

DEVNEXUS™



The Groovy Puzzlers

As usual - Traps, Pitfalls, and End Cases

Who we are

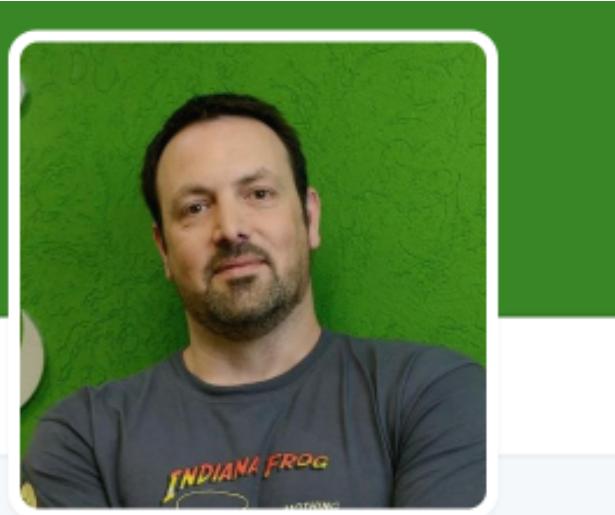
<https://github.com/yoav>



Yoav Landman
Founder and CTO at JFrog Ltd
Israel | Computer Software
Current JFrog Ltd.
Previous AlphaCSP, Attunity, Verve Inc.
Education RMIT



@_yoav_



Yoav Landman

JFrog co-founder and CTO. Artifactory and Bintray creator.

- 📍 The holy land
- 🔗 jfrog.com
- ⌚ Joined February 2009

Who we are?

@jbaruch

github.com/jbaruch



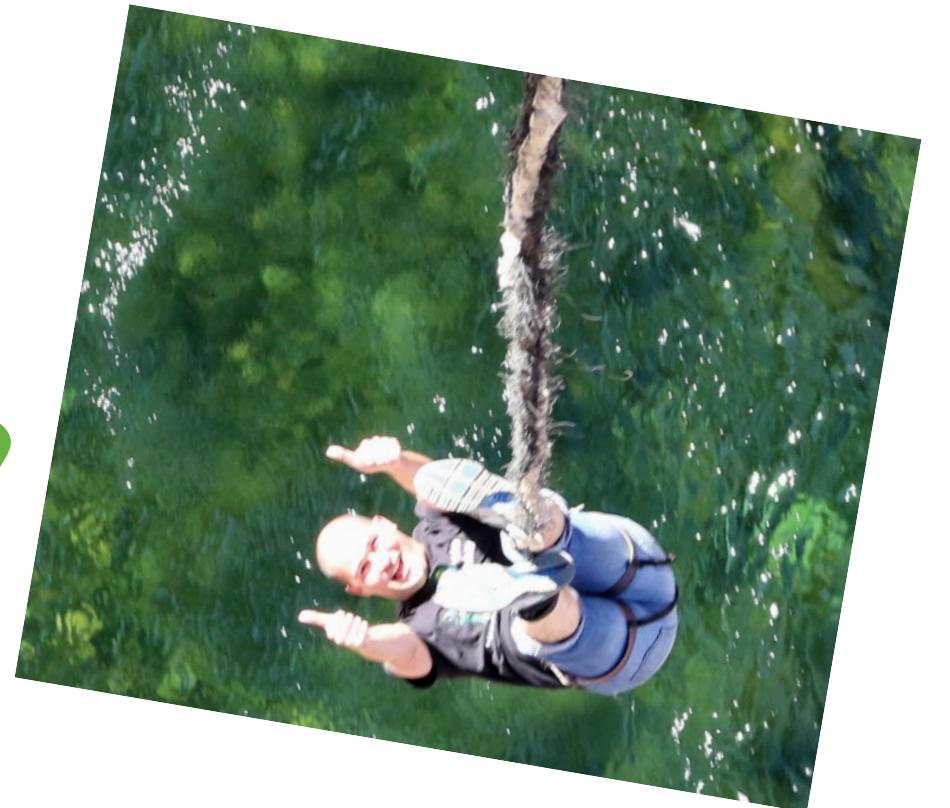
Baruch Sadogursky

J*, G* and Public Speaking Geek with JFrog FTW.
Israel | Computer Software

Current

Developer Advocate at JFrog Ltd

stackoverflow.com/users/402053/jbaruch



What Frog?



What Frog?



What Frog?



artifactory

What Frog?



artifactory

CLICK AND HACK



THE TYPING BROTHERS

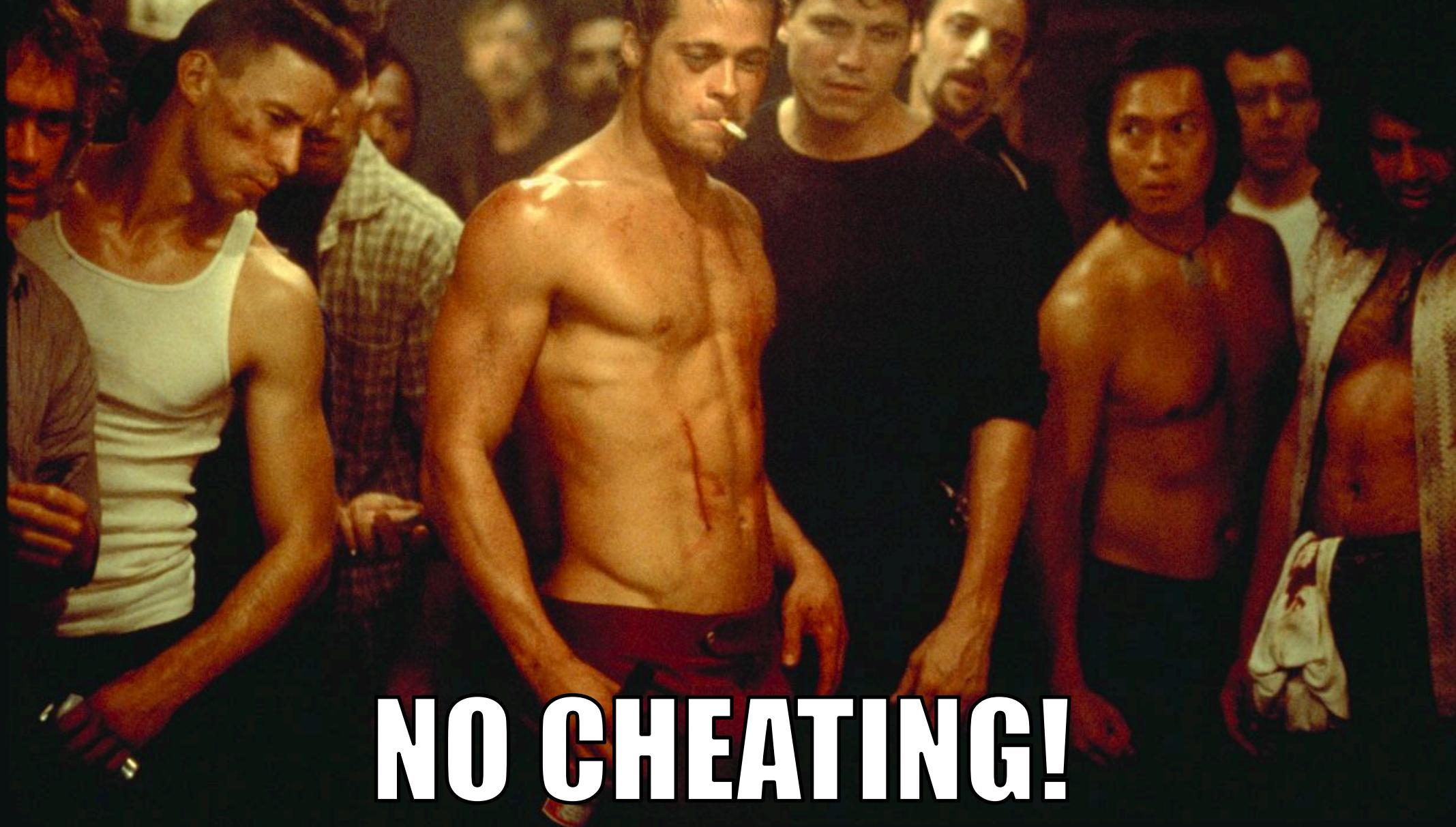
BTW...



1. Two entertaining guys on stage
2. Funny Puzzling questions
3. You think and vote
4. Lots of T-shirts flying in the air
5. Official twitter handle!
groovypuzzlers

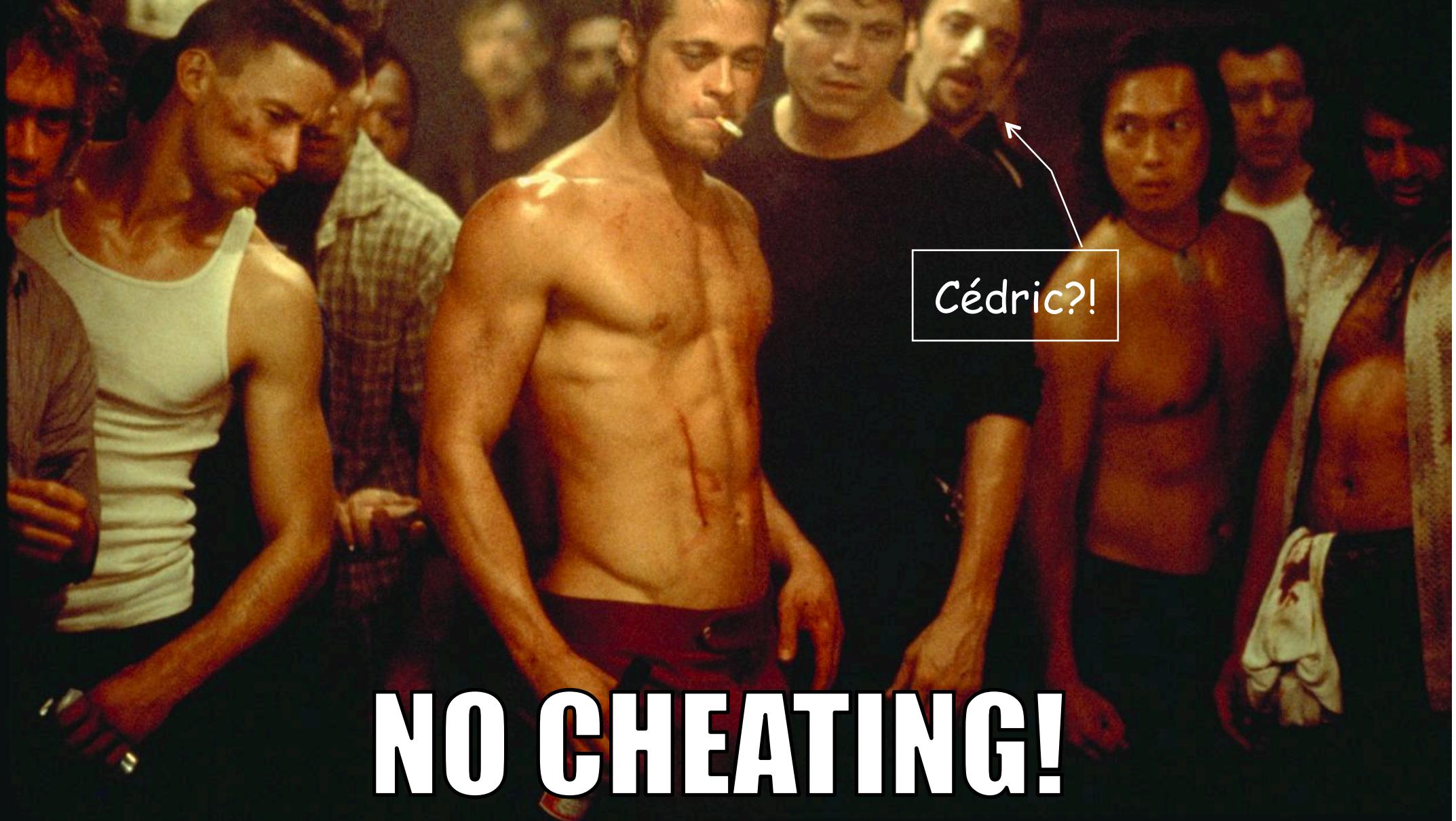
#

FIRST RULE OF THE PUZZLERS:

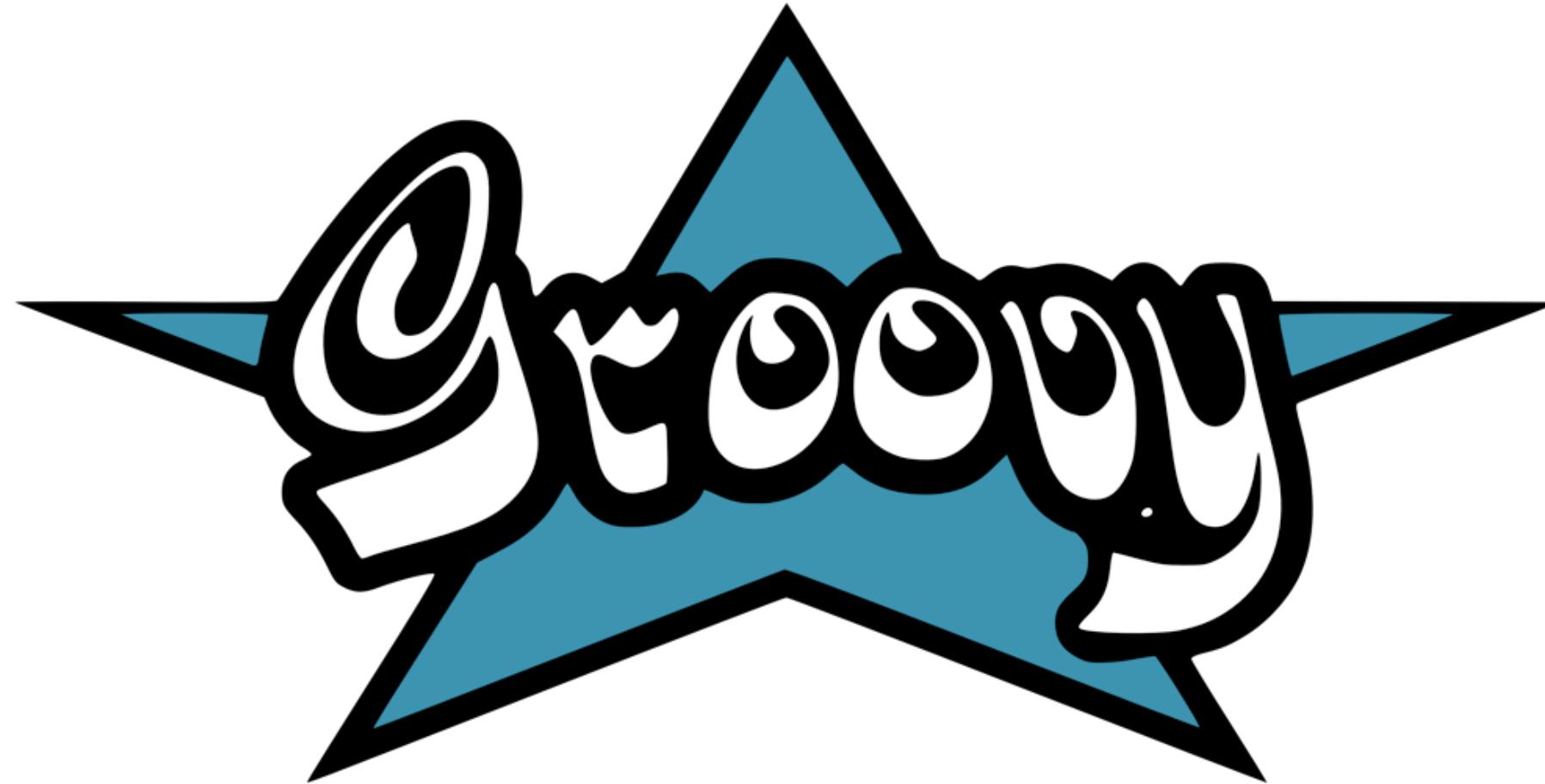


NO CHEATING!

FIRST RULE OF THE PUZZLERS:



Consider the puzzlers correct as of
2.4.1



CHALLENGE

ACCEPTED

Greach

the Groovy spanish conf

**MADRID - MARCH
28TH & 29TH**

```
class Conference {def name; def year}

def gr = new Conference(name: 'Greach', year: 2014)

gr.each {println it}
```

- A. class=class
Conference
name=Greach
year=2014
 - B. Conference@XXXXXX
 - C. Startup
failure
 - D. Greach
2014
- 

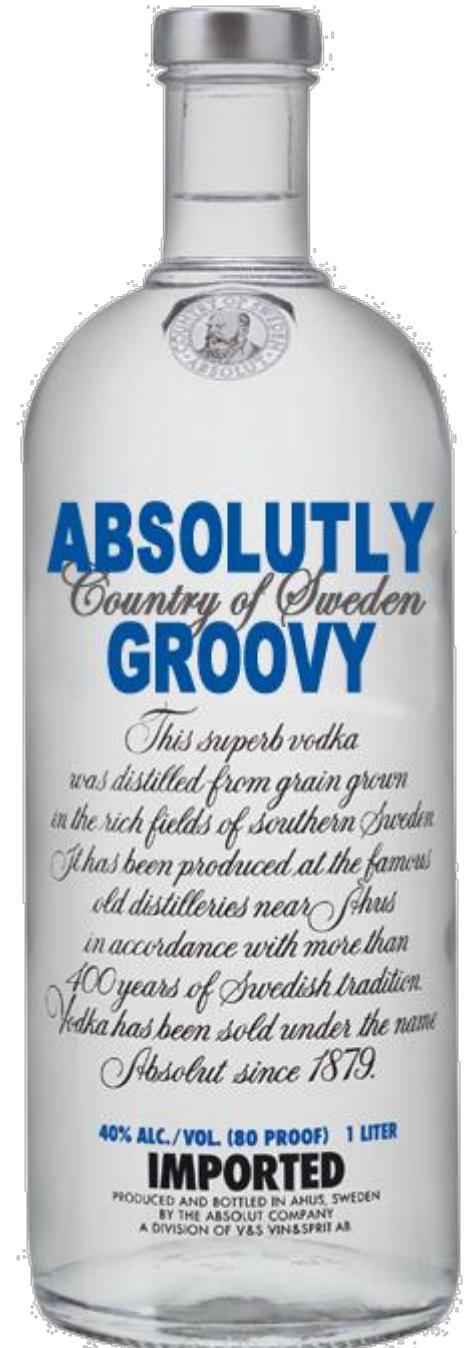
NOT EXACTLY ITERATION, IS IT?



RTFS (READ THE F***ING SOURCE)!

```
public static Collection asCollection(Object value) {  
    if (value == null) {...}  
    else if (value instanceof Collection) {...}  
    else if (value instanceof Map) {...}  
    else if (value.getClass().isArray()) {...}  
    else if (value instanceof MethodClosure) {...}  
    else if (value instanceof String) {...}  
    else if (value instanceof GString) {...}  
    else if (value instanceof File) {...}  
    else if (value instanceof Class && ((Class)value).isEnum()) {...}  
    else {  
        // let's assume it's a collection of 1  
        return Collections.singletonList(value);  
    }  
}
```

ABSOLUTELY GROOVY



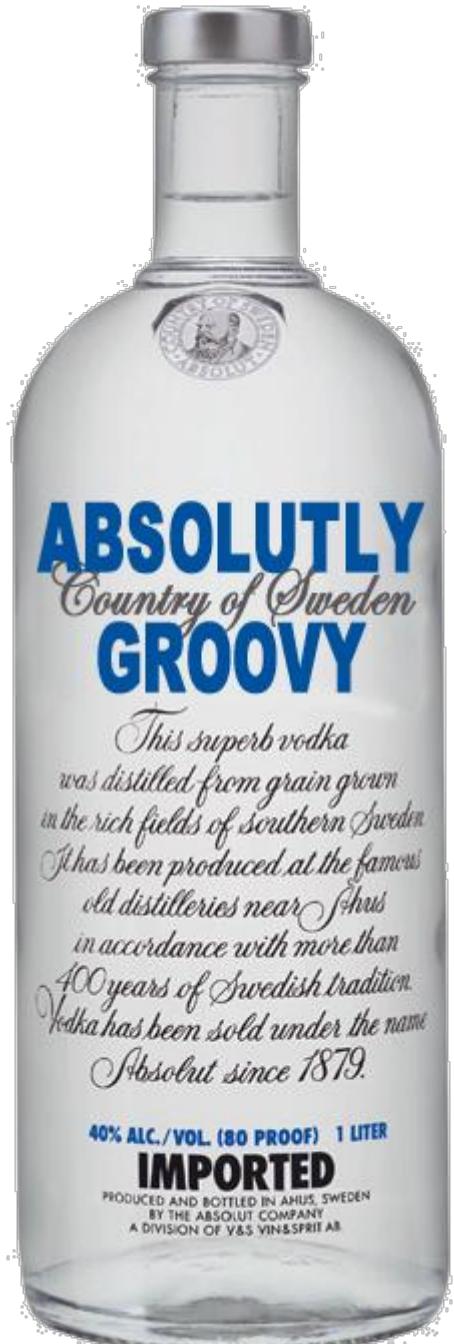
- 3. abs()

A.3

B.NoSuchMethodError

C.-3
The Groovy logo features the word "Groovy" in a stylized, bubbly font. The letter "G" has a circular hole through it, and the letters are surrounded by a blue starburst shape.

D.Execution Failure





WHAAATPP

G Groovy AST Browser

Show Script View Help

At end of Phase: Conversion

ClassNode - script1406423965039

Name	Value

Source Bytecode

```
public class script1406423965039 extends groovy.lang.Script {  
  
    public script1406423965039() {  
    }  
  
    public script1406423965039(groovy.lang.Binding context) {  
        super.setBinding(context)  
    }  
  
    public static void main(java.lang.String[] args) {  
        org.codehaus.groovy.runtime.InvokerHelper.runScript(script1406423965039, args)  
    }  
  
    public java.lang.Object run() {  
        -(3.abs())  
    }  
}
```

Unary minus is done last!

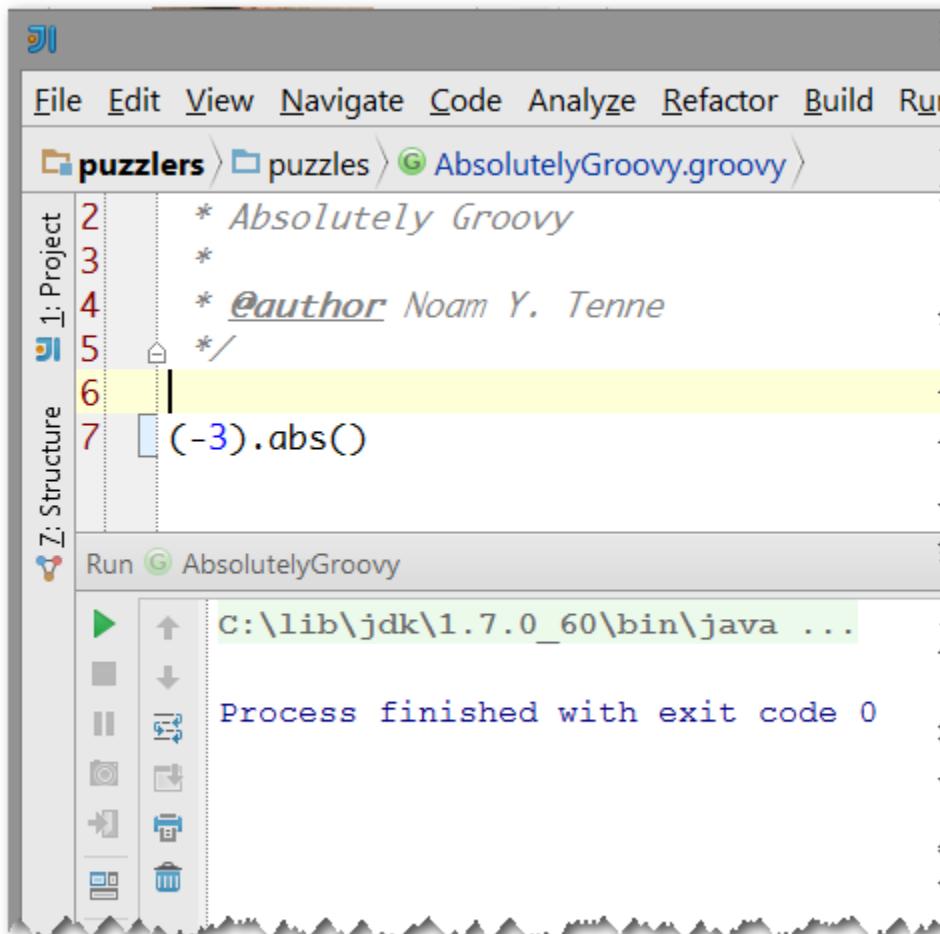
FIX IS SIMPLE

(-3).abs()

OR

```
int value = -3  
value.abs()
```

MY IDEA PRINTS NOTHING WHEN I COPY PASTE THIS CODE. WHAT GIVES?



