ATLASSIAN

ABOUT ME

From Sweden, Rock climber, Dev on the JSW integrations team

My Kotlin journey

About 9 months long

Rushed project, looking for speed

Saw Kotlin’s collections API

Didn’t stop to look at the performance

The Price of Readability

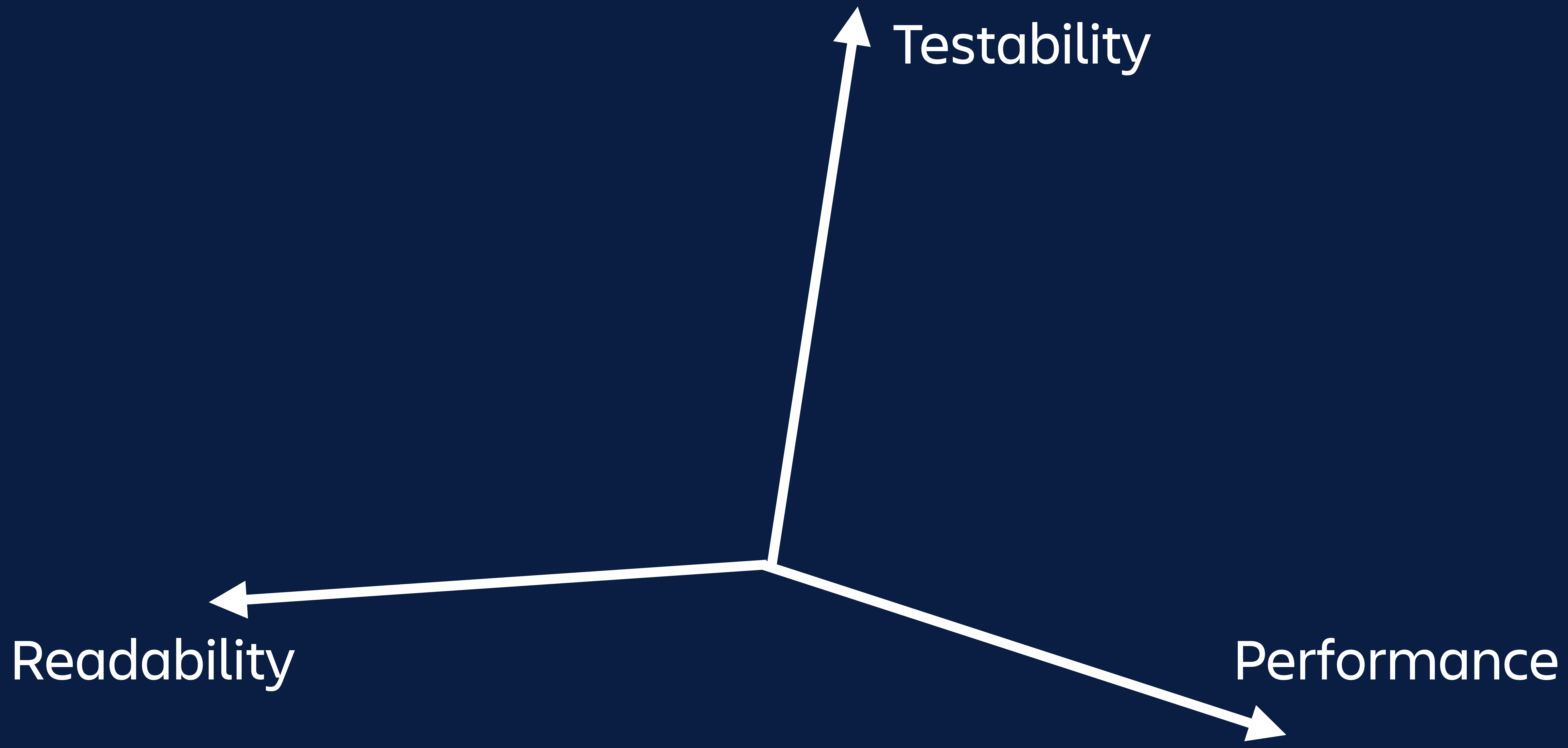
CHRISTIAN ROLF | SENIOR DEVELOPER | @CCROLF

