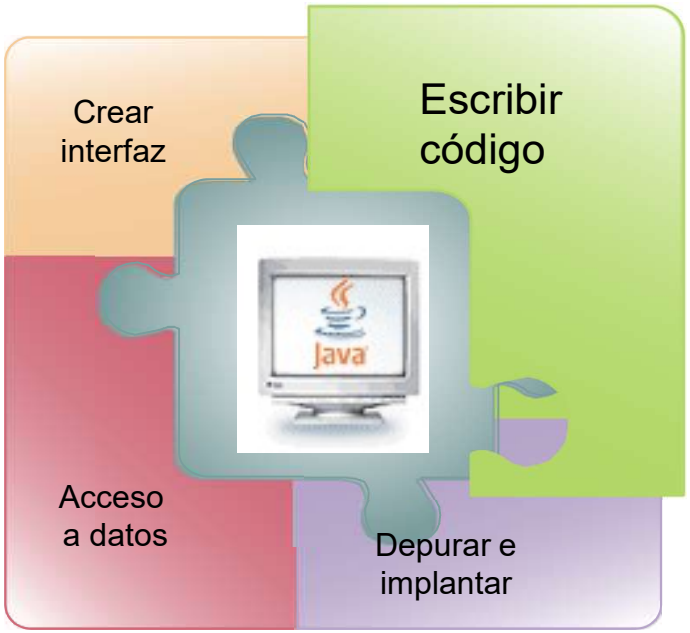
U1. Introducción

2021-2022

Descripción



1. Qué es JAVA
   - JDK
   - IDE
   - Fichero Fuente
   - Hola mundo!
2. Instalación de Netbeans
3. Entorno de Netbeans
4. Ejercicios

¿QUÉ ES JAVA?

¿Que es JAVA?

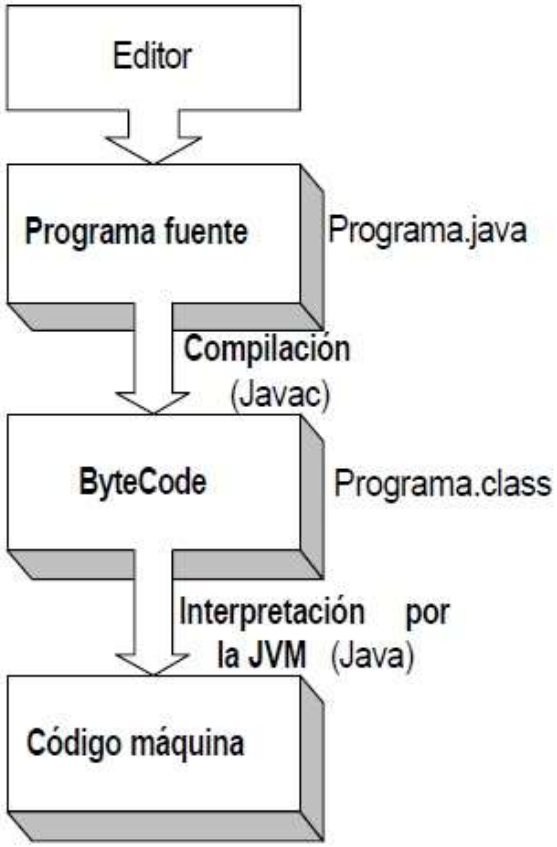
- Java es un lenguaje de desarrollo de propósito general, y como tal es válido para realizar todo tipo de aplicaciones profesionales.
- Los programas "ejecutables", creados por el compilador de Java, son independientes de la arquitectura
  - Se ejecutan indistintamente en una gran variedad de equipos con diferentes microprocesadores y sistemas operativos.

