2. Programación Orientada a Objetos (POO)
2.1. Conceptos fundamentales

Programación Modular y Orientada a Objetos

Felipe Ibañez y Juan Miguel Lopez
felipe.anfurrutia@ehu.es
juanmiguel.lopez@ehu.es
Dpto. de Lenguajes y Sistemas Informáticos
UPV/EHU

Conceptos fundamentales de la POO

- Objetos
- Atributos y métodos
- Encapsulación
  - ocultación de la información
- Clases
  - Clases vs objetos
- Programa Orientado a Objetos
- Mensajes y métodos
- Tipos de datos

¿Qué es un objeto?

- Los objetos son/representan/modelan cosas: reloj, avión, empleado, etc.
- Los objetos pueden ser simples o complejas
- Los objetos pueden ser reales o imaginarios

Atributos (abstracción de datos)

- Describen valores o características de los objetos

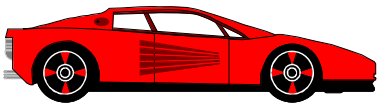
•marca
•color
•potencia
•velocidad máxima
•carburante

Constantes

•velocidad
•aceleración
•capacidad de combustible

Variables

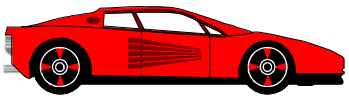
- Permiten definir el estado del objeto u otras cualidades



coche1: Coche	
marca	"Ferrari"
color	"rojo"
potencia	550
velocidadMax	300
carburante	"gasolina"
velocidad	0
...	

Métodos (abstracción funcional)

- Acciones que puede realizar un objeto



- arrancar motor()
- parar motor()
- acelerar(): real
- frenar(): real
- girar a la derecha(grados: real)
- girar a la izquierda(grados: real)
- cambiar marcha(nueva marcha: entero)

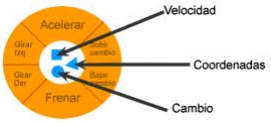
Tipo de dato

Métodos

argumentos o parámetros

- Los métodos pueden devolver un valor al acabar su ejecución (p.ej. frenar): valor de retorno

Encapsulación



- Se denomina **encapsulamiento** al ocultamiento del estado de un objeto de manera que sólo se puede cambiar mediante los métodos definidos para ese objeto.
- La clave está en el envoltorio del objeto:
  - Nos interesa **qué** es lo que puede hacer (el interfaz)
  - No es necesario saber **cómo** lo hace (implementación)
- El encapsulamiento se logra mediante:
  - La abstracción y
  - La ocultación de la información
- La **clase** representa la encapsulación de una abstracción (datos y comportamientos).

Ocultación de la información (I)

- Permite definir qué partes del objeto son visibles (el interfaz público) y qué partes son ocultas o protegidas (privadas)

Print *es la interfaz publica de la impresora*
La implementación, es decir, cómo se imprime el documento que hemos enviado a la impresora es privado.

Ventajas

El objeto puede cambiar y su interfaz público ser compatible con el original. Esto facilita la reutilización de código

Ocultación de la información (II)

- UML permite asociar tres niveles de protección diferentes a cada miembro de la clase:
  - Miembros públicos (+). Sin ningún tipo de protección especial
  - Miembros privados (-). Inaccesibles desde el exterior de la clase
  - Miembros protegidos (#). Similares a los privados aunque se permite su acceso desde las clases descendientes*

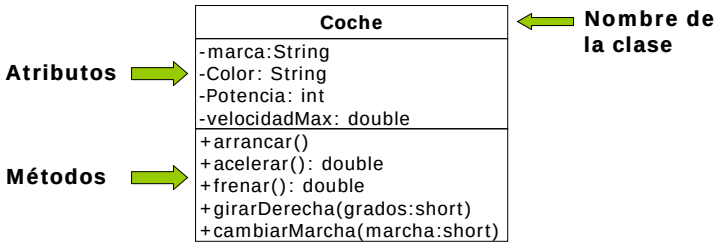
NombreClase
+atributoPublico: Tipo -atributoPrivado: Tipo #atributoProtegido: Tipo
+metodoPublico() -metodoPrivado() #metodoProtegido()

Normalmente en una clase:
- Los atributos son **privados**
- Los métodos son **públicos**
- ***Puede haber métodos privados***
- **Es peligroso tener atributos públicos!!!!**

