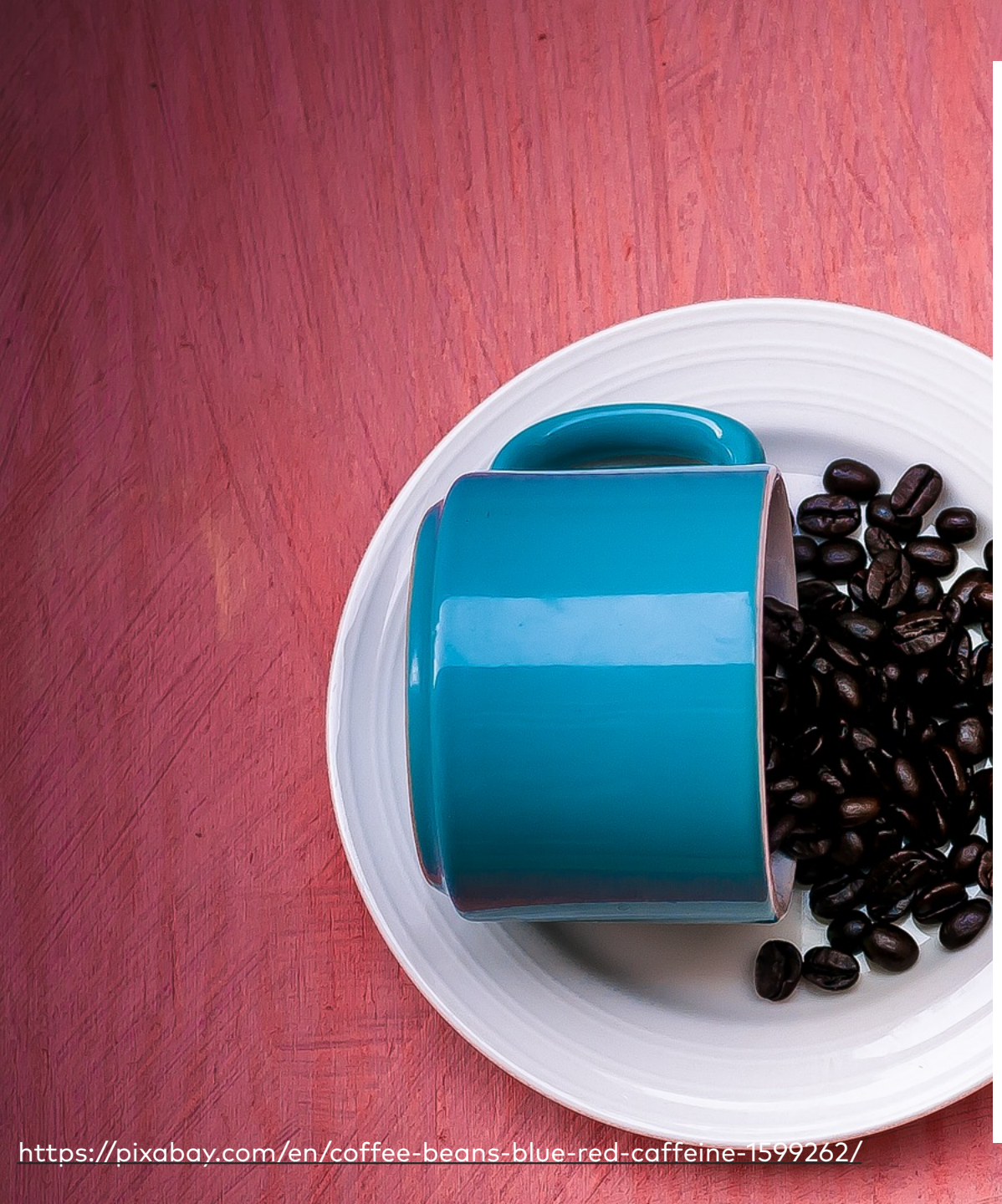


2019-05-16

Zürich / JUG Switzerland

Java 10, 11, 12, and beyond

Michael Vitz

INNOQ

@michaelvitz

https://pixabay.com/en/coffee-beans-blue-red-caffeine-1599262/

Michael Vitz

Senior Consultant
at INNOQ

- Build, run, and maintain JVM applications
- JavaSPEKTRUM column owner
- ❤️Clojure

www.innoq.com

SERVICES

Strategy & technology consulting
Digital business models
Software architecture & development
Digital platforms & infrastructures
Knowledge transfer, coaching & trainings

