What is cool in Java 8 and new in 9

ORACLE CODE

developer.oracle.com

Aurelio Garcia-Ribeyro
Director of Product Management
Java Platform Group
March 2017

Live for the Code

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

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.

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Introduction

Aurelio Garcia-Ribeyro

- Director Product Management, Java Platform Group, Oracle
- In charge of managing the product requirements for Oracle's JDK since joining Oracle through the Sun acquisition in 2010
- Before joining Oracle, worked at Sun Microsystems for the Java Product Management team
- MBA from MIT Sloan and Bachelor degree in Systems Engineering from Universidad de Lima

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Java 8

“One of the biggest updates ever to a major language”

Andrew Binstock

Former Editor in Chief, Dr.Dobbs , now with Java Magazine

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abridged Content List for JDK 8

- Lambda Expressions
- Default Methods
- Method References
- Date Time APIs - JSR 310

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abstracting over Behavior

```
Collection<Person> people = ...;

for (Person p : people){
     if (p.getAge() > 18)
          ?? How do we remove this person ???
}
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abstracting over Behavior

```
Collection<Person> people = ...;

Iterator<Person> it = people.iterator();
while (it.hasNext()) {
    Person p = it.next();
    if (p.getAge() > 18)
        it.remove();
}
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abstracting over Behavior

The API Designer
Could create methods in the collection

```
interface Predicate<T> {
    boolean test(T t);
}

class Collections {
    public static<T>
        void removeIf(Collection<T> coll,
                      Predicate<T> pred) {
            ...
        }
}
```

Abstracting over Behavior

But the code to use the new method would be bloated

```
Collection<Person> people = ...;

Collections.removeIf(people,
    new Predicate<Person>() {
        public boolean test(Person p) {
            return p.getAge() > 18;
        }
    }
});
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abstracting over Behavior

```
Collection<Person> people = ...;

Collections.removeIf(people,
    new Predicate<Person>() {
        public boolean test(Person p) {
            return p.getAge() > 18;
        }
    }
});
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abstracting over Behavior

```
Collection<Person> people = ...;

Collections.removeIf(people,
                     p -> p.getAge() > 18);
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Aggregate operations

```
Collection<Person> people = ...;

int highestWeight = 0;
for (Person p : people) {
    if (p.getGender() == MALE) {
        int weight = p.getWeight();
        highestWeight = max(highestWeight,weight);
    }
}
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Parallelism

```
Collection<Person> people = ...;

int highestWeight =
    people.stream()
          .filter(p -> p.getGender() == MALE)
          .mapToInt(p -> p.getWeight())
          .max();
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Parallelism

```
class MaxProblem {

  final List<Person> people;
  final int size;

  MaxProblem(List<Person> ps) {
    this.people = ps;
    size = ps.size();
  }

  public int solveSequentially() {
    int max = 0;
    for (Person p : people) {
      if (p.getGender() == MALE)
          max = Math.max(max, p.getWeight());
    }
    return max;
  }

  public MaxProblem subproblem(int start, int end) {
    return new MaxProblem(people.subList(start, end));
  }

}
```

```
class MaxFinder extends RecursiveAction {

  private final MaxProblem problem;
  int max;

  protected void compute() {
    if (problem.size < THRESHOLD)
      sum = problem.solveSequentially();
    else {
      int m = problem.size / 2;
      MaxFinder left, right;
      left = new MaxFinder(problem.subproblem(0, m))
      right = new MaxFinder(problem.subproblem(m, problem.size));
      forkJoin(left, right);
      max = Math.max(left.max, right.max);
    }
  }

}

ForkJoinExecutor pool = new ForkJoinPool(nThreads);
MaxFinder finder = new MaxFinder(problem);
pool.invoke(finder);
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Parallelism

```
Collection<Person> people = ...;

int highestWeight =
    people.parallelStream()
          .filter(p -> p.getGender() == MALE)
          .mapToInt(p -> p.getWeight())
          .max();
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abridged Content List

- Lambda Expressions
- Default Methods
- Method References
- Date Time APIs - JSR 310

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Default methods

```
Collection<Person> people = ...;

int highestWeight =
   people.stream()
         ...

interface Collection<T> {
    ...
    default Stream<T> stream() {
        ...
    }
}
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Static Methods In Interfaces

- Previously it was not possible to include static methods in an interface
- Static methods, by definition, are not abstract
  - `@FunctionalInterface` can have zero or more static methods

```
static <T> Predicate<T> isEqual(Object target) {
        return (null == target)
                ? Objects::isNull
                : object -> target.equals(object);
}
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Method References

```
list.replaceAll(s -> s.toUpperCase());
list.replaceAll(String::toUpperCase);


list.sort(Comparator.comparing(p -> p.getName()));
list.sort(Comparator.comparing(Person::getName));
```

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Summary

- Lambdas are functions
- Java SE 8 defines new APIs using functional interfaces
- Default methods
- Method references

Lambdas make code read more like the problem statement. The resulting code is clear, concise, and easy to maintain and modify.

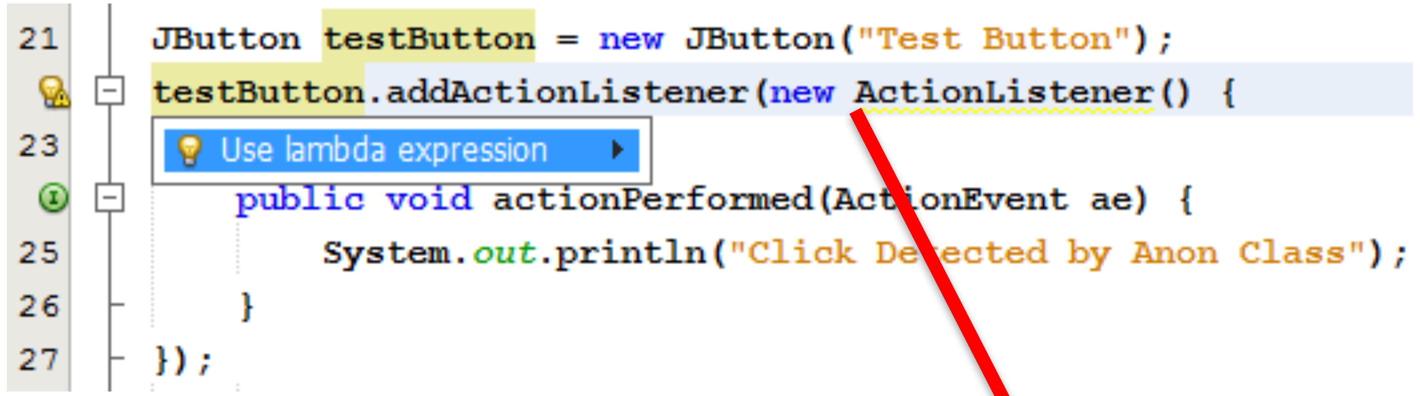
ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

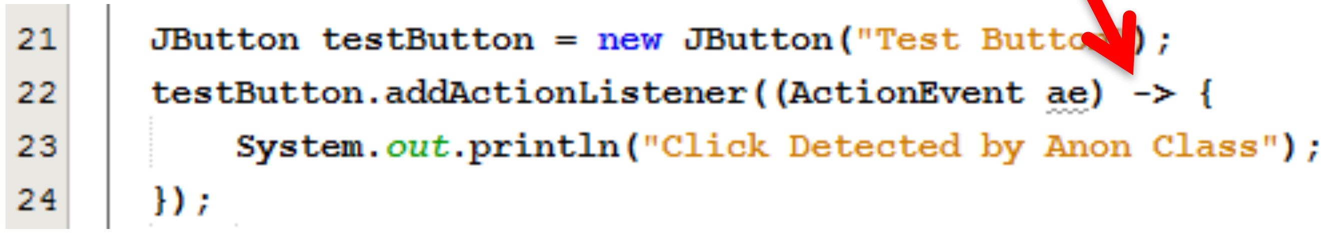
Tools for Java SE 8

Lambda Expressions

- Quickly convert anonymous inner classes to lambdas



