Mutation Testing with PIT

Bülent Yilmaz

High Line Coverage

=>

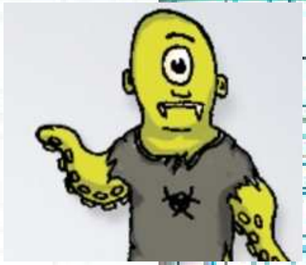
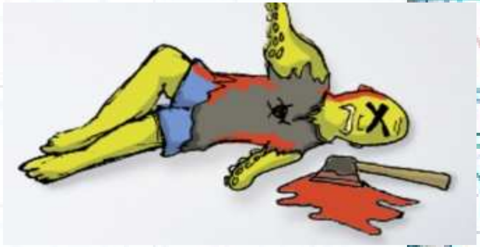
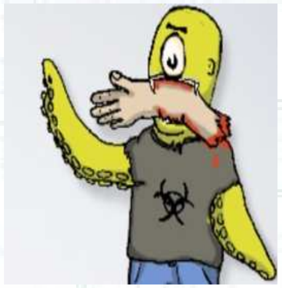
High QualityTests

?

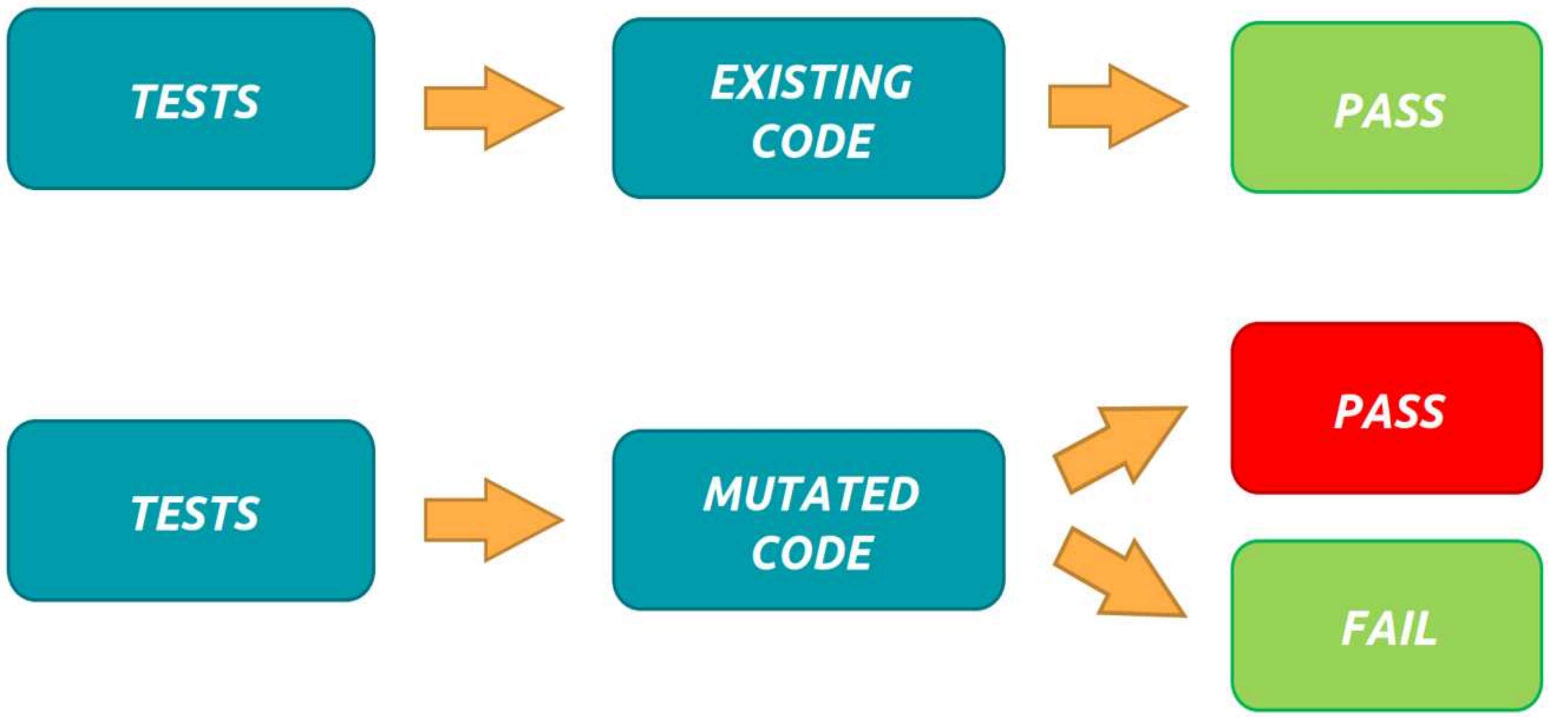
- Traditional test coverage (line, statement, branch, ...) measures only which code is executed by your tests
- does not check if your tests are able to detect faults
- => therefore only able to identify code that is not tested.

