The Epic Groovy Puzzlers Season II: The Revenge of the Parentheses

CREATE
THE FUTURE

JavaOne
October 25–29, 2015
San Francisco
#javaone

Frogs ❤️ Groovy

⭐ Groovy   Learn   Documentation   **Download**   Community   Ecosystem   Socialize   🔍

Download Groovy

Distributions

Through GVM

From your build tools

Other ways to get Groovy

System requirements

☁ Download

📝 Improve this doc

In this download area, you will be able to download the distribution (binary and source), the Windows installer (for some of the versions) and the documentation for Groovy.

⬇ Download

All the downloads are hosted in Bintray's Groovy repository.

Registering on Bintray allows you to rate, review, and register for new version notifications.

Frogs ❤️ Groovy

⭐ Distibuting G&G from Bintray

⭐ Developing G&G with Artifactory

⭐ 90% of Bintray is G&G

⭐ Artifactory plugins are Groovy

⭐ Gradle and Grails plugins on Bintray

First Season Was a Blast!

Emerging Languages
The Epic Groovy Puzzlers S02:
The Revenge of the Parentheses

Golden Gate
6/7/8

BTW,

1. TWO ENTERTAINING GUYS ON STAGE

Puzzlers Squad!

Fred Simon, JFrog Chief Architect @freddy33

Baruch (a.k.a. JBaruch), JFrog Dev Advocate @jbaruch

1. TWO ENTERTAINING GUYS ON STAGE
2. FUNNY PUZZLING QUESTIONS
3. YOU THINK AND VOTE
4. AWESOME GROOVY T-SHIRTS FLY IN THE AIR
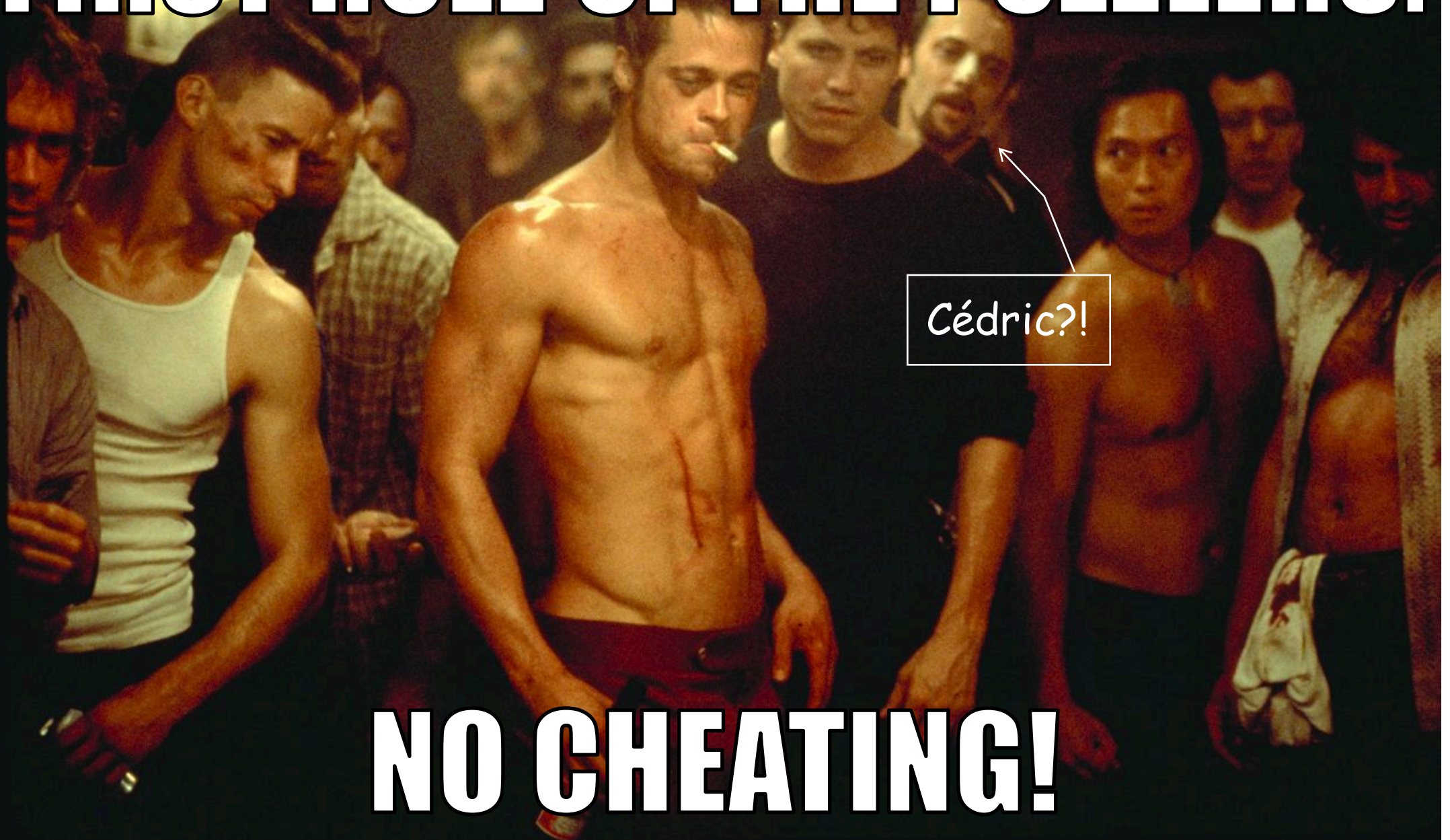5. OFFICIAL TWITTER HANDLE!
#GROOVYPUZZLERS

FIRST RULE OF THE PUZZLERS:

NO CHEATING!
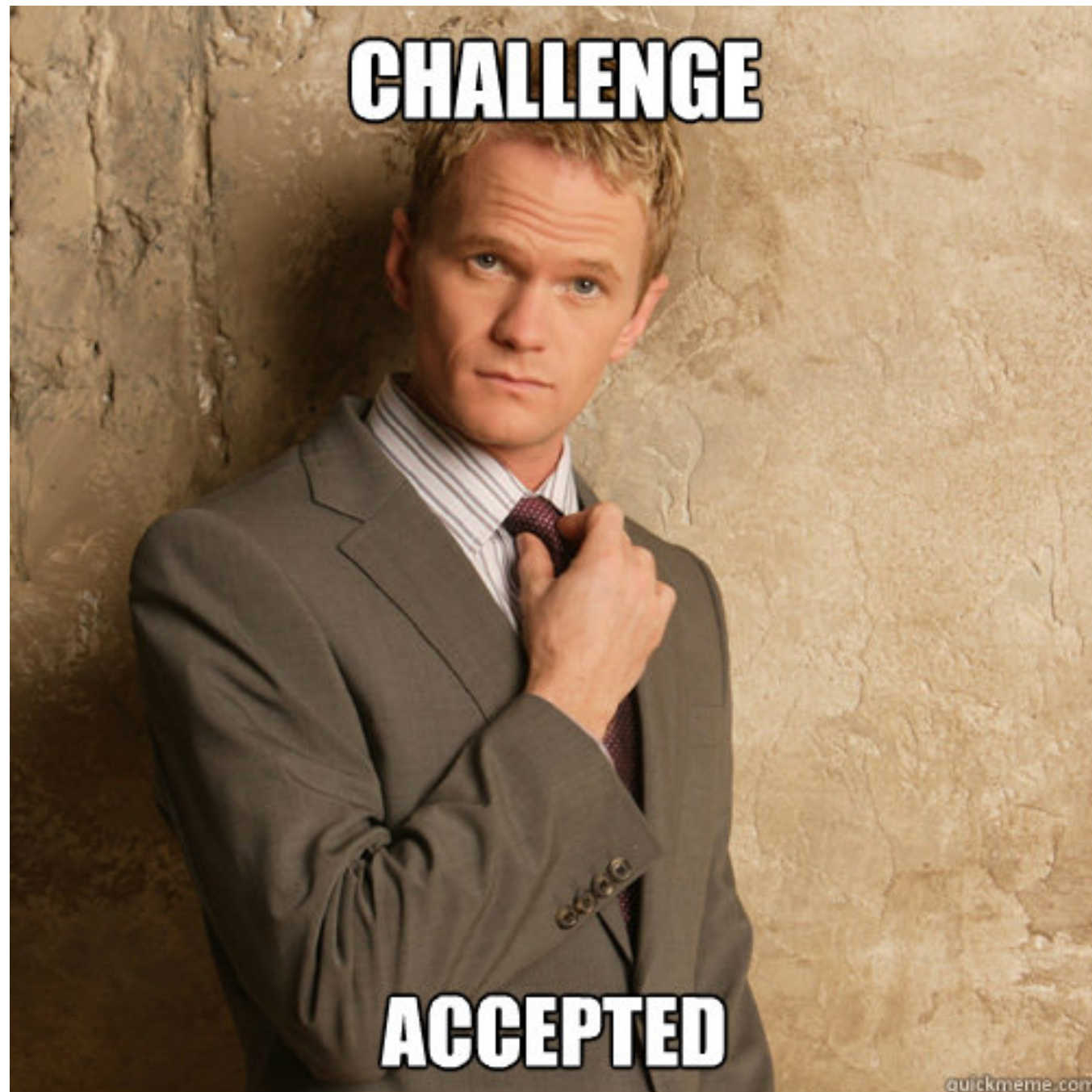
All works (or doesn't work) in Groovy 2.4.4

```
'a'..'z'.each { println it }
```

A.
a
b
c
.
.
z

B. NoSuchMethodError

C.

D. Won't run
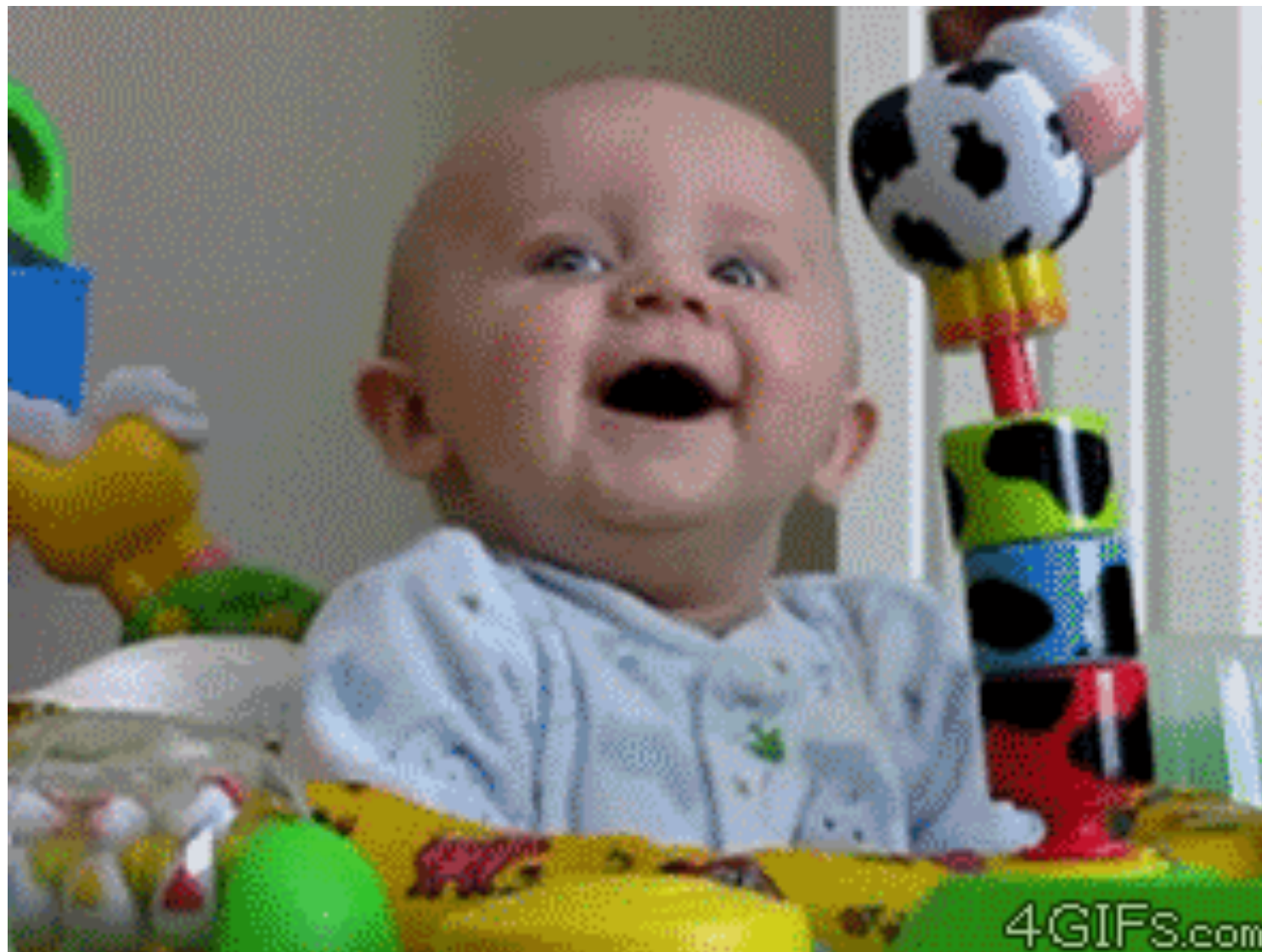
"All problems in computer science can be solved by another pair of parentheses"

John McCarthy, the inventor of LISP

And the t-shirt goes to...

