

Java Inicial

(20 horas)





Temario

1. **Programación Orientada a Objetos**
2. Introducción y Sintaxis Java
3. Sentencias Control Flujo
4. POO en Java
5. Relaciones entre Objetos
6. Polimorfismo, abstracción e interfaces
7. Excepciones
8. Conceptos avanzados



Tema 1

Programación Orientada a
Objetos



Objetivos

1. **Programación Orientada a Objetos**
2. Introducción y Sintaxis Java
3. Sentencias Control Flujo
4. POO en Java
5. Relaciones entre Objetos
6. Polimorfismo, abstracción e interfaces
7. Excepciones
8. Conceptos avanzados

- Historia y origen de la POO
- Características POO
- Elementos POO
- Visibilidad
- Métodos
- Atributos y métodos estáticos
- Relación entre clases
- Herencia
- Polimorfismo
- Abstracción
- Clases finales e internas

■ Introducción

- Su aparición se remonta a 1967 con SIMULA
 - Lenguaje diseñado para hacer simulaciones
- Enfoque diferente del mundo informático.
- Implica:
 - CREACIÓN DE MODELOS DEL MUNDO REAL
 - POO surge de la necesidad de modelizar la realidad
 - en un sistema informático
 - MODELOS TAD'S (programación tradicional) vs POO
 - Programación estructurada
 - En procedimientos
 - Estructura de Datos

■ Características

□ ABSTRACCIÓN

- Proceso mediante el cual se escogen las características esenciales de algo. No importa el “*como*” sino el “*qué*”

□ ENCAPSULACIÓN

- Ocultar la características de una abstracción
- Ocultar el comportamiento interno de la clase

□ MODULARIZACIÓN

- Característica de un sistema que puede descomponerse en un conjunto de módulos relacionados entre sí, pero poco acoplados

□ REUTILIZACIÓN

- Si ya está implementado, porqué tener que inventarlo

■ Elementos POO (1)

CLASE

Descripción de los *datos* y *operaciones* que describen el comportamiento de un cierto tipo de elementos homogéneos

Clase Coche

Marca : string

Nº puertas : entero

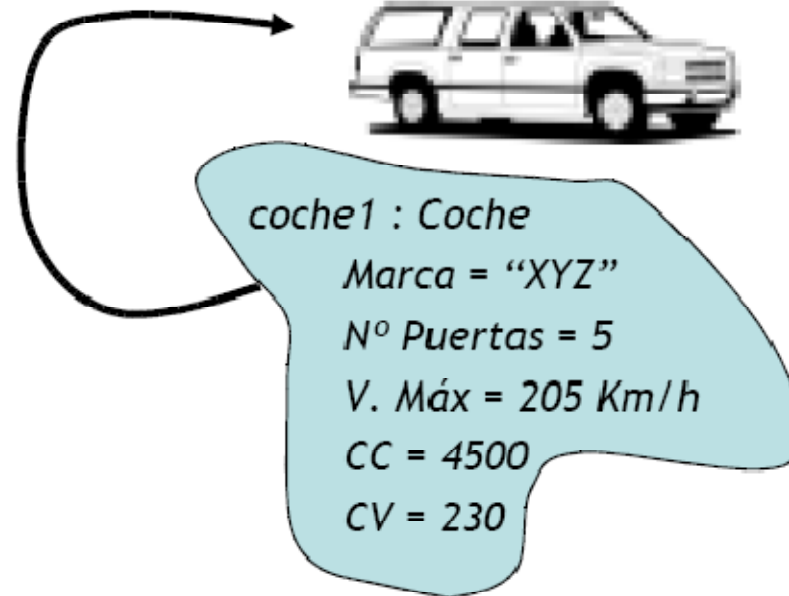
V. Max : entero

CC : entero

CV : entero

• OBJETO

Ejemplar concreto de una clase (instancia).