Mutation Tests

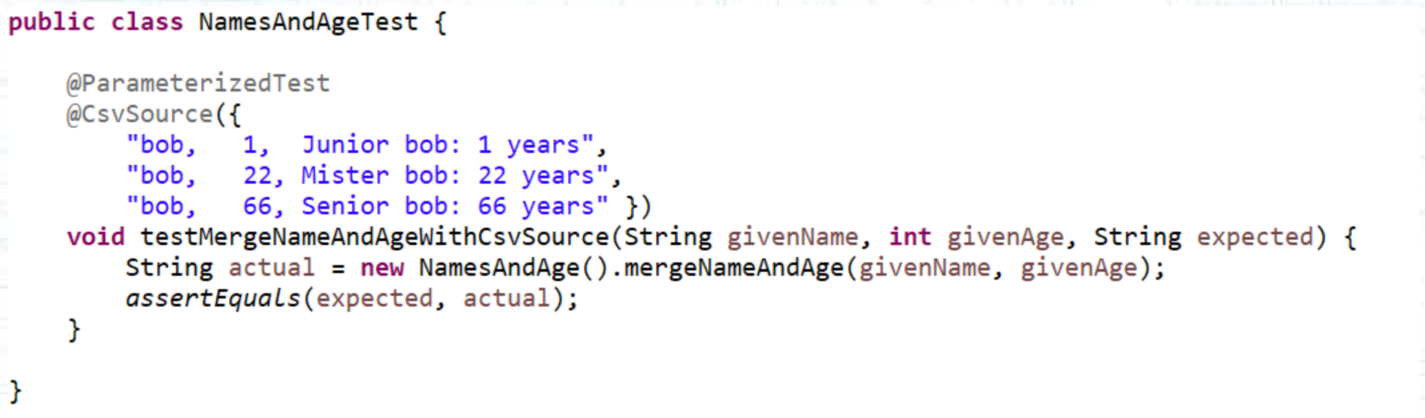
- Measure the quality of your tests
- Idea of Mutation Testing:
  - Seed artifical defects (bugs, mutations) into your application code
    - Bytecode manipulation
  - Check, whether yout tests find the defects
    - If a test fails, mutation is killed
    - If all tests pass, mutation survived

