SOUNDCLOUD

Just In Time

Running Scala on the JVM

Joe Kearney – March 2016

Just In Time

Running Scala on the JVM

1. **What is compilation** on the JVM?

   intro, types and classfiles

2. **Bytecode** – compiling for the JVM

   comparing Scala and Java representations

3. **Assembler** – compiling for the CPU

   benchmarking with JMH

1. What is compilation?

intro to compilation on the JVM, types and classfiles

What is compilation?

xkcd 303

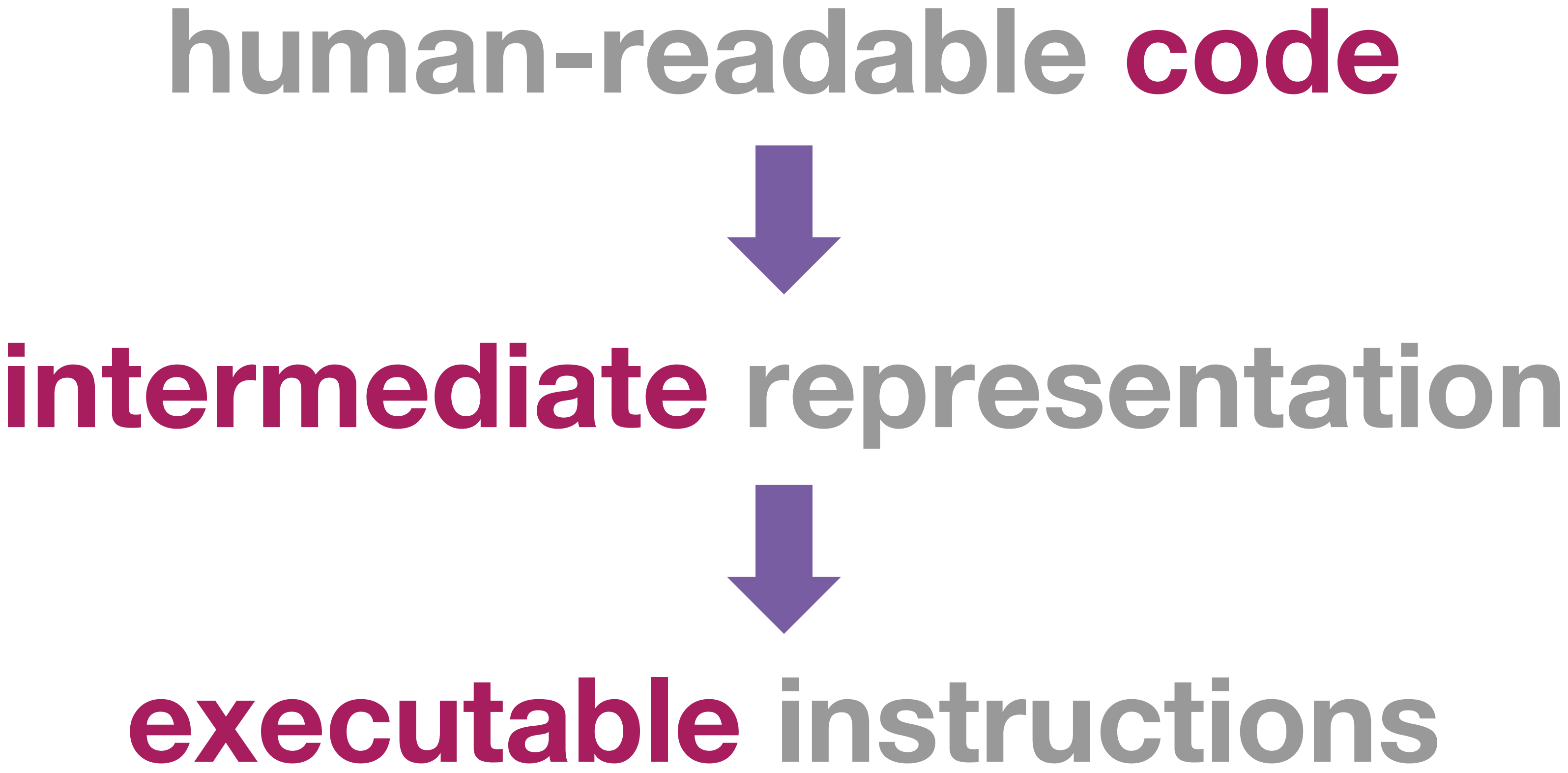
What is compilation?

human-readable **code**

↓

executable instructions

What is compilation?



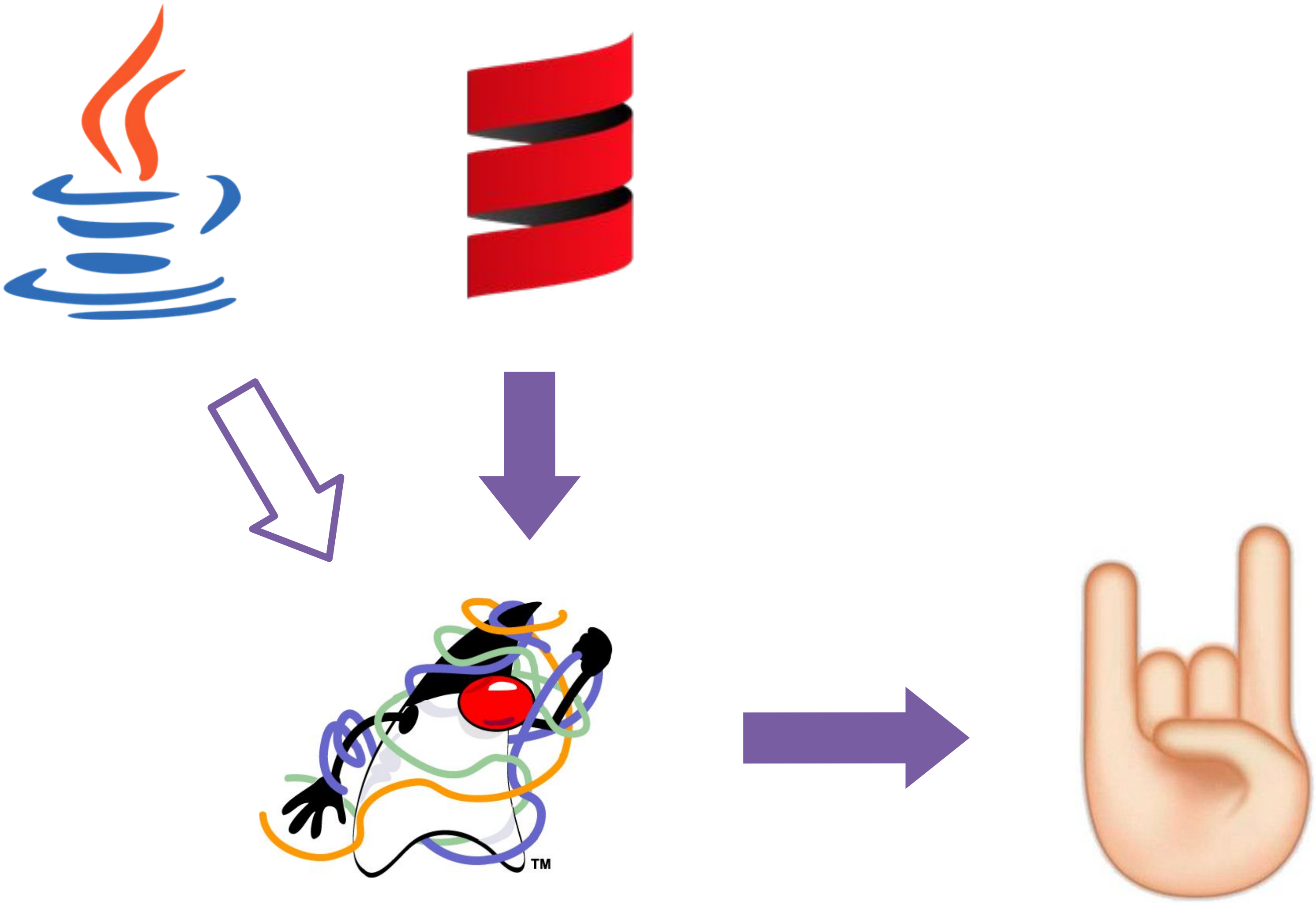
human-readable **code**

↓

intermediate representation

↓

executable instructions


TM

What is compilation?

