



# Introducción a la Programación en Java

---



# Objetivos del curso

---

- Es un curso básico, introductorio de java, para personas que no tengan conocimientos del lenguaje. Se utilizará **Netbeans** como entorno de desarrollo (en java siempre hay diversas opiniones sobre cuál es el mejor entorno de desarrollo a utilizar).

# Algunas alternativas existentes en el mercado

---

- **Nota:** El curso no trata sobre el aprendizaje de Netbeans, el entorno de desarrollo a utilizar es una decisión personal de cada quien.
- Estos son unos pocos entornos de desarrollos destacados:
  - JDK, es la solución que ofrece Oracle. Java perteneció a Sun Microsystems y después fue absorbido por Oracle.
  - Eclipse
  - BlueJ
  - JBuilder
  - JCreator
  - JDeveloper



# Temario del curso

---

## **Lunes 25**

- Clase Introductoria

## **Martes 26**

- Programación orientada a objetos
- Flujo de control. Condicionales y ciclos

## **Miércoles 27**

- Excepciones

## **Jueves 28**

- Conexión a bases de datos

## **Viernes 29**

- Aspectos básicos para crear interfaces gráficas

# Instalación Netbeans + JRE

---

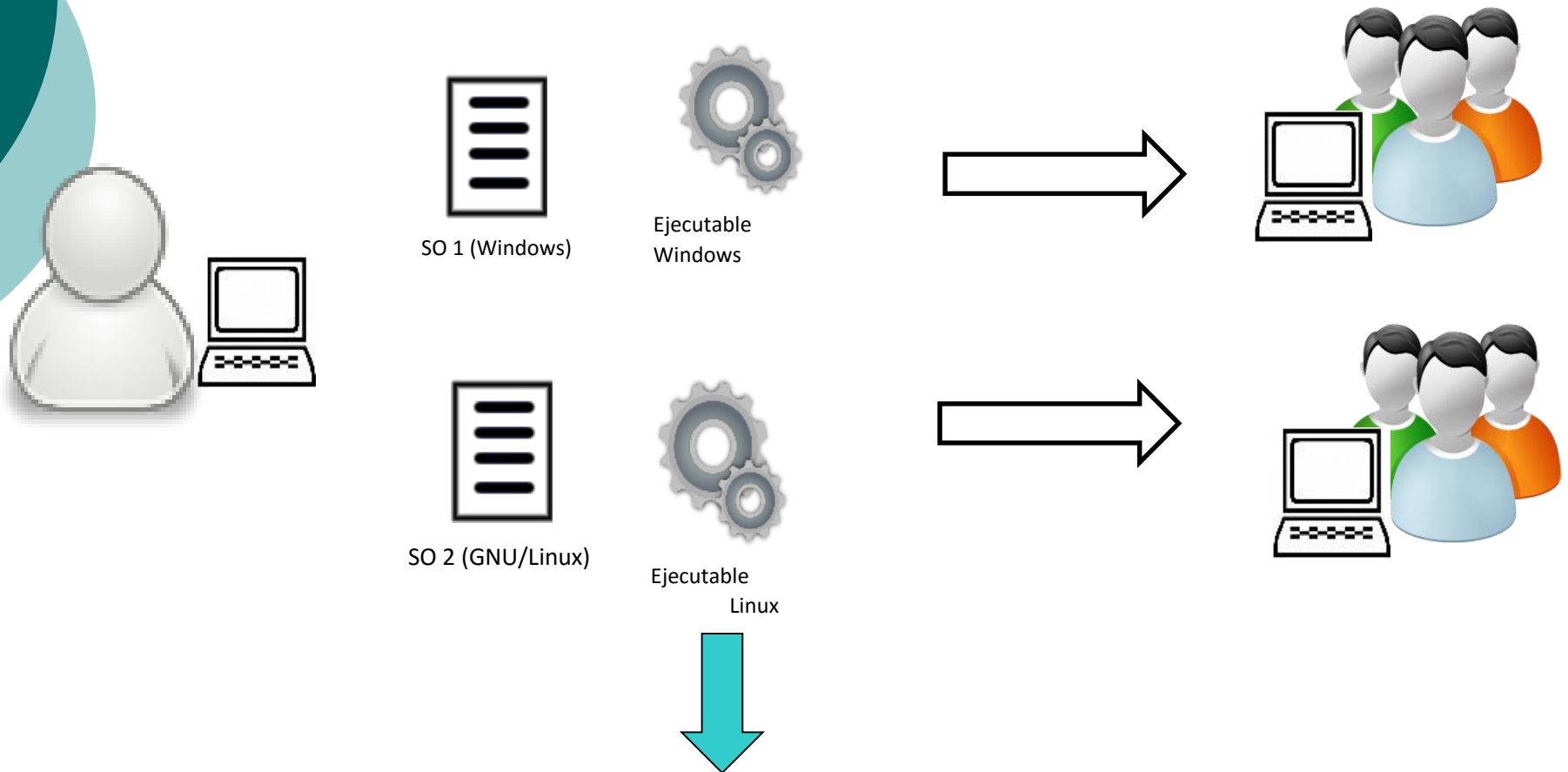
- Se utilizará como **entorno de desarrollo NetBeans**, sin embargo se va a tener que instalar otro paquete de software **indispensable** para programar en **java** que es **Java Runtime Enviroment (JRE)**.
- Este paquete de software se va a tener que instalar **independientemente de cual sea el entorno de desarrollo** que se ha escogido para programar (Netbeans, Eclipse, ...) y también se va a tener que instalar **independientemente de la plataforma** que se esté utilizando (la versión correspondiente para cualquier sistema operativo)