READABILITY

Readable code is better

PERFORMANCE

Performant code is better



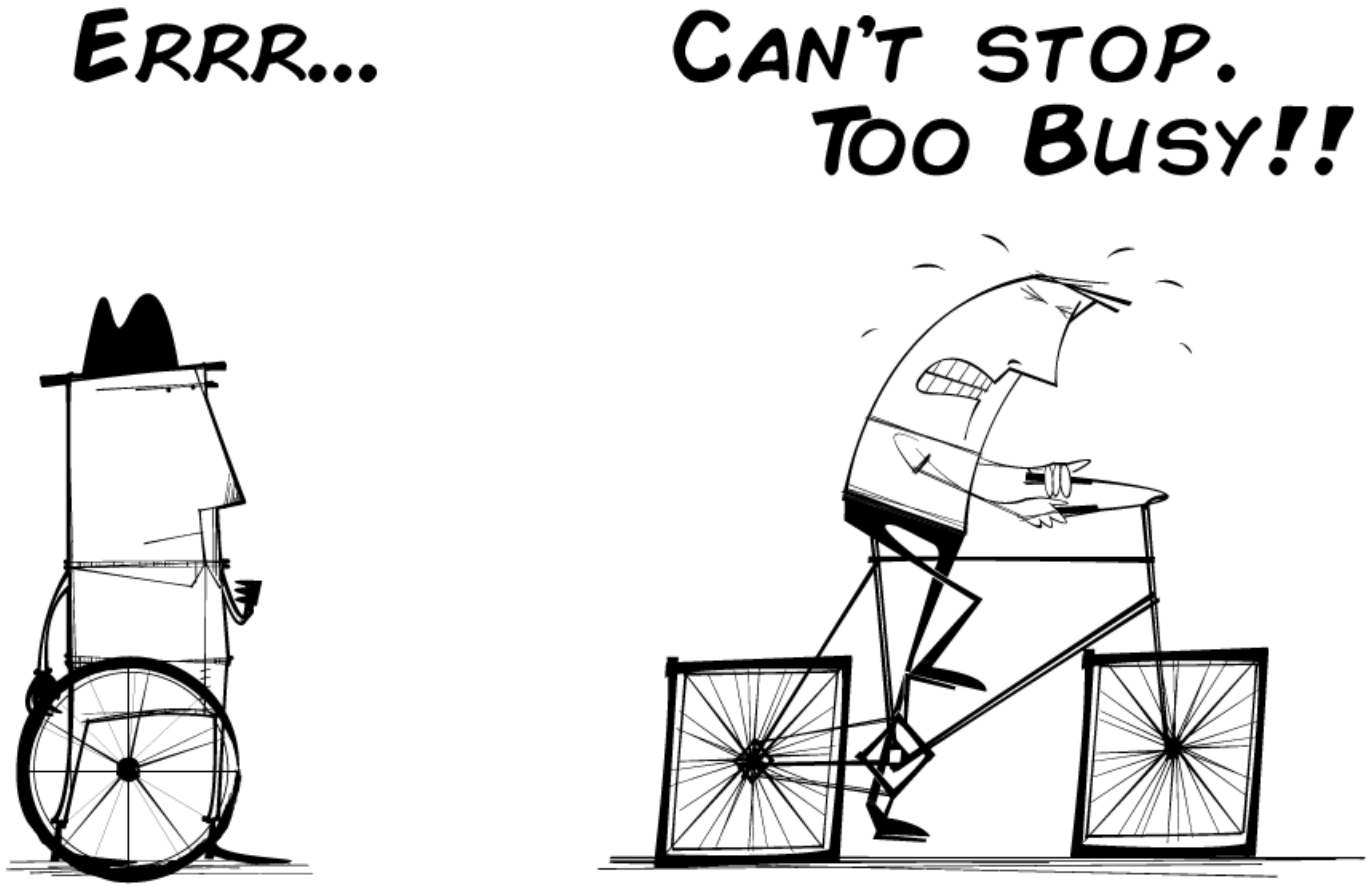
Testability

Readability

Performance

IS KOTLIN INTERESTING OR IS IT AWESOME?