Características del lenguaje Java

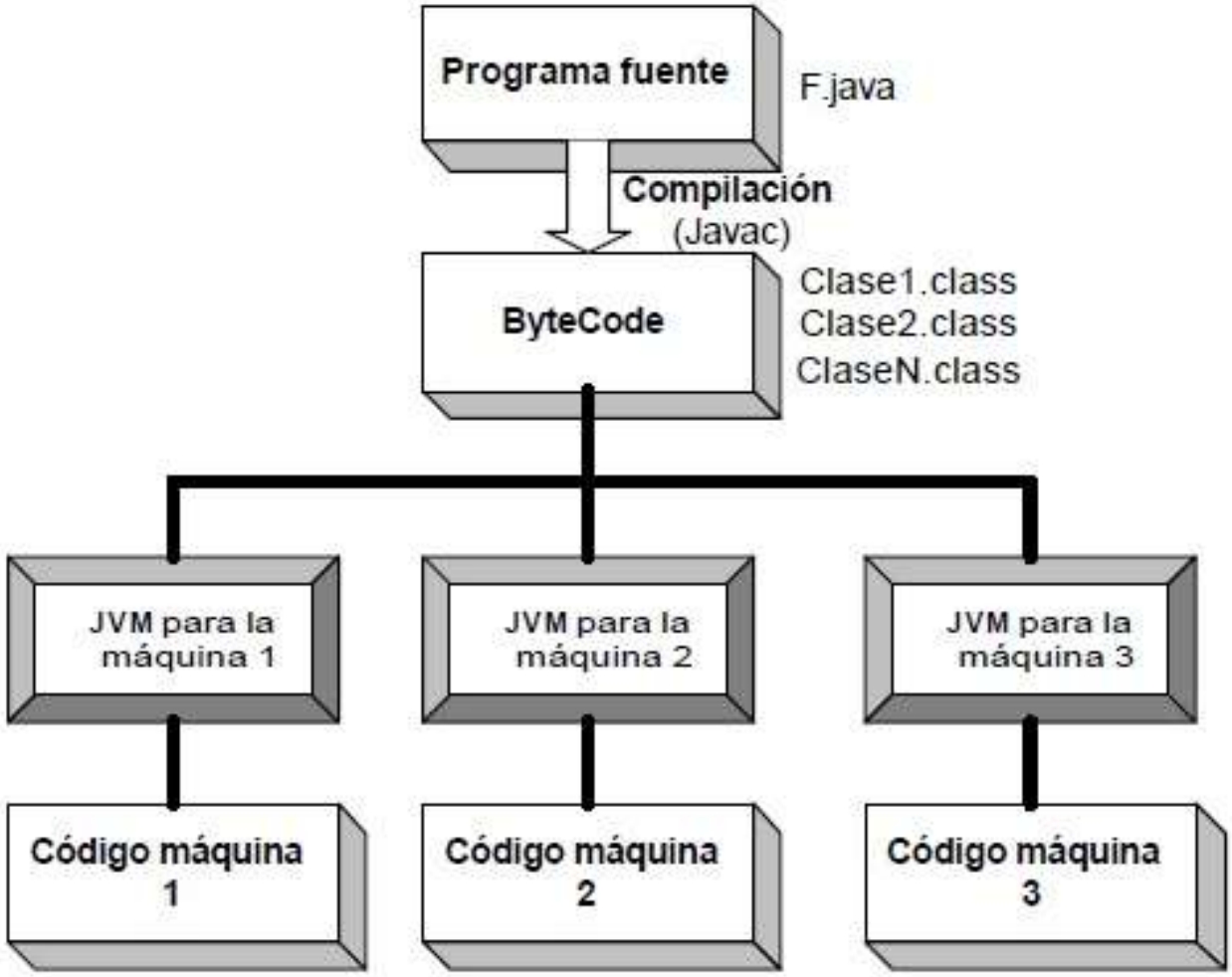
- Es intrínsecamente **orientado a objetos**
- Funciona perfectamente en red
- Aprovecha características de la mayoría de los lenguajes modernos evitando sus inconvenientes. En particular los del C++ (ej los puntos 5 y 6 que vienen a continuación)
- Tiene una gran funcionalidad gracias a sus librerías (clases)
- NO tiene punteros manejables por el programador, aunque los maneja interna y transparentemente
- El manejo de la memoria no es un problema, la gestiona el propio lenguaje y no el programador
- Genera aplicaciones con pocos errores posibles
- Incorpora *Multi-Threading* (para permitir la ejecución de tareas concurrentes dentro de un mismo programa)
- El lenguaje Java puede considerarse como una evolución del C++

¿Java es Compilado o Interpretado?

