WEEK3 - Feedbacks

Mathias COUSTÉ
18.02.2020

2

teams did not compete…

(because of mvn fail)

#TrendingErrors

5 → 0 teams had runtime errors… *(GenericClientError)*

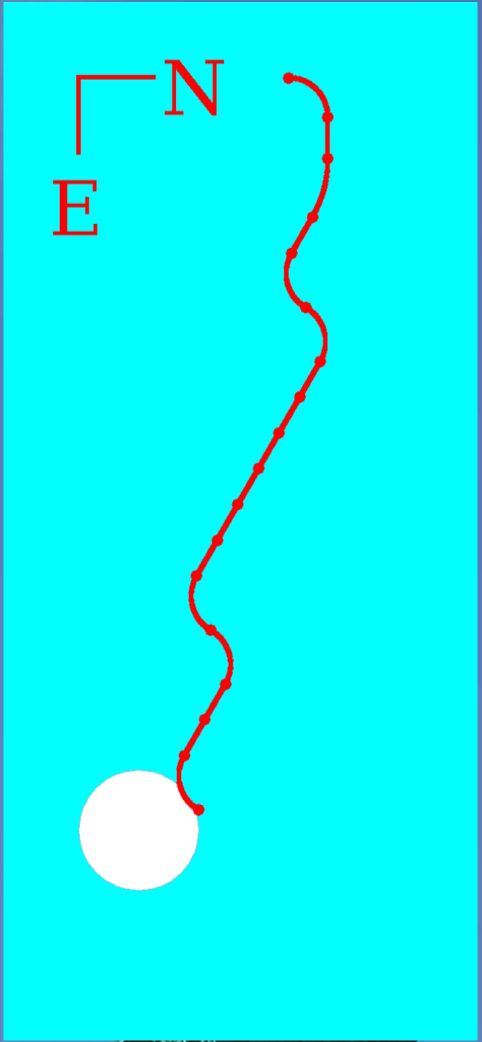
3 → 0 Wrong Format Exception

9 *(out of 18)*

team finished the race

\o/

SPECIAL
Closest miss ever
PRICE



Les Qualituriers

Cash_a_l_eau

Week's champions

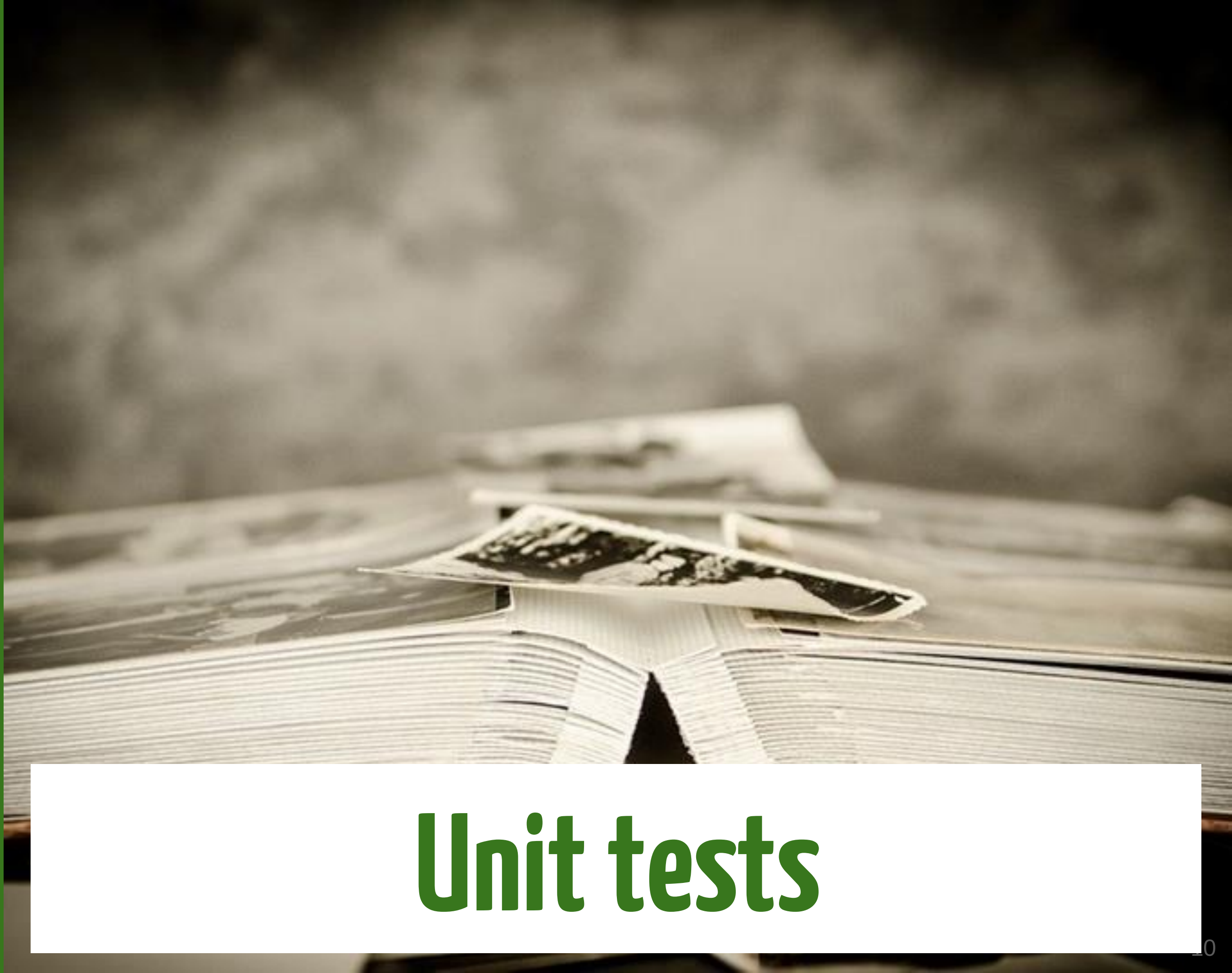
Gottagofast

Championship leader

More testing

Mathias COUSTÉ
18.02.2020

1



Unit tests

Did you said Unit test?

➔ Test a minimal piece of code as a unitary and indivisible block

➔ Make you that the smallest bricks of your solution are working

What do we test?

Classes & functions

Not a package, not a whole system

Mocks

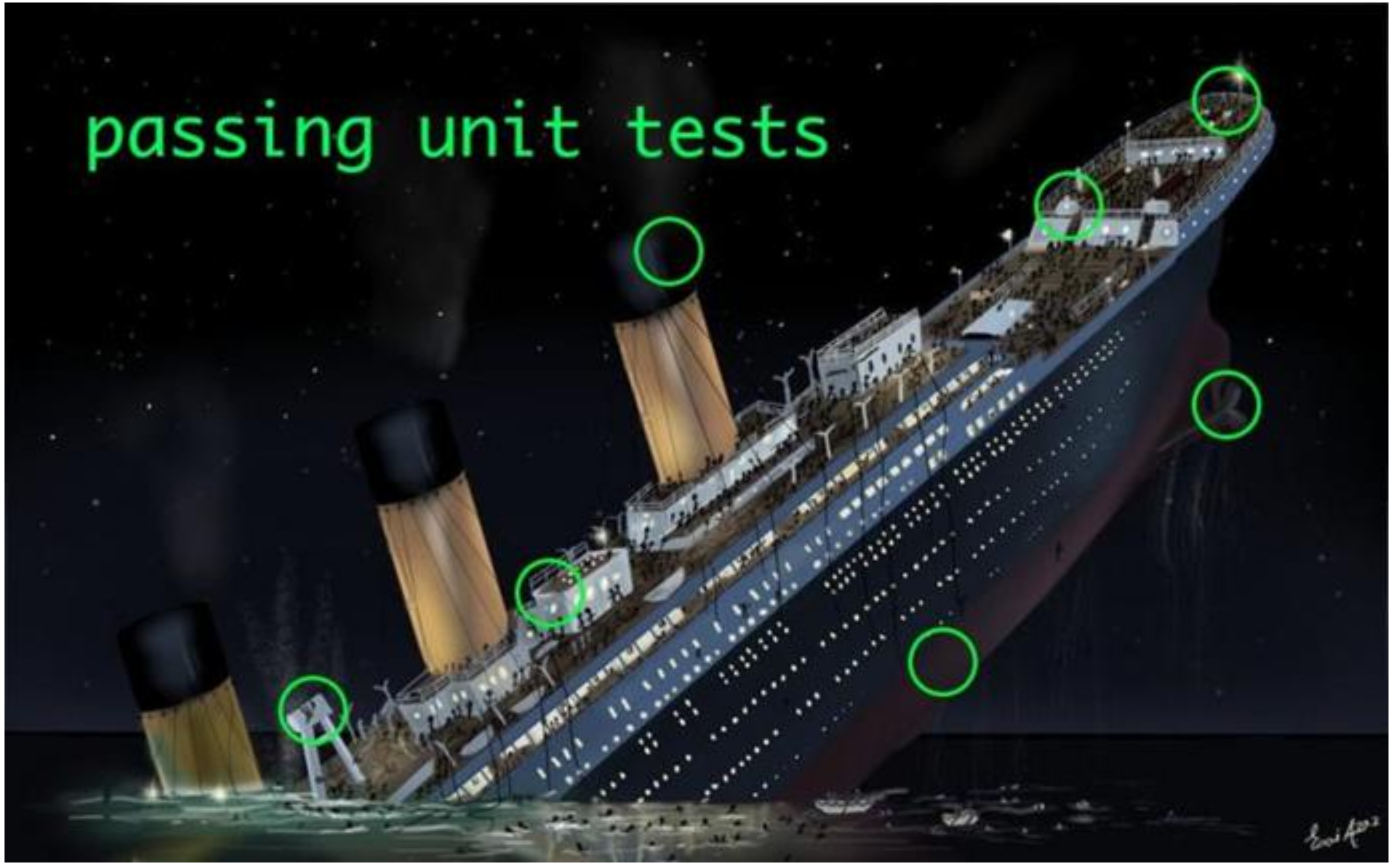
mockito

➔ Assure that your tested subject is in isolation by controlling all its neighbours

2

Integration testing

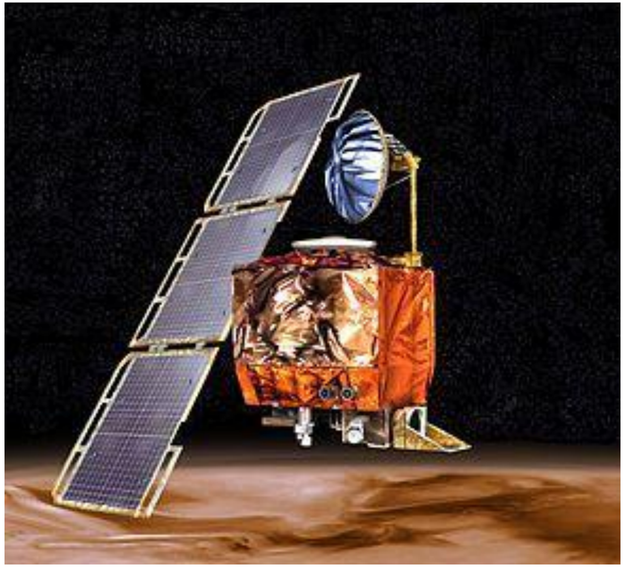
Unit test aren't enough?