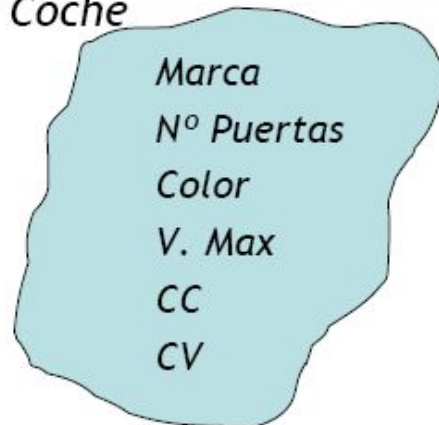


■ Elementos de la POO (2)

- ATRIBUTO

Propiedad común a todos los objetos de una clase

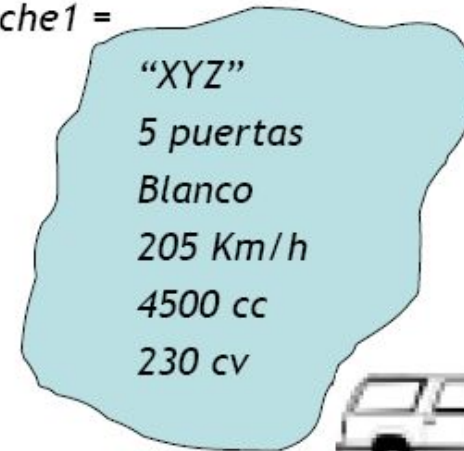
Clase Coche



- ESTADO

Conjunto de los valores de los atributos de un objeto en un determinado momento

coche1 =



■ Elementos de la POO (3)

- METODO

Definición e implementación de las operaciones que pueden realizar todos los objetos de una clase

Clase Coche

Arrancar()

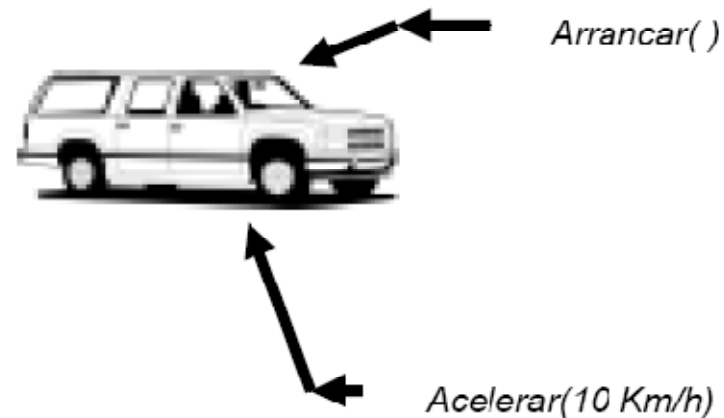
Parar()

Acelerar(Km/h)

Frenar(Km/h)

- MENSAJE

Invocación de un método de una clase sobre un objeto de dicha clase en un instante determinado



■ Elementos de la POO (4)

¿Cómo ACCEDER a los atributos de un objeto?
¿Cómo INVOCAR los métodos de una clase?

La mayoría de los lenguajes O.O. utilizan el operador “punto” (.) para acceder a los atributos y para invocar a los métodos de una clase:

```
coche1.arrancar( )  
fraccion1.sumar(5)
```

VISIBILIDAD

■ VISTA PUBLICA

- Interface entre la clase y el programador final
- Lo que el programador final puede utilizar

■ VISTA PRIVADA

- Operaciones internas a la clase
- El programador final NO puede acceder a ellas directamente

■ VISTA PROTEGIDA

- Operaciones internas a la clase que son accesibles desde sus clases derivadas

VISIBILIDAD - ÁMBITO (II)

LEY ESTRICTA DE DEMETER

“Desde la implantación de un método se tiene acceso a:

- ✓ Los atributos del objeto que recibe el mensaje*
- ✓ Los objetos locales de dicho método*
- ✓ Los parámetros que recibe dicho método*
- ✓ El objeto en sí, pero como un TODO (this)”*

■ Constructores

- TIPOS:

- ✓ Constructor por omisión
- ✓ Constructor por defecto
- ✓ Constructor con argumentos
- ✓ Constructor copia



Coche coche1 = {Marca = "XYZ", N° puertas = 5, V.Max = 205 Km/h,
CC = 4500, CV = 230}



Coche coche2 = {coche1}

■ Métodos operadores

- Aportan una información más conveniente a la hora de expresar ciertas operaciones