Source (https://www.tofugu.com/japan/chindogu-japanese-inventions/)



TOO BUSY TO IMPROVE?

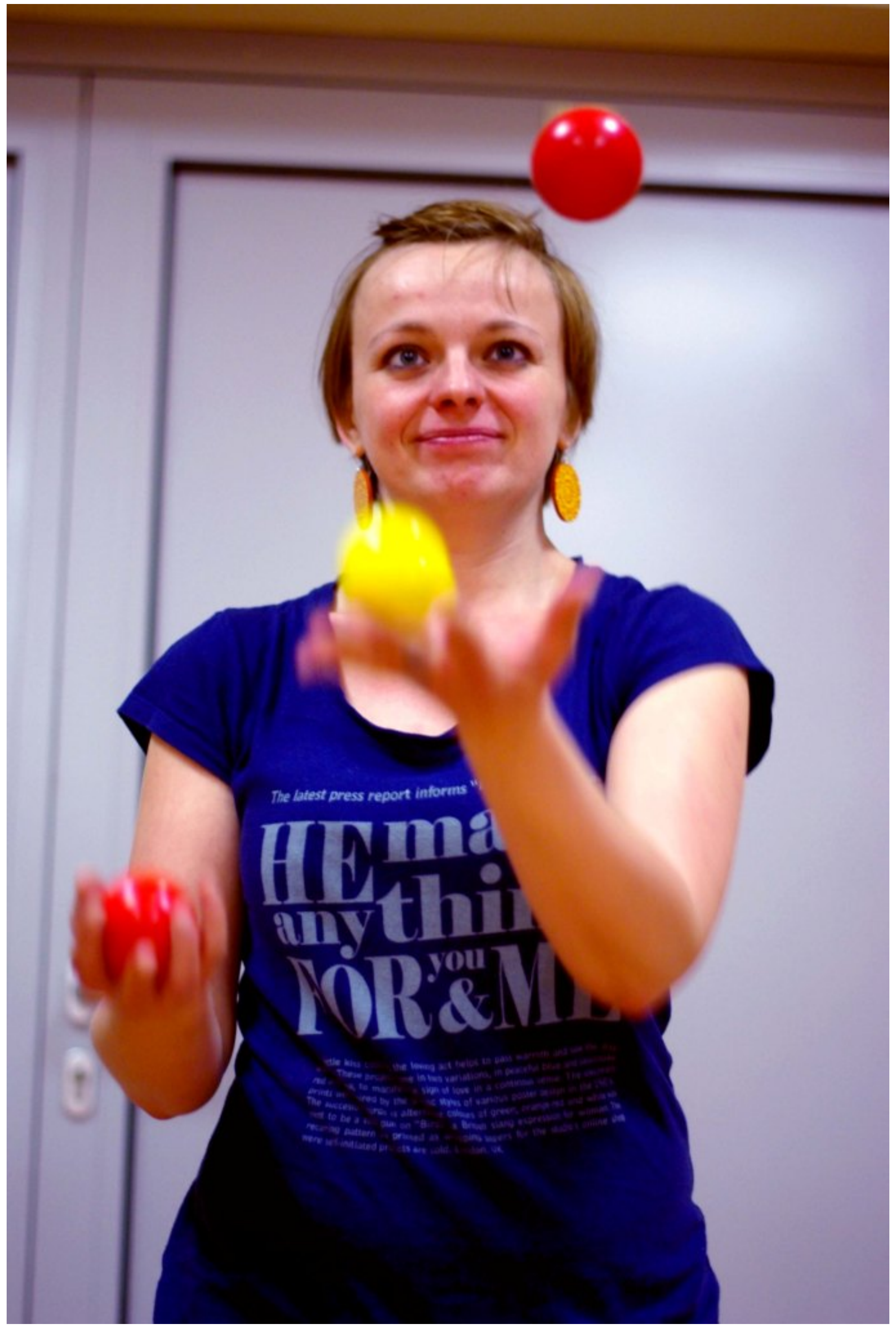
WorkCompass

Source (https://commons.wikimedia.org/wiki/File:Too_Busy_To_Improve_-_Performance_Management_-_Square_Wheels.png)