FACTS

~125 employees
Privately owned
Vendor-independent

OFFICES

Monheim
Berlin
Hamburg
Offenbach
Munich
Zurich

CLIENTS

Finance
Telecommunications
Logistics
E-commerce
Fortune 500
SMBs
Startups

Vote

https://pixabay.com/en/audience-crowd-event-cheer-945449/

JDK Version

5
6 ||
7 |||
8 ||||
9 |||
10 ||
11 |?

Usage

Module (JPMS)

Ja ||| |
Nein

Support

Ja |
Nein

Vendor

Oracle
OpenJDK
IBM |
Azul
2
.
OSS

Is Java Still Free?

https://pixabay.com/photos/dove-freedom-bird-nature-animal-2680487/

Workshop
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
Compatibility & Specification Review
Compiler
Conformance
Core Libraries
Governing Board
HotSpot
Internationalization
JMX
Members
Networking
NetBeans Projects
Porters

OpenJDK

What is this? The place to collaborate on an open-source implementation of the Java Platform, Standard Edition, and related projects. (Learn more.)

Download and install the open-source JDK for most popular Linux distributions. Oracle's OpenJDK JDK 11 binaries are at **jdk.java.net/11**; Oracle's JDK 11 product binaries for Linux, macOS, and Windows, based largely on the same code, are here.

Learn how to use the JDK to write applications for a wide range of environments.

http://openjdk.java.net/

The future of Java and OpenJDK updates without Oracle support

By Andrew Haley September 24, 2018

+13 rating, 13 votes

OpenJDK

Oracle recently announced that it would no longer supply free (as in beer) binary downloads for JDK releases after a six-month period, and neither would Oracle engineers write patches for OpenJDK bugs after that period. This has caused a great deal of concern among some Java users.

From my point of view, this is little more than business as usual. Several years ago, the OpenJDK 6 updates (jdk6u) project was relinquished by Oracle and I assumed leadership, and then the same happened with OpenJDK 7. Subsequently, Andrew Brygin of Azul took over the leadership of OpenJDK 6. The OpenJDK Vulnerability Group, with members from many organizations, collaborates on critical security issues. With the help of the wider OpenJDK community and my team at Red Hat, we have continued to provide updates for critical bugs and security vulnerabilities at regular intervals. I can see no reason why this process should not work in the same way for OpenJDK 8 and the next long-term support release, OpenJDK 11.

https://developers.redhat.com/blog/2018/09/24/the-future-of-java-and-openjdk-updates-without-oracle-support/

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 11.0.2 General-Availability Release

This page provides production-ready open-source builds of the Java Development Kit, version 11.0.2, an implementation of the Java SE 11.0.2 Platform under the GNU General Public License, version 2, with the Classpath Exception.

Commercial builds of JDK 11.0.2 from Oracle under a non-open-source license, for a wider range of platforms, can be found at the Oracle Technology Network.

Documentation

- Features
- Release notes
- API Javadoc
- Tool & command reference

Builds

Linux/x64	tar.gz (sha256)	187513052 bytes
macOS/x64	tar.gz (sha256)	182670822
Windows/x64	zip (sha256)	187383323

Notes

- The Alpine Linux build previously available on this page was removed as of JDK 11 GA. It's not production-ready because it hasn't been tested

Overview | **Downloads** | Documentation | Community | Technologies | Training

Java SE Development Kit 11 Downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

Important changes in Oracle JDK 11 License

With JDK 11 Oracle has updated the license terms on which we offer the Oracle JDK.
The new Oracle Technology Network License Agreement for Oracle Java SE is substantially different from the licenses under which previous versions of the JDK were offered. Please review the new terms carefully before downloading and using this product.

