INTRODUCCIÓN

Un poco de historia

La historia de Java comienza a finales de los años 80 principios de los 90 cuando un equipo de ingenieros de la empresa **Sun Microsystems** liderados por **James Gosling** y **Patrick Naughton**, se vio en la necesidad de crear un lenguaje de programación para utilizarlo con pequeños electrodomésticos. Comenzaron utilizando el lenguaje C++, pero pronto se encontraron con el problema de que este no cumplía del todo con las necesidades que ha de tener un lenguaje de programación orientado a pequeños electrodomésticos.

Así que decidieron crear un nuevo lenguaje que tenía características válidas de C++ pero prescindía de otras que no servían para su propósito e incorporando otras nuevas. Así nació un nuevo lenguaje al que nombraron "**Oak**", roble en inglés, nombre otorgado al parecer por un viejo roble que se veía a través de la ventana del despacho de Gosling. Por un tema de patentes (parece ser que el nombre ya estaba registrado), cambiaron el nombre a "**Java"**. Cuenta la leyenda que este nombre se debe a la afición que tenía el equipo de ingenieros liderado por Gosling a tomar café. Resulta que café en lenguaje americano coloquial es "Java" lo que le daría al lenguaje su icónico logotipo de una taza de café humeante, presente en las portadas de todos los libros dedicados a este lenguaje.

¿Quién no ha visto este logotipo o alguna de sus variantes en la portada de un libro de Java?

El que fuera creado con el objetivo de utilizarse en pequeños electrodomésticos, hacía que este nuevo lenguaje tuviera unas características muy concretas: era un lenguaje **simple**, **compacto** y sobre todo era un lenguaje **neutro respecto a la arquitectura** (el hardware) requisito indispensable para poder reutilizar el mismo software con diferentes electrodomésticos. Se trataba de que el mismo software funcionara con diferentes procesadores, circuitería etc. de tal forma que ante futuras actualizaciones de un mismo electrodoméstico no hubiera que reprogramar el software.

Sin embargo, Sun Mycrosistems no tuvo éxito en la comercialización de este nuevo lenguaje de programación y cayó en el olvido, quedando "dormido" y olvidado en algún cajón de la empresa.

James Gosling uno de los padres de Java

Un poco más tarde (llegando a mediados de los años 90) surgió el boom de Internet. De forma rápida cada vez más usuarios contaban con conexión a Internet en sus casas. Fue entonces cuando a los directivos de Sun se les ocurrió que podían darle utilidad a aquel lenguaje caído en el olvido, ya que las características de la red Internet se adaptaban como un guante al joven lenguaje Java. Un software que fuera útil en Internet debía ser neutro respecto a la arquitectura (los ordenadores conectados a Internet cuentan con diferente hardware y Sistema Operativo), y también debía ser simple y compacto lo que facilitaría su distribución a través de la red.

En esta situación Patrick Naughton decidió desarrollar un nuevo navegador de Internet escrito en lenguaje Java al que llamaron "**Hot Java**". Un navegador que se podía utilizar en ordenadores con diferentes plataformas (Windows, Unix, Solaris etc) y que además tenía la característica de poder ejecutar pequeños programas en su interior, los denominados **applets** (hoy en desuso).

El lenguaje fue evolucionando y en 1995 apareció la primera versión: **Java 1.0.** Grandes empresas tecnológicas fueron adoptando el uso del lenguaje (a excepción de Microsoft que se resistió en un principio) y con el tiempo el lenguaje se fue convirtiendo en lo que es hoy.

Desde su nacimiento en 1995, Sun Microsystems lanzó seis revisiones importantes del **Java Development Kit** (sistema de desarrollo de Java) y durante los nueve años siguientes, la **Interfaz de Programación de Aplicaciones** (API) ha crecido, pasando de 200 a más de 3000 clases.

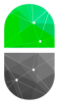
En la actualidad la **API** abarca aspectos tan diversos como la construcción de Interfaces de Usuario, la gestión de Bases de Datos, la traducción a otros idiomas, la protección y el procesamiento de XML.

El **JDK 5.0** publicado en 2004, fue una enorme actualización del lenguaje Java desde su versión original. Posteriormente ha habido más actualizaciones que han incrementado la API notablemente siendo la **versión 8** la más reciente en el momento de escribir estas líneas (Agosto 2017)

Características de Java

- ✓ **Sencillo**: La sintaxis de Java es una versión retocada de la sintaxis de C++, pero eliminando las características más confusas de este. Java no necesita ficheros de encabezado, aritmética de punteros, uniones, sobrecarga de operadores, clases base etc.
  Otro aspecto de la sencillez es la reducción de tamaño. El tamaño del intérprete básico y del soporte para clases es de unos 80 kb.

- ✓ **Orientado a objetos:** El diseño orientado a objetos es una técnica de programación que se centra en los datos (esto es, los objetos) y en las interfaces de esos objetos.
  Hoy en día no se concibe un lenguaje de programación moderno que no esté orientado a objetos.