# ¿Qué es JRE?

---

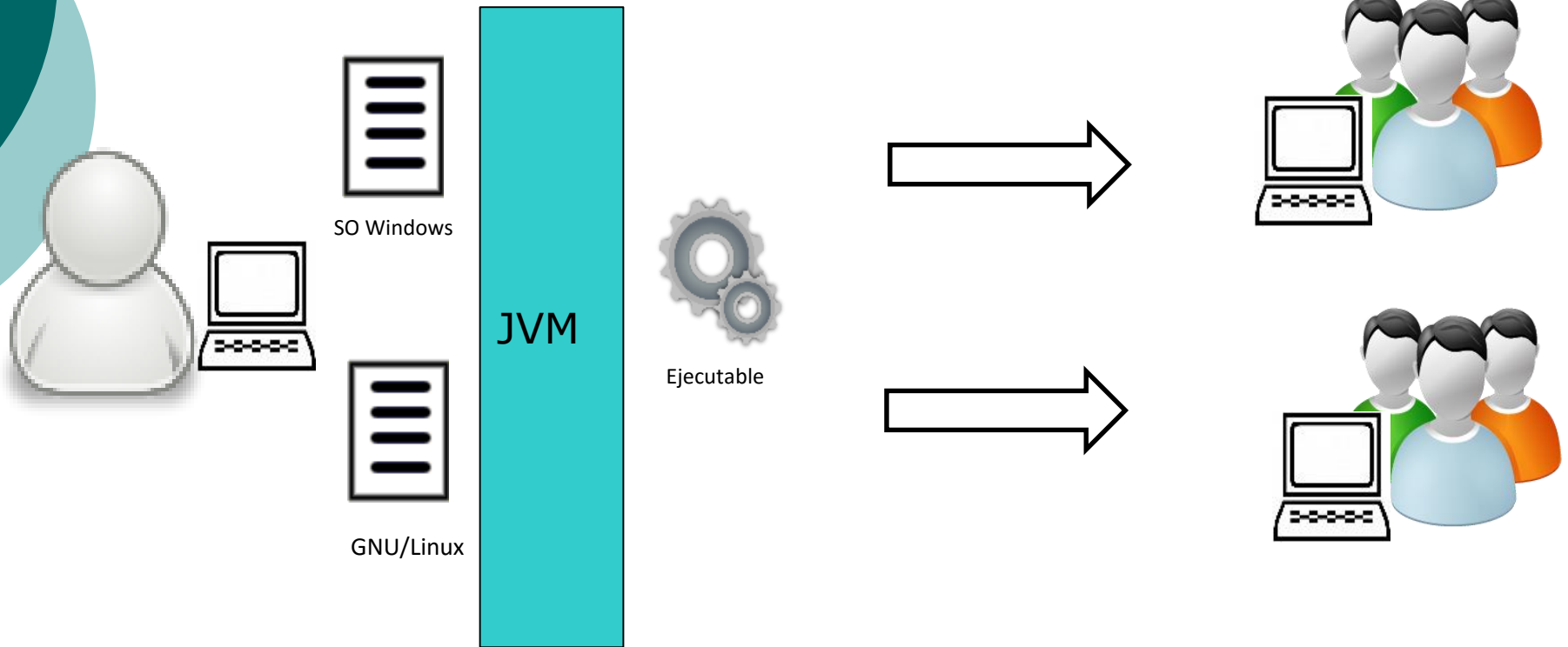
- J.R.E = Java Runtime Environment (Entorno de ejecución de Java). Es un conjunto de utilidades de Java y contiene la Máquina Virtual de Java(JVM, por sus siglas en inglés)
- ¿Por qué es necesaria la instalación?
- R/. Es debido a la **principal característica** del lenguaje de programación Java: ser **multiplataforma**. Por lo tanto el programa será válido para cualquier plataforma
- El hecho de ser multiplataforma implica que un programa escrito en Java debe ser **compilado** para posteriormente ser **interpretado** por la **máquina virtual**