The screenshot shows the IntelliJ IDEA interface with a Groovy script named `AbsolutelyGroovy.groovy`. The code contains the following lines:

```
* Absolutely Groovy
*
* @author Noam Y. Tenne
*/
(-3).abs()
```

The code is run, and the output window shows:

```
C:\lib\jdk\1.7.0_60\bin\java ...
Process finished with exit code 0
```

PRINTLN FOR THE RESCUE!

println (-3).abs()

A.-3

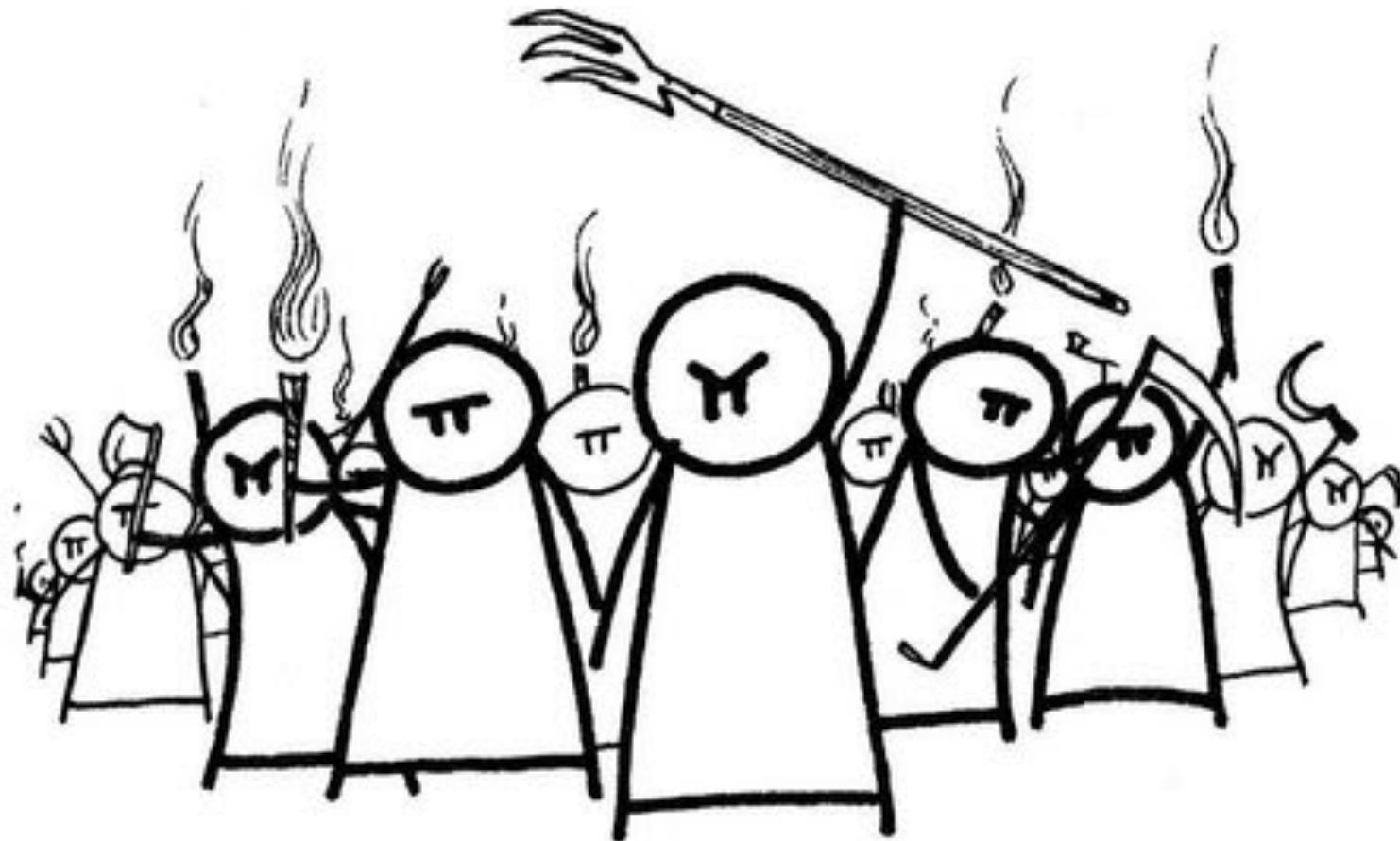
B.3

C.3 and NullPointerException

D.-3  NullPointerException



DELETE ALL PARENTHESES!



COMPILER WILL FIGURE IT OUT!

`println (-3).abs()`

-3

Caught: java.lang.NullPointerException: Cannot invoke method abs() on null object

java.lang.NullPointerException: Cannot invoke method abs() on null object
at AbsolutelyGroovy.run([AbsolutelyGroovy.groovy:7](#))
at com.intellij.rt.execution.application.AppMain.main(AppMain.java:134)

“All problems in computer science can be solved by another pair of parentheses”



John McCarthy, the inventor of LISP

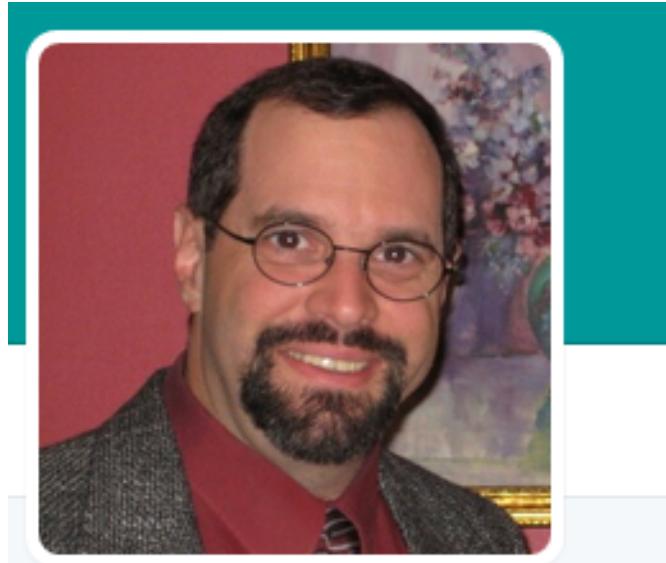
JUST THROW IN SOME MORE
PARENTHESES!

`println ((-3).abs())`

OR

```
int value = -3
println value.abs()
```

AND THE T-SHIRT GOES TO...



Ken Kousen

@kenkousen FOLLOW YOU

Software trainer and developer, NFJS speaker, and author of *Making Java Groovy*

📍 Marlborough, CT

🔗 kousenit.com

⌚ Joined August 2008

WHY SO PRIMITIVE?



WHY SO PRIMITIVE?

```
def x = int  
println x
```

```
if ((x = long)) {  
    println x  
}
```

```
if (x = boolean ) {  
    println x  
}
```



WHY SO PRIMITIVE?

```
def x = int  
println x
```

```
if ((x = long)) {  
    println x  
}
```

```
if (x = boolean) {  
    println x  
}
```

- 
- A. Standard or
 - B. int, long, boolean
 - C. java.lang.Integer,
java.lang.Long,
java.lang.Boolean

THAT WAS EASY. BUT WHY?

That's OK

A screenshot of an IDE showing Java code. The code defines an integer variable `x` and prints its value. It contains two `//if` statements: one for `x = long` and one for `x = boolean`. The second `//if` statement has a missing closing brace `}`. A yellow tooltip box highlights the cursor at the end of the second `//if` block. The status bar at the bottom shows the command line: `C:\lib\jdk\1.7.0_60\bin\java ...`, followed by the output: `int` and `Process finished with exit code 0`.

```
def x = int
println x

//if ((x = long)) {
//    println x
//}
//
//if (x = boolean ) {
//    println x
//}
//}
```

That's not OK

A screenshot of an IDE showing Java code. The code is identical to the one in the previous screenshot, but it results in a compilation error. A yellow tooltip box appears over the word `boolean` in the second `//if` statement, indicating a syntax error. The message box says: `'}' expected` and `Unexpected symbol`.

```
def x = int
//println x
//
//if ((x = long)) {
//    println x
//}
//
if (x = boolean ) {
}
//}
```

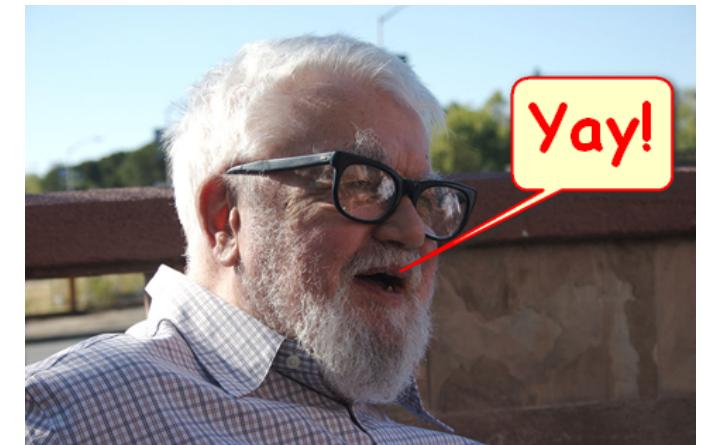
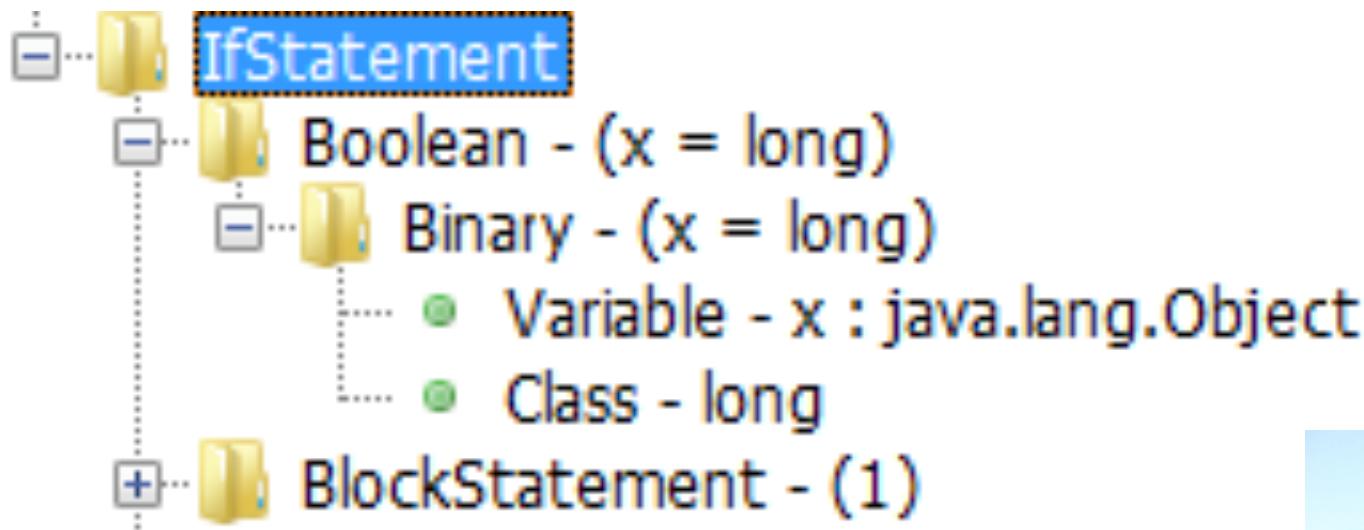
HOW ABOUT THAT:

```
def x = int
//println x
//
if ((x = long)) {
    println x
}
//
//if (x = boolean ) {
//    println x
//}

```



(GROOVY) TRUTH CAN HURT SO JUST PARENTHESES FOR A RESCUE (OR NOT)



WHODUNIT



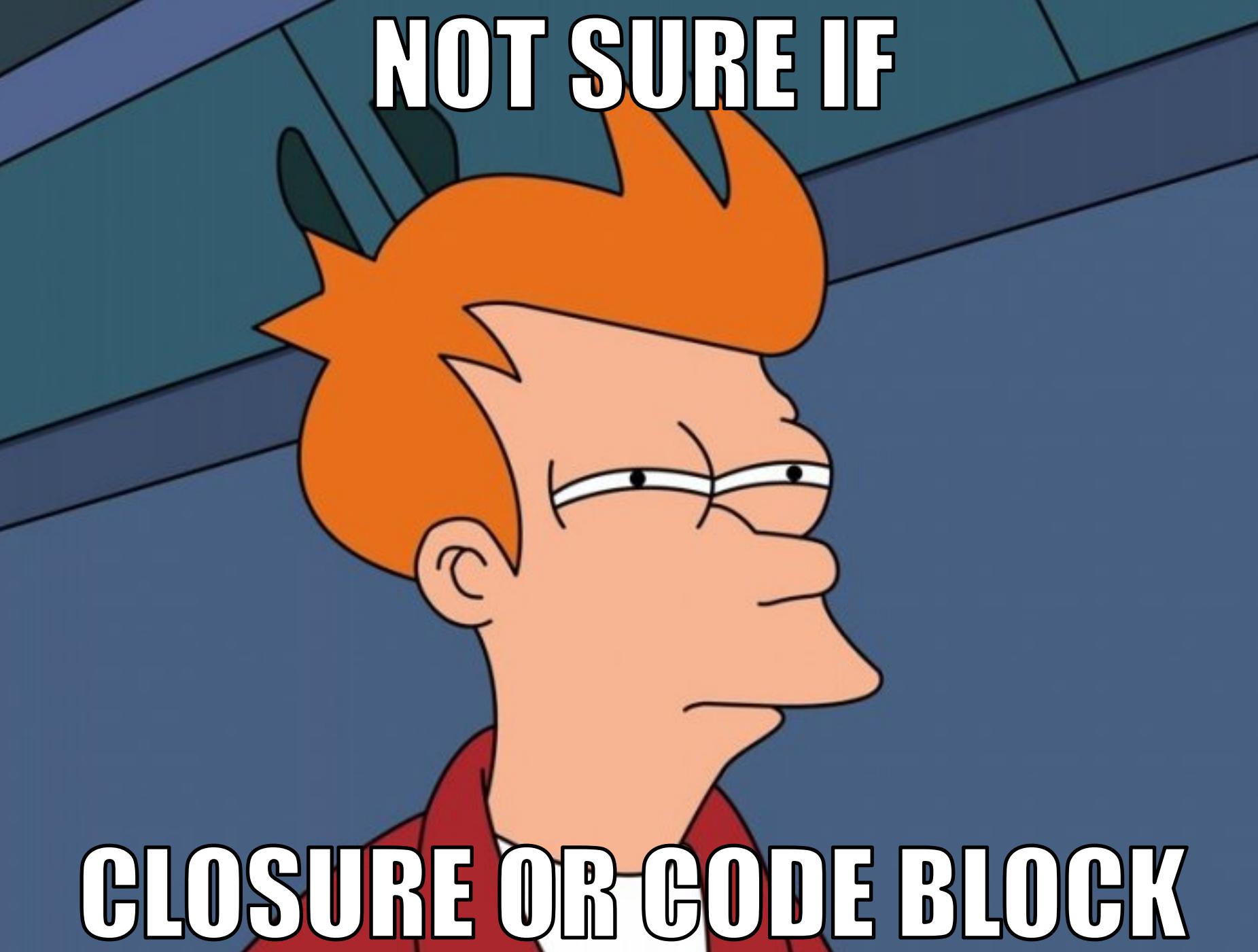
WHODUNIT

```
Closure whodunit() {  
    {  
        'The butler did it.'  
    }  
    println whodunit()
```

- A. NullPointerException
- B. whodunit_closure1@xxxxxx
- C. Startu or
- D. The butler did it.

