

Type/Test/Whatever Driven Development

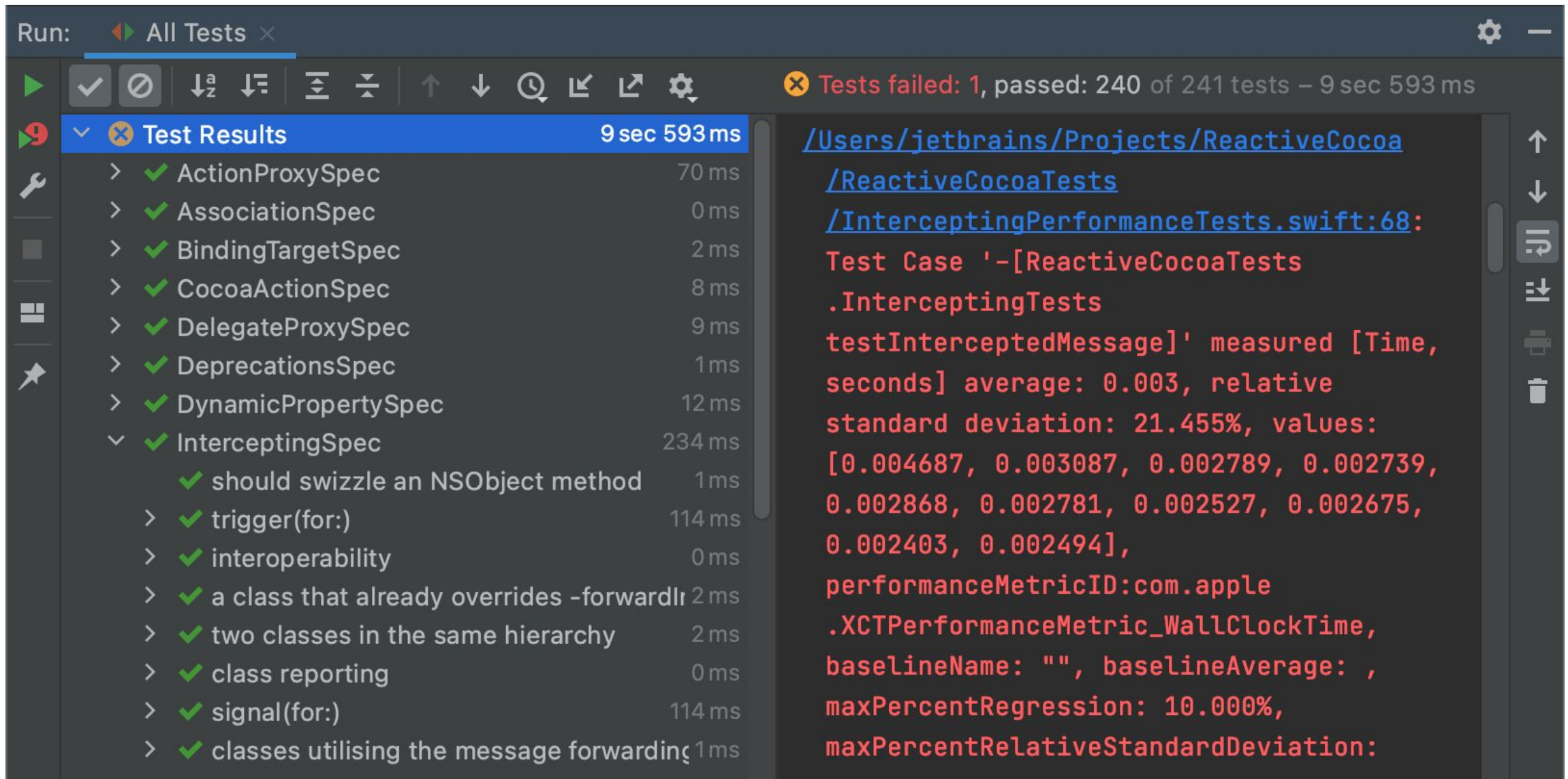


Qui suis je?