# Que suele suceder...



Esto es lo que Java evita, al ser multiplataforma

# Que suele suceder...





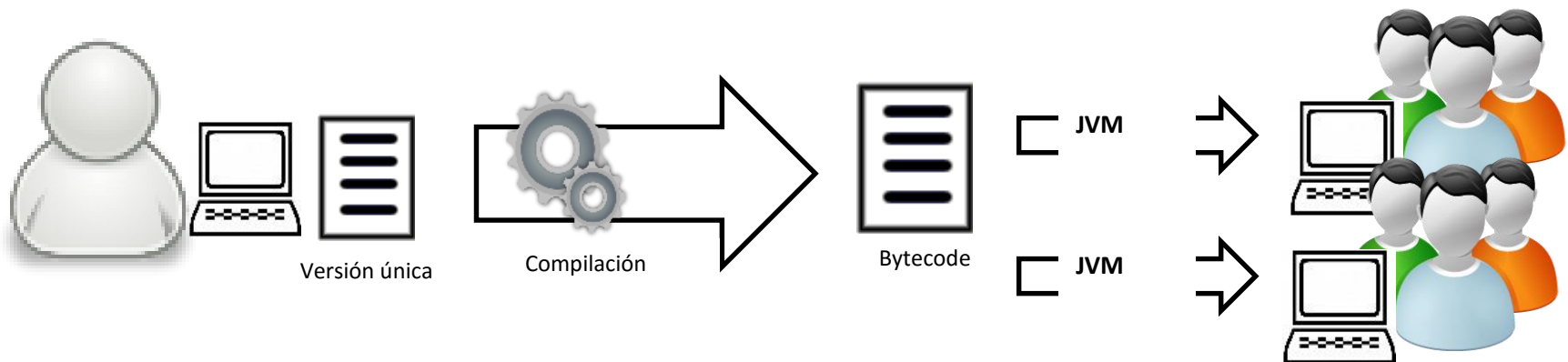
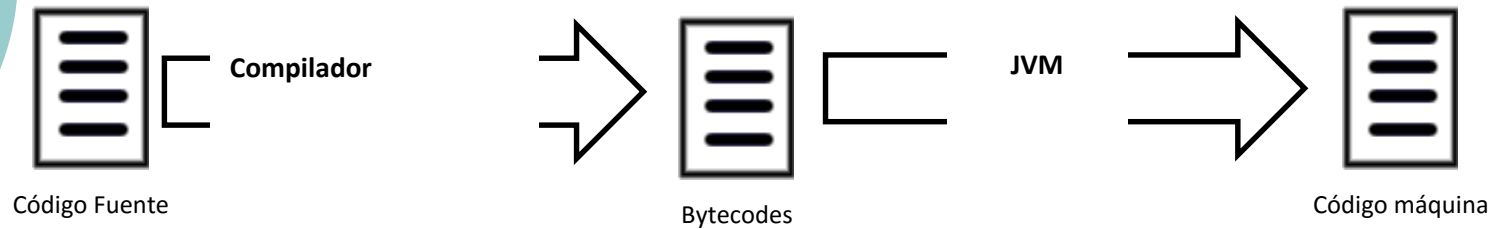
# Solución Java independencia de plataforma

---

- Una vez creado el **código fuente** (ficheros con extensión .java) es necesario **compilarlo** y se genera un archivo **intermedio**, denominado **bytecode** y lleva por extensión .class.
- La **interoperabilidad** de ejecución de Java se debe a que el bytecode es interpretado por la **JVM**, y es a través de este componente que se logra una ejecución idéntica en distintos ambientes, obteniéndose el mismo resultado.

