De Java 8 a Java 14

Víctor Orozco - @tuxtor

30 de enero de 2020

Academik

ACADEMIK

¿Java ya no es gratis/libre?

De Java 8 a Java 14

Java 9

Java 10

Java 11

Java 12

Java 13

Java 14

Mundo real

9

10 13

11 14

12

@tuxtor

¿Java ya no es gratis/libre?

¿Que es Java?

- Lenguaje de programación
- Maquina virtual
- Bibliotecas/API

Todas conforman la plataforma Java

¿Que es Java?

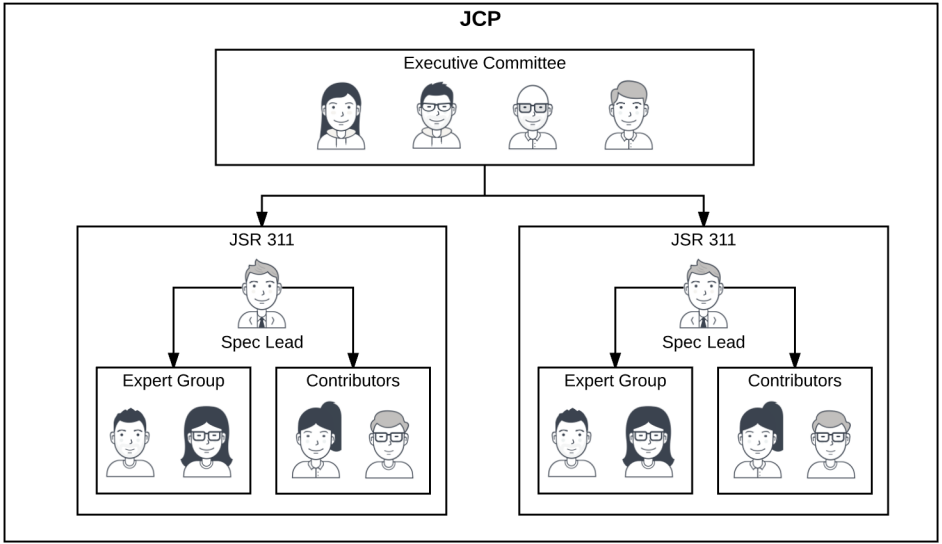
- Lenguaje de programación
- Maquina virtual
- Bibliotecas/API

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

OpenJDK

Workshop
OpenJDK FAQ
Installing
Contributing
Sponsoring
Developers' Guide
Vulnerabilities

Mailing lists
IRC · Wiki

Bylaws · Census
Legal

JEP Process

search

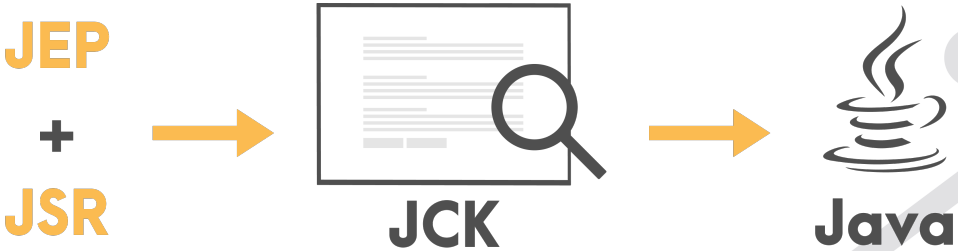
Source code
Mercurial
Bundles (6)