```
JButton testButton = new JButton("Test Button");
testButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent ae) {
        System.out.println("Click Detected by Anon Class");
    }
});
```

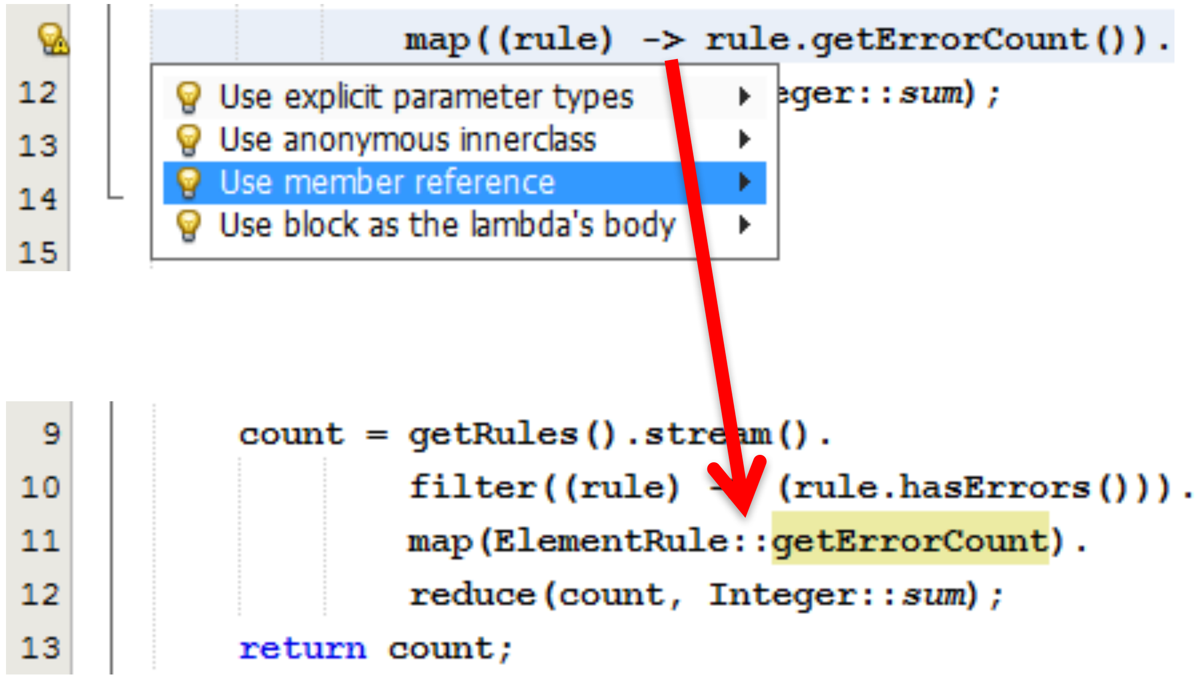


```
JButton testButton = new JButton("Test Button");
testButton.addActionListener((ActionEvent ae) -> {
    System.out.println("Click Detected by Anon Class");
});
```

Tools for Java SE 8

Method References

- Easily convert from lambdas to method references



```
map((rule) -> rule.getErrorCount()).
```

```
count = getRules().stream().
        filter((rule) -> (rule.hasErrors())).
        map(ElementRule::getErrorCount).
        reduce(count, Integer::sum);
return count;
```

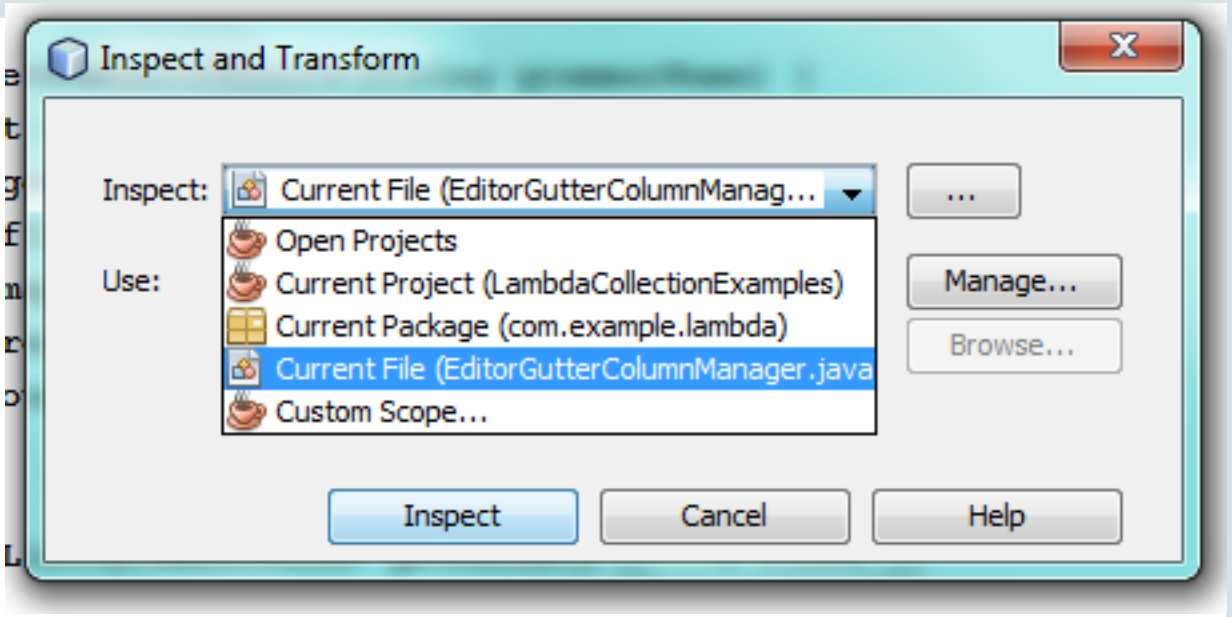
ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Tools for Java SE 8

Refactoring in Batch Mode

- Specify a scope for upgrading to Java 8
  - All/current projects
  - Specific package
  - Specific class
- Run converters
- Visually preview proposals for refactoring

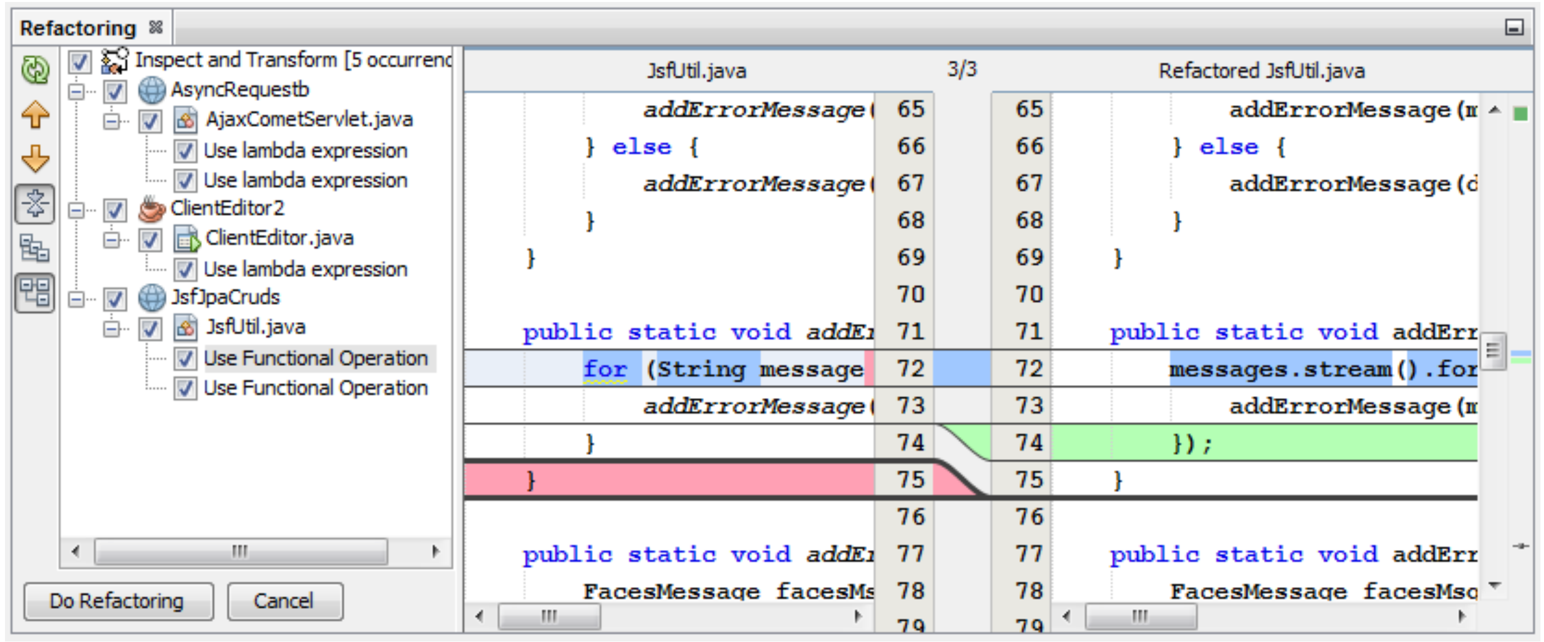


Inspect and Transform

Inspect: Current File (EditorGutterColumnManag...

Use:

- Open Projects
- Current Project (LambdaCollectionExamples)
- Current Package (com.example.lambda)
- Current File (EditorGutterColumnManager.java
- Custom Scope...



Refactoring

- Inspect and Transform [5 occurrenc
  - AsyncRequestb
    - AjaxCometServlet.java
      - Use lambda expression
      - Use lambda expression
  - ClientEditor2
    - ClientEditor.java
      - Use lambda expression
  - JsfJpaCruds
    - JsfUtil.java
      - Use Functional Operation
      - Use Functional Operation

JsfUtil.java | 3/3 | Refactored JsfUtil.java