THAT IS VERY SAD.





NOT SURE IF

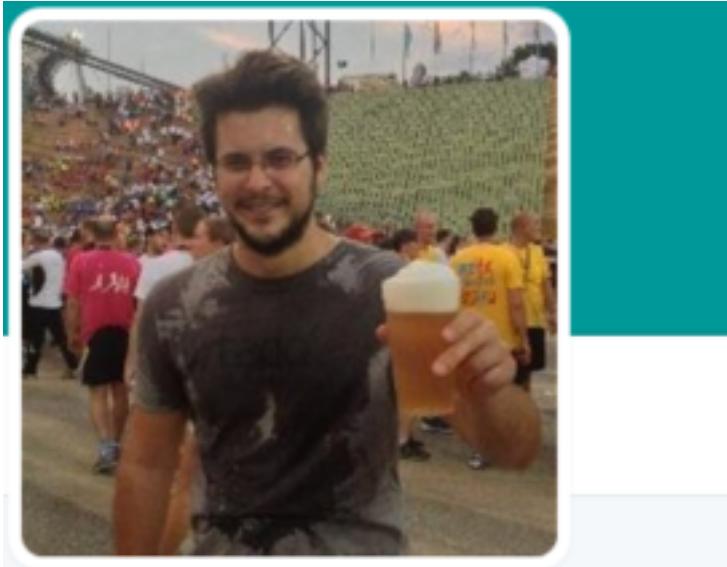
CLOSURE OR CODE BLOCK

CHEER UP, THE FIX IS EASY!

```
Closure whodunit() {  
    { ->  
        'The butler did it.'  
    }  
}
```



AND THE T-SHIRT GOES TO...



Deigote

@deigote

Software developer at [@tado](#), beer taster wannabe at [@unacervezaaldia](#) :)

 deigote.com

 Joined March 2007



PUBLIC -

GET YOUR PIECE OF PUBLIC PROPERTY.

```
trait Public {  
    public String property = "I am all public!"  
}  
  
class Property implements Public {}  
  
Property publicProperty = new Property()
```

```
trait Public {  
    public String property = "I am all public!"  
}
```

```
class Property implements Public {}
```

```
Property publicProperty = new Property()
```

- A. publicProperty.@property
- B. publicPropGroovyPublic__property
- C. publicProperty.getProperty()
- D. publicProperty.property



TWO QUESTIONS:

- 1.What's up with the
strange name?!
- 2.How the hell I am
supposed to know?!

ANSWER TO 1:

IN SOVIET RUSSIA



**YOU DON'T ACCESS
PUBLIC PROPERTY**

ANSWER TO 2:

The name of the field depends on the fully qualified name of the trait. All dots (.) in package are replaced with an underscore (_), and the final name includes a double underscore. So if the type of the field is `String`, the name of the package is `my.package`, the name of the trait is `Foo` and the name of the field is `bar`, in the implementing class, the public field will appear as:

```
String my_package_Foo__bar
```



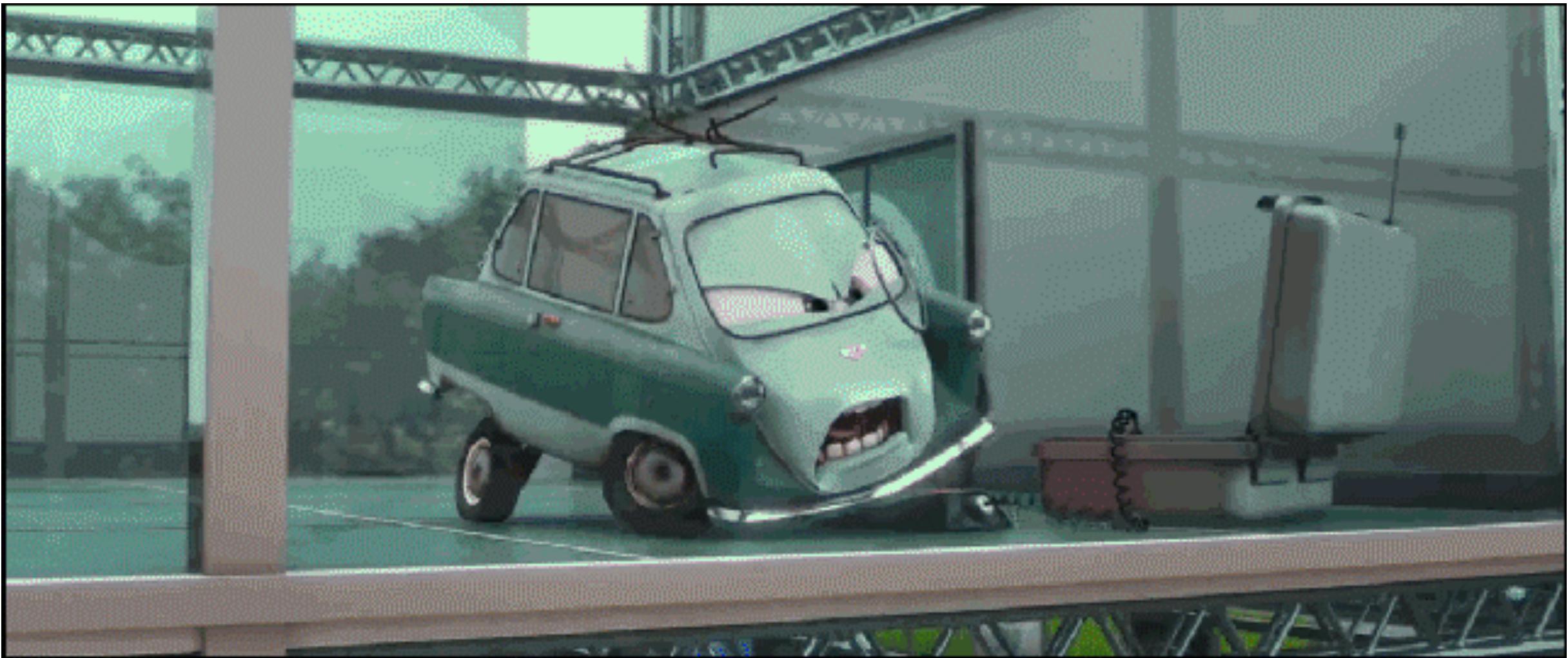
While traits support public fields, it is not recommended to use them and considered as a bad practice.

READ THE TRAITS DOCS



IT'S ALL THERE

OUT OF RANGE



OUT OF RANGE

```
def range = 1.0..10.0  
assert range.contains(5.0)  
println range.contains(5.6)
```

A.Assertion Failed

B.fail

C.true

D.NullPointerException

But... but...



why?

ROAD TO ENLIGHTENMENT, STEP 1:

The screenshot shows a code editor and a terminal window. The code editor contains the following Groovy script:

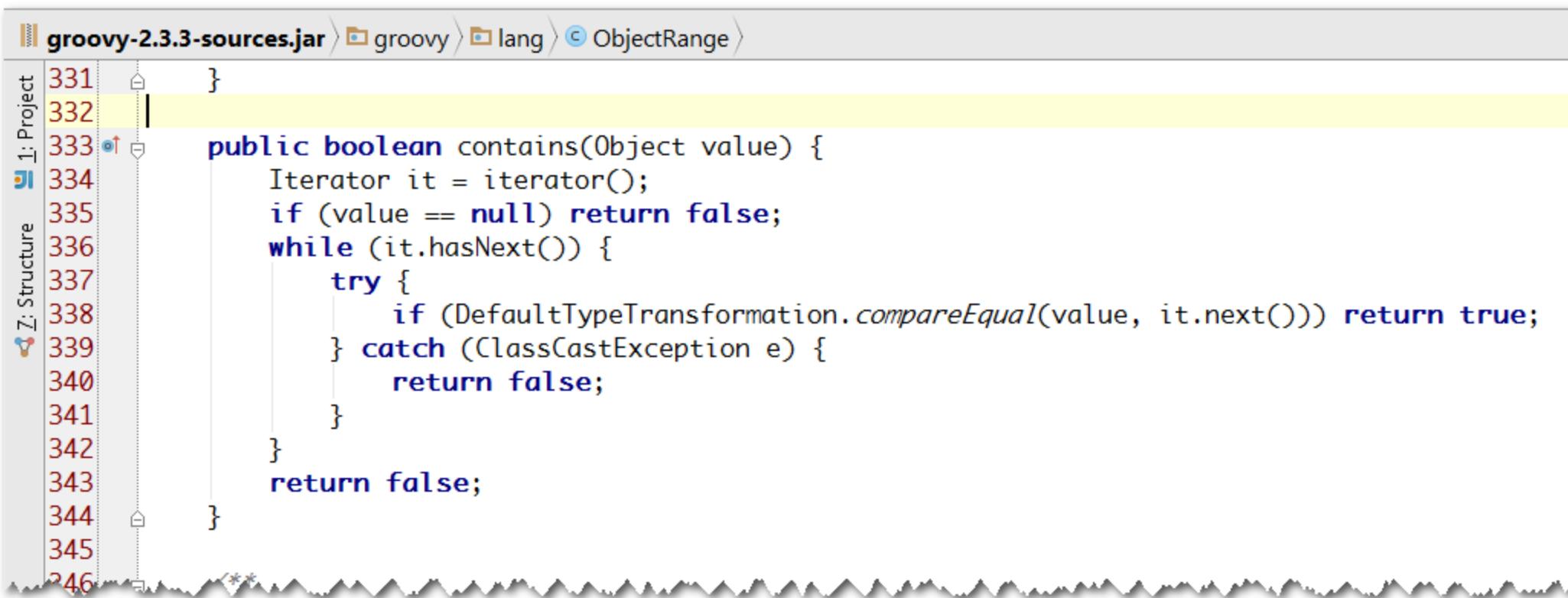
```
6
7 def range = 1.0..10.0
8
9 println range.class.name
```

The terminal window shows the command run and the output:

```
Run OutOfRange
C:\lib\jdk\1.7.0_60\bin\java ...
groovy.lang.ObjectRange
```

At the bottom, the message "Process finished with exit code 0" is displayed.

ROAD TO ENLIGHTENMENT, STEP 2:



The screenshot shows a code editor window with the following details:

- Project Path:** groovy-2.3.3-sources.jar > groovy > lang > ObjectRange
- Code Editor:** The code is displayed in a syntax-highlighted editor. The highlighted line contains the method signature: `public boolean contains(Object value) {`.
- Code Content:**

```
331     }
332 }
333 public boolean contains(Object value) {
334     Iterator it = iterator();
335     if (value == null) return false;
336     while (it.hasNext()) {
337         try {
338             if (DefaultTypeTransformation.compareEqual(value, it.next())) return true;
339         } catch (ClassCastException e) {
340             return false;
341         }
342     }
343     return false;
344 }
```
- Toolbars and Menus:** Standard Java IDE toolbars and menus are visible at the top of the window.

ROAD TO ENLIGHTENMENT, STEP 2:

The screenshot shows a code editor window with the following file path: groovy-2.3.3-sources.jar > groovy > lang > ObjectRange. The code is written in Groovy and implements the `ObjectRange` interface. A red callout box points to line 333, which contains the `contains` method implementation. The callout box contains the text "Why no JavaDoc, Doc?".