Ken Kousen
@kenkousen FOLLOWS YOU

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

Marlborough, CT

kousenit.com

Joined August 2008

I'LL BE BACH

How Many Bachs?

```groovy
def back = 'back'
def quotes = ["I'll be $back",
"I'll be ${-> back}",
"I'll be ${back}",
"I'll be "+back]
println quotes
back = 'Bach'
println quotes
```

A. No Bachs

B. One Bach

C. Two Bachs

D. Three Bachs

Only closures are evaluated at runtime
Others are inlined

Only closures are evaluated at runtime
Others are inlined

```
def back = 'back'
def quotes = ["I'll be $back",
"I'll be ${-> back}",
"I'll be ${back}",
"I'll be "+back]
```

Only closures are evaluated at runtime
Others are inlined

```groovy
def back = 'back'
def quotes = ["I'll be $back",
"I'll be ${-> back}",
"I'll be ${back}",
"I'll be "+back]
```

That is the only closure

And the t-shirt goes to...

Cédric Champeau
@CedricChampeau FOLLOWS YOU

Software Engineer @Gradleware.
Conference speaker. Introvert
(carlkingdom.com/10-myths-abou
Wrote the static compiler for
#groovylang.

📍 St Hilaire de Loulay, France
🔗 melix.github.io/blog
🕐 Joined January 2010

InfoQ

En | 中文 | 日本 | Fr | Br

1,527,766 Aug unique visitors

Development   Architecture & Design   Process & Practices   Operations & Infrastructure   Enterprise Architecture

QCon
San Francisco 2015, Nov 16 - 20
London 2016, Mar 07 - 11
New York 2016, Jun 13 - 17

Plugging the Users in - Extend Your Application with Pluggable Groovy DSL

Recorded at:

springone2GX

by **Baruch Sadogursky** on Mar 12, 2014 | Discuss

Share | 🔲 f dz 🐦 reddit del ✉   🔖 My Reading List   🔖 Read later

View Presentation 🔲 ▯ 🖥

```
A problem has been detected and Windows has been shut down to prevent damage
to your computer.

DRIVER_POWER_STATE_FAILURE

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any Windows updates you might need.

If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use Safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.

Technical information:

*** STOP: 0x0000009F (0x0000000000000003,0xFFFFFA800FF75440,0xFFFFF80000B9C3D8,0
xFFFFFA8010AB1270)


Collecting data for crash dump ...
Initializing disk for crash dump ...
Beginning dump of physical memory.
Dumping physical memory to disk: 5
```

⏸ ▮▮   00:32:20   🔊 ⛶

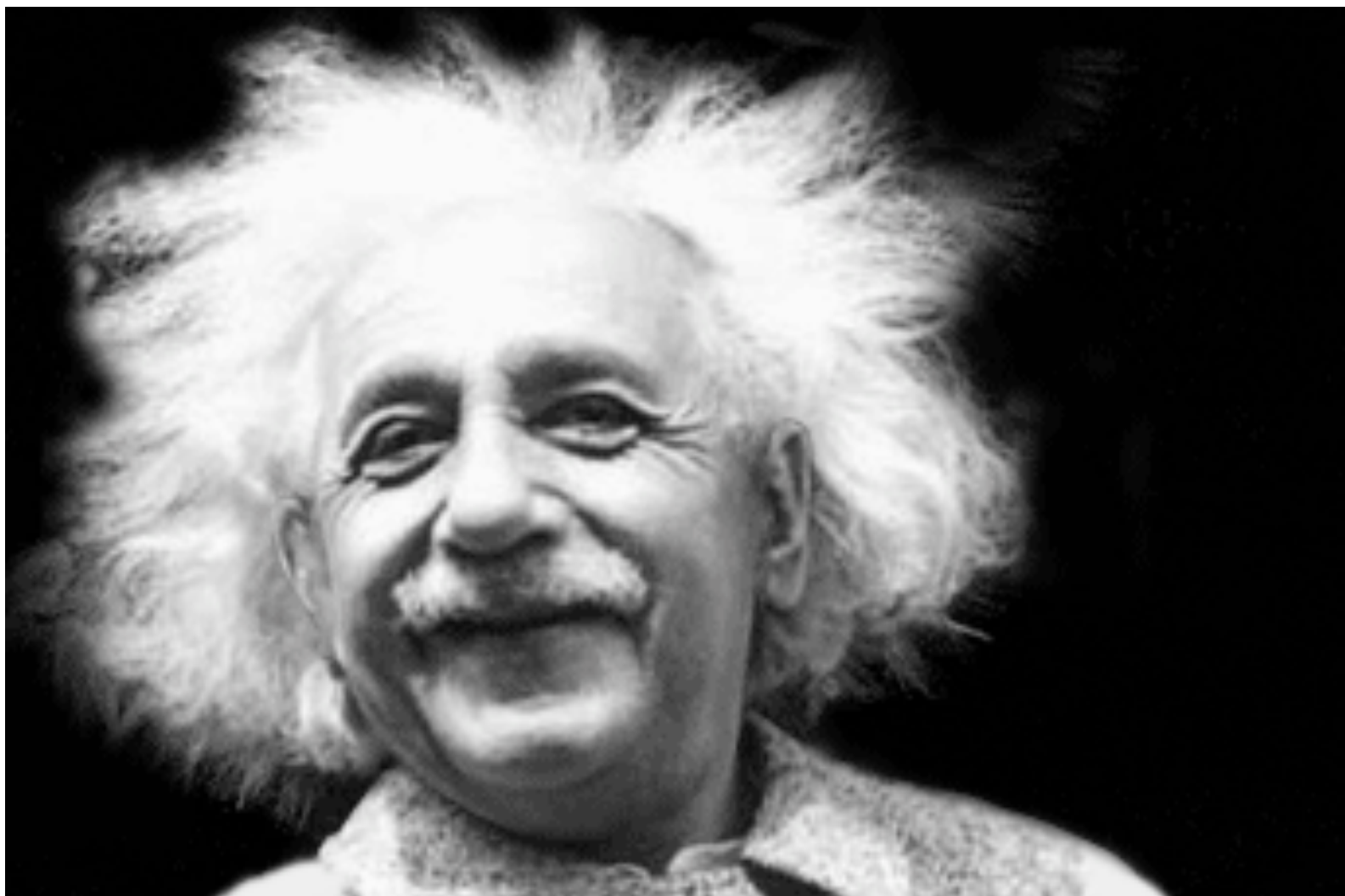Download   **MP3** | **Slides** | **Android app**   01:17:30

How one prints 666?

```
A. def beast = '6' * Math.PI
B. def beast = '6' * '3'
C. def beast = '667' - 1
D. def beast = '6' + '6' + 0 + 6
```

```
def beast = '6' * Math.PI
```

```
def beast = '6' * 3.1415926…
```

📖 **groovy** / **groovy-core**

👁 Watch ▾   106   ★ Unstar   1,233   ⑂ Fork   374

Groovy programming language http://www.groovy-lang.org

| 🕙 **12,326** commits | ⑂ **35** branches | 🏷 **149** releases | 👥 **129** contributors |

↻   ⑂ branch: **master** ▾   **groovy-core** / +   ☰

Merge pull request **#647** from christoph-frick/master   ...