The performance problem

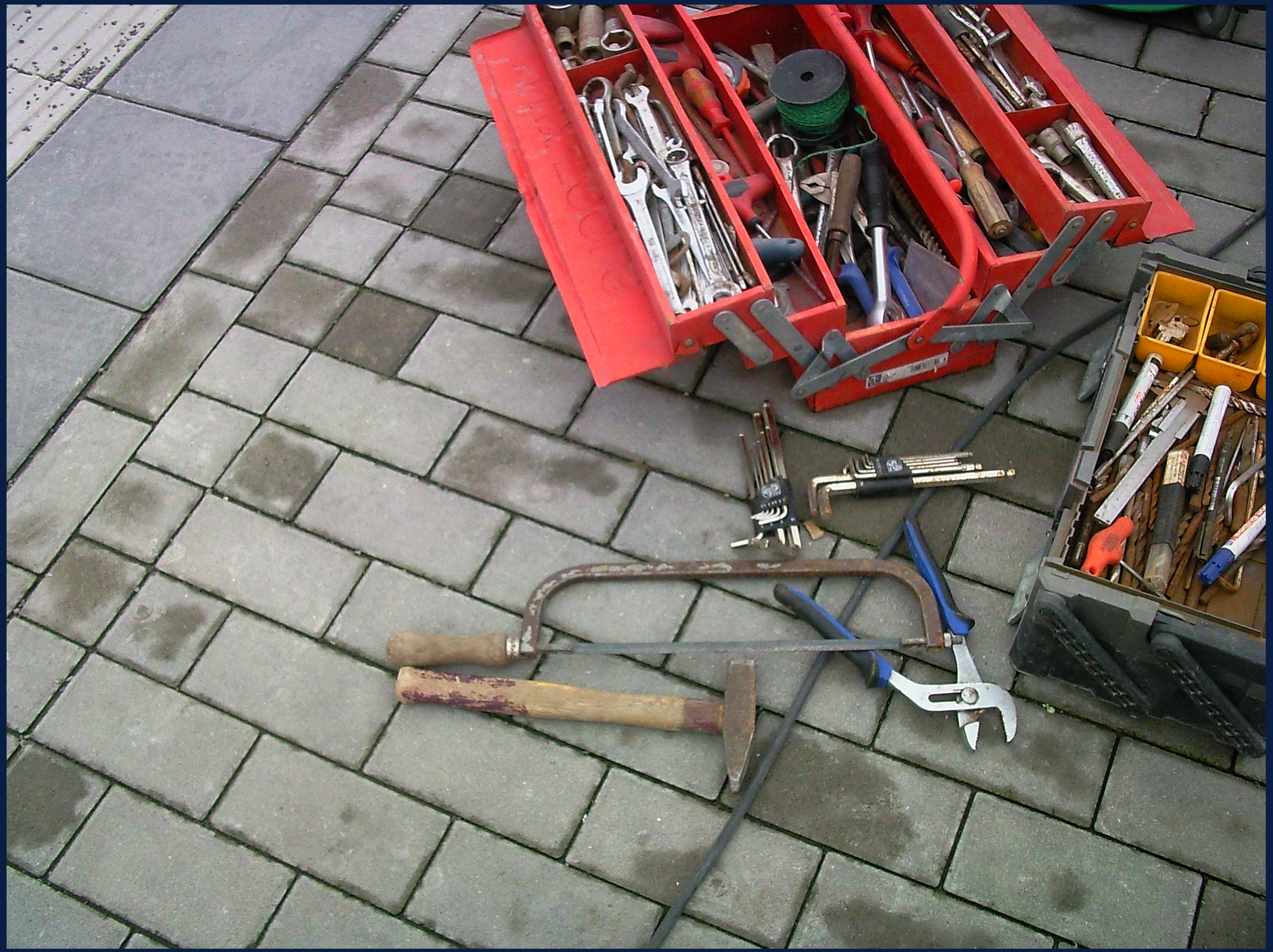
How many people know GraphQL?