Definition

"Integration testing is the phase in software testing in which individual software modules are combined and tested as a group."

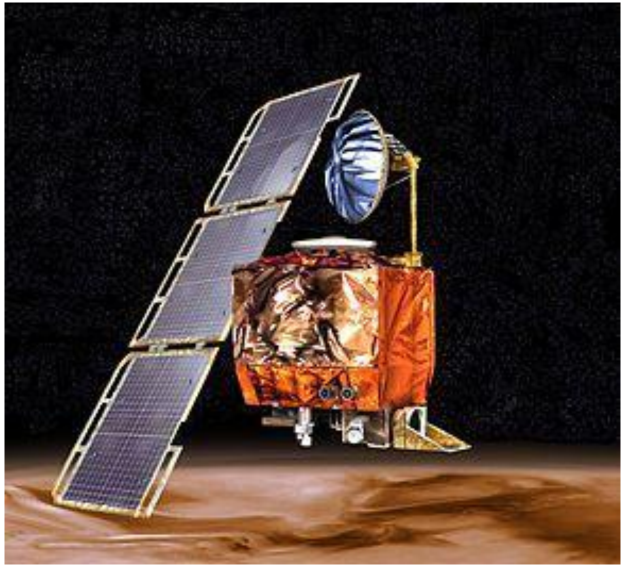
A sad example



Mars Climate Orbiter

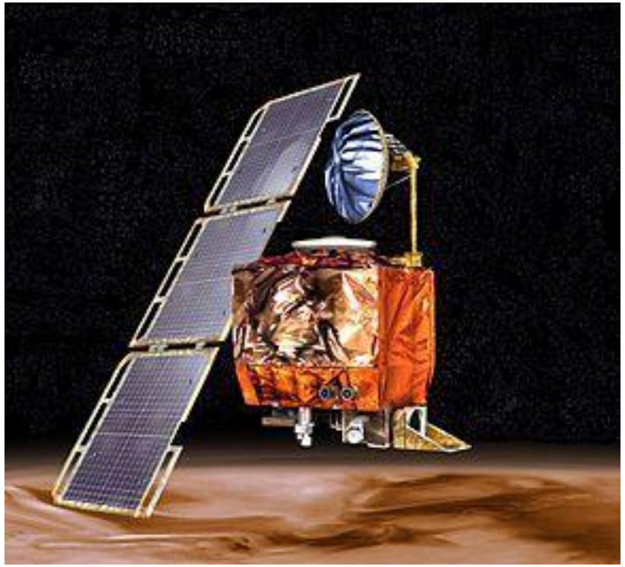
(year 1998)

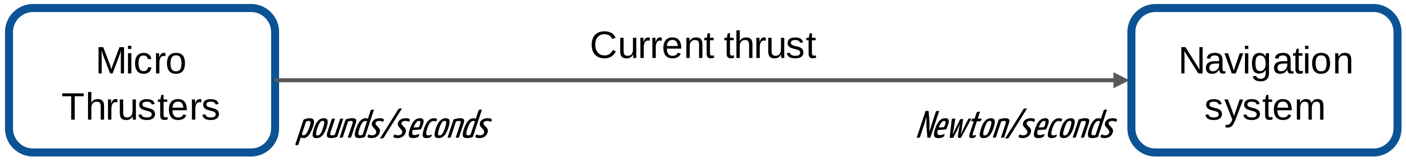
A sad example



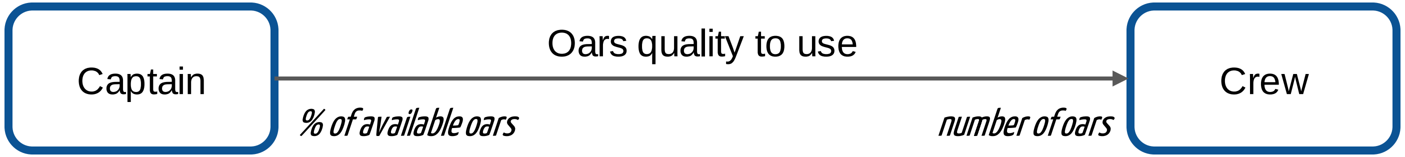
$$\begin{tikzcd}\boxed{\text{Micro Thrusters}} \arrow[r, "\text{Current thrust}"] & \boxed{\text{Navigation system}}\end{tikzcd}$$

A sad example





A sad example



What do we test?

Assert that code bricks are **interacting** as expected.

Which tools?

→ Same tools as Units test

→ It is **how you write** your test that make the difference

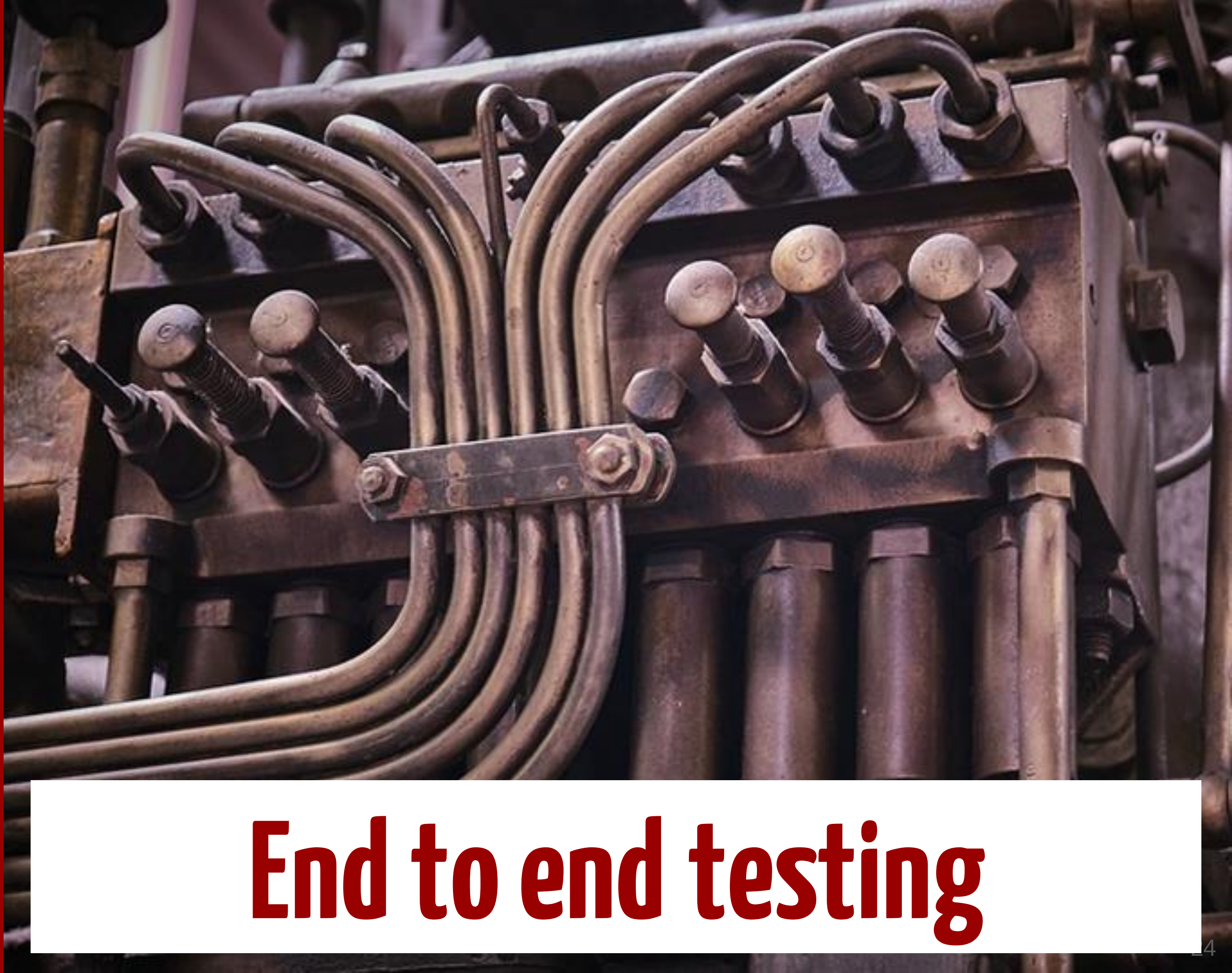
Which tools?

Differential unit test from integration test during the build execution

Maven™

https://www.baeldung.com/maven-integration-test

3



End to end testing

Definition

“End-to-end testing is a methodology used to test whether the flow of an application is performing as designed from start to finish.”

What do we test?

We put ourselves in the head of the final user and execute some scenarios.

The more your program is complex, the more scenarios you have.

A big part of the job is to identify which scenarios you will test and at which frequency

Let's test *Amazon.com*

Any suggestions ?

