ToIntBiFunction Functional Interface

Before we begin to discuss about ToIntBiFunction interface we need to know about [BiFunction](http://data-structure-learning.blogspot.com/2015/07/java-lambda-bifunction-functional.html) interface. BiFunction interface represents a function that accepts two arguments *T* and *U* and returns result of type *R*. Input and result are of type Object. Previously we have discussed about BiFunction interface and [Function](http://data-structure-learning.blogspot.com/2015/07/java-lambda-function-functional.html) interface. I would highly recommend you to read it. I also have written on [higher order functions](http://data-structure-learning.blogspot.com/2015/07/higher-order-functions-using-function.html).

ToIntBiFunction interface represents a function that accepts two argument of type T and U and returns int-valued result.

|  |
| --- |
| **ToIntBiFunction Interface Declaration**  **public** **interface** ToIntBiFunction<T, U>  T and U are the input types. |
| **applyAsInt() method**  **int** applyAsInt(T t, U u);  T and U are input for the method applyAsInt(). This function applies the given arguments and returns int result. Let us take simple example of accepting two strings and returning their length.  ToIntBiFunction<String, String> intBiFunc = (str1, str2)->str1.length()+str2.length();  **int** result=intBiFunc.applyAsInt("Dell", "Microsoft");  System.***out***.println(result); //Outputs 13 |

That’s all on ToIntBiFunction interface.

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.