Nicolas François
Software Engineer - SRE
nfrancois@mediarithmics.com
@koisell

Motivations de cette présentation



Run: All Tests ×

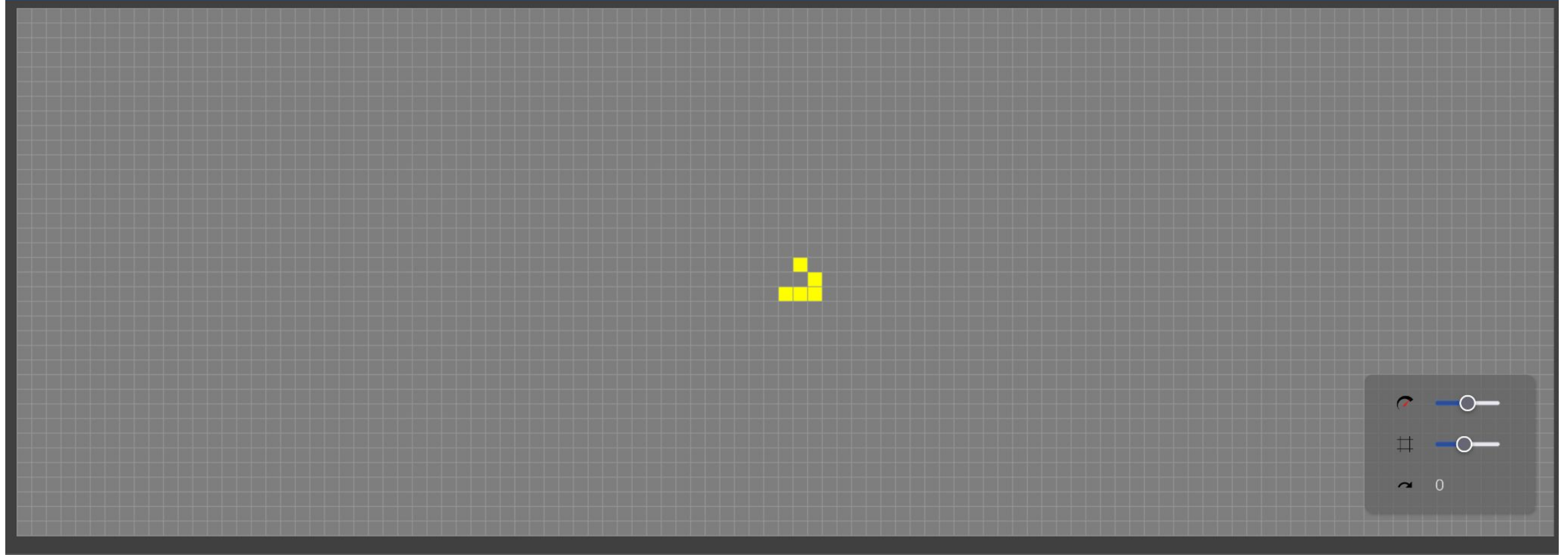
Tests failed: 1, passed: 240 of 241 tests – 9 sec 593 ms

Test Results 9 sec 593 ms

- > ✓ ActionProxySpec 70 ms
- > ✓ AssociationSpec 0 ms
- > ✓ BindingTargetSpec 2 ms
- > ✓ CocoaActionSpec 8 ms
- > ✓ DelegateProxySpec 9 ms
- > ✓ DeprecationsSpec 1 ms
- > ✓ DynamicPropertySpec 12 ms
- ✓ InterceptingSpec 234 ms
 - ✓ should swizzle an NSObject method 1 ms
 - > ✓ trigger(for:) 114 ms
 - > ✓ interoperability 0 ms
 - > ✓ a class that already overrides -forwardI 2 ms
 - > ✓ two classes in the same hierarchy 2 ms
 - > ✓ class reporting 0 ms
 - > ✓ signal(for:) 114 ms
 - > ✓ classes utilising the message forwarding 1 ms

/Users/jetbrains/Projects/ReactiveCocoa
/ReactiveCocoaTests
/InterceptingPerformanceTests.swift:68:
Test Case '-[ReactiveCocoaTests
.InterceptingTests
testInterceptedMessage]' measured [Time,
seconds] average: 0.003, relative
standard deviation: 21.455%, values:
[0.004687, 0.003087, 0.002789, 0.002739,
0.002868, 0.002781, 0.002527, 0.002675,
0.002403, 0.002494],
performanceMetricID:com.apple
.XCTPerformanceMetric_WallClockTime,
baselineName: "", baselineAverage: ,
maxPercentRegression: 10.000%,
maxPercentRelativeStandardDeviation:

Game of Life



<https://playgameoflife.com/>

Live Coding here

```

import java.text.SimpleDateFormat
import java.time.{LocalDate, Period}

import S3Cleaning.dateOlderThanMoreAPeriod
import org.scalatest.{FunSuite, Matchers}

class S3CleaningTest extends FunSuite with Matchers {
  val dateParser = new SimpleDateFormat("yyyy-MM-dd")

  //          x          |-----x-----|
  // 2020-09-23          30 days          2020-11-20  2020-11-23
  val localDate = LocalDate.of(2020, 11, 23)

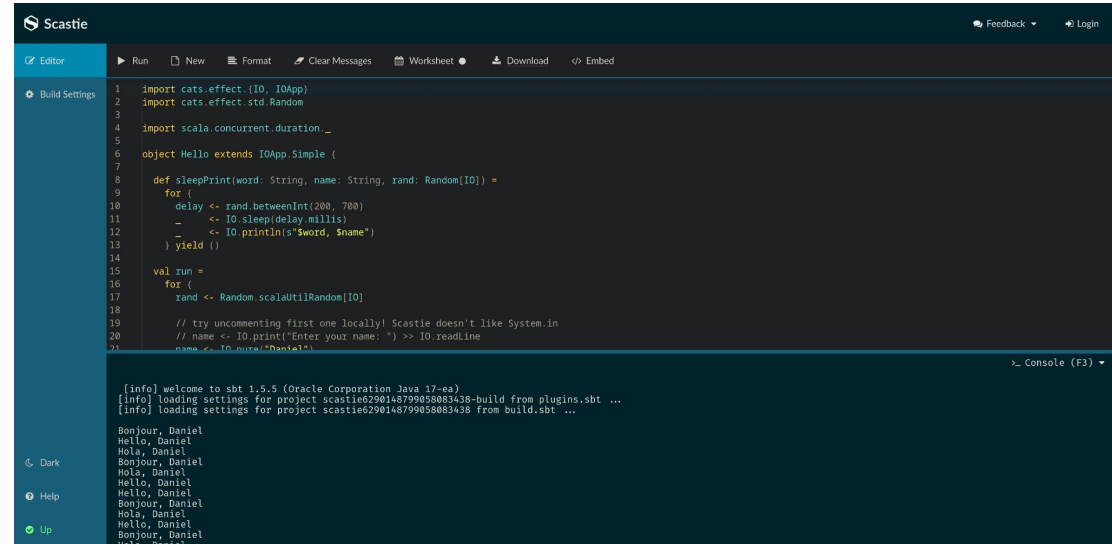
  test("date is older than duration") {
    dateOlderThanMoreAPeriod(dateParser.parse("2020-09-23"), localDate, Period.ofDays(30)) should
    be(true)
  }

  test("Date is NOT older than duration") {
    dateOlderThanMoreAPeriod(dateParser.parse("2020-11-20"), localDate, Period.ofDays(30)) should
    be(false)
  }
}

```

Repl Driven Development

- Read (lire une expression)
- Eval (évaluer cette expression)
- Print (afficher le résultat)
- Loop (retour à la première étape)

