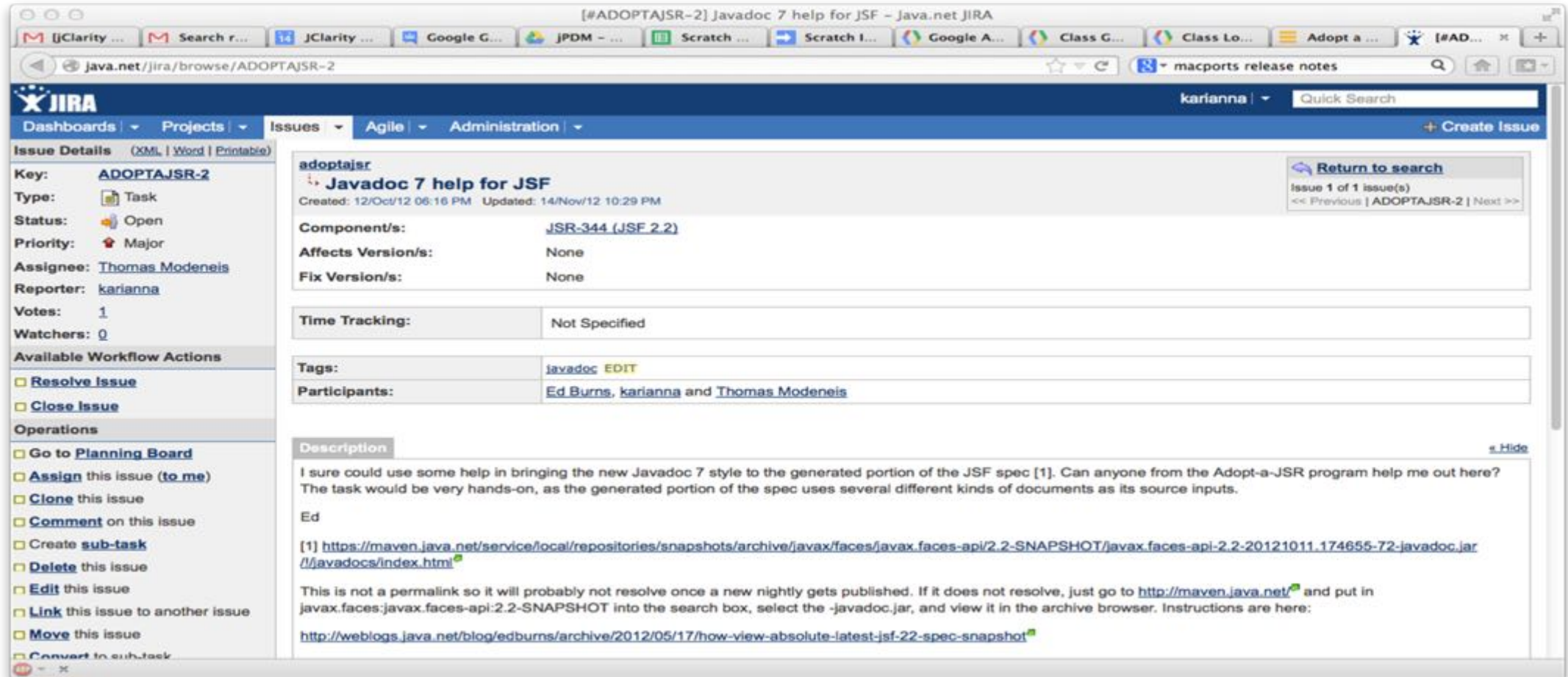
2) Communication: Two Way Street



3) Decide on Steps: Take Action

- Share ideas and feedback, comment on list and public issue trackers.
- Read early versions and share feedback on specifications and Javadocs.
- Download and provide feedback on early access reference implementation.
- Try writing sample applications using early builds of reference implementation.
- Write or speak about the technology and encourage others to participate. Translate into your native language.
- Evangelize the JSR -social media, blogging or lightning talks.
- Help with documentation.

4) Follow Public Discussions & Comment - Issue Tracker



The screenshot shows a JIRA issue page for the project 'ADOPTAJSR-2'. The issue title is 'Javadoc 7 help for JSF', created on 12/Oct/12 06:16 PM and updated on 14/Nov/12 10:29 PM. The issue is a 'Task' with a status of 'Open' and a priority of 'Major'. The assignee is 'Thomas Modeneis' and the reporter is 'karianna'. The issue has 1 vote and 0 watchers. The component is 'JSR-344 (JSF 2.2)'. The issue is tagged with 'javadoc' and has participants 'Ed Burns, karianna and Thomas Modeneis'. The description asks for help in bringing the new Javadoc 7 style to the generated portion of the JSF spec, providing a link to a Maven repository snapshot and instructions on how to view it.

