Deep Dive Kotlin : du Hello World au ByteCode

#DevoxxFR #Kotlin @emmanuelVinas @ilaborie

DEVOXX France

Emmanuel Vinas

Expert Android & Java

@emmanuelvinas

emmanuel@monkeypatch.io

Igor Laborie

Expert Java & Web

@ilaborie

igor@monkeypatch.io

Roadmap

I. ByteCode Java ?
II. Introduction Kotlin
III. Les bases
IV. null-safety
V. Fonction
VI. Lambda
VII. Class
VIII. Types
IX. Extensions de fonction
X. Structure
XI. Pause
XII. ByteCode Android
XIII. Collection
XIV. Delegate
XV. Plus sur les fonctions
XVI. Serialization
XVII. Coroutines
XVIII. DSL
XIX. Conclusion

ByteCode Java ?

HelloWorld.java


```
package _00_helloworld;

public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello Devoxx");
    }
}
```

```
$ javac HelloWorld.java
```

Java ByteCode binary


```
$ hexdump -C HelloWorld.class
```

```
00000000  ca fe ba be 00 00 00 34  00 1d 0a 00 06 00 0f 09  |.......4........|
00000010  00 10 00 11 08 00 12 0a  00 13 00 14 07 00 15 07  |................|
00000020  00 16 01 00 06 3c 69 6e  69 74 3e 01 00 03 28 29  |.....<init>...()|
00000030  56 01 00 04 43 6f 64 65  01 00 0f 4c 69 6e 65 4e  |V...Code...LineN|
00000040  75 6d 62 65 72 54 61 62  6c 65 01 00 04 6d 61 69  |umberTable...mai|
00000050  6e 01 00 16 28 5b 4c 6a  61 76 61 2f 6c 61 6e 67  |n...([Ljava/lang|
00000060  2f 53 74 72 69 6e 67 3b  29 56 01 00 0a 53 6f 75  |/String;)V...Sou|
00000070  72 63 65 46 69 6c 65 01  00 0f 48 65 6c 6c 6f 57  |rceFile...HelloW|
00000080  6f 72 6c 64 2e 6a 61 76  61 0c 00 07 00 08 07 00  |orld.java.......|
00000090  17 0c 00 18 00 19 01 00  0c 48 65 6c 6c 6f 20 44  |.........Hello D|
000000a0  65 76 6f 78 78 07 00 1a  0c 00 1b 00 1c 01 00 19  |evoxx...........|
000000b0  5f 30 30 5f 68 65 6c 6c  6f 77 6f 72 6c 64 2f 48  |_00_helloworld/H|
000000c0  65 6c 6c 6f 57 6f 72 6c  64 01 00 10 6a 61 76 61  |elloWorld...java|
000000d0  2f 6c 61 6e 67 2f 4f 62  6a 65 63 74 01 00 10 6a  |/lang/Object...j|
000000e0  61 76 61 2f 6c 61 6e 67  2f 53 79 73 74 65 6d 01  |ava/lang/System.|
000000f0  00 03 6f 75 74 01 00 15  4c 6a 61 76 61 2f 69 6f  |..out...Ljava/io|
00000100  2f 50 72 69 6e 74 53 74  72 65 61 6d 3b 01 00 13  |/PrintStream;...|
00000110  6a 61 76 61 2f 69 6f 2f  50 72 69 6e 74 53 74 72  |java/io/PrintStr|
00000120  65 61 6d 01 00 07 70 72  69 6e 74 6c 6e 01 00 15  |eam...println...|
00000130  28 4c 6a 61 76 61 2f 6c  61 6e 67 2f 53 74 72 69  |(Ljava/lang/Stri|
00000140  6e 67 3b 29 56 00 21 00  05 00 06 00 00 00 00 00  |ng;)V.!.........|
00000150  02 00 01 00 07 00 08 00  01 00 09 00 00 00 1d 00  |................|
```