Why not compile straight to machine code?

`scalac` compiler

(actually any JVM language compiler)

```
String  =>

                        Seq[ByteCode]
```

`scalac` compiler

(actually any JVM language compiler)

```
Set[String] =>
                          Set[Seq[ByteCode]]
```

scalac compiler

(actually any JVM language compiler)

```
Set[String] =>

Either[CompileError, Set[Seq[ByteCode]]]
```

Compilation in the JVM at runtime

```
Seq[ByteCode] => Seq[Instructions]
```

The JVM is “just” this function

Multiple JIT compilers

```
Seq[ByteCode] => Seq[Instructions]
```

- interpreter
- client C1
- server C2
- tiered

Types and classfiles

Types and classfiles

Kinds

(see the language specs and JVM spec for the gory details)

How many kinds in Scala?

3ish

```
class
trait
object
```

Scala language spec §5

How many kinds in Java?

2_{ish}

```
class
interface
```

JLS §4.3

How many kinds in the JVM?

2_{ish}

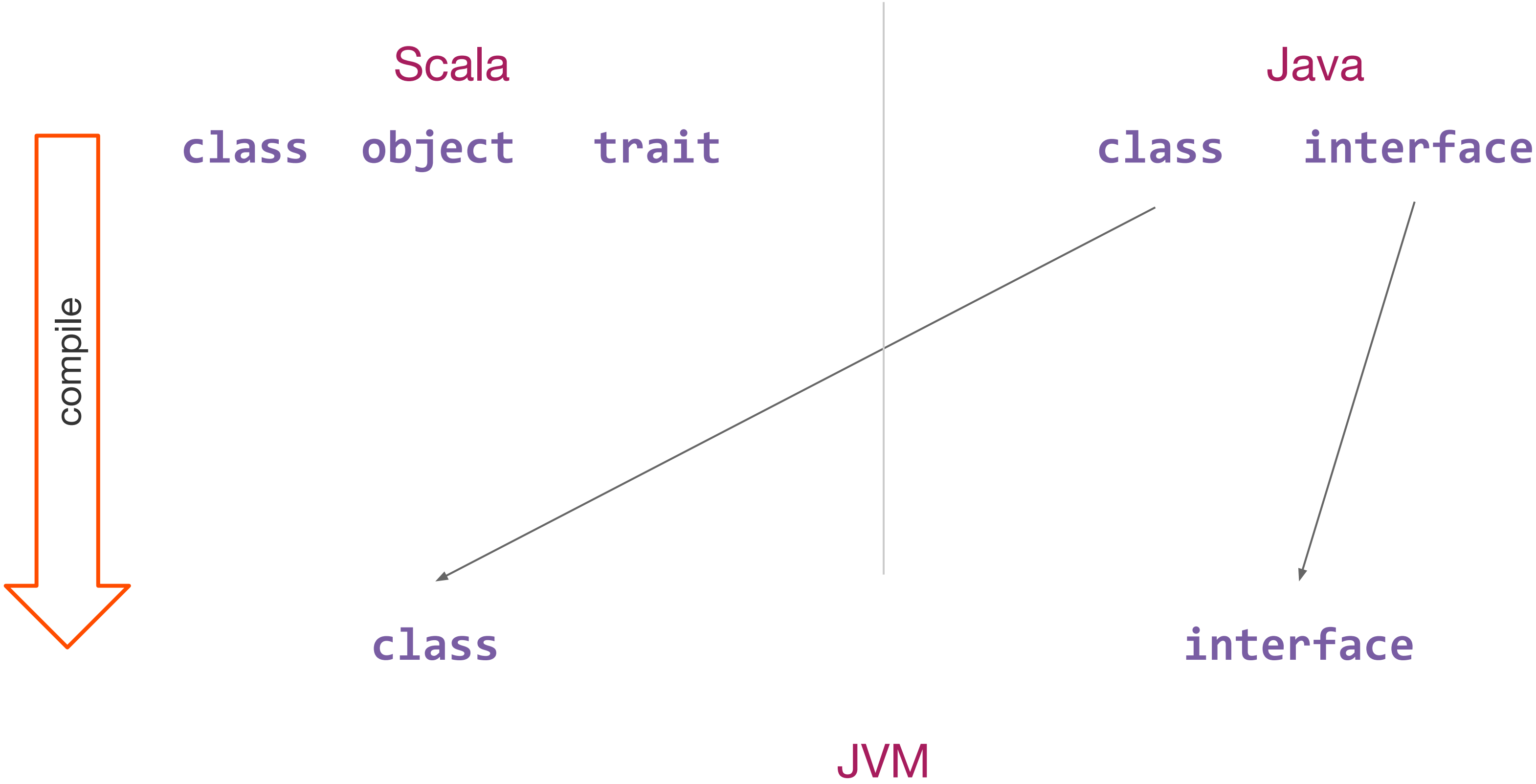
```
class
interface
```

This is the unit of class-loading

JVM spec §2.4

Scala/Java equivalences

Types



Scala

class object trait

Java

class interface

compile

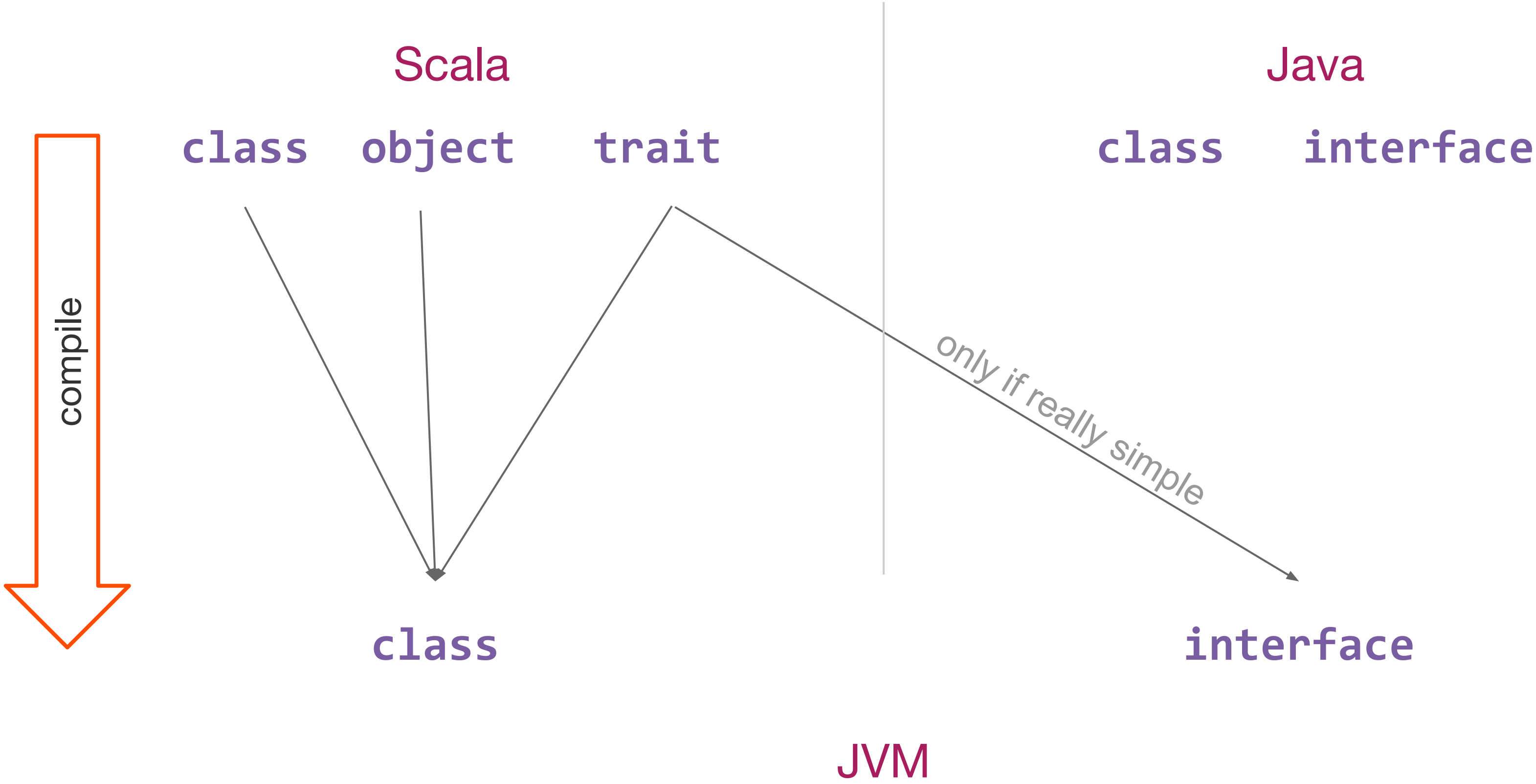
class

interface

JVM

Scala/Java equivalences

Types



Scala

class object trait

Java

class interface

compile

only if really simple

class

interface

JVM

Scala/Java equivalences

Methods

Scala	Java
`def` on a `class` or `trait`	method on a `class`
`def` on an `object`	`static` method on a `class`

Scala/Java equivalences

Methods

Scala	Java
`def` on a `class` or `trait`	method on a `class`
`def` on an `object`	`static` method on a `class`
`val` on a `class` or `trait`	field on a `class`
`val` on an `object`	`static final` field on a `class`

Scala/Java equivalences

Example

Scala

```scala
object ScalaConstants {
  val ichBinEinConstant = "some string"

  def ichBinEinUtilityFunction(param: Int) =
    param.toString
}
```

Java

```java
public class JavaConstants {
  public static final String jeSuisUnConstant
         = "some string";
  public static String jeSuisUnUtilityFunction(
         int param) {
    return String.valueOf(param);
  }
}
```

Just In Time

Running Scala on the JVM

1. **What is compilation** on the JVM?

   intro, types and classfiles

2. **Bytecode** – compiling for the JVM

   comparing Scala and Java representations

3. **Assembler** – compiling for the CPU

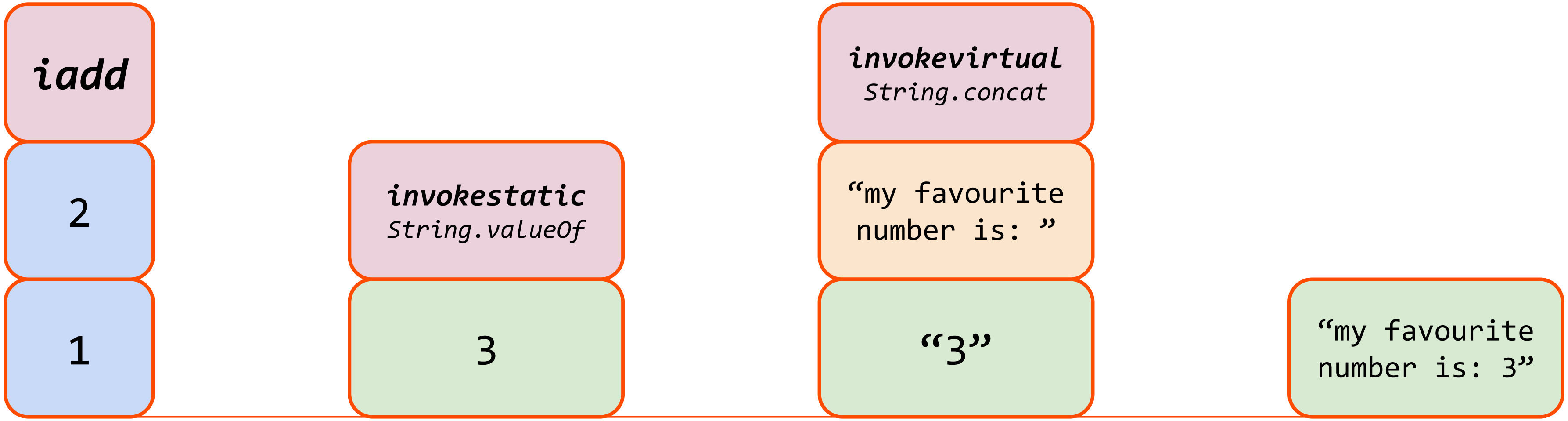
   benchmarking with JMH

2. What is bytecode?

compiling for the JVM

Bytecode is a stack language

What’s my favourite number?



“my favourite number is: ” + (1 + 2)

...and the bytecode?

Java ships with a disassembler

```
javap <class-name>

javap -p -c <class-name>