```
            addErrorMessage(
        } else {
            addErrorMessage(
        }
    }

    public static void addE
        for (String message
            addErrorMessage(
        }
    }

    public static void addE
        FacesMessage facesMs
```

```
            addErrorMessage(m
        } else {
            addErrorMessage(d
        }
    }

    public static void addErr
        messages.stream().for
            addErrorMessage(m
        });
    }

    public static void addErr
        FacesMessage facesMsg
```

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Abridged Content List

- Lambda Expressions
- Default Methods
- Method References
- Date Time APIs - JSR 310

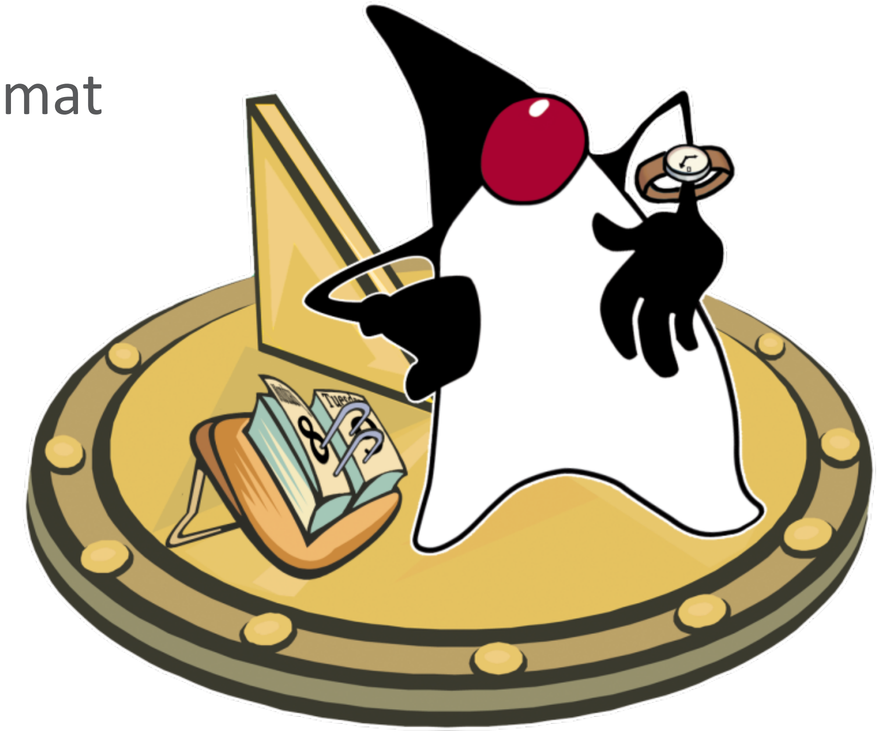
ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JDK 8 Java Time Features – JSR 310

New Improved Date Time API

- Replaces java.util.Date, Calendar, TimeZone, DateFormat
- Fluent, Immutable, Thread Safe, Easy to use
- Strong typing with fit for purpose types
- Easy to use formatting and parsing
- Interoperable with java.util.Calendar/Date
- Extensible Units, Fields, and Chronologies
- Supports Regional Calendars
- Supported by JDBC, java.sql.Date/Time/Timestamp
- The essential ISO 8601 Calendar for global business



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Range of types

- Date consists of year, month and day
- Time-of-day consists of hour, minute, second
  - **Loca** e
  - **Loca** e
  - **Loca** Time
- Time-zones I
  - **Zone** lar
- Instantaneou
  - **Instant** - closest class to java.util.Date

In JDK 8, you have to choose the correct date/time type when designing an application!

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Local Date

- Stores year-month-day
  - 12th March 2017
- Use cases: birthdays, start/end dates, holiday dates

```
LocalDate current = LocalDate.now();
LocalDate date = LocalDate.of(2013,Month.SEPTEMBER,12);
if (current.isAfter(date)) …

String str = date.toString();  // 2013-09-12

boolean leap = date.isLeapYear();
int monthLen = date.lengthOfMonth();
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Local Time

- Stores hour-minute-second-nanosecond
  - 13:30 (1:30pm)
- Use cases: shop opening hours, alarm clock

```
LocalTime current = LocalTime.now();
LocalTime time = LocalTime.of(13,30);
if (current.isAfter(time)) …

String str = time.toString();  // 13:30

time = time.plusHours(4).minusMinutes(1).withNano(0);
time = time.truncatedTo(ChronoUnit.SECONDS);
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Local Date-Time

- Stores **`LocalDate`** and **`LocalTime`**
  - 12th September 2013 at 13:30
- Use case: local date-time a flight takes off

```
dt1 = LocalDateTime.now();
dt2 = LocalDateTime.of(2013,SEPTEMBER,12, 13,30);

dt1 = dt1.plusDays(2).minusHours(1);
dt1 = dt1.with(next(TUESDAY));

dt2.toString();  // 2013-09-12T13:30

dt2 = dt2.truncatedTo(MINUTES);
```

ORACLE
Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

http://www.flickr.com/photos/estevesm/473866656/

Instant

- Stores nanoseconds from 1970-01-01Z
- Closest equivalent to java.util.Date
- Use case: timestamp in logging

```
instant1 = Instant.now();
instant2 = Instant.now();

if (instant1.isAfter(instant2)) { … }
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Time zones

- World is divided into various time zones
- Time in
- Rules ch

If you can avoid time-zones
your application will be simpler!

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Time zone design

- Four classes manage time-zone complexity

- **`ZoneId`** - "Europe/Paris", as per **`java.util.TimeZone`**
- **`ZoneOffset`** - "-05:00", offset from UTC/Greenwich
- **`ZoneRules`** - behind the scenes class defining the rules
- **`ZonedDateTime`** - main date/time class with time-zones

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Calendar systems

- All main classes use "ISO" calendar system
- Calendar system defined in ISO-8601
  - current 'civil' calendar applied to all time
  - not historically accurate
- Other calendar systems also supported
  - not supported to the same degree as ISO
  - Hijrah, Japanese, Minguo, ThaiBuddhist in the JDK
- Only affect dates, not times

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Duration

- Time-based amount
  - hours, minutes, seconds and nanoseconds
  - some support for 24-hour days
- Use cases: sport stopwatch, timeout

```
duration = Duration.ofHours(6);          // PT6H
duration = duration.multipliedBy(3);     // PT18H
duration = duration.plusMinutes(30);     // PT18H30M

dt = LocalDateTime.now();
dt = dt.plus(duration);
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Period

- Date-based amount
  - years, months and days
- Use cases: length of pregnancy, length of holiday

```
period = Period.ofMonths(9);    // P9M
period = period.plusDays(6);    // P9M6D

dt = LocalDateTime.now();
dt = dt.plus(period);

period = Period.between(startDate, endDate);
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Summary

`LocalDate`	`2016-12-03`
`LocalTime`	`11:05:30`
`LocalDateTime`	`2016-12-03T11:05:30`
`ZonedDateTime`	`2016-12-03T11:05:30+01:00 Europe/Paris`
`Instant`	*`2576458258.266 seconds after 1970-01-01`*
`Duration`	`PT30S` *`(30 seconds)`*
`Period`	`P1Y6M` *`(1 year and 6 months)`*

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JDK 8

Innovation
- Lambda aka Closures
- Language Interop
- Nashorn

Core Libraries
- Parallel operations for core collections APIs
- Improvements in functionality
- Improved type inference

Security
- Limited doPrivilege
- NSA Suite B algorithm support
- SNI Server Side support
- DSA updated to FIPS186-3
- AEAD JSSE CipherSuites

Java for Everyone
- Profiles for constrained devices
- JSR 310-Date & Time APIs
- Non-Gregorian calendars
- Unicode 6.2
- ResourceBundle
- BCP47 locale matching
- Globalization & Accessibility

Client
- Deployment enhancements
- JavaFX 8
- Public UI Control API
- Java SE Embedded support
- Enhanced HTML5 support
- 3D shapes and attributes
- Printing

Tools
- JSR 308-Annotations on Java Type
- Native app bundling
- App Store Bundling tools
- jdeps

General Goodness
- JVM enhancements
- No PermGen limitations
- Performance improvements

Enterprise
- Mission Control
- Flight Recorder
- Usage Tracker
- Advanced Management Console
- MSI Enterprise JRE Installer