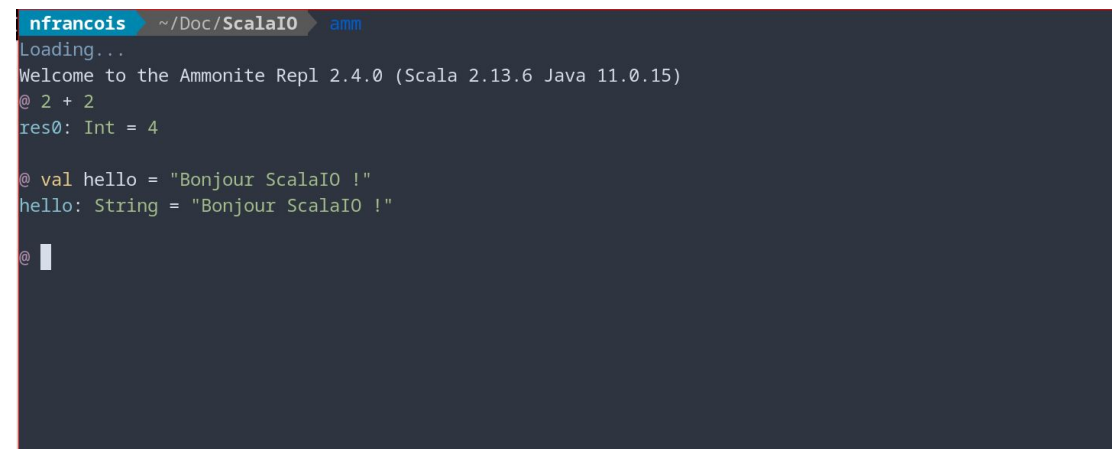


The screenshot shows the Scastie web IDE interface. The editor contains the following Scala code:

```
1 import cats.effect.{IO, IOApp}
2 import cats.effect.std.Random
3
4 import scala.concurrent.duration._
5
6 object Hello extends IOApp.Simple {
7
8   def sleepPrint(word: String, name: String, rand: Random[IO]) =
9     for {
10       delay <- rand.betweenInt(200, 700)
11       _ <- IO.sleep(delay.millis)
12       _ <- IO.println(s"$word, $name")
13     } yield ()
14
15   val run =
16     for {
17       rand <- Random.scalaUtilRandom[IO]
18
19       // try uncommenting first one locally! Scastie doesn't like System.in
20       // name <- IO.println("Enter your name: ") >> IO.readLine
21       name <- IO.pure("Daniel")
22     } yield {
23       sleepPrint(word, name, rand)
24     }
25 }
```

The console output shows the following messages:

```
[info] welcome to sbt 1.5.5 (Oracle Corporation Java 17-ea)
[info] loading settings for project scastie6290148799058083438-build from plugins.sbt ...
[info] loading settings for project scastie6290148799058083438 from build.sbt ...
Bonjour, Daniel
Hello, Daniel
Hola, Daniel
Bonjour, Daniel
Hola, Daniel
Hola, Daniel
Hello, Daniel
Hello, Daniel
Bonjour, Daniel
Hola, Daniel
Hello, Daniel
Bonjour, Daniel
```