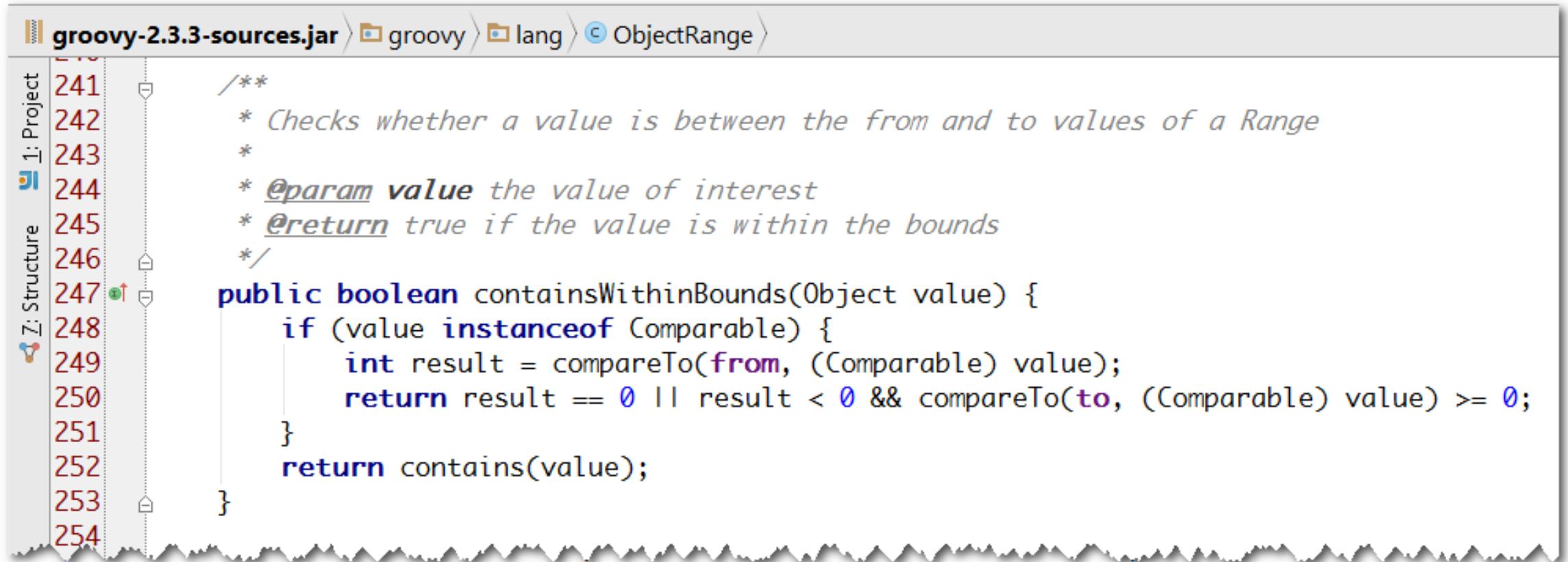
```
331     }
332 }
333 public boolean contains(Object value) {
334     Iterator it = iterator();
335     if (value != null) {
336         while (it.hasNext()) {
337             try {
338                 if (DefaultTypeTransformation.compareEqual(value, it.next())) return true;
339             } catch (ClassCastException e) {
340                 return false;
341             }
342         }
343     }
344     return false;
345 }
346 /**
347 *
```

ROAD TO ENLIGHTENMENT, STEP 3:

```
Iterator iterator = (1.0..10.0).iterator()  
while (iterator.hasNext()) {  
    print "${iterator.next()}"  
}
```

1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0

FIX IS SIMPLE, BUT QUESTIONABLE



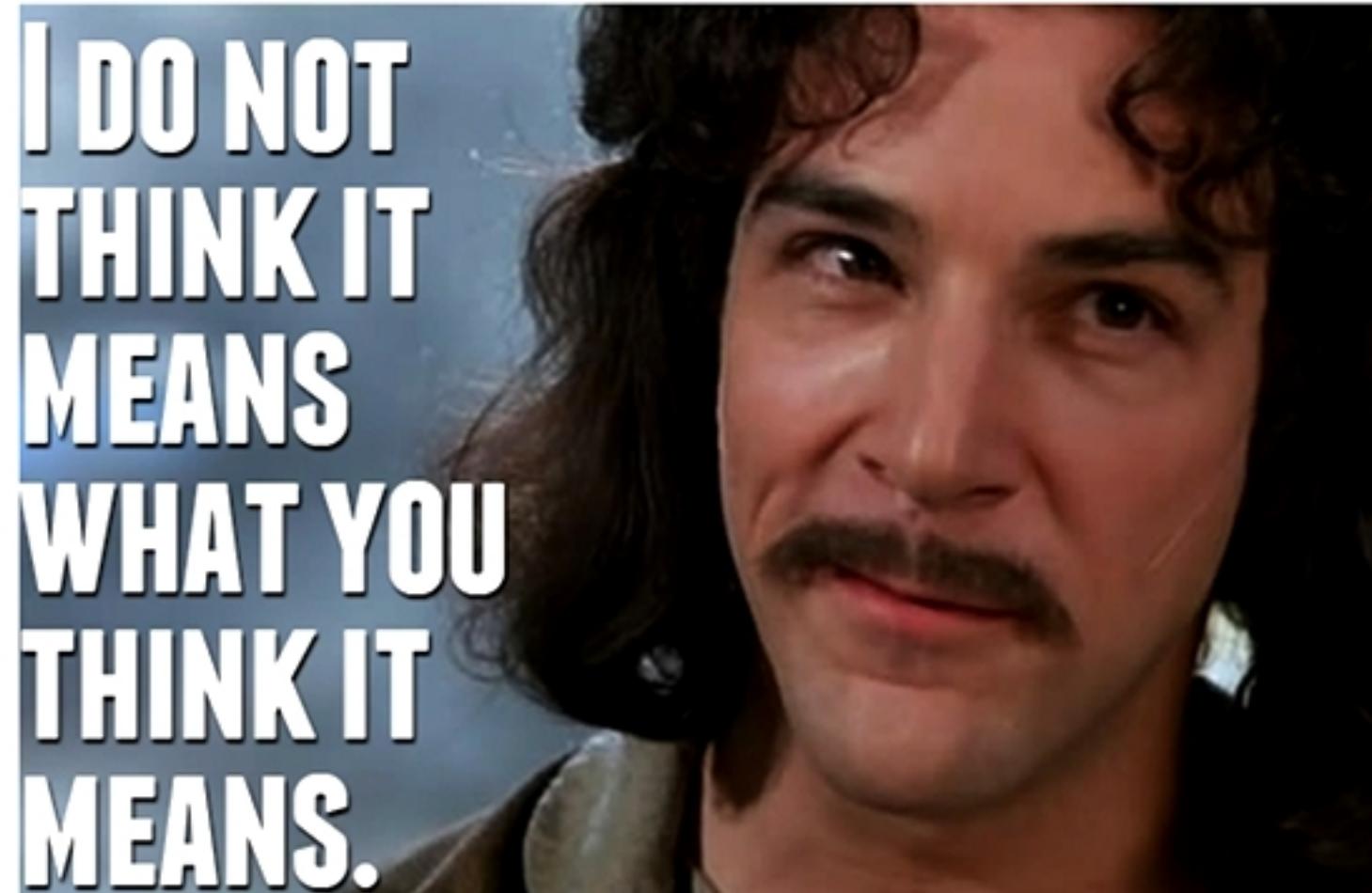
The screenshot shows a code editor interface with the following details:

- Project:** groovy-2.3.3-sources.jar
- Structure:** groovy / lang / ObjectRange
- Code (Groovy):**

```
241     /**
242      * Checks whether a value is between the from and to values of a Range
243      *
244      * @param value the value of interest
245      * @return true if the value is within the bounds
246      */
247     public boolean containsWithinBounds(Object value) {
248         if (value instanceof Comparable) {
249             int result = compareTo(from, (Comparable) value);
250             return result == 0 || result < 0 && compareTo(to, (Comparable) value) >= 0;
251         }
252         return contains(value);
253     }
```

SO, GROOVY, YOU SAY THAT 1.0..10.0
DOES NOT CONTAIN 5.6?

I DO NOT
THINK IT
MEANS
WHAT YOU
THINK IT
MEANS.



WHAT CAN BE DONE?

- We don't care about implementation class
- Make it “RealRange”
“ContinuousRange”
- In this implementation contains to
containsWithinBounds.
- Fix GROOVY-2771

groovy / GROOVY-2771
range of double-values broken?

[Log In](#)

Details

Type:	<input checked="" type="radio"/> Bug	Status:	 Closed
Priority:	 Major	Resolution:	Won't Fix
Affects Version/s:	1.5.4	Fix Version/s:	None
Component/s:	None		
Labels:	None		
Environment:	Groovy Shell (1.5.4, JVM: 1.5.0_11-b03)		
Number of attachments :	0		

Description

I got following error:
D:\>groovysh
Groovy Shell (1.5.4, JVM: 1.5.0_11-b03)
Type 'help' or '\h' for help

People

Assignee:  Paul King
Reporter:  Peter Fürholz
Votes:  Vote for this issue
Watchers:  Start watching this issue

Dates

Created: 23/Apr/08 4:29 AM

Thank you.

AND THE T-SHIRT GOES TO...



Scott Leberknight

@sleberknight

 Joined August 2008

POWER RANGE(RS)



LET'S TAKE ONE DOWN...

[0..9].each { println(it - 1) }

A. -1
0
1
2
3
4
5
6
7
8

B. 0
1
2
3
4
5
6
7
8
9

C. [0, 2, , , 5, 6, 7, 8, 9]

D. [-1, 0, 1, 2, 3, 4, 5, 6, 7, 8]



I JUST TRICKED YOU. HA!

```
[0..9].each { println(it - 1) }
```

I JUST TRICKED YOU. HA!

```
[0..9].each { println(it - 1) }
```

This is range

I JUST TRICKED YOU. HA!

```
[0 .. 9].each { println(it - 1) }
```

This is range

This is list with one
element - the range

I JUST TRICKED YOU. HA!

```
[0 .. 9].each { println(it - 1) }
```

This is range

In the sole iteration
the it is the range

This is list with one
element - the range

I JUST TRICKED YOU. HA!

```
[0 .. 9].each { println(it - 1) }
```

This is range

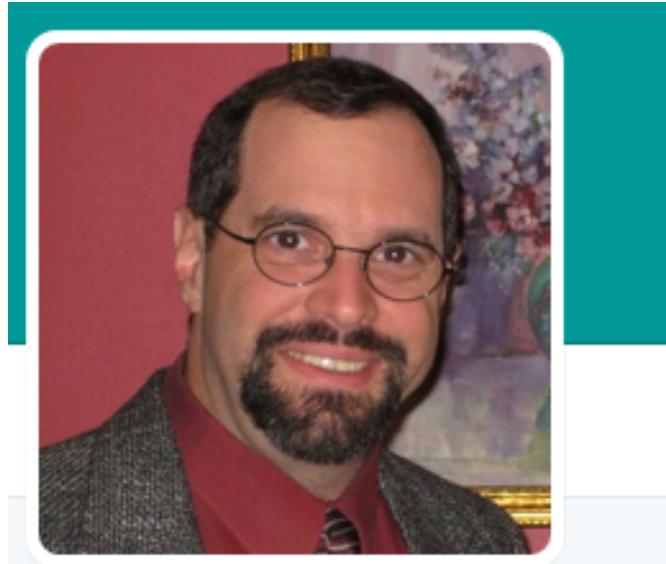
This is list with one element - the range

In the sole iteration the it is the range

And we just remove 1 from it



AND THE T-SHIRT GOES TO...



Ken Kousen

@kenkousen FOLLOW YOU

Software trainer and developer, NFJS speaker, and author of *Making Java Groovy*

📍 Marlborough, CT

🔗 kousenit.com

⌚ Joined August 2008

FRESH IN MEMORY!

International Talk Like a Pirate Day



Type	Parodic
Date	September 19
Next time	19 September 2014
Frequency	annual

TREAYE MAP, ARRR!



TREAYE MAP, ARRR!

```
def key = 'x'  
def map = [key: 'treasure']  
def value = map.get(key)  
println value
```

```
def key = 'x'  
def map = [key: 'treasure']  
def value = map.get(key)  
println value
```

A.NoSuchElementException



B.null

C.treasure

D.I ilv-livered swab. BSOD!



BY DEFAULT, KEY IS A LITERAL, NO MATTER WHAT

```
7 def key = 'x'  
8 def map = [key: 'treasure']  
9 def value = map.get(key)  
10 printl
```

See the green?
Green means "String"

FIXES, LOTS OF THEM

1. `def map = [(key): 'treasure']`
2. `map.put(key, 'treasure')`
3. `map."$key" = 'treasure'`
4. `map[key] = 'treasure'`



MORE ROUTES T' TO TREAAYE!



A NUMBER OF ROUTES

```
def map = [2: 'treasure']
def key = 2
def value = map."$key"
println value
```

```
def map = [2: 'treasure']
def key = 2
def value = map."$key"
println value
```

A.NoSuchElementException

B.me

C.treasure

D.Kernel Panic, ye lice-infested

NOW WHAT?!



BY DEFAULT, KEY IS A LITERAL, NO
MATTER WHAT, EXCEPT NUMBERS.

```
def map = [2: 'treasure']
println map.keySet().first().class.name
```

java.lang.Integer



AND THE T-SHIRT GOES TO...