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Search for:
Java 8 Launch

Where did most of this information come from

Java 8 Launch Event

- Java SE 8—Language and Library Features, Brian Goetz
- Introduction to Lambda Expressions, Stuart Marks
- A New Date and Time API—JSR-310, Stephen Colebourne

https://www.oracle.com/**java8launch**

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

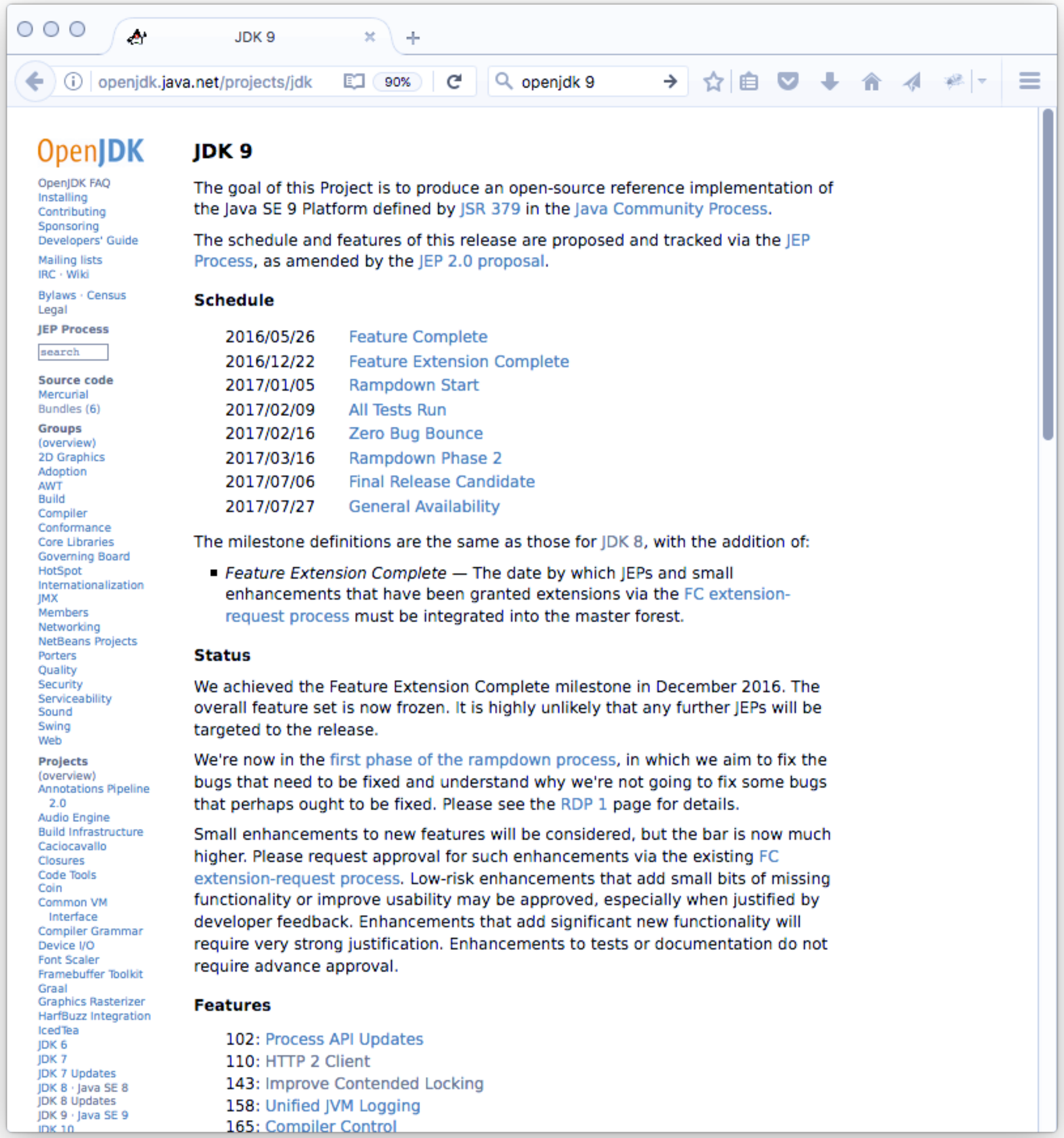
Coming in JDK 9

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Agenda

Search for:
OpenJDK 9

- Behind the scenes improvements
- New features and functionality
- Adopting new standards
- Housekeeping
- Gone, gone, gone!



OpenJDK

OpenJDK FAQ
Installing
Contributing
Sponsoring
Developers' Guide

Mailing lists
IRC · Wiki

Bylaws · Census
Legal

JEP Process

search

Source code
Mercurial
Bundles (6)

Groups
(overview)
2D Graphics
Adoption
AWT
Build
Compiler
Conformance
Core Libraries
Governing Board
HotSpot
Internationalization
JMX
Members
Networking
NetBeans Projects
Porters
Quality
Security
Serviceability
Sound
Swing
Web

Projects
(overview)
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 · Java SE 8
JDK 8 Updates
JDK 9 · Java SE 9
JDK 10

JDK 9

The goal of this Project is to produce an open-source reference implementation of the Java SE 9 Platform defined by JSR 379 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 2
2017/07/06	Final Release Candidate
2017/07/27	General Availability

The milestone definitions are the same as those for JDK 8, with the addition of:

- *Feature Extension Complete* — The date by which JEPs and small enhancements that have been granted extensions via the FC extension-request process must be integrated into the master forest.

Status

We achieved the Feature Extension Complete milestone in December 2016. The overall feature set is now frozen. It is highly unlikely that any further JEPs will be targeted to the release.

We're now in the first phase of the rampdown process, in which we aim to fix the bugs that need to be fixed and understand why we're not going to fix some bugs that perhaps ought to be fixed. Please see the RDP 1 page for details.

Small enhancements to new features will be considered, but the bar is now much higher. Please request approval for such enhancements via the existing FC extension-request process. Low-risk enhancements that add small bits of missing functionality or improve usability may be approved, especially when justified by developer feedback. Enhancements that add significant new functionality will require very strong justification. Enhancements to tests or documentation do not require advance approval.

Features

102: Process API Updates
110: HTTP 2 Client
143: Improve Contended Locking
158: Unified JVM Logging
165: Compiler Control

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

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Behind the scenes improvements

Goodness you get for free just by updating to JDK 9
No need for user to change anything to benefit from these

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 250: Store Interned Strings in CDS Archives

hotspot / runtime

- Store interned strings in class-data sharing (CDS) archives
- Reduce memory consumption by sharing the String objects and underlying char array objects amongst different JVM processes
- Only support shared strings for the G1 GC. Shared strings require a pinned region, and G1 is the only HotSpot GC that supports pinning
- Only support 64-bit platforms with compressed object and class pointers

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 254: Compact Strings

core-libs / java.lang

- Adopt a more space-efficient internal representation for strings

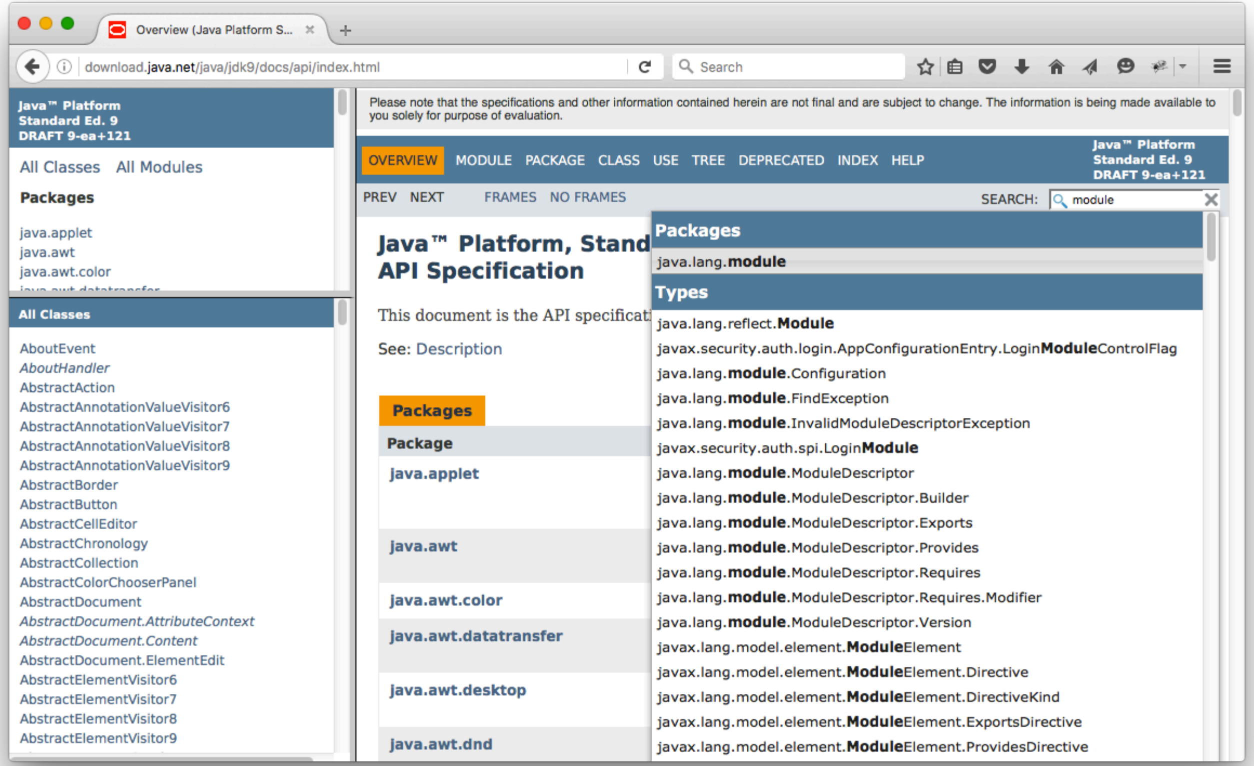
- Less memory used for storing strings
- String class stores characters in a char array, using two bytes (sixteen bits) for each character
- Change the internal representation of the String class to a byte array plus an encoding-flag field