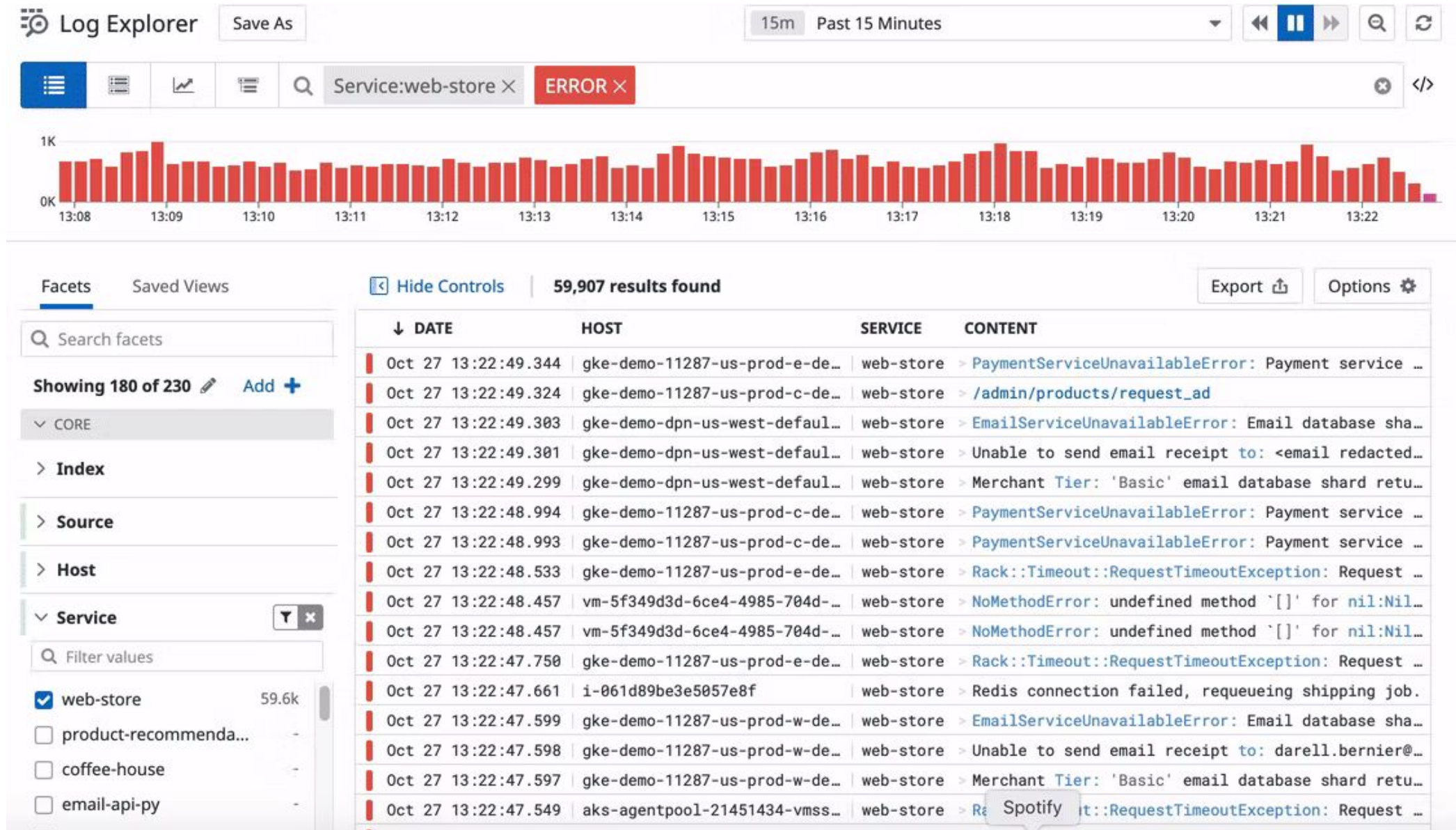


The screenshot shows the Ammonite REPL terminal. The prompt is `nfrancois ~/Doc/ScalaIO amm`. The terminal output is as follows:

```
Loading...
Welcome to the Ammonite Repl 2.4.0 (Scala 2.13.6 Java 11.0.15)
@ 2 + 2
res0: Int = 4

@ val hello = "Bonjour ScalaIO !"
hello: String = "Bonjour ScalaIO !"

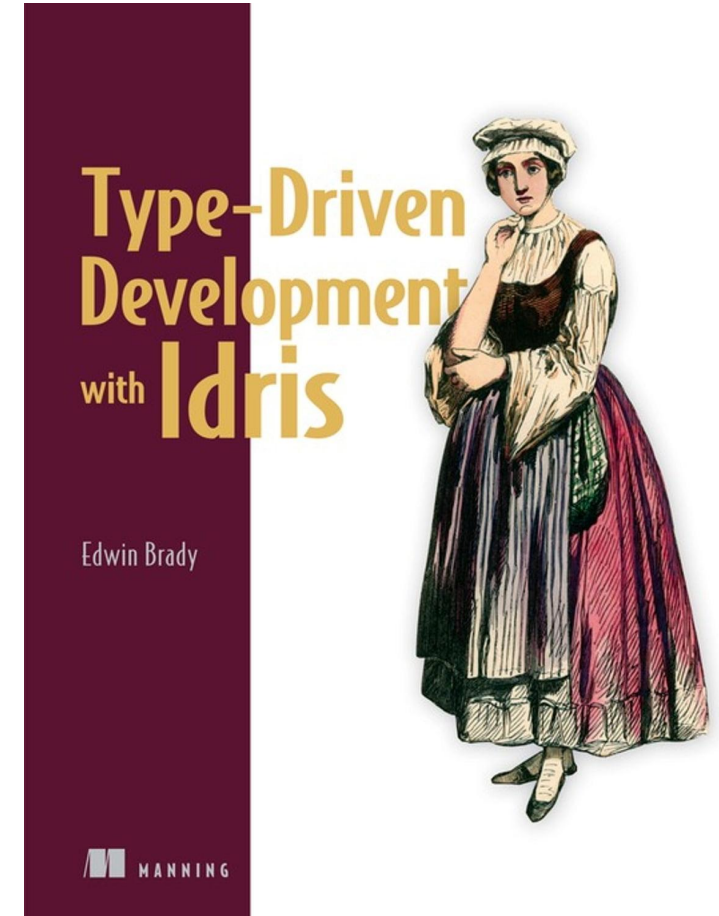
@
```