🟪 **PascalSchumacher** authored 6 hours ago   latest commit 49c533de71 📋

📁 benchmark	minor refactor: remove some checkstyle warnings	8 months ago
📁 buildSrc	minor refactor: remove some checkstyle warnings	8 months ago
📁 config	GROOVY-3457: Preparing for addition of new StreamingTeamplateEngine	7 months ago
📁 gradle	Disable license and japicmp plugin since they still conflict with Art…	9 hours ago
📁 lib	Raw modifications to run Groovy on Android	10 months ago
📁 security	GROOVY-5305: Update dependencies (hsqldb)	3 years ago
📁 src	Merge pull request #647 from christoph-frick/master	6 hours ago
📁 subprojects	Documentation: add section on StreamingTemplateEngine	2 days ago
📁 xdocs/images	Remove the Maven 1 build files since we have now officially moved to …	8 years ago
📄 .gitignore	Tests that fail when using **@Grab** with Extension Modules	3 months ago

<> Code

⑂ Pull requests   32

⋀⋁ Pulse

📊 Graphs

HTTPS clone URL

https://github.com/g   📋

You can clone with HTTPS, SSH,
or Subversion. ⑦

🖥 **Clone in Desktop**

☁ **Download ZIP**

```java
/**
 * Repeat a String a certain number of times.
 *
 * @param self   a String to be repeated
 * @param factor the number of times the String should be repeated
 * @return a String composed of a repetition
 * @throws IllegalArgumentException if the number of repetitions is &lt; 0
 * @since 1.0
 */
public static String multiply(String self, Number factor) {
    int size = factor.intValue();

    …
}
```

```
def beast = '6' * 3.1415926…
```

```
def beast = '6' * 3
```

Pop Quiz!

```
B. def beast = '6' * '3'
C. def beast = '667' - 1
D. def beast = '6' + '6' + 0 + 6
```

Pop Quiz!

```
B. def beast = '6' * '3'
C. def beast = '667' - 1
D. def beast = '6' + '6' + (0 + 6)
```

And the t-shirt goes to...

Lari Hotari
@lhotari FOLLOWS YOU

Software Engineer at @Gradleware,
#Grailsfw core contributor, Software
Craftsman, #DDDesign & #LeanStartup
practitioner, #InfoSec paranoid, Seeking
simplicity

Ontario, Canada

github.com/lhotari

Joined April 2009

```
class THERE_CAN_BE_ONLY_ONE { }

class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class clazz = THERE_CAN_BE_ONLY_ONE
        return clazz.newInstance()
    }

}

println new MacLeod().THERE_CAN_BE_ONLY_ONE
```

```groovy
class THERE_CAN_BE_ONLY_ONE { }

class MacLeod {

    THERE_CAN_BE_ONLY_ONE
getTHERE_CAN_BE_ONLY_ONE() {
        Class clazz = THERE_CAN_BE_ONLY_ONE
        return clazz.newInstance()
    }

}

println new MacLeod().THERE_CAN_BE_ONLY_ONE
```

A. Won't start
B. No such property: THERE_CAN_BE_ONLY_ONE for class: MacLeod
C. THERE_CAN_BE_ONLY_ONE@3d74bf60
D. Another exception

A.MultipleCompilationErrorsException

B.Stack Groovy flowError

C.NullPointerException

D.Yet Another Exception

```
class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class clazz = THERE_CAN_BE_ONLY_ONE
        return clazz.newInstance()
    }

}
```

```
class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class clazz = getTHERE_CAN_BE_ONLY_ONE()
        return clazz.newInstance()
    }

}
```

Let's Fix It!

```
Class<THERE_CAN_BE_ONLY_ONE> clazz = THERE_CAN_BE_ONLY_ONE.class
```

```
class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class<THERE_CAN_BE_ONLY_ONE> clazz = THERE_CAN_BE_ONLY_ONE.class
        return clazz.newInstance()
    }

}
```

```
class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class<THERE_CAN_BE_ONLY_ONE> clazz = getTHERE_CAN_BE_ONLY_ONE().class
        return clazz.newInstance()
    }

}
```

Let's Fix It!

```
Class<THERE_CAN_BE_ONLY_ONE> clazz = (THERE_CAN_BE_ONLY_ONE as Class)
```

```
class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class<THERE_CAN_BE_ONLY_ONE> clazz = (THERE_CAN_BE_ONLY_ONE as Class)
        return clazz.newInstance()
    }

}
```

```
class MacLeod {

    THERE_CAN_BE_ONLY_ONE getTHERE_CAN_BE_ONLY_ONE() {
        Class<THERE_CAN_BE_ONLY_ONE> clazz = (getTHERE_CAN_BE_ONLY_ONE() as Class)
        return clazz.newInstance()
    }

}
```

Let's Fix It!