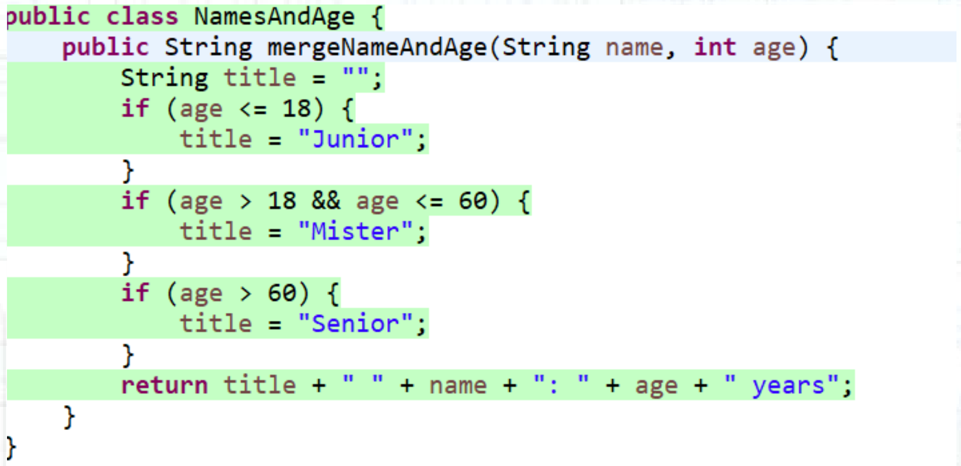
MUTATION TESTS



Example



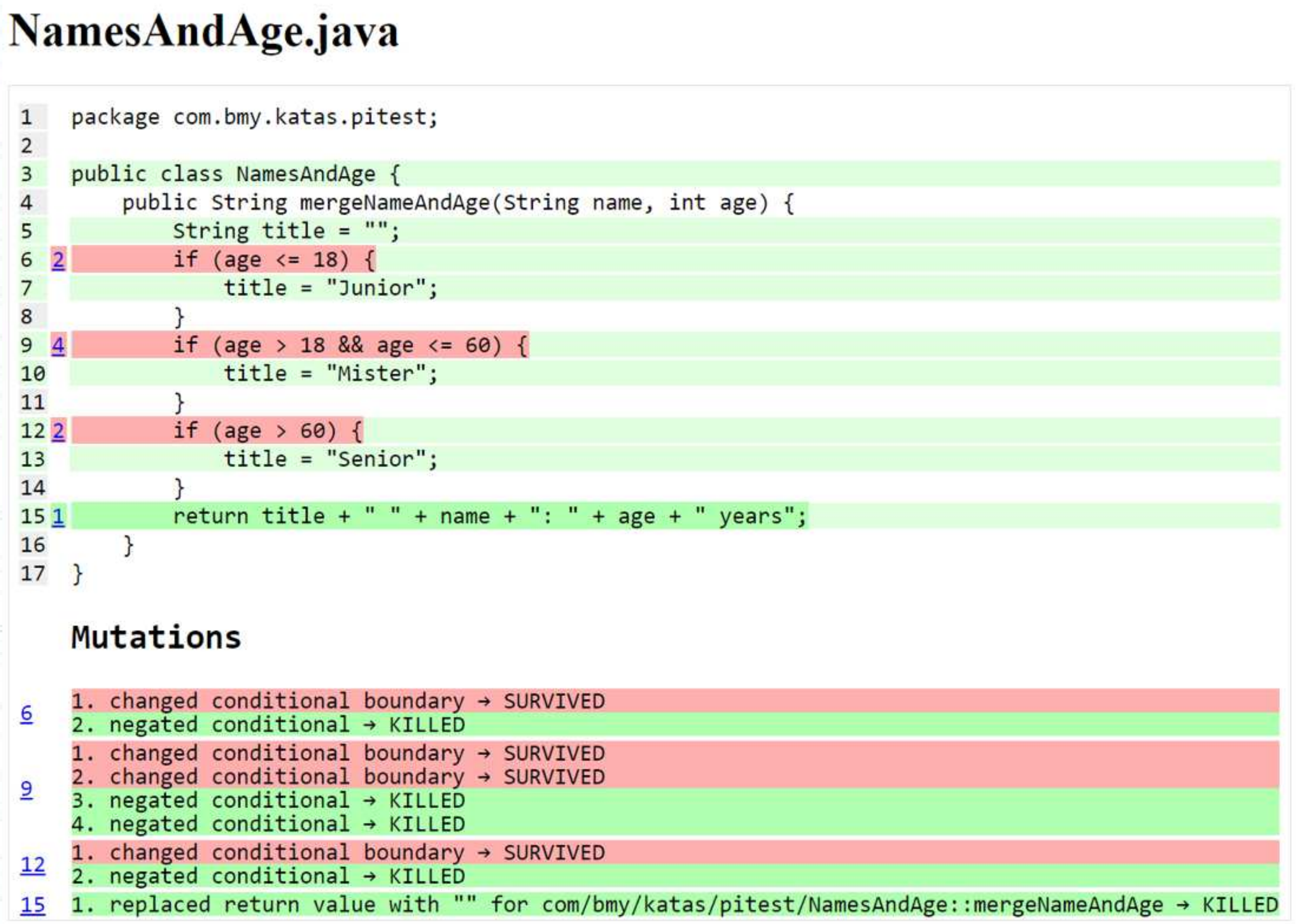
```java
public class NamesAndAgeTest {

    @ParameterizedTest
    @CsvSource({
        "bob,    1,  Junior bob: 1 years",
        "bob,    22, Mister bob: 22 years",
        "bob,    66, Senior bob: 66 years" })
    void testMergeNameAndAgeWithCsvSource(String givenName, int givenAge, String expected) {
        String actual = new NamesAndAge().mergeNameAndAge(givenName, givenAge);
        assertEquals(expected, actual);
    }

}
```



```java
public class NamesAndAge {
    public String mergeNameAndAge(String name, int age) {
        String title = "";
        if (age <= 18) {
            title = "Junior";
        }
        if (age > 18 && age <= 60) {
            title = "Mister";
        }
        if (age > 60) {
            title = "Senior";
        }
        return title + " " + name + ": " + age + " years";
    }
}
```