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 225: Javadoc Search

tools / javadoc(tool)

- Add a search box to generated API documentation that can be used to search for program elements and tagged words and phrases within the documentation



Overview (Java Platform S...

download.java.net/java/jdk9/docs/api/index.html

Search

Java™ Platform
Standard Ed. 9
DRAFT 9-ea+121

All Classes All Modules

Packages

java.applet
java.awt
java.awt.color

All Classes

AboutEvent
AboutHandler
AbstractAction
AbstractAnnotationValueVisitor6
AbstractAnnotationValueVisitor7
AbstractAnnotationValueVisitor8
AbstractAnnotationValueVisitor9
AbstractBorder
AbstractButton
AbstractCellEditor
AbstractChronology
AbstractCollection
AbstractColorChooserPanel
AbstractDocument
AbstractDocument.AttributeContext
AbstractDocument.Content
AbstractDocument.ElementEdit
AbstractElementVisitor6
AbstractElementVisitor7
AbstractElementVisitor8
AbstractElementVisitor9

Please note that the specifications and other information contained herein are not final and are subject to change. The information is being made available to you solely for purpose of evaluation.

OVERVIEW MODULE PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Java™ Platform
Standard Ed. 9
DRAFT 9-ea+121

PREV NEXT FRAMES NO FRAMES SEARCH: module

Java™ Platform, Stand
API Specification

This document is the API specificat

See: Description

Packages

Package
java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.desktop
java.awt.dnd

Packages

java.lang.**module**

Types

java.lang.reflect.**Module**
javax.security.auth.login.AppConfigurationEntry.Login**Module**ControlFlag
java.lang.**module**.Configuration
java.lang.**module**.FindException
java.lang.**module**.InvalidModuleDescriptorException
javax.security.auth.spi.Login**Module**
java.lang.**module**.ModuleDescriptor
java.lang.**module**.ModuleDescriptor.Builder
java.lang.**module**.ModuleDescriptor.Exports
java.lang.**module**.ModuleDescriptor.Provides
java.lang.**module**.ModuleDescriptor.Requires
java.lang.**module**.ModuleDescriptor.Requires.Modifier
java.lang.**module**.ModuleDescriptor.Version
javax.lang.model.element.**Module**Element
javax.lang.model.element.**Module**Element.Directive
javax.lang.model.element.**Module**Element.DirectiveKind
javax.lang.model.element.**Module**Element.ExportsDirective
javax.lang.model.element.**Module**Element.ProvidesDirective

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

New features and functionality

New tools and capabilities likely to be useful to most developers
Will have to choose to use these

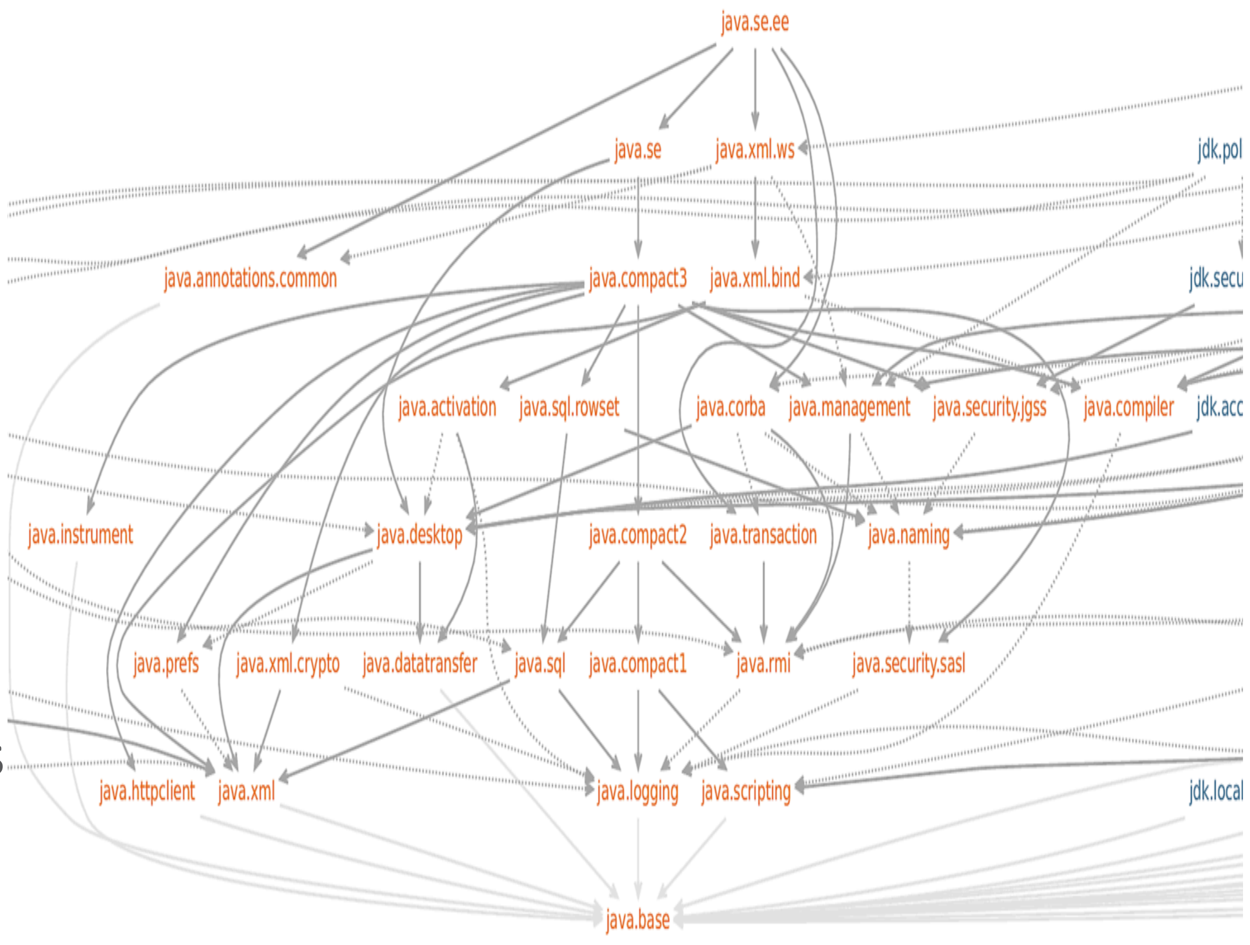
ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

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



ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

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

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 277: Enhanced Deprecation

core-libs / java.lang

@Deprecated(since=9, condemned=true)

- Revamp the deprecation annotation, and provide tools to strengthen the API life cycle
- Provide better information about the status and intended disposition of APIs in the specification
- Provide a tool to analyze an application's static usage of deprecated APIs
- Provide a tool to detect an application's dynamic usage of of deprecated APIs in order to emit warnings at runtime

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 269: Convenience Factory Methods for Collections

core-libs / java.util:collections

- Define library APIs to make it convenient to create instances of collections and maps with small numbers of elements, so as to ease the pain of not having collection literals in the Java programming language

- Decrease the amount of code needed for creating small collections and maps

```
Set<String> alphabet = Set.of("a", "b", "c");
```

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 222: jshell: The Java Shell (Read-Eval-Print Loop)

tools / jshell

- Provide an interactive tool to evaluate declarations, statements, and expressions of the Java programming language, together with an API so that other applications can leverage this functionality
- A Read-Eval-Print Loop (REPL) is an interactive programming tool which loops, continually reading user input, evaluating the input, and printing the value of the input or a description of the state change the input caused. Scala, Ruby, JavaScript, Haskell, Clojure, and Python all have REPLs and all allow small initial programs. JShell adds REPL functionality to the Java platform

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 238: Multi-Release JAR Files

tools / jar

- Extend the JAR file format to allow multiple, Java-release-specific versions of class files to coexist in a single archive

- Write JDK-version-specific variants of the same code into a single jar file

```
jar root
  - A.class
  - B.class
  - C.class
  - D.class
  - META-INF
     - versions
        - 9
           - A.class
           - B.class
```

Adopting new standards

JDK 9 keeping up with improvements in the industry

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 267: Unicode 8.0

core-libs / java.lang

- Upgrade existing platform APIs to support version 8.0 of the Unicode Standard

Ahom

Anatolian Hieroglyphs

Hatran

Multani

Old Hungarian

Sutton SignWriting

🤓 🍾

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 226: UTF-8 Property Files

core-libs / java.util:i18n

- Define a means for applications to specify property files encoded in UTF-8, and extend the ResourceBundle API to load them
- The platform has a properties-file format based on ISO-8859-1 and an escape mechanism for characters that cannot be represented in that encoding

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 249: OCSP Stapling for TLS

security-libs / javax.net.ssl

- Implement OCSP stapling via the TLS Certificate Status Request extension and the Multiple Certificate Status Request Extension
- Certificate status checking using OCSP typically involves a network request for each certificate being checked. Because of the additional network requests, enabling OCSP checking for TLS on the client side can have a significant impact on performance.
- OCSP stapling allows the presenter of a certificate, rather than the issuing Certificate Authority (CA), to bear the resource cost of providing OCSP responses

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 287: SHA-3 Hash Algorithms

security-libs / java.security

- Implement the SHA-3 cryptographic hash functions (BYTE-only) specified in NIST FIPS 202

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 224: HTML5 Javadoc

tools / javadoc(tool)

- Enhance the javadoc tool to allow generating HTML5 markup.

