Property-based Testing

Norbert Schneider

DEFINITION

Let's have a look at the dictionary

Property

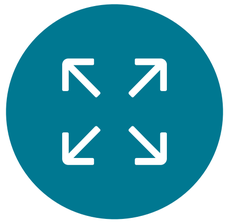
An attribute, quality, or characteristic of something.

OXFORD DICTIONARIES

'the property of heat to expand metal at uniform rates'

PROPERTY OF THE REVERSE FUNCTION

The „hello world“ of PBT



Preserves the size

```
size(L) == size(reverse(L))
```

Doubble reverse inverts operation

```
reverse(reverse(L)) == L
```

Preserves the content

```
containsAll(L, reverse(L))
```

PROPERTY OF FIZZBUZZ

You know it, don't you?

Multiples of three
print "Fizz"

```
N%3 == 0 &&
fizzbuzz(N) == „Fizz"
```

Multiples of five
print "Buzz"

```
N%5 == 0 &&
fizzbuzz(N) == „Buzz“
```

Numbers which are
multiples of three
and five print
"FizzBuzz"

```
N%3 == 0 && N%5 == 0 &&
fizzbuzz(N) == „FizzBuzz"
```

PROPERTY OF A SHOPPING CART

Somewhat more realistic business properties

Positive aggregated cost

```
cost(C) ⩾ 0
```

Positive integer amounts of items

```
forAll(C.items,
    item →
item.amount instanceOf Int
    && item.amount > 0)
```

Contains any item only once

```
forAll(C.items,
    item →
C.count(item) == 1)
```

TESTING PROPERTIES

1
Feature

3 - 4
Testcases

TESTING PROPERTIES

n
Feature

(3 - 4) * n
Testcases
O(n)

TESTING PROPERTIES

`n * n`

`O(n * n)`

TESTING PROPERTIES

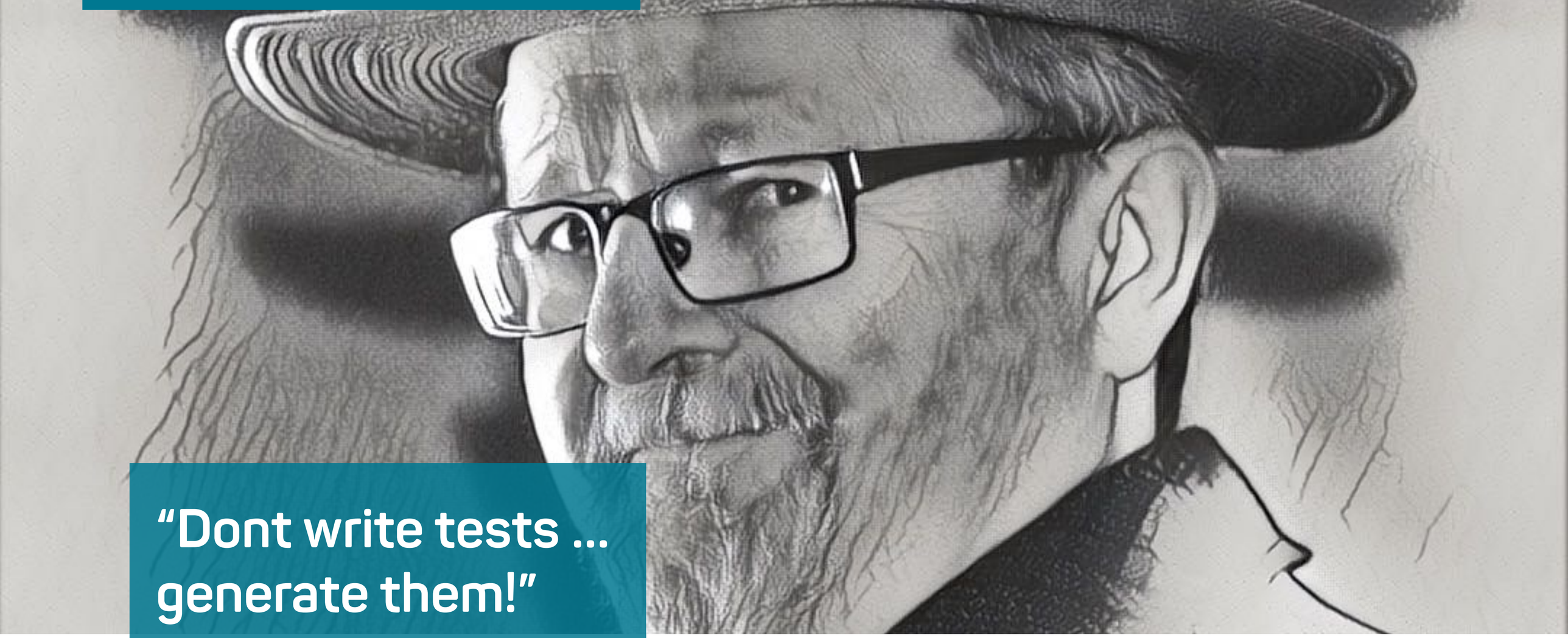
n * n

O(n * n)

