Home Java Programs OOPs String Exception Multithreading

⇧ SCROLL TO TOP

Java Lambda Expressions

Lambda expression is a new and important feature of Java which was included in Java SE 8. It provides a clear and concise way to represent one method interface using an expression. It is very useful in collection library. It helps to iterate, filter and extract data from collection.

The Lambda expression is used to provide the implementation of an interface which has functional interface. It saves a lot of code. In case of lambda expression, we don't need to define the method again for providing the implementation. Here, we just write the implementation code.

Java lambda expression is treated as a function, so compiler does not create .class file.

Functional Interface

Lambda expression provides implementation of *functional interface*. An interface which has only one abstract method is called functional interface. Java provides an anotation @*FunctionalInterface*, which is used to declare an interface as functional interface.

Why use Lambda Expression

1. To provide the implementation of Functional interface.
2. Less coding.

Java Lambda Expression Syntax

⇧ SCROLL TO TOP

```
body}
```

Java lambda expression is consisted of three components.

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.

No Parameter Syntax

```
() -> {
//Body of no parameter lambda
}
```

One Parameter Syntax

```
(p1) -> {
//Body of single parameter lambda
}
```

Two Parameter Syntax

```
(p1,p2) -> {
//Body of multiple parameter lambda
}
```

Let's see a scenario where we are not implementing Java lambda expression. Here, we are implementing an interface without using lambda expression.

⇧ SCROLL TO TOP

Without Lambda Expression

```
interface Drawable{
    public void draw();
}
public class LambdaExpressionExample {
    public static void main(String[] args) {
        int width=10;

        //without lambda, Drawable implementation using anonymous class
        Drawable d=new Drawable(){
            public void draw(){System.out.println("Drawing "+width);}
        };
        d.draw();
    }
}
```

Test it Now

Output:

```
Drawing 10
```

Java Lambda Expression Example

Now, we are going to implement the above example with the help of Java lambda expression.

```
@FunctionalInterface  //It is optional
interface Drawable{
    public void draw();
}

public class LambdaExpressionExample2 {
    public static void main(String[] args) {
```

⇧ SCROLL TO TOP

```
        //with lambda
        Drawable d2=()->{
          System.out.println("Drawing "+width);
        };
        d2.draw();
    }
}
```

Test it Now

Output:

```
Drawing 10
```

A lambda expression can have zero or any number of arguments. Let's see the examples:

Java Lambda Expression Example: No Parameter

```
interface Sayable{
    public String say();
}
public class LambdaExpressionExample3{
public static void main(String[] args) {
    Sayable s=()->{
        return "I have nothing to say.";
    };
    System.out.println(s.say());
}
}
```

Test it Now

Output:

⇧ SCROLL TO TOP

```
I have nothing to say.
```

Java Lambda Expression Example: Single Parameter

```
interface Sayable{
    public String say(String name);
}

public class LambdaExpressionExample4{
    public static void main(String[] args) {

        // Lambda expression with single parameter.
        Sayable s1=(name)->{
            return "Hello, "+name;
        };
        System.out.println(s1.say("Sonoo"));

        // You can omit function parentheses
        Sayable s2= name ->{
            return "Hello, "+name;
        };
        System.out.println(s2.say("Sonoo"));
    }
}
```

Test it Now

Output:

```
Hello, Sonoo
Hello, Sonoo
```

Java Lambda Expression Example: Multiple Parameters

⇧ SCROLL TO TOP

```
interface Addable{
    int add(int a,int b);
}

public class LambdaExpressionExample5{
    public static void main(String[] args) {

        // Multiple parameters in lambda expression
        Addable ad1=(a,b)->(a+b);
        System.out.println(ad1.add(10,20));

        // Multiple parameters with data type in lambda expression
        Addable ad2=(int a,int b)->(a+b);
        System.out.println(ad2.add(100,200));
    }
}
```

Test it Now

Output:

```
30
300
```

Java Lambda Expression Example: with or without return keyword

In Java lambda expression, if there is only one statement, you may or may not use return keyword. You must use return keyword when lambda expression contains multiple statements.

