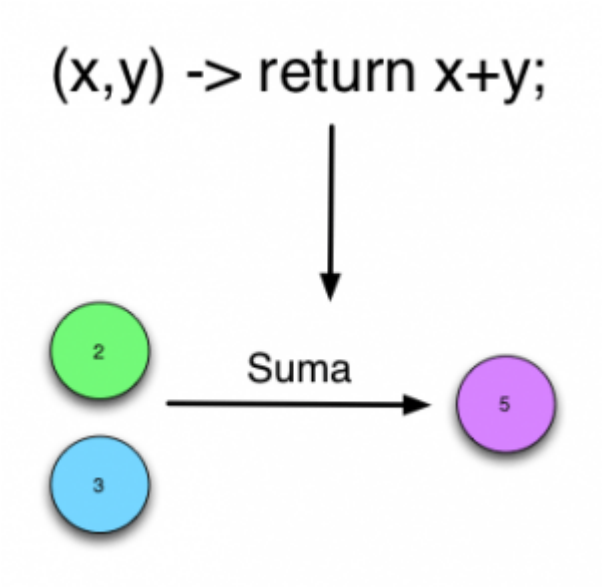
¿Qué es un Java Functional Interface? . Java8 incluye muchas novedades y entra ellas destacan las expresiones lambda. Recordemos que una expresión lambda define el comportamiento de una función.



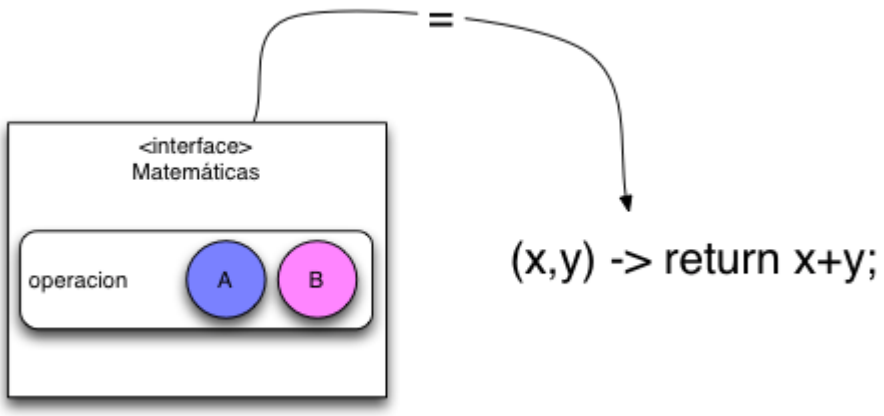
Podemos usar la expresión lambda en nuestro código Java :

```
package com.arquitecturajava.functional;

public class Principal {

public static void main(String[] args) {

(x, y) -> x + y;

}

}
```

Sin embargo ella sola no compila ya que necesitamos igualarla a una variable lo que no tenemos claro es a qué exactamente.

El concepto de Java Functional Interface

Es aquí donde aparece el concepto de Java Functional Interface. Un interface funcional es aquel interface que solo dispone de un método abstracto. Recordemos que a día de hoy los interfaces han evolucionado mucho y pueden disponer de métodos por defecto. Así pues vamos a igualar nuestra expresión lambda a un interface funcional

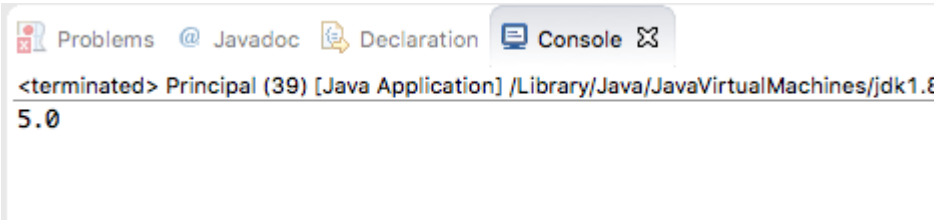


Vamos a verlo en código:

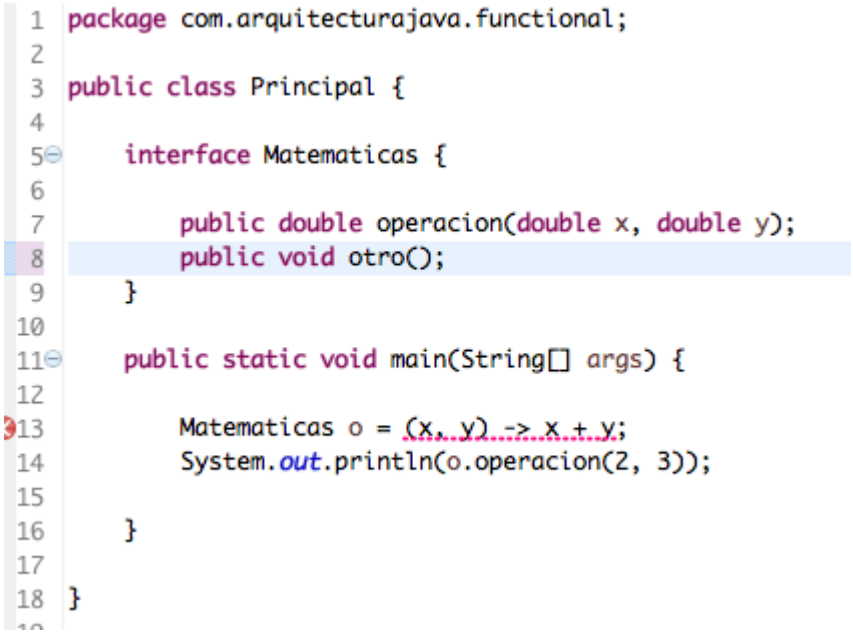
```
package com.arquitecturajava.functional;

public class Principal {

interface Matematicas {
```

```
public double operacion(double x, double y);

}

public static void main(String[] args) {

Matematicas o = (x, y) -> x + y;
System.out.println(o.operacion(2, 3));

}

}
```

Acabamos de declarar el interface Matemáticas y hemos añadido un método para la operación , si ejecutamos el código nos sumara los dos valores por la consola:



Modificamos el interface y añadimos un nuevo método el programa no compilará ya que nos avisará de que no estamos apuntando a un Java Functional Interface , recordemos que estos únicamente disponen de un método abstracto.



```
package com.arquitecturajava.functional;

public class Principal {

    interface Matematicas {

        public double operacion(double x, double y);
        public void otro();
    }

    public static void main(String[] args) {

        Matematicas o = (x, y) -> x + y;
        System.out.println(o.operacion(2, 3));

    }

}
```

Para evitar este tipo de problemas se usa la anotación @FunctionalInterface que obliga al desarrollador a solo incluir un método abstracto.

```
package com.arquitecturajava.functional;

public class Principal {
@FunctionalInterface
interface Matematicas {

public double operacion(double x, double y);

}

public static void main(String[] args) {

Matematicas o = (x, y) -> x + y;
```

```
System.out.println(o.operacion(2, 3));

}

}
```

De esta forma cometeremos menos errores. Este es en resumen el concepto de Java Functional Interface.

Otros artículos relacionados: Java Lambda , Java Lambda Reduce ,Java Lambda WorkFlows