TESTING PROPERTIES

TESTING IS REALLY HARD!*

* Not in the TDD sense



“Dont write tests … generate them!”

JOHN HUGHES

Quick Check

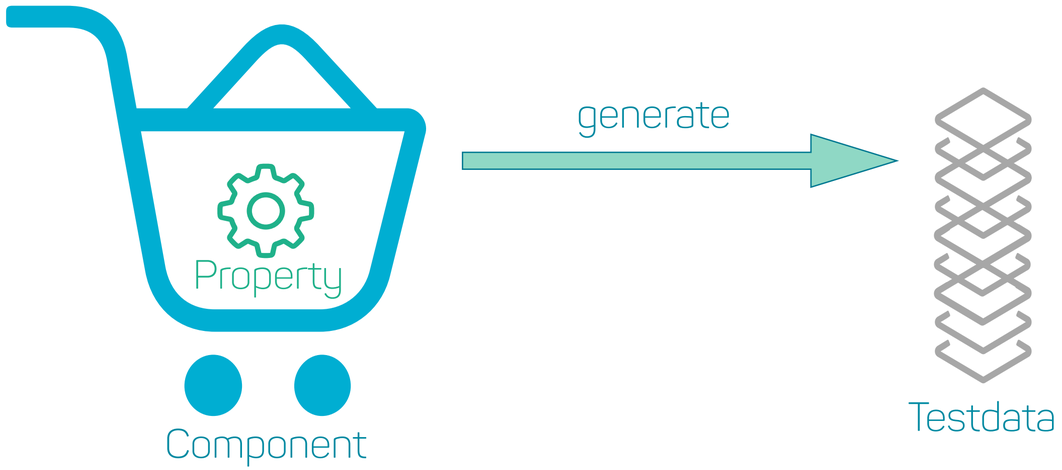
developed by John Hughes and Koen Claessen for Haskell in 1999

