

Evolución de Java

¿Qué es java?

Java es un lenguaje con características de programación orientado a objetos y procedurales, fue creado por James Gosling, Mike Sheridan y Patrick Naughton en Sun Microsystems en la década de 1990. Fue diseñado con la intención de ser una plataforma independiente, lo que significa que las aplicaciones escritas en Java podrían ejecutarse en cualquier sistema que tuviera una Máquina Virtual Java (JVM).

Java ha experimentado una evolución constante a lo largo de las décadas, pasando de un lenguaje inicialmente simple a una plataforma de desarrollo poderosa y versátil. La versión 8 marcó un cambio significativo con la introducción de características modernas que han influido en la forma en que se escriben aplicaciones Java. A continuación se muestran características importantes de las versiones de java a lo largo del tiempo

Ediciones y versiones de Java

La versión inicial de Java, conocida como Java 1.0, se lanzó en 1996. Esta versión introdujo características básicas, como la gestión de objetos y la capacidad de ejecutar aplicaciones en múltiples plataformas. La verdadera innovación de Java radicaba en su capacidad para proporcionar portabilidad a través de la JVM.

Java 1.1, lanzada en 1997, trajo consigo mejoras significativas en rendimiento y estabilidad. También se introdujeron las "clases internas" en esta versión, lo que permitió una mejor organización del código.

Java 1.2, o Java 2, lanzada en 1998, fue un gran salto en la evolución de Java. Esta versión incluyó la famosa Java 2 Platform, que se dividió en tres ediciones: Java 2 Standard Edition (J2SE), Java 2 Enterprise Edition (J2EE) y Java 2 Micro Edition (J2ME). J2SE trajo consigo la API Swing para interfaces gráficas de usuario, lo que permitió un desarrollo de aplicaciones más atractivas visualmente.

Con Java 1.3 (también conocida como J2SE 1.3), lanzada en el año 2000, se introdujo la tecnología HotSpot, un compilador Just-In-Time que mejoró significativamente el rendimiento de las aplicaciones Java.

La versión 1.4 (J2SE 1.4) de Java, lanzada en 2002, introdujo numerosas mejoras en la plataforma. Entre las características destacadas se incluyen las assertions, la API de Logging y la integración de Java Web Start para la distribución de aplicaciones.

En 2004, se lanzó Java 5 (también conocida como J2SE 5 o Java 1.5) con una característica crucial: las anotaciones. Además, se introdujeron enumeraciones y mejoras en el manejo de excepciones.

Java 6, o J2SE 6, lanzada en 2006, se centró en el rendimiento y la optimización del lenguaje. Además, incluyó mejoras en la gestión de memoria y una mayor integración de XML.

La versión 7 de Java (también conocida como Java 7 o J2SE 7), lanzada en 2011, introdujo características como el manejo de múltiples excepciones y el soporte para tipos numéricos binarios.

Java 8 se lanzó en 2014. Esta versión fue un hito importante en la historia de Java debido a la introducción de las expresiones lambda, Streams API y la nueva fecha y hora API. Las expresiones lambda permitieron una programación más funcional, mientras que Streams API simplificó la manipulación de colecciones de datos.

El JDK

El Kit de Desarrollo de Java (JDK), representa un conjunto integral de herramientas y recursos esenciales que permiten la creación, desarrollo y ejecución de aplicaciones de software basadas en el lenguaje de programación Java. sus principales componentes son:

1.- El compilador de Java: contenido en el JDK, desempeña un papel central en el proceso de transformar el código fuente escrito por el programador en un formato legible por la Máquina Virtual de Java (JVM). Este proceso, conocido como compilación, traduce el código fuente Java en bytecode, un código intermedio que la JVM puede ejecutar. El compilador verifica la sintaxis y la semántica del código fuente para garantizar la corrección y eficiencia del programa.

2.- La Máquina Virtual de Java (JVM): es un componente vital del JDK que interpreta y ejecuta el bytecode generado por el compilador. La JVM es una parte fundamental de la plataforma Java, ya que proporciona portabilidad al permitir que el mismo código bytecode se ejecute en diferentes sistemas operativos sin modificaciones. La JVM también se encarga de la gestión de la memoria y la administración de recursos.

