



Programación en Lenguaje Ruby

POO

Delio Tolivia





Índice

Objetos

Atributos y métodos

Variables/Objetos

Clases y Objetos

Clases

Características

Encapsulación

Herencia

Polimorfismo

Sobrecarga y sobrescritura

Ventajas

Objetos

- En POO un **objeto** representa cualquier cosa
- Puede ser algo físico como una bicicleta o algo abstracto como una idea.
- Los objetos tienen un **estado** y una **conducta**

Objetos: Ejemplos

bicicleta

color=negra
velocidad=0
marcha_actual=5
presion_ruedas=8 bar

.....

cambiar_marcha
cambiar_color
ver_presión
inflar_rueda
desinflar_rueda

.....

idea

texto=salir a cenar
autor=pepe
fecha=hoy

.....

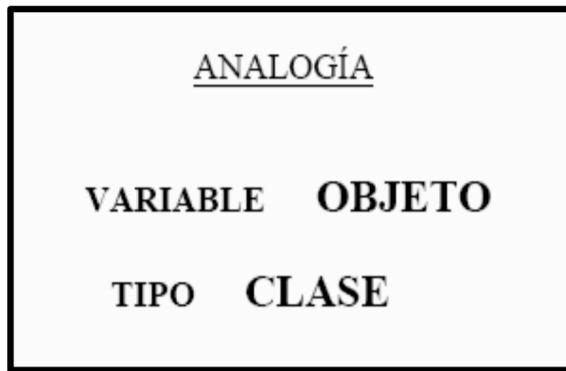
crear
cambiar-texto

.....

Atributos y métodos

		Bicicleta	Idea
Estado	Atributos (propiedades)	color velocidad marcha_actual presion_ruedas	texto autor fecha
Conducta	Métodos (funciones o procedimientos)	cambiar_marcha cambiar_color ver_presión inflar_rueda desinflar_rueda	crear cambiar- texto

Variables/Objetos



En **C**, se definen dos variables X e Y de tipo entero de la forma siguiente:

```
int X,Y;
```

La forma de declarar objetos en **Java** es la misma:

```
Bicicleta bici1,bici2;
```

Clases y Objetos

idea
texto=salir a cenar autor=pepe fecha=hoy
crear cambiar-texto

otra_idea
texto=ir al cine autor=patricia fecha=hoy
crear cambiar-texto

idea y otra_idea son objetos: instancias de la clase Idea

Idea
texto autor fecha
crear cambiar-texto

Clase

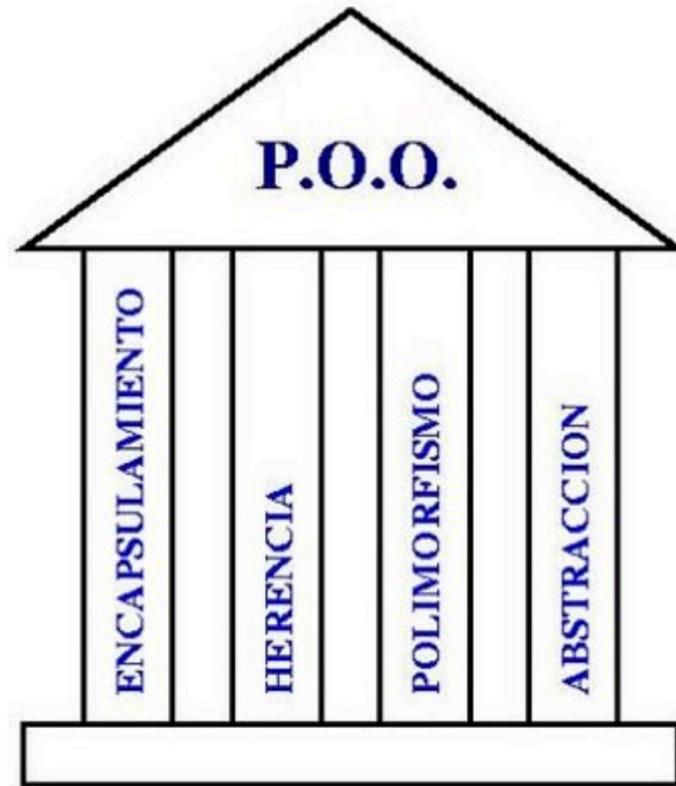


Clase es la descripción de objetos que tienen una estructura y comportamiento comunes mediante atributos y métodos

Características POO

Todo lenguaje POO implementa:

- Encapsulación
- Herencia
- Polimorfismo
- Abstracción



Encapsulación

Los objetos se usan como cajas negras

Así, un objeto encapsula atributos y métodos, que están dentro del objeto y cuyo acceso puede estar permitido o no fuera de la clase.

