Introducción a la Programación Orientada a Objetos

Unidad 1 - Objetos y Clases


MARK
CREATED facebook
Programar es una de las pocas cosas en el mundo que puedes hacer

Unidad 1: Clases y Objetos

- Logro: Al finalizar la Unidad 1 el alumno verifica los conceptos de clases y objetos usando pruebas unitarias.

Agenda

- Definición de Programación Orientada a Objetos (POO).
- Clases y Objetos.
- Terminología.
- Encapsulamiento.
- Ventajas y Desventajas de la POO.
- Conclusiones.

Definición de POO

Objeto

Es la unidad fundamental en la programación orientada a objetos. Un objeto es una entidad que sirve como contenedor de datos y además controla el acceso a esos datos.

Asociado con un objeto viene un grupo de atributos que son características que pertenecen y definen al objeto.

Además asociado con un objeto tenemos una serie de funciones que definirán el comportamiento de un objeto. A estas funciones se les llama métodos.

Definición de POO

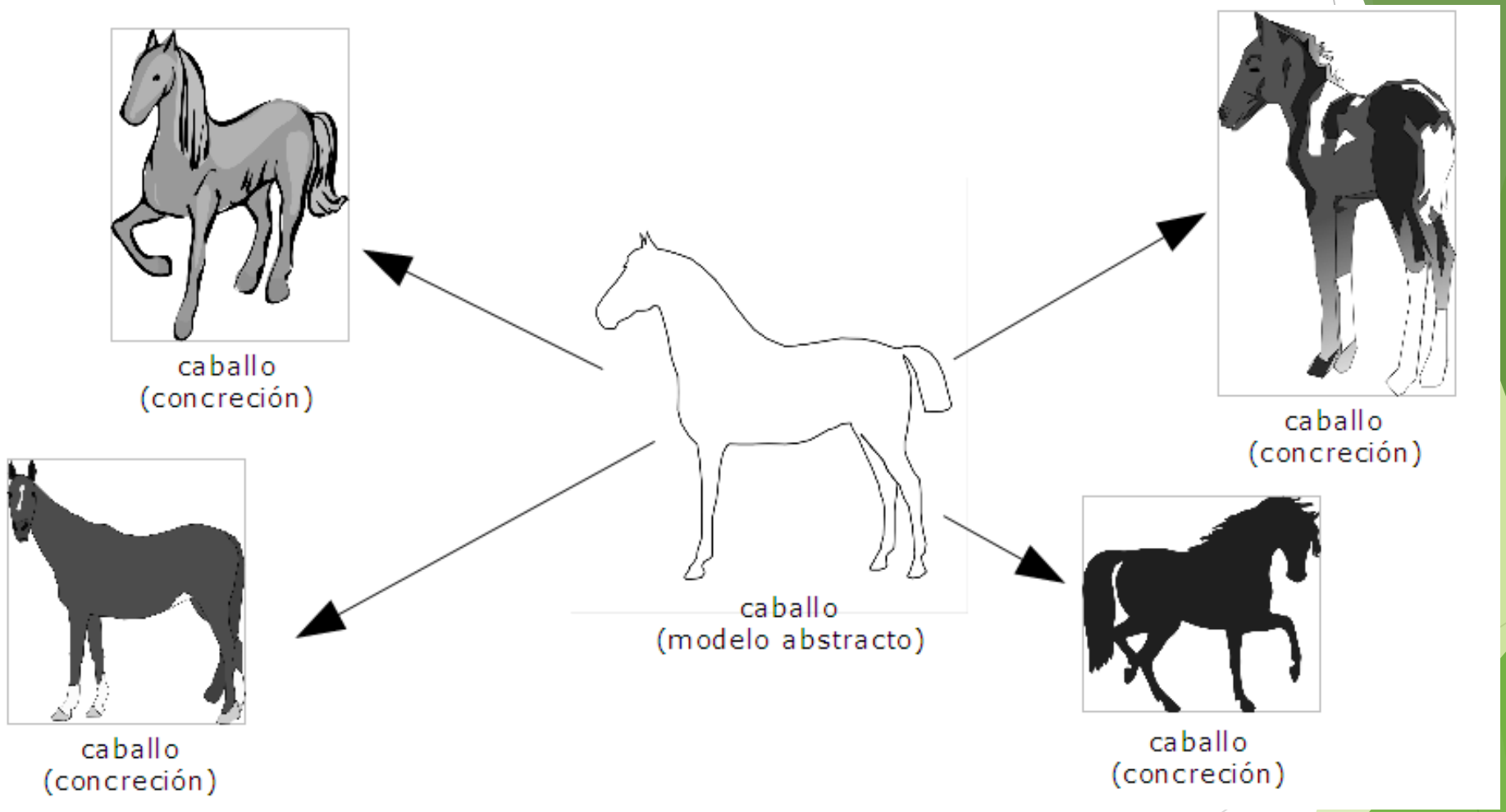
Programación Orientada a Objetos:

Es un paradigma de programación que se basa en los objetos , las relaciones que se establecen entre ellos y los mensajes que se envían unos a otros cooperando de manera conjunta para lograr cumplir con los requisitos del programa.

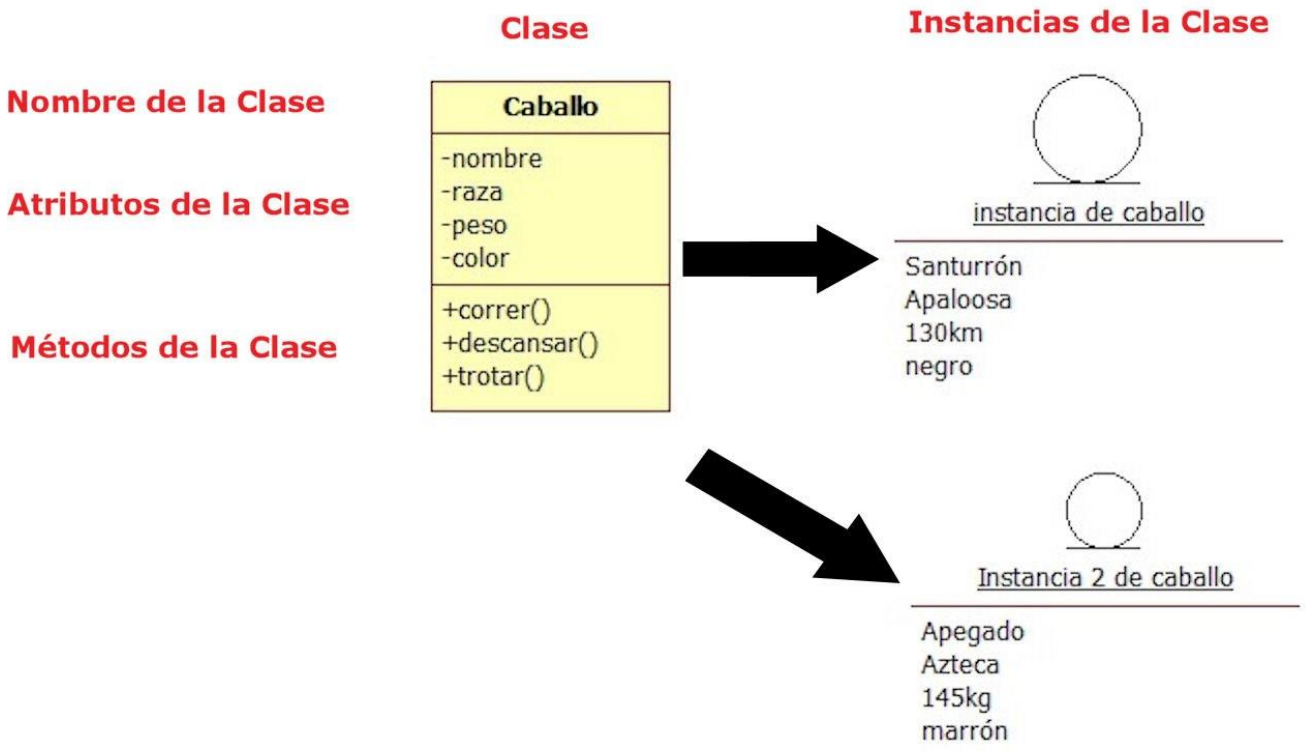
Definición de POO

Clase:

Un objeto se considera una instancia o manifestación del objeto clase (llamado símplemente: la clase) que es un plano de la estructura que tiene un objeto. Debido a que este plano sirve para determinar la estructura de cualquier objeto que pueda ser clasificado como perteneciente a dicha clase se dice que un objeto x es instacia de la clase x.


caballo
(concreción)
caballo
(concreción)
caballo
(concreción)
caballo
(modelo abstracto)
caballo
(concreción)
caballo
(concreción)

Así por ejemplo un objeto instancia de la clase Caballo tiene la estructura que hemos definido que tendrán los objetos del tipo Caballo.



Historia de POO

Origen:

El paradigma de la programación orientada a objetos data de los años sesentas. Sin embargo su uso se popularizó y extendió en los años noventas. Posesionándose hoy en día como el paradigma de programación más usado. Cabe resaltar que no es el único paradigma de programación pues existen muchos otros como la Programación Orientada a Aspectos, Programación Orientada a Eventos, Programación Estructurada, Programación Funcional, etc.

Es preciso señalar que el paradigma de la Programación Orientada a Objetos toma principios de la Programación Estructurada y de la Programación Modular.

Lenguajes Orientados a Objetos

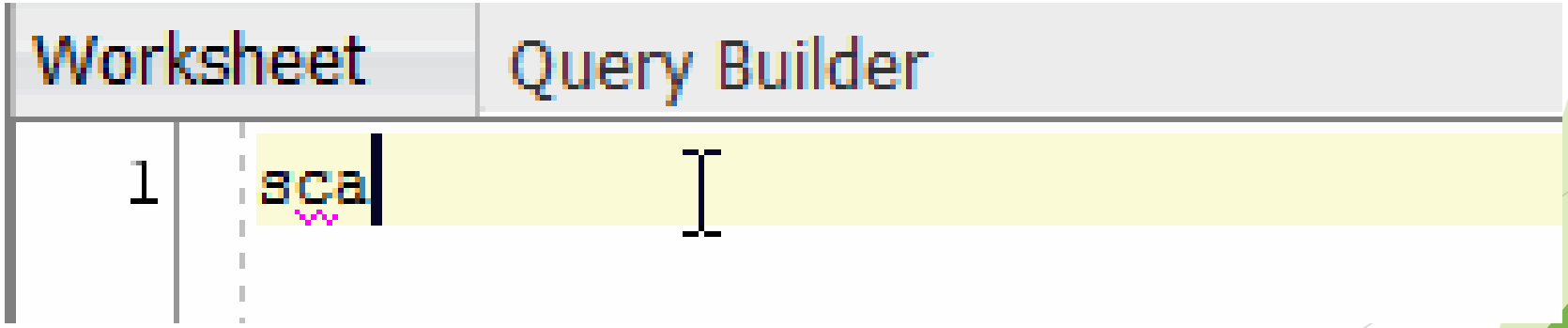
Java, Ruby, Python , C++, PHP entre otros permiten el uso del paradigma de Programación Orientada a Objetos. Cabe resaltar que el uso de uno de estos lenguajes no asegura el uso correcto de la POO pues depende de la correcta aplicación de sus principios.

```
object Persona
 attribute + nombre = "Juan";
 attribute + apellidos = "Nadie";
 attribute + telefono = "906414141";
 attribute + edad = 18;
 attribute + direccion;

 method + toString()
 {
  return nombre.concat( apellidos );
 }
endObject
```

Lenguajes Orientados a Objetos

Por otro lado lenguajes como Prolog, Haskell, Miranda o SQL no son lenguajes orientados a objetos pues tienen otros objetivos como la programación funcional (mediante funciones lógico matemáticas) o el manejo de datos en una base de datos relacional como en el caso de SQL



POO y el mundo real

Parte del éxito del paradigma de la Programación Orientada a Objetos es el hecho de clasificar los objetos de manera muy similar a como lo hacemos en el mundo real. Esto es, a partir de las características de los objetos que percibimos. De esa manera podemos saber que un auto es un vehículo mientras que el perro es un animal.