And the t-shirt goes to...
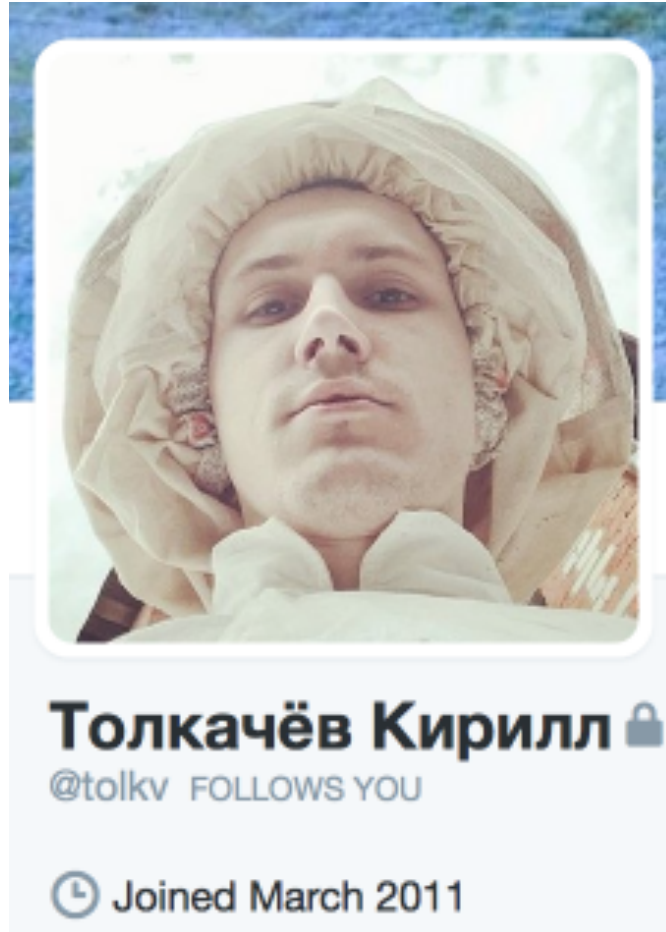
MISSED ME?

```groovy
@groovy.transform.InheritConstructors
class TreaayeMap extends HashMap {
}

TreaayeMap a = [5]
TreaayeMap b = [6]

println "${a.getClass()} ${a.equals(b)}"
```

A. class HashMap true

B. class TreaayeMap false

C. class TreaayeMap true

D. class HashMap false

List and map constructors

In addition to the assignment rules above, if an assignment is deemed invalid, in type checked mode, a *list* literal or a *map* literal A can be assigned to a variable of type T if:

- the assignment is a variable declaration and A is a list literal and T has a constructor whose parameters match the types of the elements in the list literal

- the assignment is a variable declaration and A is a map literal and T has a no-arg constructor and a property for each of the map keys

```
TreaayeMap a = [5]
```

List and map constructors

In addition to the assignment rules above, if an assignment is deemed invalid, in type checked mode, a *list* literal or a *map* literal A can be assigned to a variable of type T if:

- the assignment is a variable declaration and A is a list literal and T has a constructor whose parameters match the types of the elements in the list literal

- the assignment is a variable declaration and A is a map literal and T has a no-arg constructor and a property for each of the map keys

```
TreaayeMap a = [5]
```

List and map constructors

In addition to the assignment rules above, if an assignment is deemed invalid, in type checked mode, a *list* literal or a *map* literal A can be assigned to a variable of type T if:

- the assignment is a variable declaration and A is a list literal and T has a constructor whose parameters match the types of the elements in the list literal

- the assignment is a variable declaration and A is a map literal and T has a no-arg constructor and a property for each of the map keys

```
TreaayeMap a = [5]
```

List and map constructors

In addition to the assignment rules above, if an assignment is deemed invalid, in type checked mode, a *list* literal or a *map* literal A can be assigned to a variable of type T if:

- the assignment is a variable declaration and A is a list literal and T has a constructor whose parameters match the types of the elements in the list literal

- the assignment is a variable declaration and A is a map literal and T has a no-arg constructor and a property for each of the map keys

```
TreaayeMap a = [5]
```

```
@groovy.transform.InheritConstructors
class TreaayeMap extends HashMap {
}
```

List and map constructors

In addition to the assignment rules above, if an assignment is deemed invalid, in type checked mode, a *list* literal or a *map* literal A can be assigned to a variable of type T if:

- the assignment is a variable declaration and A is a list literal and T has a constructor whose parameters match the types of the elements in the list literal

- the assignment is a variable declaration and A is a map literal and T has a no-arg constructor and a property for each of the map keys

```java
/**
 * Constructs an empty <tt>HashMap</tt> with the specified initial
 * capacity and the default load factor (0.75).
 *
 * @param  initialCapacity the initial capacity.
 * @throws IllegalArgumentException if the initial capacity is negative.
 */
public HashMap(int initialCapacity)
```

```groovy
@groovy.transform.InheritConstructors
class TreaayeMap extends HashMap {
}
```

```groovy
TreaayeMap a = [5]
```

List and map constructors

In addition to the assignment rules above, if an assignment is deemed invalid, in type checked mode, a *list* literal or a *map* literal
A can be assigned to a variable of type T if:

- the assignment is a variable declaration and A is a list literal and T has a constructor whose parameters match the types of the elements in the list literal

- the assignment is a variable declaration and A is a map literal and T has a no-arg constructor and a property for each of the map keys

equals() doesn't care about capacity, mappings only

```
*/
public boolean equals(Object o) {
    if (o == this)
        return tru
    if (!(o instan
        return fal
    Map<?,?> m = (
    if (m.size() !
        return fal
    try {
        Iterator<E
```

Documentation for equals(Object)

< 1.8 >

java.util.AbstractMap

public boolean **equals**(*@Nullable* java.lang.Object o)

Compares the specified object with this map for equality. Returns `true` if the given object is also a map and the two maps represent the same mappings. More formally, two maps m1 and m2 represent the same mappings if m1.entrySet().equals(m2.entrySet()). This ensures that the equals method works properly across different implementations of the Map interface.

And the t-shirt goes to...

Sergey Tselovalnikov
@SerCeMan

Love IT, love other interesting things.
Develop in Java, Groovy, C++. JUG.EKB
(jugekb.ru) creator.

📍 Saint-Petersburg
🕐 Joined November 2010

```
ArrayList<String> expendables = ['Arnold', 'Chuck', 'Sylvester']
def expendable = //someone from the list

for(String hero in expendables) {
    if(hero == expendable){
        expendables.remove(hero)
    }
}

println expendables
```

