Java

Lenguaje Interpretado

Compilar vs Interpretar

- Para un programa compilado, debes crear ejecutables para cada uno de los S.O.
- Un lenguaje compilado es mucho más rápido que uno interpretado.
- Cuando es ejecutado ya se encuentra en código de máquina y eso también le permite hacer algunas optimizaciones que no son posibles con un lenguaje interpretado.

Ventajas de interpretados

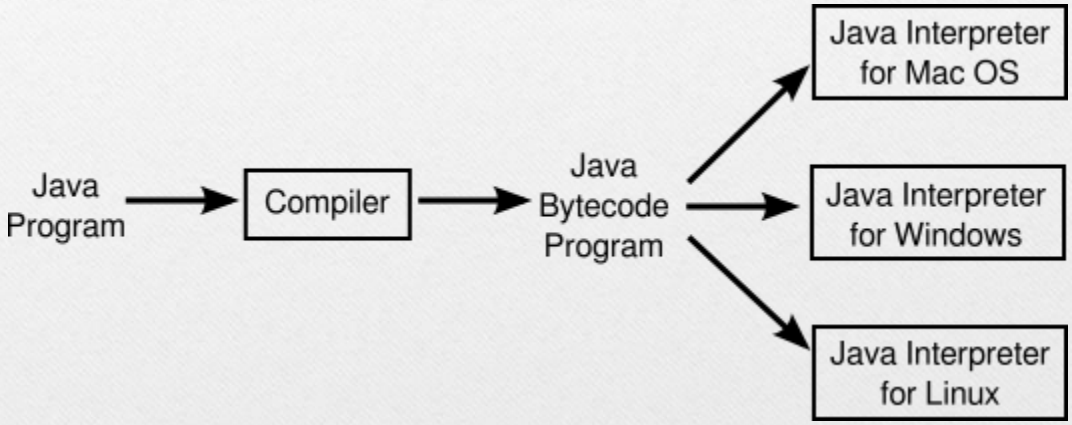
- Multiplataforma. El intérprete suele estar en varios sistemas operativos, así que no tienes que adaptar tu código a una plataforma en concreto.
- Portabilidad. El mismo programa puede llevarse a diferentes plataformas.
- Aumento del rendimiento. Esto es una ventaja en entornos web. Los lenguajes interpretados como JavaScript, se ejecutan en el navegador cliente, lo que hace disminuir la carga de trabajo del servidor web.

Desventaja de interpretados

- Aunque es multiplataforma, es necesario que en la máquina dónde va a funcionar tenga el interprete, ya sea como framework o como máquina virtual.

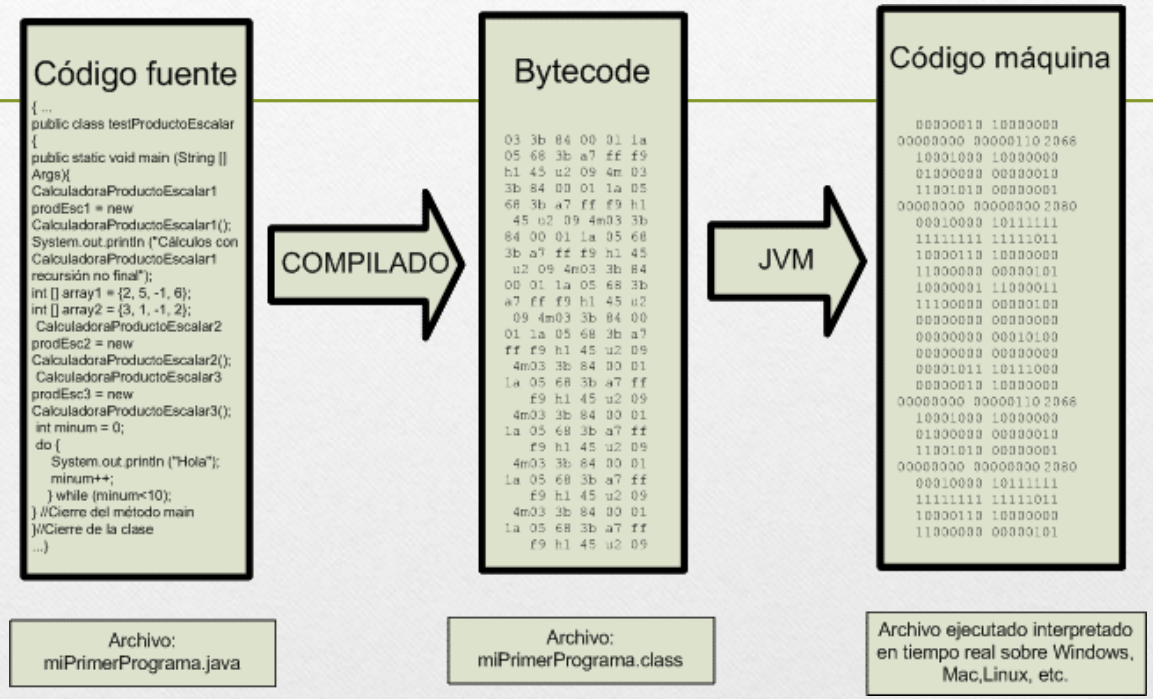
- Dependiendo de cual es el objetivo final, habrá que valorar si es mejor un lenguaje interpretado vs compilado.