Let's test *Amazon.com*

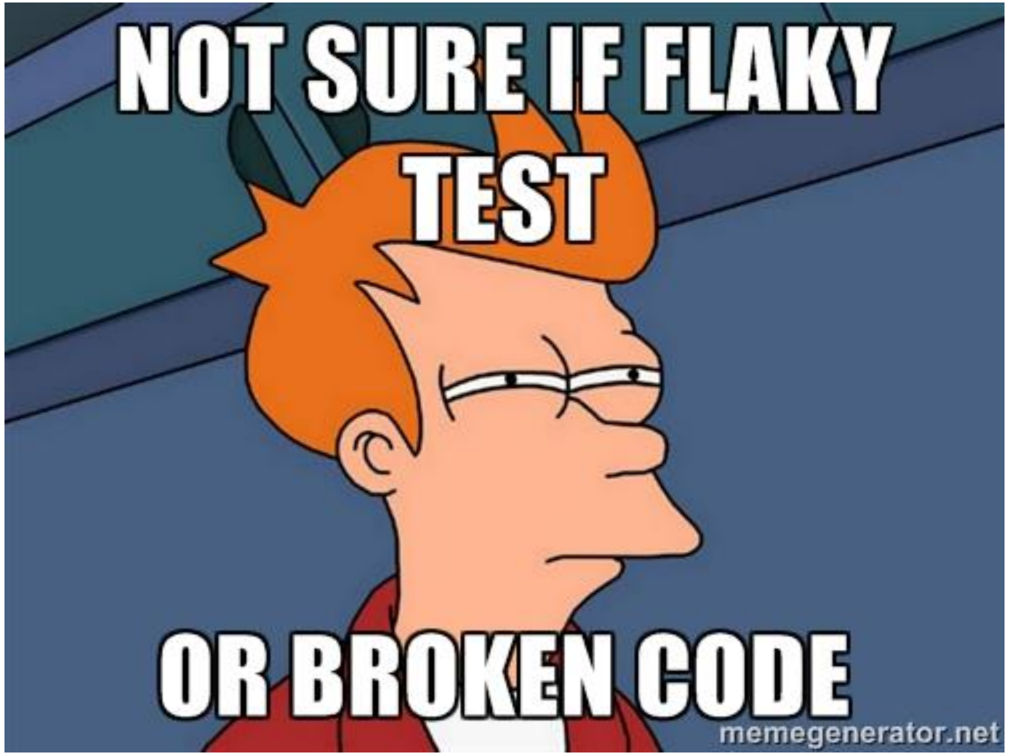
1. A *user* clicks on a link on google and is redirected to Amazon.com
2. The *user* sees a product
3. The *user* looks at others products related to the given product
4. The *user* clicks on "buy"
5. The *user* registers
6. The *user* fill the payment information form
7. The *user* click on "pay"
8. The *user* receives a confirmation email
9. Money transfer is done
10. Amazon warehouse is notified and shipping is enabled

Which tools?

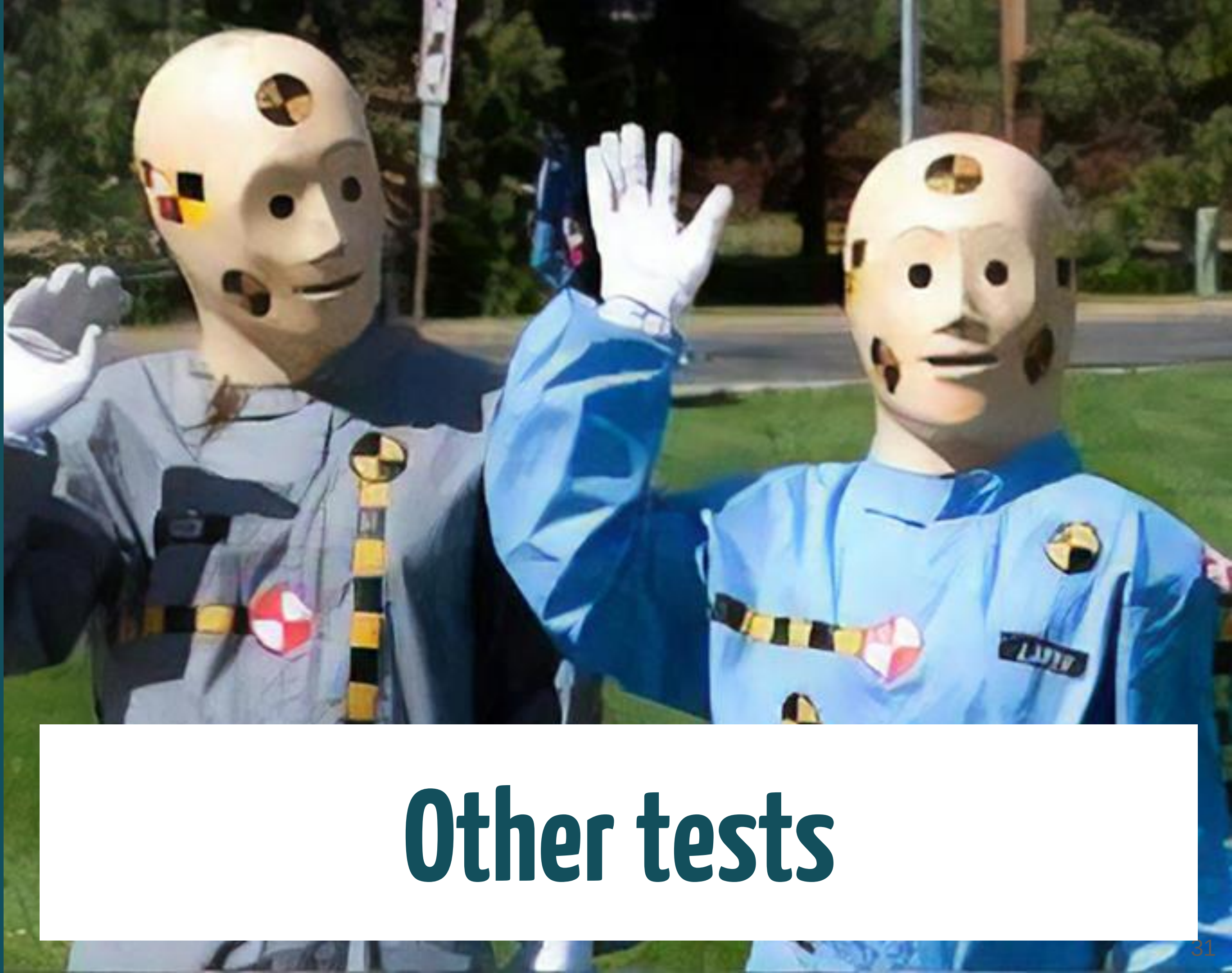
Selenium

cypress

Pros and Cons



4



Other tests

Performance testing

Performance testing

100 milliseconds

Performance testing

100 milliseconds

Amazon

1% in revenu loss

Google search

-8 000 000 requests/day

Intrusion testing

User tests

Q&A

Mutation testing

Mathias COUSTÉ
18.02.2020

1

Do you test well?

Unit tests - Some limits

```
@Test
public void mySuperTest() {
  functionThatDoesEverything();
  assertTrue(true);
}
```

Unit tests - Some limits

```
@Test
public void mySuperTest() {
  functionThatDoesEverything();
  assertTrue(true);
}
```

$\longrightarrow$ Code coverage

100%

Unit tests - Some limits

The student

The teacher

2

Mutation testing

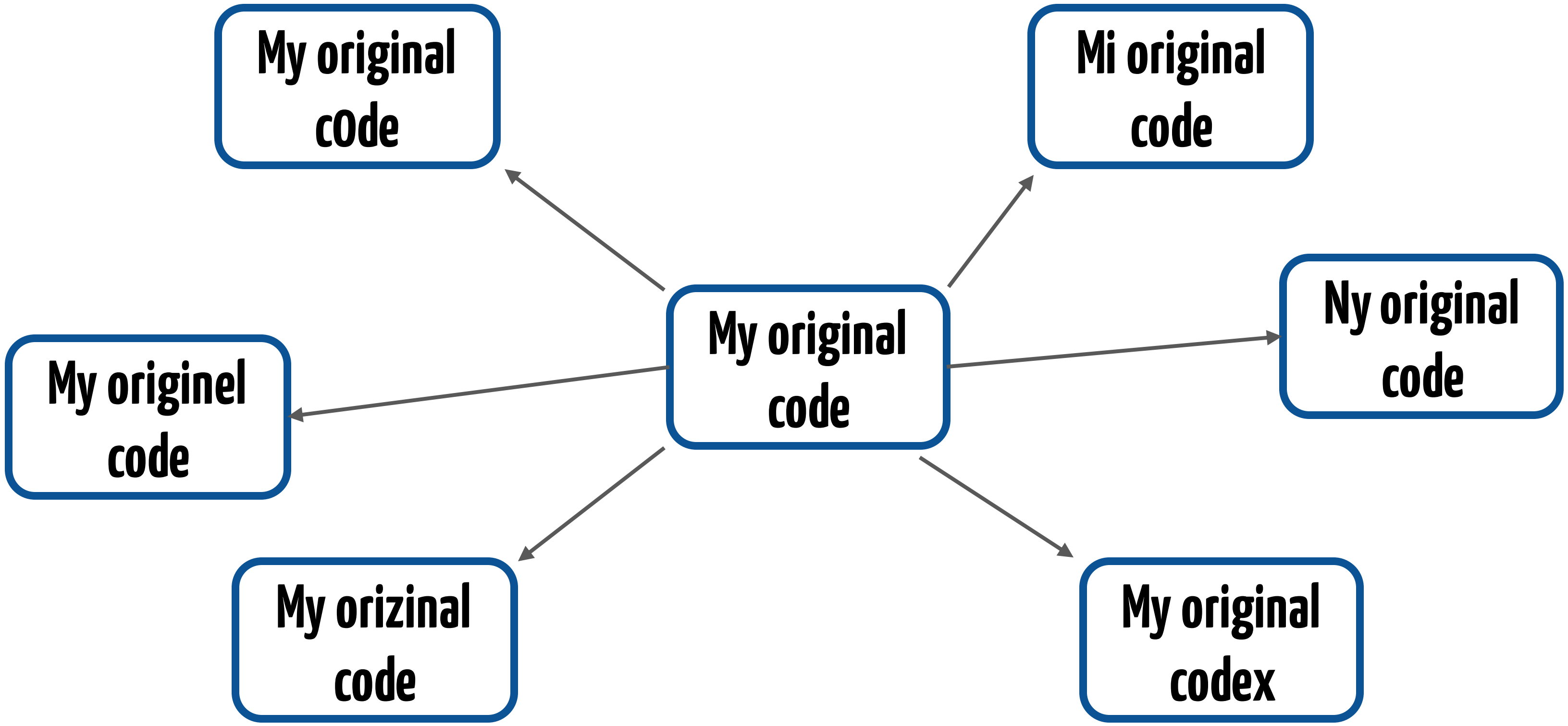
Principe

“**Mutation testing** involves modifying a program in small ways. Each mutated version is called a mutant and tests detect and reject mutants by causing the behavior of the original version to differ from the mutant.”