http://www.cse.chalmers.se/~rjmh/QuickCheck/

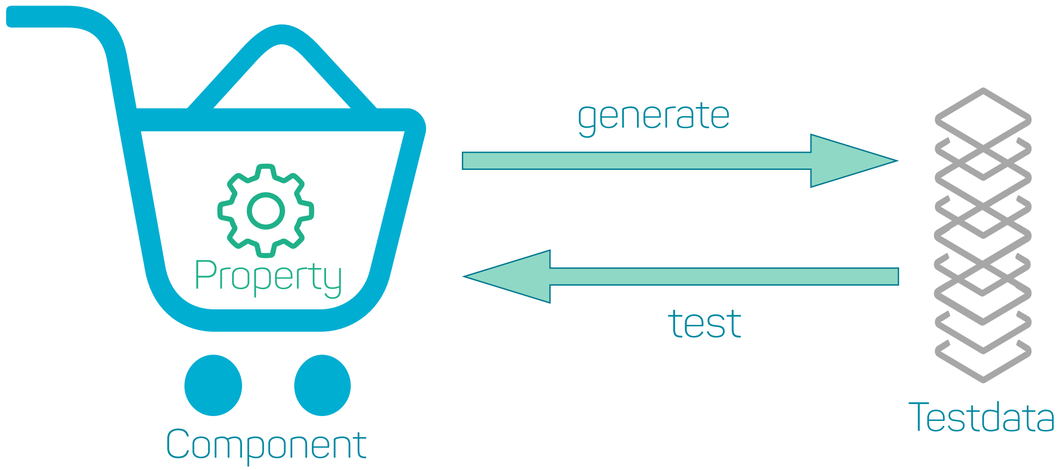
HOW DOES IT WORK

HOW DOES IT WORK

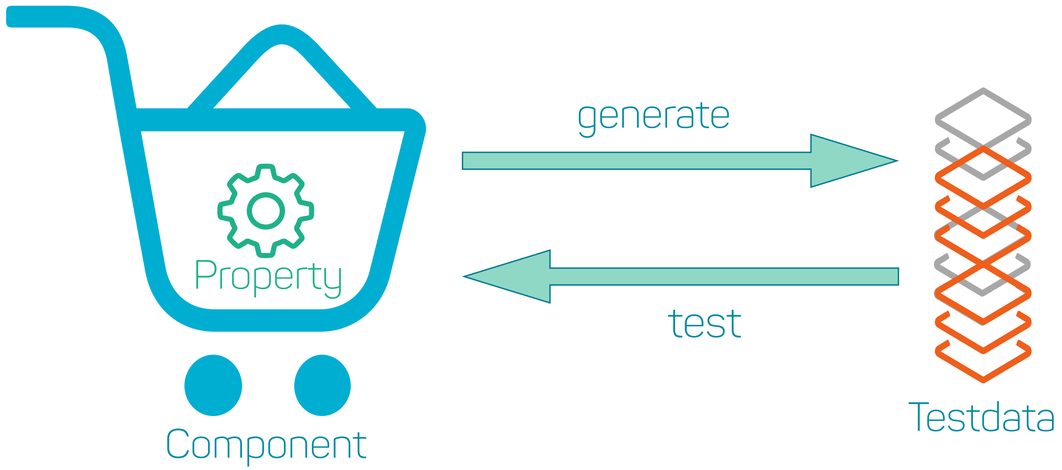
HOW DOES IT WORK



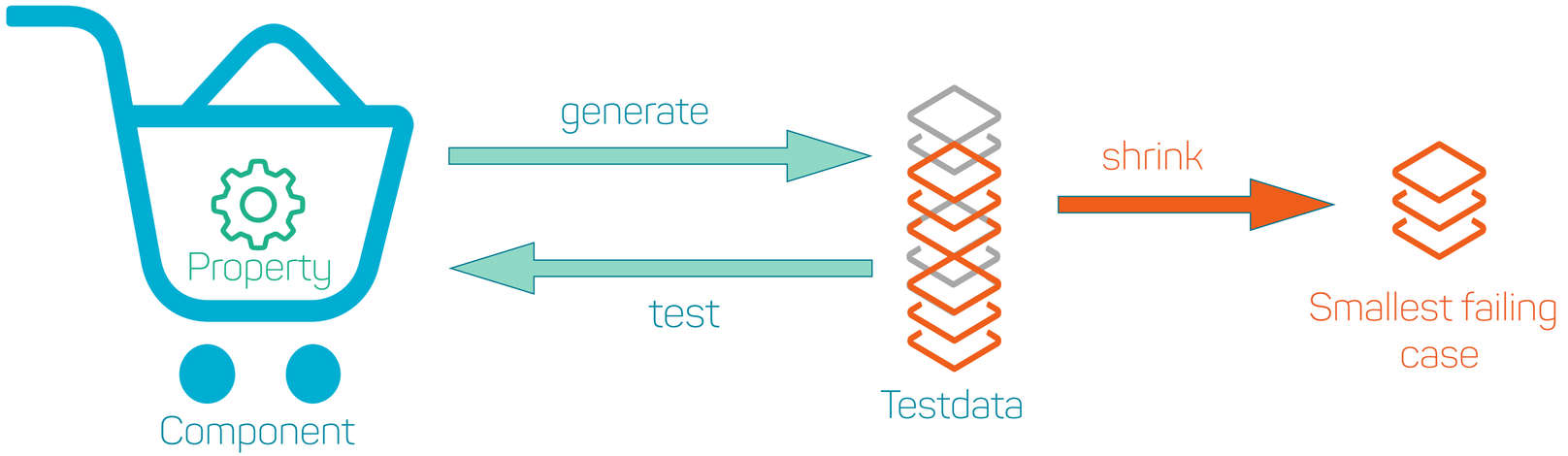
HOW DOES IT WORK



HOW DOES IT WORK



HOW DOES IT WORK



EXAMPLE PBT

Java QuickTheory

```
@Test
public void addTwoIntegersIsCommutative() {
    qt().forAll(integers().all(), integers().all())
            .check((i, j) →
                    add(i, j) == add(j, i)
            );
}
```

Quantifier

Generator

Function under test

Property

EXAMPLE PBT

Java QuickTheory

```
@Test
public void addingTwoIntegersIsCommutative() {
    qt().forAll(integers().all(), integers().all())
            .check((i, j) →
                    add(i, j) == add(j, i + i%2)
            );
}
```

EXAMPLE PBT

Java QuickTheory

```
java.lang.AssertionError: Property falsified after 3
example(s)

Smallest found falsifying value(s) :-

{1, 0}

Other found falsifying value(s) :-

{65, 0}

{119, 0}

{127, 0}

{229, 0}

…

Seed was 57873390394314
```