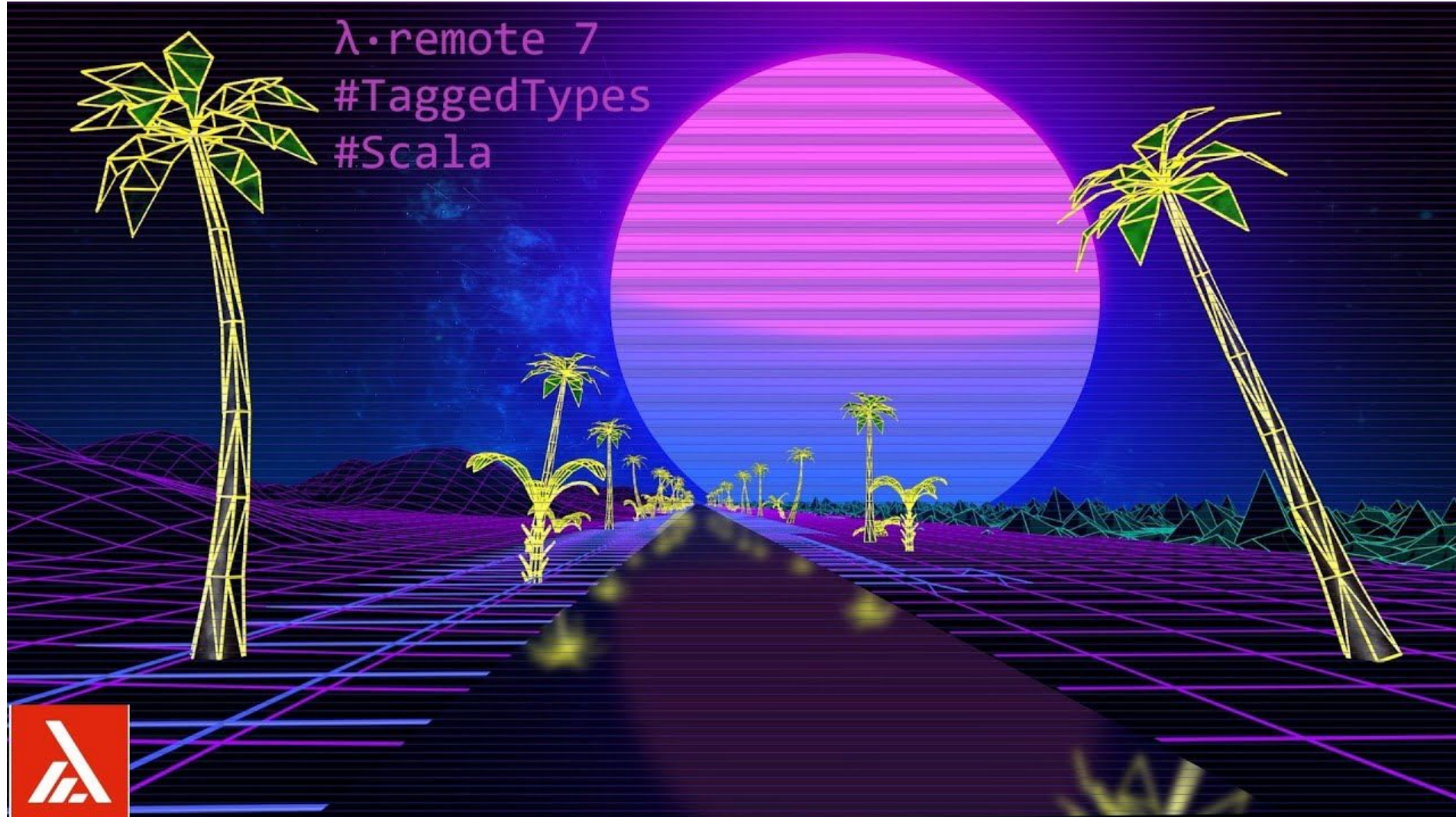
<https://www.datadoghq.com/product/log-management/>

Type Driven Development

- Type - Écrire un type comme entrée ou sortie de fonction
- Define - Créer une implémentation initiale pour cette fonction, avec potentiellement des trous
- Refine - Compléter la définition possiblement en modifiant les types au fur et à mesure que vous comprenez le problème



Les types tagués et raffinés et leur limites



<https://www.youtube.com/watch?v=FiD2ZtwzluU>

Property Based Testing

Much Ado About Testing
Nicolas Rinaudo



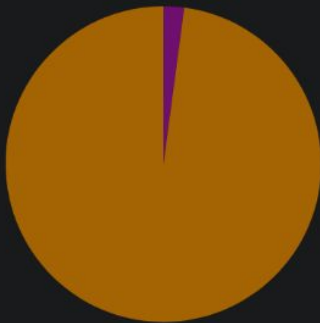
<https://youtu.be/Jhzc7fxY5lw>

https://www.youtube.com/watch?v=803USWX_E80

Une propriété est une spécification de haut niveau qui reste toujours correcte pour un ensemble de points (https://www.scalatest.org/user_guide/property_based_testing)

Quick Values for *root_cause_class_name*

Add to dashboard Customizer X



Found 232,160 messages with field *root_cause_class_name*, and 3,339 messages without field *root_cause_class_name*.