**Nota:** El **inconveniente** es que se está obligado a tener instalado JRE, JDK...

# Write once, and run everywhere



# Descarga e instalación de Netbeans y JRE

- La descarga de JRE o JKD se puede realizar desde el sitio: [java.com](http://java.com)(versión de **32/64 bits**)



Java™

Buscar

Descargar Ayuda

# JAVA Y TÚ, DESCARGAR HOY

[» ¿Qué es Java?](#) » [¿Tengo Java?](#) » [¿Necesita ayuda?](#)

Acerca de Java (sitio en inglés)

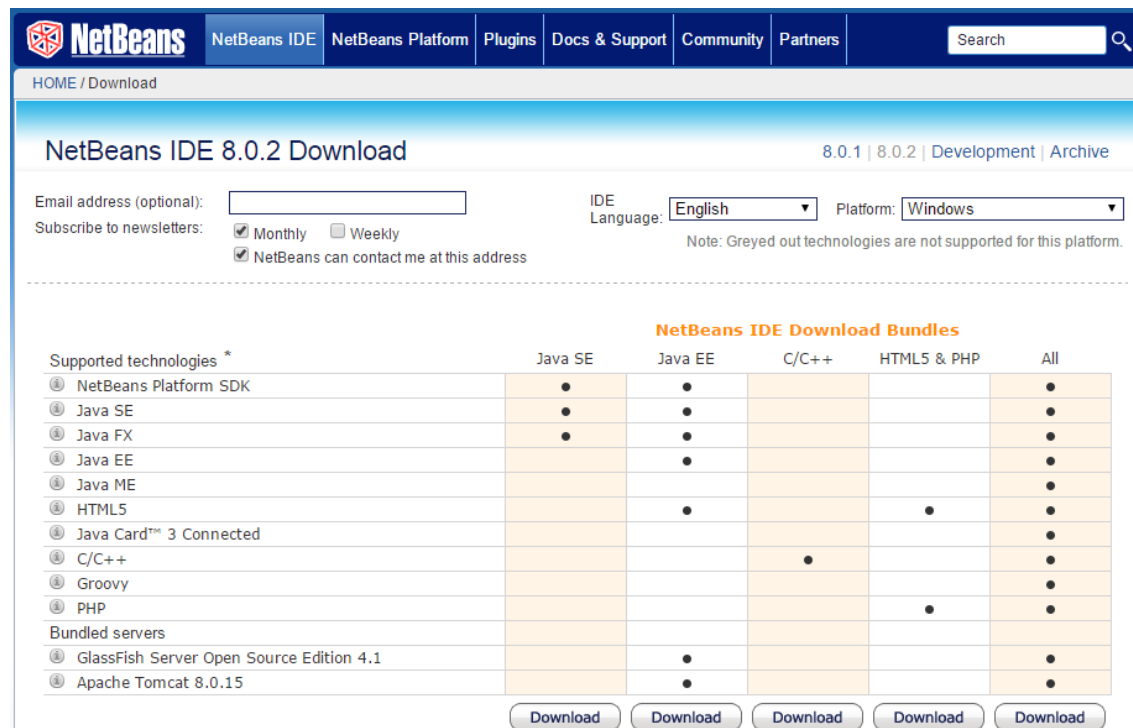
 Java Developer Conference	 Java + Alice	 Java + Greenfoot	 Java + BlueJ	 Oracle Academy	 Java Magazine
--	---	--	---	---	--

[Seleccionar idioma](#) | [Acerca de Java](#) | [Soporte](#) | [Desarrolladores](#)  
[Privacidad](#) | [Condiciones de uso](#) | [Marcas registradas](#) | [Descargo de responsabilidad](#)

ORACLE™

# Descarga e instalación de Netbeans y JRE (cont.)

- La descarga de Netbeans se puede realizar desde el sitio: <https://netbeans.org/downloads/index.html>
- Es un producto libre.



NetBeans IDE 8.0.2 Download

8.0.1 | 8.0.2 | Development | Archive

Email address (optional):

Subscribe to newsletters: ☒ Monthly ☐ Weekly

☒ NetBeans can contact me at this address

IDE Language:  Platform:

Note: Greyed out technologies are not supported for this platform.