Oracle also offers this software under the GPL License on jdk.java.net/11

https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html

Oracle JDK and OpenJDK builds from Oracle

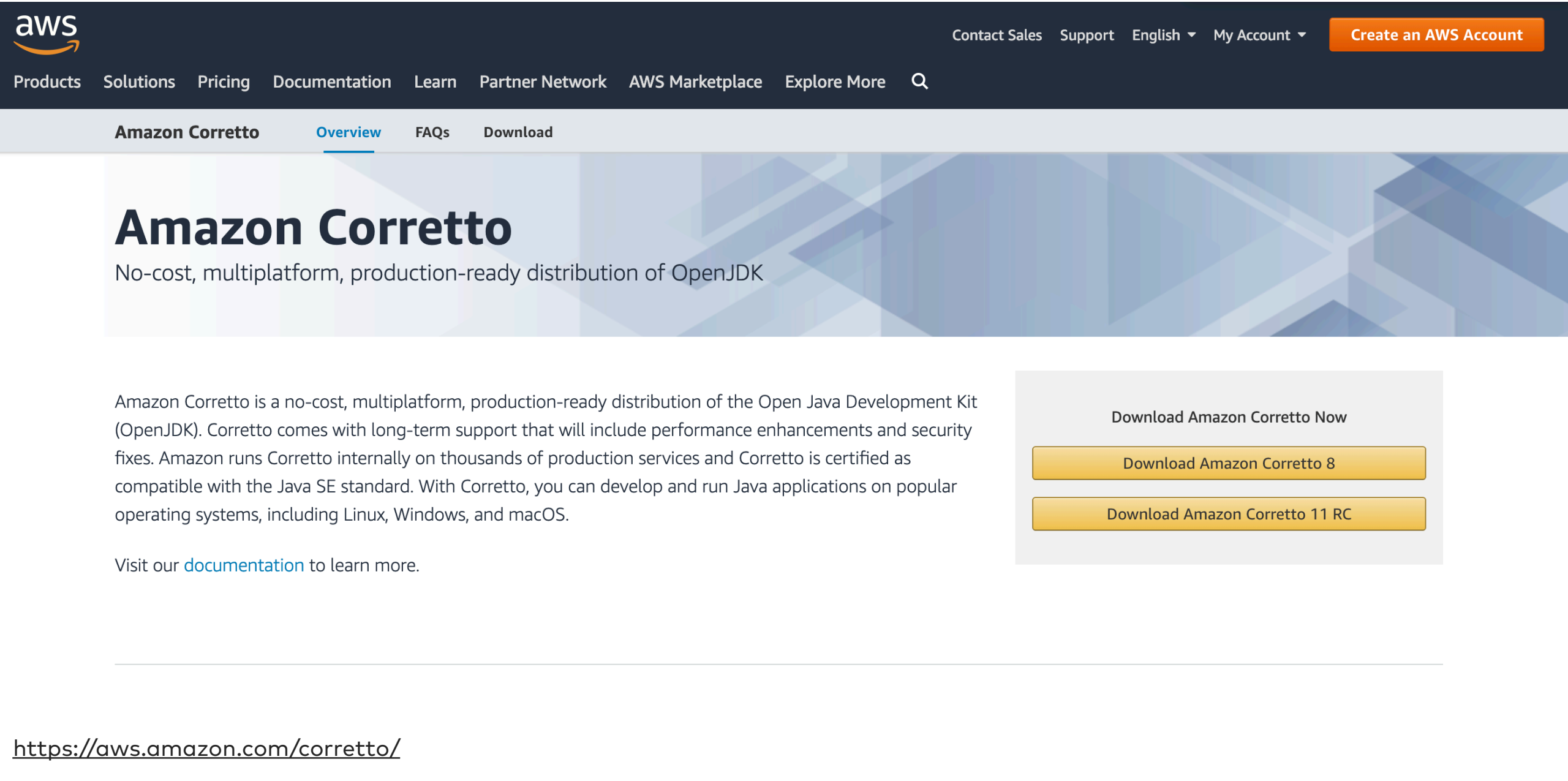
Starting with Java SE 9, in addition to providing Oracle JDK for free under the BCL, Oracle also started providing builds of OpenJDK under an open source license (similar to that of Linux). Oracle is working to make the Oracle JDK and OpenJDK builds from Oracle interchangeable - targeting developers and organizations that do not want commercial support or enterprise management tools. Beginning with Oracle Java SE 11 (18.9 LTS), the Oracle JDK will continue to be available royalty-free for development, testing, prototyping or demonstrating purposes. As announced in September 2017, with the OracleJDK and builds of Oracle OpenJDK being interchangeable for releases of Java SE 11 and later, the Oracle JDK will primarily be for commercial and support customers and OpenJDK builds from Oracle are for those who do not want commercial support or enterprise management tools.

