INSTITUTO TECNOLÓGICO SUPERIOR DE JEREZ

Jerez de García Salinas

20/03/2020

Ivan Gamboa Ultreras

16070125

gamboita9@gmail.com

Ingeniería en Sistemas Computacionales

Programación Lógica y Funcional

8vo Semestre

Actividad:

Mapa Conceptual: Interfaces Funcionales

ISC Salvador Acevedo Sandoval

1.- Matemáticamente, ¿qué es el cálculo lambda?

Es un sistema formal diseñado para investigar la definición de función, la noción de aplicación de funciones y la recursión. (wikipedia, 2020)

Tiene por objeto explicitar el concepto que representa el empleo de funciones como medio de transformación de argumentos en resultados. (rinconmatematico, 2006)

2.- ¿Que son las 'functional interfaces' en Java?

Una interfaz funcional es una interfaz que contiene solo un método abstracto. Solo pueden tener una funcionalidad para exhibir. Desde Java 8 en adelante, las expresiones lambda se pueden usar para representar la instancia de una interfaz funcional. Una interfaz funcional puede tener cualquier cantidad de métodos predeterminados. (geeksforgeeks, s.f.)

3.- ¿Cuáles son las 6 interfaces funcionales del paquete java.util.function?

Existe una gran variedad de interfaces funcionales en Java siendo unas muy parecidas, pero nunca iguales, de las mas utilizadas son las siguientes 6:

Function:

La “Function Interface” de Java (java.util.function.Function) es una de las interfaces funcionales centrales de Java. La “Function Interface” representa una función (método) que toma un parámetro y devuelve solo un valor.

Predicate:

La “Predicate Interface” (java.util.function.Predicate), representa una función simple que toma un parámetro y devuelve un valor verdadero o falso.

UnaryOperator:

La “UnaryOperator Interface” es una interfaz funcional que representa una operación la cual toma un parámetro y devuelve un valor. Ambos del mismo tipo.

BinaryOperator:

La “BinaryOperator Interface” es una interfaz funcional que representa una operación la cual toma dos parámetros y devuelve un valor. Ambos parámetros y el valor de retorno deben ser del mismo tipo.

Supplier:

La “Supplier Interface“ es una interfaz funcional que representa una función que proporciona un valor de algunos tipos. La “Supplier Interface” también puede considerarse como una “Factory Interface”.

Consumer:

La “Consumer Interface” es una interfaz funcional que representa una función que consume un valor sin devolver ningún valor.

(Herrera, 2017) (Jenkov, 2020)

4.- ¿Qué son las expresiones lambda?

Las expresiones lambda básicamente expresan instancias de interfaces funcionales. Estas expresiones implementan solo una función abstracta y, por lo tanto, implementan interfaces funcionales. (geeksforgeeks, s.f.)

5.- Sintaxis de las expresiones lambda

La sintaxis para las expresiones lambda consiste en 3 partes:

- Una lista separada por comas de parámetros formales entre paréntesis.
- El símbolo de flecha compuesto por un guion y el símbolo mayor que, `->`
- Un cuerpo, que consiste en una expresión o un bloque de instrucciones

(oracle, s.f.)

6.- ¿Qué son los “streams” y para qué sirve?

Se utiliza para procesar colecciones de objetos. Un “Stream” es una secuencia de objetos que admite varios métodos los cuales pueden canalizarse para producir el resultado deseado.

Las características del “Stream” son:

- Un “Stream” no es una estructura de datos, sino que toma datos de las colecciones, matrices o canales de E / S.
- El “Stream” no cambia la estructura de datos original, solo proporcionan el resultado según los métodos canalizados.
- Cada operación intermedia se ejecuta perezosamente y devuelve un “Stream” como resultado, por lo tanto, se pueden canalizar varias operaciones intermedias. Las operaciones de terminal marcan el final del “Stream” y devuelven el resultado.

(geeksforgeeks, s.f.)

7.- Funciones más relevantes de la clase “Stream”

En un “Stream” hay dos tipos de operaciones las intermedias y terminales:

Operaciones intermedias:

- map: El método de “map” se usa para devolver un “Stream” que consiste en los resultados de aplicar la función dada a los elementos de este “Stream”.
- filter: el método de “filter” se utiliza para seleccionar elementos según el predicado pasado como argumento.
- sorted: el método “sorted” se utiliza para ordenar el “Stream”.