Supported technologies *	Java SE	Java EE	C/C++	HTML5 & PHP	All
NetBeans Platform SDK	•	•			•
Java SE	•	•			•
Java FX	•	•			•
Java EE		•			•
Java ME					•
HTML5		•		•	•
Java Card™ 3 Connected					•
C/C++			•		•
Groovy					•
PHP				•	•
Bundled servers					
GlassFish Server Open Source Edition 4.1		•			•
Apache Tomcat 8.0.15		•			•

Download Download Download Download Download



# Características de Java

---

- **Es simple:** lenguaje de programación basado principalmente en los lenguajes C y C++ pero elimina algunas características de estos lenguaje que creaban problemas
- **Es orientado a objeto:** Java trabaja con sus datos como objetos. Soporta las características propias del paradigma orientado a objetos:
  - Abstracción
  - Encapsulación
  - Herencia
  - Polimorfismo



# Características de Java (cont.)

---

- **Es distribuido**
- **Es seguro**
- **Es neutro respecto a la arquitectura**
- **Es adaptable**
- **Es de alto rendimiento**
- **Es multihilo**
- **Es robusto**
- **Es portable**

# ¿Qué tipos de programas se pueden crear en Java?

---

- **Aplicaciones de consola**
- **Aplicaciones de propósito general**
- **Applets**



# Anatomía de un programa java

```
package clase.pkg1;

/**
 *
 * @author maimara
 */
public class Clase1 {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        System.out.println("Hola alumnos");
    }
}
```

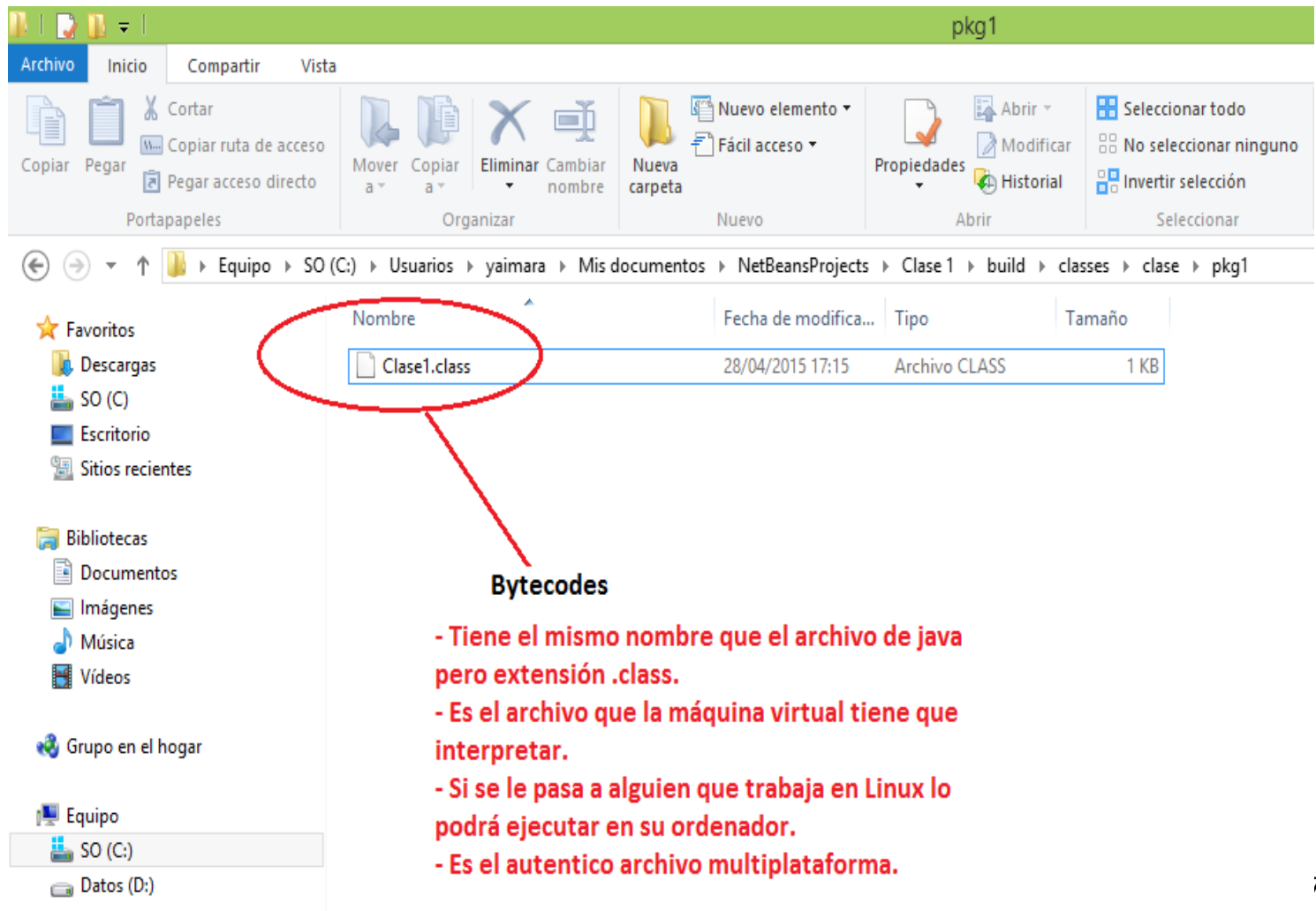
Modificador de acceso

Definición de clase

- **Nota:** en java las sentencias terminan en punto y coma
- **Nota:** java es un **lenguaje case sensitive**