```

Some code examples

1. `ScalaConstants` class members
2. Bytecode of a `def` on an `object`
3. Singleton initialisation
4. More function calls

1. What does bytecode look like?

brackets = method, no brackets = field

Scala

```
object ScalaConstants {
  val ichBinEinConstant = "some string"
  def ichBinEinUtilityFunction(param: Int) = param.toString
}
```

bytecode

```
public final class ScalaConstants$ {
  public static final ScalaConstants$ MODULE$;
  private final java.lang.String ichBinEinConstant;
  public static {};
  public java.lang.String ichBinEinConstant();
  public java.lang.String ichBinEinUtilityFunction(int);
  private ScalaConstants$();
}

public final class ScalaConstants {
  public static java.lang.String ichBinEinUtilityFunction(int);
  public static java.lang.String ichBinEinConstant();
}
```

2. What does bytecode look like?

a def on an object

Scala

```
object ScalaConstants {
  val ichBinEinConstant = "some string"
  def ichBinEinUtilityFunction(param: Int) = param.toString
}
```

bytecode

```
public final class ScalaConstants {
  public static java.lang.String ichBinEinUtilityFunction(int);
    Code:
       0: getstatic     #16  // Field ScalaConstants$.MODULE$:LScalaConstants$;
       3: iload_0
       4: invokevirtual #18  // Method ScalaConstants$.ichBinEinUtilityFunction:...
       7: areturn







}
```

```
ScalaConstants.ichBinEinUtilityFunction(3)
```

3. What does bytecode look like?

more delegating

Scala

```
object ScalaConstants {
  val ichBinEinConstant = "some string"
  def ichBinEinUtilityFunction(param: Int) = param.toString
}
```

bytecode

```
public final class ScalaConstants$ {
  public java.lang.String ichBinEinUtilityFunction(int);
    Code:
       0: iload_1
       1: invokestatic  #26 // Method scala/runtime/BoxesRunTime.boxToInteger:(I)Ljava/lang/Integer;
       4: invokevirtual #29 // Method java/lang/Object.toString:()Ljava/lang/String;
       7: areturn

}
```

```
ScalaConstants.ichBinEinUtilityFunction(3)
```

4. What does bytecode look like?

singleton initialisation

Scala

```
object ScalaConstants {
  val ichBinEinConstant = "some string"
  def ichBinEinUtilityFunction(param: Int) = param.toString
}
```

bytecode

```
public final class ScalaConstants$ {
  public static final ScalaConstants$ MODULE$;

  private ScalaConstants$();
    Code:
       0: aload_0
       1: invokespecial #32 // Method java/lang/Object."<init>":()V
       4: aload_0
       5: putstatic     #34 // Field MODULE$:LScalaConstants$;
       8: aload_0
       9: ldc           #36 // String “some string”
      11: putfield      #17 // Field ichBinEinConstant:Ljava/lang/String;
      14: return
}
```

Paws – questions?