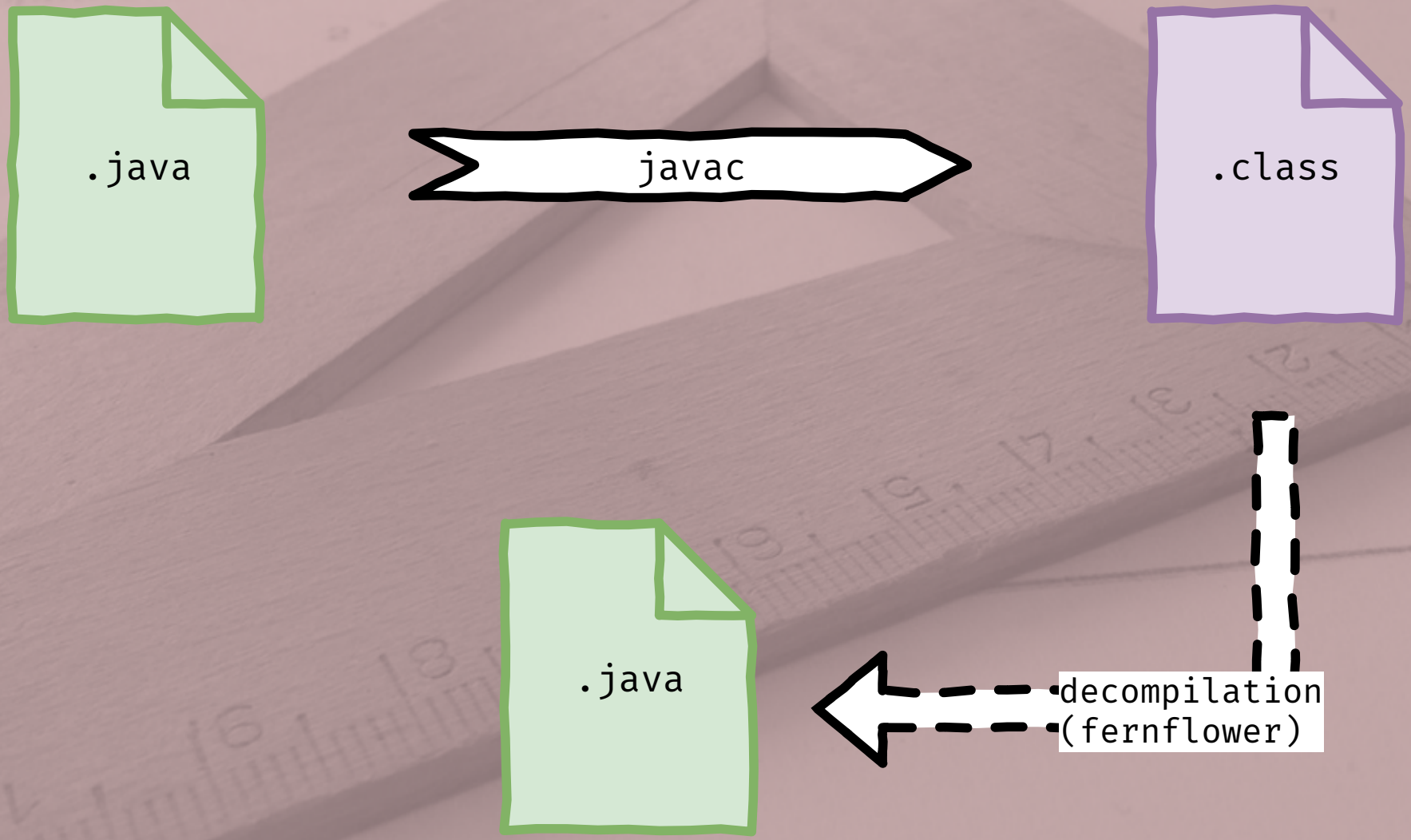
Explorons le ByteCode


```
$ javap -c HelloWorld.class
```

```
Compiled from "HelloWorld.java"
public class _00_helloworld.HelloWorld {
  public _00_helloworld.HelloWorld();
    Code:
       0: aload_0
       1: invokespecial #1                  // Method java/lang/Object."":()V
       4: return

  public static void main(java.lang.String[]);
    Code:
       0: getstatic     #2                  // Field java/lang/System.out:Ljav
       3: ldc           #3                  // String Hello Devoxx
       5: invokevirtual #4                  // Method java/io/PrintStream.prin
       8: return
}
```

