ToiletPaper #119

Kotlin/Java Memory Leaks

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

✗ Problem

Bei der Verwendung von Kotlin und Java kann es im Zusammenspiel mit Java Lambdas zu Memory Leaks kommen. Beispiel: In Kotlin wird ein Listener erzeugt, der dann bei einer Java-Klasse registriert und vermeintlich wieder deregistriert wird:

```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)
}
```

Beim Ausführen der Kotlin `main`-Funktion würde man eigentlich keine Ausgabe erwarten, allerdings funktioniert der Aufruf von `c.removeListener` nicht wie erwartet. Ein Klick in Android Studio auf "Tools → Kotlin → Show Kotlin Bytecode" gefolgt von einem Klick auf "Decompile" zeigt die Ursache im generierten Java-Code:

```java
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` wird jeweils nochmal in `MainKt$sam$test_Calculator_Listener$0(listener)` gewrapped und erst dann dem `addListener` bzw. `removeListener` übergeben, was zu dem Zeitpunkt dann zwei unterschiedliche Java-Objekte sind.

✓ Lösung

Die Deklaration von `listener` kann leicht abgewandelt werden, damit entfällt dann der nötige generierte 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);
```

+ Weiterführende Aspekte

- Kotlin Under the Hood: https://www.youtube.com/watch?v=Ta5wBJsC39s