http://www.oracle.com/technetwork/java/eol-135779.html

Prebuilt OpenJDK Binaries

Java™ is the world's leading programming language and platform. The code for Java is open source and available at OpenJDK™. AdoptOpenJDK provides prebuilt OpenJDK binaries from a fully open source set of build scripts and infrastructure.

https://adoptopenjdk.net



https://aws.amazon.com/corretto/

https://www.azul.com/products/zulu-enterprise/

BELLSOFT
Java Platform & Applications Experts

PRODUCTS SERVICES BLOG SUPPORT ABOUT CONTACT US

Download: Liberica JDK

https://bell-sw.com/pages/java-11.0.2/ Liberica JDK version 11.0.2

Release Notes

IBM Runtimes for Business

Access full, unlimited support and updates for your open source Java™ runtime environments

Starting at 46,32 € per AU

View pricing and buy

What does IBM® Runtimes for Business do?

IBM® Runtimes for Business provides commercial support for open source Java™ runtime environments, plus monitoring and management capabilities for Java applications.

IBM support is provided for builds of OpenJDK 8 provided by the AdoptOpenJDK community. It enables you to obtain support for specific servers and clients in your enterprise. Monthly and perpetual licenses are available.

Java application monitoring and management: This offering provides entitlement to the APM Advanced Private server and the J2SE data collector for resource-level monitoring of Java applications.

https://www.ibm.com/de-en/marketplace/support-for-runtimes

Let's talk

jClarity
Performance Tuning via Machine Learning

PRODUCTS SUPPORT COMMUNITY COMPANY 0

OpenJDK for Business

AdoptOpenJDK / Java™ Support and Performance Tuning by the Experts!

Oracle has changed its licensing and subscription model for Oracle's Java. For Java™ 8 or 11+, you will now need to choose one of the following options:

1. Stay on an unsupported/insecure version of Java™
2. Pay Oracle a subscription fee for Oracle JDK / Java™
3. **Use a drop-in OpenJDK™ distribution and get free LTS binaries for 5+ years!**

The jClarity team co-founded **AdoptOpenJDK** which is the **free** ($ and usage) OpenJDK™ distribution replacement for Oracle's JDK, backed by major vendors such as

Rock Solid AdoptOpenJDK / Java™ Support

AdoptOpenJDK

Talk to our Engineers!

Offline - Leave a Message

https://www.jclarity.com/index.php

OpenJDK Life Cycle and Support Policy

Updated March 1 2019 at 8:18 PM - English

Index

Information in this article is subject to change as necessary.

Overview

OpenJDK (Open Java Development Kit) is an open source implementation of the Java Platform, Standard Edition (Java SE). The upstream community project OpenJDK is currently sponsored and led by Oracle and is released under the GNU General Public License (GNU GPL 2 and 2+)

https://access.redhat.com/articles/1299013

View on GitHub

SapMachine

An OpenJDK release maintained and supported by SAP

SapMachine

This project contains a downstream version of the OpenJDK project. It is used to build and maintain a SAP supported version of OpenJDK for SAP customers and partners who wish to use OpenJDK to run their applications.

We want to stress that this is clearly a "*friendly fork*". SAP is committed to ensuring the continued success of the Java platform:

- We are members of the JCP Executive committee since 2001 and recently served in the JSR 379 (Java SE 9), JSR 383 (Java SE 18.3), JSR 384 (Java SE 11), JSR 386 (Java SE 12) and JSR 388 (Java SE 13) Expert Groups.
- SAP is among the biggest external contributors to the OpenJDK project (currently leading the PowerPC/AIX and s390 porting projects).
- We intend to bring as many features as possible into the upstream project and keep the diff of this project as small as possible.

https://sap.github.io/SapMachine/

Java is Still Free (see https://medium.com/@javachampions/java-is-still-free-2-0-0-6b9aa8d6d244)

https://pixabay.com/photos/dove-freedom-bird-nature-animal-2680487/

JDK 10

https://pixabay.com/en/number-ad-yellow-color-asphalt-437928/

OpenJDK

Workshop

OpenJDK FAQ
Installing
Contributing
Sponsoring
Developers' Guide

Mailing lists
IRC · Wiki

Bylaws · Census
Legal

JEP Process

JDK 10

JDK 10 is the open-source reference implementation of the Java SE 10 Platform as defined by JSR 383 in the Java Community Process.

