Bloque II. Elementos del lenguaje de programación Java

- 1.Introducción a los lenguajes de programación
- 2. Estructura de un programa
- 3. Datos y expresiones simples
- 4. Instrucciones de control
- 5. Entrada/salida simple
- 6. Arrays, secuencias y tablas
- 7. Métodos

2. Estructura de un programa

- 2.1. Introducción al lenguaje Java
- 2.2. Concepto de clase y objeto
- 2.3. Estructura de un programa

2.1. Introducción al lenguaje Java

UNIVERSIDAD DE CANTABRIA

Desarrollado por la empresa Sun Microsystems en 1995

Recibe una amplia aceptación

Objetivos generales:

- ***WORA*: (*W*rite *O*nce, *R*un *A*nywhere)**
- **Portabilidad sin necesidad de recompilar**
- **Programación distribuida**
- **Orientado a objetos**
  - **abstracción de datos**
  - **modularidad, encapsulado, y ocultamiento de información**
  - **herencia y polimorfismo**

Introducción al lenguaje Java

UNIVERSIDAD DE CANTABRIA

Objetivos generales (cont.):

- **Más fiable y seguro que C o C++:**
  - **memoria dinámica automática, que evita los punteros explícitos**
  - **tipificación estricta**
  - **comprobación automática de tamaños de variables**
- **Sintaxis similar a la del C/C++**
- **Concurrencia integrada en el lenguaje**
- **Excepciones declaradas**
- **Interfaz gráfica integrada en el lenguaje**

La versión actual es la 1.6

2.2. Concepto de clase y objeto

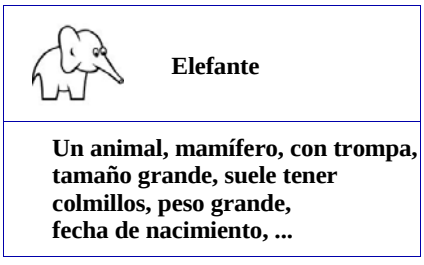
Los programas Java se construyen mediante *clases*

Una *clase* representa una definición de un módulo de programa

- A partir de ella se pueden crear muchos *objetos*
- Cada uno de ellos se dice que es una instancia de la clase

Diferencia entre clase y objeto

En la clase se definen las características comunes de un conjunto de objetos

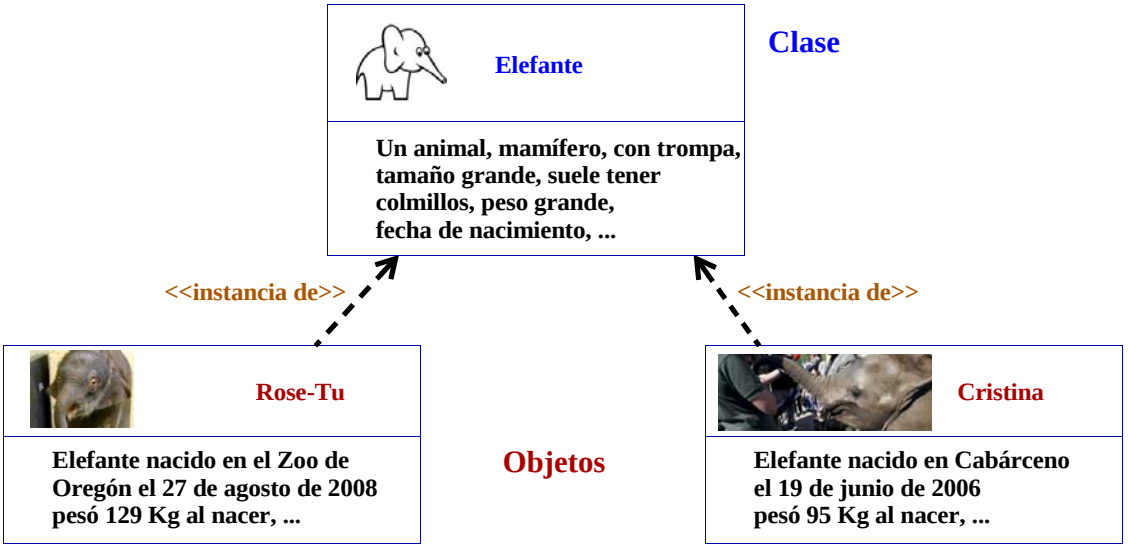


Clase

Define las características de todos los elefantes

Diferencia entre clase y objeto

Las instancias de una clase son objetos concretos que tienen las características de la clase y valores concretos de sus datos