⇧ SCROLL TO TOP

```
b);
```

```
}

public class LambdaExpressionExample6 {
    public static void main(String[] args) {

        // Lambda expression without return keyword.
        Addable ad1=(a,b)->(a+b);
        System.out.println(ad1.add(10,20));

        // Lambda expression with return keyword.
        Addable ad2=(int a,int b)->{
                            return (a+b);
                            };
        System.out.println(ad2.add(100,200));
    }
}
```

Test it Now

Output:

```
30
300
```

Java Lambda Expression Example: Foreach Loop

```
import java.util.*;
public class LambdaExpressionExample7{
    public static void main(String[] args) {

        List<String> list=new ArrayList<String>();
        list.add("ankit");
        list.add("mayank");
                        ");
```

⇧ SCROLL TO TOP

```
        list.forEach(
            (n)->System.out.println(n)
        );
    }
}
```

Test it Now

Output:

```
ankit
mayank
irfan
jai
```

Java Lambda Expression Example: Multiple Statements

```
@FunctionalInterface
interface Sayable{
    String say(String message);
}

public class LambdaExpressionExample8{
    public static void main(String[] args) {

        // You can pass multiple statements in lambda expression
        Sayable person = (message)-> {
            String str1 = "I would like to say, ";
            String str2 = str1 + message;
            return str2;
        };
            System.out.println(person.say("time is precious."));
```

⇧ SCROLL TO TOP

Test it Now

Output:

```
I would like to say, time is precious.
```

Java Lambda Expression Example: Creating Thread

You can use lambda expression to run thread. In the following example, we are implementing run method by using lambda expression.

```
public class LambdaExpressionExample9{
    public static void main(String[] args) {

        //Thread Example without lambda
        Runnable r1=new Runnable(){
            public void run(){
                System.out.println("Thread1 is running...");
            }
        };
        Thread t1=new Thread(r1);
        t1.start();
        //Thread Example with lambda
        Runnable r2=()->{
                System.out.println("Thread2 is running...");
        };
        Thread t2=new Thread(r2);
        t2.start();
    }
}
```

Test it Now

Output:

⇧ SCROLL TO TOP

```
Thread1 is running...
Thread2 is running...
```

Java lambda expression can be used in the collection framework. It provides efficient and concise way to iterate, filter and fetch data. Following are some lambda and collection examples provided.

Java Lambda Expression Example: Comparator

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
class Product{
    int id;
    String name;
    float price;
    public Product(int id, String name, float price) {
        super();
        this.id = id;
        this.name = name;
        this.price = price;
    }
}
public class LambdaExpressionExample10{
    public static void main(String[] args) {
        List<Product> list=new ArrayList<Product>();

        //Adding Products
        list.add(new Product(1,"HP Laptop",25000f));
        list.add(new Product(3,"Keyboard",300f));
        list.add(new Product(2,"Dell Mouse",150f));

        ntln("Sorting on the basis of name...");
```

⇧ SCROLL TO TOP

```
        // implementing lambda expression
        Collections.sort(list,(p1,p2)->{
        return p1.name.compareTo(p2.name);
        });
        for(Product p:list){
            System.out.println(p.id+" "+p.name+" "+p.price);
        }

    }
}
```

Test it Now

Output:

```
Sorting on the basis of name...
2 Dell Mouse 150.0
1 HP Laptop 25000.0
3 Keyboard 300.0
```

Java Lambda Expression Example: Filter Collection Data

```
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Stream;
class Product{
    int id;
    String name;
    float price;
    public Product(int id, String name, float price) {
        super();
        this.id = id;
        this.name = name;
        ce;
```

⇧ SCROLL TO TOP

```
    }
}
public class LambdaExpressionExample11{
    public static void main(String[] args) {
        List<Product> list=new ArrayList<Product>();
        list.add(new Product(1,"Samsung A5",17000f));
        list.add(new Product(3,"Iphone 6S",65000f));
        list.add(new Product(2,"Sony Xperia",25000f));
        list.add(new Product(4,"Nokia Lumia",15000f));
        list.add(new Product(5,"Redmi4 ",26000f));
        list.add(new Product(6,"Lenevo Vibe",19000f));

        // using lambda to filter data
        Stream<Product> filtered_data = list.stream().filter(p -> p.price > 20000);

        // using lambda to iterate through collection
        filtered_data.forEach(
                product -> System.out.println(product.name+": "+product.price)
        );
    }
}
```

Test it Now

Output:

```
Iphone 6S: 65000.0
Sony Xperia: 25000.0
Redmi4 : 26000.0
```

Java Lambda Expression Example: Event Listener

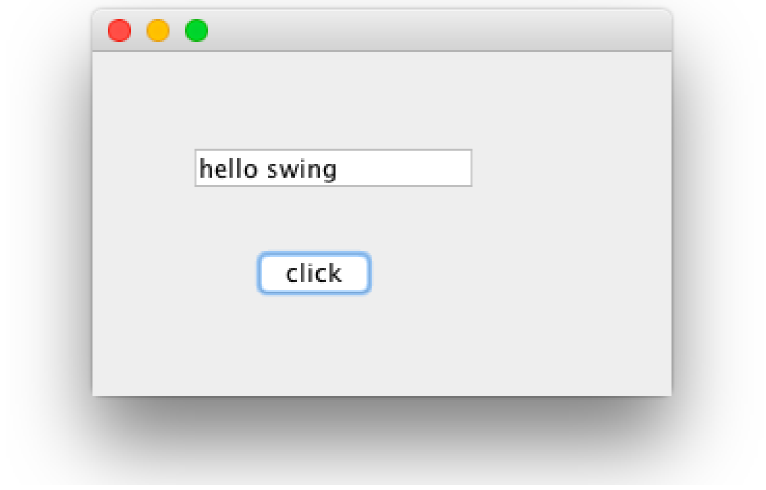
```
import javax.swing.JButton;
            .JFrame;
             .JTextField;
```

⇧ SCROLL TO TOP

```
public class LambdaEventListenerExample {
    public static void main(String[] args) {
        JTextField tf=new JTextField();
        tf.setBounds(50, 50,150,20);
        JButton b=new JButton("click");
        b.setBounds(80,100,70,30);

        // lambda expression implementing here.
        b.addActionListener(e-> {tf.setText("hello swing");});

        JFrame f=new JFrame();
        f.add(tf);f.add(b);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setLayout(null);
        f.setSize(300, 200);
        f.setVisible(true);

    }

}
```

Output:

⇧ SCROLL TO TOP



← Prev Next →

Youtube For Videos Join Our Youtube Channel: Join Now

Feedback

- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share

⇧ SCROLL TO TOP

Learn Latest Tutorials

Splunk tutorial
Splunk

SPSS tutorial
SPSS

Swagger tutorial
Swagger

T-SQL tutorial
Transact-SQL

Tumblr tutorial
Tumblr

React tutorial
ReactJS

Regex tutorial
Regex

Reinforcement learning tutorial
Reinforcement Learning

R Programming tutorial
R Programming

RxJS tutorial
RxJS

React Native tutorial
React Native

Python Design Patterns
Python Design Patterns

Python Pillow tutorial
Python Pillow

Python Turtle tutorial
Python Turtle

Keras tutorial
Keras

⇧ SCROLL TO TOP

Preparation

Aptitude

Aptitude

Logical Reasoning

Reasoning

Verbal Ability

Verbal Ability

Interview Questions

Interview Questions

Company Interview Questions

Company Questions

Trending Technologies

Artificial Intelligence Tutorial

Artificial Intelligence

AWS Tutorial

AWS

Selenium tutorial

Selenium

Cloud Computing tutorial

Cloud Computing

Hadoop tutorial

Hadoop

ReactJS Tutorial

ReactJS

Data Science Tutorial

Data Science

Angular 7 Tutorial

Angular 7

Blockchain Tutorial

Blockchain

Git Tutorial

Git

Machine Learning Tutorial

Machine Learning

DevOps Tutorial

DevOps

⇧ SCROLL TO TOP

B.Tech / MCA

DBMS tutorial

DBMS

Data Structures tutorial

Data Structures

DAA tutorial

DAA

Operating System tutorial

Operating System

Computer Network tutorial

Computer Network

Compiler Design tutorial

Compiler Design

Computer Organization and Architecture

Computer Organization

Discrete Mathematics Tutorial

Discrete Mathematics

Ethical Hacking Tutorial

Ethical Hacking

Computer Graphics Tutorial

Computer Graphics

Software Engineering Tutorial

Software Engineering

html tutorial

Web Technology

Cyber Security tutorial

Cyber Security

Automata Tutorial

Automata

C Language tutorial

C Programming

C++ tutorial

C++

Java tutorial

Java

.Net Framework tutorial

.Net

Python tutorial

Python

List of Programs

Programs

⇧ SCROLL TO TOP

Control Systems tutorial

Control System

Data Mining Tutorial

Data Mining

Data Warehouse Tutorial

Data Warehouse

⇧ SCROLL TO TOP