Further down the rabbit hole: JIT

What is compilation?

human-readable **code**

↓

intermediate representation

↓

executable instructions

Just In Time

Running Scala on the JVM

1. **What is compilation** on the JVM?

   intro, types and classfiles

2. **Bytecode** – compiling for the JVM

   comparing Scala and Java representations

3. **Assembler** – compiling for the CPU

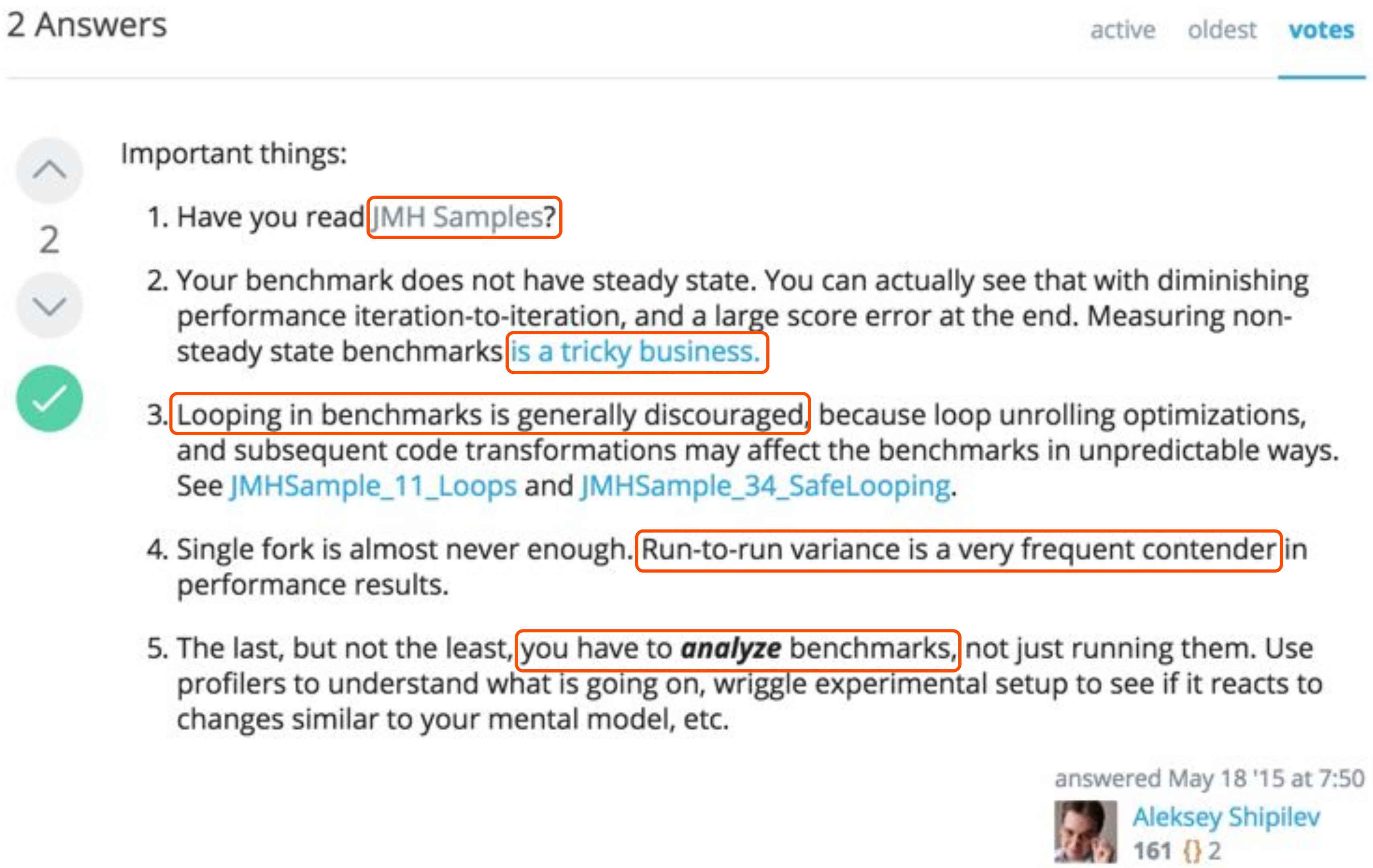
   benchmarking with JMH

3. Benchmarking

Benchmarking is hard

because compiler optimisations

Understanding why is instructive



2 Answers

Important things:

1. Have you read JMH Samples?
2. Your benchmark does not have steady state. You can actually see that with diminishing performance iteration-to-iteration, and a large score error at the end. Measuring non-steady state benchmarks is a tricky business.
3. Looping in benchmarks is generally discouraged, because loop unrolling optimizations, and subsequent code transformations may affect the benchmarks in unpredictable ways. See JMHSample_11_Loops and JMHSample_34_SafeLooping.
4. Single fork is almost never enough. Run-to-run variance is a very frequent contender in performance results.
5. The last, but not the least, you have to ***analyze*** benchmarks, not just running them. Use profilers to understand what is going on, wriggle experimental setup to see if it reacts to changes similar to your mental model, etc.

answered May 18 '15 at 7:50

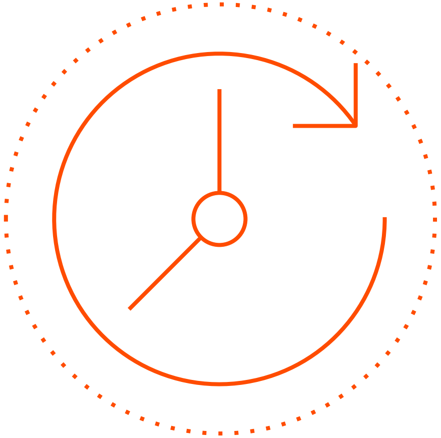
Aleksey Shipilev

161 {} 2

Java Microbenchmarking Harness

What does a benchmark look like in JMH?

```
class BenchmarkHelloWorld {
  @Benchmark
  def wellHelloThere() {
    // code here is timed
  }
}
```

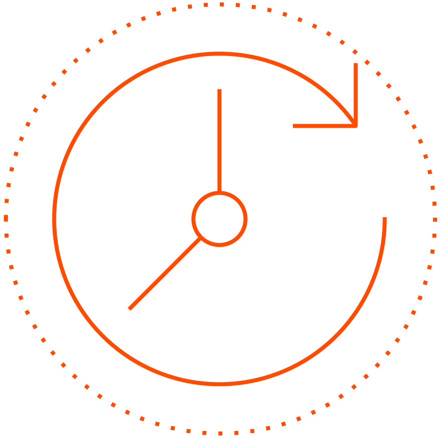


Java Microbenchmarking Harness

What does JMH do?

Code generated that:

- times invocation of your test code
- gets averages from many runs
- handles many common pitfalls

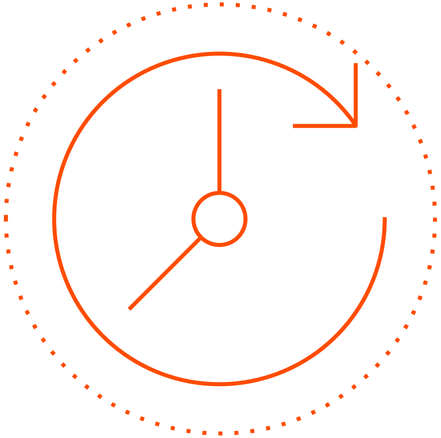


Java Microbenchmarking Harness

What is it for?