Transpile

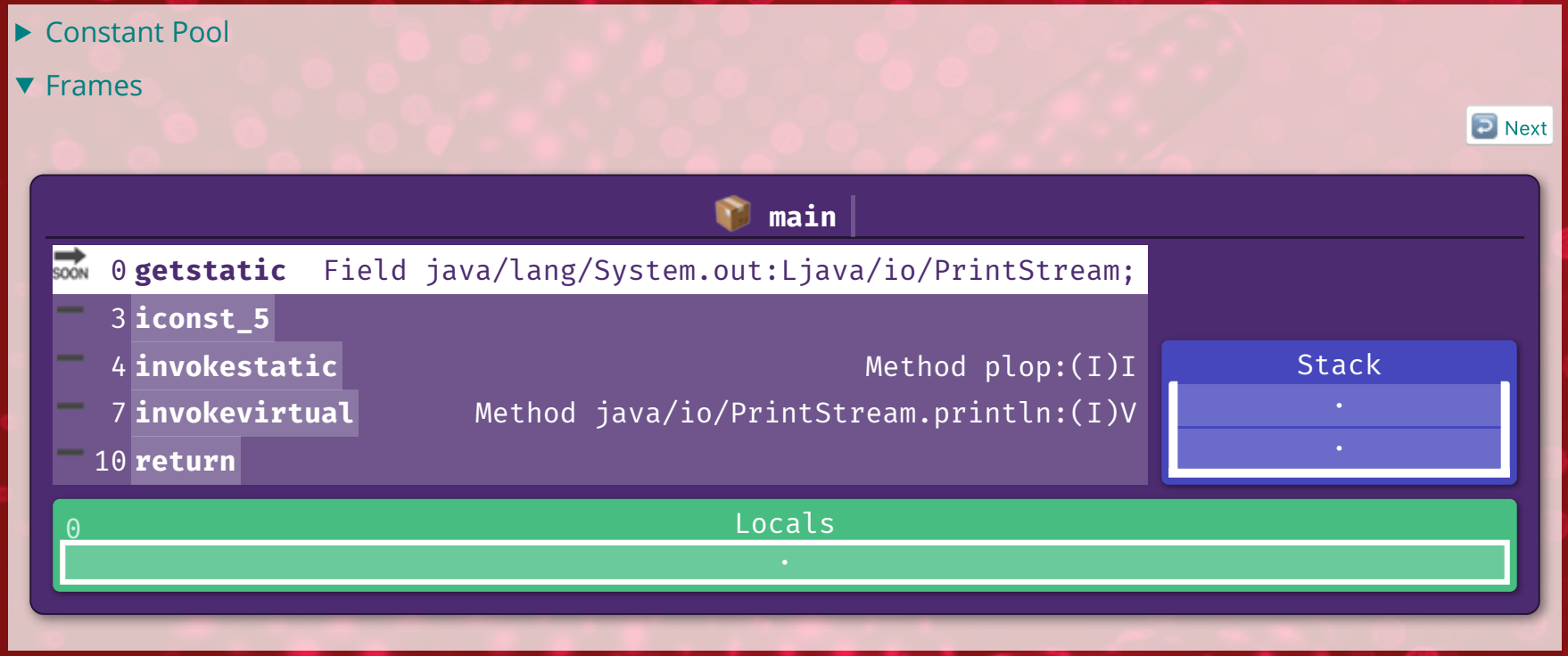


À propos du ByteCode

- environ 200 opérations possibles (maxi. 256 opscodes)
- préfix pour le type d'opérations (`i` pour entier, `d` pour double, ...)