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

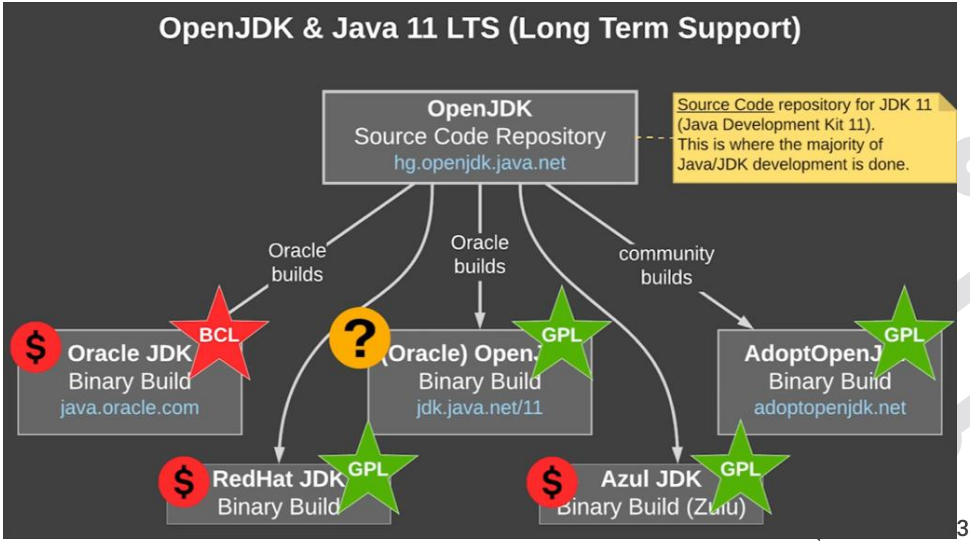
JEP 126: Lambda Expressions & Virtual Extension Methods

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



¿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
  - Factory methods
- 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
- Java 14
  - Pattern matching
  - Records
  - Helpfull NPE

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

```
HttpRequest request = HttpRequest.newBuilder()
    .uri(new URI("https://swapi.co/api/starships/9"))
    .GET()
    .build();

HttpResponse<String> response = HttpClient.newHttpClient()
    .send(request, BodyHandlers.ofString());

System.out.println(response.body());
```

JEP 269: Convenience Factory Methods for Collections

Antes

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

”Pro”

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

Ahora

```
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

```
BufferedReader reader = new BufferedReader(new FileReader("
    langs.txt"));

try(BufferedReader innerReader = reader){
    System.out.println(reader.readLine());
}
```

Ahora

```
BufferedReader reader = new BufferedReader(new FileReader("
    langs.txt"));

try(reader){
    System.out.println(reader.readLine());
}
```

ACADEMIK

Java 10

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

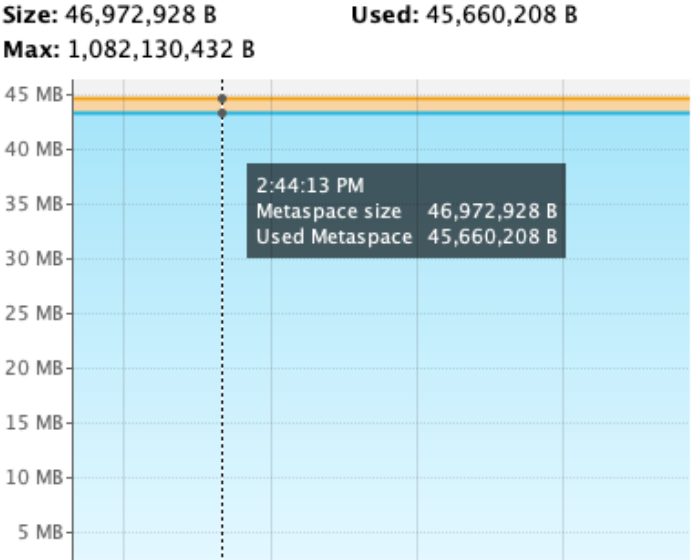
JEP 286: Local-Variable Type Inference

```
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

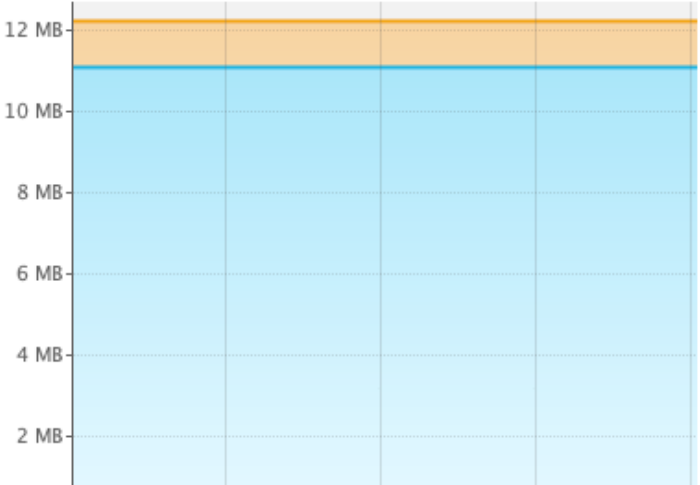
```
java -XX:ArchiveClassesAtExit=app-cs.jsa -jar payara-micro-5.192.jar
java -XX:SharedArchiveFile=app-cs.jsa -jar fpjava.jar
```