Comparing implementation choices

Not for absolute timing values



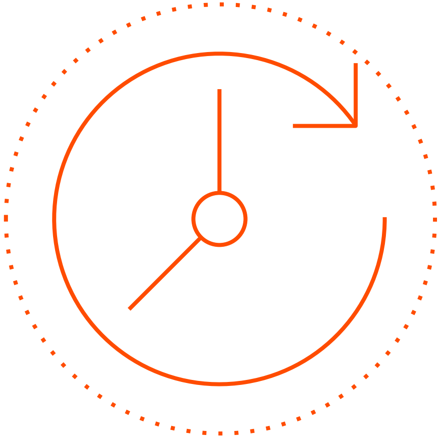
Java Microbenchmarking Harness

How do you run it?

- Using sbt it’s straightforward

```
sbt jmh:run <BenchmarkClass>
```

- Better as a separate project



Benchmarks

1. compiled vs interpreted
2. inlining
3. dead code elimination
4. parameter specialisation
5. class hierarchy analysis, polymorphism

1. Is compiling worth it?

What's the cost of turning off JIT compilation completely?

JMH allows control over which methods are compiled and which aren't

```
def target_default: Unit = ()
@CompilerControl(CompilerControl.Mode.EXCLUDE)
def target_interpreted: Unit = ()
```

```
@Benchmark
def compiled: Unit = target_default
@Benchmark
def interpreted: Unit = target_interpreted
```

nanoseconds per invocation

2.3

89.5

x40

2. Inlining

What’s the cost of method invocation?

JIT can inline methods, including their code at the call site

```
@CompilerControl(CompilerControl.Mode.DONT_INLINE)
def target_dontInline: Unit = ()
@CompilerControl(CompilerControl.Mode.INLINE)
def target_inline: Unit = ()

@Benchmark                                                  nanoseconds per invocation
def dontInlineBench: Unit = target_dontInline               2.3
@Benchmark
def inlineBench: Unit = target_inline                       0.3
```

3. Dead code elimination

What’s the cost of unnecessary code?

JIT can delete unnecessary code.

How expensive is the `Math.log` function?

```
var x = Math.PI                                      nanoseconds per invocation
@Benchmark
def measureLogWrong: Unit = { Math.log(x) }           0.3
@Benchmark
def measureLogRight: Double = Math.log(x)            21.8
```

4. Parameter specialisation

What's the cost of not knowing your input?

JIT can optimise for
provable parameter values

```
var x = Math.PI                                                nanoseconds per invocation
@Benchmark
def measureLogConst: Double = Math.log(Math.PI)                  3.1
@Benchmark
def measureLogParam: Double = Math.log(x)                        21.8
```

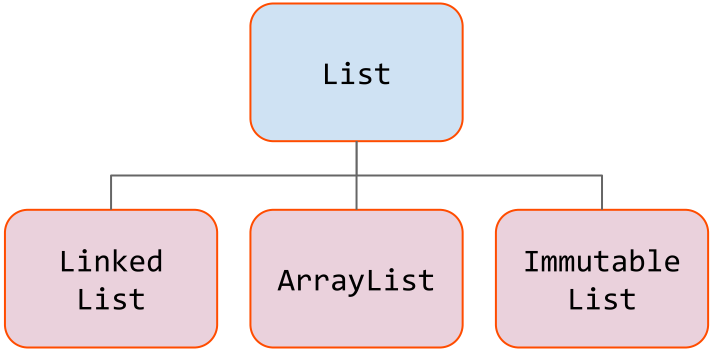
5. Class Hierarchy Analysis

How do you choose which method to call?

5. Class Hierarchy Analysis

How do you choose which method to call?

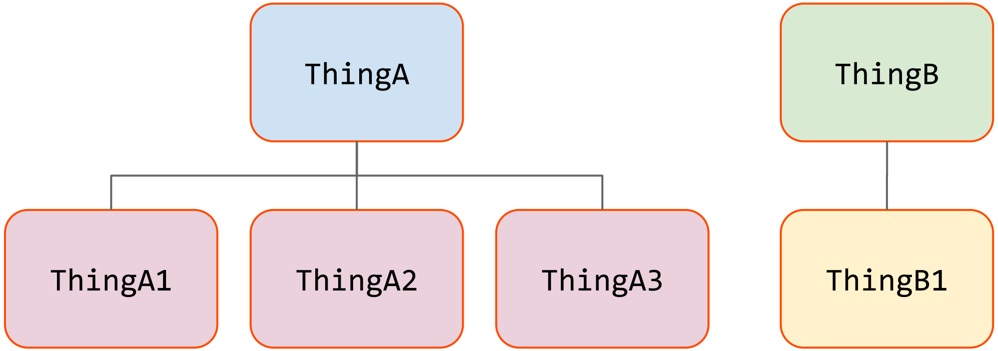
JIT can optimise away polymorphic method calls if it can prove the target class



```
List<String> list = ...
list.size();
```

5. Class Hierarchy Analysis

What classes have been loaded?



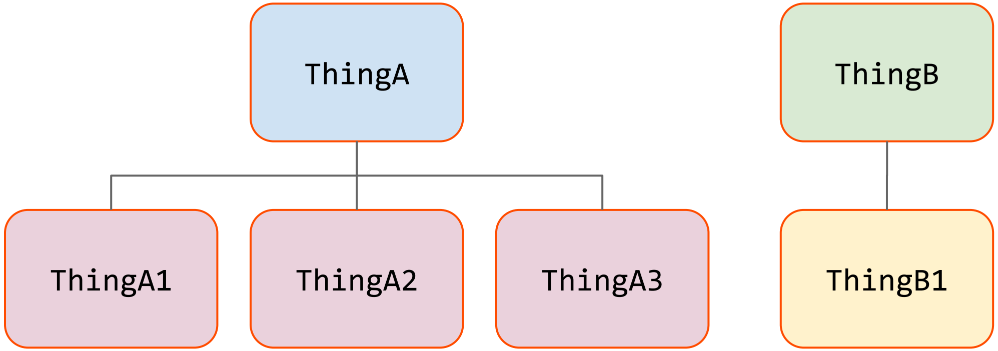
JIT can optimise away polymorphic method calls if it can prove the target class

… if there is only one target

```
trait ThingA {
  def get: Int
}
class ThingA1(value: Int) extends ThingA {
  def get = value
}
// etc
```

5. Class Hierarchy Analysis

What classes have been loaded?



JIT can optimise away polymorphic method calls if it can prove the target class

… if there is only one target

```
val a1: ThingA = new ThingA1(1)
val a2: ThingA = new ThingA2(2)
val a3: ThingA = new ThingA3(3)
val a4: ThingA = new ThingA4(4)
```

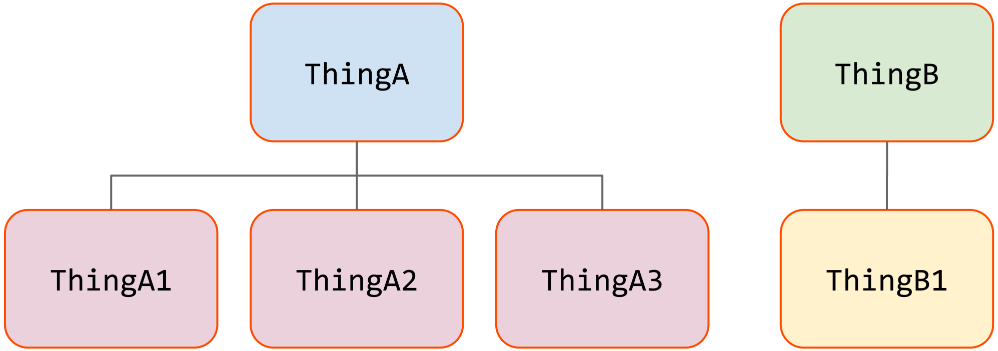
megamorphic

```
val b1: ThingB = new ThingB1(1)
val b2: ThingB = new ThingB1(1)
```

monomorphic

5. Class Hierarchy Analysis

What classes have been loaded?