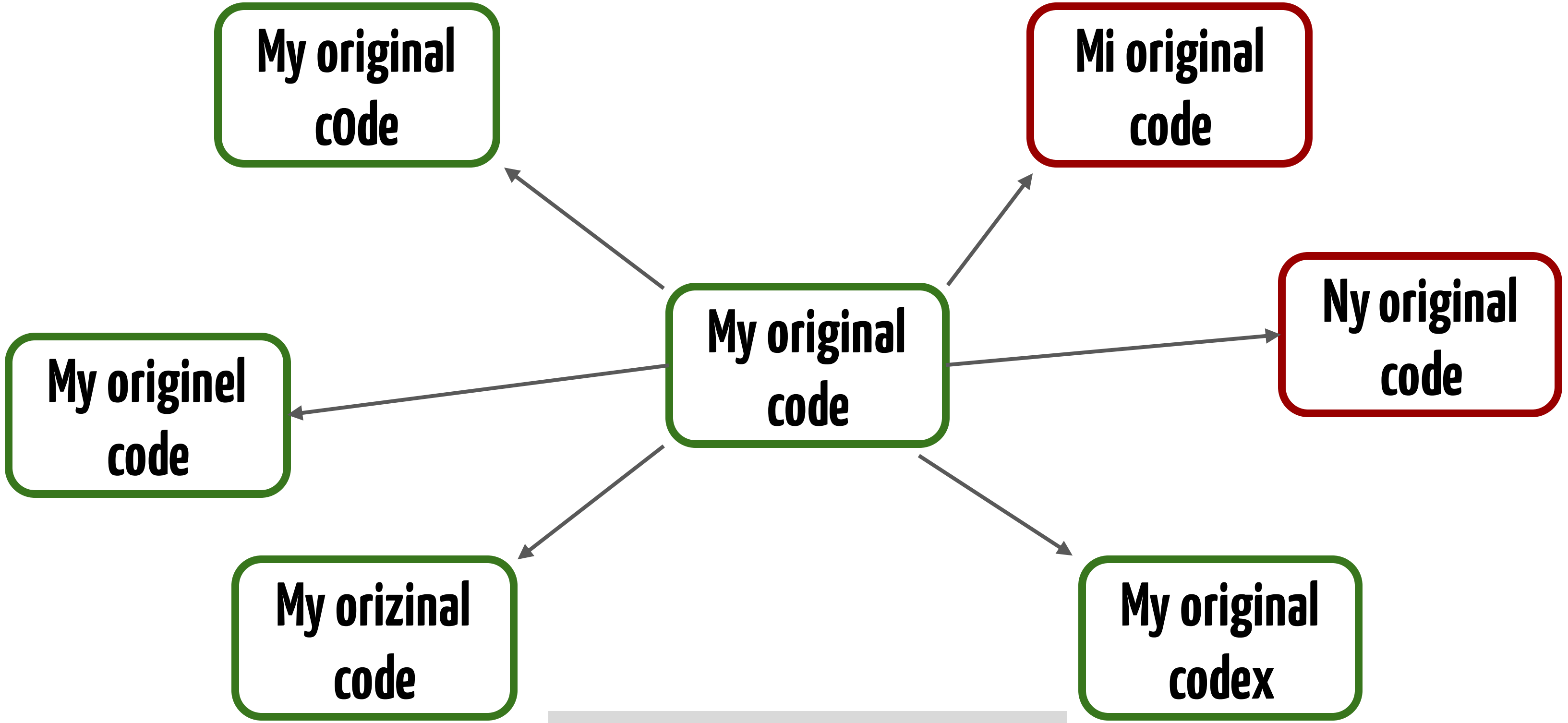
What is a mutant?

My original code

What is a mutant?

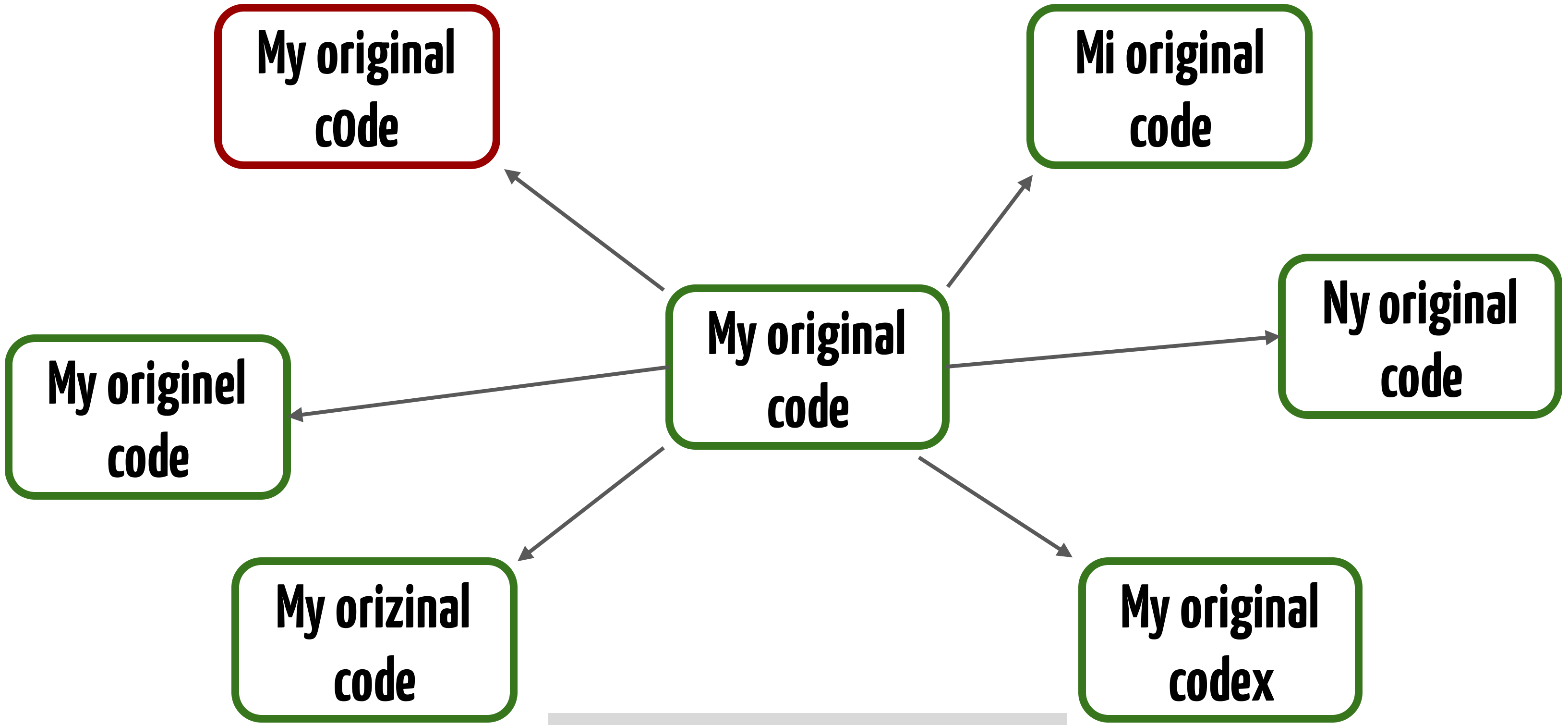


Testing your mutants



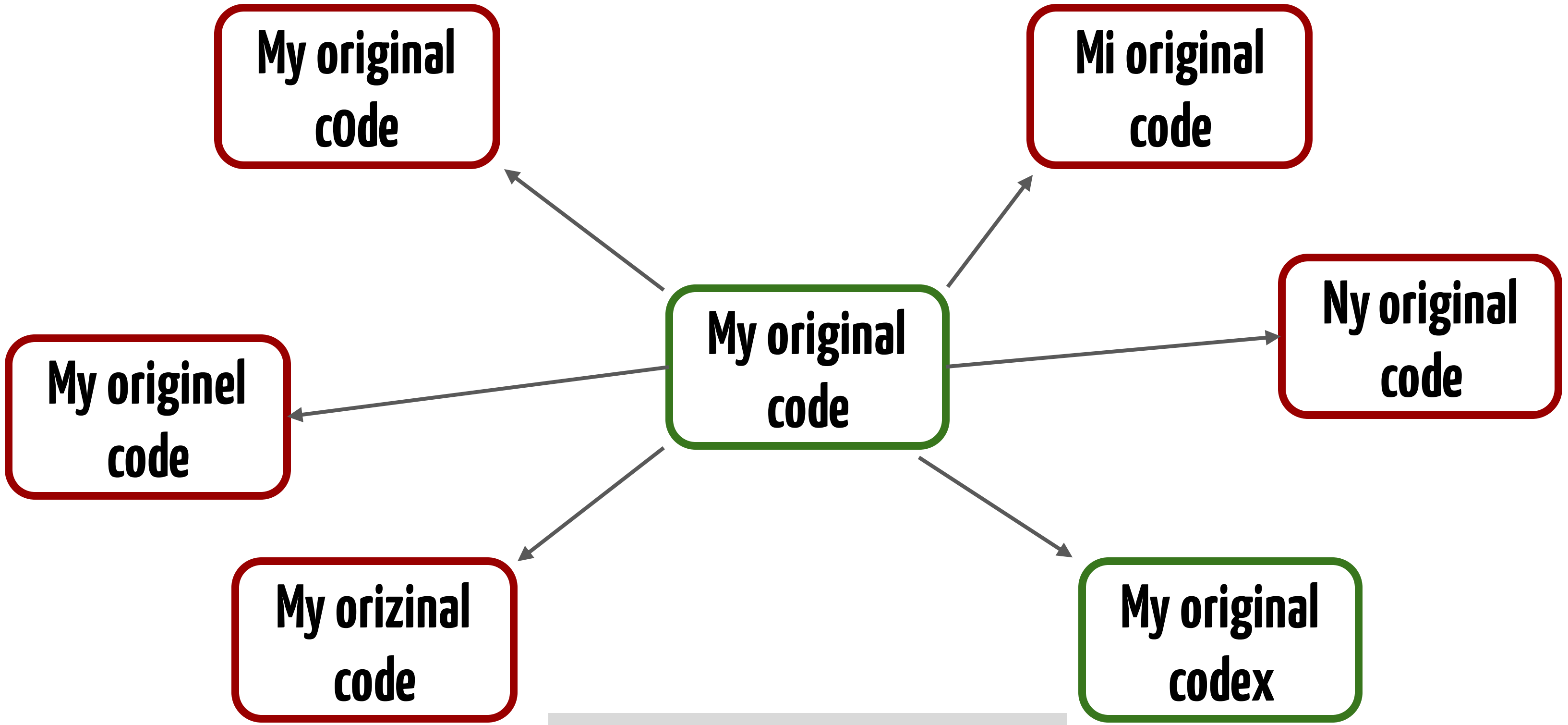
contains "My"!

