



ACADEMIK

De Java 8 a Java 14

Víctor Orozco - @tuxtor

27 de enero de 2020

GuateJUG



¿Java ya no es gratis/libre?

—

¿Que es Java?

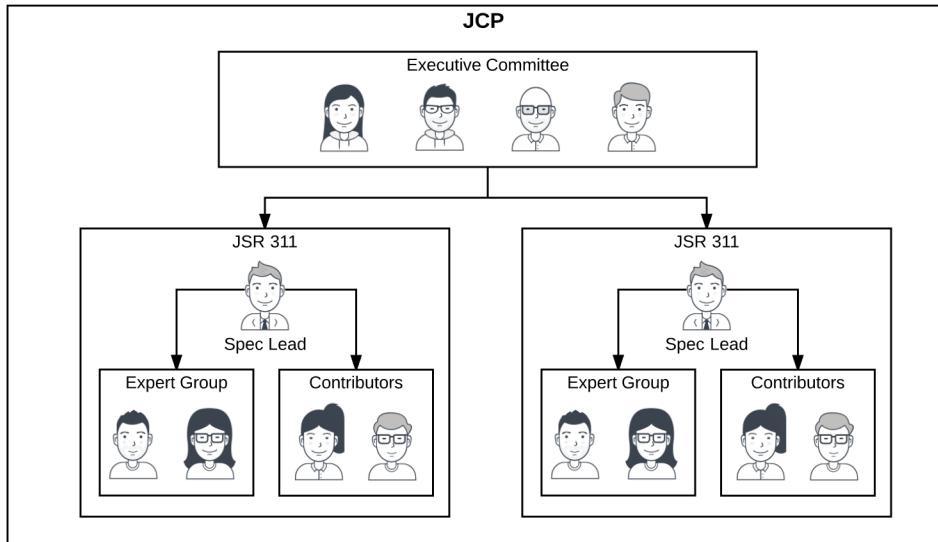
- Lenguaje de programación
- Maquina virtual
- Bibliotecas/API

Todas conforman la plataforma Java(TM)

¿Como se hace Java?

- JCP - Java Community Process
- JSR - Java Specification Request
- JEP - Java Enhancement Proposal
- JCK - Java Compatibility Kit

¿Como se hace Java? - Java Specification Request



¿Como se hace Java? - Java Enhancement Proposal



Workshop

[OpenJDK FAQ](#)

[Installing](#)

[Contributing](#)

[Sponsoring](#)

[Developers' Guide](#)

[Vulnerabilities](#)

[Mailing lists](#)

[IRC · Wiki](#)

[Bylaws · Census](#)

[Legal](#)

JEP Process

Source code

[Mercurial](#)

[Bundles \(6\)](#)

Groups

[\(overview\)](#)

[2D Graphics](#)

[Adoption](#)

[AWT](#)

[Build](#)

[Compatibility &](#)

[Specification](#)

[Review](#)

[Compiler](#)

[Conformance](#)

[Core Libraries](#)

[Governing Board](#)

[HotSpot](#)

JEP 126: Lambda Expressions & Virtual Extension Methods

Author [Joseph D. Darcy](#)

Owner [Brian Goetz](#)

Type [Feature](#)

Scope [SE](#)

Status [Closed / Delivered](#)

Release [8](#)

Component [tools / javac](#)

JSRs [269 MR](#), [335](#)

Discussion [lambda dash dev at openjdk dot java dot net](#)

Effort [XL](#)

Duration [XL](#)

Blocks [JEP 101: Generalized Target-Type Inference](#)
[JEP 107: Bulk Data Operations for Collections](#)
[JEP 109: Enhance Core Libraries with Lambda](#)
[JEP 155: Concurrency Updates](#)

Reviewed by [Brian Goetz](#)

Endorsed by [Brian Goetz](#)

Created [2011/11/01 20:00](#)

Updated [2015/01/09 17:52](#)

Issue [8046116](#)

