LongConsumer Functional Interface

LongConsumer Interface represents an operation that accepts long valued argument and returns no result. Previously we had discussed [Consumer](http://data-structure-learning.blogspot.com/2015/07/java-lambda-consumer-functional.html) interface. I would recommend you to read that as LongConsumer is primitive long data type specialization of [Consumer](http://data-structure-learning.blogspot.com/2015/07/java-lambda-consumer-functional.html) interface.

There are two different methods in LongConsumer interface. Let us understand both of them one by one.

**accept() method**

**void** accept(**long** value);

accept() method performs this operation on give argument. This method does not return any value. It accepts a parameter of type long.

Let us take simple example to understand this method.

LongConsumer longConsumer = (a) -> System.***out***.println(a);

longConsumer.accept(10); //Prints 10

**andThen()** method

**default** LongConsumer andThen(LongConsumer after) {

Objects.*requireNonNull*(after);

**return** (**long** t) -> { accept(t); after.accept(t); };

}

andThen() method returns a composed version of this and after LongConsumer that will perform respective operation in sequence. This method will throw NullPointerException if after is null.

Let us write two different LongConsumer. One for adding and another for multiplication.

LongConsumer adder = (a) -> System.***out***.println(a + a);

LongConsumer multiplier = (a) -> System.***out***.println(a \* a);

adder.andThen(multiplier).accept(50);

Output

100

2500

Read about important java.util.function package’s interface [here](http://data-structure-learning.blogspot.com/p/functional-programming-in-java.html). [Consumer](http://data-structure-learning.blogspot.com/2015/07/java-lambda-consumer-functional.html), [Function](http://data-structure-learning.blogspot.com/2015/07/java-lambda-function-functional.html), [Supplier](http://data-structure-learning.blogspot.com/2015/07/java-lambda-supplier-functional.html), [BinaryOperator](http://data-structure-learning.blogspot.com/2015/07/java-lambda-binaryoperator-functional.html) & [Predicate](http://data-structure-learning.blogspot.com/2015/07/java-lambda-predicate-functional.html) Functional Interfaces. I have also written on [High Order functions](http://data-structure-learning.blogspot.com/2015/07/higher-order-functions-using-function.html) using Function functional interface.