Operaciones Terminales:

- collect: el método “collect” se utiliza para devolver el resultado de las operaciones intermedias realizadas en el “Stream”.
- forEach: el método “forEach” se usa para recorrer cada elemento del “Stream”.
- Reduce: El método “reduce” se usa para reducir los elementos de un “Stream” a un solo valor. Este método toma un “BinaryOperator” como parámetro.

(geeksforgeeks, s.f.)

Referencias

geeksforgeeks. (s.f.). Recuperado el 18 de Marzo de 2020, de https://www.geeksforgeeks.org/functional-interfaces-java/

geeksforgeeks. (s.f.). Recuperado el 18 de Marzo de 2020, de https://www.geeksforgeeks.org/lambda-expressions-java-8/

geeksforgeeks. (s.f.). Recuperado el 18 de Marzo de 2020, de https://www.geeksforgeeks.org/stream-in-java/

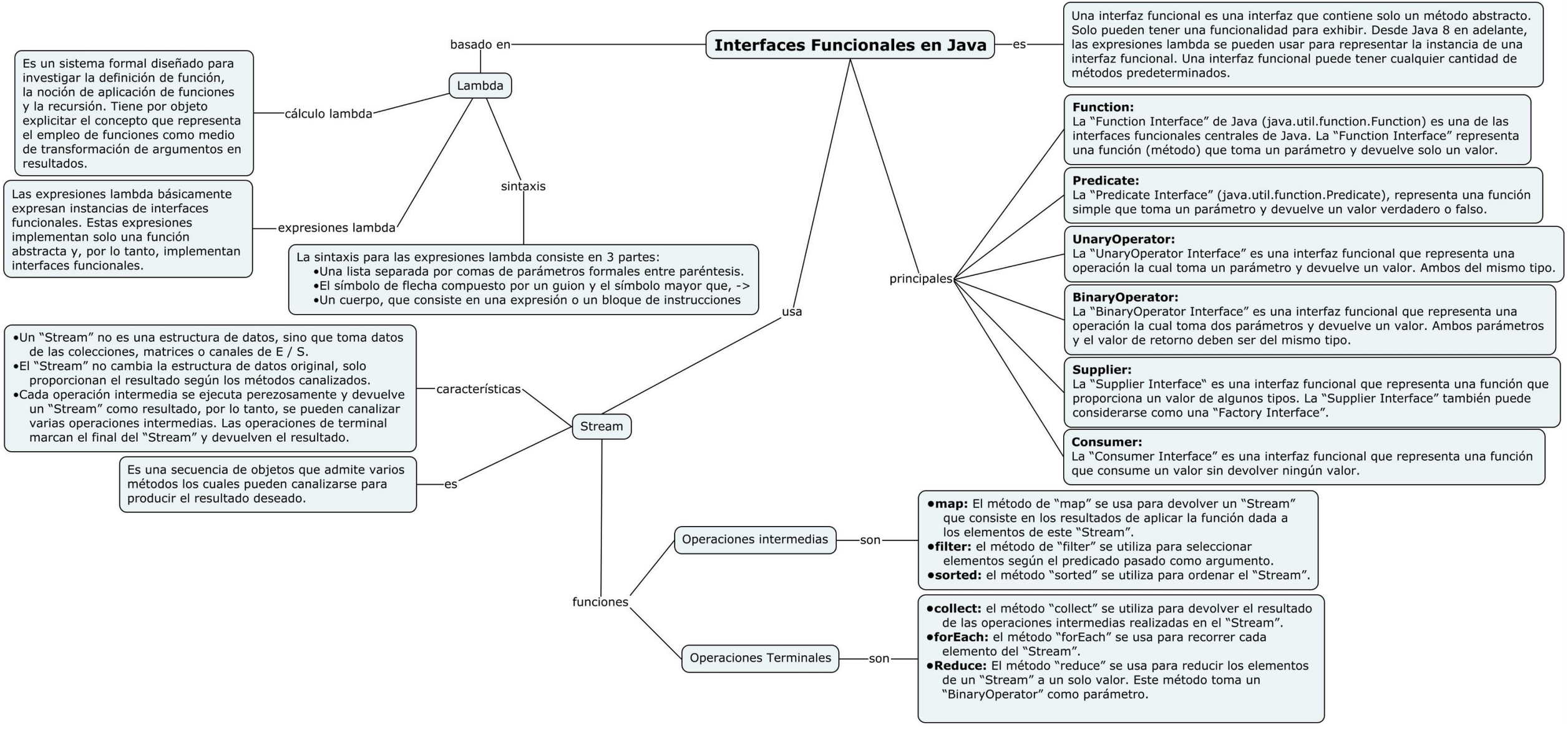
Herrera, E. (2017). *eherrera*. Recuperado el 18 de Marzo de 2020, de http://eherrera.net/ocpj8-notes/04-lambda-built-in-functional-interfaces