¿Como se hace Java? - Java Compatibility Kit

Oracle Accessibility Conformance Report

VPAT® Version 2.0 - October 2017

Name of Product: Java Compatibility Kit 11

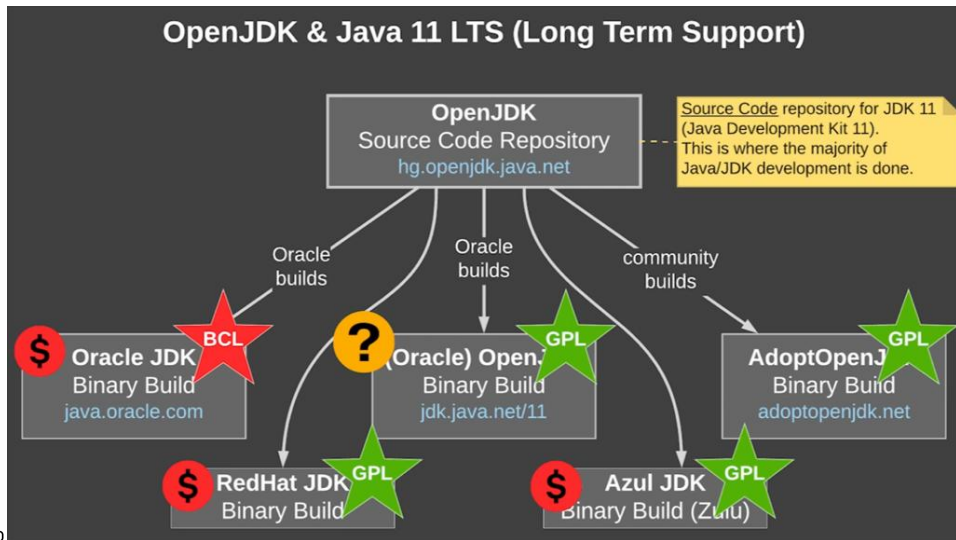
Product Description:

Java Compatibility Kit 11 (JCK 11) is a collection of tests that is used to ensure compatible behavior between implementations of the Java SE (Standard Edition) 11 platform. The tests in JCK 11 are derived from statements in the specification of the Java SE 11 platform.

JCK 11 can be run from the command line or from the JavaTest test harness. Some of the tests in JCK 11 are interactive, and present a Graphical User Interface (GUI) that the user must interact with. Interactions with the GUI are excluded from this VPAT.

Due to the JCK having to meet the criteria specified in the JSRs (Java Specification Requests), it is not possible to meet all accessibility criteria as they are in conflict with the JSR. To meet the criteria would require a fundamental alteration of the product.

¿Como se hace Java? - Java Builds



¿Java ya no es gratis/libre?

Java **es gratis y libre.**

Algunas empresas cobran por soporte en su "versión" de Java.



ACADEMIK

De Java 8 a Java 14



¿Una nueva versión de Java?

- Java - Lenguaje de programación
- Java - La plataforma (Bibliotecas y APIs)
- Java - La máquina virtual

Java - Mejoras importantes

- Java 9
 - Modulos
 - JShell
 - HTTP/2
- Java 10
 - Inferencia de tipos
 - Class Data Sharing
 - Time based release
- Java 11
 - String methods
 - File methods
 - Ejecución directa de .java
- Java 12
 - Switch expressions
- Java 13
 - Text blocks (preview)
- Java 14
 - Pattern matching
 - Records
 - Helpfull NPE messages