JDK 10 reached General Availability on 20 March 2018. Production-ready binaries under the GPL are available from Oracle; binaries from other vendors will follow shortly.

The features and schedule of this release were proposed and tracked via the JEP Process, as amended by the JEP 2.0 proposal.

http://openjdk.java.net/projects/jdk/10/

var

Cheat Sheet: https://snyk.io/blog/local-type-inference-java-cheat-sheet/
Style Guidelines: http://openjdk.java.net/projects/amber/LVTIstyle.html

JEP286: Local-Variable Type Inference

http://openjdk.java.net/jeps/286

```
[1-9][0-9]*((\.0)*\.[1-9][0-9]*)*
```

JEP322:Time-Based Release Versioning

http://openjdk.java.net/jeps/322

search

Source code
Mercurial
Bundles (6)

Groups
(overview)
2D Graphics
Adoption
AWT
Build
Compatibility & Specification Review
Compiler
Conformance
Core Libraries
Governing Board
HotSpot
Internationalization
JMX
Members

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

http://openjdk.java.net/projects/jdk/10/

More small enhancements

- `@summary`
  https://bugs.openjdk.java.net/browse/JDK-8173425
- 
  https://bugs.openjdk.java.net/browse/JDK-8146115
- `List#copyOf, Map#copyOf, Set#copyOf`
  https://bugs.openjdk.java.net/browse/JDK-8177290
- `Collectors#toUnmodifiableList/Set/Map`
  https://bugs.openjdk.java.net/browse/JDK-8184690
- `Optional::orElseThrow`
  https://bugs.openjdk.java.net/browse/JDK-8140281

JDK 11

https://pixabay.com/en/pool-balls-billiard-eleven-eight-1003955/

OpenJDK

Workshop
OpenJDK FAQ
Installing
Contributing
Sponsoring
Developers' Guide

Mailing lists
IRC · Wiki

Bylaws · Census
Legal

JEP Process

search

Source code
Mercurial

JDK 11

JDK 11 is the open-source reference implementation of version 11 of the Java SE 11 Platform as specified by by JSR 384 in the Java Community Process.

JDK 11 reached General Availability on 25 September 2018. Production-ready binaries under the GPL are available from Oracle; binaries from other vendors will follow shortly.

The features and schedule of this release were proposed and tracked via the JEP Process, as amended by the JEP 2.0 proposal. The release was produced using the JDK Release Process (JEP 3).

Features

http://openjdk.java.net/projects/jdk/11/

```
<jaxb with="jdk11">
   <is provided="false" />
</jaxb>
```

JEP320: Remove the Java EE and CORBA Modules

http://openjdk.java.net/jeps/320

```
HttpClient client = HttpClient.newHttpClient();
HttpRequest request = HttpRequest.newBuilder()
      .uri(URI.create("http://openjdk.java.net/"))
      .build();
client.sendAsync(request, ofString())
      .thenApply(HttpResponse::body)
      .thenAccept(System.out::println)
      .join();
```

http://openjdk.java.net/groups/net/httpclient/intro.html

JEP321: HTTP Client

http://openjdk.java.net/jeps/321

```
(@Nonnull var x) -> process(x)
```

JEP323: Local-Variable Syntax for Lambda Parameters

http://openjdk.java.net/jeps/323

```
javac Foo.java \
  && java Foo \
  && rm Foo.class
```

JEP330: Launch Single-File Source-Code Programs

http://openjdk.java.net/jeps/330

Source code
Mercurial
Bundles (6)

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

Projects
(overview)