JEP 310: Application Class-Data Sharing

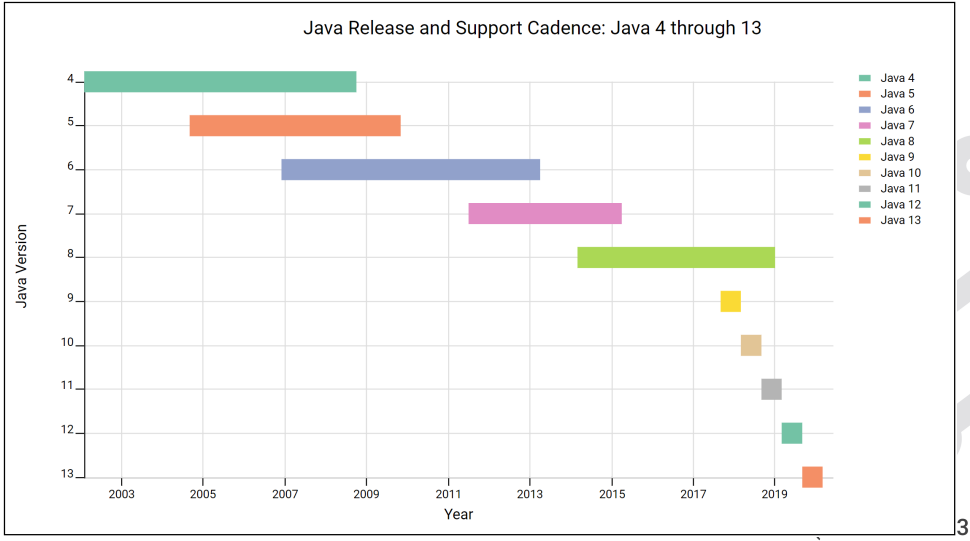


JEP 310: Application Class-Data Sharing

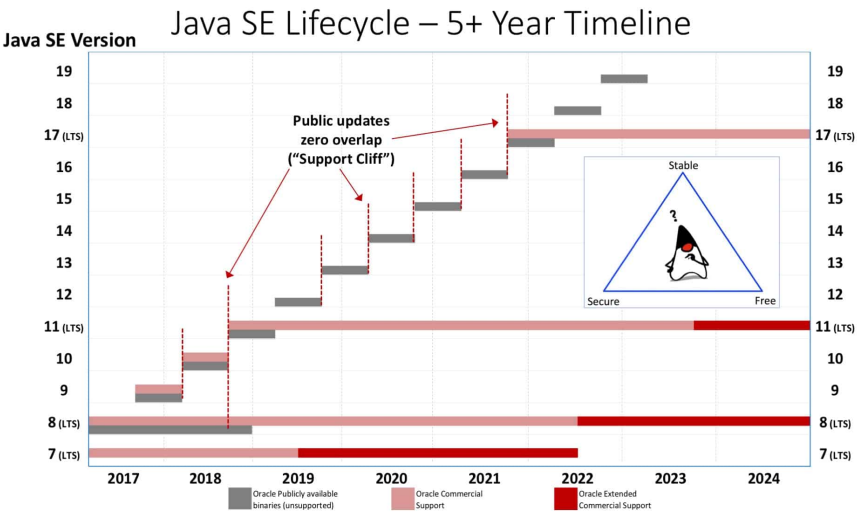
Size: 12,845,056 B **Used:** 11,672,656 B
Max: 1,082,130,432 B



JEP 322: Time-Based Release Versioning



JEP 322: Time-Based Release Versioning



ACADEMIK

Java 11

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

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

Ahora

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

Posibilidades

```
(@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

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

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

ACADEMIK

Java 13

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

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

Ahora

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

ACADEMIK

Java 14

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) {}
```