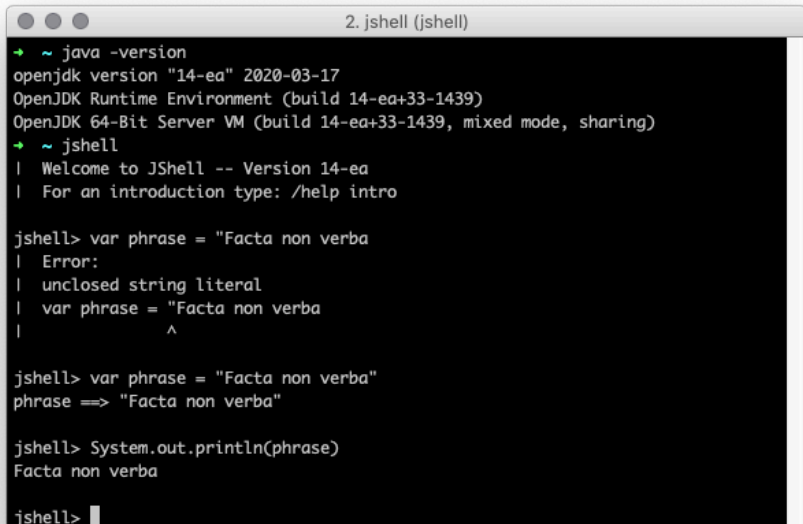


ACADEMIK

Java 9



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



```
2. jshell (jshell)
→ ~ java -version
openjdk version "14-ea" 2020-03-17
OpenJDK Runtime Environment (build 14-ea+33-1439)
OpenJDK 64-Bit Server VM (build 14-ea+33-1439, mixed mode, sharing)
→ ~ jshell
| Welcome to JShell -- Version 14-ea
| For an introduction type: /help intro

jshell> var phrase = "Facta non verba
| Error:
| unclosed string literal
| var phrase = "Facta non verba
|                               ^

jshell> var phrase = "Facta non verba"
phrase ==> "Facta non verba"

jshell> System.out.println(phrase)
Facta non verba

jshell> |
```

JEP 110: HTTP/2 Client

```
1 HttpRequest request = HttpRequest.newBuilder()
2   .uri(new URI("https://swapi.co/api/starships/9"))
3   .GET()
4   .build();
5
6 HttpResponse<String> response = HttpClient.newHttpClient()
7   .send(request, BodyHandlers.ofString());
8
9 System.out.println(response.body());
```

JEP 269: Convenience Factory Methods for Collections

Antes

```
1 Set<String> set = new HashSet<>();  
2 set.add("a");  
3 set.add("b");  
4 set.add("c");  
5 set = Collections.unmodifiableSet(set);
```

"Pro"

```
1 Set<String> set = Collections.unmodifiableSet(new HashSet  
    <>(Arrays.asList("a", "b", "c")));
```

Ahora

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


JEP 213: Milling Project Coin - Private methods in interfaces

Antes

```
1 public interface Vehicle{
2     public void move();
3 }
```

Ahora

```
1 public interface Vehicle{
2     public default void makeNoise() {
3         System.out.println("Making noise!");
4         createNoise();
5     }
6
7     private void createNoise() {
8         System.out.println("Run run");
9     }
10 }
```

JEP 213: Milling Project Coin - Try-with-resources

Antes

```
1 | BufferedReader reader = new BufferedReader(new FileReader("
   |     langs.txt"));
2 |
3 | try(BufferedReader innerReader = reader){
4 |     System.out.println(reader.readLine());
5 | }
```

Ahora

```
1 | BufferedReader reader = new BufferedReader(new FileReader("
   |     langs.txt"));
2 |
3 | try(reader){
4 |     System.out.println(reader.readLine());
5 | }
```



ACADEMIK

Java 10



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

JEP 286: Local-Variable Type Inference

```
1 public static void main(String args[]){  
2     var localValue = 99;  
3     System.out.println(++localValue);  
4     //localValue = "Foo"  
5 }
```

JEP 310: Application Class-Data Sharing

```
java -XX:ArchiveClassesAtExit=app-cs.jsa -jar payara-micro-5.192.jar  
java -XX:SharedArchiveFile=app-cs.jsa -jar fpjava.jar
```

