Tabla de Contenidos

Java Stream if else . ¿Como podemos manejar una estructura if/else dentro de un Java Stream? . Esta es una pregunta bastante habitual cuando uno comienza a trabajar con Streams en el día a día. Vamos a ver esta situación a detalle.

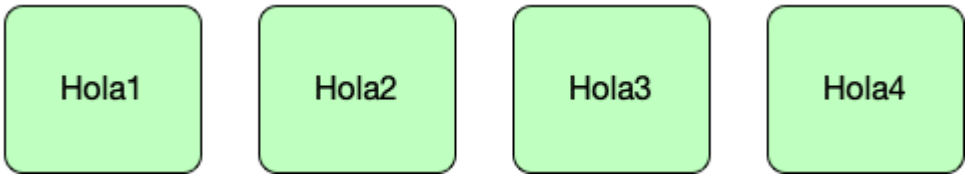
Java Optionals y Streams

Para ello vamos a partir de una lista de cadenas . Esta lista contendrá varios textos y nosotros únicamente deseamos encontrar la primera coincidencia del texto en la cadena. La forma más sencilla de hacerlo es apoyarnos en los métodos filter y findFirst del API de Java Streams.

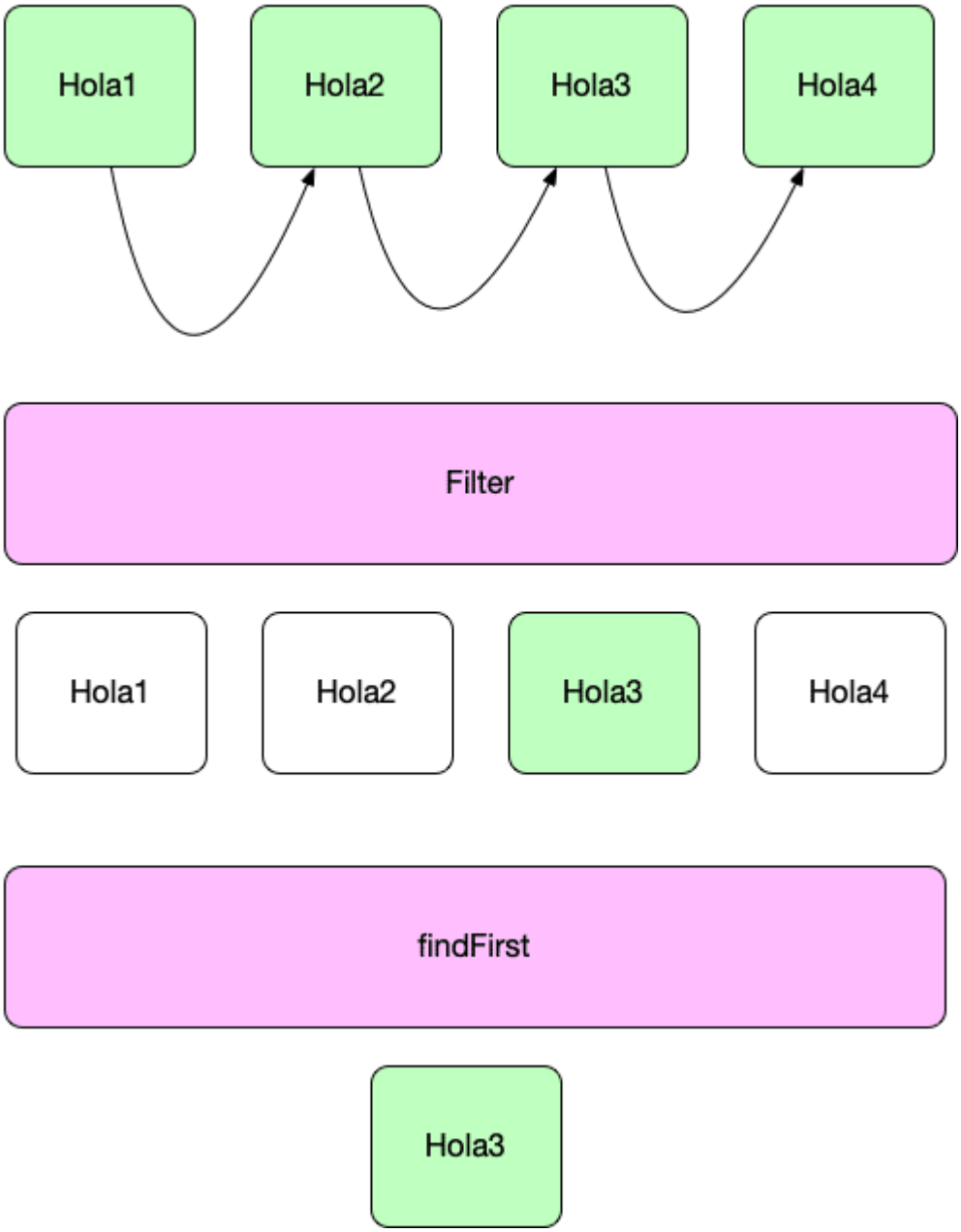
```
package com.arquitecturajava;

import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

public class Principal2 {
        public static void main(String[] args) {

                List<String> listaCadenas= new ArrayList<String>();
                listaCadenas.add("hola");
                listaCadenas.add("hola2");
                listaCadenas.add("hola2");
```

```
                listaCadenas.add("hola3");
                listaCadenas.add("hola4");
                Optional<String> textoEncontrado=listaCadenas
                .stream()
                .filter((cadena)->cadena.equals("hola3"))
                .findFirst();
                if (textoEncontrado.isPresent()) {
                        System.out.println(textoEncontrado.get());
                }

        }
}
```