Testing your mutants



no digits !

Testing your mutants



no spelling mistake !

What is a mutant?

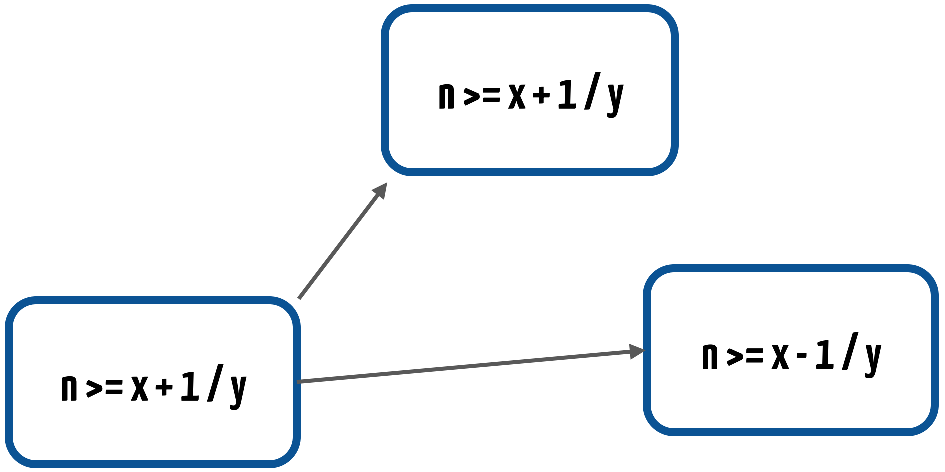
```
n >= x + 1 / y
```

What is a mutant?

$$\begin{tikzcd} & \boxed{\texttt{n == x + 1 / y}} \\ \boxed{\texttt{n >= x + 1 / y}} \arrow[ur] & \end{tikzcd}$$

$$\begin{tikzcd} \texttt{>=} \arrow[r] & \texttt{==} \end{tikzcd}$$

What is a mutant?



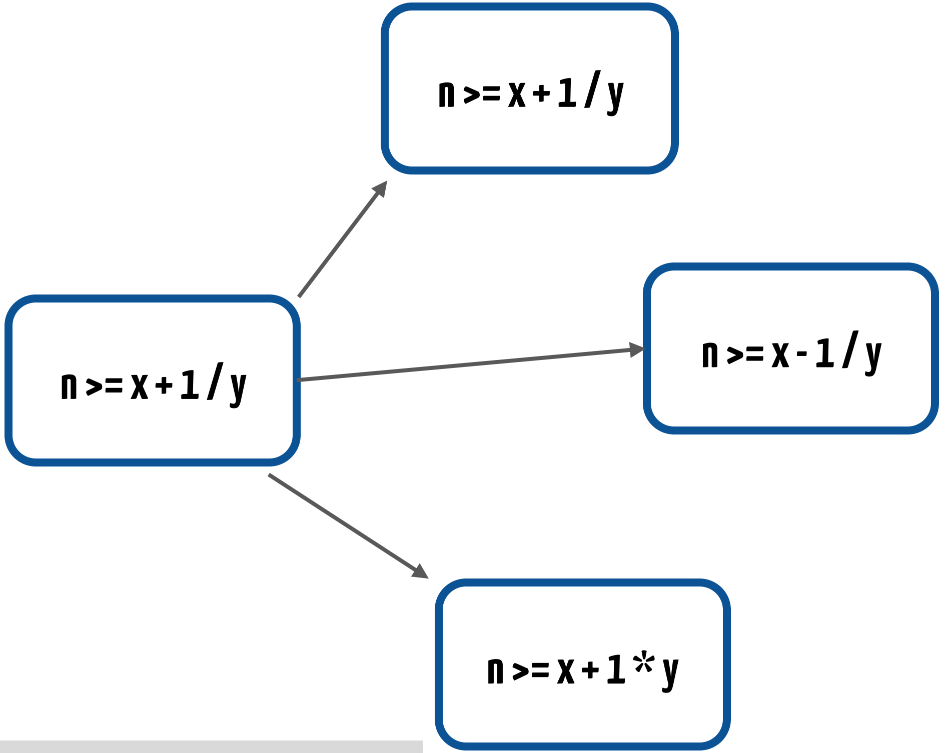
```
n>=x+1/y
```

```
n>=x+1/y
```

```
n>=x-1/y
```

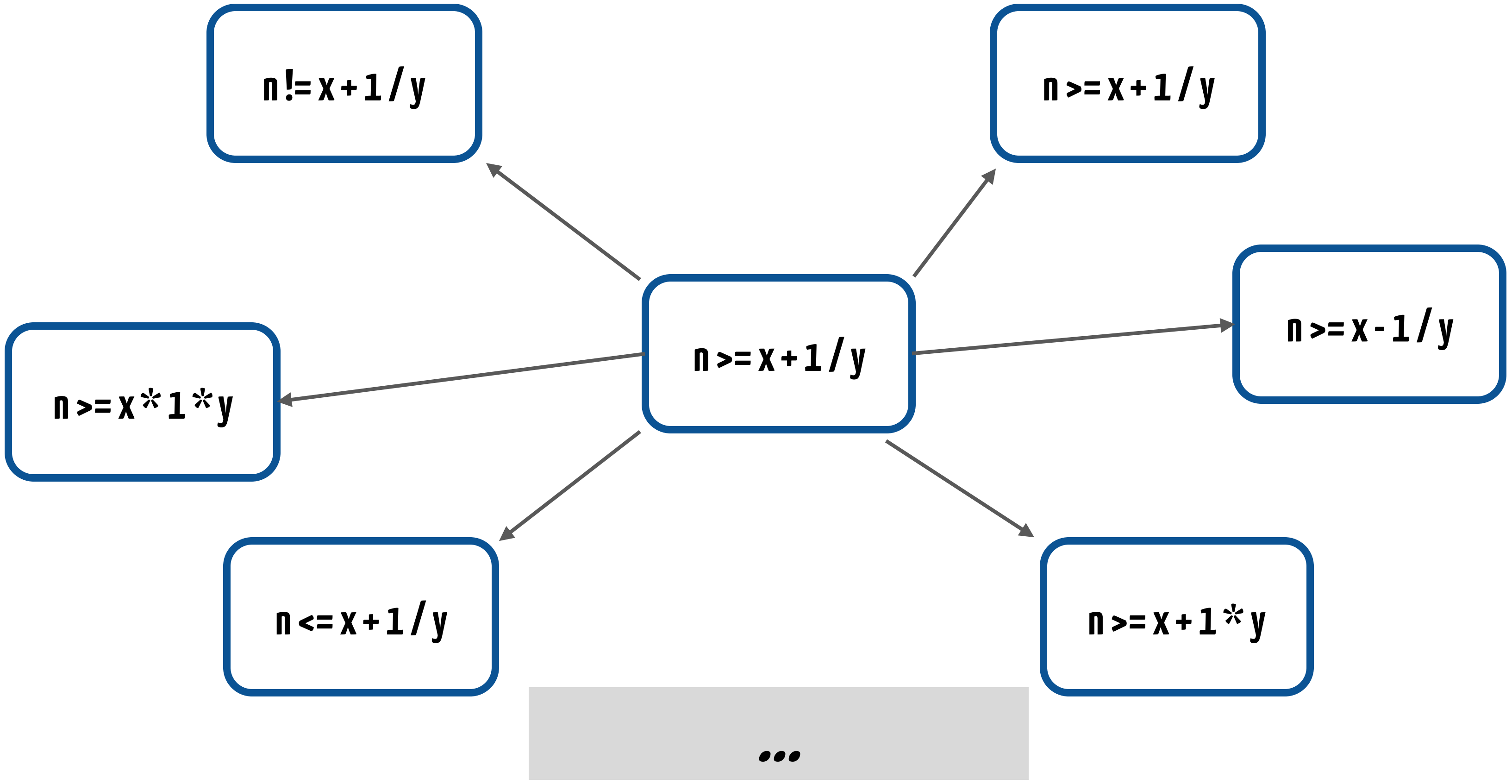
$+ \longrightarrow -$

What is a mutant?



$/ \longrightarrow *$

What is a mutant?



3

Using PITest on the project

PITest

“PIT is a state of the art mutation testing system, providing gold standard test coverage for Java and the jvm. It's fast, scalable and integrates with modern test and build tooling.”