# Archivo Bytecodes



The screenshot shows a Windows File Explorer window with the address bar path: Equipo > SO (C:) > Usuarios > yaimara > Mis documentos > NetBeansProjects > Clase 1 > build > classes > clase > pkg1. The file list contains one item:

Nombre	Fecha de modifica...	Tipo	Tamaño
Clase1.class	28/04/2015 17:15	Archivo CLASS	1 KB

A red circle highlights the file 'Clase1.class', and a red arrow points from the circle to the word **Bytecodes**.

- Tiene el mismo nombre que el archivo de java pero extensión .class.
- Es el archivo que la máquina virtual tiene que interpretar.
- Si se le pasa a alguien que trabaja en Linux lo podrá ejecutar en su ordenador.
- Es el autentico archivo multiplataforma.

# Estructuras principales del lenguaje.

## Tipos de datos primitivos

---

- Enteros
  - **Int** (4 bytes de espacio para almacenamiento. Desde -2,147,483,648 hasta 2,147,483,647)
  - **Short** (2 bytes de espacio para almacenamiento. Desde -32,768 hasta 32,767)
  - **Long** (8 bytes de espacio para almacenamiento. Una barbaridad)
  - **Byte** (1 byte de espacio para almacenamiento. Desde -128 hasta 127)



# Estructuras principales del lenguaje

## Tipos de datos (Primitivos)

---

- Decimales
  - **Float** (4 bytes de espacio para almacenamiento)
  - **Double** (8 bytes de espacio para almacenamiento)
- Char: representa caracteres
- Boolean: true/false

# Variables

---

- **Variable:** espacio en la memoria del ordenador donde se almacenará un valor
- **¿Por qué hay que utilizar variables?**
  - **R./** Cuando se crea un programa surge la necesidad de guardar datos temporalmente que se necesitarán en el futuro en ese mismo programa.

## **¿Cómo se crea una variable en java?**

- **R./** tipo de dato + nombre de la variable.
- Ej. `int salario;`

Variable inicializada: **`salario = 200;`**

# Declaración de constantes y operadores

- **Constantes:** espacio en la memoria del ordenador donde se almacenará un **valor fijo**
- **¿Cómo se crea una constante en java?**
  - R./ palabra reservada “final”+ tipo de dato + nombre de la variable = asignación del valor.
- Ej. **final double pulgadas = 2.54;**

PI

```
public static final double PI
```

“static final” se crea una constante de clase, un atributo común a todos los objetos de esa clase.

# Operadores

---

- Aritméticos:
  - Suma +
  - Resta -
  - Multiplicación \*
  - División /
  - Resto %
- Lógicos, relacionales y booleanos
  - Es igual ==
  - Es distinto !=
  - Menor, menor o igual, mayor, mayor o igual <, <=, >, >=
  - Operador Y &&
  - Operador Or ||
  - Operador not !