100 % Coverage
wow, my tests are f#@* awesome

But: look at the PIT report



NamesAndAge.java

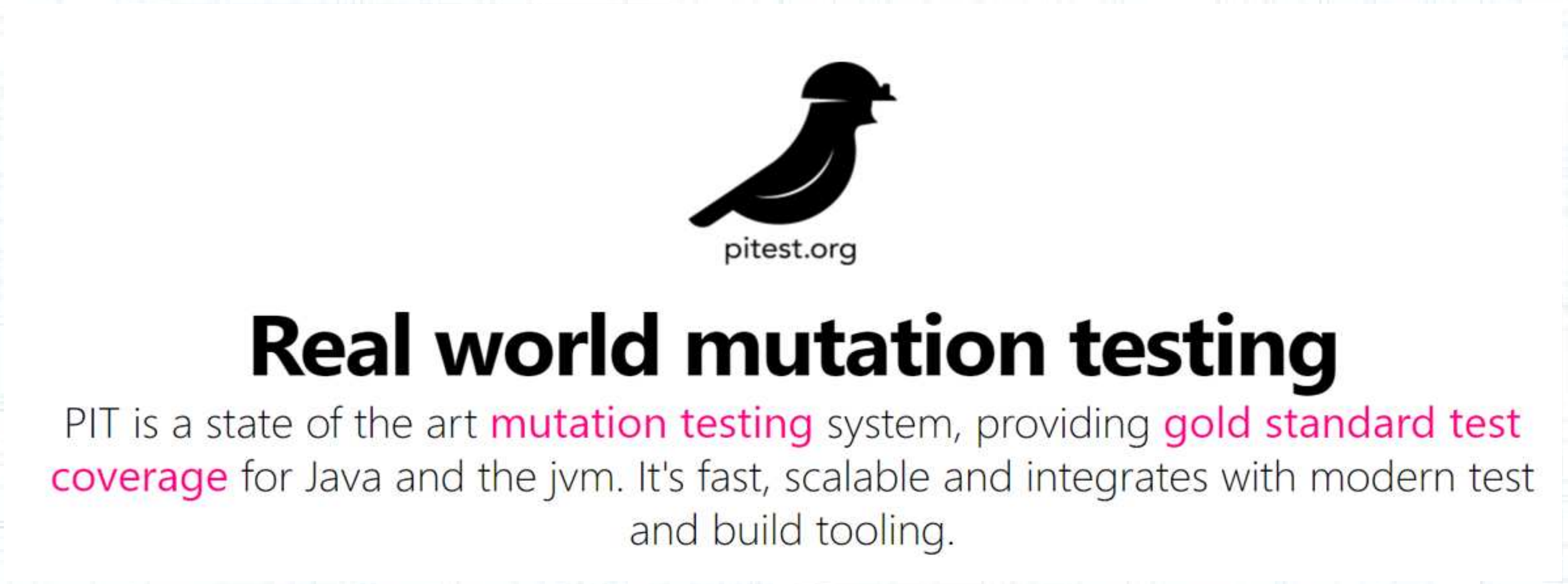
```
1     package com.bmy.katas.pitest;
2
3     public class NamesAndAge {
4         public String mergeNameAndAge(String name, int age) {
5             String title = "";
6  2          if (age <= 18) {
7                 title = "Junior";
8             }
9  4          if (age > 18 && age <= 60) {
10                title = "Mister";
11            }
12 2          if (age > 60) {
13                title = "Senior";
14            }
15 1          return title + " " + name + ": " + age + " years";
16        }
17    }
```