- Aunque estrictamente hablando es interpretado, necesita de un proceso previo de compilación
- Una vez "compilado" el programa, se crea un fichero que almacena lo que se denomina ***bytecode*** (pseudocódigo prácticamente al nivel de código máquina, pero de la máquina VIRTUAL)
- Para ejecutarlo, es necesario un "intérprete", la JVM (Java Virtual Machine) ó Máquina Virtual Java.
- De esta forma, es posible compilar el programa en una estación UNIX y ejecutarlo en otra con Windows utilizando la máquina virtual Java para Windows
- Esta **JVM** (java virtual machine) se encarga de leer los bytecodes y traducirlos a instrucciones ejecutables directamente en un determinado microprocesador, de una forma bastante eficiente



La Máquina Virtual Java (JVM)



La Máquina Virtual Java (JVM)

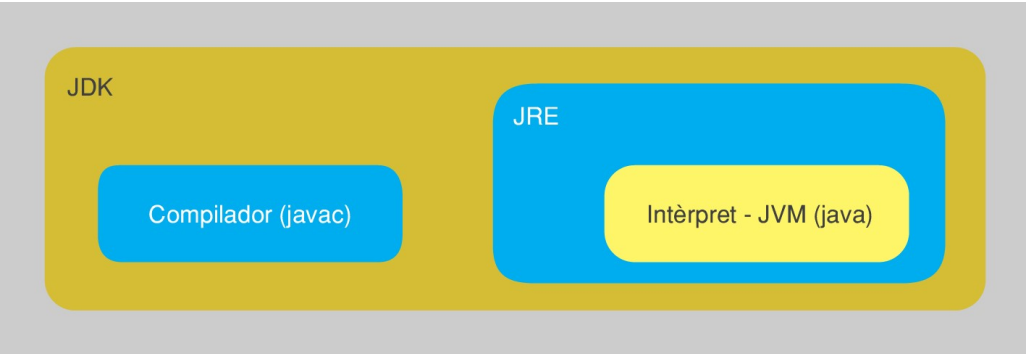
- Un mismo programa fuente compilado en distintas plataformas o sistemas operativos, genera el mismo fichero en byte-code
- La JVM realiza la traducción de ese byte-code a código nativo de la máquina sobre la que se ejecuta
- Existe una versión distinta de esta JVM para cada plataforma. Esta JVM se carga en memoria y va traduciendo "al vuelo", los byte-codes a código máquina
- La JVM no ocupa mucho espacio en memoria

Java Runtime Environment (JRE)

- Java Runtime Environment o JRE es un conjunto de utilidades que permite la ejecución de programas Java
- Está formado básicamente por:
  - Una Máquina Virtual Java o JVM
    - Es el programa que ejecuta el código Java previamente compilado (bytecode)
  - Un conjunto de bibliotecas Java para proporcionar los servicios que pueda necesitar la aplicación
    - (El API de JAVA formada por librerías de clases estándar)
  - Otros componentes…

El entorno de desarrollo JDK

- Un usuario sólo necesita el JRE para ejecutar las aplicaciones desarrolladas en lenguaje Java
- Para desarrollar nuevas aplicaciones en Java la herramienta básica es el JDK (*Java Developer's Kit)* o Kit de Desarrollo Java
- Incluye entre otros:
  - Un compilador
  - Un JRE (máquina virtual JVM y librerías)