Which one won't cause a ConcurrentModificationException?

```groovy
ArrayList<String> expendables = ['Arnold', 'Chuck', 'Sylvester']
def expendable = //someone from the list

for(String hero in expendables) {
    if(hero == expendable){
        expendables.remove(hero)
    }
}

println expendables
```
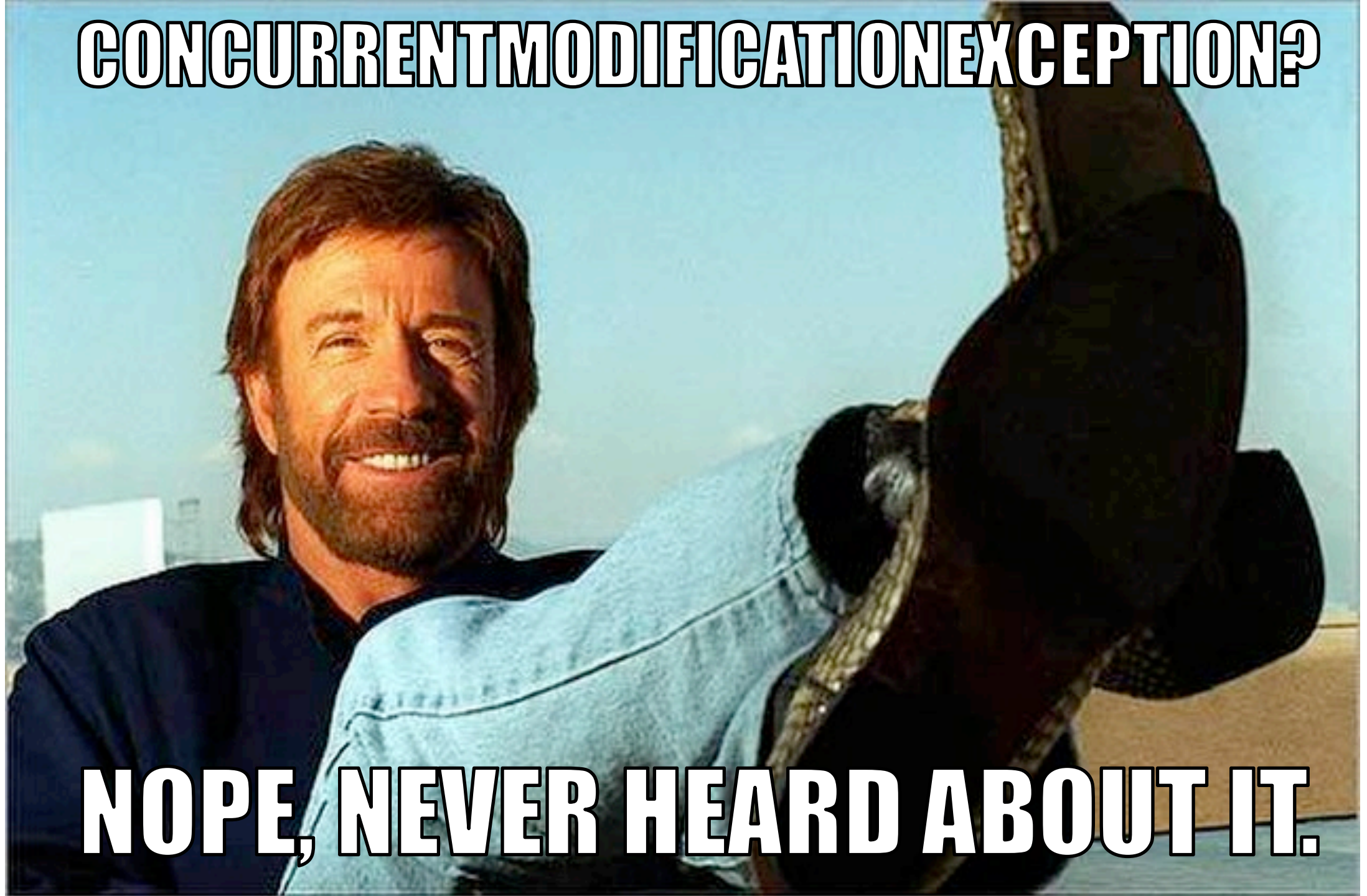
A. Can't avoid CME

B. Arnold

C.

D. Sylvester

You have been warned.

```
public class ConcurrentModificationException
extends RuntimeException
```

This exception may be thrown by methods that have detected concurrent modification of an object when such modification is not permissible.

For example, it is not generally permissible for one thread to modify a Collection while another thread is iterating over it. In general, the results of the iteration are undefined under these circumstances. Some Iterator implementations (including those of all the general purpose collection implementations provided by the JRE) may choose to throw this exception if this behavior is detected. Iterators that do this are known as *fail-fast* iterators, as they fail quickly and cleanly, rather that risking arbitrary, non-deterministic behavior at an undetermined time in the future.

Note that this exception does not always indicate that an object has been concurrently modified by a *different* thread. If a single thread issues a sequence of method invocations that violates the contract of an object, the object may throw this exception. For example, if a thread modifies a collection directly while it is iterating over the collection with a fail-fast iterator, the iterator will throw this exception.

Note that fail-fast behavior cannot be guaranteed as it is, generally speaking, impossible to make any hard guarantees in the presence of unsynchronized concurrent modification. Fail-fast operations throw `ConcurrentModificationException` on a best-effort basis. Therefore, it would be wrong to write a program that depended on this exception for its correctness: `ConcurrentModificationException` *should be used only to detect bugs.*

Let's decompile this, baby!

```java
List expendables = Arrays.asList(new String[]{"Arnold", "Chuck", "Sylvester"});
String expendable = "Chuck";
Iterator iterator = expendables.iterator();

while(iterator.hasNext()) {
    String hero = (String)iterator.next();
    if(hero.equals(expendable)) {
        expendables.remove(hero);
    }
}
```

```java
while(iterator.hasNext()) {
    String hero = (String)iterator.next();
    if(hero.equals(expendable)) {
//        expendables.remove(hero);
    }
}
```

```java
while(iterator.hasNext()) {
    String hero = (String)iterator.next();
    if(hero.equals(expendable)) {
//        expendables.remove(hero);
    }
}

public E next() {
    checkForComodification();
    …
    cursor = i + 1;
}

public boolean hasNext() {
    return cursor != size();
}
```

Modifications are only checked in the next cycle

Getting ready for hasNext() check in the next cycle

Exit on last element +1 == size()

```java
while(iterator.hasNext()) {
    String hero = (String)iterator.next();
    if(hero.equals(expendable)) {
//        expendables.remove(hero);
    }
}

public E next() {
    checkForComodification();

    …
    cursor = i + 1;
}

public boolean hasNext() {
    return cursor != size();
}
```

After Sylvester cursor is 3

And size is 3 as well

All good. Now let's mess with it.

```java
while(iterator.hasNext()) {
    String hero = (String)iterator.next();
    if(hero.equals(expendable)) {
        expendables.remove(hero);
    }
}

public E next() {
    checkForComodification();
    …
    cursor = i + 1;
}

public boolean hasNext() {
    return cursor != size();
}
```

After Chuck the cursor is 2

Then we remove the element

And now the size now is 2!

It won't get to the next() to run checkForComodification!

Hey, what about me?!

```java
while(iterator.hasNext()) {
    String hero = (String)iterator.next();
    if(hero.equals(expendable)) {
        expendables.remove(hero);
    }
}

public E next() {
    checkForComodification();
    …
    cursor = i + 1;
}

public boolean hasNext() {
    return cursor != size();
}
```

After Sylvester the cursor is 3

Then we remove the element

But the size now is 2!

It will go to another loop and fail on checkForComodification!