fResultado ← *fraccion1.sumar(fraccion2)*

VS

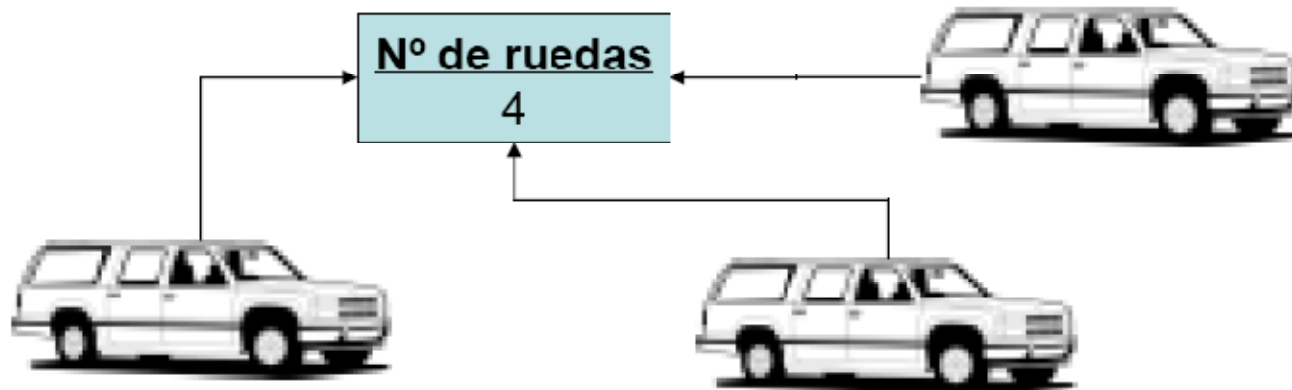
fResultado ← *fraccion1 + fraccion2*

■ Atributos y Métodos Estáticos

- **ATRIBUTO ESTÁTICO:**
 - Es compartido por todos los objetos de la clase
 - Atributo a nivel de clase, no de objeto
- Un método estático **SOLO** puede acceder a atributos estáticos.

■ Atributos estáticos

- Atributo compartido por todas las instancias de la clase



■ Método estático

- Método de clase
- Se puede invocar a través de la clase, sin tener ninguna instancia de ella.
- Un método estático **NO PUEDE** acceder a un atributo no estático
- Un método no estático **PUEDE** acceder a un atributo estático

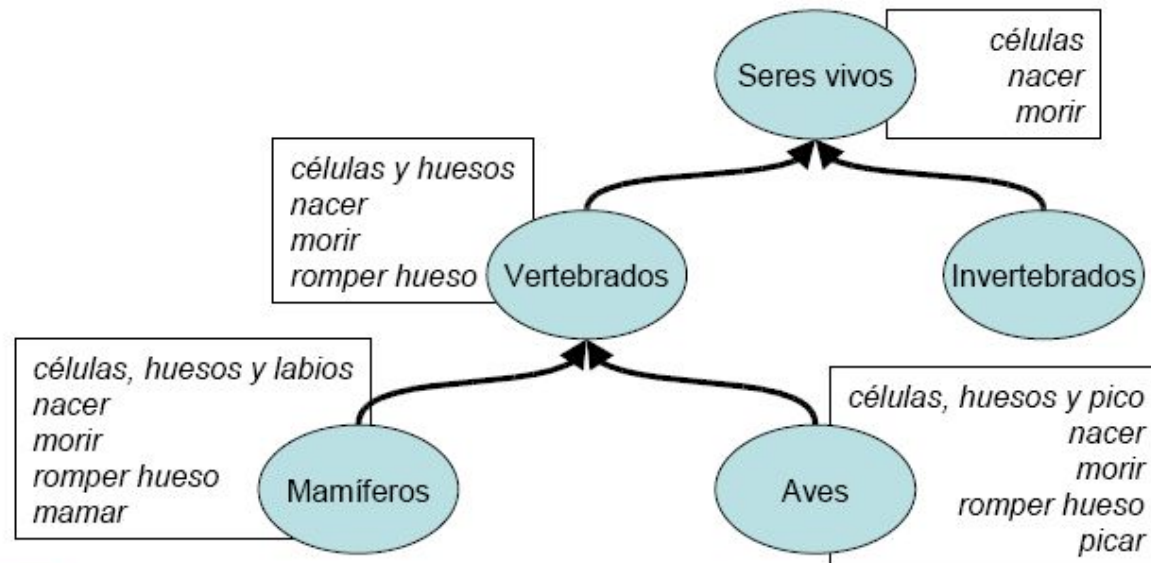
■ Clases

- PROGRAMA O.O.