Concepto de clase y objeto

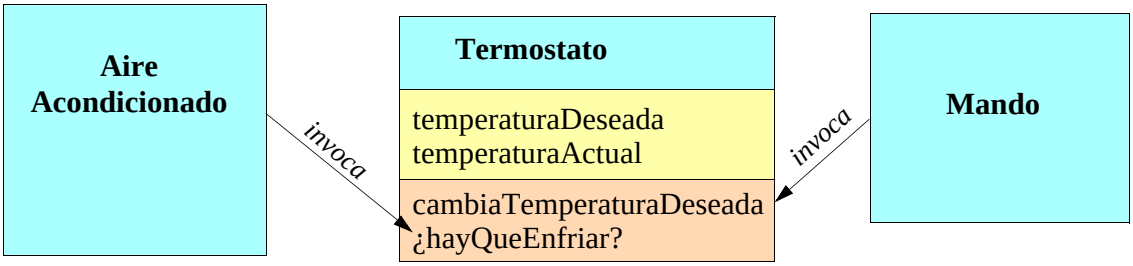
Un *objeto* es un elemento de programa que se caracteriza por:

- ***atributos***: son datos contenidos en el objeto, y que determinan su estado
- ***operaciones*** o ***métodos***: son acciones con las que podemos solicitar información del objeto, o modificarla
  - Están compuestas por secuencias de instrucciones que operan con los atributos
  - y pueden invocar operaciones de otros objetos

Ambos, atributos y métodos, se llaman *miembros*, y se definen en la clase

Se intenta siempre corresponder los objetos de un programa con objetos del problema que éste resuelve

Ejemplo, sistema con un termostato



Atributos del termostato

- temperatura deseada
- temperatura actual (que lee de un termómetro interno)

Métodos del termostato

- cambia la temperatura deseada
- saber si hay que enfriar o no

Ejemplo: gráficos de funciones de una variable:

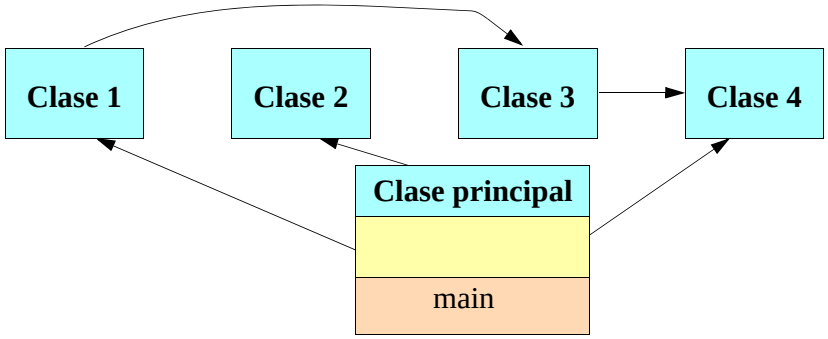
Los gráficos se representan con una clase de objetos, y cada objeto es un gráfico concreto

- ***Atributos*** **del gráfico:**
  - el título
  - los puntos del gráfico (X,Y) y el número de puntos
  - títulos de los ejes
- ***Operaciones*** **a realizar con el gráfico:**
  - crear un gráfico nuevo
  - añadir un punto
  - pintar el gráfico
  - cambiar el título
  - cambiar títulos de los ejes

2.3. Estructura de un programa

Un programa java es un conjunto de clases, donde una de ellas es especial en dos aspectos

- **contiene una operación llamada `main`**
- **habitualmente no se crean objetos de esta clase**
  - **el sistema operativo invoca el método `main` al ejecutar el programa**



Estructura de una clase

El caso más sencillo es el de un programa con una sola clase, que tiene esta estructura:

```
import clases_importadas;
public class Nombre {
   atributos
   operaciones
}
```

Las clases importadas son aquellas pertenecientes a paquetes externos

- **un *paquete* es una colección de clases**

En una clase principal, una de las operaciones se llama `main`

Estructura de una operación

Las operaciones o métodos (como `main`) tienen la siguiente estructura:

```
descriptores valor_retornado nombre (argumentos)
{
    declaraciones;

    instrucciones;
}
```

Las declaraciones declaran datos (variables o constantes), clases y objetos que se van a usar en las instrucciones

Las instrucciones representan cálculos que se hacen con esos datos.

Ejemplo de programa

Por ejemplo, el siguiente programa pone un mensaje en la pantalla:

```
public class Hola {

  /** este es el método principal */

  public static void main(String[] args) {
     // No hay declaraciones
     System.out.println("Hola, Que tal estas?");
  }
}
```

Explicación:

- `public`: el método se puede usar desde fuera
- `static`: el método pertenece a la clase (no a los objetos de la clase)
- `void`: no retorna nada
- `String[] args`: es el argumento, datos que se pasan a la operación
- `System`: es una clase predefinida que representa al computador
- `out`: es un objeto de la clase `System`, predefinido: representa la pantalla
- `println`: método para poner un texto en la pantalla

Algunos comentarios

- formato libre
- *sangrado*: margen del texto, que se usa para remarcar la estructura del programa
- *comentarios*:
  - de documentación (`/** */`),
  - normales, con principio y final (`/* */`)
  - normales, hasta fin de línea (`//`)
- uso de "`;`" para finalizar declaraciones e instrucciones
- uso de `{}` para definir los contenidos de clases y métodos

Algunos comentarios

- *nombre de la clase*: primera letra con mayúscula y demás minúsculas, excepto inicios de palabra
- las declaraciones e instrucciones pueden mezclarse en cualquier orden
  - pero es habitual poner primero las declaraciones, y luego las instrucciones