# Operadores(cont.)

---

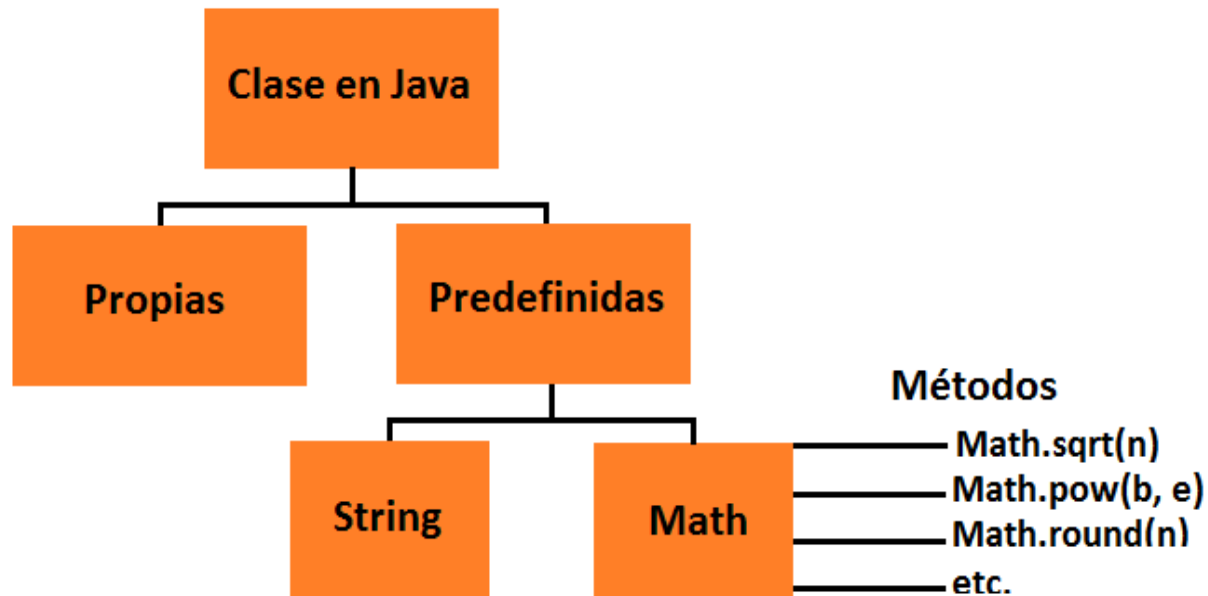
- Incremento y decremento
  - Incremento ++
  - Decremento --
  - Incremento en x valor += x Ej. += 3 (incrementa en 3 al valor donde se esté aplicando)
  - Decremento en y valor -= y Ej. -= 3 (decremento en 3 al valor donde se esté aplicando)
- Concatenación
  - Une o concatena +

# Otros cálculos numéricos en java.

## Utilización de la Clase Math

---

- Las clases en java pueden ser de dos tipos: **propias y predefinidas**.
  - Las propias son las que elaboramos.
  - Las predefinidas ya están construidas dentro del lenguaje de programación java





# API de java

---

- **¿Qué es la API de java?**
  - **R./** Es una **biblioteca** donde vienen todas las clases del lenguaje de programación que todo programador deberá consultar con frecuencia.
- **Nota:** en el siguiente enlace se podrán ver todas las clases con las que se cuenta para programar en java.  
<http://docs.oracle.com/javase/7/docs/api/>

# Clase Math y sus métodos

---

- `Math.sqrt(n)`: raíz cuadrada de `n`
- `Math.pow(base, exp)`: potencia de un número.
  - Base y exp. son doubles
- `Math.round(decimal)`: redondea

**Ver ejemplo**

# Clase String y sus métodos

---

- String no es un tipo primitivos, es una clase
- **String mi\_nombre = "Xochil";** //donde mi\_nombre es un **objeto** de la **clase String**
- La clase String tiene definidos varios **métodos** para manipular cadenas de textos.
  - **Length()**: devuelve la **longitud** de una cadena de caracteres. Ej: `mi_nombre.length() = 6`

# Clase String y sus métodos (cont.)

---

- **charAt(n)**: devuelve la **posición** de un carácter dentro de una cadena (las posiciones comienzan en 0)
- **substring(x, y)**: devuelve un **subcadena**, siendo x el carácter a partir del cual se extrae and y la cantidad de caracteres
- **equals(cadena)**: devuelve **true** si dos cadenas que se comparan son iguales y **false** si no lo son. Distingue mayúsculas de minúsculas
- **equalsIgnoreCase(otra cadena)**: igual que el anterior pero ignora las mayúsculas