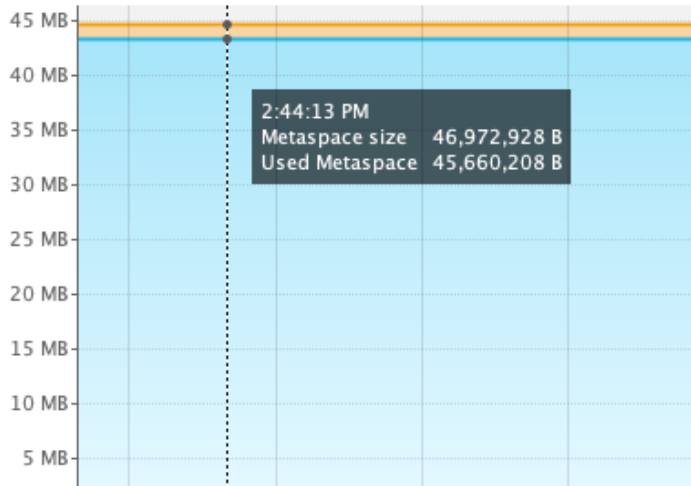
1
2

JEP 310: Application Class-Data Sharing

Size: 46,972,928 B

Used: 45,660,208 B

Max: 1,082,130,432 B

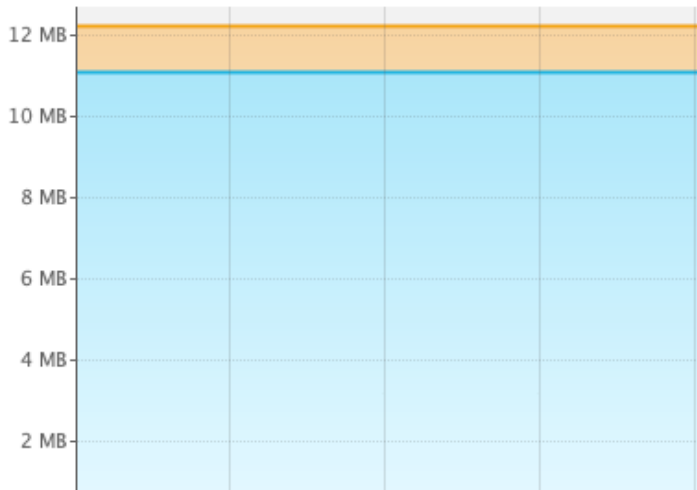


JEP 310: Application Class-Data Sharing

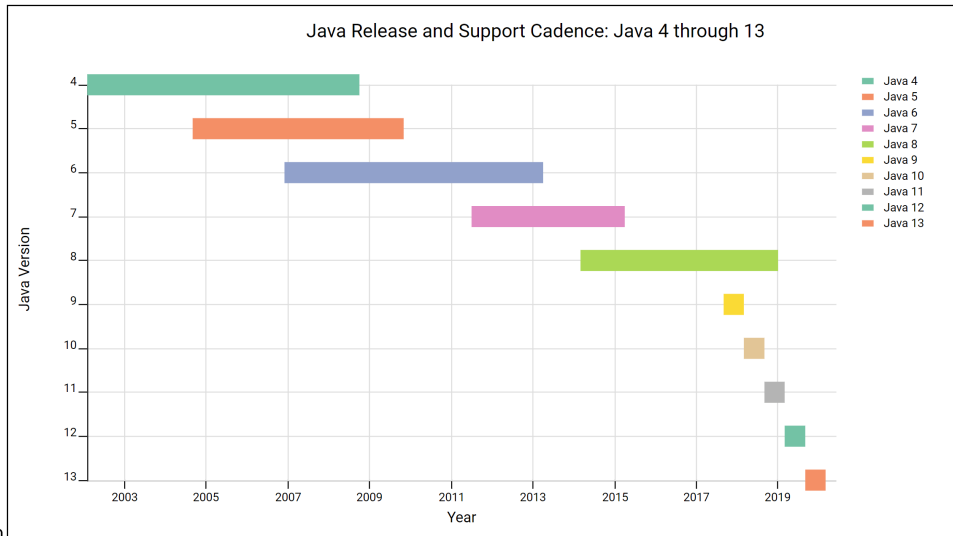
Size: 12,845,056 B

Used: 11,672,656 B

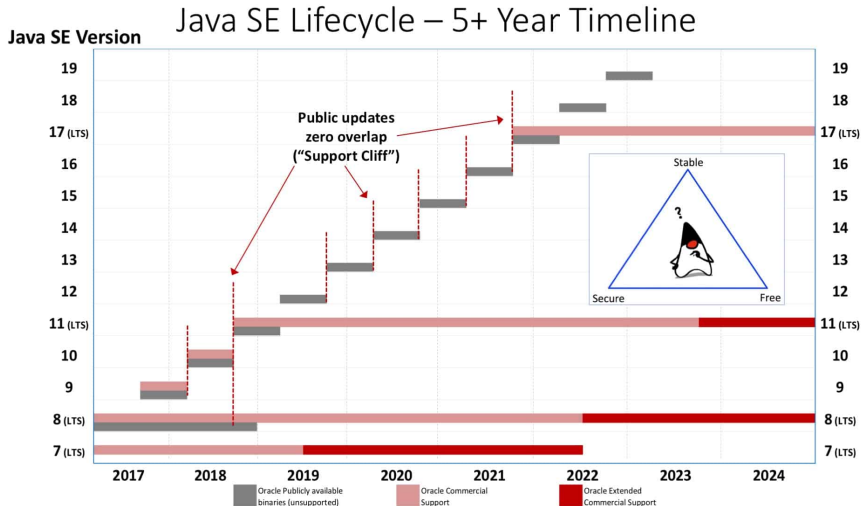
Max: 1,082,130,432 B



JEP 322: Time-Based Release Versioning



JEP 322: Time-Based Release Versioning





ACADEMIK

Java 11



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

JEP 323: Local-Variable Syntax for Lambda Parameters

Antes

```
1 BiPredicate<String,String> demoPredicate =  
2 (String a, String b) -> a.equals(b);  
3 BiPredicate<String,String> demoPredicate =  
4 (a, b) -> a.equals(b);
```