*** Las veremos más adelante, al estudiar el mecanismo de la herencia**

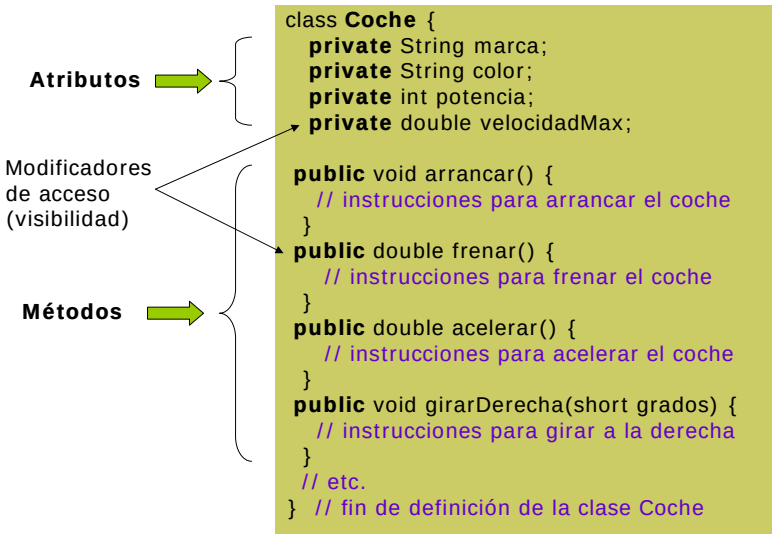
Clases

- Representan un tipo particular de objetos
  - Objetos con características y comportamiento similar
  - Categorías/clasificación de objetos
- La definición de clases determina:



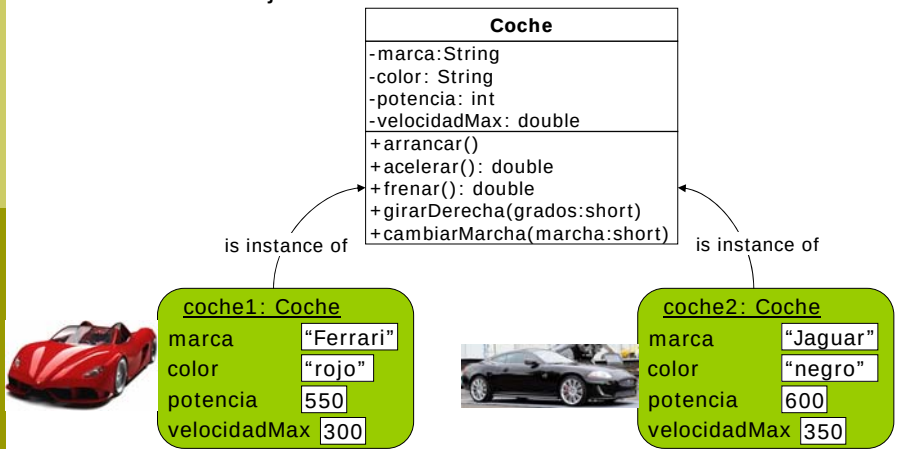
- Son los elementos básicos de la POO,
  - que consiste en escribir código de clases de objetos

Definición de clases en Java (Coche.java)



Clases vs objetos

- De cada clase pueden crearse/instanciarse múltiples objetos
- Cada objeto tiene valores propios asignados a los atributos: **estado** del objeto

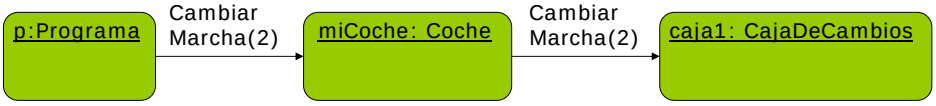


Un programa en POO

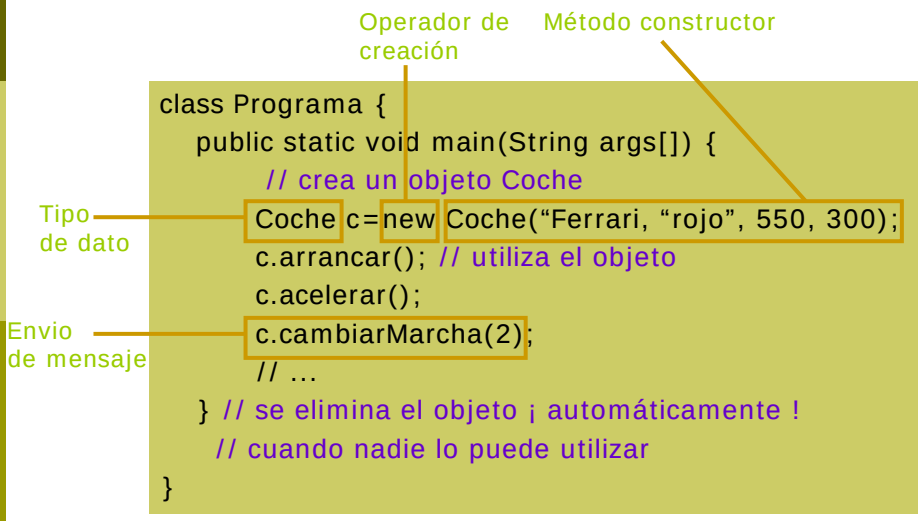
- Un programa consta de un conjunto de instancias o ejemplares de objetos (object instances) y un flujo de control principal (main)
- Durante la ejecución del programa:
  - Los objetos se crean y se destruyen
    - Gestión dinámica de la memoria
  - Los objetos se comunican entre ellos diciendose lo qué deben de realizar mediante el envio de mensajes

