{dev/talles}

Object & Generics en Java

Por: Gabriel Chaldú

¿Qué son los Object?

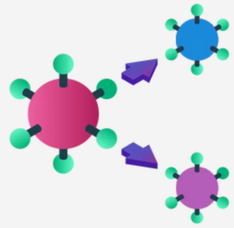
Es la **superclase** de todas las clases en Java.

Caraterísticas

Herencia universal

Todas las clases en Java heredan de Object.

Polimorfismo



Permite almacenar cualquier tipo de objeto.

Caraterísticas

Métodos básicos

toString(), equals(), hashCode(), getClass(), entre otros.

Object

```java
public class ObjectExample {
    public static void main(String[] args) {
        Object obj1 = "Hola, soy un String";
        Object obj2 = 42;
        Object obj3 = new Persona("Juan", 30);

        System.out.println(obj1.toString()); // Hola, soy un String
        System.out.println(obj2.toString()); // 42
        System.out.println(obj3.toString()); // Persona{nombre='Juan', edad=30}
    }
}
```

Object vs Objects

La clase ***Objects*** (java.util.Objects) *no es una superclase*, es una clase utilitaria que contiene ***métodos estáticos*** para realizar operaciones seguras con objetos.

{dev/talles}

Generics en Java

Por: Gabriel Chaldú

¿Qué son los generics?

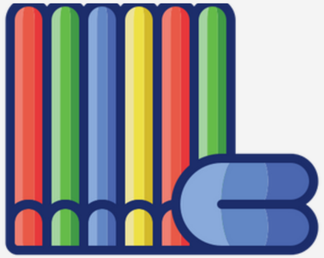
Los Generics en Java son como **plastilina** inteligente.

Ventajas

Reutilización

Única implementación de código

Flexibilidad



Trabajar con distintos tipos de datos

Ventajas

Seguridad

Verificar los tipos en tiempo de compilación

Casting

Trabajar con distintos tipos de datos

Generics Inseguros

```java
public class SinGenerics {
    public static void main(String[] args) {
        ArrayList lista = new ArrayList(); // Sin Generics
        lista.add("Hola");
        lista.add(10); // Se puede agregar cualquier tipo

        for (Object obj : lista) {
            // Se espera un String, pero hay un Integer en la lista
            String texto = (String) obj; // ERROR en tiempo de ejecución
            System.out.println(texto);
        }
    }
}
```

Generics Seguros

```java
public class ConGenerics {
    public static void main(String[] args) {
        ArrayList<String> lista = new ArrayList<>(); // Lista de Strings
        lista.add("Hola");
        // lista.add(10); // ❌ ERROR en compilación

        for (String texto : lista) {
            System.out.println(texto.toUpperCase()); // Seguro
        }
    }
}
```

Generics con clases

```
class Box<T> {
    private T value;

    public void setValue(T value) {
        this.value = value;
    }

    public T getValue() {
        return value;
    }
}
```

Generics interface

```java
// Definimos una interfaz genérica
interface Repository<T> {
    void save(T entity);
    T findById(int id);
}

// Implementación de la interfaz para un tipo específico
class UserRepository implements Repository<String> {
    @Override
    public void save(String entity) {
        System.out.println("Guardando usuario: " + entity);
    }

    @Override
    public String findById(int id) {
        return "Usuario " + id;
    }
}
```

{dev/talles}

¡A seguir mejorando!