



# UD4

## LENGUAJE DE PROGRAMACIÓN JAVA

### 4.3 Métodos Equals y hashCode

MP\_0485  
Programación

## HashCode y Equals

Cualquier objeto que creemos, independientemente de la clase a la que pertenezca, tendrá un método llamado *hashCode()*, que retorna el código *hash* asociado a dicho objeto. Esta es una característica heredada de la superclase *Object*. El código *hash* es un identificador de 32 bits.

Otro método también heredado de la superclase *Object* es *equals()*, que compara dos objetos y retorna *true* si los objetos son iguales, considerando que son iguales cuando tienen el mismo código *hash*.

Es común que cada clase sobrescriba los métodos *hashCode()* y *equals()* para que obedezcan a los intereses particulares de la lógica de la clase, pero hay que respetar el hecho de que si el método *equals()* dice que dos objetos son iguales, estos deben retornar el mismo código *hash*.

**Dos cadenas de texto con el mismo valor retornarán el mismo código *hash* y la comparación será *true*.**

**Dos objetos que envuelven un valor numérico (*Integer*, *Float*, *Long*, *Double*, etc.) tendrán el mismo código *hash* si tienen el mismo valor.**

**Dos objetos de cualquier clase personalizada tendrán distinto código *hash*, aunque los valores de las propiedades de ambos objetos sean idénticas.**

Podemos ver un ejemplo con la clase *Triangulo*:

```

public class Triangulo {
    private float lado1;
    private float lado2;
    private float lado3;

    public Triangulo(float lado1, float lado2, float lado3) {
        this.lado1 = lado1;
        this.lado2 = lado2;
        this.lado3 = lado3;
    }

    @Override
    public String toString() {
        return "Triangulo [lado1=" + lado1 + ", lado2=" + lado2
            + ", lado3=" + lado3 + "] " + this.tipo();
    }

    public String tipo() {
        if (this.lado1 == this.lado2 && this.lado2 == this.lado3)
            return "EQUILATERO";
        else if (this.lado1 == this.lado2 || this.lado2 == this.lado3
            || this.lado1 == this.lado3)
            return "ISOSCELES";
        else
            return "ESCALENO";
    }
}

```

Ahora en la clase *Principal* probamos a comparar dos objetos *Triangulo* con los mismos valores:

```

Triangulo t1 = new Triangulo(3,3,3);
Triangulo t2 = new Triangulo(3,3,3);
System.out.println(t1.toString());
System.out.println(t2.toString());
System.out.println(t1.hashCode());

System.out.println(t2.hashCode());
System.out.println(t1.equals(t2));

```

Podríamos cambiar este comportamiento **sobrescribiendo** los métodos *hashCode()* y *equals()* en la clase *Triangulo* de esta forma:

```
@Override
public int hashCode() {
    return (int) Math.round(this.lado1+this.lado2+this.lado3);
}
@Override
public boolean equals(Object obj) {
    if (this.hashCode() == obj.hashCode())
        return true;
    else
        return false;
}
```

Ahora considerará que dos triángulos son iguales cuando las sumas de sus lados sean iguales.

**En conclusión:** cuando implementas una clase puedes sobrescribir los métodos *equals()* y *hashCode()* considerando según tu criterio o necesidad del proyecto cuándo dos objetos deben considerarse iguales.

## Referencias

Apuntes elaborados a partir de la siguiente documentación:

[1] Apuntes de Programación Edix Digital Workers.