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 110: HTTP/2 Client

core-libs / java.net

- Define a new HTTP client API that implements HTTP/2 and WebSocket, and can replace the legacy HttpURLConnection API
- For JDK 9 this API will be on the incubator modules (JEP 11) with the goal of adding them to the standard on a later release.

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Housekeeping

Setting up for future improvements and reducing complexity

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 260: Encapsulate Most Internal APIs

- Make most of the JDK's internal APIs inaccessible by default but leave a few critical, widely-used internal APIs accessible, until supported replacements exist for all or most of their functionality
  - In order to keep critical APIs without a replacement accessible by default sun.misc and sun.reflect will not be hidden and a few APIs kept "public"
    - sun.misc.Unsafe
    - sun.misc.{Signal,SignalHandler}
    - sun.reflect.Reflection::getCallerClass
    - sun.reflect.ReflectionFactory
  - All other APIs in these packages (e.g. sun.misc.Base64) will be removed

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 275: Modular Java Application Packaging

deploy / packager

- Integrate features from Project Jigsaw into the Java Packager, including module awareness and custom runtime creation
- Leverage jlink in our packager to create smaller packages
- The packager will only create applications that use the JDK 9 runtime.

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 223: New Version-String Scheme

- Revise the JDK's version-string scheme so that it is easier to distinguish major, minor, and security-update releases
- Align with current industry practices, in particular Semantic Versioning
- Provide a simple API for version-string parsing, validation, and comparison

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 295: Ahead-of-Time Compilation

hotspot / compiler

- Compile Java classes to native code prior to launching the virtual machine.
- Improve the start-up time of both small and large Java applications, with at most a limited impact on peak performance.
- AOT compilation is done by a new tool, jaotc
- For the initial release, the only supported module is java.base.
- AOT compilation of any other JDK module, or of user code, is experimental

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 280: Indify String Concatenation

tools / javac

- Change the static String-concatenation bytecode sequence generated by javac to use invokedynamic calls to JDK library functions. This will enable future optimizations of String concatenation without requiring further changes to the bytecode emitted by javac

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 271: Unified GC Logging

hotspot / gc

- Reimplement GC logging using the unified JVM logging framework introduced in JEP 158

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 248: Make G1 the Default Garbage Collector

hotspot / gc

- Make G1 the default garbage collector on 32- and 64-bit server configurations

- Limiting GC pause times is, in general, more important than maximizing throughput. Switching to a low-pause collector such as G1 should provide a better overall experience, for most users, than a throughput-oriented collector such as the Parallel GC, which is currently the default

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 213: Milling Project Coin

tools / javac

- Address the rough edges of the changes included in Project Coin / JSR 334 as part of JDK 7 / Java SE 7

1. Allow @SafeVargs on private instance methods
2. Allow effectively-final variables to be used as resources in the try-with-resources statement
3. Allow diamond with anonymous classes if the argument type of the inferred type is denotable
4. Complete the removal, begun in Java SE 8, of underscore from the set of legal identifier names
5. Support for private interface methods

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 290: Filter Incoming Serialization Data

core-libs / java.io:serialization

- Allow incoming streams of object-serialization data to be filtered in order to improve both security and robustness

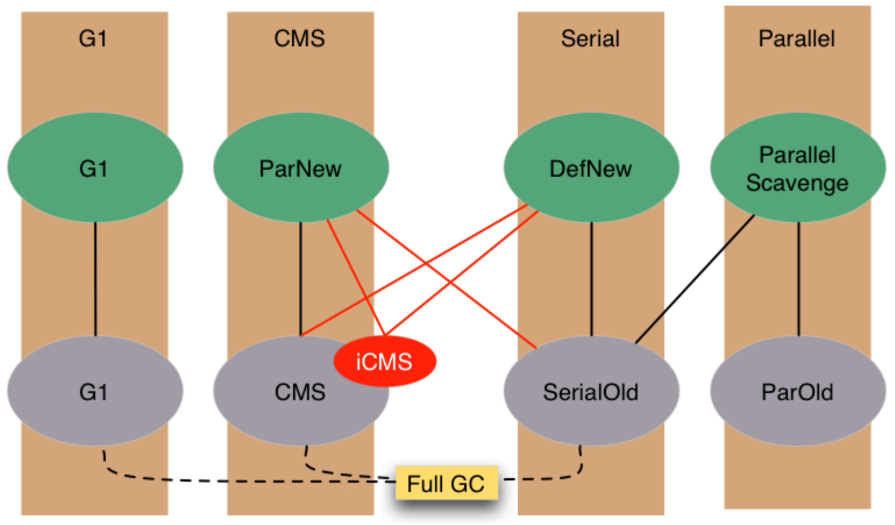
- Allows developers to lock out some serialized data

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

JEP 214: Remove GC Combinations Deprecated in JDK 8

hotspot / gc

- JEP 173 deprecated some GC combinations with JDK 8.
- Unsupported and untested since JDK 8
- Incremental CMS (iCMS) removed



Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

General Rule: Look Out for Unrecognized VM Options

- Launching JRE with unrecognized VM options fails.
- Using deprecated options in JDK 8 triggers warning messages:

```
$ java -XX:MaxPermSize=1G -version
Java HotSpot(TM) 64-Bit Server VM warning: ignoring option MaxPermSize; support was removed in 8.0
```

- Previously deprecated options, removed in JDK 9, will fail to start.
  - Perm generation was removed in JDK 8 (JEP 122).
  - Expect many programs to run into this problem.

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

- Store Interned Strings in CDS Archives
- Improve Contended Locking
- Compact Strings
- Improve Secure Application Performance
- Leverage CPU Instructions for GHASH and RSA
- Tiered Attribution for javac
- Javadoc Search
- Marlin Graphics Renderer
- HiDPI Graphics on Windows and Linux
- Enable GTK 3 on Linux
- Update JavaFX/Media to Newer Version of GStreamer

Behind the scenes

- **Jigsaw – Modularize JDK**
- Enhanced Deprecation
- Stack-Walking API
- Convenience Factory Methods for Collections
- Platform Logging API and Service
- jshell: The Java Shell (Read-Eval-Print Loop)
- Compile for Older Platform Versions
- Multi-Release JAR Files
- Platform-Specific Desktop Features
- TIFF Image I/O\
- Multi-Resolution Images

New functionality

- Process API Updates
- Variable Handles
- Spin-Wait Hints
- Dynamic Linking of Language-Defined Object Models
- Enhanced Method Handles
- More Concurrency Updates
- Compiler Control

Specialized

- HTTP 2 Client
- Unicode 8.0
- UTF-8 Property Files
- Implement Selected ECMAScript 6 Features in Nashorn
- Datagram Transport Layer Security (DTLS)
- OCSP Stapling for TLS
- TLS Application-Layer Protocol Negotiation Extension
- SHA-3 Hash Algorithms
- DRBG-Based SecureRandom Implementations
- Create PKCS12 Keystores by Default
- Merge Selected Xerces 2.11.0 Updates into JAXP
- XML Catalogs
- HarfBuzz Font-Layout Engine
- HTML5 Javadoc

New standards

- Parser API for Nashorn
- Prepare JavaFX UI Controls & CSS APIs for Modularization
- Modular Java Application Packaging
- New Version-String Scheme
- Reserved Stack Areas for Critical Sections
- Segmented Code Cache
- Ahead-of-Time Compilation
- Indify String Concatenation
- Unified JVM Logging
- Unified GC Logging
- Make G1 the Default Garbage Collector
- Use CLDR Locale Data by Default
- Validate JVM Command-Line Flag Arguments
- Java-Level JVM Compiler Interface
- Disable SHA-1 Certificates
- Simplified Doclet API
- Deprecate the Applet API
- Process Import Statements Correctly
- Annotations Pipeline 2.0
- Elide Deprecation Warnings on Import Statements
- Milling Project Coin
- Filter Incoming Serialization Data

Housekeeping

- Remove GC Combinations Deprecated in JDK 8
- Remove Launch-Time JRE Version Selection
- Remove the JVM TI hprof Agent
- Remove the jhat Tool

