Method And Constructor References by using :: (Double Colon) Operator

- Functional Interface method can be mapped to our specified method by using :: (double colon) operator. This is called method reference.
- Our specified method can be either static method or instance method.
- Functional Interface method and our specified method should have same argument types, except this the remaining things like return type, methodname, modifiers etc are not required to match.

Syntax

If our specified method is static method	If the method is instance method
Classname::methodName	Objref::methodName

- Functional Interface can refer lambda expression and Functional Interface can also refer method reference. Hence lambda expression can be replaced with method reference. Hence method reference is alternative syntax to lambda expression.

Ex: With Lambda Expression

```
1)  class Test {
2)      public static void main(String[] args) {
3)              Runnable r = () → {
4)                                  for(int i=0; i<=10; i++) {
5)
    System.out.println("Child Thread");
6)                                  }
7)                                 };
8)              Thread t = new Thread(r);
9)              t.start();
10)             for(int i=0; i<=10; i++) {
11)                 System.out.println("Main Thread");
12)             }
13)     }
14) }
```

With Method Reference

```
1)  class Test {
2)      public static void m1() {
```


DURGASOFT, # 202, 2nd Floor, HUDA Maitrivanam, Ameerpet, Hyderabad - 500038,
☎ 040 – 64 51 27 86, 80 96 96 96 96, 92 46 21 21 43 | www.durgasoft.com


```
3)              for(int i=0; i<=10; i++) {
4)                   System.out.println("Child Thread");
5)              }
6)   }
7)   public static void main(String[] args) {
8)             Runnable r = Test:: m1;
9)             Thread t = new Thread(r);
10)             t.start();
11)             for(int i=0; i<=10; i++) {
12)                  System.out.println("Main Thread");
13)             }
14) }
```

In the above example Runnable interface run() method referring to Test class static method m1().
Method reference to Instance method:

Ex:

```
1)   interface Interf {
2)        public void m1(int i);
3)    }
4)   class Test {
5)         public void m2(int i) {
6)              System.out.println("From Method Reference:"+i);
7)         }
8)         public static void main(String[] args) {
9)              Interf  f = I ->sop("From Lambda Expression:"+i);
10)             f.m1(10);
11)             Test t = new Test();
12)             Interf i1 = t::m2;
13)             i1.m1(20);
14)       }
15) }
```

In the above example functional interface method m1() referring to Test class instance method m2().
The main advantage of method reference is we can use already existing code to implement functional interfaces (code reusability).

Constructor References

We can use :: (double colon)operator to refer constructors also

Syntax: classname :: new

Ex:

```
Interf f = sample :: new;
```

functional interface f referring sample class constructor

Ex:

```
1)  class Sample {
2)          private String s;
3)          Sample(String s) {
4)                  this.s = s;
5)                  System.out.println("Constructor Executed:"+s);
6)           }
7)  }
8)  interface Interf {
9)         public Sample get(String s);
10) }
11) class Test {
12)         public static void main(String[] args) {
13)               Interf f = s -> new Sample(s);
14)               f.get("From Lambda Expression");
15)               Interf f1 = Sample :: new;
16)               f1.get("From Constructor Reference");
17)         }
18) }
```

Note: In method and constructor references compulsory the argument types must be matched.