Java 8 Features

# Hi there, this document belongs to Sania, you are most welcome to go through it.

# It was a revolutionary release of the Java for software development platform. It includes various upgrades to the Java programming, JVM, Tools and libraries.

Java 8 features are as below:-

* Lambda expressions,
* Method references,
* Functional interfaces,
* Stream API,
* Default methods,
* Static methods in interface,
* Optional class,
* ForEach() method and many more…

We will go through each one of the mentioned above

**Lambda Expressions**: Lambda expression helps us to write our code in functional style. It provides a concise way to implement functional interface(Single Abstract Method) by using an expression. It is very useful in collection library in which it helps to iterate, filter and extract data.

Syntax:

(argument-list) -> {body}

1) Argument-list: It can be empty or non-empty as well.

2) Arrow-token: It is used to link arguments-list and body of expression.

3) Body: It contains expressions and statements for lambda expression.

**Method references:** It is used to refer method of functional interface . It is compact and easy form of lambda expression. Each time when you are using lambda expression to just referring a method, you can replace your lambda expression with method reference.

There are following types of method references in java:

1. Reference to a static method.
2. Reference to an instance method.
3. Reference to a constructor.

1. Reference to a static method: You can refer to static method defined in the class.

Syntax:

ContainingClass::staticMethodName

2. Reference to an instance method: like static methods, you can refer instance methods also.

Syntax:

containingObject::instanceMethodName

3. Reference to a constructor: You can refer a constructor by using the new keyword.

Syntax:

ClassName::**new**

**Functional interfaces**: An Interface that contains only one abstract method is known as functional interface. It can have any number of default and static methods. It can also declare methods of object class.

Functional interfaces are also known as Single Abstract Method Interfaces (SAM Interfaces).

**Stream API:** Java 8 java.util.stream package consists of classes, interfaces and an enum to allow functional-style operations on the elements. It performs lazy computation. So, it executes only when it requires.

* Stream does not store elements. It simply conveys elements from a source such as a data structure, an array, or an I/O channel, through a pipeline of computational operations.
* Stream is functional in nature. Operations performed on a stream does not modify it's source. For example, filtering a Stream obtained from a collection produces a new Stream without the filtered elements, rather than removing elements from the source collection.
* Stream is lazy and evaluates code only when required.
* The elements of a stream are only visited once during the life of a stream. Like an Iterator, a new stream must be generated to revisit the same elements of the source.

You can use stream to filter, collect, print, and convert from one data structure to other etc. In the following examples, we have apply various operations with the help of stream.

Stream API has many methods which you can read from here

<https://www.javatpoint.com/java-8-stream>

**Optional:** It is a public final class which is used to deal with NullPointerException in Java application. We must import *java.util* package to use this class. It provides methods to check the presence of value for particular variable

Optional has many methods which you can read from here

<https://www.javatpoint.com/java-8-optional>