THE DEVELOPER'S CONFERENCE

Como as Value Classes podem melhorar a performance da sua aplicação Kotlin?

Miguel Jr

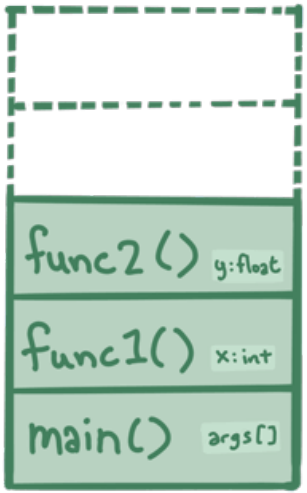
Agenda

THE DEVELOPER'S CONFERENCE

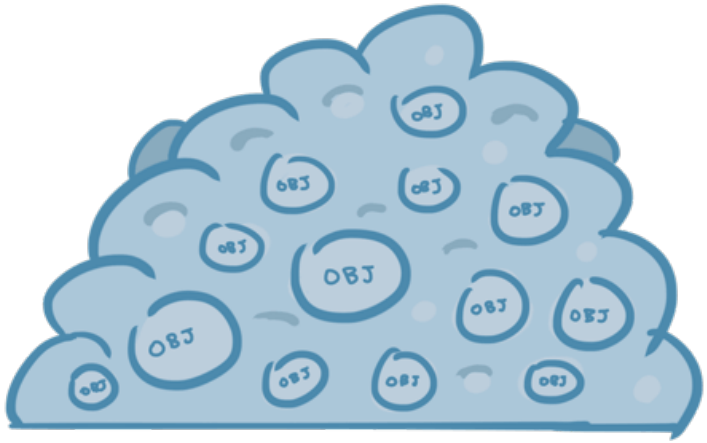
- Problema
- Objetivo
- Motivos do uso
- Por que não Type Alias?

Problema

THE DEVELOPER'S CONFERENCE



THE "STACK"



THE "HEAP"

Problema

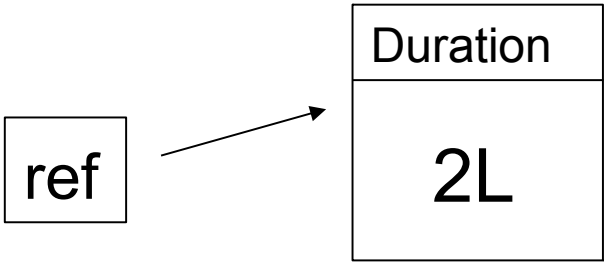
THE DEVELOPER'S CONFERENCE

```
data class Duration(val millis: Long)


fun greetAfterTimeout(duration: Duration): Unit {
    println("value = ${duration.millis}")
}
```

Ao chamar a função

greetAfterTimeout(Duration(2L))



Veja o que acontece...

THE DEVELOPER'S CONFERENCE

Objetivo

THE DEVELOPER'S CONFERENCE

"Code like a class. Works like an int "

Svetlana Isakova, Developer Advocate Jetbrains

Motivos do uso

THE DEVELOPER'S CONFERENCE

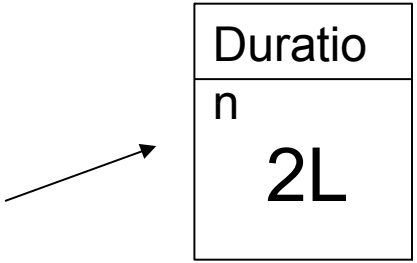
> Indicar unidade de medida

```
@JvmInline
value class Duration(val millis: Long)

fun greetAfterTimeout(duration: Duration): Unit {
    println("value = ${duration.millis}")
}
```

Ao chamar a função

greetAfterTimeout(Duration(2L))



Debaixo dos panos

THE DEVELOPER'S CONFERENCE

```
public final class Duration {
   private final long millis;
   public final long getMillis() { return this.value; }

   // $FF: synthetic method
   private Duration(long value) { this.value = value; }
   public static long constructor_impl(long value) { return value; }

   // $FF: synthetic method
   @NotNull
   public static final Duration box_impl(long m) { return new Duration(m); }

   // $FF: synthetic method
   public final long unbox_impl() { return this.value; }

   //more Object related implementations
}
```

Motivos do uso

THE DEVELOPER'S CONFERENCE

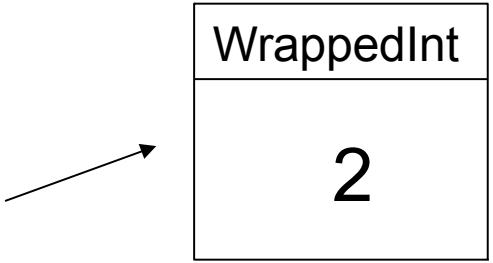
> Proteger de uso indevido

```
@JvmInline
value class WrappedInt(val value: Int)

fun showResult(wrapper: WrappedInt): Unit {
    println("result = $wrappedInt")
}
```