Máquina virtual



Fuente:
https://math.hws.edu/eck/cs124/javanotes7/c1/s3.html

Máquina virtual



Fuente:
https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=392:la-maquina-virtual-java-jvm-o-java-virtual-machine-compilador-e-interprete-bytecode-cu00611b&catid=68&Itemid=188

Java es un lenguaje robusto

- Es interpretado. Es convertido a **lenguaje de máquina** a medida que es ejecutado.
- En lugar de los punteros se emplean **referencias a objetos**, los cuales son identificadores simbólicos.
- El gestor de memoria de Java lleva una contabilidad de las referencias a los objetos. Cuando ya no existe una referencia a un objeto, éste se convierte en candidato para la **recogida de basura** (garbage collection).

Lenguaje portable

- El principal objetivo de los diseñadores de Java, fue el de desarrollar un lenguaje cuyas aplicaciones una vez compiladas pudiesen ser inmediatamente **ejecutables en cualquier máquina y sobre cualquier sistema operativo**.
- Por ejemplo, un programa desarrollado en Java en una estación Sun que emplea el S.O Solaris, debería poderse llevar a un PC que utilice S.O. Windows NT.

Lo mas simple posible

- Los diseñadores de Java trataron de mantener las facilidades básicas del lenguaje en un mínimo
- y proporcionar un gran número de extras con las **librerías de clases**.

Lenguaje seguro

- Se pretendía construir un lenguaje que no pudiera acceder a los **recursos** del sistema de **manera incontrolada**.
- Se **eliminó** la posibilidad de manipular la memoria mediante el **uso** de **punteros** evitando así todo acceso ilegal a la memoria.

Crear programas en java

- **Aplicaciones**
  - Escribir el programa fuente en cualquier editor y guardarlo con extensión **.java**
  - **Compilar** el fichero fuente mediante:

    javac miPrograma.java. //**genera** el fichero **.class**
  - **Ejecutar** (interpreta los byte-code) :

    java miPrograma

Características

- Java no tiene funciones definidas fuera de las clases.
- Métodos y variables deben estar definidas dentro de una clase.
- El main debe ir dentro de una clase aunque no pertenezca a ninguna.

Sintaxis

JAVA

Comentarios

// comentarios para una sola línea

/* comentarios de una o más líneas */

Tipos

- Java es un lenguaje con control fuerte de Tipos (*Strongly Typed*).
- Esto significa que cada variable y cada expresión tiene un **Tipo** que es **conocido** en el **momento** de la **compilación**

Tipos primitivos

- **boolean** : Puede contener los valores **true** o **false**.
- **byte** : Enteros. 1 Byte. Valores entre -128 y 127.
- **short** : Enteros. 2 Bytes. Entre -32768 y 32767.
- **int** : Enteros. 4 Bytes. Entre -2147483648 y 2147483647.
- **long** : Enteros. 8 Bytes. Entre -9223372036854775808 y 9223372036854775807.
- **float** : Números en coma flotante. 4 Bytes.
- **double** : Números en coma flotante. 8 Bytes.
- **char** : Caracteres. Tamaño 2 Bytes. Unicode. Desde '\u0000' a '\uffff' inclusive. Esto es desde 0 a 65535

Valores predeterminados

byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
String (o cualquier objeto)	null
boolean	false

Identificadores

- Los identificadores nombran **variables**, **funciones**, **clases** y **objetos**; cualquier cosa que el programador necesite identificar o usar.
- En Java, un identificador comienza con una letra, un subrayado (_) o un símbolo de dólar ($). Los siguientes caracteres pueden ser letras,dígitos,$,_.
- Se distinguen las mayúsculas de las minúsculas y no hay longitud máxima.

Ejemplos

- nombre_usuario
- Nombre_Usuario
- _variable_de_sistema
- $transacción
- var2

Variables

- Una variable es un área en memoria que tiene un nombre y un tipo asociado.
- Es obligatorio declarar las variables antes de usarlas.

tipo_variable nombre **;**

Ejemplos:

```
int i;          // Declaracion de un entero
char letra;     // Declaracion de un caracter
boolean flag;   // Declaracion de un booleano
```

Asignación

- La declaración y la asignación se pueden combinar en una sola expresión:

int i = 5**;**

char letra = 'c'**;**

boolean flag = **false;**

No existen variables globales pues todas deben estar dentro de alguna clase.