Prerequisites

➔ Already have unit test

➔ The project is configured with maven

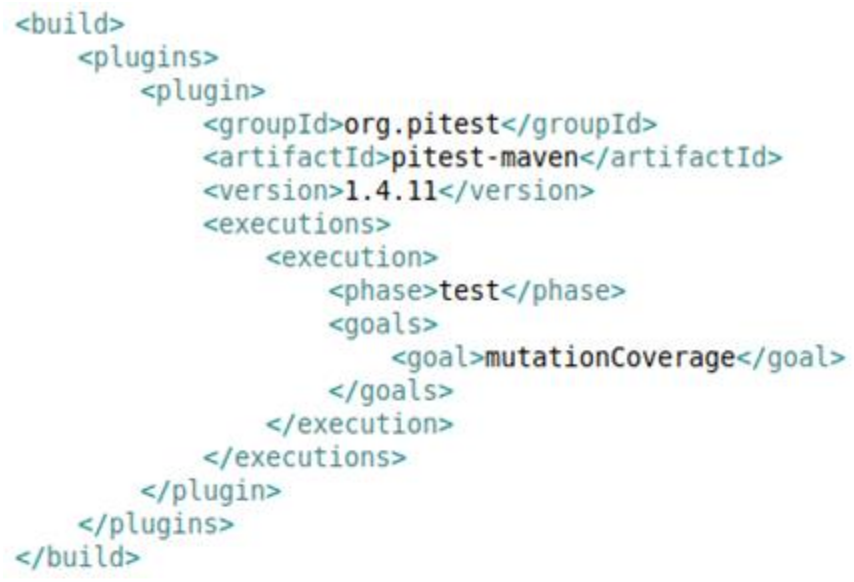
Maven command

Release the mutants when you want!

```
$ > mvn org.pitest:pitest-maven:mutationCoverage
```

Maven plugin

Release the mutants at each builds!

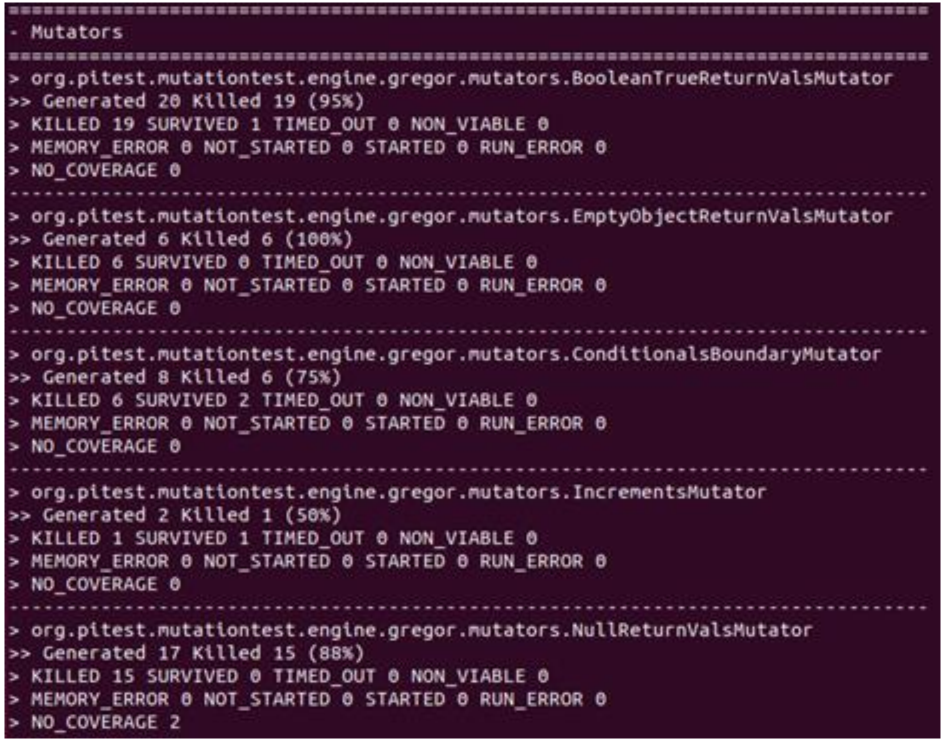


```
<build>
    <plugins>
        <plugin>
            <groupId>org.pitest</groupId>
            <artifactId>pitest-maven</artifactId>
            <version>1.4.11</version>
            <executions>
                <execution>
                    <phase>test</phase>
                    <goals>
                        <goal>mutationCoverage</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

Maven plugin

```xml
<plugin>
    <groupId>org.pitest</groupId>
    <artifactId>pitest-maven</artifactId>
    <version>1.4.11</version>
    <dependencies>
        <dependency>
            <groupId>org.pitest</groupId>
            <artifactId>pitest-junit5-plugin</artifactId>
            <version>0.12</version>
        </dependency>
    </dependencies>
    <executions>
        <execution>
            <phase>test</phase>
            <goals>
                <goal>mutationCoverage</goal>
            </goals>
        </execution>
    </executions>
    <configuration>
        <targetClasses>
            <param>fr.unice.polytech.si3.qgl.${artifactId}*</param>
        </targetClasses>
        <targetTests>
            <param>fr.unice.polytech.si3.qgl.${artifactId}*</param>
        </targetTests>
    </configuration>
</plugin>
```

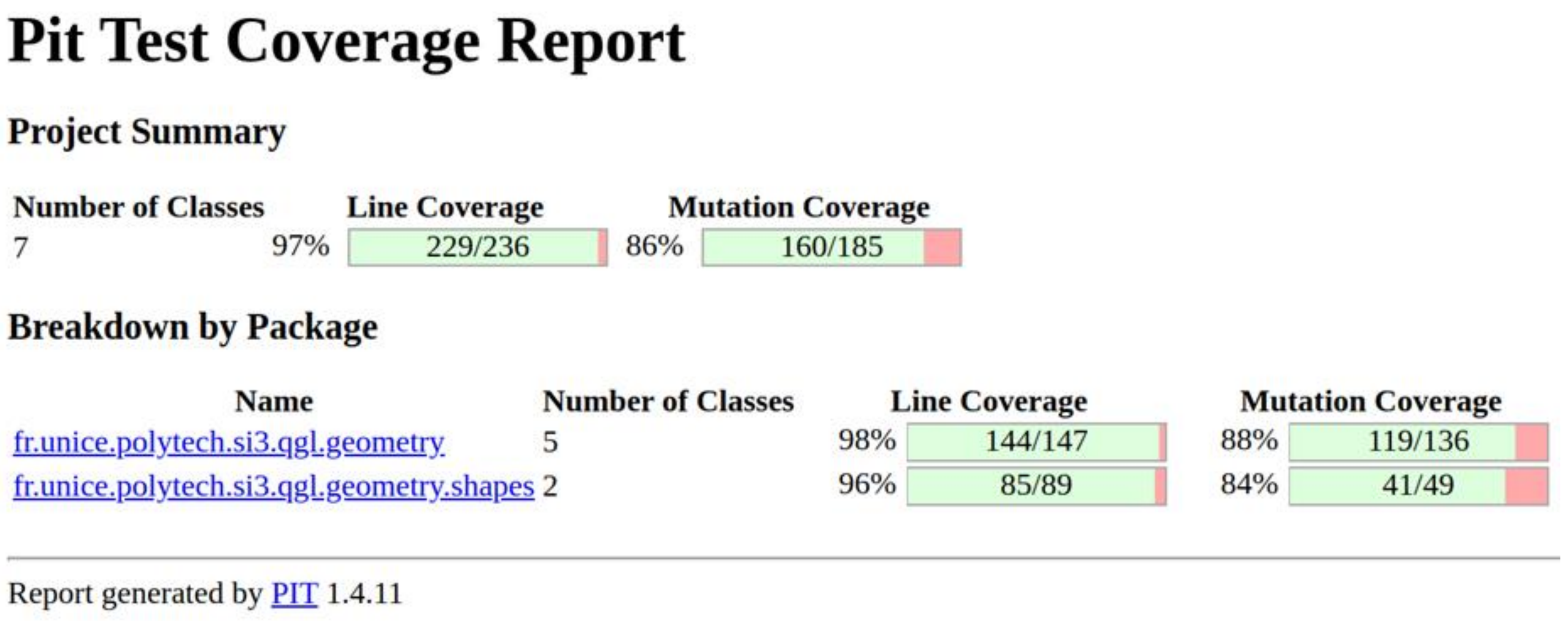
Understand the reports