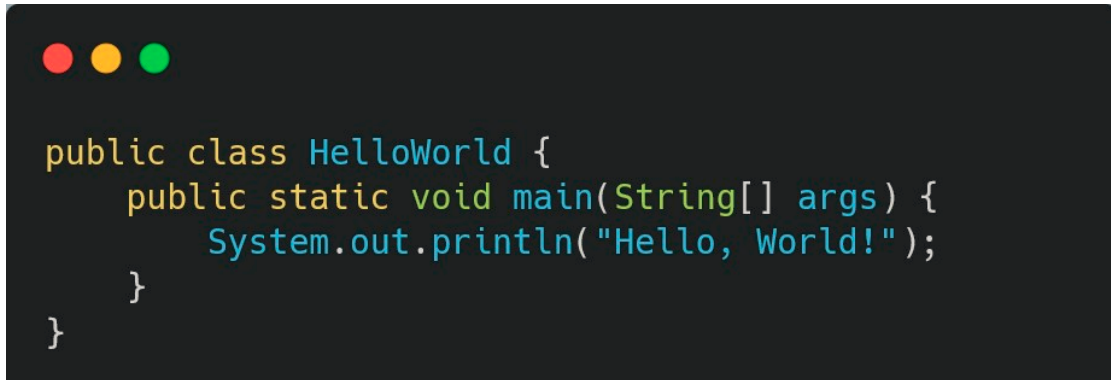
3.- El JDK incluye un conjunto de bibliotecas estándar, conocidas como "bibliotecas de clase", que proporcionan una amplia variedad de clases y métodos predefinidos. Estas bibliotecas abarcan desde operaciones básicas de entrada y salida hasta estructuras de datos complejas y funcionalidades de red. Los programadores pueden aprovechar estas bibliotecas para acelerar el desarrollo de aplicaciones y reducir la necesidad de escribir código desde cero.

4.- El JDK incluye una serie de herramientas de desarrollo que simplifican el proceso de creación y depuración de aplicaciones Java. Algunas de estas herramientas incluyen el depurador, el perfilador, el generador de documentación y la herramienta de empaquetado de aplicaciones. Estas herramientas ofrecen un entorno de desarrollo integral para programadores de Java.

El JDK es la piedra angular de la plataforma Java y permite a los programadores aprovechar la portabilidad, la seguridad y la versatilidad que caracterizan a este lenguaje de programación.

Tu primer programa en java

En este programa, class HelloWorld define una clase llamada HelloWorld. La clase contiene un método llamado main, que es el punto de entrada para cualquier programa Java. El método main imprime "Hello, World!" en la pantalla.

A screenshot of a code editor with a dark background. At the top left, there are three colored circles (red, yellow, green) representing window control buttons. The code is written in a light blue and yellow monospace font. It defines a public class HelloWorld with a static main method that prints "Hello, World!" to the console.

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

Revisando la Java Virtual Machine (JVM)

La JVM se compone de tres componentes principales: el classloader, el runtime data area y el execution engine. El class loader es responsable de cargar las clases Java en la memoria. La runtime data area es donde se almacenan los datos en tiempo de ejecución, como los objetos Java y las variables locales. El execution engine es responsable de ejecutar el código Java compilado.

La JVM también proporciona otras características importantes, como la recolección automática de basura, que ayuda a liberar la memoria no utilizada, y la verificación de seguridad, que ayuda a garantizar que el código Java no tenga acceso no autorizado a recursos del sistema.

Documentación de la API de Java

La documentación de la API de Java es una referencia completa para los desarrolladores que trabajan con el lenguaje de programación Java. La documentación describe todas las clases, interfaces, métodos y variables que se pueden utilizar en Java, junto con ejemplos y explicaciones detalladas.

La documentación de la API de Java se puede encontrar en línea en el sitio web de Oracle. Hay varias versiones disponibles, cada una correspondiente a una versión diferente del lenguaje Java. La documentación está organizada por paquetes, lo que facilita la búsqueda de información sobre un tema específico. También hay una función de búsqueda que permite a los desarrolladores buscar términos específicos en toda la documentación.