ThingA

ThingA1

ThingA2

ThingA3

ThingB

ThingB1

JIT can optimise away polymorphic method calls if it can prove the target class

… if there is only one target

nanos per invocation

```
@Benchmark                                                     5.3
def multipleClasses = a1.get + a2.get + a3.get + a4.get
@Benchmark
def singleClass     = b1.get + b2.get + b3.get + b4.get        3.6
```

Summary of effects benchmarked

1. compiled vs interpreted
2. inlining
3. dead code elimination
4. parameter specialisation
5. class hierarchy analysis, polymorphism

JMH `-prof perfasm`

but why believe me?

You can see the generated assembler

JMH `-prof perfasm`

`hsdis`, `perf`, linux

… but it can be tricky to make to work

Inlining really works

```
@CompilerControl(CompilerControl.Mode.DONT_INLINE)
def target_dontInline: Unit = ()
@CompilerControl(CompilerControl.Mode.INLINE)
def target_inline: Unit = ()

@Benchmark
def dontInlineBench: Unit = target_dontInline
@Benchmark
def inlineBench: Unit = target_inline
```

Snippet from `inline_avgt_jmhStub`:

```
result.startTime = System.nanoTime();
do {
  l_deadcodebench0_0.inlineBench();
  operations++;
} while(!control.isDone);
result.stopTime = System.nanoTime();
```

Note:

- start timer
- loops with method invocation, increment counter
- control.isDone
- stop timer

Inlining really works

```
cycles   instrs      instr address        instruction                 source instruction
                     0x00007f24446351b7: nopw   0x0(%rax,%rax,1)     ;*invokevirtual inlineBench
23.92%   21.41%  ↗   0x00007f24446351c0: movzbl 0x94(%r11),%r10d     ;*getfield isDone
          0.05%  │   0x00007f24446351c8: add    $0x1,%r14
                 │
          0.05%  │   0x00007f24446351cc: test   %eax,0xa627e2e(%rip) ;*ifeq
                 │                                                   ;   {poll}
21.38%   23.76%  │   0x00007f24446351d2: test   %r10d,%r10d
                 ╰   0x00007f24446351d5: je     0x00007f24446351c0   ;*ifeq
                     0x00007f24446351d7: mov    $0x7f244d7ee2b0,%r10
                     0x00007f24446351e1: callq  *%r10                ;*invokestatic nanoTime
                     0x00007f24446351e4: mov    %rax,0x30(%r13)      ;*putfield stopTime
```

Note:

- start timer
- loops with method invocation, increment counter
- control.isDone
- stop timer

```
result.startTime = System.nanoTime();
do {
  l_deadcodebench0_0.inlineBench();
  operations++;
} while(!control.isDone);
result.stopTime = System.nanoTime();
```

`0.3` nanoseconds

Coincidence?

Can anyone guess my CPU clock speed?

How did all of the instructions execute?

- inlined benchmark executed one loop per clock cycle
- counter increment, `isDone` read

How did all of the instructions execute?

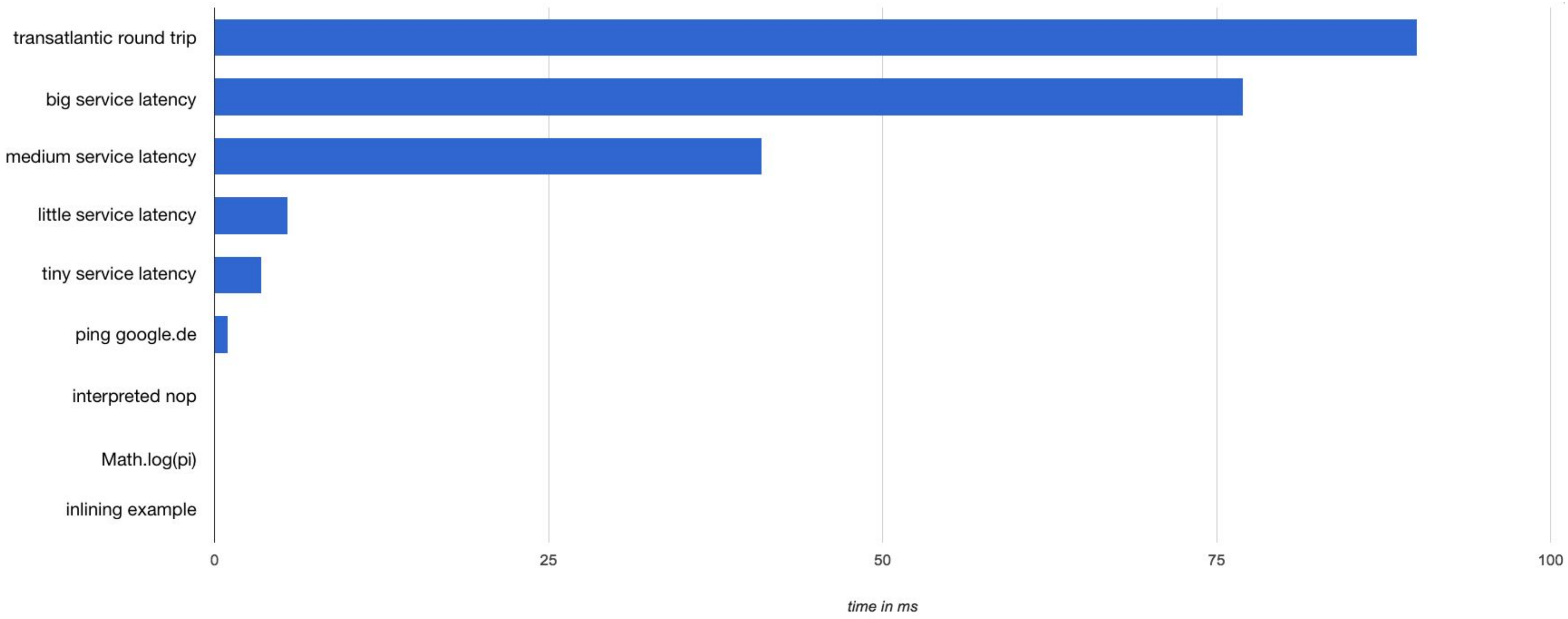
- Difficult to prove anything from here
- No instrumentation on the CPU for that detail

How did all of the instructions execute?

Could include things like

- branch prediction
- CPU pipelining
- instruction-parallelism in CPU pipeline

Perspective



When should I microbenchmark?

1. Architectural changes
2. Algorithmic improvements
3. Microbenchmarking

What have we learned?