```
================================================================================
- Mutators
================================================================================
> org.pitest.mutationtest.engine.gregor.mutators.BooleanTrueReturnValsMutator
>> Generated 20 Killed 19 (95%)
> KILLED 19 SURVIVED 1 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 0
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.EmptyObjectReturnValsMutator
>> Generated 6 Killed 6 (100%)
> KILLED 6 SURVIVED 0 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 0
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.ConditionalsBoundaryMutator
>> Generated 8 Killed 6 (75%)
> KILLED 6 SURVIVED 2 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 0
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.IncrementsMutator
>> Generated 2 Killed 1 (50%)
> KILLED 1 SURVIVED 1 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 0
--------------------------------------------------------------------------------
> org.pitest.mutationtest.engine.gregor.mutators.NullReturnValsMutator
>> Generated 17 Killed 15 (88%)
> KILLED 15 SURVIVED 0 TIMED_OUT 0 NON_VIABLE 0
> MEMORY_ERROR 0 NOT_STARTED 0 STARTED 0 RUN_ERROR 0
> NO_COVERAGE 2
```

Check the reports at *<your project>***/target/pit-reports/***<date>***/index.html**

Understand the reports



Pit Test Coverage Report

Project Summary

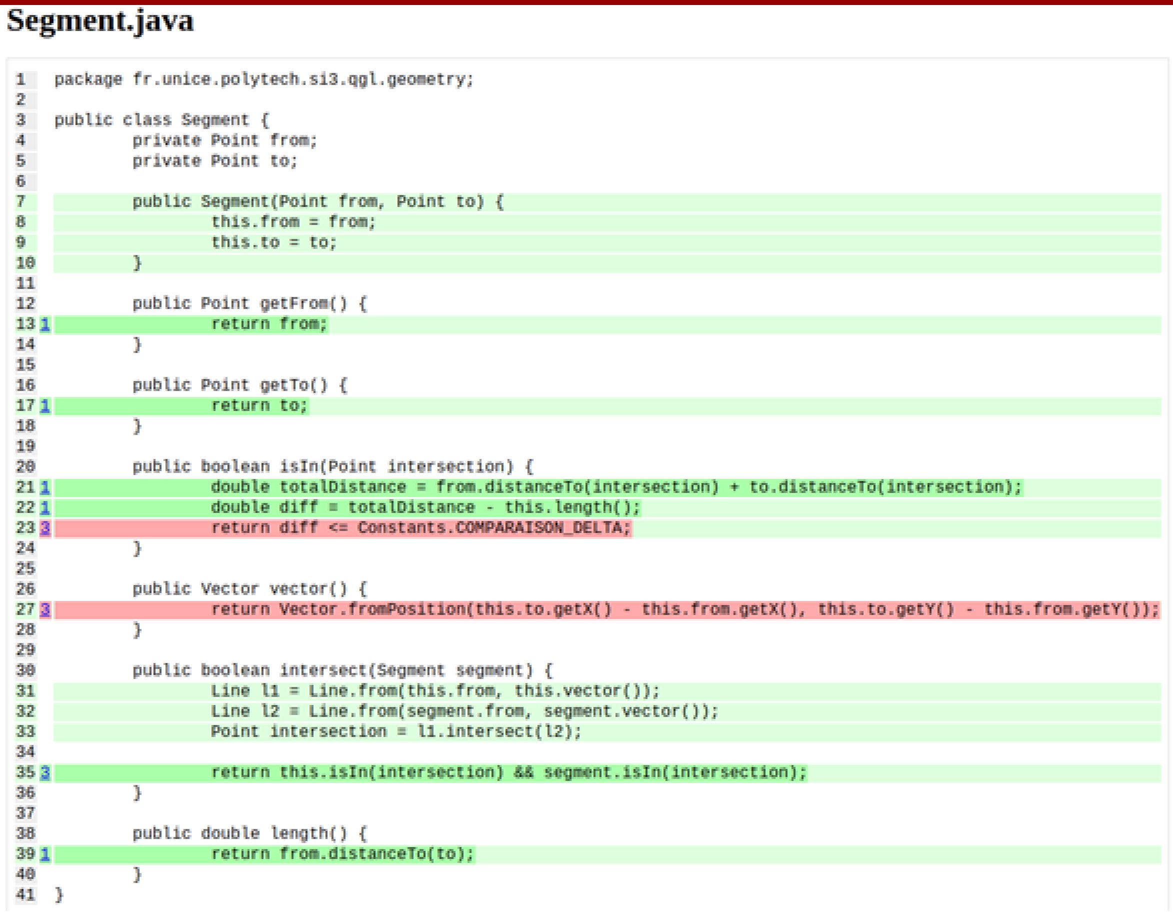
Number of Classes	Line Coverage		Mutation Coverage	
7	97%	229/236	86%	160/185

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
fr.unice.polytech.si3.qgl.geometry	5	98%	144/147	88%	119/136
fr.unice.polytech.si3.qgl.geometry.shapes	2	96%	85/89	84%	41/49

Report generated by PIT 1.4.11

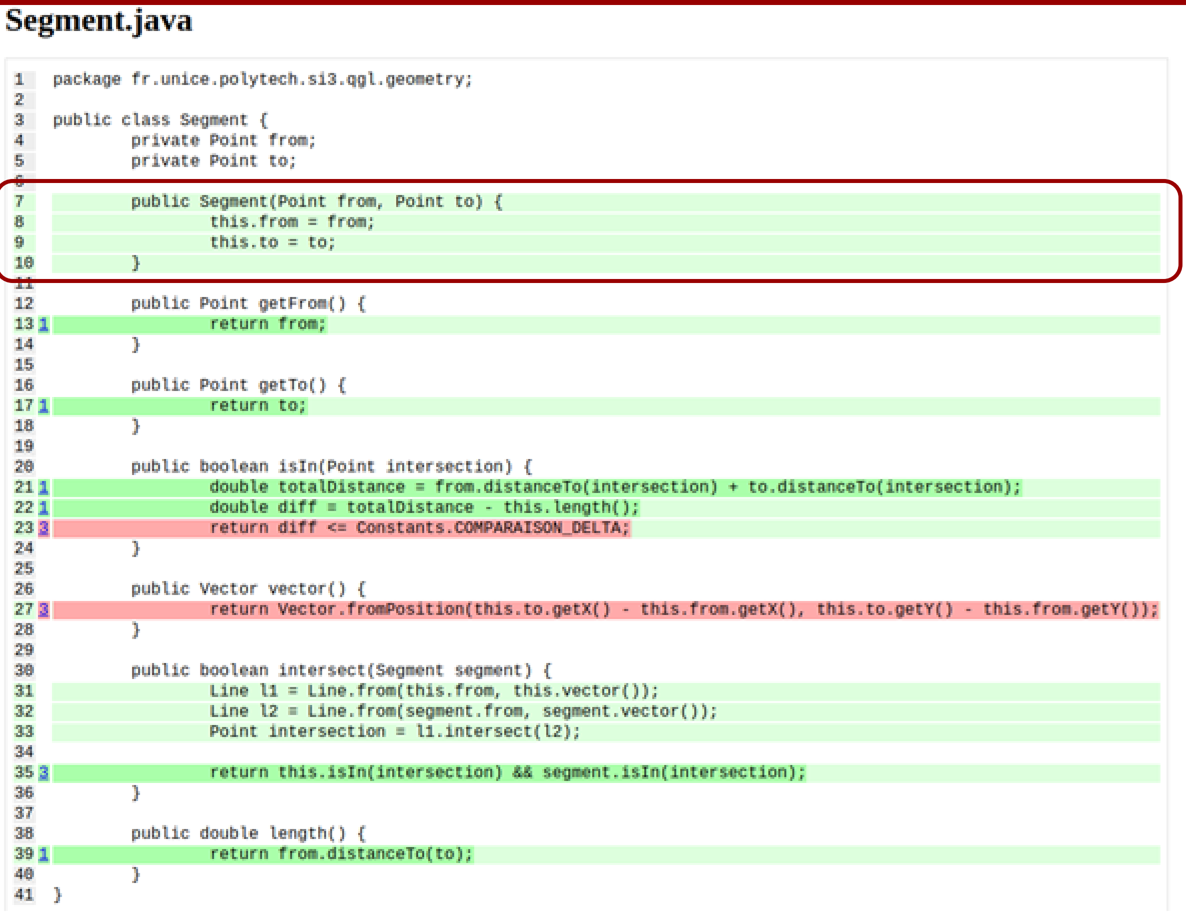
Understand the reports



Segment.java

```
package fr.unice.polytech.si3.qgl.geometry;

public class Segment {
        private Point from;
        private Point to;

        public Segment(Point from, Point to) {
                this.from = from;
                this.to = to;
        }

        public Point getFrom() {
                return from;
        }

        public Point getTo() {
                return to;
        }

        public boolean isIn(Point intersection) {
                double totalDistance = from.distanceTo(intersection) + to.distanceTo(intersection);
                double diff = totalDistance - this.length();
                return diff <= Constants.COMPARAISON_DELTA;
        }

        public Vector vector() {
                return Vector.fromPosition(this.to.getX() - this.from.getX(), this.to.getY() - this.from.getY());
        }

        public boolean intersect(Segment segment) {
                Line l1 = Line.from(this.from, this.vector());
                Line l2 = Line.from(segment.from, segment.vector());
                Point intersection = l1.intersect(l2);

                return this.isIn(intersection) && segment.isIn(intersection);
        }

        public double length() {
                return from.distanceTo(to);
        }
}
```

Understand the reports



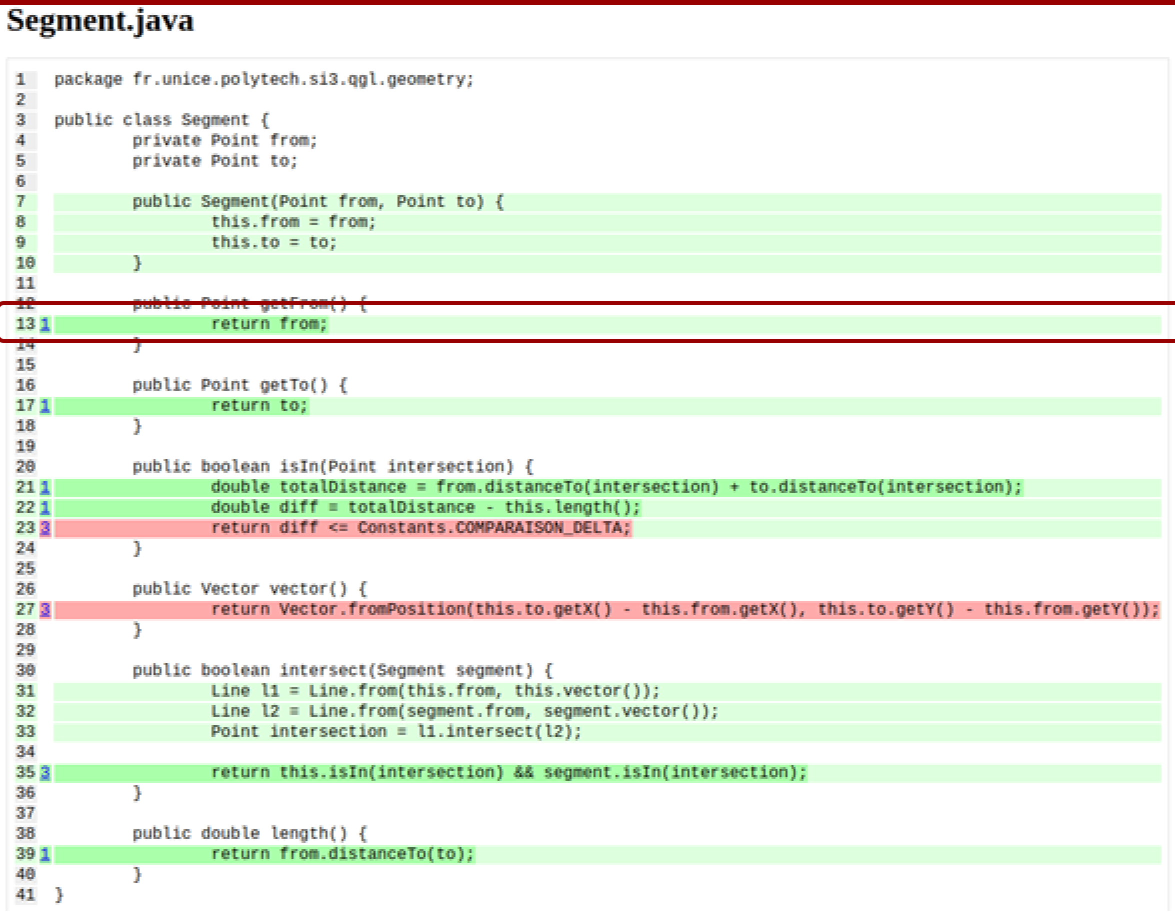
Segment.java