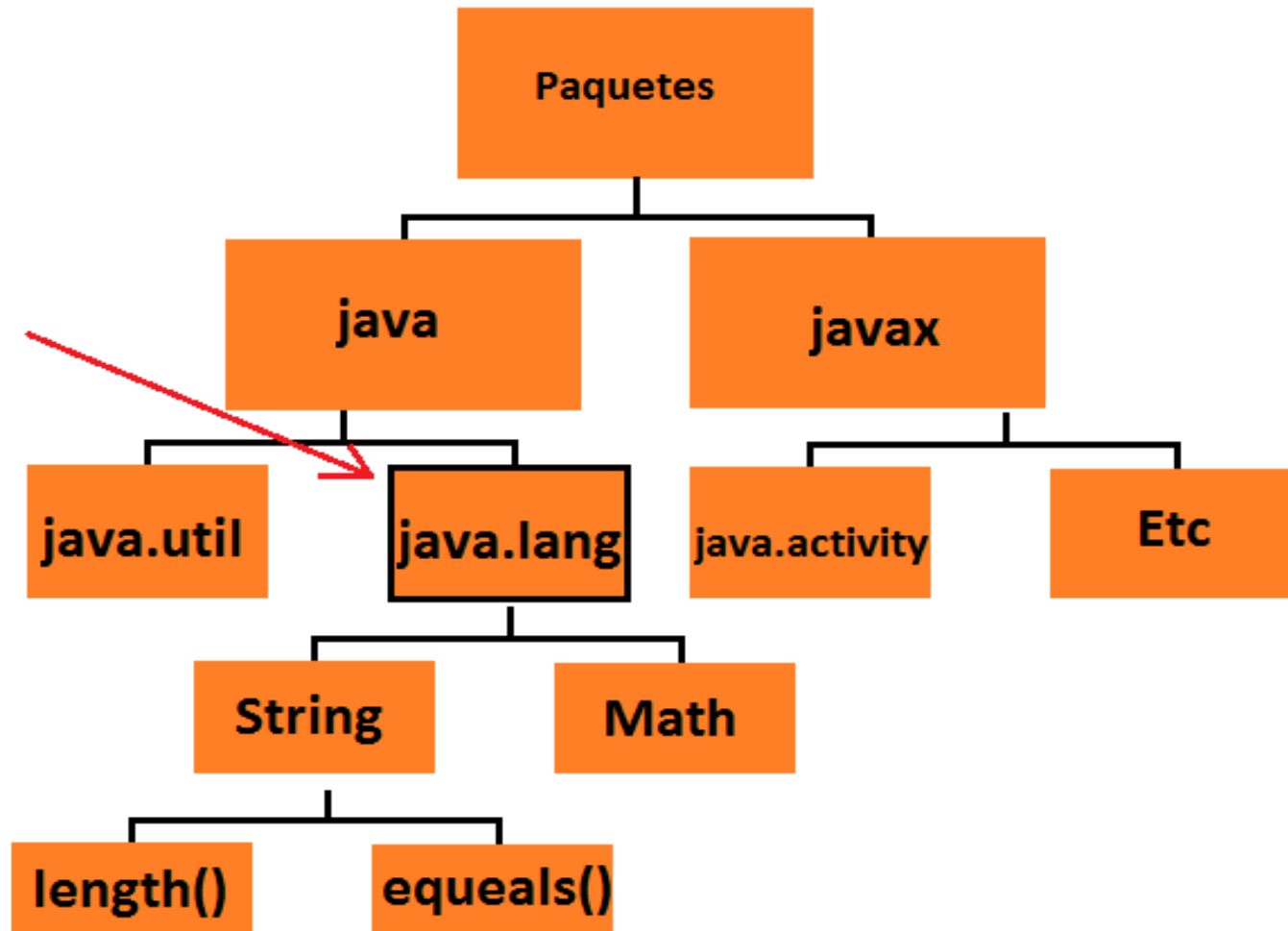
# Paquetes

---

- Clases: predefinidas y propias
- **¿Cómo se organizan?**
  - **R./** Las clases de java están organizadas en **paquetes**, representando su orden jerárquico
- Los paquetes ayudan a:
  - Organizar las clases
  - Evitar conflictos de nombres
  - Controlar la visibilidad de las clases

# Paquetes (cont.)

---



## Paquetes(cont.)

---

- En java hay un paquete que se considera paquete principal o por defecto es **java.lang**
- **¿Qué ocurre si necesitamos utilizar otra clase de la API de java que se encuentra en otro paquete?**
  - R./ No se puede usar libremente sin antes haberlo indicado
  - **import java.util.\*;**
- **Nota:** al poner el \* indica que se importan todas las clases del paquete java.util. Si solo vas a usar una clase en específico, se sustituye el \* por el **nombre de la clase**.
  - **import java.util.Scanner;**

# Entrada salida de datos

---

- Introducir datos utilizando un par de clases que vienen en la API de java:
- **Scanner**, permite introducir información utilizando la consola
  - `nextLine()`
  - `nextInt()`
  - `nextDouble()`
- **JOptionPane**, se construye una ventana muy sencilla con un cuadro de texto y dos botones.
  - `showInputDialog()`
  - `showMessageDialog()`
  - `showConfirmDialog()`

**Ver ejemplo**