Mutations

```
6     1. changed conditional boundary → SURVIVED
      2. negated conditional → KILLED
9     1. changed conditional boundary → SURVIVED
      2. changed conditional boundary → SURVIVED
      3. negated conditional → KILLED
      4. negated conditional → KILLED
12    1. changed conditional boundary → SURVIVED
      2. negated conditional → KILLED
15    1. replaced return value with "" for com/bmy/katas/pitest/NamesAndAge::mergeNameAndAge → KILLED
```

=> Oh, I should also test boundaries

PIT



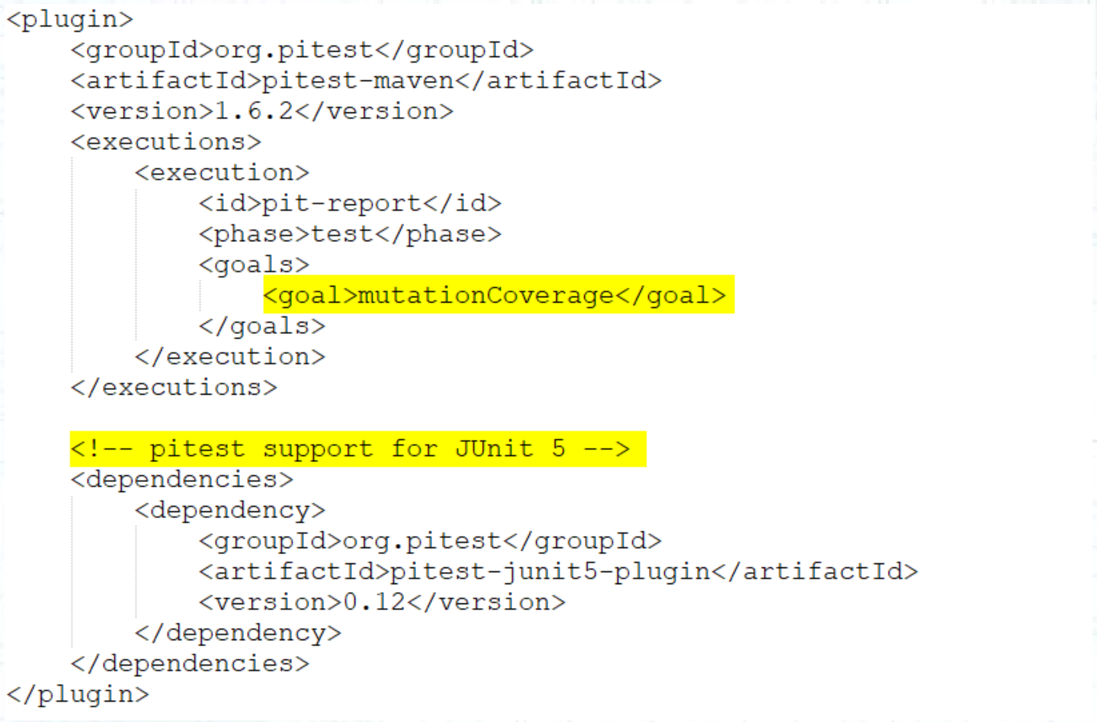
pitest.org

Real world mutation testing

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.

PIT: Maven

- pom.xml: add to build/plugins:

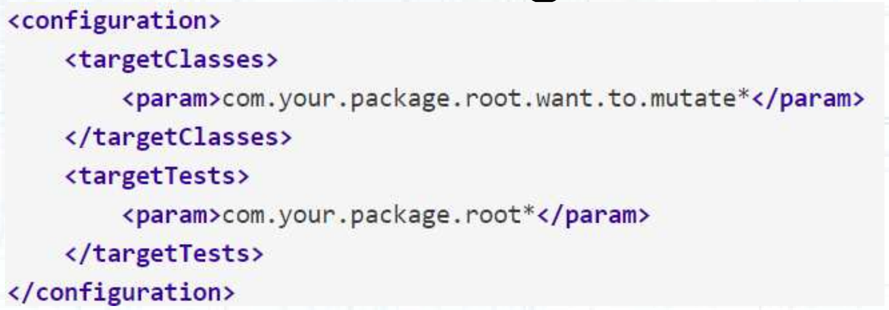


```xml
<plugin>
    <groupId>org.pitest</groupId>
    <artifactId>pitest-maven</artifactId>
    <version>1.6.2</version>
    <executions>
        <execution>
            <id>pit-report</id>
            <phase>test</phase>
            <goals>
                <goal>mutationCoverage</goal>
            </goals>
        </execution>
    </executions>

    <!-- pitest support for JUnit 5 -->
    <dependencies>
        <dependency>
            <groupId>org.pitest</groupId>
            <artifactId>pitest-junit5-plugin</artifactId>
            <version>0.12</version>
        </dependency>
    </dependencies>
</plugin>
```