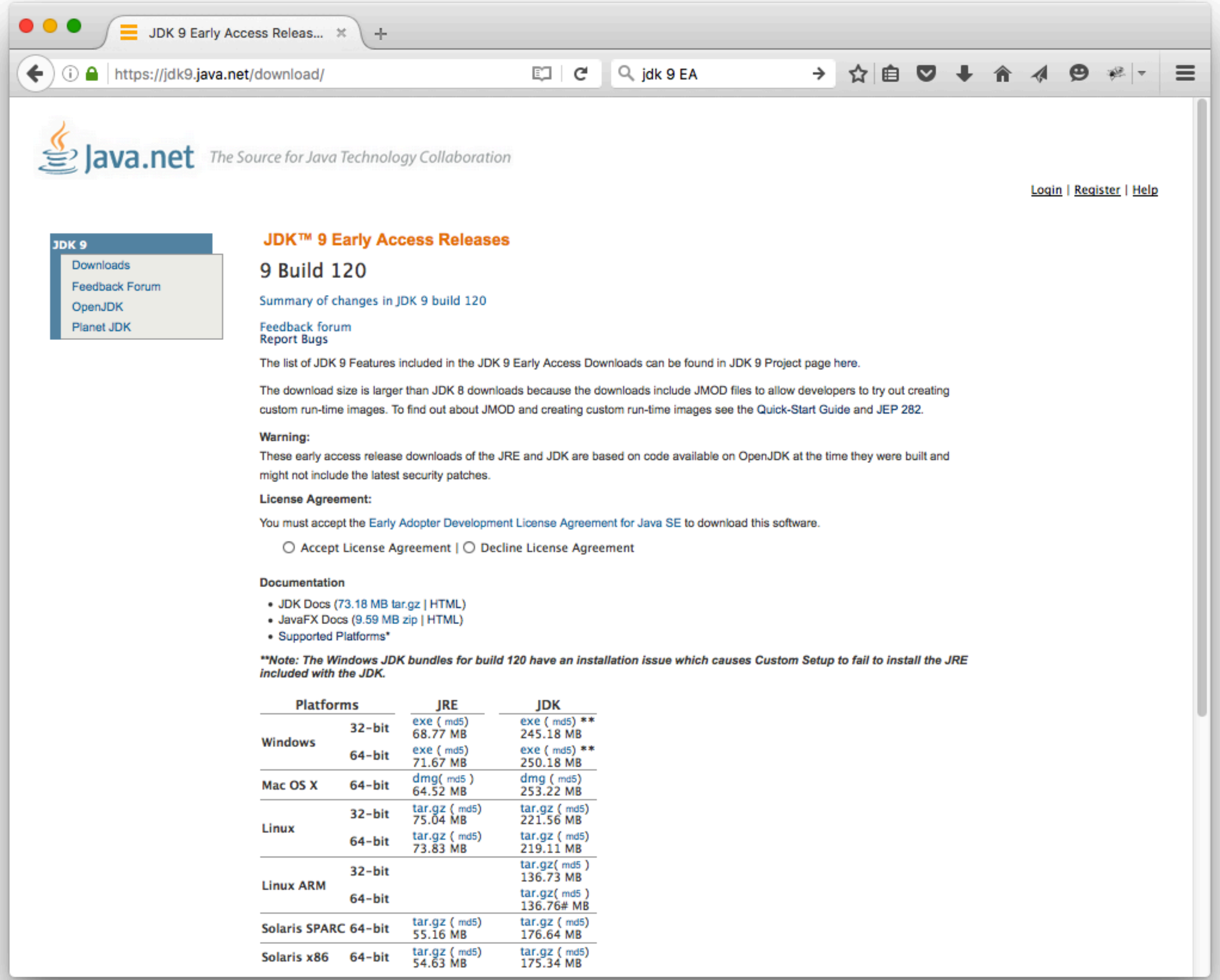
Gone

ORACLE

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Download JDK 9 EA

- Early access builds of JDK 9 available for testing:
  - https://jdk9.java.net/download/
- There are periodic updates, so check frequently for newer builds.
  - See "Summary of changes" on each build.



JDK 9 Early Access Releas...
https://jdk9.java.net/download/
jdk 9 EA
Java.net The Source for Java Technology Collaboration
Login | Register | Help
JDK 9
Downloads
Feedback Forum
OpenJDK
Planet JDK

JDK™ 9 Early Access Releases

9 Build 120

Summary of changes in JDK 9 build 120

Feedback forum
Report Bugs

The list of JDK 9 Features included in the JDK 9 Early Access Downloads can be found in JDK 9 Project page here.

The download size is larger than JDK 8 downloads because the downloads include JMOD files to allow developers to try out creating custom run-time images. To find out about JMOD and creating custom run-time images see the Quick-Start Guide and JEP 282.

Warning:
These early access release downloads of the JRE and JDK are based on code available on OpenJDK at the time they were built and might not include the latest security patches.

License Agreement:

You must accept the Early Adopter Development License Agreement for Java SE to download this software.

○ Accept License Agreement | ○ Decline License Agreement

Documentation

- JDK Docs (73.18 MB tar.gz | HTML)
- JavaFX Docs (9.59 MB zip | HTML)
- Supported Platforms*

*****Note: The Windows JDK bundles for build 120 have an installation issue which causes Custom Setup to fail to install the JRE included with the JDK.***

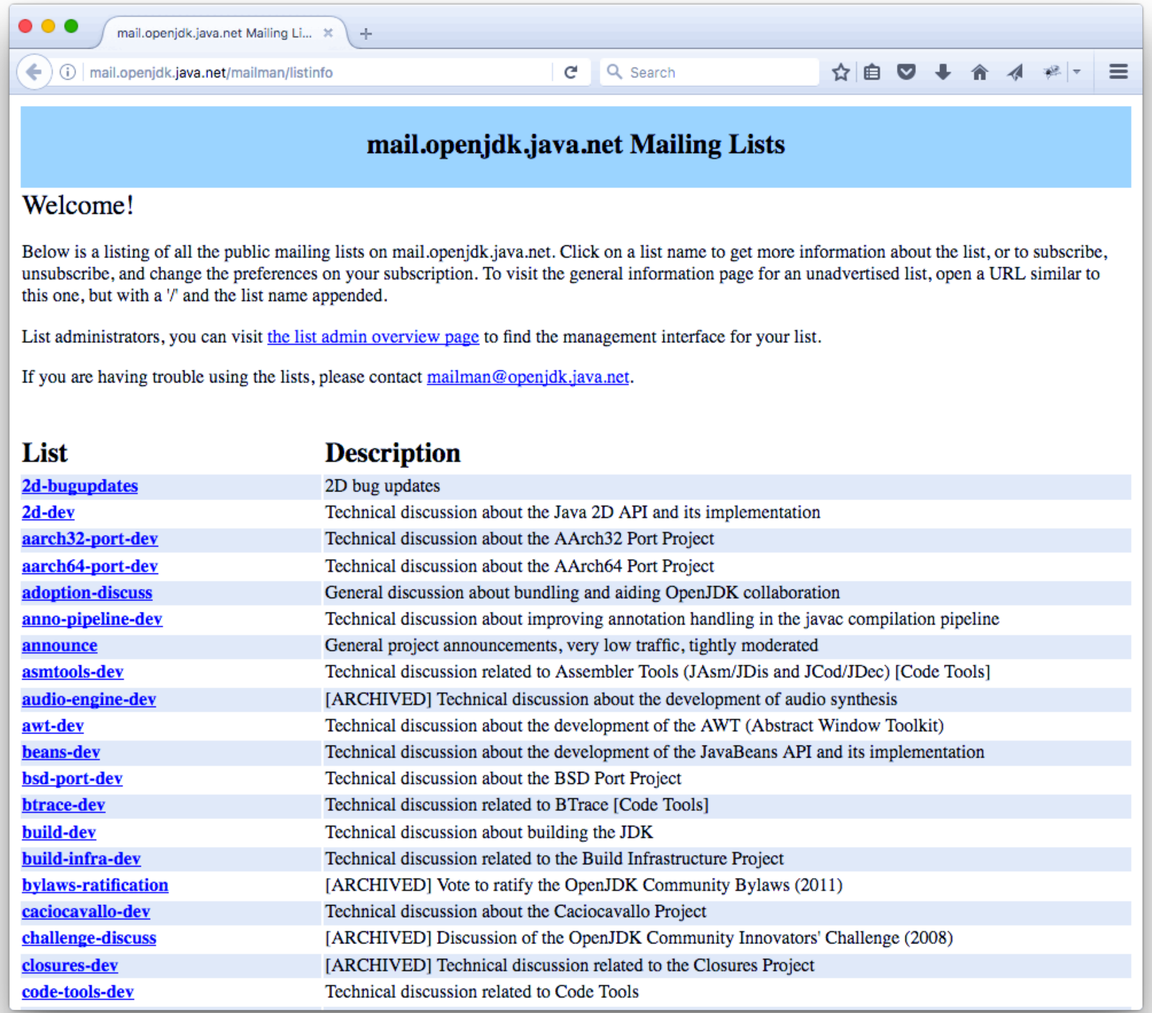
Platforms		JRE	JDK
Windows	32-bit	exe (md5) 68.77 MB	exe (md5) ** 245.18 MB
	64-bit	exe (md5) 71.67 MB	exe (md5) ** 250.18 MB
Mac OS X	64-bit	dmg(md5) 64.52 MB	dmg (md5) 253.22 MB
Linux	32-bit	tar.gz (md5) 75.04 MB	tar.gz (md5) 221.56 MB
	64-bit	tar.gz (md5) 73.83 MB	tar.gz (md5) 219.11 MB
Linux ARM	32-bit		tar.gz(md5) 136.73 MB
	64-bit		tar.gz(md5) 136.76# MB
Solaris SPARC	64-bit	tar.gz (md5) 55.16 MB	tar.gz (md5) 176.64 MB
Solaris x86	64-bit	tar.gz (md5) 54.63 MB	tar.gz (md5) 175.34 MB

ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Join the conversation

- There are OpenJDK aliases for all questions
  - http://mail.openjdk.java.net/mailman/listinfo
- Every JEP lists its alias
  - Look for "Discussion"


mail.openjdk.java.net Mailing Li...
mail.openjdk.java.net/mailman/listinfo
Search

mail.openjdk.java.net Mailing Lists

Welcome!