```
package fr.unice.polytech.si3.qgl.geometry;

public class Segment {
        private Point from;
        private Point to;

        public Segment(Point from, Point to) {
                this.from = from;
                this.to = to;
        }

        public Point getFrom() {
                return from;
        }

        public Point getTo() {
                return to;
        }

        public boolean isIn(Point intersection) {
                double totalDistance = from.distanceTo(intersection) + to.distanceTo(intersection);
                double diff = totalDistance - this.length();
                return diff <= Constants.COMPARAISON_DELTA;
        }

        public Vector vector() {
                return Vector.fromPosition(this.to.getX() - this.from.getX(), this.to.getY() - this.from.getY());
        }

        public boolean intersect(Segment segment) {
                Line l1 = Line.from(this.from, this.vector());
                Line l2 = Line.from(segment.from, segment.vector());
                Point intersection = l1.intersect(l2);

                return this.isIn(intersection) && segment.isIn(intersection);
        }

        public double length() {
                return from.distanceTo(to);
        }
}
```

Light green

Covered by test but with not mutation

Understand the reports



Segment.java

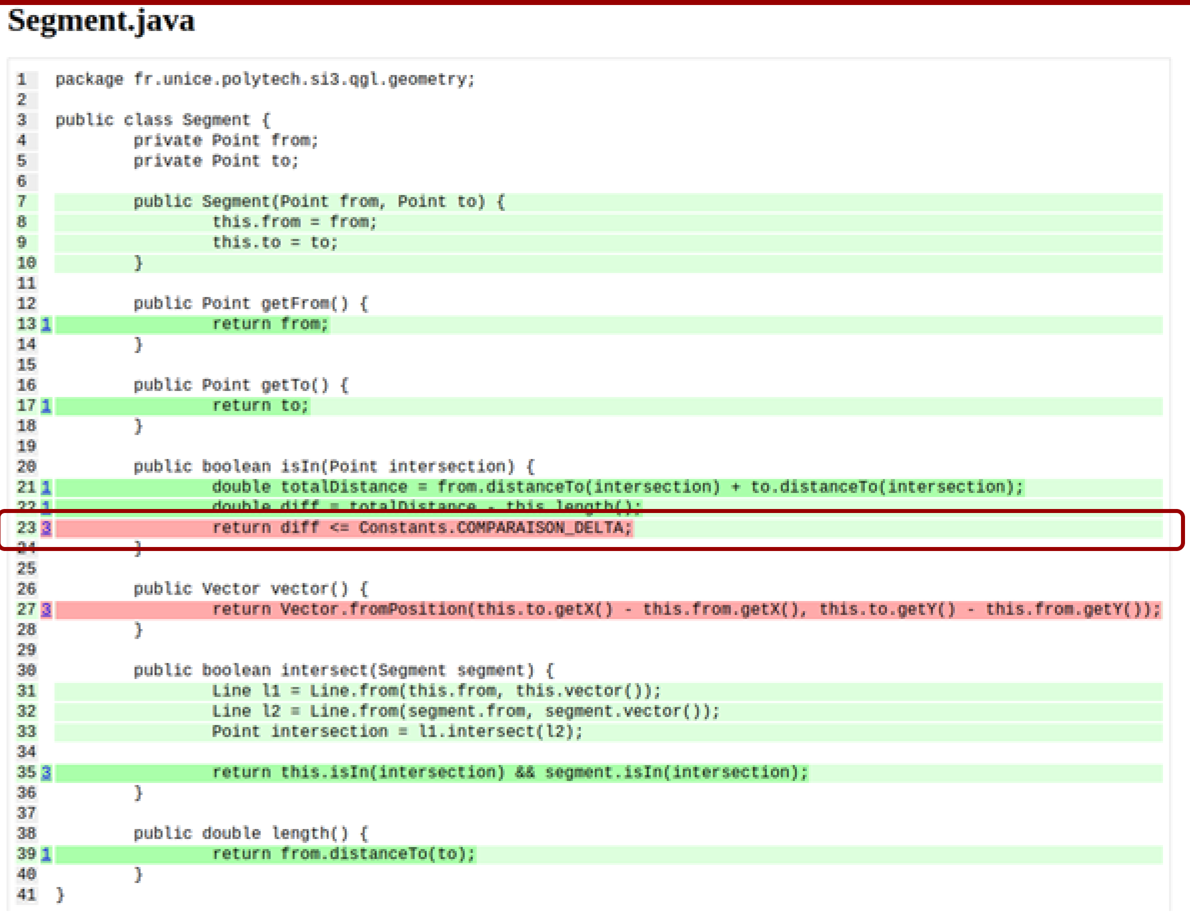
```
  package fr.unice.polytech.si3.qgl.geometry;

  public class Segment {
          private Point from;
          private Point to;

          public Segment(Point from, Point to) {
                  this.from = from;
                  this.to = to;
          }

          public Point getFrom() {
1                 return from;
          }

          public Point getTo() {
1                 return to;
          }

          public boolean isIn(Point intersection) {
1                 double totalDistance = from.distanceTo(intersection) + to.distanceTo(intersection);
1                 double diff = totalDistance - this.length();
3                 return diff <= Constants.COMPARAISON_DELTA;
          }

          public Vector vector() {
3                 return Vector.fromPosition(this.to.getX() - this.from.getX(), this.to.getY() - this.from.getY());
          }

          public boolean intersect(Segment segment) {
                  Line l1 = Line.from(this.from, this.vector());
                  Line l2 = Line.from(segment.from, segment.vector());
                  Point intersection = l1.intersect(l2);

3                 return this.isIn(intersection) && segment.isIn(intersection);
          }

          public double length() {
1                 return from.distanceTo(to);
          }
  }
```

<u>Dark green</u>

Covered by test and has mutations.

Mutations all killed.

Understand the reports



Segment.java

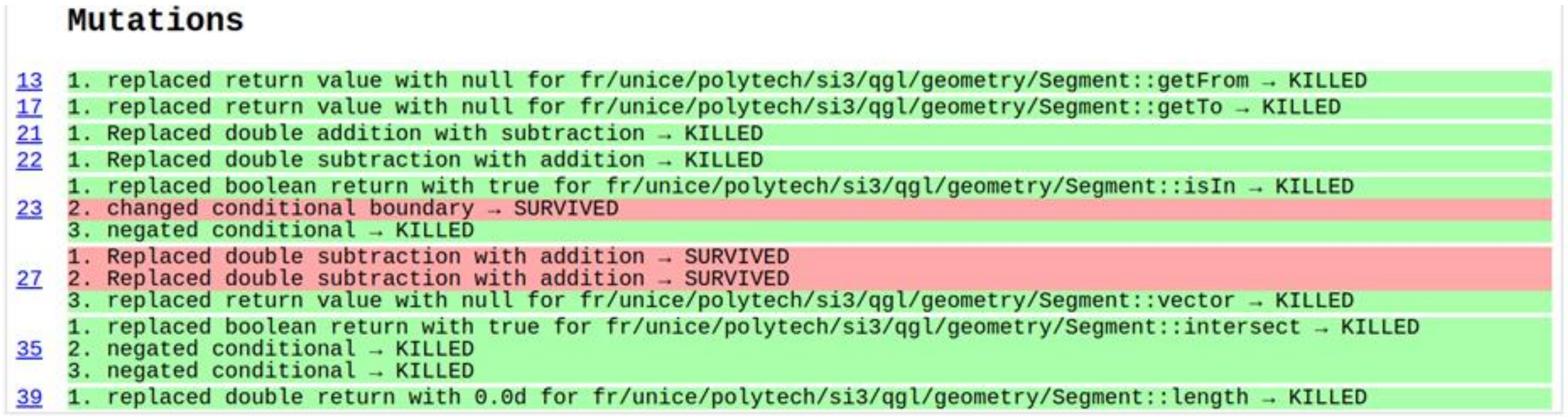
```
package fr.unice.polytech.si3.qgl.geometry;

public class Segment {
        private Point from;
        private Point to;

        public Segment(Point from, Point to) {
                this.from = from;
                this.to = to;
        }

        public Point getFrom() {
1               return from;
        }

        public Point getTo() {
1               return to;
        }

        public boolean isIn(Point intersection) {
1               double totalDistance = from.distanceTo(intersection) + to.distanceTo(intersection);
1               double diff = totalDistance - this.length();
3               return diff <= Constants.COMPARAISON_DELTA;
        }

        public Vector vector() {
3               return Vector.fromPosition(this.to.getX() - this.from.getX(), this.to.getY() - this.from.getY());
        }

        public boolean intersect(Segment segment) {
                Line l1 = Line.from(this.from, this.vector());
                Line l2 = Line.from(segment.from, segment.vector());
                Point intersection = l1.intersect(l2);

3               return this.isIn(intersection) && segment.isIn(intersection);
        }

        public double length() {
1               return from.distanceTo(to);
        }
}
```

Dark pink

Covered by test and has mutations.

Some mutations survived.

Understand the reports



```
   Mutations

13 1. replaced return value with null for fr/unice/polytech/si3/qgl/geometry/Segment::getFrom → KILLED
17 1. replaced return value with null for fr/unice/polytech/si3/qgl/geometry/Segment::getTo → KILLED
21 1. Replaced double addition with subtraction → KILLED
22 1. Replaced double subtraction with addition → KILLED
   1. replaced boolean return with true for fr/unice/polytech/si3/qgl/geometry/Segment::isIn → KILLED
23 2. changed conditional boundary → SURVIVED
   3. negated conditional → KILLED
   1. Replaced double subtraction with addition → SURVIVED
27 2. Replaced double subtraction with addition → SURVIVED
   3. replaced return value with null for fr/unice/polytech/si3/qgl/geometry/Segment::vector → KILLED
   1. replaced boolean return with true for fr/unice/polytech/si3/qgl/geometry/Segment::intersect → KILLED
35 2. negated conditional → KILLED
   3. negated conditional → KILLED
39 1. replaced double return with 0.0d for fr/unice/polytech/si3/qgl/geometry/Segment::length → KILLED
```

4

What do we expect from you?

What do we expect from you?

Apply PITest in your project

What do we expect from you?

Apply PITest in your project

(it will be evaluated)

Q&A