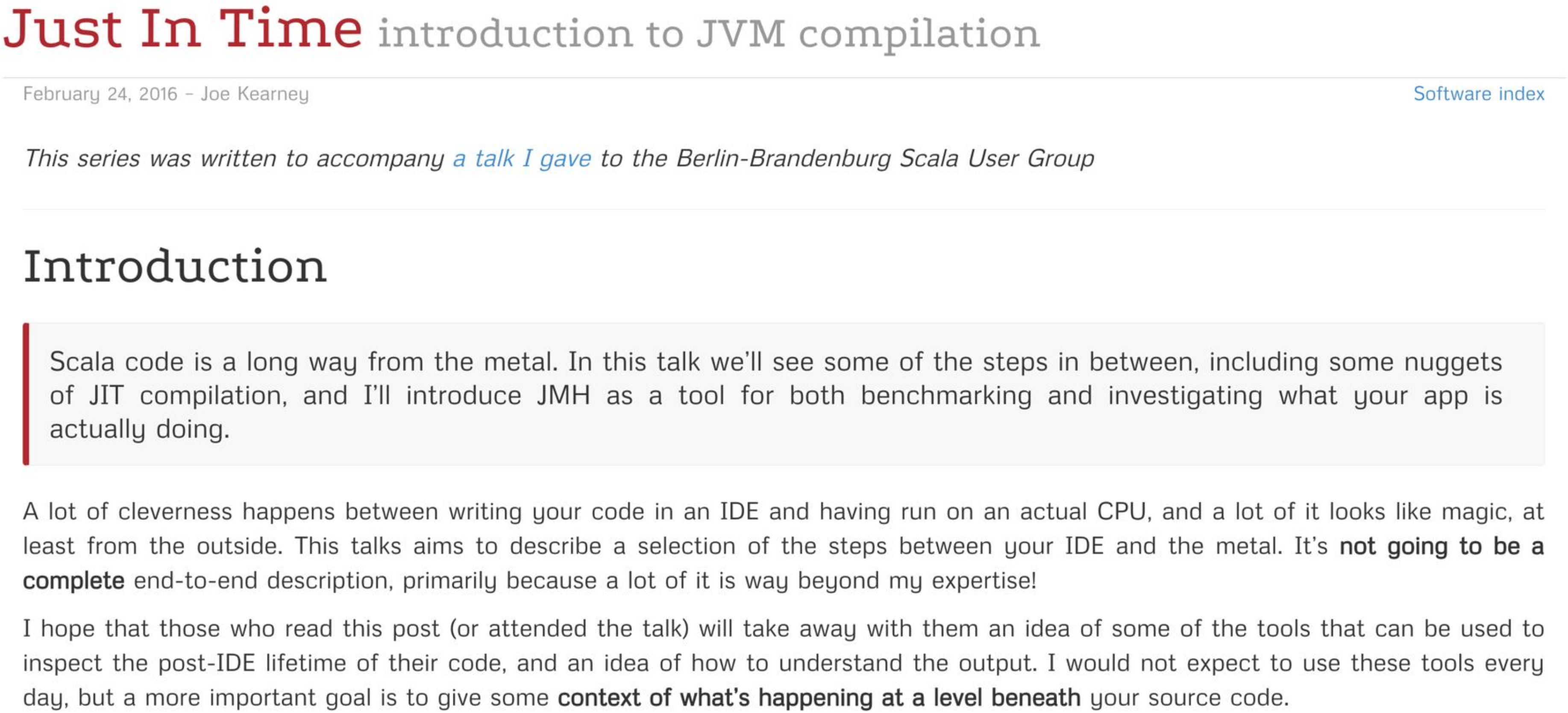
- Mappings between Java/Scala types
- How bytecode works and how to see it
- Intro to JMH and microbenchmarking
- How to see generated machine code

What have we learned?

- Your code is a lie
- The compiler is sneaky
- Benchmarking is hard, because the compiler is sneaky
- There’s a lot going on, and it can look like magic

Blog version

joekearney.co.uk/just-in-time



Just In Time introduction to JVM compilation

February 24, 2016 - Joe Kearney

Software index

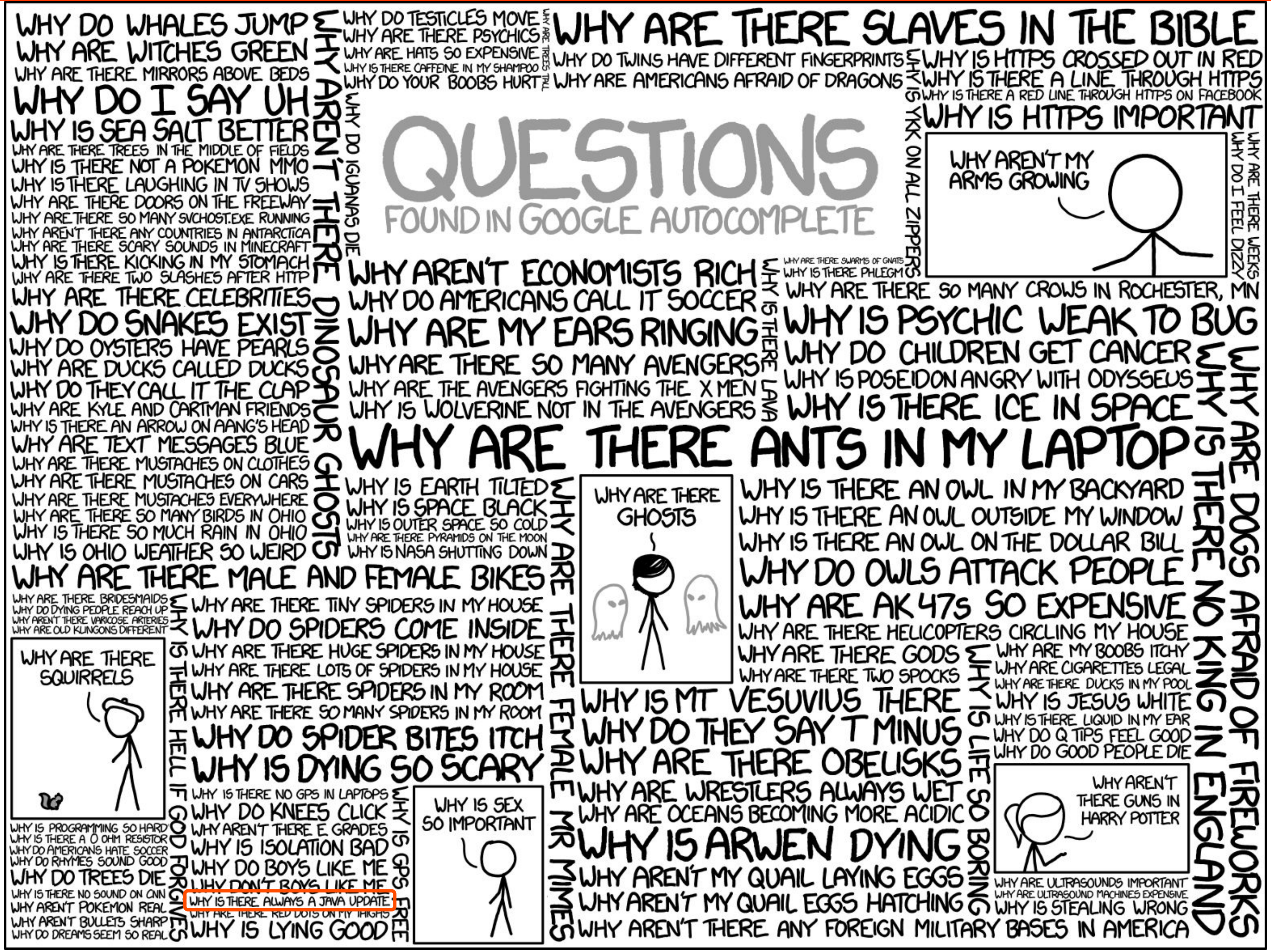
This series was written to accompany a talk I gave to the Berlin-Brandenburg Scala User Group

Introduction

> Scala code is a long way from the metal. In this talk we'll see some of the steps in between, including some nuggets of JIT compilation, and I'll introduce JMH as a tool for both benchmarking and investigating what your app is actually doing.

A lot of cleverness happens between writing your code in an IDE and having run on an actual CPU, and a lot of it looks like magic, at least from the outside. This talks aims to describe a selection of the steps between your IDE and the metal. It's **not going to be a complete** end-to-end description, primarily because a lot of it is way beyond my expertise!

