Experiences Unlimited

RAMBLINGS OF A DEVELOPER

SATURDAY, MAY 26TH, 2018

JAVA

Introduction to Functional Interfaces – A concept recreated in Java 8

BY MOHAMED SANAULLA ON MARCH 21, 2013 • (9 COMMENTS)

Any java developer around the world would have used at least one of the following interfaces: java.lang.Runnable, java.awt.event.ActionListener, java.util.Comparator, java.util.concurrent.Callable. There is some common feature among the stated interfaces and that feature is they have only one method declared in their interface definition. There are lot more such interfaces in JDK and also lot more created by java developers. These interfaces are also called **S**ingle **A**bstract **M**ethod interfaces (SAM Interfaces). And a popular way in which these are used is by creating Anonymous Inner classes using these interfaces, something like:

```
1  public class AnonymousInnerClassTest {
2    public static void main(String[] args) {
3      new Thread(new Runnable() {
4        @Override
5        public void run() {
6          System.out.println("A thread created and running ..
7        }
8      }).start();
9    }
10 }
```

With Java 8 the same concept of SAM interfaces is recreated and are called Functional interfaces. These can be represented using Lambda expressions (http://blog.sanaulla.info/2013/03/11/using-lambda-expression-to-sort-a-list-in-java-8-using-netbeans-lambda-support/), Method reference and constructor references(I will cover these two topics in the upcoming blog posts). There's an annotation introduced- @FunctionalInterface

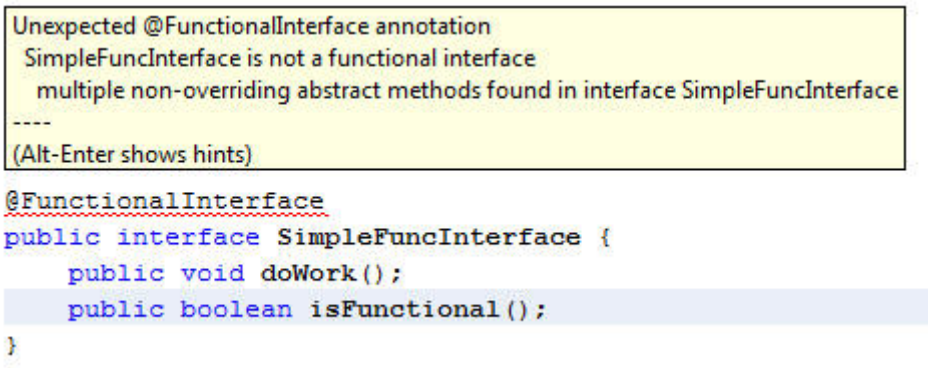
(http://download.java.net/lambda/b78/docs/api/java/lang/FunctionalInterface.html) which can be used for compiler level errors when the interface you have annotated is not a valid Functional Interface. Lets try to have a look at a simple functional interface with only one abstract method:

```
1  @FunctionalInterface
2  public interface SimpleFuncInterface {
3    public void doWork();
4  }
```

The interface can also declare the abstract methods from the java.lang.Object class, but still the interface can be called as a Functional Interface:

```
1  @FunctionalInterface
2  public interface SimpleFuncInterface {
3    public void doWork();
4    public String toString();
5    public boolean equals(Object o);
6  }
```

Once you add another abstract method to the interface then the compiler/IDE will flag it as an error as shown in the screenshot below:



(https://sanaulla.files.wordpress.com/2013/03/funcinterface_error1.jpg)

Interface can extend another interface and in case the Interface it is extending in functional and it doesn't declare any new abstract methods then the new interface is also functional. But an interface can have one abstract method and any number of default methods and the interface would still be called an functional interface. To get an idea of default methods please read here (http://blog.sanaulla.info/2013/03/20/introduction-to-default-methods-defender-methods-in-java-8/).

```
1  @FunctionalInterface
2  public interface ComplexFunctionalInterface extends SimpleFu
3    default public void doSomeWork(){
4      System.out.println("Doing some work in interface impl...
5    }
6    default public void doSomeOtherWork(){
7      System.out.println("Doing some other work in interface i
8    }
9  }
```

The above interface is still a valid functional interface. Now lets see how we can use the lambda expression as against anonymous inner class for implementing functional interfaces:

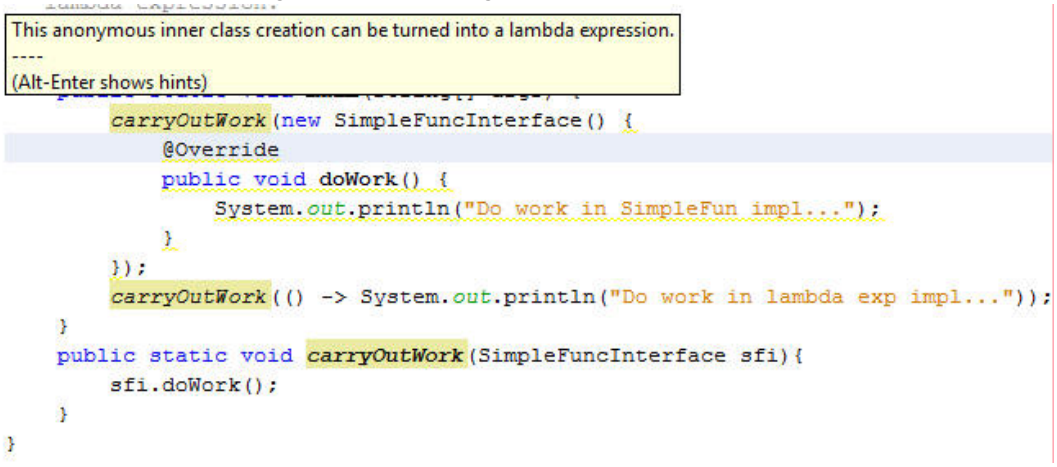
```
1  /*
2  * Implementing the interface by creating an
```

```
 * anonymous inner class versus using
 * lambda expression.
 */
public class SimpleFunInterfaceTest {
  public static void main(String[] args) {
    carryOutWork(new SimpleFuncInterface() {
      @Override
      public void doWork() {
        System.out.println("Do work in SimpleFun impl...");
      }
    });
    carryOutWork(() -> System.out.println("Do work in lambd
  }
  public static void carryOutWork(SimpleFuncInterface sfi){
    sfi.doWork();
  }
}
```

And the output would be ...

```
Do work in SimpleFun impl...
Do work in lambda exp impl...
```

In case you are using an IDE which supports the Java Lambda expression syntax(Netbeans 8 Nightly builds (http://bertram2.netbeans.org:8080/job/jdk8lambda/lastSuccessfulBuild/artifact/nbbuild/)) then it provides an hint when you use an anonymous inner class as used above



(https://sanaulla.files.wordpress.com/2013/03/funcinterface_hint.jpg)

This was a brief introduction to the concept of functional interfaces in java 8 and also how they can be implemented using Lambda expressions.

9 replies »

Congratulations for the articles.

Reply

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=1782&_wpnonce=68bc1283cb)
Like

> Thanks a lot!
>
> Reply
>
> ★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=1783&_wpnonce=3cf753770b)
> Like

Thank you for this informative article. 🙂

Reply

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=1784&_wpnonce=7c099ae476)
Like

Going through ur articles. Thanks a lot. I like to suggest as why not a downl Reply with all ur articles in it in pdf format. So that people like me can view offline and can also look as a book when needed. As new articles come in, can be added to pdf as well. Please consider about this.

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=1785&_wpnonce=c4ba94021b)
Like

1.What is functional programming? Why? Explain with real time scenario. Reply
2.What is functional Interface and what purpose it has been introduced in Java and what can we achieve with functional Interfaces?
3.Without functional interfaces, what can't we do in java 8?

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=1787&_wpnonce=124847911a)
Like

The same content everywhere in different websites about the Functional In Reply Who is copying from where ? Are you the real author ?? Please expand it further with your own thoughts. We will appreciate you.

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=3140&_wpnonce=2edc78a10b)
Like

> Regarding same content being everywhere – people cannot tell an apple Reply orange. It has to be explained as apple only.
> Regarding your first question- you can judge who copied from where based on the time when it was posted.
> I have expanded in my own thoughts.
> Thanks for reading the article

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=3141&_wpnonce=4640623a2d)
Like

Hello sir, this is really very useful for me.
you explained very well in detail
thank you so much

Reply

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=3761&_wpnonce=a642323e22)
Like

Above all the examples of java 8 new features are quite useful and well exp
a simple way.
Good explanation of Functional Interfaces in java 8 version.

Reply

★(https://sanaulla.info/2013/03/21/introduction-to-functional-interfaces-a-concept-recreated-in-java-8/?like_comment=4743&_wpnonce=1ad1e58edd)
Like