Features

181: Nest-Based Access Control
309: Dynamic Class-File Constants
315: Improve Aarch64 Intrinsics
318: Epsilon: A No-Op Garbage Collector
320: Remove the Java EE and CORBA Modules
321: HTTP Client (Standard)
323: Local-Variable Syntax for Lambda Parameters
324: Key Agreement with Curve25519 and Curve448
327: Unicode 10
328: Flight Recorder
329: ChaCha20 and Poly1305 Cryptographic Algorithms
330: Launch Single-File Source-Code Programs
331: Low-Overhead Heap Profiling
332: Transport Layer Security (TLS) 1.3
333: ZGC: A Scalable Low-Latency Garbage Collector (Experimental)
335: Deprecate the Nashorn JavaScript Engine
336: Deprecate the Pack200 Tools and API

Schedule

http://openjdk.java.net/projects/jdk/11/

java.io

- java.io.FileReader(java.lang.String, java.nio.charset.Charset)
- java.io.FileWriter(java.lang.String, java.nio.charset.Charset)

- java.io.InputStream#nullInputStream
- java.io.OutputStream#nullOutputStream

- java.io.Reader#nullReader
- java.io.Writer#nullWriter

java.lang.String

- `String::repeat(int)`
  https://bugs.openjdk.java.net/browse/JDK-8197594
- `String::lines`
  https://bugs.openjdk.java.net/browse/JDK-8200380
- `String::strip`, `String::stripLeading`, `String::stripTrailing`
  https://bugs.openjdk.java.net/browse/JDK-8200377
- `String::isBlank`
  https://bugs.openjdk.java.net/browse/JDK-8200436

java.util.Optional/Predicate

- `Optional::isEmpty`
  https://bugs.openjdk.java.net/browse/JDK-8184693
- `Predicate#not`
  https://bugs.openjdk.java.net/browse/JDK-8050818

java.nio.file.Files/Path

- `Files#isSameContent`
  https://bugs.openjdk.java.net/browse/JDK-8202302
- `Files#readString, Files#writeString`
  https://bugs.openjdk.java.net/browse/JDK-8202055
- `Path#of(URI), Path#of(String, String…)`
  https://bugs.openjdk.java.net/browse/JDK-8199485

java.util.regex.Pattern

- `Pattern::asMatchPredicate`
  https://bugs.openjdk.java.net/browse/JDK-8201308

java.lang.Thread

- `Thread::stop(Throwable), Thread::destroy`
  https://bugs.openjdk.java.net/browse/JDK-8204243

JDK 12

https://pixabay.com/en/gate-wrought-iron-iron-number-3267016/

OpenJDK

Workshop
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
Compatibility & Specification Review
Compiler

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

http://openjdk.java.net/projects/jdk/12/

```
int numLetters = switch (day) {
    case MONDAY, FRIDAY, SUNDAY -> 6;
    case TUESDAY -> 7;
    case THURSDAY, SATURDAY -> 8;
    case WEDNESDAY -> 9;
};
```

JEP325: Switch Expressions (Preview)

http://openjdk.java.net/jeps/325

Summary

Enhance the G1 garbage collector to automatically return Java heap memory to the operating system when idle.

Non-Goals

- Sharing of committed but empty pages between Java processes. Memory should be returned (uncommitted) to the operating system.
- The process of giving back memory does not need to be frugal with CPU resources, nor does it need to be instantaneous.
- Use of different methods to return memory other than available uncommit of memory.
- Support for other collectors than G1.

JEP 346: Promptly Return Unused Committed Memory from G1

http://openjdk.java.net/jeps/346

Conformance
Core Libraries
Governing Board
HotSpot
Internationalization
JMX
Members
Networking
NetBeans Projects
Porters
Quality
Security
Serviceability
Sound
Swing
Vulnerability
Web

Projects
(overview)
Amber
Annotations Pipeline 2.0

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

Last update: 2019/2/25 17:56 UTC

http://openjdk.java.net/projects/jdk/12/

java.lang.String

- ~~`String::align`~~
  https://bugs.openjdk.java.net/browse/JDK-8215490
- `String::indent(int)`
  https://bugs.openjdk.java.net/browse/JDK-8200435
- `String::transform(Function)`
  https://bugs.openjdk.java.net/browse/JDK-8203703