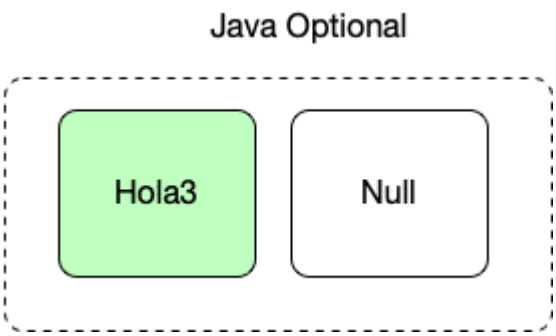
En este caso podemos ver como construimos una lista de textos y la convertimos en un Stream.



Este Stream es recorrido y filtrado quedándonos con aquellos valores que coincidan con “hola3” . Estos valores pueden ser 0 ,1 o más . Al invocar luego al método findFirst estamos limitando la lista a un único registro .



Pero claro este registro no siempre contendrá un valor ya que puede que ningún item de la lista cumpla las condiciones . Por lo tanto nos devuelve un Java Optional que es un objeto que puede contener un valor o no.



Manejando Optionals

Una vez que devuelve un objeto de tipo Java Optional no nos queda más que comprobar con el método isPresent() si el optional almacena un valor. En el caso de que así sea pues decidimos imprimirlo por la consola

```
if (textoEncontrado.isPresent()) {
                        System.out.println(textoEncontrado.get());
                }
```

Podemos complementar el código con una sentencia else en la cual simplemente imprimimos un mensaje de que no hay coincidencias

```
if (textoEncontrado.isPresent()) {
                        System.out.println(textoEncontrado.get());
                }else System.out.println("no hay coincidencias")
```

Java Stream if else con Optionals

El código funciona pero lamentablemente no es tan compacto como nos gustaría ya que debemos extraer de un Stream un Java Optional y luego procesarlo . ¿Se puede hacer de una forma más compacta?. La realidad es que sí podemos transformar el código anterior en el siguiente usando el método ifPresentOrElse que recibe dos expresiones lambda.

```
package com.arquitecturajava;
```

```
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

public class Principal2 {
        public static void main(String[] args) {

                List<String> listaCadenas= new ArrayList<String>();
                listaCadenas.add("hola");
                listaCadenas.add("hola2");
                listaCadenas.add("hola2");
                listaCadenas.add("hola3");
                listaCadenas.add("hola4");
                listaCadenas
                .stream()
                .filter((cadena)->cadena.equals("hola3"))
                .findFirst()
                .ifPresentOrElse((texto)->System.out.println(texto),
()->System.out.println("no hay datos"));

        }
}
```

Java Stream if else y Conclusiones

De esta forma habremos integrado dentro de un Stream una estructura if else usando Java Optionals. Este método esta soportado desde Java 9

Otros artículos relacionados

- ¿Que es un Java Optional?
- ¿Que es un Java Stream?

- Java Stream for loop y programación funcional