Guillaume Laforge

@glaforge FOLLOWS YOU

Groovy project manager

📍 Paris, France

🔗 glaforge.appspot.com

⌚ Joined April 2008

SAVE THE DATE



JAVAONE IS COMING

JAVA2, ANYONE?

```
List<Long> list = [1,2,3]
def now = new Date()
list << now
println list
```

```
List<Long> list = [1,2,3]
def now = new Date()
list << now
println list
```

A.Startup Failure

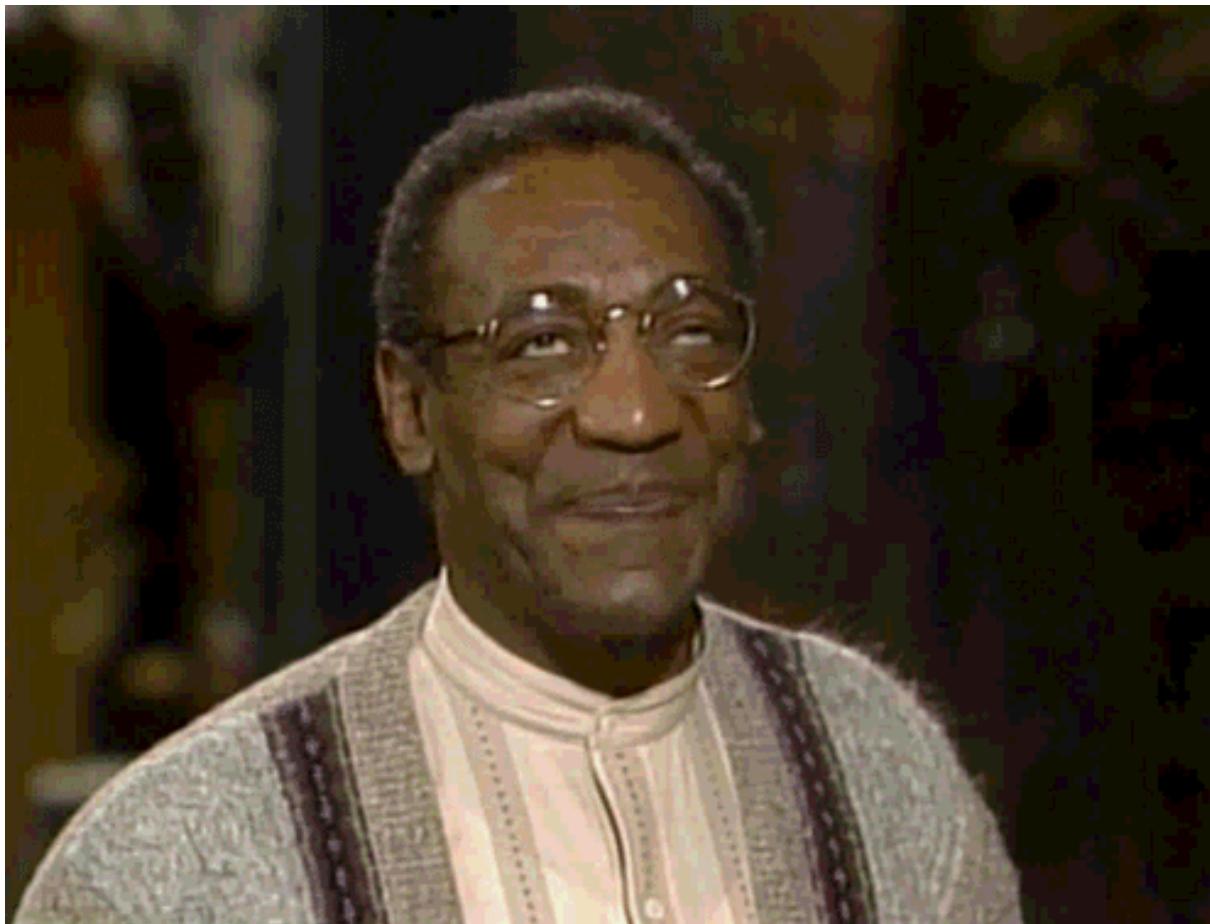
B.[1, 2, 3, WED
2014]

C.[1, 2, 3, 1410388511456]

D. ClassCastException



OH, WAIT, I KNOW WHAT
HAPPENED!



ERASURE!



ERASURE!



The Java™ Tutorials

[Download Ebooks](#)
[Download JDK](#)
[Search Java Tutorials](#)
[Hide TOC](#)

[Generics \(Updated\)](#)

[Why Use Generics?](#)

[Generic Types](#)

[Raw Types](#)

[Generic Methods](#)

[Bounded Type Parameters](#)

[Generic Methods and
Bounded Type](#)

[Parameters](#)

[Generics, Inheritance, and
Subtypes](#)

[Type Inference](#)

[Wildcards](#)

[Upper Bounded](#)

[Wildcards](#)

[Unbounded Wildcards](#)

[Lower Bounded](#)

[« Previous](#) • [Trail](#) • [Next »](#) [Home Page](#) > [Learning the Java Language](#) > [Generics \(Updated\)](#)

Type Erasure

Generics were introduced to the Java language to provide tighter type checks at compile time and to support generic programming. To implement generics, the Java compiler applies type erasure to:

- Replace all type parameters in generic types with their bounds or `Object` if the type parameters are unbounded. **The produced bytecode, therefore, contains only ordinary classes, interfaces, and methods.**
- Insert type casts if necessary to preserve type safety.
- Generate bridge methods to preserve polymorphism in extended generic types.

Type erasure ensures that no new classes are created for parameterized types; consequently, generics incur no runtime overhead.

**BUT WHERE ARE MY
COMPILE TIME CHECKS?!**



“IN GROOVY YOU CAN INSERT
ANYTHING ANYWHERE”

```
List<Long> list = [1,2,3]
def now = new Date()
list << now
list << 'foo'
println list*.class.name
```

```
[java.lang.Long, java.lang.Long,
java.lang.Long, java.util.Date,
java.lang.String]
```

COMPILESTATIC!

The screenshot shows an IDE interface with a code editor and a 'Structure' tool window.

Code Editor:

```
    ^/  
    @CompileStatic  
    def saveTheDate() {  
        List<Long> list = [1l, 2l, 3l]  
        def now = new Date()  
        list << now  
        print list
```

Structure Window:

Z: Structure

Tool Window:

'leftShift' in 'org.codehaus.groovy.runtime.DefaultGroovyMethods' cannot be applied to '(java.util.Date)'

saveTheDate()

W H A T O S T H E F R E Q U E N C Y
K E N N E T H ?

R.E.M.

Compact Disc Maxi-Single

WHAT'S THE METACLASS, BARUCH?

```
def map = [metaClass: 'frequency']
println "What's the $map.metaClass, Baruch?"
```

A. MissingMethodException

B. What's the

org.codehaus.groovy.runtime.HandleMetaClass@YYYYYYYY
[groovy.lang.MetaClassImpl@YYYYYYYY[class
java.util.LinkedHashMap]], Baruch?

C. What's the java.util.LinkedHashMap@XXXXXX.metaClass,
Baruch?

D. What's the frequency, Baruch?



IT WORKS, BUT...



NOT SURE IT'S A GOOD THING

REMEMBER JAMES GOSLING TOLD YOU



THAT OPERATOR OVERLOADING IS BAD?

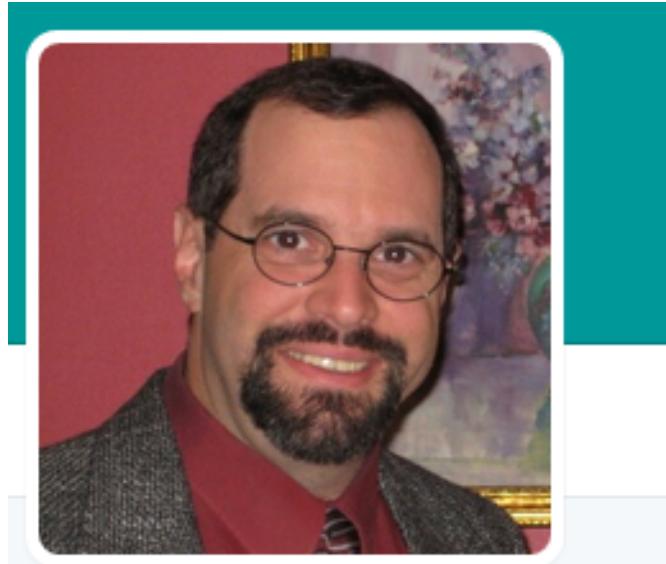
TOO MUCH OPERATOR OVERLOADING!

If `map.metaClass` is overloaded
for `map.get('metaClass')`, it can't
be overloaded for
`map.getMetaClass()` in the
same time!



MAYYCE

AND THE T-SHIRT GOES TO...



Ken Kousen

@kenkousen FOLLOW YOU

Software trainer and developer, NFJS speaker, and author of *Making Java Groovy*

📍 Marlborough, CT

🔗 kousenit.com

⌚ Joined August 2008

PRIME CUTS



HEY VEGETARIANS

Explain these!

PRIME CUTS

```
boolean isPrime(def x) {  
    if (x == 2) return true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            return false  
        }  
    }  
    true  
}  
  
println isPrime("4" as Double)
```

CONFUSED? SO ARE WE.



LET'S LOOK AT IT AGAIN...

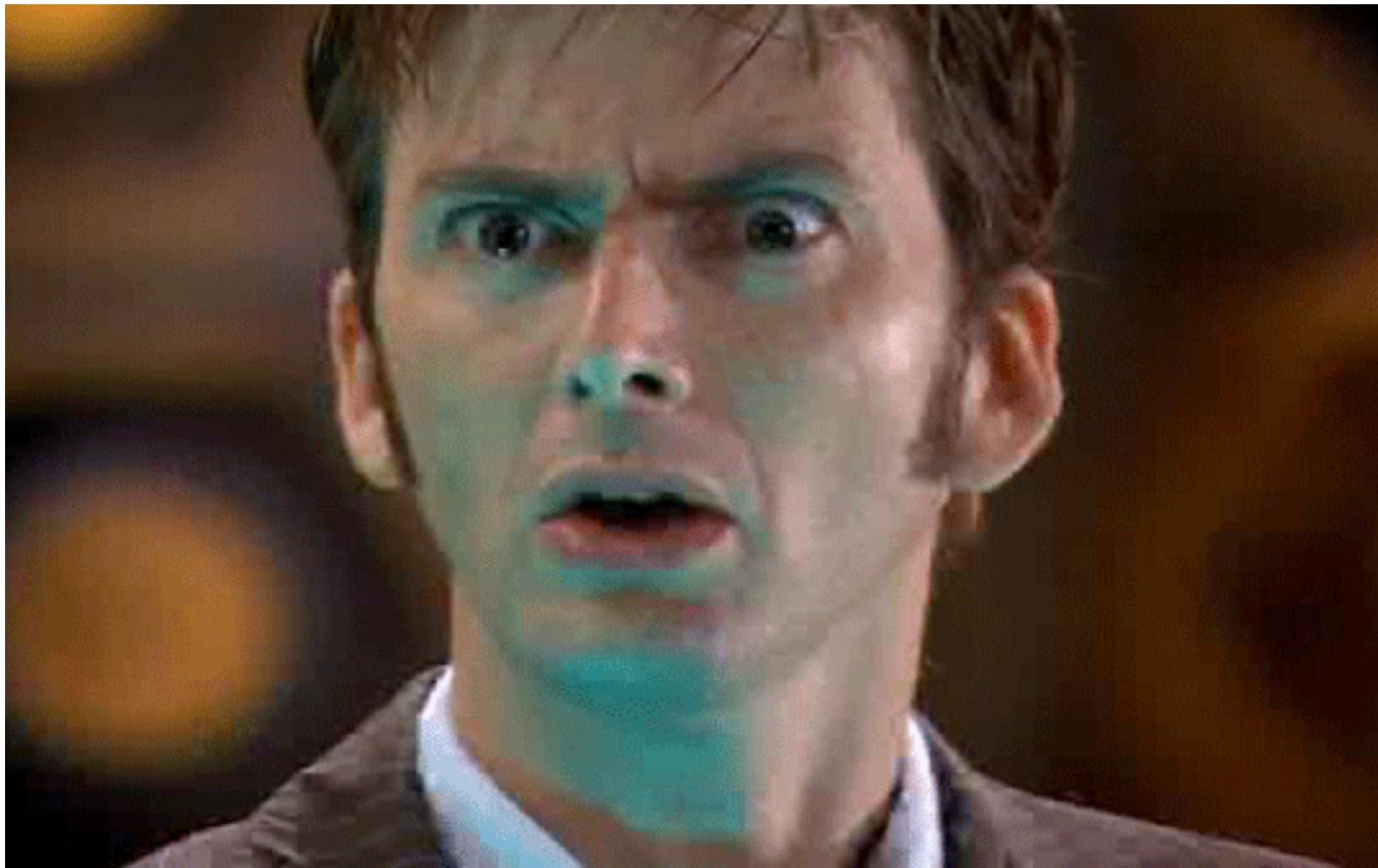
```
boolean isPrime(def x) {  
    if (x == 2) return true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            return false  
        }  
    }  
    true  
}  
  
println isPrime("4" as Double)
```

```
boolean isPrime(def x) {  
    if (x == 2) return true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            return false  
        }  
    }  
    true  
}
```

```
println isPrime("4" as Double)
```