API Changes

- `{@systemProperty}`
  https://bugs.openjdk.java.net/browse/JDK-5076751
- `Files#mismatch(Path, Path)`
  https://bugs.openjdk.java.net/browse/JDK-8202302
- `Collectors#teeing(Collector, Collector, BiFunction)`
  https://bugs.openjdk.java.net/browse/JDK-8209685

JDK 13

https://pixabay.com/photos/house-number-13-house-number-13-3143265/

OpenJDK

Workshop
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
Compatibility &
Specification

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

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

Last update: 2019/3/26 18:54 UTC

https://openjdk.java.net/projects/jdk/13/

Future

https://pixabay.com/en/binoculars-looking-man-discovery-1209011/

Project Amber

https://pixabay.com/en/heart-amber-pendant-necklace-1202129/

Project Amber

The goal of Project Amber is to explore and incubate smaller, productivity-oriented Java language features that have been accepted as candidate JEPs under the OpenJDK JEP process. This Project is sponsored by the Compiler Group.

Status of JEPs

Currently in progress:

- JEP 305 Pattern Matching
- JEP 325 Switch Expressions (preview, JDK 12)
- JEP 326 Raw String Literals
- JEP 348 Java Compiler Intrinsics for JDK APIs

Delivered:

- JEP 286 Local-Variable Type Inference (`var`) *(JDK 10)*
  See also:
  - Style Guidelines
  - Frequently Asked Questions
- JEP 323 Local-Variable Syntax for Lambda Parameters *(JDK 11)*

On hold:

- JEP 301 Enhanced Enums. See explanation.

Documents

- Data Classes for Java *(February 2018)*
- tyle Guidelines for Local Variable Type Inference *(March 2018)*
- Local Variable Type Inference: Frequently Asked Questions *(October 2018)*

```
if (obj instanceof String s) {
    // can use s here
} else {
    // can't use s here
}

@Override
public boolean equals(Object o) {
    return (o instanceof CaseInsensitiveString cis) &&
        cis.s.equalsIgnoreCase(s);
}
```

JEP305: Pattern Matching for instanceof (Preview)

http://openjdk.java.net/jeps/305

```
String formatted;
switch (obj) {
    case Integer i: formatted = String.format("int %d", i); break;
    case Byte b:    formatted = String.format("byte %d", b); break;
    case Long l:    formatted = String.format("long %d", l); break;
    case Double d:  formatted = String.format("double %f", l); break;
    case String s:  formatted = String.format("String %s", s); break;
    default:        formatted = obj.toString();
}
```

JEP draft 8213076: Pattern matching for switch (Preview)

https://openjdk.java.net/jeps/8213076

JEP 354: Switch Expressions

Author	Gavin Bierman
Owner	Jan Lahoda
Type	Feature
Scope	SE
Status	Candidate
Component	tools / javac
Discussion	amber dash dev at openjdk dot java dot net
Effort	S
Duration	M
Relates to	JEP 325: Switch Expressions (Preview)
Reviewed by	Alex Buckley, Brian Goetz
Created	2019/04/09 12:38
Updated	2019/05/03 11:11
Issue	8222184

Summary

Extend `switch` so it can be used as either a statement or an expression, and so that both forms can use either traditional `case ... :` labels (with fall through) or new `case ... ->` labels (with no fall through), with a further new statement for yielding a value from a `switch` expression. These changes will simplify everyday coding, and prepare the way for the use of pattern matching (JEP 305) in `switch`.

History

Switch expressions were announced in December 2017 in JEP 325. They were targeted to JDK 12 in August 2018 as a preview feature. Feedback was sought initially on the design and later on the experience of using the enhancements to `switch`. Based on that feedback, this JEP makes [illegible] in a new version:

> To yield a value from a `switch` expression, the `break` with value statement will be dropped in favor of a `break-with` statement. (`break-with` is a hyphenated keyword.)