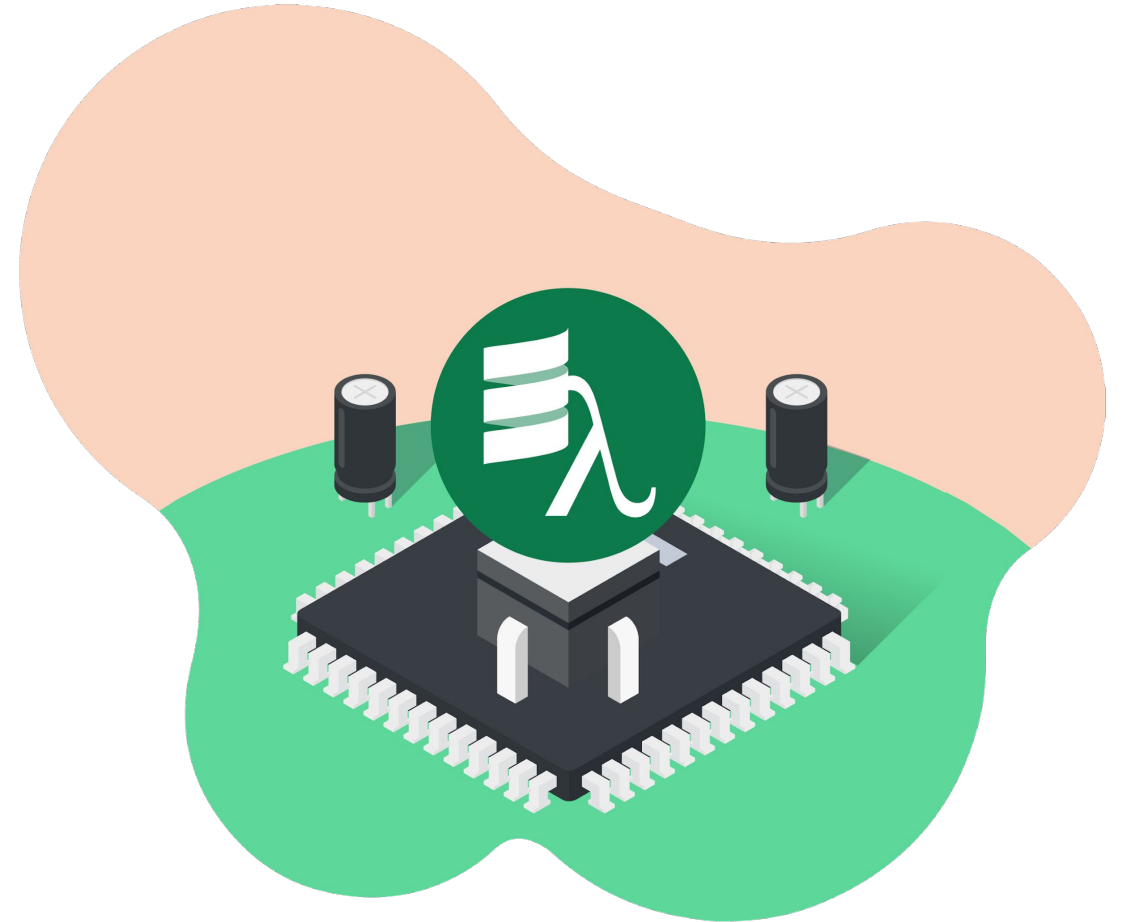
Value	%	Count
Top 5 values		
com.google.protobuf.InvalidProtocolBufferException	97.90%	227,282
java.util.concurrent.TimeoutException	2.08%	4,828
java.lang.RuntimeException	0.02%	50

Behavior Driven Development

- Boucle de feedback avec le métier
- Voir event storming, ...
- Ne pas voir Gherkin