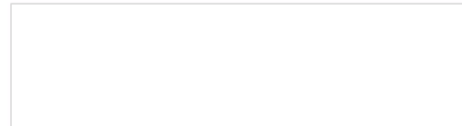# And the t-shirt goes to...

Evgeny Borisov
Senior Java Consultant at Trainologic

Israel | Education Management

Current	Democracy Startup, Trainologic, JFrog Ltd
Previous	IDI Israel, AlphaCSP
Education	Polytechnic

```
def numbers = [[2, 3, 5][2, 4, 8][42, 73, Integer.MAX_VALUE, 0]]
println numbers
```

```
def numbers = [[2, 3, 5],[2, 4, 8],[42, 73, Integer.MAX_VALUE, 0]]
println numbers
```

A. [[2, 3, 5], [2, 4, 8], [42, 73, 2147483647, 0]]

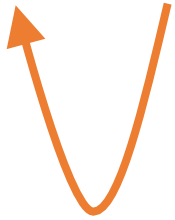B. Won't run

C. [[null, null, 5]]

D. null

```
def numbers = [[2, 3, 5][2, 4, 8][42, 73, Integer.MAX_VALUE, 0]]
println numbers
```

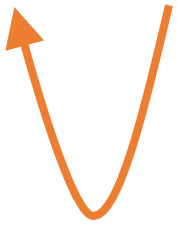**assert ['a','b','c'][0,2]==['a','c']**

[2, 3, 5][2, 4, 8]

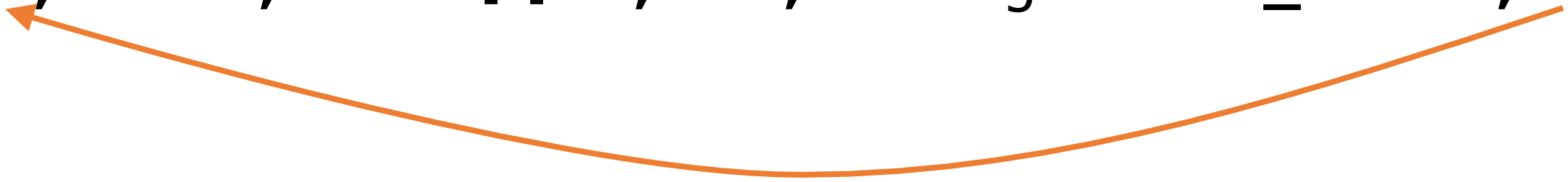[5, **null**, **null**][42, 73, Integer.MAX_VALUE, 0]

```
def numbers = [[2, 3, 5][2, 4, 8][42, 73, Integer.MAX_VALUE, 0]]
println numbers
```

[2, 3, 5][2, 4, 8]

[5, **null**, **null**][42, 73, Integer.MAX_VALUE, 0]

[[**null**, **null**, **null**, 5]]

No t-shirt for this guy ☹

groovy spock

share edit close flag

asked Apr 8 at 17:44

orbfish
2,157 ● 4 ● 22 ● 40

We used this awesome question as a puzzler in the second season of the #groovypuzzlers and want to send you a thank-you t-shirt. How can I reach you? – JBaruch Sep 13 at 4:49

add a comment

start a bounty

```groovy
String truth = 'false'
boolean groovyTruth = truth
println groovyTruth
```

```
String truth = 'false'
boolean groovyTruth = truth
println groovyTruth
```

A. false

B. true

C. ClassCastException

D. Startup error

§ Strings

Non-empty Strings, GStrings and CharSequences are coerced to true.

```groovy
assert 'a'
assert !''
def nonEmpty = 'a'
assert "$nonEmpty"
def empty = ''
assert !"$empty"
```

And the t-shirt goes to...

Andrey Hihlovskiy
@AndreyHihlovski FOLLOWS YOU

Software Engineer at startext GmbH.
Groovy, Gradle, Spring Framework. V/I is
futile. github.com/akhikhl

Bonn, Germany

akhikhl.org
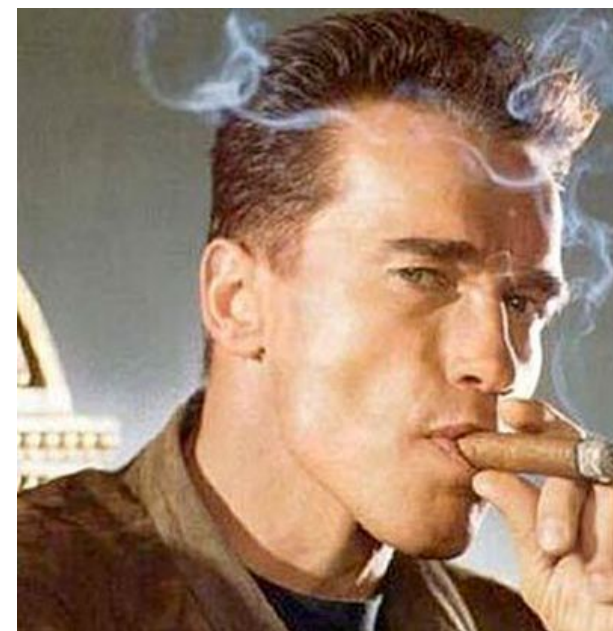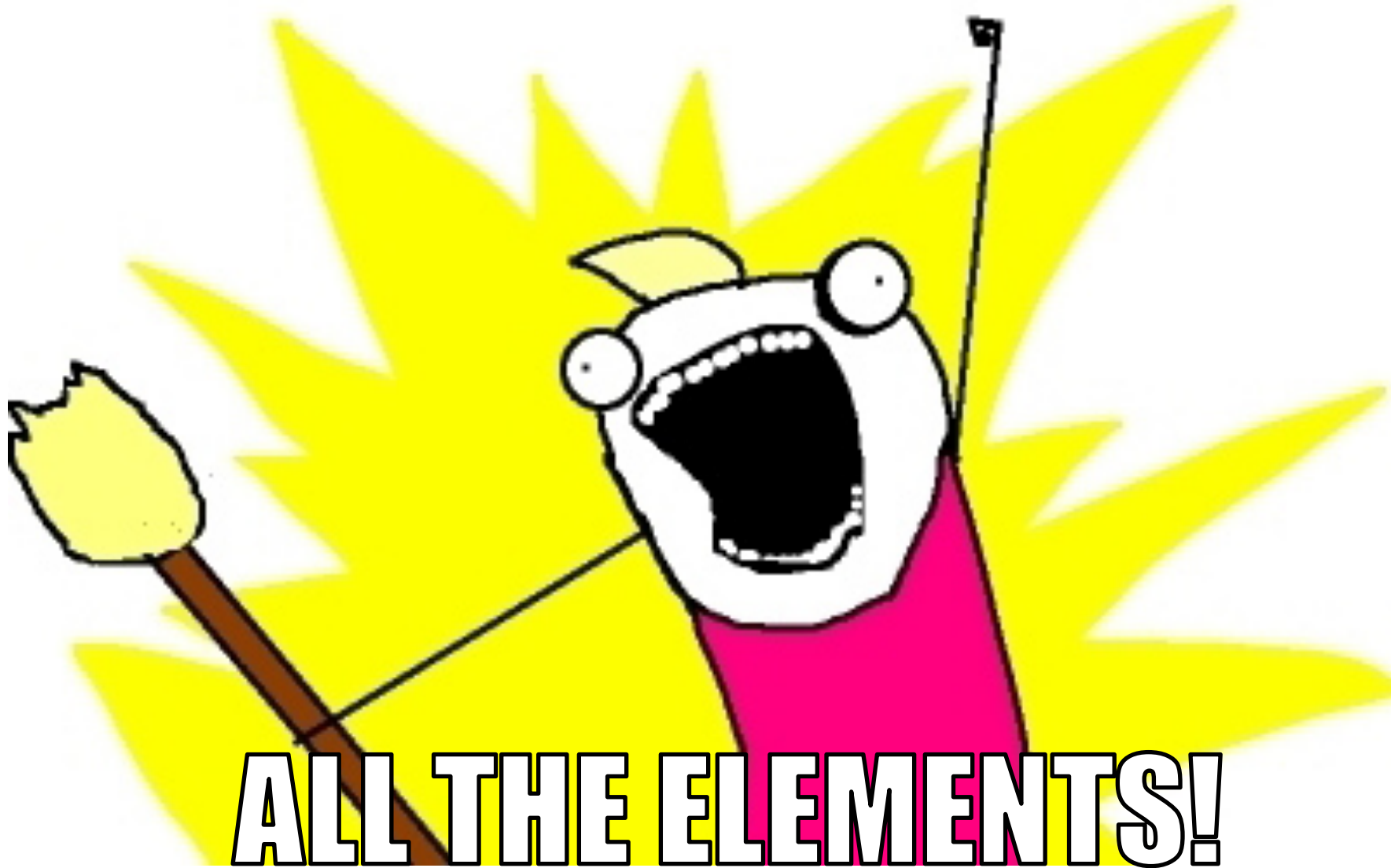
COLLECT ALL THE ELEMENTS!

```
class Kitty { def fur }
def kitties = [new Kitty(fur: 'soft'),new Kitty(fur: 'warm'),new Kitty(fur: 'purr') ]

println kitties.collect { it.fur }
println kitties*.fur
println kitties.fur
```

