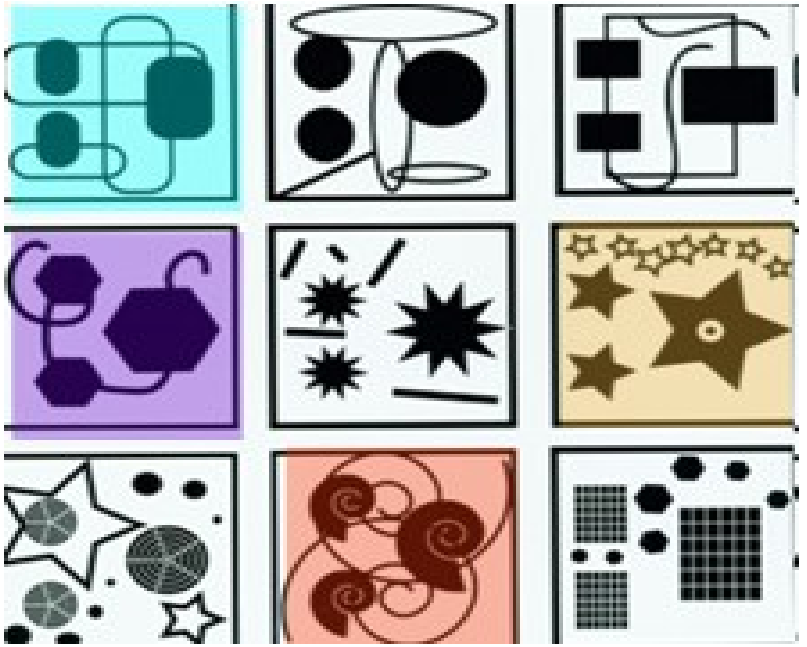
Tema 4



Conceptos complementarios en Orientación a Objetos

Lección 4.3

Comparación de objetos

1. Comparación de objetos: identidad y estado

- A la hora de comparar objetos podemos comparar identidad o estado.
- Dependiendo de cómo se hayan construido los objetos a comparar, así será el resultado de la comparación:

	Comparando identidad **¿objetos idénticos?**	Comparando estado **¿objetos iguales?**
Cuando **a** es idéntico a **b**	true	true
Cuando **a** es copia de **b**	false	true
Cuando **a** no es ni idéntico ni copia de **b**	false	true o false

- Todos los lenguajes de programación proporcionan funcionalidad para comparar identidad e igualdad (estado) de objetos.

2. Comparación de objetos en Java

obj1 **==** *obj2*, *obj1* **!=** *obj2* y *obj1.***equals***(obj2)*

- Comparan identidad por defecto y devuelven `true` o `false`
- Para comparar estado se redefine *equals(obj)*
- *obj1 != obj2* es equivalente a *!(obj1 == obj2)*

Ejemplo

```
class MiClase{
        private String saludo;
}
MiClase mc1 = new MiClase("hola");
MiClase mc2 = new MiClase("hola");
MiClase mc3 = mc1;


mc1 == mc2 // false
mc1 == mc3 // true
mc2 == mc3 // false
mc1.equals(mc2) // false
mc1.equals(mc3) // true
mc2.equals(mc3) // false
```

Entender y manipular el código en ComparacionJava

2. Comparación de objetos en Java

Código recomendado para redefinir equals(obj)

```
@Override
public boolean equals(Object obj) {

        if (obj == null)
                return false;

        if (obj == this)
                return true;

        if (!(obj.getClass().getSimpleName().equals("MiClase")))
                return false;

        MiClase mc = (MiClase) obj;
        if (!saludo.equals(mc.saludo))
                return false;

            // compara según el atributo saludo de obj
        return true;
}
```

Cabecera ya proporcionada y que no podemos cambiar (`@Override`)

Consulta el nombre de la clase a la que pertenece `obj`

Referenciar `obj` por una variable, `mc`, que sea de tipo `MiClase`

3. Comparación de objetos en Ruby

obj1 **==** *obj2*, *obj1* **!=** *obj2*, *obj1.***equal?***(obj2) y obj1.***eql?***(obj2)*

- Comparan identidad por defecto y devuelven true o false
- *obj1 != obj2* equivalente a *!(obj1 == obj2)*
- Recomendaciones para la redefinición de estos métodos:
  - Para comparar estado redefinir *obj1 == obj2*
  - No redefinir *equal?(obj2)* ya que se usa internamente para determinar identidad
  - Mantener el mismo significado de == y eql?(obj2): para Hash compara keys

```
class MiClase
        attr_reader :saludo
end

mc1 = MiClase.new("hola")
mc2 = MiClase.new("hola")
mc3 = mc1


mc1 == mc2 # false
mc1 == mc3 # true
mc2 == mc3 # false

mc1.equal?(mc2) # false
mc1.equal?(mc3) # true
mc2.equal?(mc3) # false
```

Entender y manipular el código en comparacion_ruby

Lectura interesante: comparando objetos en Ruby

3. Comparación de objetos en Ruby

Código recomendado para redefinir ==(obj)

Consulta el nombre de la clase a la que pertenece obj

```
def ==obj

   if (obj == nil)
     return false
   end

   if obj.class.name.split('::').last != 'MiClase'
     return false
   end

   if @saludo != obj.saludo
     return false
   end
         # compara según el atributo saludo de obj
   return true

end
```

Pruebas

Para afianzar y comprender mejor los conceptos aprendidos en esta lección haz pruebas con los siguientes ejemplos en Java y Ruby:

- Ejemplos de comparadores: ComparacionJava y comparacion_ruby