- Generates HTML report: target/pit-reports/YYYYMMDDHHMI
- mutationCoverage goal can be run from the commandline:
  - mvn org.pitest:pitest-maven:mutationCoverage

Configuration

- By default PIT will mutate all classes in your project
- targetClasses, targetTests

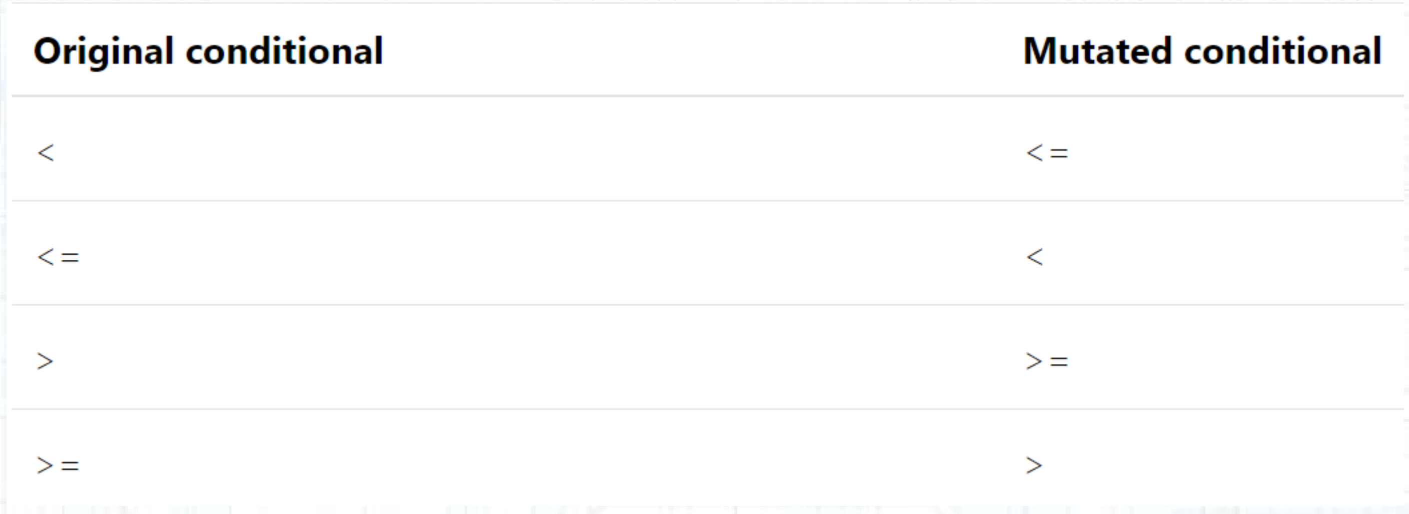


```
<configuration>
    <targetClasses>
        <param>com.your.package.root.want.to.mutate*</param>
    </targetClasses>
    <targetTests>
        <param>com.your.package.root*</param>
    </targetTests>
</configuration>
```

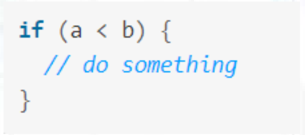
- reportsDirectory
- excludedClasses
- excludedTestClasses
- many other: https://pitest.org/quickstart/maven/

Mutators

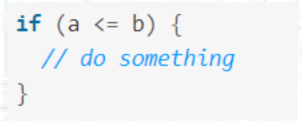
- PIT applies mutation operations (mutators) to your bytecode
- Conditionals Boundary Mutator
  replaces the relational operators <, <=, >, >=



Original conditional	Mutated conditional
<	<=
<=	<
>	>=
>=	>



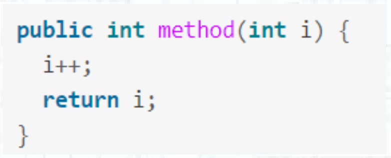
```
if (a < b) {
  // do something
}
```



```
if (a <= b) {
  // do something
}
```

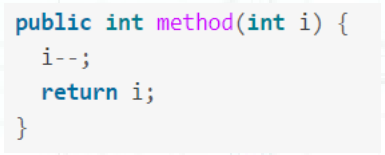
Mutators ...