Ao chamar a função

showResult(WrappedInt(2))



Debaixo dos panos

THE DEVELOPER'S CONFERENCE

```
public final class WrappedInt {
   private final int value;
   public final int getValue() { return this.value; }

   // $FF: synthetic method
   private WrappedInt(int value) { this.value = value; }
   public static int constructor_impl(int value) { return value; }

   // $FF: synthetic method
   @NotNull
   public static final WrappedInt box_impl(int v) { return new WrappedInt(v); }

   // $FF: synthetic method
   public final int unbox_impl() { return this.value; }

   //more Object related implementations
}
```

Motivos do uso

THE DEVELOPER'S CONFERENCE

> Limitar escopo de função de extensão

```
fun String.asJson() = jacksonObjectMapper().readValue(this)
```

```
"""{ "x":200, "y":300 }""".asJson()
```

```
"teste".asJson()
```

```
@JvmInline
value class JsonString(val value: String)
fun JsonString.asJson() = jacksonObjectMapper().readValue(this.value)
```

Limitações

THE DEVELOPER'S CONFERENCE

```
@JvmInline value class Seconds()            // erro - precisa de um valor como construtor primário
@JvmInline value class Minutes(value: Int)  // erro - o valor necessita ser uma propriedade
@JvmInline value class Hours(var value: Int) // erro - a propriedade precisa ser somente leitura


// erro - não permite mais de uma propriedade sendo construtor primário
@JvmInline value class Years(val count: Int, val startYear: Int)


open class TimeUnit
@JvmInline value class Seconds(val value: Int) : TimeUnit() // erro - não pode herdar classes
@JvmInline open value class Minutes(val value: Int)        // erro - 'Value Class' é "final"


class TimeUnit {
    @JvmInline inner value class Minutes(val value: Int)  // erro - não pode ser classe interna
}
fun main() {
    // erro - não pode definir 'Value Class' numa função/rotina.
    @JvmInline value class Seconds(val value: Int)
}
// erro - modificador 'value' não é aplicavél numa 'enum class'
@JvmInline enum value class TimeUnits(val value: Int) {
    SECONDS_PER_MINUTE(60),
    MINUTES_PER_HOUR(60),
    HOURS_PER_DAY(24)
}
```

Limitações

THE DEVELOPER'S CONFERENCE

```
@JvmInline value class Passwords(val value: Int) {

  lateinit var partialContent : String      // erro – não pode ter inicialização tardia
  var letter by Delegates.notnull<Char>()   // erro – não pode ter propriedade delegada
  var counter = 0                           // erro – não pode ter backing fields
    set(value) { field = value + 1 }
    get() = ++field
}
interface Security {
  fun check() : Boolean
}

@JvmInline value class Passwords(val value: Int) : Security {       // isto é permitido

   override fun check() : Boolean = true
}
```

Glossário

Backing fields : *personalizar o acesso ao conteúdo das propriedades e/ou dos campos em memória.*

Por que não Type Aliases?

THE DEVELOPER'S CONFERENCE

```
typealias NameTypeAlias = String

@JvmInline
value class NameInlineClass(val s: String)

fun acceptString(s: String) {}
fun acceptNameTypeAlias(n: NameTypeAlias) {}
fun acceptNameInlineClass(p: NameInlineClass) {}

fun main() {
    val nameAlias: NameTypeAlias = ""
    val nameInlineClass: NameInlineClass = NameInlineClass("")
    val string: String = ""

    acceptString(nameAlias) // OK: passe o alias em vez do tipo principal
    acceptString(nameInlineClass) // Erro de compilação : não pode passar value class ao invés do tipo principal

    // e vice versa:
    acceptNameTypeAlias(string) // OK: passe o alias em vez do tipo principal
    acceptNameInlineClass(string) // Erro de compilação : não pode passar value class ao invés do tipo principal
}
```

Teste de Performance

THE DEVELOPER'S CONFERENCE

```
Benchmark                                  Mode  Cnt  Score   Error  Units
Benchmark_2.benchmarkClassMillis           avgt    2  0,004          ms/op
Benchmark_2.benchmarkClassMillis:·stack    avgt         NaN          ---
Benchmark_2.benchmarkClassSeconds          avgt    2  0,004          ms/op
Benchmark_2.benchmarkClassSeconds:·stack   avgt         NaN          ---
Benchmark_2.benchmarkValueClassMillis      avgt    2  0,003          ms/op
Benchmark_2.benchmarkValueClassMillis:·stack avgt       NaN          ---
Benchmark_2.benchmarkValueClassSeconds     avgt    2  0,003          ms/op
Benchmark_2.benchmarkValueClassSeconds:·stack avgt      NaN          ---
```

Fonte do Java Microbenchmark Harness (JMH) :
https://github.com/openjdk/jmh

THE DEVELOPER'S CONFERENCE

Obrigado!!

Dúvidas?

migueljrdev