Optionals

* isPresent() → Devuelve si el optional tiene valor
* ifPresent() →
* get() → Devuelve el valor que se encuentra en Optional
* orElse() → Retorna un valor por defecto si el valor no se encuentra presente

|  |
| --- |
| Optional nonEmptyOptional = Optional.ofNullable(“original”);  String value1 = nonEmptyOptional.orElse(“default Value”); |

* Optional.of() → No puede ser nulo
* Optional.ofNullable() → Puede ser nulo
* Optional.empty() → Devolverá true si el valor devuelto es nulo

Streams

forEach:

Persona p1 = new Persona("juan", "sanchez", 20)

Persona p2 = new Persona("ana", "gomez", 12);

Persona p3 = new Persona("pedro", "gutierrez", 40);

List<Persona> lista=Arrays.asList(p1,p2,p3);

// Tradicional

for (Persona p:lista) {

System.out.println(p.getNombre());

System.out.println(p.getApellidos());

System.out.println(p.getEdad());

}

//Stream

lista.stream().forEach((p)-> {

System.out.println(p.getNombre());

System.out.println(p.getApellidos());

System.out.println(p.getEdad());

});

reduce():

public static void main(String[] args) {

List<Integer> gastos= new ArrayList<Integer>();

gastos.add(100);

gastos.add(200);

gastos.add(300);

//reduce() normal

gastos.stream().reduce((acumulador,numero)-> {

return acumulador+numero;}).ifPresent(System.out::println);

//simplificado

gastos.stream().reduce(Integer::sum).ifPresent(System.out::println);

Operador Elvis:

Retorna el resultado si el primer operando existe y es diferente de null.

if (x>y)

mayor = x;

else

mayor = y;

resultado = (condicion)?valor1:valor2;

mayor=(x>y)?x:y;

Método toString():

Siempre que la referencia que se utilice no apunte a null.

No acepta ningún parámetro.