- Increments Mutator (INCREMENTS)
  mutates increments, decrements and assignment increments and decrements of local variables



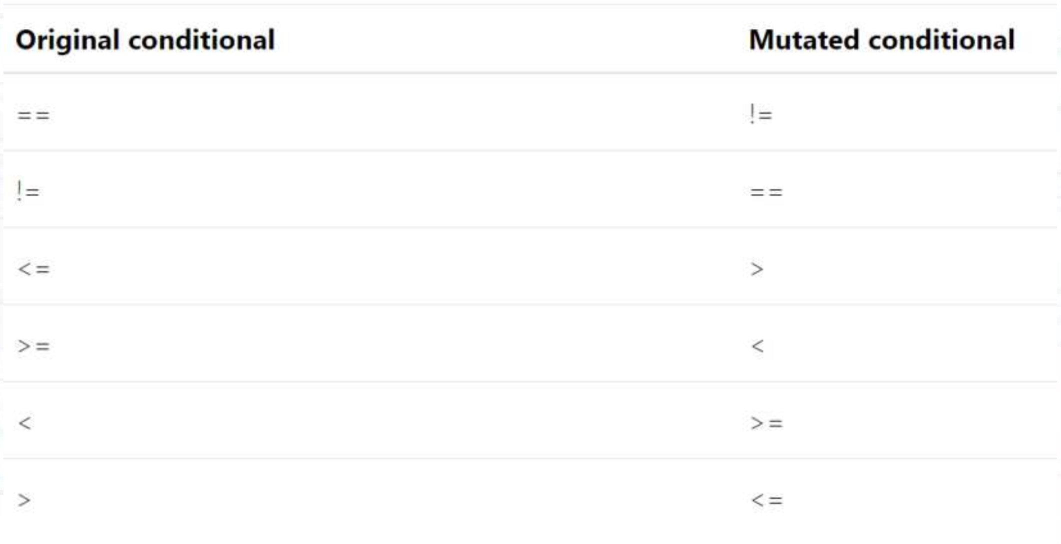
```
public int method(int i) {
  i++;
  return i;
}
```

