El uso de Optional Map a veces le resulta extraño a los desarrolladores . Recordemos que los tipos Optional están diseñados para la gestión de valores que pueden ser o no ser nulos. Al apoyarnos en un Optional nos aseguramos que siempre se comprueba su valor antes de usarlo . Vamos a ver un ejemplo sencillo.

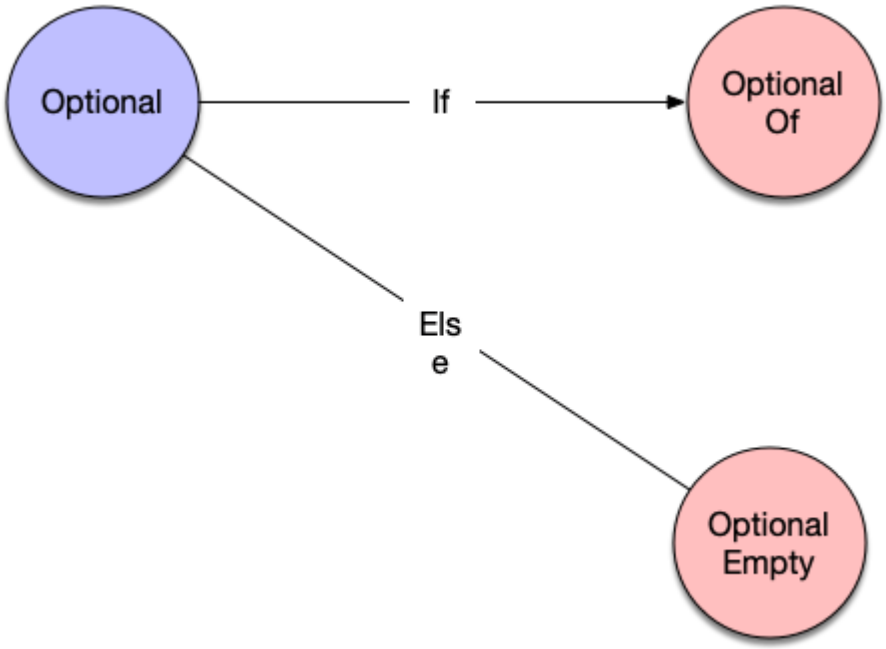
```
        Optional<String> texto = Optional.ofNullable("hola");
                Optional<String> mayusculas;

                if (texto.isPresent()) {
                        mayusculas =
Optional.of(texto.get().toUpperCase());
                } else {
                        mayusculas = Optional.empty();
                }

                if (mayusculas.isPresent()) {

                        System.out.println(mayusculas.get());
                }
        }
```

En este caso disponemos de una variable de texto que en algún momento puede ser nula y en otros momentos no. Hasta aquí todo es muy muy sencillo. El problema le tenemos que queremos convertir esa variable de texto que es un Optional en un optional que almacene un valor en mayusculas. La situación parece directa. Pero como podemos ver el código que hemos construido es laborioso ya que tenemos que preguntar si el Optional contiene un valor o no lo contiene y con ese resultado generar un nuevo optional con el valor en mayúscula. Como nos podemos dar cuenta esto es un poco engorroso.



Optional Map

Podemos realizar la mismo operación con Optional.map de una forma mucho más directa ya que nos permite convertir un optional en otro y realizar una transformación sin tener que volver a generar nuevos optionals.

```
Optional<String> texto = Optional.ofNullable("hola");
Optional<String> mayusculas = texto.map(String::toUpperCase);
        if (mayusculas.isPresent()) {
                System.out.println(mayusculas.get());
        }
```

El resultado será idéntico pero más compacto y sencillo de entender:



Aprendamos a usar mejor Java 8 y Optional Map para simplificar nuestro código y reducir la

complejidad de este.

Otros artículos relacionados

- Java Optional ifPresent y como utilizarlo
- Java Stream for loop y programación funcional
- Java Stream Sum y Business Objects
- Java Optional Stream y reference methods