**Java 8 features**

1. [Java 8 – Lambda Expression](https://beginnersbook.com/2017/10/java-lambda-expressions-tutorial-with-examples/)  
2. [Java 8 – Method references](https://beginnersbook.com/2017/10/method-references-in-java-8/)  
3. [Java 8 – Functional interfaces](https://beginnersbook.com/2017/10/java-functional-interfaces/)  
4. [Java 8 – Interface changes: Default and static methods](https://beginnersbook.com/2017/10/java-8-interface-changes-default-method-and-static-method/)  
5. [Java 8 – Streams](https://beginnersbook.com/2017/10/java-8-stream-tutorial/)  
6. [Java 8 – Stream filter](https://beginnersbook.com/2017/10/java-8-stream-filter/)  
7. [Java 8 – forEach()](https://beginnersbook.com/2017/10/java-8-foreach/)  
8. [Java 8 – Collectors class with example](https://beginnersbook.com/2017/10/java-8-stream-collectors-class-with-examples/)  
9. [Java 8 – StringJoiner class with example](https://beginnersbook.com/2017/10/java-8-stringjoiner/)  
10. [Java 8 – Optional class with example](https://beginnersbook.com/2017/10/java-8-optional-class/)  
11. [Java 8 – Arrays Parallel Sort](https://beginnersbook.com/2017/10/java-8-arrays-parallel-sort-with-example/)

1. [Java 8 – Lambda Expression](https://beginnersbook.com/2017/10/java-lambda-expressions-tutorial-with-examples/)

Lambda expression is a new feature which is introduced in Java 8. A lambda expression is an anonymous function. A function that doesn’t have a name and doesn’t belong to any class.

## Java Lambda Expression Syntax

To create a lambda expression, we specify input parameters (if there are any) on the left side of the lambda operator ->, and place the expression or block of statements on the right side of lambda operator. For example, the lambda expression (x, y) -> x + y specifies that lambda expression takes two arguments x and y and returns the sum of these.

//Syntax of lambda expression

(parameter\_list) -> {function\_body}

-> This is arrow operator

## Lambda expression vs method in Java

A method (or function) in Java has these main parts:  
1. Name  
2. Parameter list  
3. Body  
4. return type.

A lambda expression in Java has these main parts:  
Lambda expression **only has body and parameter list**.  
1. **No** name – function is anonymous so we don’t care about the name  
2. Parameter list  
3. Body – This is the main part of the function.  
4. **No** return type – The java 8 compiler is able to infer the return type by checking the code. you need not to mention it explicitly.

## Where to use the Lambdas in Java

To use lambda expression, you need to either create your own functional interface or use the pre defined functional interface provided by Java. An interface with **only single abstract method** is called functional interface(or Single Abstract method interface), for example: Runnable, callable, ActionListener etc.

**To use function interface:**  
Pre Java 8: We create anonymous inner classes.  
Post Java 8: You can use lambda expression instead of anonymous inner classes.

<https://beginnersbook.com/2017/10/java-8-features-with-examples/>

<https://howtodoinjava.com/java-8-tutorial/>