Ahora

```
1 BiPredicate<String,String> demoPredicate =  
2 (var a, var b) -> a.equals(b);
```

Posibilidades

```
1 (@NonNull var x, @Nullable var y) -> x.process(y)
```

JEP 330: Launch Single-File Source-Code Programs

```
2. tuxtor@millenium-falcon-2: ~/Sandbox/JavaTrain/fileexecution (zsh)
→ fileexecution echo "public class HelloWorld{
    public static void main(String args[]){
        System.out.println(\"Hello world\");
    }
}" > HelloWorld.java
→ fileexecution java HelloWorld.java
Hello world
→ fileexecution ls
HelloWorld.java
→ fileexecution
```



ACADEMIK

Java 12



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

325: Switch Expressions (Preview)

Antes

```
1 String langType = "";
2 switch (args[0]) {
3     case "Java":
4     case "Scala":
5     case "Kotlin":
6         langType = "Static typed";
7         break;
8     case "Groovy":
9     case "JavaScript":
10        langType = "Dynamic typed";
11        break;
12 }
13 System.out.println(langType);
```

325: Switch Expressions (Preview)

Ahora

```
1 String langType = switch (args[0]) {  
2     case "Java", "Scala", "Kotlin" -> "Static typed";  
3     case "Groovy", "JavaScript" -> "Dynamic typed";  
4     default -> {  
5         System.out.println("This meant to be a processing  
6             block");  
7         yield "Probably LISP :)";  
8     }  
9 };  
10 System.out.println(langType);
```



ACADEMIK

Java 13



350: Dynamic CDS Archives

351: ZGC: Uncommit Unused Memory

353: Reimplement the Legacy Socket API

354: **Switch Expressions (Preview)**

355: **Text Blocks (Preview)**



355: Text Blocks (Preview)

Antes

```
1 String html = "<html>\n" +  
2 "    <body>\n" +  
3 "        <p>Hello, world</p>\n" +  
4 "    </body>\n" +  
5 "</html>\n";
```

Ahora

```
1 String html = """  
2 <html>  
3     <body>  
4         <p>Hello, world</p>  
5     </body>  
6 </html>  
7 """;
```



ACADEMIK

Java 14



305: Pattern Matching for instanceof (Preview)

343: Packaging Tool (Incubator)

345: NUMA-Aware Memory Allocation for G1

349: JFR Event Streaming

352: Non-Volatile Mapped Byte Buffers

358: Helpful NullPointerExceptions

359: Records (Preview)

361: Switch Expressions (Standard)

362: Deprecate the Solaris and SPARC Ports

363: Remove the Concurrent Mark Sweep (CMS)
Garbage Collector

364: ZGC on macOS

365: ZGC on Windows

366: Deprecate the ParallelScavenge + SerialOld
GC Combination

367: Remove the Pack200 Tools and API

368: Text Blocks (Second Preview)

370: Foreign-Memory Access API (Incubator)

JEP 359: Records (Preview)

Data carrier

```
1 record Person(String name, String email, int age) {}
```

Use

```
1 Person foo = new Person("Marco", "example@mail.com", 99);  
2 System.out.println(foo);  
3 //foo.name = "Polo";
```


305: Pattern Matching for instanceof (Preview)

Antes

```
1  if(o instanceof Person){
2      Person p = (Person)o;
3      System.out.println("Hello " + p.name());
4  }else{
5      System.out.println("Unknown object");
6  }
```

Ahora

```
1  if(o instanceof Person p){
2      System.out.println("Hello " + p.name());
3  }else{
4      System.out.println("Unknown object");
5  }
```



Oracle
Groundbreakers



ORACLE®
Certified Professional
Java SE 8 Programmer

ORACLE®
Certified Associate
Java SE 8 Programmer

- vorozco@nabenik.com
- @tuxtor
- <http://voroazco.com>
- <http://tuxtor.shekalug.org>



This work is licensed under
Creative Commons Attribution-
NonCommercial-ShareAlike 3.0
Guatemala (CC BY-NC-SA 3.0 GT).



ACADEMIK

Esríbenos a cursos@academik.io

www.academik.io

(CC BY-NC-SA3.0 GT)