- A.true
- B.false
- C.NumberFormatException
- D.MissingMethodException



LET'S TRY TO FIGURE IT OUT

```
boolean isPrime(def x) {  
    if (x == 2) return true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            return false  
        }  
    }  
    true  
}
```

LET'S TRY TO FIGURE IT OUT

```
boolean isPrime(def x) {  
    if (x == 2) return true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            return false  
        }  
    }  
    true  
}
```

This is closure on every element
in range, it has its own scope

LET'S TRY TO FIGURE IT OUT

```
boolean isPrime(def x) {  
    if (x == 2) return true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            return false  
        }  
    }  
    true  
}
```

This returns from the closure,
not from method isPrime
a.k.a. "local return"

LET'S TRY TO FIGURE IT OUT

```
boo  
    each method returns null, so  
    return from closure is ignored {  
        true  
        int limit = Math.sqrt(x) + 1  
        (2..limit).each {  
            if (x % it == 0) {  
                return false  
            }  
        }  
        true  
    }
```

This returns from the closure,
not from method isPrime
a.k.a. "local return"

LET'S TRY TO FIGURE IT OUT

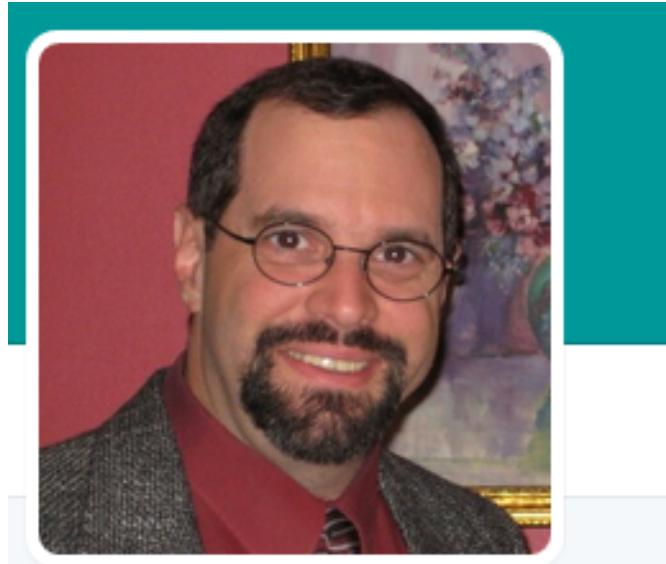
```
boo each method returns null, so  
return from closure is ignored {  
    true  
    int limit = Math.sqrt(x) + 1  
    (2..limit).each {  
        if (x % it == 0) {  
            isPrime always  
            return false  
            returns true  
        }  
    }  
}  
This returns from the closure,  
not from method isPrime  
a.k.a. "local return"
```

SOLUTION? USE ANY() OR
RETHINK THE ALGORITHM



[http://kousenit.wordpress.com/2014/04/18/
responses-to-the-closure-of-no-return/](http://kousenit.wordpress.com/2014/04/18/responses-to-the-closure-of-no-return/)

AND THE T-SHIRT GOES TO...



Ken Kousen

@kenkousen FOLLOW YOU

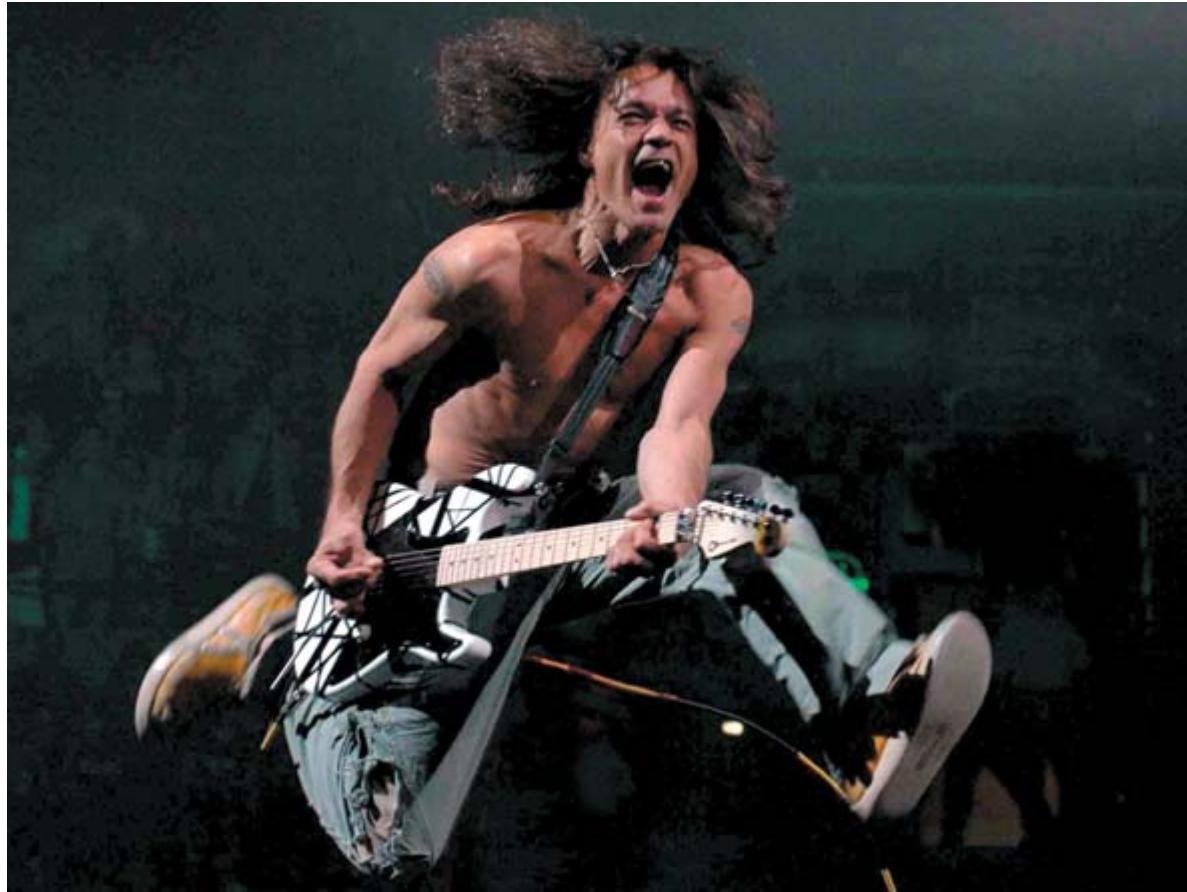
Software trainer and developer, NFJS speaker, and author of *Making Java Groovy*

📍 Marlborough, CT

🔗 kousenit.com

⌚ Joined August 2008

JUMP! A.K.A. THE MISSING LYRICS



JUMP! A.K.A. THE MISSING LYRICS

```
class VanHalen {  
  
    public static jump() {  
        "Here are the ${lyrics()}"  
    }  
  
    def methodMissing(String name, def args) {  
        'lyrics'  
    }  
}  
  
println VanHalen.jump()
```

JUMP! A.K.A. THE MISSING LYRICS

```
class VanHalen {  
  
    public static jump() {  
        "Here are the ${lyrics()}"  
    }  
  
    def methodMissing(String name, def args) {  
        'lyrics'  
    }  
  
    println VanHalen.jump()
```

- A. Here are the lyrics
 - B. Here are the null
 - C. Startup failure
 - D. MissingMethodException
- 

I CAN'T BELIEVE YOU FELL FOR THE
OLDEST TRICK IN THE BOOK



I CAN'T BELIEVE YOU FELL FOR THE OLDEST TRICK IN THE BOOK

The screenshot shows a code editor with JavaDoc documentation for the `groovy.lang.MetaClass` class. The code editor displays the following code:

```
Object invokeMissingMethod(Object instance, String methodName, Object[] arguments)
```

The JavaDoc documentation is as follows:

Documentation for `invokeMissingMethod(Object, String, Object[])`

groovy-2.3.3

`groovy.lang.MetaClass`

```
Object invokeMissingMethod(Object instance,
                           String methodName,
                           Object[] arguments)
```

Attempts to invoke the methodMissing method otherwise throws a MissingMethodException

Parameters:

- `instance` - The instance to invoke methodMissing on
- `methodName` - The name of the method
- `arguments` - The arguments to the method

I CAN'T BELIEVE YOU FELL FOR THE OLDEST TRICK IN THE BOOK

```
class VanHalen {  
    public static jump() {  
        "Here are the ${lyrics()}"  
    }  
  
    def methodMissing(String name, def args) {  
        'lyrics'  
    }  
}
```

Can't invoke instance
method from static context!

I CAN'T BELIEVE YOU FELL FOR THE OLDEST TRICK IN THE BOOK

```
'  
8  class VanHalen {  
9  
10     public static jump() {  
11         "Here are the ${lyrics()}"  
12     }  
13  
14     def methodMissing(String name, def args) {  
15         'lyrics'  
16     }  
17  
18 }  
19  
println VanHalen.jump()
```

Run  TheMissingLyrics

```
C:\lib\jdk\1.7.0_60\bin\java ...  
Caught: groovy.lang.MissingMethodException: No signature of method: static VanHalen.lyrics() is applicable for argument types: () values: []  
Possible solutions: print(java.io.PrintWriter), print(java.lang.Object), is(java.lang.Object), notify(), wait(), grep()  
groovy.lang.MissingMethodException: No signature of method: static VanHalen.lyrics() is applicable for argument types: () values: []  
Possible solutions: print(java.io.PrintWriter), print(java.lang.Object), is(java.lang.Object), notify(), wait(), grep()  
    at VanHalen.jump(TheMissingLyrics.groovy:11)  
    at VanHalen$jump.call(Unknown Source)  
    at TheMissingLyrics.run(TheMissingLyrics.groovy:19) <1 internal calls>
```

SOLUTION: STATIC METHODMISSING

```
class VanHalen {  
  
    public static jump() {  
        "Here are the ${lyrics()}"  
    }  
  
    static $static_methodMissing(String name, def args) {  
        'lyrics'  
    }  
  
}  
  
println VanHalen.jump()
```

BETTER SOLUTION: USE GOOD OOP

```
class VanHalen {  
  
    public jump() {  
        "Here are the ${lyrics()}"  
    }  
  
    def methodMissing(String name, def args) {  
        'lyrics'  
    }  
}  
  
println new VanHalen().jump()
```

π - THE FOOD AND THE NUMBER



DOUBLE THE PI

```
double value = 3  
println "$value.14".isDouble()
```

- A. true
- B. MissingPropertyException
- C. false
- D. MissingMethodException

GOT THIS ONE!



PIECE OF PIE!

```
class Person{ String name }
Person person = new Person(name: 'Andres')
println "The name is $person.name"
double value = 3
println "$value.14".isDouble()
```

Refers to property

PIECE OF PIE!

```
class Person{ String name }
Person person = new Person(name: 'Andres')
println "The name is $person.name"
double value = 3
println "$value.14".isDouble()
```

Refers to property

Refers to property `14` of double? WTF!

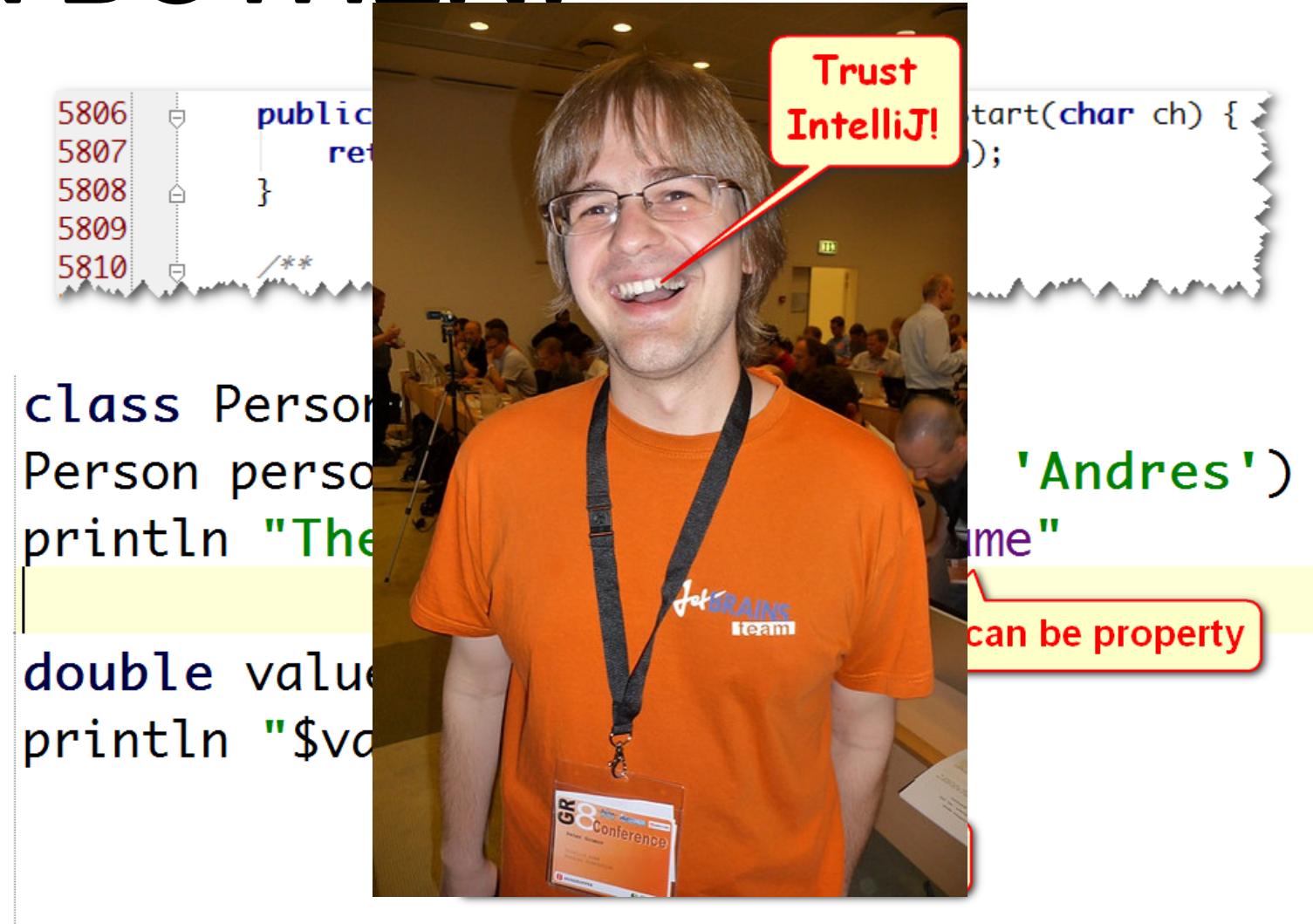
DOUBLE THE PI

```
double value = 3  
println "$value.14".isDouble()
```

- A. true
- B. MissingPropertyException
- C. fail
- D. MissingMethodException



IF IT CAN'T BE A PROPERTY, WHY EVEN BOTHER?