- Conjunto de objetos que colaboran entre sí para resolver un problema
- Debe ser reflejo de la realidad
- Diseño orientado a responsabilidades: *una clase realiza una determinada tarea y ninguna otra debe inmiscuirse en dicha tarea*

■ Relaciones entre clases (HERENCIA)

- **HERENCIA:** *TRANSMISIÓN DE ATRIBUTOS DE UNA GENERACIÓN A LAS SIGUIENTES*
- **Jerarquía de clasificación:** cada nodo de la jerarquía establece un dominio de elementos, incluidos los nodos padre e hijo



■ Polimorfismo

“Habilidad de un objeto (cosa) de ser visto desde múltiples perspectivas”

- ENLACE ESTÁTICO VS ENLACE DINÁMICO

- E. ESTÁTICO:

- ✓ Tiempo de compilación
- ✓ La expresión siempre devuelve objetos de la misma clase

- E. DINÁMICO:

- ✓ Tiempo de ejecución
- ✓ En la ejecución se determina a que clase pertenece el objeto → *EXPRESIÓN POLIMÓRFICA*

■ Clases Abstractas

- *Define el comportamiento (métodos) de una clase sin ser posible realizar su implementación por diversos motivos*
- Por tanto:

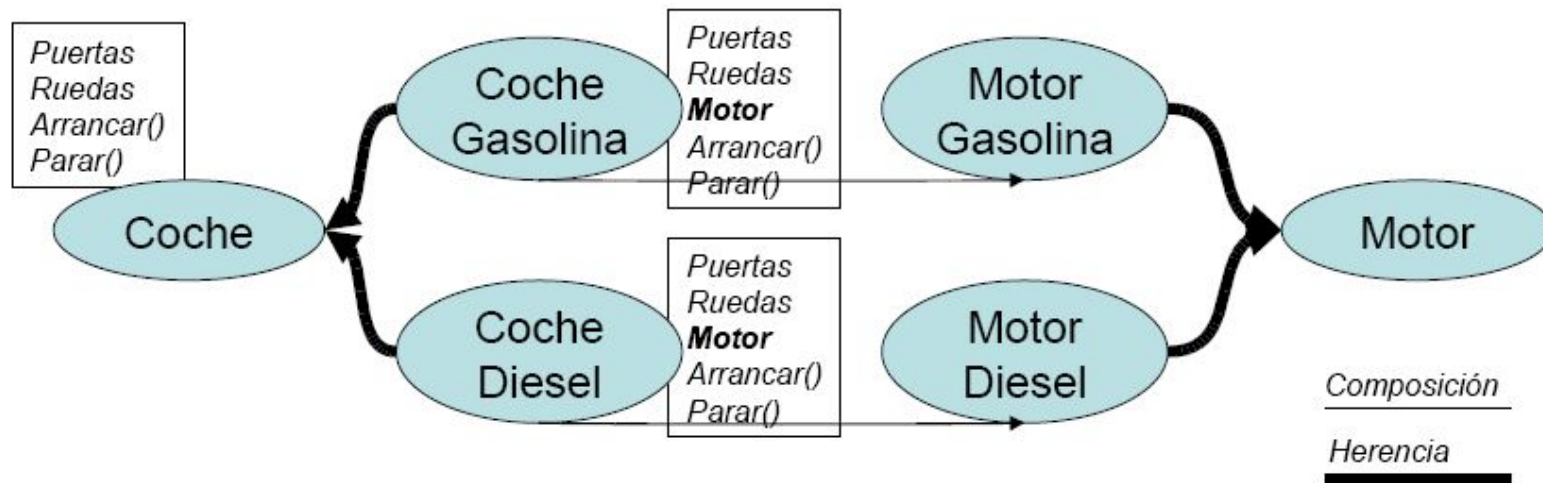
***NO SE PUEDE INSTANCIAR NINGÚN OBJETO
DE UNA CLASE ABSTRACTA
PORQUE FALTA ALGUNA FUNCIONALIDAD***

■ Clases Abstractas - Ejemplo

¿Cómo puede arrancar un coche si no tiene motor?

Entonces, ¡NO LO ESPECIFICAMOS!

Pero, ¿podemos decir que TODOS LOS COCHES ARRANCAN?



■ Clases Finales y Clases Internas

- Una clase FINAL es aquella de la que no se puede heredar
- Una clase INTERNA es aquella definida dentro de otra clase, siendo sólo accesible desde la que se define



Conclusiones

1. Programación Orientada a Objetos
2. Introducción y Sintaxis Java
3. Sentencias Control Flujo
4. POO en Java
5. Relaciones entre Objetos
6. Polimorfismo, abstracción e interfaces
7. Excepciones
8. Conceptos avanzados

- Historia y origen de la POO
- Características y elementos
- Visibilidad
- Métodos
- Relación entre clases
- Polimorfismo
- Abstracción
- Clases Finales e internas



Referencias

- Introducción al Lenguaje Java:
<http://java.sun.com/new2java/gettingstarted.jsp>