Literales caracter

- Un carácter entre apóstrofes (') o bien una secuencia de escape (también entre ').
- Las secuencias de escape están formadas por el símbolo \ y una letra o un número. Algunas secuencias de escape útiles:
  - \n: Salto de línea
  - \t: Tabulador
  - \b: Backspace.
  - \r: Retorno de carro // coloca el cursor al inicio de línea
  - \uxxxx: donde xxxx es el código Unicode del carácter.
  - Por ejemplo \u0027 es el apostrofe (')

Operaciones sobre tipos primitivos

Tipos	Grupo de operadores	Operadores
Enteros	Operadores de comparación que devuelven un valor `boolean`	`< (menor) , <= (menor o igual) , > (mayor), >= (mayor o igual), == (igual), != (distinto)`
	Operadores numéricos, que devuelven un valor `int` o `long`	`+ (unario, positivo), - unario, negativo), + (suma), - (resta),* (multiplicación), / (división), % (resto), ++ (incremento), -- (decremento)`

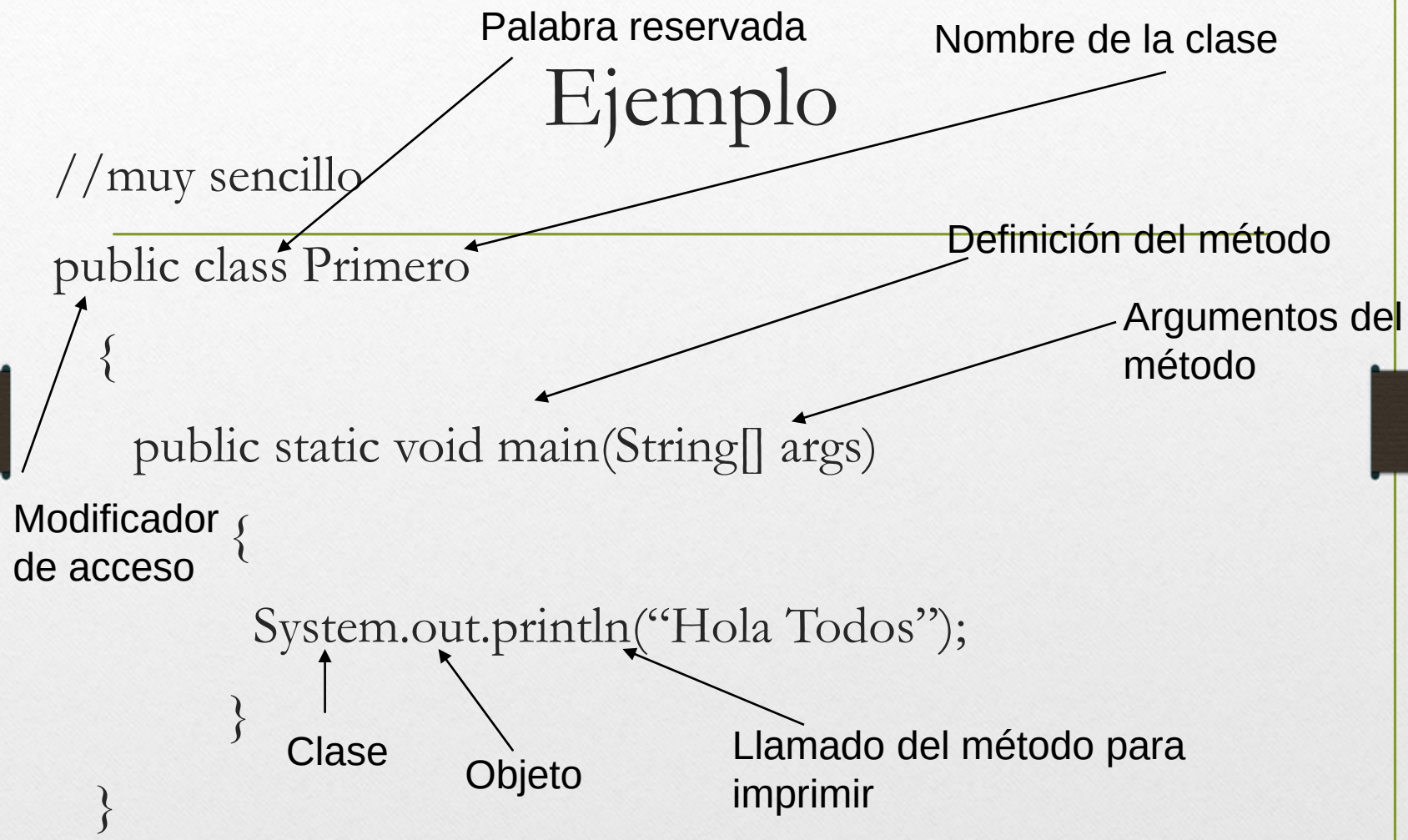
Tipos	Grupo de operadores	Operadores
Coma flotante	Operadores de comparación que devuelven un valor `boolean`	`< (menor) , <= (menor o igual) , > (mayor), >= (mayor o igual), == (igual), != (distinto)`
	Operadores numéricos, que devuelven un valor `float` o `double`	`+ (unario, positivo), - (unario, negativo), + (suma) , - (resta) , * (multiplicación), / (división), % (resto), ++ (incremento), -- (decremento).`
Booleanos	Operadores booleanos	`== (igual), != (distinto), && (AND condicional), || (OR condicional)`