MIND THE TOSTRING() OF A DOUBLE

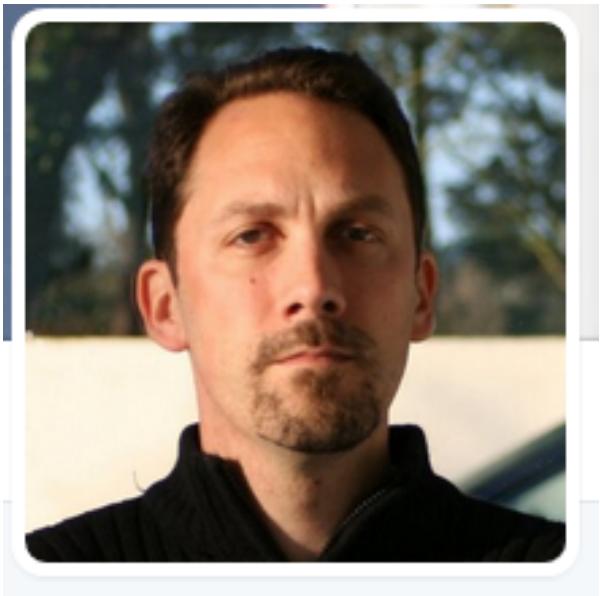
The screenshot shows a Java code editor and a terminal window. The code in the editor is:

```
7
8     double value = 3
9     println value
10    println "$value.14"
```

The terminal window shows the following output:

```
Run G DoubleThePi
C:\lib\jdk\1.7.0_60\bin\java ...
3.0
3.0.14
```

EVER WONDERED WHAT RSVP
STANDS FOR?



ASK THE FRENCH
GUYS!

RSVP

```
class Invite {  
    int attending = 1  
}  
  
def invite = new Invite()  
def attendees = (invite.attending) +1  
println attendees
```

```
class Invite {  
    int attending = 1  
}
```

```
def invite = new Invite()  
def attendees = (invite.attending) +1  
println attendees
```

A. StartGroovyFailure

B. 1

C. 2

D. MissingPropertyException



```
class Invite {  
    int attending = 1  
}  
  
def invite = new Invite()  
def attendees = (invite.attending) +1  
println attendees
```

WAT?!

Rsvp

```
C:\lib\jdk\1.7.0_60\bin\java ...  
org.codehaus.groovy.control.MultipleCompilationErrorsException: startup failed:  
C:\Users\jbaruch\projects\puzzlers\puzzles\Rsvp.groovy: 12: unable to resolve class invite.attending  
@ line 12, column 17.  
    def attendees = (invite.attending) +1  
                           ^  
1 error
```



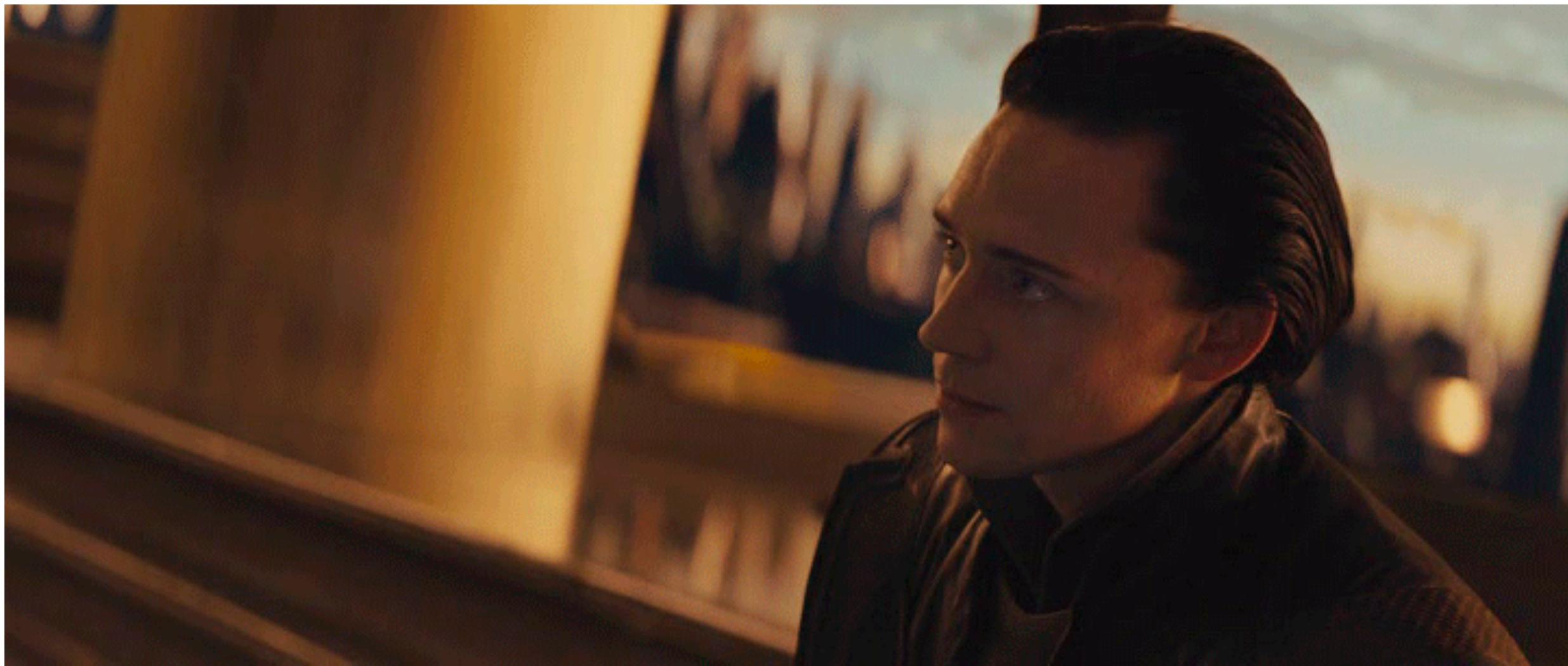
THAT'S WHY:

```
def invite = new Invite()  
def attendees = (invite.attending) + 1  
println attendees
```

This is class cast!

Package name

Class name



FIX: WHATEVER DOESN'T LOOK LIKE CASTING.

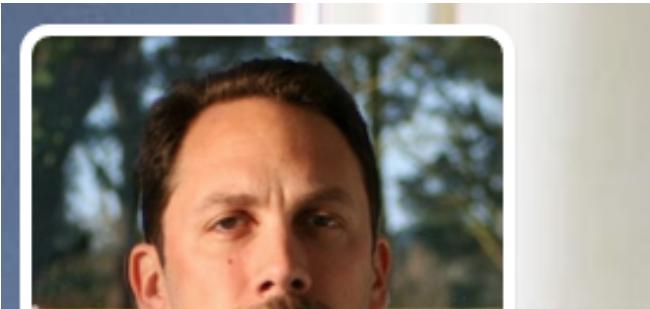
THIS WORKS:

```
def attendees = (new Invite()).attending + 1
println attendees
```

THIS WORKS AS WELL:

```
def invite = new Invite()
println (invite.attending + 1)
```

AND THE T-SHIRT GOES TO....



Cédric Champeau

@CedricChampeau FOLLOWERS

Software Engineer at Pivotal. OSS advocate, Groovy language. Conference speaker. Introvert (carlkingdom.com/10-myths-about...). Wrote the static compiler for #groovylang.

📍 St Hilaire de Loulay, France

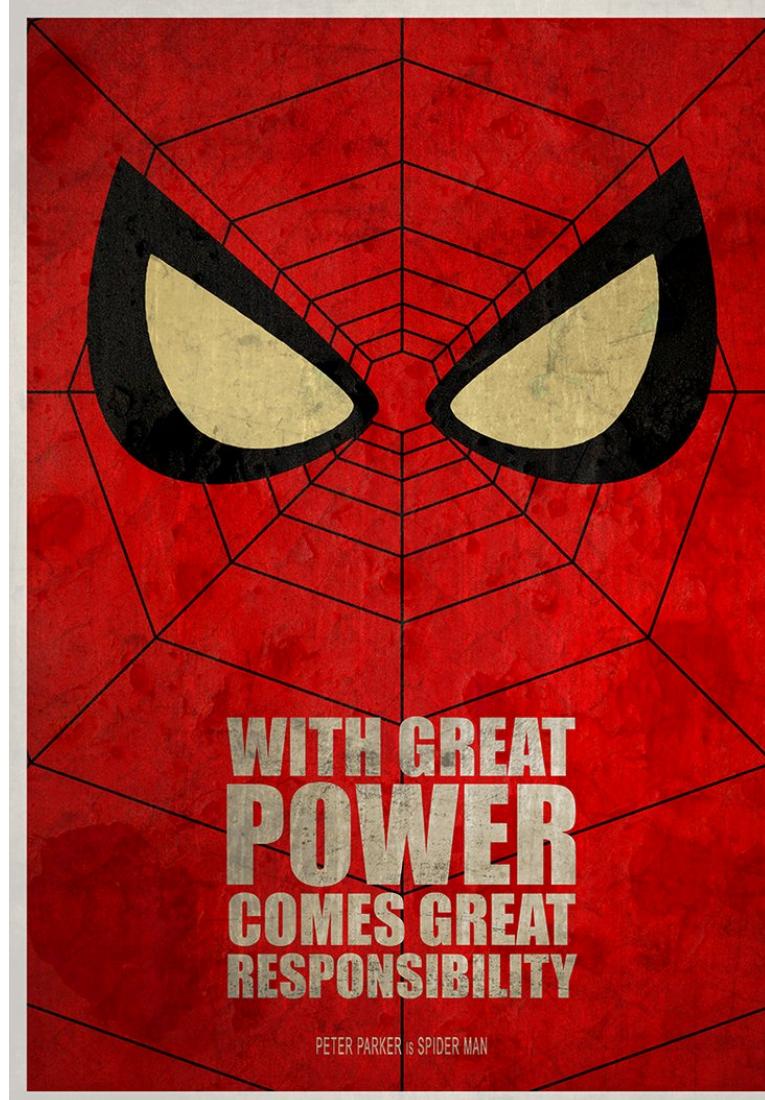
🔗 melix.github.io/blog

⌚ Joined January 2010

French guy,
naturally.

He's the one to
blame, BTW.

CONCLUSIONS



1. Write readable code
 2. Comment neat tricks
 3. Sometimes it is just a
 4. Use static code analy
 5. Rtfm
-
6. Don't code like my brother



We have just started!
(may end up in proper uniform)

Puzzlers? Gotchas?

- puzzlers jfrog.com

- Groovypuzzlers

@

JFROG ALWAYS PAYS ITS DEBTS

 **Deigote**
@deigote

As promised by @NoamTenne, @jfrog pays its debts :-D t-shirt received for sending them a #groovylang puzzle. Thanks!

[Reply](#) [Retweeted](#) [Favorited](#) [More](#)



 **Iván López**
@ilopmar

I've received an amazing t-shirt from @jfrog for sending them a #Groovylang puzzler. Thank you @NoamTenne :-)

[Reply](#) [Retweeted](#) [Favorite](#) [More](#)



Positive feedback?

Praise us on twitter

#groovypuzzlers

- @Groovypuzzlers
- @yoav_
- @baruch

Negative feedback?

> /dev/null

NO, THANK YOU!