Jenkov, J. (18 de Marzo de 2020). *jenkov*. Obtenido de tutorials.jenkov.com/java-functional-programming/functional-interfaces.html

oracle. (s.f.). Recuperado el 18 de Marzo de 2020, de https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html#syntax

rinconmatematico. (2006). Recuperado el 18 de Marzo de 2020, de http://www.rinconmatematico.com/foros/index.php?action=dlattach;topic=19069.0;attach=4367

wikipedia. (19 de Marzo de 2020). Recuperado el 20 de marzo de 2020, de https://es.wikipedia.org/wiki/C%C3%A1lculo_lambda


Interfaces Funcionales en Java
es
Una interfaz funcional es una interfaz que contiene solo un método abstracto. Solo pueden tener una funcionalidad para exhibir. Desde Java 8 en adelante, las expresiones lambda se pueden usar para representar la instancia de una interfaz funcional. Una interfaz funcional puede tener cualquier cantidad de métodos predeterminados.
basado en
Lambda
cálculo lambda
Es un sistema formal diseñado para investigar la definición de función, la noción de aplicación de funciones y la recursión. Tiene por objeto explicitar el concepto que representa el empleo de funciones como medio de transformación de argumentos en resultados.
expresiones lambda
Las expresiones lambda básicamente expresan instancias de interfaces funcionales. Estas expresiones implementan solo una función abstracta y, por lo tanto, implementan interfaces funcionales.
sintaxis
La sintaxis para las expresiones lambda consiste en 3 partes:
•Una lista separada por comas de parámetros formales entre paréntesis.
•El símbolo de flecha compuesto por un guion y el símbolo mayor que, ->
•Un cuerpo, que consiste en una expresión o un bloque de instrucciones
usa
Stream
características
•Un "Stream" no es una estructura de datos, sino que toma datos de las colecciones, matrices o canales de E / S.
•El "Stream" no cambia la estructura de datos original, solo proporcionan el resultado según los métodos canalizados.
•Cada operación intermedia se ejecuta perezosamente y devuelve un "Stream" como resultado, por lo tanto, se pueden canalizar varias operaciones intermedias. Las operaciones de terminal marcan el final del "Stream" y devuelven el resultado.
es
Es una secuencia de objetos que admite varios métodos los cuales pueden canalizarse para producir el resultado deseado.
funciones
Operaciones intermedias
son
•map: El método de "map" se usa para devolver un "Stream" que consiste en los resultados de aplicar la función dada a los elementos de este "Stream".
•filter: el método de "filter" se utiliza para seleccionar elementos según el predicado pasado como argumento.
•sorted: el método "sorted" se utiliza para ordenar el "Stream".
Operaciones Terminales
son
•collect: el método "collect" se utiliza para devolver el resultado de las operaciones intermedias realizadas en el "Stream".
•forEach: el método "forEach" se usa para recorrer cada elemento del "Stream".
•Reduce: El método "reduce" se usa para reducir los elementos de un "Stream" a un solo valor. Este método toma un "BinaryOperator" como parámetro.
principales
Function:
La "Function Interface" de Java (java.util.function.Function) es una de las interfaces funcionales centrales de Java. La "Function Interface" representa una función (método) que toma un parámetro y devuelve solo un valor.
Predicate:
La "Predicate Interface" (java.util.function.Predicate), representa una función simple que toma un parámetro y devuelve un valor verdadero o falso.
UnaryOperator:
La "UnaryOperator Interface" es una interfaz funcional que representa una operación la cual toma un parámetro y devuelve un valor. Ambos del mismo tipo.
BinaryOperator:
La "BinaryOperator Interface" es una interfaz funcional que representa una operación la cual toma dos parámetros y devuelve un valor. Ambos parámetros y el valor de retorno deben ser del mismo tipo.
Supplier:
La "Supplier Interface" es una interfaz funcional que representa una función que proporciona un valor de algunos tipos. La "Supplier Interface" también puede considerarse como una "Factory Interface".
Consumer:
La "Consumer Interface" es una interfaz funcional que representa una función que consume un valor sin devolver ningún valor.