- El Kit de desarrollo puede obtenerse en la dirección siguiente:

  http://www.oracle.com/technetwork/java/javase/downloads/index.html

El entorno de desarrollo JDK

- En el entorno para Windows está formado por un fichero ejecutable que realiza la instalación, creando toda la estructura de directorios
- El kit contiene básicamente:
  - El **compilador**: javac.exe
  - El **depurador**: jdb.exe
  - El **intérprete**: java.exe y javaw.exe
  - El visualizador de applets: appletviewer.exe
  - El generador de documentación: javadoc.exe
  - Un desensamblador de clases: javap.exe
  - El generador de archivos fuentes y de cabecera (.c y .h) para clases nativas en C: javah.exe

El IDE

- Un IDE (*integrated development environment o entorno integrado de desarrollo*) es una herramienta que integra todo lo necesario para generar programas de ordenador de manera que el trabajo sea mas cómodo
- Algunos ejemlos de IDE para trabajar con Java:
  - Netbeans, Eclipse y Jcreator

Fichero fuente

Aquí es donde escribiremos nuestro programa en lenguaje Java. ¿Recuerdas lo que es el código fuente?

- La **extensión** de los ficheros de código fuente en Java es **.java**
- Convención para dar **nombre** al fichero de código fuente:

  UpperCamelCase (**notación de camello en mayúsculas)**

  usar solo letras consecutivas sin acentos (ni espacios, subrayados o números) y con la inicial de cada palabra siempre en mayúsculas

  - No es imprescindible pero si recomendable
- Algunos ejemplos:
  - Prova.java
  - HolaMundo.java
  - ElMeuProrama.java

El primer programa

```
public class HolaMundo  {
   public static void main (String[ ] args)  {
      System.out.println(“¡Hola, mundo!");
     }
}
```

Conceptos iniciales

- Todo programa en Java se compone de una o más clases.
- Por lo pronto crearemos programas con 1 sola clase
- En este caso la clase debe contener el método **main** que es por el que se comienza la ejecución del programa

El primer programa

- Primera línea: **class HolaMundo { }**
  - Hemos creado una clase llamada HolaMundo:
  - La manera de declarar una clase es mediante el uso de la palabra reservada **class** seguido del nombre de la clase. El contenido de la clase irá encerrado entre llaves { … }
  - **El nombre que se le da a la clase**, en nuestro caso HolaMundo, **debe coincidir exactamente** – incluidas minúsculas y mayúsculas – **con el nombre del fichero que contiene el código**. El nombre del fichero finalizará con la extensión .java, en nuestro caso, HolaMundo.java
- RECUERDA!: Java es sensible a mayúsculas. HolaMundo, holaMundo y Holamundo son cosas diferentes

El primer programa

- Segunda línea:

public static void main (String[ ] args) { }

  - Esta es la definición de la cabecera del método main( ). El método main es el **punto de inicio de la ejecución del programa**
  - Las palabras **public, static y void** son palabras reservadas que estudiaremos más adelante, cuando abordemos la programación orientada a objetos
  - Los parámetros que puede recibir el método main( ) están entre paréntesis y se trata de un conjunto de cadenas de caracteres (String). El nombre del parámetro args se utiliza por convención, aunque se podría utilizar cualquier otro. Ya lo iremos viendo…

El primer programa

- Tercera línea:

System.out.println(“¡Hola mundo!”);

  - El programa debe imprimir por pantalla el mensaje ¡Hola mundo!
  - Para ello, hacemos uso del método println(), que pertenece al *espacio de nombres* *System.out* y que imprime por pantalla la cadena de caracteres que se le pasa como parámetro

Bloques

- Bloques: {…}
  - En Java las sentencias se agrupan en bloques que vienen delimitados por una llave de apertura y una de cierre
  - Vemos que las llaves nos permiten definir clases, métodos, bloques de código,…

Terminología

- ¿parámetros?
- ¿método?
- ¿bloque?

- Busca la definición adecuada para estos términos