Mensajes

- Los objetos se comunican e interaccionen entre sí por medio de mensajes
- Si un objeto desea que otro objeto haga algo le envía un mensaje que puede tener información adicional en forma de parámetros
- Cuando un objeto recibe un mensaje ejecutará un método u operación
- Componentes de un mensaje:
  - Objeto destinatario del mensaje (*miCoche*)
  - Método que se debe ejecutar como respuesta (*cambiar marcha*)
  - Parámetros necesarios del método (*segunda*)
  - Sintaxis de Java: *miCoche.cambiarMarcha(2)*



Un programa en Java



Tipos de datos

- Indican la naturaleza de los datos
  - que se pasan como parámetro o que devuelven los métodos
  - de los atributos de los objetos
- Hay algunos básicos o primitivos (dependen del lenguaje de programación)
  - int
  - boolean
  - double
  - String
  - ...
- Y otros los definen las clases
  - Objetos de una clase determinada
    - Definida por el usuario (p.ej. Coche)
    - De una librería (p.ej. java.util.ArrayList)

2 roles de programador

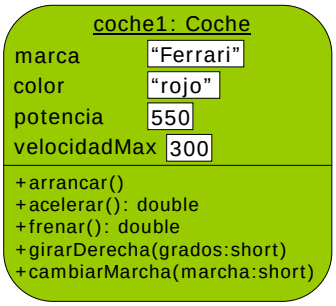
- Programador cliente:
  - Utiliza las clases como caja de herramientas para desarrollar rapidamente aplicaciones
  - No se preocupa de cómo está implementado
  - No puede acceder a la parte oculta
- Creadores de clases:
  - Definen nuevos tipos de datos
  - El objetivo es ofrecer al *programador cliente* justo lo que necesita (el interfaz público)
  - y oculta todo lo demás (la abstracción de datos y la implementación), para tener flexibilidad en los cambios internos, y reducir los fallos del programa

Resumen: Clases vs objetos

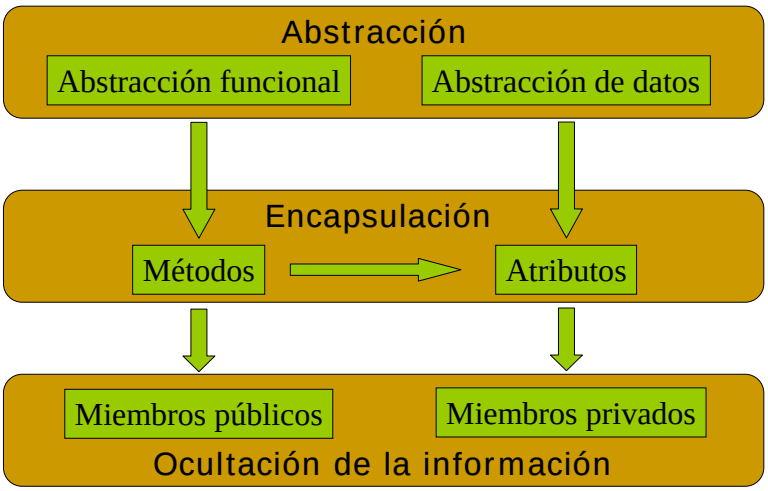
- Una **clase** es una entidad abstracta
- Es un tipo de clasificación de datos
- Define el comportamiento y atributos de un grupo de objetos con características (propiedades y comportamiento) similares

Coche
-marca:String -color: String -potencia: int -velocidadMax: double
+arrancar() +acelerar(): double +frenar(): double +girarDerecha(grados:short) +cambiarMarcha(marcha:short)

- **Un objeto** es una instancia de una clase
- Un objeto se distingue de otros miembros de la clase por sus atributos.



Resumen: principios POO



Bibliografía

(1) "Objects First with Java" (3ª edición) D.J. Barnes & M. Kölling. Prentice Hall, 2006

(2) "Programación Orientada a objetos con Java" (3ª edición) D.J. Barnes & M. Kölling. Prentice Hall, 2007

(3) "Thinking in Java". http://www.mindview.net/Books/TIJ/index_html

(4) "Construcción de Software Orientado a Objetos" (2ª edición). B. Meyer. Prentice-Hall, 1999