I hope that those who read this post (or attended the talk) will take away with them an idea of some of the tools that can be used to inspect the post-IDE lifetime of their code, and an idea of how to understand the output. I would not expect to use these tools every day, but a more important goal is to give some **context of what's happening at a level beneath** your source code.


QUESTIONS
FOUND IN GOOGLE AUTOCOMPLETE
WHY DO WHALES JUMP
WHY ARE WITCHES GREEN
WHY ARE THERE MIRRORS ABOVE BEDS
WHY DO I SAY UH
WHY IS SEA SALT BETTER
WHY ARE THERE TREES IN THE MIDDLE OF FIELDS
WHY IS THERE NOT A POKEMON MMO
WHY IS THERE LAUGHING IN TV SHOWS
WHY ARE THERE DOORS ON THE FREEWAY
WHY ARE THERE SO MANY SVCHOST.EXE RUNNING
WHY AREN'T THERE ANY COUNTRIES IN ANTARCTICA
WHY ARE THERE SCARY SOUNDS IN MINECRAFT
WHY IS THERE KICKING IN MY STOMACH
WHY ARE THERE TWO SLASHES AFTER HTTP
WHY ARE THERE CELEBRITIES
WHY DO SNAKES EXIST
WHY DO OYSTERS HAVE PEARLS
WHY ARE DUCKS CALLED DUCKS
WHY DO THEY CALL IT THE CLAP
WHY ARE KYLE AND CARTMAN FRIENDS
WHY IS THERE AN ARROW ON AANG'S HEAD
WHY ARE TEXT MESSAGES BLUE
WHY ARE THERE MUSTACHES ON CLOTHES
WHY ARE THERE MUSTACHES ON CARS
WHY ARE THERE MUSTACHES EVERYWHERE
WHY ARE THERE SO MANY BIRDS IN OHIO
WHY IS THERE SO MUCH RAIN IN OHIO
WHY IS OHIO WEATHER SO WEIRD
WHY AREN'T THERE DINOSAUR GHOSTS
WHY DO IGUANAS DIE
WHY DO TESTICLES MOVE
WHY ARE THERE PSYCHICS
WHY ARE HATS SO EXPENSIVE
WHY IS THERE CAFFEINE IN MY SHAMPOO
WHY DO YOUR BOOBS HURT
WHY ARE TREES TALL
WHY ARE THERE SLAVES IN THE BIBLE
WHY DO TWINS HAVE DIFFERENT FINGERPRINTS
WHY ARE AMERICANS AFRAID OF DRAGONS
WHY IS YKK ON ALL ZIPPERS
WHY IS HTTPS CROSSED OUT IN RED
WHY IS THERE A LINE THROUGH HTTPS
WHY IS THERE A RED LINE THROUGH HTTPS ON FACEBOOK
WHY IS HTTPS IMPORTANT
WHY AREN'T MY ARMS GROWING
WHY ARE THERE WEEKS
WHY DO I FEEL DIZZY
WHY AREN'T ECONOMISTS RICH
WHY DO AMERICANS CALL IT SOCCER
WHY ARE MY EARS RINGING
WHY ARE THERE SO MANY AVENGERS
WHY ARE THE AVENGERS FIGHTING THE X MEN
WHY IS WOLVERINE NOT IN THE AVENGERS
WHY IS THERE LAVA
WHY ARE THERE SWARMS OF GNATS
WHY IS THERE PHLEGM
WHY ARE THERE SO MANY CROWS IN ROCHESTER, MN
WHY IS PSYCHIC WEAK TO BUG
WHY DO CHILDREN GET CANCER
WHY IS POSEIDON ANGRY WITH ODYSSEUS
WHY IS THERE ICE IN SPACE
WHY ARE THERE ANTS IN MY LAPTOP
WHY IS THERE NO KING IN ENGLAND
WHY ARE DOGS AFRAID OF FIREWORKS
WHY IS EARTH TILTED
WHY IS SPACE BLACK
WHY IS OUTER SPACE SO COLD
WHY ARE THERE PYRAMIDS ON THE MOON
WHY IS NASA SHUTTING DOWN
WHY ARE THERE FEMALE MR MIMES
WHY ARE THERE GHOSTS
WHY IS THERE AN OWL IN MY BACKYARD
WHY IS THERE AN OWL OUTSIDE MY WINDOW
WHY IS THERE AN OWL ON THE DOLLAR BILL
WHY DO OWLS ATTACK PEOPLE
WHY ARE AK 47s SO EXPENSIVE
WHY ARE THERE HELICOPTERS CIRCLING MY HOUSE
WHY ARE THERE GODS
WHY ARE THERE TWO SPOCKS
WHY IS LIFE SO BORING
WHY ARE MY BOOBS ITCHY
WHY ARE CIGARETTES LEGAL
WHY ARE THERE DUCKS IN MY POOL
WHY IS JESUS WHITE
WHY IS THERE LIQUID IN MY EAR
WHY DO Q TIPS FEEL GOOD
WHY DO GOOD PEOPLE DIE
WHY ARE THERE MALE AND FEMALE BIKES
WHY ARE THERE BRIDESMAIDS
WHY DO DYING PEOPLE REACH UP
WHY AREN'T THERE VARICOSE ARTERIES
WHY ARE OLD KLINGONS DIFFERENT
WHY IS THERE HELL IF GOD FORGIVES
WHY ARE THERE SQUIRRELS
WHY IS PROGRAMMING SO HARD
WHY IS THERE A 0 OHM RESISTOR
WHY DO AMERICANS HATE SOCCER
WHY DO RHYMES SOUND GOOD
WHY DO TREES DIE
WHY IS THERE NO SOUND ON CNN
WHY AREN'T POKEMON REAL
WHY AREN'T BULLETS SHARP
WHY DO DREAMS SEEM SO REAL
WHY ARE THERE TINY SPIDERS IN MY HOUSE
WHY DO SPIDERS COME INSIDE
WHY ARE THERE HUGE SPIDERS IN MY HOUSE
WHY ARE THERE LOTS OF SPIDERS IN MY HOUSE
WHY ARE THERE SPIDERS IN MY ROOM
WHY ARE THERE SO MANY SPIDERS IN MY ROOM
WHY DO SPIDER BITES ITCH
WHY IS DYING SO SCARY
WHY IS THERE NO GPS IN LAPTOPS
WHY DO KNEES CLICK
WHY AREN'T THERE E GRADES
WHY IS ISOLATION BAD
WHY DO BOYS LIKE ME
WHY DON'T BOYS LIKE ME
WHY IS THERE ALWAYS A JAVA UPDATE
WHY ARE THERE RED DOTS ON MY THIGHS
WHY IS LYING GOOD
WHY IS GPS FREE
WHY IS SEX SO IMPORTANT
WHY IS MT VESUVIUS THERE
WHY DO THEY SAY T MINUS
WHY ARE THERE OBELISKS
WHY ARE WRESTLERS ALWAYS WET
WHY ARE OCEANS BECOMING MORE ACIDIC
WHY IS ARWEN DYING
WHY AREN'T MY QUAIL LAYING EGGS
WHY AREN'T MY QUAIL EGGS HATCHING
WHY AREN'T THERE ANY FOREIGN MILITARY BASES IN AMERICA
WHY AREN'T THERE GUNS IN HARRY POTTER
WHY ARE ULTRASOUNDS IMPORTANT
WHY ARE ULTRASOUND MACHINES EXPENSIVE
WHY IS STEALING WRONG

[xkcd 1256](https://xkcd.com/1256)

Resources

- Alexey Shipilev's blog – Nanotrusting Nanotime
- Slides available on Slideshare

- Blog version at joekearney.co.uk/just-in-time