Java 8 features:-

1. Lambda Expression:-

-> It is Anonymous function which is having no name, no returnType,no modifier.

It Provide The implementation of Functional interface.

It is applicable for only functional Interface.

Sysntax:-

()->{System.out.println(“Hii”);

2. Functional Interface:-

-> Inteface contain only one abstract method .

It can contain static ,default method.

Ex.Runnable,callable

Ex:- 1

@ FunctionalInterface

**interface** Test5{

**public** **void** m1();

}

**public** **class** StringClass{

**public** **static** **void** main(String[] args) {

Test5 t=()->{

System.***out***.println("hii");

};

t.m1();

}

}

Ex:-2

@ FunctionalInterface

**interface** Test5{

**public** **int** m1(**int** a,**int** b);

}

**public** **class** StringClass{

**public** **static** **void** main(String[] args) {

Test5 t=(a,b)->{

System.***out***.println("hii");

**return** a\*b;

};

**int** c=t.m1(50,40);

System.***out***.println(c);

}

}

Ex:-3 Runnable to lambda?

C**lass** Test5 **implements** Runnable{

**public** **void** run() {

System.***out***.println("hii");

}

}

**public** **class** StringClass{

**public** **static** **void** main(String[] args) {

Test5 t= **new** Test5();

Thread d= **new** Thread(t);

d.start();

}

}

**public** **class** StringClass{

**public** **static** **void** main(String[] args) {

Runnable r=()->{

System.***out***.println("hii");

};

Thread d= **new** Thread();

d.start();

}

}

Lambda Expression with collection:-

**public** **class** StringClass{

**public** **static** **void** main(String[] args) {

ArrayList al= **new** ArrayList();

al.add(2);

al.add(4);

al.add(3);

al.add(6);

Comparator<Integer> c=(al1,al2)->(al1>al2)?1:(al1<al2)?-1:0;

Collections.*sort*(al,c);

System.***out***.println(al);

}

}

Employee1 e= **new** Employee1(20,"rahul",4000);

Employee1 e1= **new** Employee1(21,"m",23423);

Employee1 e2= **new** Employee1(19,"p",4353);

ArrayList<Employee1> al= **new** ArrayList<Employee1>();

al.add(e);

al.add(e1);

al.add(e2);

List<Employee1> l=al.stream().filter(i->i.salary>4353).collect(Collectors.*toList*());

**for**(Employee1 a:l){

System.***out***.println(a.age+a.name+a.salary);

}

Or. Predicate<Employee1> p=i->i.salary>3000;

No is even or not

**int** a[]={2,5,32,7,47};

Predicate<Integer> p= i->(i%2)==0;

**for**(**int** x1:a){

**if**(p.test(x1)){

System.***out***.println(x1);

}

}