Data Driven Development

- Boucle de feedback avec des KPI



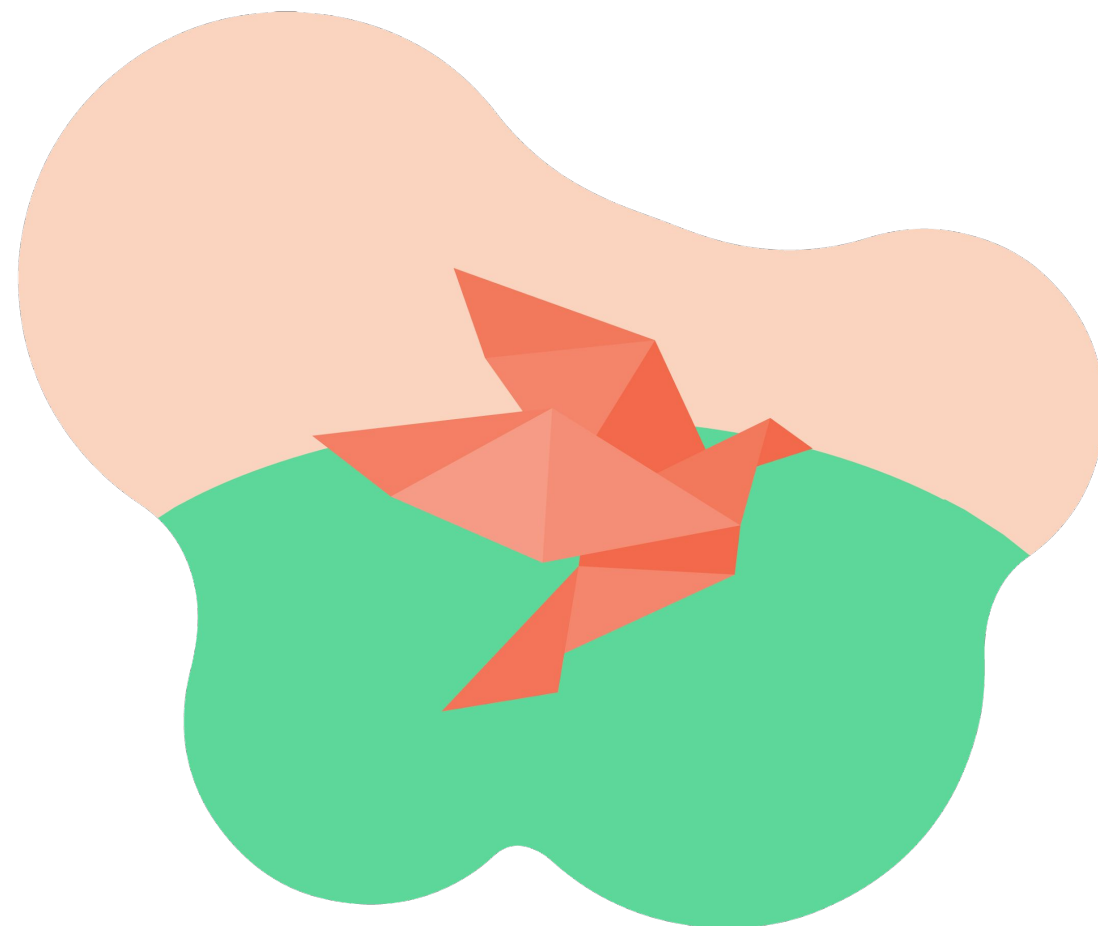
Pour résumer

- En tant que développeur et développeuse Scala nous sommes *implicitement* exposés à ce type de processus de développement
- La principale confusion pour comprendre ces techniques vient de leur *effet de bord*
- C'est l'utilisation *récurrente* de tests, types, ... qui permet d'être efficace. Mais chaque xDD peut être compris en quelques minutes (easy to learn, hard to master)
- Toutes ses techniques peuvent se combiner

Source

- <https://www.youtube.com/watch?v=KpQ-t9wWU3k>
- <https://www.manning.com/books/type-driven-development-with-idris>
- <https://www.manning.com/books/functional-and-reactive-domain-modeling>
- <https://www.manning.com/books/secure-by-design>
- https://www.youtube.com/watch?v=itGmiTS_IPw
- https://xvw.github.io/longs/introduction_algebraic_effects.html
- <https://medium.com/@wiemzin/make-your-program-testable-cee543c6fbbf>
- <https://www.signifytechnology.com/blog/2019/01/security-with-scala-refined-types-and-object-capabilities-by-will-sargent>
- <https://martinfowler.com/books/refactoring.html>
- <https://www.meetup.com/fr-FR/paris-software-craftsmanship/>
- <https://youtu.be/Jhzc7fxY5lw>

Questions ?



Merci !



<https://github.com/MEDIARITHMICS/talks>