Below is a listing of all the public mailing lists on mail.openjdk.java.net. Click on a list name to get more information about the list, or to subscribe, unsubscribe, and change the preferences on your subscription. To visit the general information page for an unadvertised list, open a URL similar to this one, but with a '/' and the list name appended.

List administrators, you can visit the list admin overview page to find the management interface for your list.

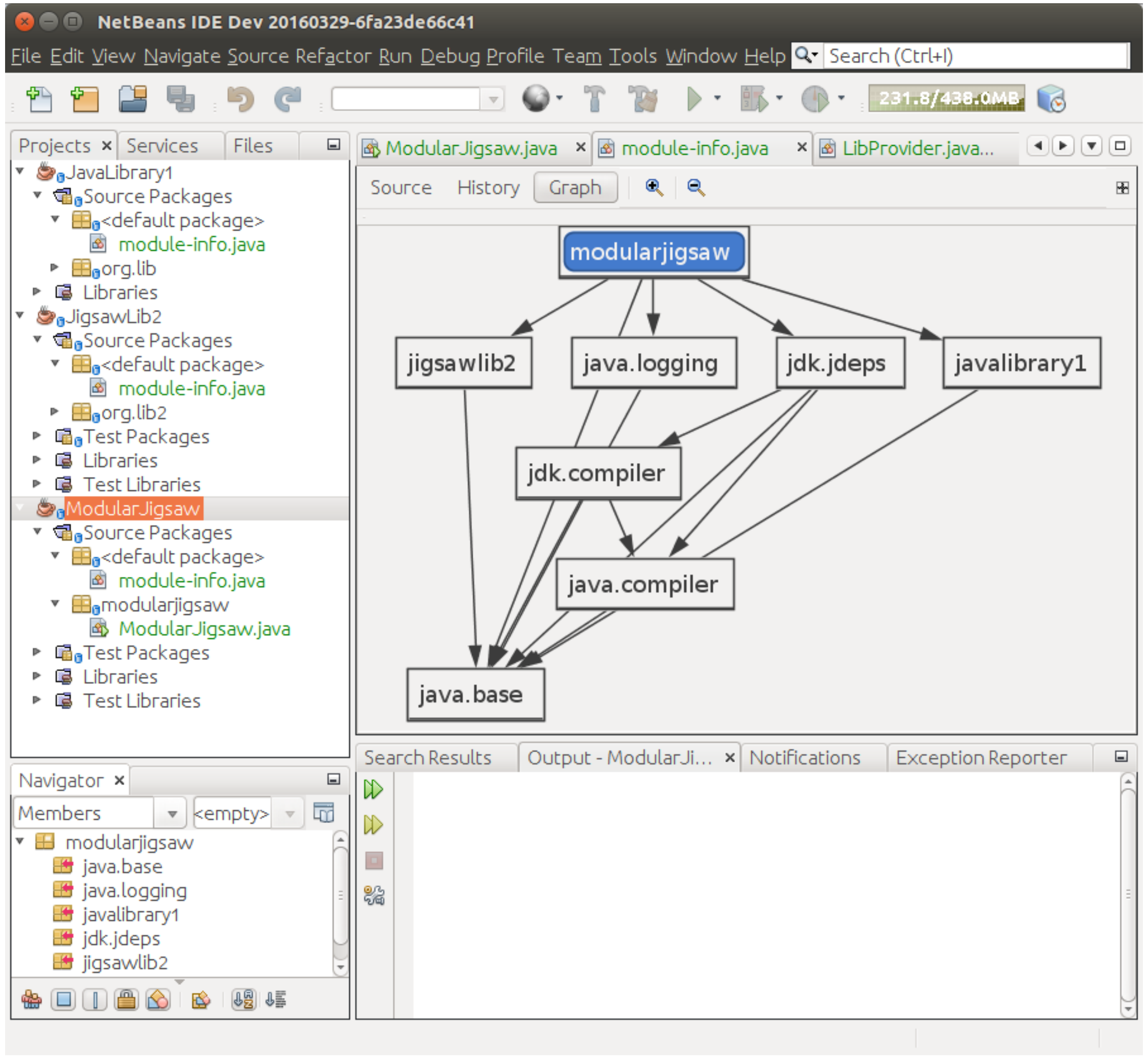
If you are having trouble using the lists, please contact mailman@openjdk.java.net.

List	Description
2d-bugupdates	2D bug updates
2d-dev	Technical discussion about the Java 2D API and its implementation
aarch32-port-dev	Technical discussion about the AArch32 Port Project
aarch64-port-dev	Technical discussion about the AArch64 Port Project
adoption-discuss	General discussion about bundling and aiding OpenJDK collaboration
anno-pipeline-dev	Technical discussion about improving annotation handling in the javac compilation pipeline
announce	General project announcements, very low traffic, tightly moderated
asmtools-dev	Technical discussion related to Assembler Tools (JAsm/JDis and JCod/JDec) [Code Tools]
audio-engine-dev	[ARCHIVED] Technical discussion about the development of audio synthesis
awt-dev	Technical discussion about the development of the AWT (Abstract Window Toolkit)
beans-dev	Technical discussion about the development of the JavaBeans API and its implementation
bsd-port-dev	Technical discussion about the BSD Port Project
btrace-dev	Technical discussion related to BTrace [Code Tools]
build-dev	Technical discussion about building the JDK
build-infra-dev	Technical discussion related to the Build Infrastructure Project
bylaws-ratification	[ARCHIVED] Vote to ratify the OpenJDK Community Bylaws (2011)
caciocavallo-dev	Technical discussion about the Caciocavallo Project
challenge-discuss	[ARCHIVED] Discussion of the OpenJDK Community Innovators' Challenge (2008)
closures-dev	[ARCHIVED] Technical discussion related to the Closures Project
code-tools-dev	Technical discussion related to Code Tools

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Download NetBeans Dev

- Early access of NetBeans with support for JDK 9 available at:
  - http://wiki.netbeans.org/JDK9Support



ORACLE®

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Identify Problematic Dependencies

Use Java Dependency Analysis Tool (`jdeps`)

- Available since JDK 8, Best results from the version in JDK 9 EA
- Option to find internal dependencies

```
tzupdater-2.0.3-2015b $ jdeps tzupdater.jar
tzupdater.jar -> java.base
   com.sun.tools.tzupdater                 -> com.sun.tools.tzupdater.utils       tzupdater.jar
(...)
   com.sun.tools.tzupdater                 -> java.util.regex                     java.base
   com.sun.tools.tzupdater                 -> java.util.zip                       java.base
   com.sun.tools.tzupdater                 -> sun.util.calendar                   JDK internal API (java.base)
   com.sun.tools.tzupdater                 -> tools.javazic                       tzupdater.jar
(...)
   com.sun.tools.tzupdater.utils           -> java.util                           java.base
   com.sun.tools.tzupdater.utils           -> sun.util.calendar                   JDK internal API (java.base)
   tools.javazic                           -> java.io                             java.base
(...)
```

https://wiki.openjdk.java.net/display/JDK8/Java+Dependency+Analysis+Tool

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Stopgap: Expose Internal APIs

But come back and fix!

- Sample command for earlier dev version of Netbeans

```
$ bin/netbeans --jdkhome ~/jdk9ea --add-exports java.desktop/sun.awt=ALL-UNNAMED --add-exports
java.base/jdk.internal.jrtfs=ALL-UNNAMED --add-exports java.desktop/java.awt.peer=ALL-UNNAMED -
-add-exports java.desktop/com.sun.beans.editors=ALL-UNNAMED --add-exports
java.desktop/sun.awt.im=ALL-UNNAMED  --add-exports java.desktop/com.sun.java.swing.plaf.gtk=ALL-
UNNAMED --add-exports java.management/sun.management=ALL-UNNAMED,
```

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.