- manipulation de la pile, des variables locales (`iconst_0`, `istore`, `iload`, ...)
- controle du flux des instructions (`if_icmpgt`, `goto`, ...)
- manipulation d'objets (`invokevirtual`, `invokedynamic`, ...)
- arithmétiques et conversion de type (`iadd`, `iinc`, `i2d`, ...)
- autres (`athrow`, ...)

Jouons un peu



Liens

- Mastering Java Bytecode at the Core of the JVM
- Introduction to Java Bytecode
- The Java® Virtual Machine Specification
- The Java Virtual Machine Instruction Set

> Soyez curieux: regardez comment ça marche avec `javap -c`

Introduction Kotlin

Historique

- **2011**

  Dévoilé par JetBrains
- **2016:**

  v1.0

  Supporté par Spring Framework
- **2017:**

  v1.1: coroutines, ...

  Officiellement supportée par Google

  v1.2: multiplatform
- **2018:**

  Kotlin Native (external) 0.6

Cibles

HelloWorld.kt


```kotlin
package _00_helloworld

fun main(args: Array<String>) {
    println("Hello Devoxx")
}
```

```
$ kotlinc HelloWorld.kt
```

hexdump


```
00000000  ca fe ba be 00 00 00 32  00 33 01 00 1b 5f 30 30  |.......2.3..._00|
00000010  5f 68 65 6c 6c 6f 77 6f  72 6c 64 2f 48 65 6c 6c  |_helloworld/Hell|
00000020  6f 57 6f 72 6c 64 4b 74  07 00 01 01 00 10 6a 61  |oWorldKt......ja|
00000030  76 61 2f 6c 61 6e 67 2f  4f 62 6a 65 63 74 07 00  |va/lang/Object..|
00000040  03 01 00 04 6d 61 69 6e  01 00 16 28 5b 4c 6a 61  |....main...([Lja|
00000050  76 61 2f 6c 61 6e 67 2f  53 74 72 69 6e 67 3b 29  |va/lang/String;)|
00000060  56 01 00 23 4c 6f 72 67  2f 6a 65 74 62 72 61 69  |V..#Lorg/jetbrai|
00000070  6e 73 2f 61 6e 6e 6f 74  61 74 69 6f 6e 73 2f 4e  |ns/annotations/N|
00000080  6f 74 4e 75 6c 6c 3b 01  00 04 61 72 67 73 08 00  |otNull;...args..|
00000090  08 01 00 1e 6b 6f 74 6c  69 6e 2f 6a 76 6d 2f 69  |....kotlin/jvm/i|
000000a0  6e 74 65 72 6e 61 6c 2f  49 6e 74 72 69 6e 73 69  |nternal/Intrinsi|
000000b0  63 73 07 00 0a 01 00 17  63 68 65 63 6b 50 61 72  |cs......checkPar|
000000c0  61 6d 65 74 65 72 49 73  4e 6f 74 4e 75 6c 6c 01  |ameterIsNotNull.|
000000d0  00 27 28 4c 6a 61 76 61  2f 6c 61 6e 67 2f 4f 62  |.'(Ljava/lang/Ob|
000000e0  6a 65 63 74 3b 4c 6a 61  76 61 2f 6c 61 6e 67 2f  |ject;Ljava/lang/|
000000f0  53 74 72 69 6e 67 3b 29  56 0c 00 0c 00 0d 0a 00  |String;)V.......|
00000100  0b 00 0e 01 00 0c 48 65  6c 6c 6f 20 44 65 76 6f  |......Hello Devo|
00000110  78 78 08 00 10 01 00 10  6a 61 76 61 2f 6c 61 6e  |xx......java/lan|
00000120  67 2f 53 79 73 74 65 6d  07 00 12 01 00 03 6f 75  |g/System......ou|
00000130  74 01 00 15 4c 6a 61 76  61 2f 69 6f 2f 50 72 69  |t...Ljava/io/Pri|
00000140  6e 74 53 74 72 65 61 6d  3b 0c 00 14 00 15 09 00  |ntStream;.......|
00000150  13 00 16 01 00 13 6a 61  76 61 2f 69 6f 2f 50 72  |......java/io/Pr|
00000160  69 6e 74 53 74 72 65 61  6d 07 00 18 01 00 07 70  |intStream......p|
00000170  72 69 6e 74 6c 6e 01 00  15 28 4c 6a 61 76 61 2f  |rintln...(Ljava/|
00000180  6c 61 6e 67 2f 4f 62 6a  65 63 74 3b 29 56 0c 00  |lang/Object;)V..|
00000190  1a 00 1b 0a 00 19 00 1c  01 00 13 5b 4c 6a 61 76  |...........[Ljav|
000001a0  61 2f 6c 61 6e 67 2f 53  74 72 69 6e 67 3b 01 00  |a/lang/String;..|
```