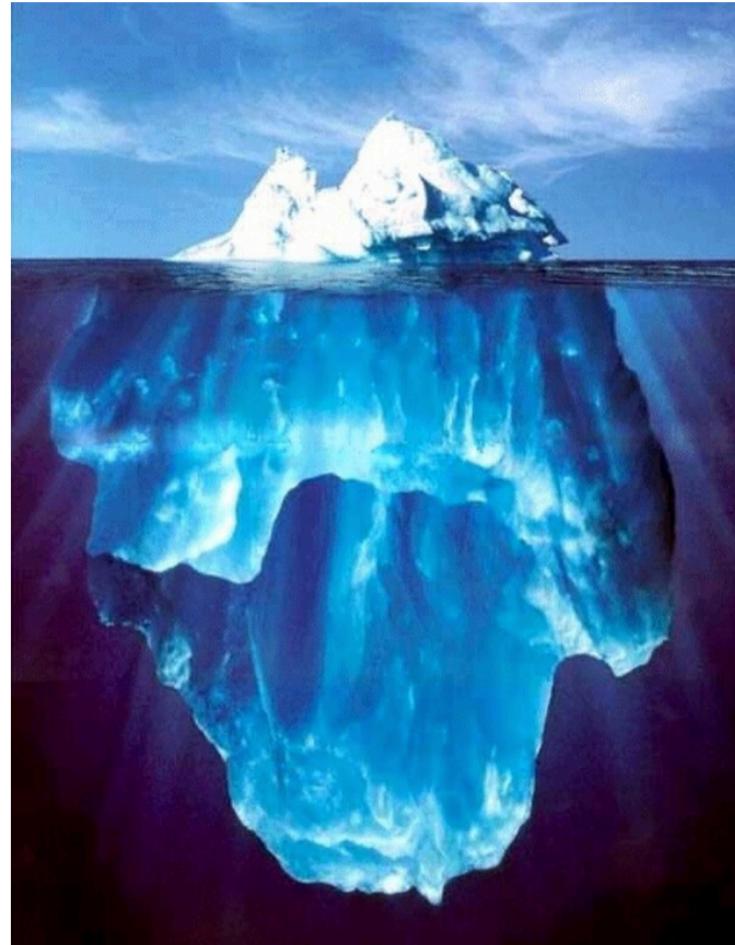
Ejemplo: una clase Coche puede tener un método público acelerar que se puede usar sin saber cómo lo hace

Y métodos y/o atributos privados que implementan el funcionamiento interno de la clase



Encapsulación

- Ocultación de la información



Encapsulación

- Los atributos de una clase normalmente se declaran privados
- Los métodos para acceder y modificar dichos atributos deberán ser públicos
- Se declaran atributos públicos siempre y cuando sean constantes y se utilicen en clases de utilidades

Ejemplo: Math.PI

- Métodos que se utilicen exclusivamente internamente en otros métodos los debemos declarar privados

Herencia

- Una clase puede definirse en relación a otra clase heredando sus atributos y métodos
- Los atributos y métodos de la superclase (clase madre) se incorporan a la subclase (clase hija)
- Los métodos heredados se pueden sobrescribir
- Algunos lenguajes permiten la herencia múltiple (que una clase puede tener varias superclases)
- Ruby y Java solo permiten herencia simple (cada clase solo puede heredar de una superclase)

Polimorfismo

Comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre



Polimorfismo

Polimorfismo en Java → métodos que tengan el mismo nombre:

- **Sobrecarga** (overload)

Con distintos parámetros para una misma clase
Es habitual para constructores

- **Sobrescritura** (overwrite)

Con los mismos parámetros para clases hijas o que implementen un interface

Sobrecarga

```
/* Métodos sobrecargados */  
int calculaSuma(int x, int y, int z){  
    ...  
}  
int calculaSuma(double x, double y, double z){  
    ...  
}  
  
/* Ojo: estos métodos no están sobrecargados */  
int calculaSuma(int x, int y, int z){  
    ...  
}  
double calculaSuma(int x, int y, int z){
```

Sobrecarga

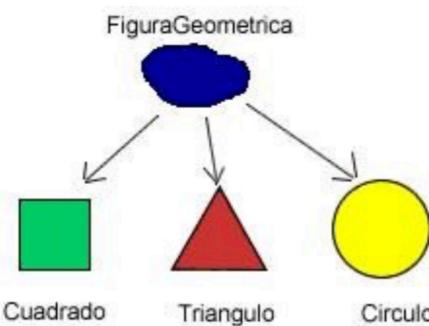
```
class Usuario {  
    String nombre, direccion; int edad;  
  
    /* El constructor de la clase Usuario esta sobrecargado */  
    Usuario( ) {  
        nombre = null;  
        edad = 0;  
        direccion = null;  
    }  
    Usuario(String nombre, int edad, String direccion) {  
        this.nombre = nombre;  
        this.edad = edad;  
        this.direccion = direccion;  
    }  
    Usuario(Usuario usr) {  
        nombre = usr.getNombre();  
        edad = usr.getEdad();  
        direccion = usr.getDireccion();  
    }  
    .....  
}
```

Sobreescritura

```
public abstract class Figura
{
    protected double x;
    protected double y;

    public Figura (double x, double y)
    {
        this.x = x;
        this.y = y;
    }

    public abstract double area ();
}
```



```
public class Cuadrado extends Figura
{
    private double lado;

    public Cuadrado (double x, double y, double lado)
    {
        super(x,y);
        this.lado = lado;
    }

    public double area ()
    {
        return lado*lado;
    }
}
```

Ventajas POO

- Modela de manera sencilla objetos del mundo real → menor coste de diseño
- Facilita la reutilización de código → menor coste implementación
- Facilita la modificación de código → menor coste mantenimiento
- Los lenguajes modernos de programación permiten POO (Java, C++, C#, Objective-c, PHP, Ruby on Rails, Phyton...)

Los conceptos de POO son complejos y están relacionados entre sí.

Se necesita cierto tiempo y experiencia para una comprensión profunda de la POO