EXAMPLES

Let’s have a look at some code

Java - jqwik

http://jqwik.net/

```
class FizzBuzzTests {

  @Property
  boolean everyThirdElementStartsWithFizz(@ForAll("divisibleBy3") int i) {
    return fizzBuzz().get(i - 1).startsWith("Fizz");
  }

  @Provide
  Arbitrary<Integer> divisibleBy3() {
    return Arbitraries.integers().between(1, 100).filter(i -> i % 3 == 0);
  }
}
```

Scala - ScalaCheck

http://scalacheck.org/

```
object StringSpecification extends Properties("String") {
    property("startsWith") = forAll { (a: String, b: String) ⇒
        (a+b).startsWith(a)
    }

    property("concatenate") = forAll { (a: String, b: String) ⇒
        (a+b).length > a.length && (a+b).length > b.length
    }

    property("substring") = forAll { (a: String, b: String, c: String) ⇒
        (a+b+c).substring(a.length, a.length+b.length) == b
    }
}
```

Clojure - test.check

https://github.com/clojure/test.check

```
(prop/for-all
  [a (gen/vector gen/any)
   b (gen/vector gen/any)]
    (= (count (concat a b))
       (+ (count a) (count b))))
```

JavaScript - JSCheck

http://www.jscheck.org

```
function le(a, b) { return a ⩽ b; }

JSC.test("Less than",
  function (verdict, a, b) {
    return verdict(le(a, b));
  },
  [JSC.integer(10), JSC.integer(20)],
  function (a, b) {
    if (a < b) { return 'lt'; }
    else if (a ⩵ b) { return 'eq'; }
    else {  return 'gt'; } } );
```

Haskell - QuickCheck

http://www.cse.chalmers.se/~rjmh/QuickCheck/

```
quickCheck((\s → (reverse.reverse) s == s) :: [Char] → Bool)
```

COMMON PATTERN

Some common pattern

„Different paths, same destination“

„There and back“

„The test oracle“

„Hard to prove, easy to solve“

„Some things newer change“

„The more things change, the more they stay the same“

List from Scott W.s great blog post at https://fsharpforfunandprofit.com/posts/property-based-testing-2

DISTINCTION TO TDD

Use the right tool for the job

TDD

- **One specific input to one specific output**
- Great to drive design
- Documentation of intention
- Many more ...

PBT

- **Random inputs to general property**
- More general and so less brittle
- One test can „include“ many examples
- Random inputs can reveal overlooked cases
- Force a cleaner design

Property-based tests are, like everything, no silver bullet, they are no verification and you can't test everything.
For example not every property can be tested without duplicating the production code or relaxing the property.

PBT and TDD share the same goal of building better and bug freer software and thus should be used in conjunction.

LINKS

Further material

QuickCheck
http://www.cse.chalmers.se/~rjmh/QuickCheck/

Property-based Testing in Java: Introduction
http://blog.johanneslink.net/2018/03/24/property-based-testing-in-java-introduction/

Choosing properties for property-based testing
https://fsharpforfunandprofit.com/posts/property-based-testing-2/

QuickCheck in Every Language
https://hypothesis.works/articles/quickcheck-in-every-language/

Testing the Hard Stuff and Staying Sane
https://www.youtube.com/watch?v=zi0rHwfiX1Q

An introduction to property based testing
https://fsharpforfunandprofit.com/pbt/

Property-Based Testing for everyone
https://www.youtube.com/watch?v=5pwv3cuo3Qk

Testing Asynchronous APIs With QuickCheck
https://www.youtube.com/watch?v=iW2J70f8jsE

Stay connected

... for test generation and otherwise

Our mission – to promote agile development, innovation and technology – extends through everything we do.

Address

codecentric AG
Hochstraße 11
42697 Solingen

Contact Info

E-Mail: norbert.schneider@codecentric.de
www.codecentric.de

Telephone

Telefon: +49 (0) 151 10 86 70 21

Images

Photo by Giammarco Boscaro on Unsplash

Photo by Shane Aldendorff on Unsplash

Photo by Caleb Jones on Unsplash

Photo by Fabian Grohs on Unsplash

Photo by Chris Lawton on Unsplash

Photo by Ián Tormo on Unsplash

Photo by Andrew Ridley on Unsplash

Photo by beasty . on Unsplash

Photo by Anthony Intraversato on Unsplash

Photo by Susan Yin on Unsplash