Java ByteCode


```
Compiled from "HelloWorld.kt"
public final class _00_helloworld.HelloWorldKt {
  public static final void main(java.lang.String[]);
    Code:
       0: aload_0
       1: ldc           #9                  // String args
       3: invokestatic  #15                 // Method kotlin/jvm/internal/Intr
       6: ldc           #17                 // String Hello Devoxx
       8: astore_1
       9: getstatic     #23                 // Field java/lang/System.out:Ljav
      12: aload_1
      13: invokevirtual #29                 // Method java/io/PrintStream.prin
      16: return
}
```

HelloWorld-java


```
package _00_helloworld;

import kotlin.Metadata;
import kotlin.jvm.internal.Intrinsics;
import org.jetbrains.annotations.NotNull;

@Metadata(
   mv = {1, 1, 9},
   bv = {1, 0, 2},
   k = 2,
   d1 = {"\u0000\u0012\n\u0000\n\u0002\u0010\u0002\n\u0000\
   d2 = {"main", "", "args", "", "", "([Ljava/lang/String;)
)
public final class HelloWorldKt {
   public static final void main(@NotNull String[] args) {
      Intrinsics.checkParameterIsNotNull(args, "args");
      String var1 = "Hello Devoxx";
      System.out.println(var1);
```

Bilan HelloWorld.kt

- Kotlin ajoute des contrôles
- du coup on a besoin de JAR en plus

jar	taille
kotlin-runtime.jar	921K
kotlin-reflect.jar	2.5M

- Performances ?

Performance ?

- https://github.com/JetBrains/kotlin-benchmarks
- Kotlin Hidden Costs Benchmark
  - ⚠️ kotlin 1.3
  - Part 1
  - Part 2
  - Part 3
  - Kotlin Hidden Costs - Benchmarks
  - MaJ version kotlin, JMH

Performance HelloWorld.kt

> Ne croyez pas les benchmarks, faites les vous-même !

https://github.com/MonkeyPatchIo/kotlin-perf

```
# VM version: JDK 9, VM 9+181
# VM invoker: /Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/Home/bin/java
# VM options: <none>
# Warmup: 20 iterations, 1 s each
# Measurement: 20 iterations, 1 s each
# Timeout: 10 min per iteration
# Threads: 1 thread, will synchronize iterations
# Benchmark mode: Throughput, ops/time
# Benchmark: io.monkeypatch.talks.MyBenchmark.testJava
```

Benchmark	Mode	Cnt	Score	Error	Units
MyBenchmark.testJava	thrpt	200	66490.271	± 879.996	ops/s

Les bases

val/var, Number, String

null-safety

Fonction

function

Lambda

Lambda

Class

Class

Types

Hierarchie des types

Extensions de fonction

Extension

Structure

Structures

Pause

ByteCode Android

Android ByteCode

Collection

Collection

Delegate

Delegate

Plus sur les fonctions

Plus sur les fonctions

Serialization

serialisation

Coroutines

Coroutines

DSL

DSL

Conclusion

Kotlin vs Java

Liens