- ✓ **Distribuido**: Java posee una extensa biblioteca de rutinas para tratar protocolos de TCP/IP como HTTP y FTP. Las aplicaciones de Java pueden abrir objetos y acceder a objetos remotos con tanta facilidad como acceder a un sistema local de ficheros.

  En la actualidad la arquitectura J2EE admite aplicaciones distribuidas a gran escala.

- ✓ **Robusto:** Java está diseñado para escribir programas que deben resultar fiables en múltiples sentidos. Java hace hincapié en la comprobación temprana de posibles errores, efectúa después una comprobación dinámica (en el momento de la ejecución) y elimina las situaciones proclives a error.

  El compilador de Java detecta muchos errores que en otros lenguajes de programación solo sería posible detectarlos en tiempo de ejecución.

- ✓ **Seguro**: Java está diseñado para utilizarse en entornos distribuidos o en entornos de red. Con este fin, se ha hecho mucho hincapié en la seguridad. Java permite construir sistemas libres de virus y manipulaciones.

  Java se ha diseñado para evitar ciertos ataques como por ejemplo:

  - Desbordar la pila de ejecución, ataque común de gusanos y virus.
  - Corromper la memoria fuera de su propio proceso.
  - Leer o escribir ficheros sin permiso.

  Con el tiempo se han ido añadiendo nuevas características de seguridad como por ejemplo las clases con firma digital. Esto permite saber quién ha escrito las clases.

- ✓ **Neutro respecto a la arquitectura:** Al compilar un programa Java, se genera automáticamente un fichero de bytecodes independiente de la arquitectura de la máquina. El fichero de bytecodes será interpretado por la máquina virtual de Java (JRE).

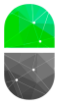
  Es decir, lo único que habrá que tener instalado en la máquina donde se desea ejecutar el programa Java es la máquina virtual de Java.

  Un programa escrito en Java funcionará igual en una máquina con Windows, Linux, Unix, Solaris etc. También será independiente del hardware (procesador, placa, memoria etc.)

- ✓ **Adaptable**: Los tamaños de los datos primitivos están especificados, así como el comportamiento de la aritmética que se les aplica

  Ejemplo: en Java un int siempre es un entero de 32 bits. En C o C++ puede ser de 16 bits o de 32 bits.

  Al tener un tamaño prefijado para los tipos de datos se elimina uno de los principales quebraderos de cabeza a la hora de crear adaptaciones.



Crear un programa que tenga el mismo aspecto en Windows, Macintosh o variantes de Unix es un gran esfuerzo para el programador. Java proporciona una interfaz de usuario común a todas las plataformas.

- ✓ **Interpretado**: Al compilar un programa Java, el compilador genera un fichero de bytecodes que posteriormente será interpretado por la JRE (máquina virtual de Java).

- ✓ **Alto rendimiento:** El rendimiento de los bytecodes suele ser aceptable en la mayoría de las ocasiones, pero para aquellas ocasiones en las que no resulte suficiente, existe la posibilidad de traducir sobre la marcha esos bytecodes a código máquina. Los resultados de esa traducción se guardan en memoria y cuando se vuelvan a necesitar se recurre a ellos. Esto acelera en muchas ocasiones el proceso de Interpretación de los bytecodes.

  A esta característica se la denomina JIT (Just in Time), que viene a querer decir traducción sobre la marcha.

- ✓ **Multihilo**: La ejecución multihilo consiste fundamentalmente en la ejecución de varias partes de código a la vez, de forma que a la hora de ejecutar el programa parecerá una ejecución multitarea.

  Esta es una de las características más atractivas de Java.

- ✓ **Dinámico:** Java se ha diseñado para adaptarse a un entorno cambiante. Las bibliotecas pueden añadir libremente nuevos métodos y variables de ejemplar sin que sus clientes se vean afectados.

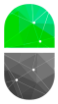
  Esta es una característica muy útil cuando se necesite añadir código a un programa en ejecución.

Algunas ideas erróneas de Java

- ✓ **Java es una extensión de HTML:** No tienen nada en común salvo que existen extensiones de HTML para ubicar applets de Java en una página web.

- ✓ **Yo uso XML, así que no necesito Java**: Java es un lenguaje de programación; XML es una forma de describir datos.

- ✓ **Java es un lenguaje de programación fácil de aprender**: No hay ningún lenguaje de programación tan potente como Java que sea fácil de aprender. Las bibliotecas de Java contienen miles de clases e interfaces, y decenas de miles de funciones. No hay que conocerlas todas, pero si un número notable de ellas para trabajar en Java de manera seria.



- ✓ **Todos los programas de Java se ejecutan en una página web:** Java es un lenguaje de propósito general. Esto quiere decir que podemos crear aplicaciones de escritorio, aplicaciones de consola o aplicaciones web. No estamos limitados al ámbito de la web