Programación Orientada a Objetos

Unidad 3 - Herencia y Polimorfismo

Unidad 3 - Herencia y Polimorfismo

- Logro: Al finalizar la unidad el alumno construye programas aplicando los principios de herencia y polimorfismo.

Agenda

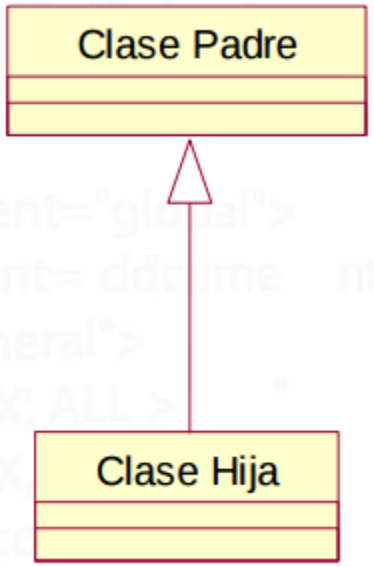
- Herencia
- Polimorfismo

Herencia

- Es un mecanismo que permite extender funcionalidades y características a una entidad existente creando una nueva entidad.
- Es un camino para reutlizar código.
- Podemos agregar métodos y atributos para crear nuevas clases
- La relación es una herencia (generalización / especialización)
- Debe responder a tres condiciones
  - Tener atributos y métodos comunes a las clases involucradas.
  - Debe responder a que la clase hija "es un tipo de" la clase padre.
  - La clase hija debe comportarse como la clase padre.

Herencia en Ruby

En Ruby podemos definir una clase como super clase (padre) de una sub clase (hija). Diremos entonces que la clase hija hereda de la clase padre.



Ejemplo de Herencia

```ruby
class Person

  attr_accessor :name, :age, :sex

  def initialize(name, age, sex)
    @name, @age, @sex = name, age, sex
  end

end

class Teacher < Person

  attr_accessor :hours, :grade

  def initialize(name, age, sex, hours, grade)
    super(name, age, sex)
    @hours, @grade = hours, grade
  end
end

person = Person.new("Carlos", 44, "m")
teacher = Teacher.new("Sonia", 27, "f", 28, "master")
```

Nótese el uso del operador < para indicar la herencia