==>



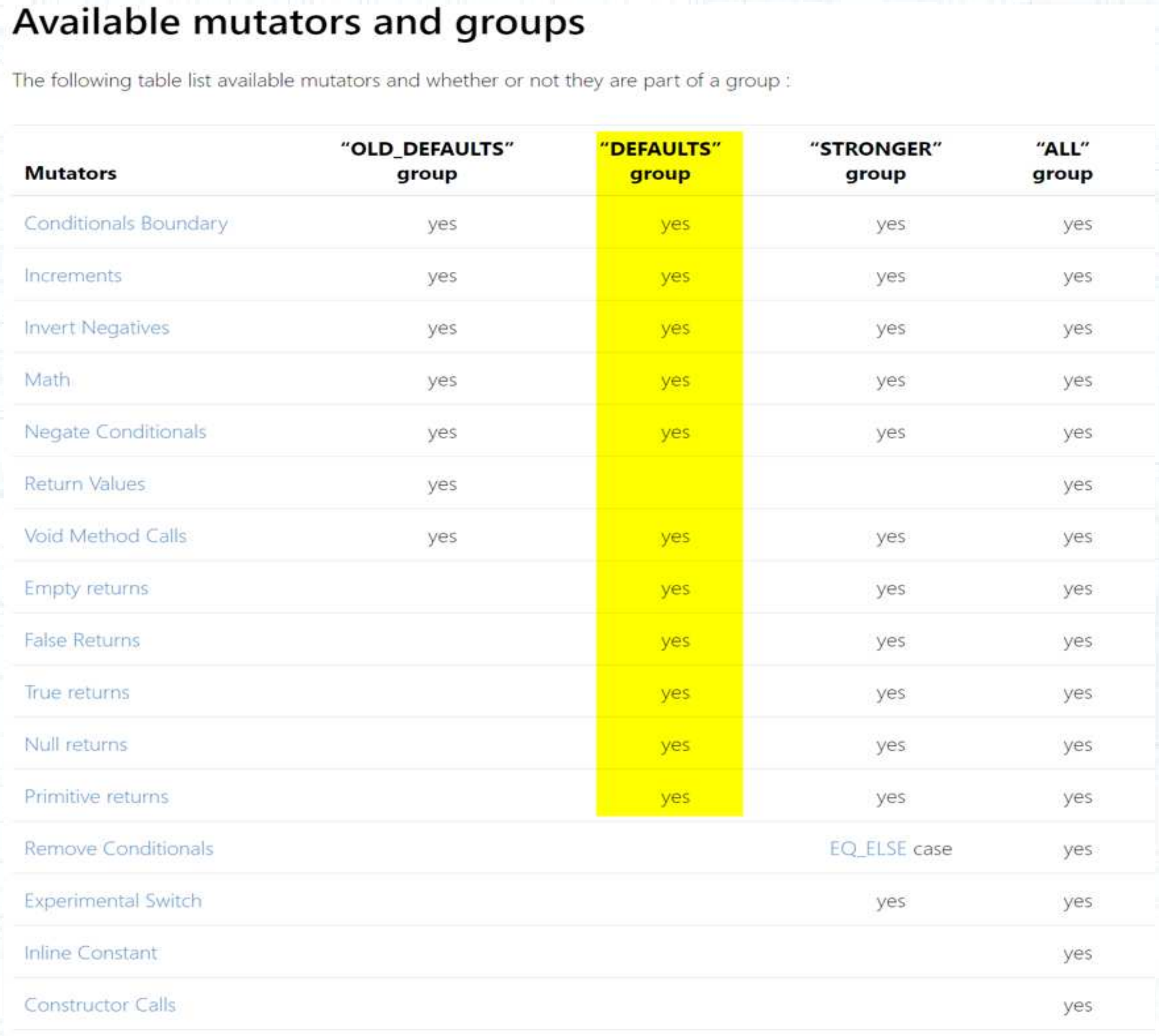
```
public int method(int i) {
  i--;
  return i;
}
```

- Negate Conditionals Mutator (NEGATE_CONDITIONALS)



Original conditional	Mutated conditional
==	!=
!=	==
<=	>
>=	<
<	>=
>	<=

Mutators…



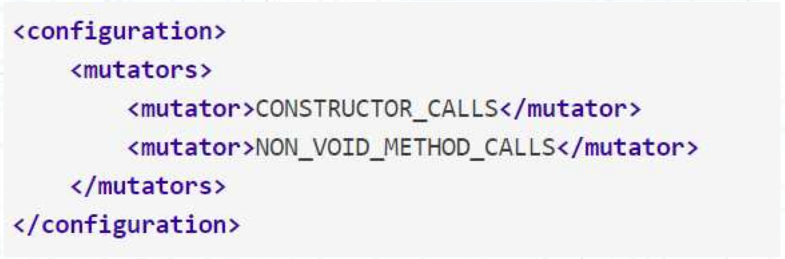
Available mutators and groups

The following table list available mutators and whether or not they are part of a group :

Mutators	"OLD_DEFAULTS" group	"DEFAULTS" group	"STRONGER" group	"ALL" group
Conditionals Boundary	yes	yes	yes	yes
Increments	yes	yes	yes	yes
Invert Negatives	yes	yes	yes	yes
Math	yes	yes	yes	yes
Negate Conditionals	yes	yes	yes	yes
Return Values	yes			yes
Void Method Calls	yes	yes	yes	yes
Empty returns		yes	yes	yes
False Returns		yes	yes	yes
True returns		yes	yes	yes
Null returns		yes	yes	yes
Primitive returns		yes	yes	yes
Remove Conditionals			EQ_ELSE case	yes
Experimental Switch			yes	yes
Inline Constant				yes
Constructor Calls				yes

https://pitest.org/quickstart/mutators/

Configure Active Mutators



```
<configuration>
    <mutators>
        <mutator>CONSTRUCTOR_CALLS</mutator>
        <mutator>NON_VOID_METHOD_CALLS</mutator>
    </mutators>
</configuration>
```

IDE support

- IntelliJ plugin: PIT intellij plugin
- Eclipse plugin: Pitclipse
  https://github.com/pitest/pitclipse

  Usage: Run As > PIT Mutation Test