En ese ejemplo Vehículo sería la clase y Carro el objeto de la clase vehículo. De igual manera Animal sería la clase y Perro el objeto de la clase Animal.

Una clase en Ruby

```ruby
class Caballo

  def initialize(nombre, raza, peso, color)
    @nombre = nombre
    @raza =  raza
    @peso = peso
    @color = color
  end

  def correr
    return "Estoy corriendo a toda velocidad"
  end

  def descansar
    return "Estoy descansando"
  end

  def trotar
    return "Troto a un paso continuo"
  end
end
```

Una clase en Ruby

Una vez creada la clase, que funciona como una plantilla, podemos crear todos los objetos que necesitemos. En este caso todos los objetos que creemos tendrán la misma estructura que hemos definido en la clase.

Una vez creadas las clases tenemos acceso a los atributos y métodos que hemos definido.

```
22
23 caballo01 = Caballo.new("Pipo", "Pinto", 132, "Negro")
24 caballo02 = Caballo.new("Gandalf", "Árabe", 150, "Blanco" )
25 puts caballo01.correr
26 puts caballo02.descansar
```

Terminología Básica

Atributos: Son las características que definen al objeto (nombre, edad, precio).

Métodos: Son el código que define lo que puede hacer un objeto determinado (correr, calcular promedio).

Mensaje: Es la forma en la que un objeto le dice a otro que quiere que haga algo. (caballo01.correr). Es la forma en la que se comunican los objetos.

Programa Orientado a Objetos: Conjuto de objetos que se comiunican mediante mensajes y generan un comportamiento mediante sus métodos.

Encapsulamiento

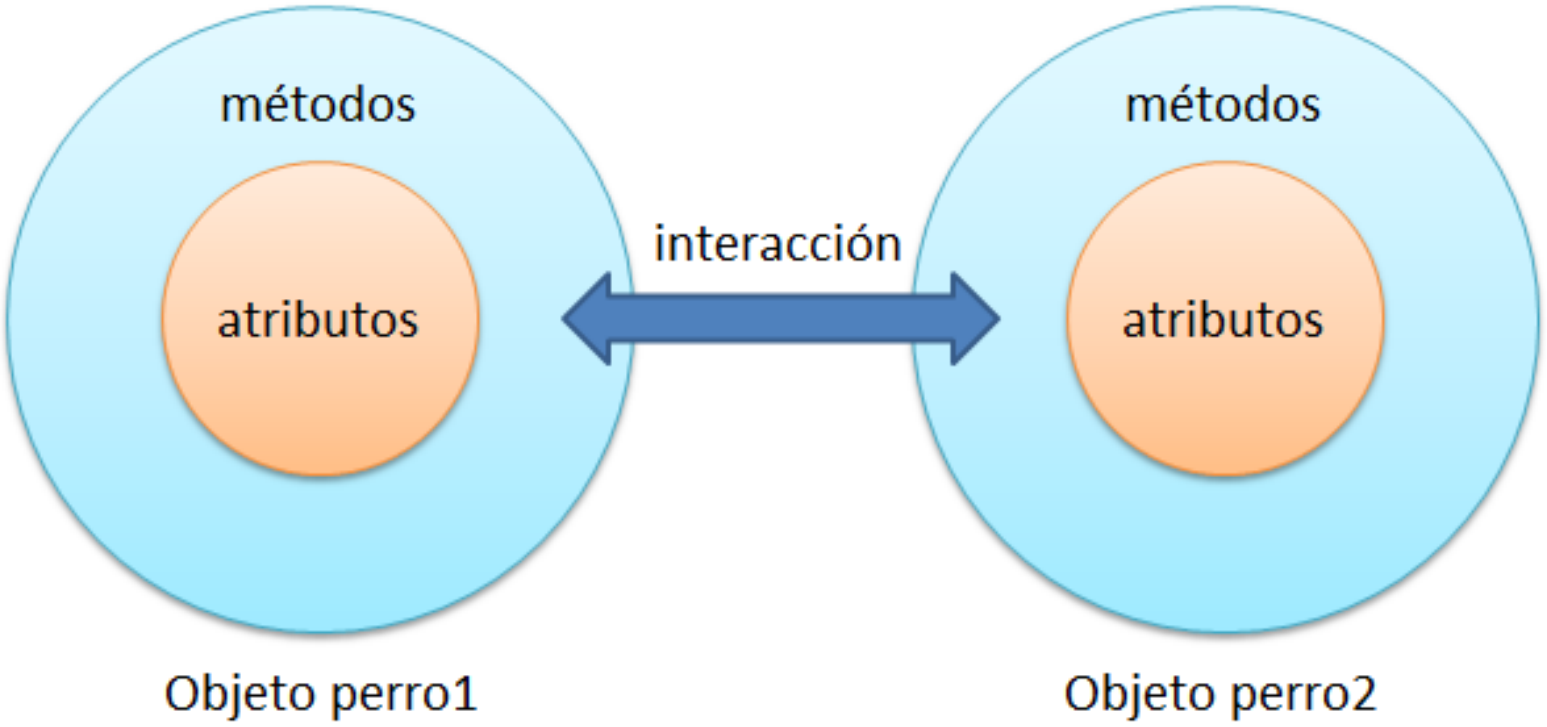
Imaginemos una medicina en cápsula como la Aspirina. Nosotros no necesitamos conocer la composición químicas e ingredientes de la Aspirina para usarla. Símplemente conocemos que sirve para quitar el dolor de cabeza. Si nosotros pudiéramos acceder a los componentes de la Aspirina y los pudiéramos modificar podríamos hasta causar la muerte de alguien.