REST



By mik Krakow - Pracownie 2012 - etiudy teatralne (21.04.2012), CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=31816249

GRAPHQL



By Erich Ferdinand from germany - (not my) toolbox, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=34359541

Trivial problem: Find min

```
{
  “foo” : 1,
  “bar” : null,
  “baz” : -1
}
```

```
fun finMin(json: Map<String, Int?>) =
    json.filter { it.value != null }
        .minWith(comparingInt{ it.value!! })!!
        .value!!
```

```
int findMin(Map<String, Integer> json) {
    return json.values().stream()
               .filter(Objects::nonNull)
               .min(comparingInt(it -> it))
               .get();
}
```

KOTLIN

320 ± 8 ops/ms

JAVA

780 ± 170 ops/ms

“

I guess Jira’s written in Kotlin then...

PROBLEM

Kotlin collection operators are eager, streams are lazy

```
fun finMin(json: Map<String, Int?>) =
    json.filter { it.value != null }
        .minWith(comparingInt{ it.value!! })!!
        .value!!
```

```
fun finMin(json: Map<String, Int?>) =
    json.values
        .minWith(comparingInt {
             it ?: Int.MAX_VALUE
        })!!
```

SMART KOTLIN

1830 ± 91
ops/ms

DUMB JAVA

780 ± 170
ops/ms

```
int findMin(Map<String, Integer> json) {
    return json.values().stream()
                .min(comparingInt(it ->
                    it == null ? MAX_VALUE: it
                ))
                .get();
}
```

SMART KOTLIN

1830 ± 91 ops/ms

SMART JAVA

1040 ± 100 ops/ms

> Performance doesn’t matter until it’s the only thing that matters

JED WESLEY-SMITH

When you care, care a lot!

```
fun finMin(json: Map<String, Int?>) =
    var min: Int = Int.MAX_VALUE
    for (it in json.values) {
        it?.let { min = min(min, it) }
    }
    return min
```

```
int findMin(Map<String, Integer> json) {
    int min = Integer.MAX_VALUE;
    for (Integer it : json.values()) {
        min = it == null ? min : min(min, it);
    }
    return min;
}
```

OPTIMIZED KOTLIN

2790 ± 210 ops/ms

OPTIMIZED JAVA

2740 ± 140 ops/ms

> But the test-case is just a loop!

EVERYONE, HERE, NOW

What about large, nested data structures?

GraphQL problem: find min

```
{
  “foo” : [
    {
      “a” : 1,
      “b” : null
    }
  ]
  “bar”: [ { “c” : null } , { “d” : -1 } ]
}
```

```
fun findMin(json:
    Map<String, List<Map<String, Int?>>>) =
    return json
              .flatMap { it.value }
              .flatMap { it.values }
              .filter { it != null }
              .minWith(comparingInt { it!! })!!
    }
```

```
int findMin(Map<String,
            List<Map<String, Integer>>> json){
    return json.values().stream()
               .flatMap(Collection::stream)
               .flatMap(it ->
                        it.values().stream())
               .filter(Objects::nonNull)
               .min(comparingInt(it -> it))
               .get();
}
```

DUMB KOTLIN

0.68 ± 0.08 ops/ms

DUMB JAVA

0.74 ± 0.05 ops/ms

9x

performance improvement we got before from optimization

```
fun findMin(json:
    Map<String, List<Map<String, Int?>>>) =
    var min: Int = Int.MAX_VALUE
    for (v in json.values) {
        for (value in v) {
            for (it in value.values) {
                it?.let { min = min(min, it) }
            }
        }
    }
    return min
}
```

OPTIMIZED KOTLIN

0.94 ± 0.24 ops/ms

OPTIMIZED JAVA

1.0 ± 0.22 ops/ms

🤔

WHEN YOU CAN'T GET SPEED
OPTIMIZE FOR READABILITY

WHEN YOU CAN GET SPEED AVOID CHAINING EAGER OPERATIONS

Keep Calm and Kotlin on

CHRISTIAN ROLF | SENIOR DEVELOPER | @CCROLF

Questions?