Issue Details (XML | Word | Printable)

Key: ADOPTAJSR-2

Type: Task

Status: Open

Priority: Major

Assignee: Thomas Modeneis

Reporter: karianna

Votes: 1

Watchers: 0

Available Workflow Actions

- Resolve Issue
- Close Issue

Operations

- Go to Planning Board
- Assign this issue (to me)
- Clone this issue
- Comment on this issue
- Create sub-task
- Delete this issue
- Edit this issue
- Link this issue to another issue
- Move this issue
- Convert to sub-task

adoptajsr

Javadoc 7 help for JSF

Created: 12/Oct/12 06:16 PM Updated: 14/Nov/12 10:29 PM

Component/s: JSR-344 (JSF 2.2)

Affects Version/s: None

Fix Version/s: None

Time Tracking: Not Specified

Tags: javadoc [EDIT](#)

Participants: Ed Burns, karianna and Thomas Modeneis

Description [Hide](#)

I sure could use some help in bringing the new Javadoc 7 style to the generated portion of the JSF spec [1]. Can anyone from the Adopt-a-JSR program help me out here? The task would be very hands-on, as the generated portion of the spec uses several different kinds of documents as its source inputs.

Ed

[1] <https://maven.java.net/service/local/repositories/snapshots/archive/javax/faces/javax.faces-api/2.2-SNAPSHOT/javax.faces-api-2.2-20121011.174655-72-javadoc.jar!/javadocs/index.html>

This is not a permalink so it will probably not resolve once a new nightly gets published. If it does not resolve, just go to <http://maven.java.net/> and put in javax.faces:javax.faces-api:2.2-SNAPSHOT into the search box, select the -javadoc.jar, and view it in the archive browser. Instructions are here: <http://weblogs.java.net/blog/edburns/archive/2012/05/17/how-view-absolute-latest-jsf-22-spec-snapshot>

5) Participate in Hack Days & Workshops – Have Fun!





Early Access

- In adoption group, you can discuss Early Access Builds – provide comments

jdk.java.net

GA Releases

JDK 11

Early-Access Releases

JDK 13

JDK 12

Jpackage

OpenJFX

Panama

Valhalla

JMC

Reference Implementations

Java SE 12

Java SE 11

Java SE 10

Java SE 9

Java SE 8

Java SE 7

Feedback

Report a bug

Archive

JDK 13 Early-Access Builds

Schedule, status, & features ([OpenJDK](#))

Documentation

- Release notes
- Test results
- API Javadoc

Latest build: 11 (2019/3/7)

- Changes in this build
- Issues addressed in this build

Builds

These early-access, open-source builds are provided under the GNU General Public License, version 2, with the Classpath Exception.

Linux/x64	tar.gz (sha256)	194685134 bytes
macOS/x64	tar.gz (sha256)	187112270
Windows/x64	zip (sha256)	193205088
Alpine Linux/x64	tar.gz (sha256)	197120161

Participate in OpenJDK

- Download Early Access Builds (13 available now)
- Adoption Group & Quality Outreach
- Join mail list prior to sending feedback:
<http://mail.openjdk.java.net/mailman/listinfo/adoption-discuss>
- <https://wiki.openjdk.java.net/display/quality/Quality+Outreach>

OpenJDK

Quality Outreach part of Adoption Group

- Test FOSS projects on EA OpenJDK Builds
- Open Source Projects - Quality Outreach examples
 - Eclipse Collections - Nikhil Nanivadekar
 - Apache Maven - Robert Scholte



We need you!

- Follow the JCP online: <http://JCP.org>
 - Join the JCP: <https://jcp.org/en/participation/membership>
 - Twitter @jcp_org #JCP, @heathervc
 - blogs.oracle.com/jcp
 - Facebook: Java Community Process

Oracle Code One

- JavaOne expansion and renaming - more Tracks, Languages and Communities - adding Go, Rust, Python, JavaScript, and R in addition to Java – join us!
- September 16-19 2019 San Francisco CA USA
- <https://www.oracle.com/code-one/index.html>



thank you, ευχαριστώ, shukriya, dziekuje , நன்றி,
tak, tack, grazie, gracias, danke, Благодаря, teşekkür ederim
merci, obrigado, bedankt, kiitos, xie xie, ありがとう.

Questions?

Find me on Twitter: @heathervc

email: heather at jcp dot org



Java™

Community Process