ToiletPaper #119

Kotlin/Java Memory Leaks

Author: Marco Pfattner / Senior Software Architect/ Business Division Automotive Bavaria

✕ Problem

When using Kotlin and Java, memory leaks can occur in combination with Java Lambdas. Example: a listener is created in Kotlin, which is then registered with a Java class and allegedly unregistered again:

```java
public class Calculator {
    interface Listener {
        void onResult(int result);
    }
    Set<Listener> listeners = new HashSet<>();
    void addListener(Listener listener) {
        listeners.add(listener);
    }
    void removeListener(Listener listener) {
        listeners.remove(listener);
    }
    ...
}
```

```kotlin
fun main() {
    val c = Calculator()

    val listener = { sum: Int ->
        println("sum: $sum")
    }

    c.addListener(listener)
    c.removeListener(listener)
    c.sum(10, 20)
}
```

When running the Kotlin `main`-function you would not expect any output, but calling `c.removeListener` does not work as expected. To see the cause in the generated Java code, you can use Android Studio – click on "Tools → Kotlin → Show Kotlin Bytecode" followed by a click on "Decompile":

```
Object var10001 = listener;
if (listener != null) {
  var10001 = new MainKt$sam$test_Calculator_Listener$0(listener);
}
c.addListener((Listener)var10001);
var10001 = listener;
if (listener != null) {
  var10001 = new MainKt$sam$test_Calculator_Listener$0(listener);
}
c.removeListener((Listener)var10001);
```

`listener` is wrapped in `MainKt$sam$test_Calculator_Listener$0(listener)` each time before it is passed to `addListener` or `removeListener`, which then are two different Java objects.

✓ Solution

The declaration of `listener` can slightly be modified by declaring the type to avoid the need for a generated wrapper:

```kotlin
val listener = Calculator.Listener {
  sum: Int -> println("sum: $sum")
}
c.addListener(listener)
c.removeListener(listener)
```

```java
Listener listener = (Listener)null.INSTANCE;
c.addListener(listener);
c.removeListener(listener);
```

+ Further Aspects

- Kotlin under the hood: https://www.youtube.com/watch?v=Ta5wBJsC39s