Uso

```
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

```
if(o instanceof Person){
    Person p = (Person)o;
    System.out.println("Hello " + p.name());
}else{
    System.out.println("Unknown object");
}
```

Ahora

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

ACADEMIK

Mundo real

Mundo real

Mi mundo real

- ERP - 10 modulos (1 EAR, 9 EJB, 1 WAR), JBoss/Wildfly
- Venta/Geocerca (5 WAR) Payara Application Server
- POS - JavaFX y Windows D:

Los dolores de cabeza

- Modulos
- sun.misc.unsafe
- Corba y Java EE
- JavaFX
- IDE
- Licencia

Mundo real

Los dolores de cabeza

- Modulos
- sun.misc.unsafe
- Corba y Java EE
- JavaFX
- IDE
- Licencia

Estrategia

1. Verificar la compatibilidad del runtime/servidor/framework compatible
2. Multiples JVM con cambio fácil en desarrollo
3. Actualizar el compilador en Maven
4. Actualizar bibliotecas
5. Incluir los modulos corba y Java EE en el war
6. Actualizar el IDE
7. Prepara el proyecto para enlazar el modulo de JavaFX
8. Verificar que Java necesito
9. Multiples JVM en producción

Compatibilidad runtime

Compatible con Java 11

- Tomcat
- Spring
- Micronaut
- Vert.x
- JakartaEE (JBoss/Wildfly, OpenLiberty, Payara)

Multiples JVMs

jEnv

Manage your Java environment

SDKMAN!

Bibliotecas

Manipulación de bytecode

- ByteBuddy
- ASM
- glib
- Spring
- Java EE
- Hibernate
- Mockito

Maven

- Maven 3.5.0
- Compiler 3.8.0
- surefire 2.22.0
- failsafe 2.22.0
- release version 11.0

Maven - JavaEE

JAF (java.activation)

```
<dependency>
    <groupId>com.sun.activation</groupId>
    <artifactId>javax.activation</artifactId>
    <version>1.2.0</version>
</dependency>
```

CORBA = RIP

JAXB (java.xml.bind)

```
<!-- API -->
<dependency>
    <groupId>jakarta.xml.bind</groupId>
    <artifactId>jakarta.xml.bind-api</artifactId>
    <version>2.3.2</version>
</dependency>

<!-- Runtime -->
<dependency>
    <groupId>org.glassfish.jaxb</groupId>
    <artifactId>jaxb-runtime</artifactId>
    <version>2.3.2</version>
</dependency>
```

Maven - JavaEE

JAX-WS (java.xml.ws)

```
<!-- API -->
<dependency>
    <groupId>jakarta.xml.ws</groupId>
    <artifactId>jakarta.xml.ws-api</artifactId>
    <version>2.3.2</version>
</dependency>

<!-- Runtime -->
<dependency>
    <groupId>com.sun.xml.ws</groupId>
    <artifactId>jaxws-rt</artifactId>
    <version>2.3.2</version>
</dependency>
```

Maven - JavaEE

Common Annotations (java.xml.ws.annotation)

```
<dependency>
    <groupId>javax.annotation</groupId>
    <artifactId>javax.annotation-api</artifactId>
    <version>1.3.1</version>
</dependency>
```

IDEs

Compatibles con Java 11

- Eclipse
- NetBeans
- IntelliJ IDEA

Algunos plugins problematicos

1. Glassfish
2. WebLogic
3. Icefaces

JavaFX ahora es un modulo OpenSource y la forma recomendada”para empaquetar la aplicación es JPMS, la forma más facil es la compilación de Gluon

Home » Products » JavaFX

JavaFX

¿Que Java necesito?

Obligatorios por contrato

- Software comercial de Oracle (HotSpot)
- Software comercial de SAP (SAP VM)
- Software comercial de Red Hat (OpenJDK + RHEL)
- Software comercial de IBM (J9)

Otras opciones

- AdoptOpenJDK (Opción a soporte de IBM en J9)
- Correto
- Azul Zulu
- Java de su distro

Multiples JVMs en producción

Linux

- Docker
- RHEL
- Debian
- Gentoo

Windows

- Docker
- Variables de entorno por proyecto/runtime
- Lo importante es la salud

Academik

Víctor Orozco

Oracle Groundbreakers

Java Champions

ORACLE Certified Professional Java SE 8 Programmer

ORACLE Certified Associate Java SE 8 Programmer

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

ACADEMIK

Escríbenos a cursos@academik.io

www.academik.io