JEP354: Switch Expressions

http://openjdk.java.net/jeps/354

```
String html = """
              <html>
                  <body>
                      <p>Hello, world</p>
                  </body>
              </html>
              """;
```

JEP draft 8222530: Text Blocks (Preview)

https://openjdk.java.net/jeps/8222530

```
//   -> is "single expression form"
int length(String s) -> s.length();

//   = is "method reference form"
int length(String s) = String::length;
```

JEP draft 8209434: Concise Method Bodies

http://openjdk.java.net/jeps/8209434

Project Loom

https://pixabay.com/en/craftsman-loom-craftsmanship-hands-1839920/

OpenJDK

Workshop

OpenJDK FAQ
Installing
Contributing
Sponsoring
Developers' Guide

Mailing lists
IRC · Wiki

Bylaws · Census

Loom

The goal of this Project is to explore and incubate Java VM features and APIs built on top of them for the implementation of lightweight user-mode threads (fibers), delimited continuations (of some form), and related features, such as explicit tail-call.

This Project is sponsored by the HotSpot Group.

http://openjdk.java.net/projects/loom/

Project Valhalla

https://pixabay.com/en/tunnel-passage-underpass-light-965720/

OpenJDK

Workshop

OpenJDK FAQ
Installing
Contributing
Sponsoring
Developers' Guide

Mailing lists
IRC · Wiki

Bylaws · Census
Legal

JEP Process

Valhalla

NOTE: See the **OpenJDK Wiki** for details and up-to-date information.

The goal of this Project is to provide a venue to explore and incubate advanced Java VM and Language feature candidates such as:

- Value Types
- Generic Specialization
- And possibly other related topics

This Project is sponsored by the HotSpot Group.

http://openjdk.java.net/projects/valhalla/

Summary

Provide JVM infrastructure for working with immutable and reference-free objects, in support of efficient by-value computation with non-primitive types.

JEP169: Value Objects

http://openjdk.java.net/jeps/169

```
class Box<T> {
}

new Box<int>();
```

JEP218: Generics over Primitive Types

http://openjdk.java.net/jeps/218

```
record Point(int x, int y) { }
```

JEP draft 8222777: Records and Sealed Types

https://openjdk.java.net/jeps/8222777

JEP draft: Keyword Management for the Java Language

Owner	Alex Buckley
Type	Informational
Scope	SE
Status	Draft
Component	specification / language
Discussion	jdk dash dev at openjdk dot java dot net
Effort	M
Duration	M
Created	2019/04/26 00:26
Updated	2019/05/07 17:32
Issue	8223002

Summary

Evolving the Java language often means new keywords for new features, but new keywords risk breaking existing programs. To balance compatibility and readability, a new kind of keyword may be used: a *hyphenated keyword* that is a compound of pre-existing keywords and identifiers, such as `non-final`, `break-with`, and `short-circuit`.

Note: All examples in this JEP are intended solely to illustrate a syntactic form under discussion. They are not intended to suggest that any particular language feature is being considered for inclusion in Java now or in the future.

Goals

- Explore the syntactic options open to Java language designers for denoting new features.
- Solve the perpetual problem of keyword tokens being so scarce and expensive that language designers have to constrain or corrupt the Java programming model to fit the keywords available.
- Keep the [illegible] of features.

Non-Goals

- In any proposal for new elements of Java syntax, it is important to avoid being influenced by the (often strawman) syntax of language features

JEP draft 8223002: Keyword Management for the Java Language

https://openjdk.java.net/jeps/8223002

Summary

Introduce a modernized general-purpose hash function into the JVM, usable by arbitrary classes and arrays.

Enable new classes to use it as a back-end for the standard `Object.hashCode` API.

Supply APIs for legacy types (including arrays and collections) to begin to make use of this hash code.

JEP draft 8201462: Better hash codes

http://openjdk.java.net/jeps/8201462

```
private final static Logger LOGGER =
    Logger.getLogger("com.foo.Bar");
```

JEP draft 8209964: Lazy Static Final Fields

http://openjdk.java.net/jeps/8209964

Thanks! Questions?

INNOQ
www.innoq.com

https://www.innoq.com/en/talks/2019/05/jugch-java-10-11-12/

Michael Vitz
michael.vitz@innoq.com

+49 151 19116015
michaelvitz

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
+49 2173 3366-0

Ohlauer Str. 43
10999 Berlin
Germany
+49 2173 3366-0

Ludwigstr. 180E
63067 Offenbach
Germany
+49 2173 3366-0

Kreuzstr. 16
80331 München
Germany
+49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
+41 41 743 0116