De igual manera en las clases no es necesario que las demás clases sepan cuáles son los atributos y mucho menos manipularlos. Es por eso que se considera que los atributos deben ser privados (nadie accede directamente a ellos). Aún así necesitamos hacer que la clase haga lo que queremos que haga. Para ello lo que hacemos es acceder a su comportamiento, esto lo hacemos mediante los métodos (que suelen ser públicos).

Es así que si quiero, por ejemplo, saber el nombre de un caballo no se accede directamente al nombre sino por medio de un método encargado de enviarme el nombre del caballo. De igual manera si quiero cambiarle el nombre porque lo puse mal no accedo directamente al atributo, lo que hago es acceder a él mediante un método que me permita modificar el nombre.

Si todas las clases pudieran acceder a los atributos de otras clases directamente todo podría ser un desorden (y causar la muerte del sistema de igual manera que la Aspirina con las personas si modificamos sus ingredientes). Nosotros decidiremos qué atributos son accesibles y cuáles no.

Finalmente , cuando unimos los atributos y los métodos en una clase entonces estamos encapsulando para manejarlo todo como una unidad. En el ejemplo anterior sería Caballo. Esta clase contiene los atributos que definen qué es un caballo para mi sistema y los métodos que definen qué es lo que puede hacer un caballo.



Ventajas y desventajas de la POO

La principal ventaja para los desarrolladores está en la gran cantidad de código que se ahorra.

Por otro lado, al ser una forma de representar el mundo real en un sistema tal cuál lo vemos en nuestro día a día, permite un diseño muy comprensible y descriptivo.

Reusabilidad de código

Capacidad de escalar el código ante el crecimiento de los procesos.

Velocidad de derrarollo

Ventajas y desventajas de la POO

Desventajas: Si bien las ventajas son mucho mayores que las desventajas es importante conocer algunas de las desventajas situacionales.

Para proyectos simples y pequeños muchas veces es más el trabajo que toma crear las clases que lo que se programa para lograr el objetivo. Hay que tomar en cuenta que POO no es el único paradigma de programación.

Aprender a abstraer un elemento del mundo real para crear una clase es un proceso complejo que requiere práctica.

Las abstracciones de un desarrollador no necesariamente coinciden con las de otros. Sin embargo ambos planteamientos pueden ser correctos a pesar de ser diferentes.

Conclusiones

La tecnología ha invadido todos los campos profesionales. En el caso de la ingeniería se ha vuelto indispensable saber programar.

La POO permite representar el mundo real en un sistema.

Se pueden crear sistemas más complejos con menos código.

Tareas como escalar el software y darle mantenimiento se facilitan.

No es el único paradigma de programación pero es, actualmente, el más utilizado por sus ventajas.