Los operadores de java son los mismos y tienen la misma prioridad y asociatividad que en C

Palabras reservadas

- Abstract, boolean, break, byte, case, catch, char, class, const, continue, default, do, double, else, extends, final, finally, float, for, goto, if, implements, import, instanceof, int, interface, long, native, new, package, private, protected, public, return, short, static, super, switch, this, throw, throws, transient, volatil, while, void, synchronized,

Ejemplo



El archivo se debe llamar Primero.java

Tipos de métodos

- Método de **instancia** es el que se invoca **siempre** sobre una instancia (objeto) de una clase. Ejemplo alumno.getNombre();
- Método de **clase** es aquel que puede ser invocado sin existir una instancia. Se define agregando la palabra clave static antes del tipo en la signatura del método

El método main

El main es el primero en ejecutarse

public: Un método público es accesible desde fuera de la clase.

static: Un método estático es aquel que se puede ejecutar sin una instancia de la clase.

- No tiene sentido que tenga un tipo de devolución distinto de void.

import

import java.util.*; //importa TODAS las clases pertenecientes al paquete "java.util“

import java.io.* //importando clases para entrada y salida (IN/OUT)

import java.math.* //clases utiles para operaciones y valores matematicos

Objeto out para flujo de salida estándar

- Algunos de sus métodos son: println(), print()
- Ejemplos

  System.out.println("Hola Todos");

  System.out.println("Factorial"+facto);

  System.out.print("El valor es %.2f", pago);

Modos de acceso

Modificador /Acceso	Clase	Paquete	Subclase	Todos
public	Sí	Sí	Sí	Sí
protected	Sí	Sí	Sí	No
default	Sí	Sí	No	No
private	Sí	No	No	No

Ejemplo: Clase servicios

```
public class Servicios
   {
    public static void main(String[] args)
           {
            System.out.println("Servicios");
           }
     public  void contratar()
           {
            System.out.println("contratar");
           }
   }
```

Creación de objetos

Una vez que una clase se modela o se define, es posible crear o **instanciar** uno o más objetos que se identifican con dicha clase.

Servicios_Juan
Amplia 150,000 3 años
Contratar() Renovar() Cancelar() Modificar()

Servicios_Teresa
Limitada 100,000 2 años
Contratar() Renovar() Cancelar() Modificar()

Ejemplo: Instanciar objetos

```
public class Servicios{
   public static void main(String[] args) {
      Servicios k = new Servicios();
      System.out.println("Hola Todos!");
          }
          public  void contratar()
             {
               System.out.println("contratar");
             }
}
```

Mensaje

- Es una **instrucción** que se envía a un objeto, el cual se ejecutará al recibirlo;
- Incluye el identificador que contiene la acción a realizar por el objeto.
- Incluye los datos que este necesita para efectuar su trabajo.

Fuente: https://shopf.off67.ml/products.aspx?cname=buz%c3%b3n+de+mensajes&cid=60

Ejemplo

Contratar()

Fuente:https://www.crushpixel.com/es/stock-vector/ringing-phone-cartoon-icon-3738788.html

Fuente: https://hogaresherso.com.mx/seguro-para-el-hogar/

Ejemplo: Mensaje a Contratar

```
public class Servicios{
   public static void main(String[] args) {
      Servicios juan = new Servicios();
      System.out.println("Hola Todos!");
      juan.contratar();
    }
        public  void contratar()
          {
           System.out.println("contratar");
          }
}
```

Practica 4

- Implementa en Java los tres programas en java, cada uno con su clase.
  1. Animales
  2. Personas
  3. Carros

Tarea 5

- Modela las clases: mochilas, juguetes, libros.
- Implementa en Java las 3 clases modeladas.