How many of the printed lines will be the same?

A. All different

B. 2 similar, one different

C. All the same

D. Won't run

```
class Kitty { def fur }
def kitties

println kitties.collect { it.fur }
println kitties*.fur
println kitties.fur
```

How many of the printed lines will be the same?

A. All different

B. 2 similar, one different

C. All the same

D. Won't compile

```
class Kitty { def fur }
def kitties

println kitties.fur
```

```
class Kitty { def fur }
def kitties

println kitties.fur
```

```
class Kitty { def fur }
def kitties

println kitties.collect { it.fur  }
```

```
class Kitty { def fur }
def kitties

println kitties.collect { it.fur  }


public static Collection asCollection(Object value) {
    if (value == null) {
        return Collections.EMPTY_LIST;
    }
  …
}
```

```
class Kitty { def fur }
def kitties


println kitties*.fur
```

```
class Kitty { def fur }
def kitties


println kitties*.fur
```

The spread operator is null-safe, meaning that if an element of the collection is null, it will return null instead of throwing a `NullPointerException`:

```
class Kitty { def fur }
def kitties

println kitties*.fur
```

The spread operator is null-safe, meaning that if an element of the collection is nu[...]a
`NullPointerException`:

Consistency, yeah.

```
[]
null
Caught: java.lang.NullPointerException:
Cannot get property 'fur' on null object
```

And the t-shirt goes to...

Dan Tanner
@edgescope

mislabeled

📍 Minneapolis MN US

🕐 Joined April 2009

GEORGE ORWELLS

ANIMAL FARM

ALL ANIMALS ARE CREATED EQUAL
BUT SOME ARE MORE EQUAL THAN OTHERS

STARRING   MAURICE DENHAM   DIRECTORS   JOY BATCHELOR
NARRATION   GORDON HEATH   JOHN HALAS
STORY BY   GEORGE ORWELL   MUSIC BY   MATYAS SEIBER

HAZZAM.CO

assert 1L == 1
println 1L.equals(1)

A. Assertion failed

B. true

C. false

D. MissingMethodException

10. Behaviour of `==`

In Java `==` means equality of primitive types or identity for objects. In Groovy `==` translates to `a.compareTo(b)==0`, iff they are `Comparable`, and `a.equals(b)` otherwise. To check for identity, there is `is`. E.g. `a.is(b)`.

```java
/**
 * Compare two Numbers.  Equality (==) for numbers dispatches to this.
 *
 * @param left  a Number
 * @param right another Number to compare to
 * @return the comparison of both numbers
 * @since 1.0
 */
public static int compareTo(Number left, Number right) {
    /** @todo maybe a double dispatch thing to handle new large numbers? */
    return NumberMath.compareTo(left, right);
}
```

```java
/**
 * Compares this object to the specified object.  The result is
 * {@code true} if and only if the argument is not
 * {@code null} and is a {@code Long} object that
 * contains the same {@code long} value as this object.
 *
 * @param   obj   the object to compare with.
 * @return  {@code true} if the objects are the same;
 *          {@code false} otherwise.
 */
public boolean equals(Object obj) {
    if (obj instanceof Long) {
        return value == ((Long)obj).longValue();
    }
    return false;
}
```

And the t-shirt goes to...

Ron Dahlgren
@ScaleItRon

Tweets generally regarding Linux, D&D, SNES, PS4, JRPGs, SRPGs, Software, and science news. PGP - FB3E4542

📍 Mechanical Island
🔗 dahlgren.so
🕐 Joined April 2013

Conclusions

We keep going! (look at the awesome t-shirts!)

Puzzlers? Gotchas? Fetal position inducing behavior?

- puzzlers@jfrog.com
- @groovypuzzlers

Jfrog always pays its debts

POSITIVE FEEDBACK?
FILL THE FEEDBACK FORM!
PRAISE US ON TWITTER
GROOVYPUZZLERS
- @GROOVYPUZZLERS
- @GAMUSSA
- @JBARUCH

NEGATIVE FEEBACK?
/DEV/NULL

No, Thank you!