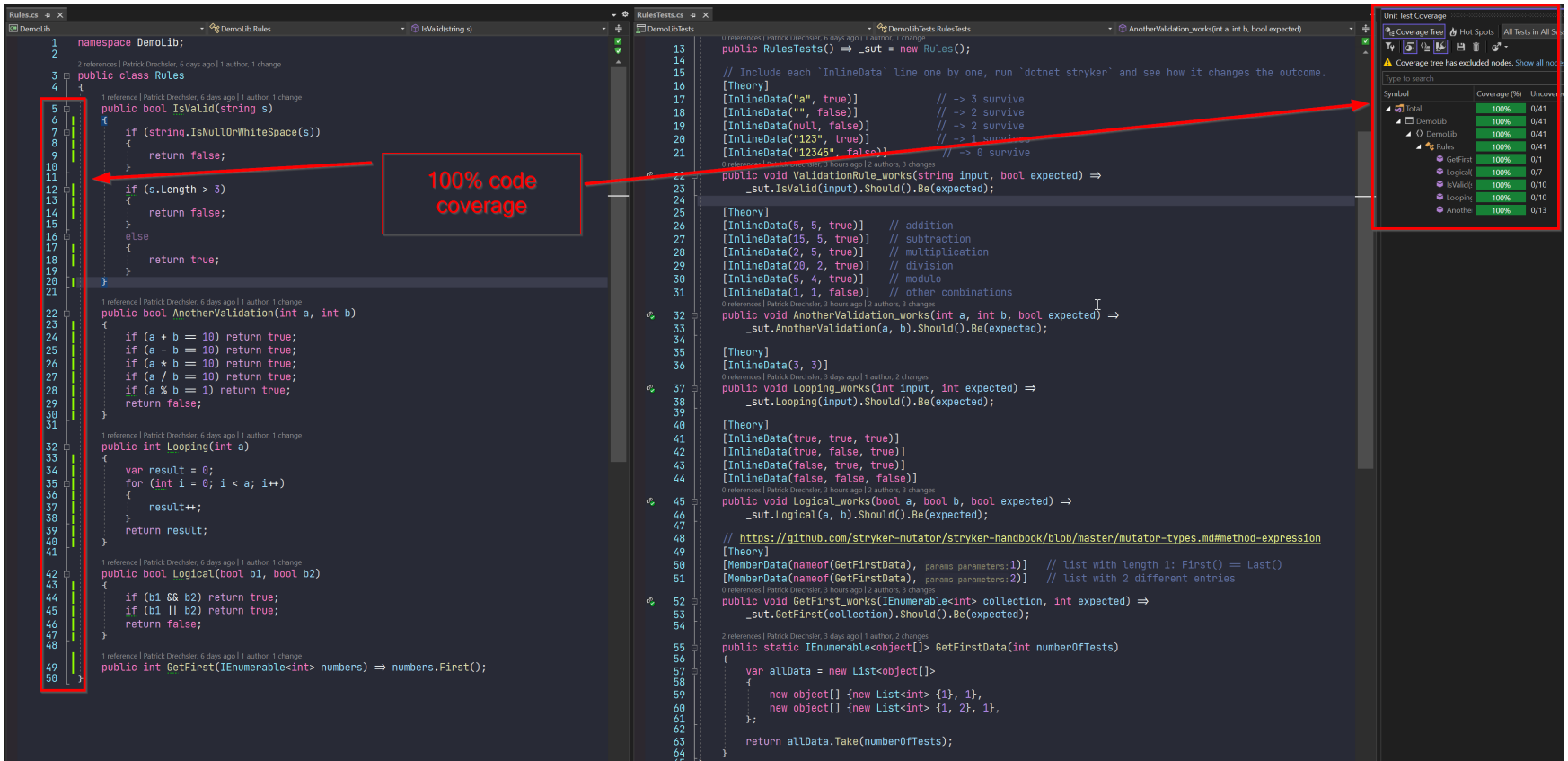
Mutation Testing

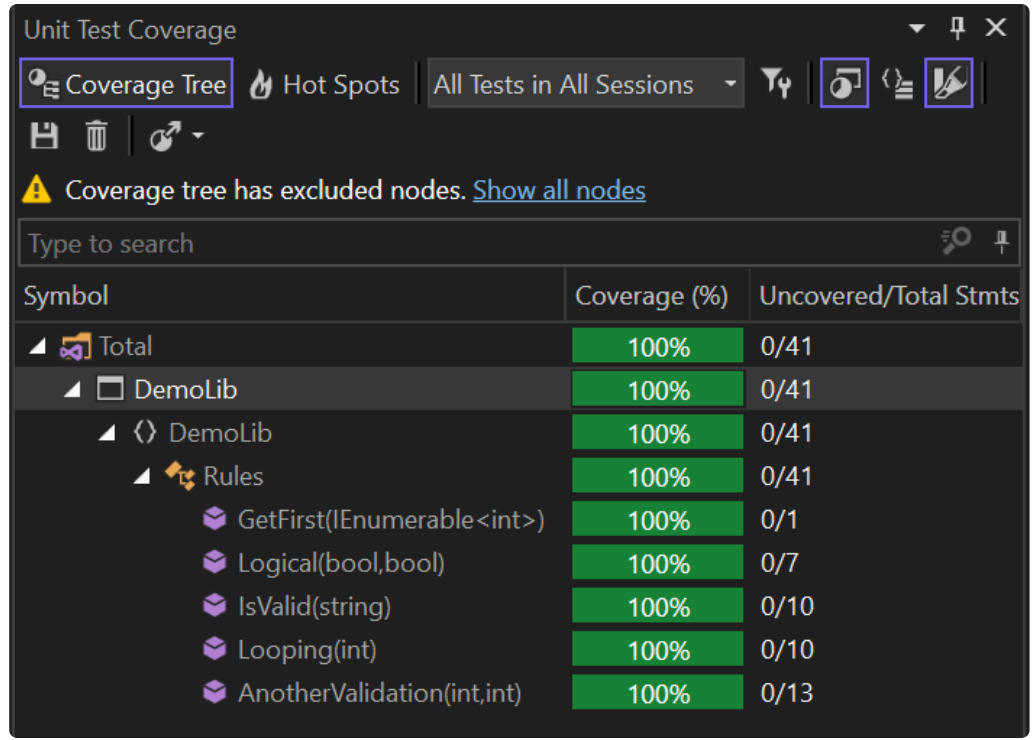
PATRICK DRECHSLER

Code Coverage 100%



Code Coverage 100%

what more do you want?



Example

```
public class Rules
{
  public bool IsValid(string s)
  {
    if (string.IsNullOrWhiteSpace(s))
    {
      return false;
    }

    if (s.Length > 3)
    {
      return false;
    }

    return true;
  }
}
```

```
public class RulesTests
{
  private readonly Rules _sut;

  public RulesTests() => _sut = new Rules();

  [Theory]
  [InlineData("", false)]
  [InlineData("a", true)]
  [InlineData("12345", true)]
  public void Validation_works(
    string input, bool expected) =>
    _sut.IsValid(input).Should().Be(expected);
}
```

- 100% coverage...
- but, are we covering all corner cases?

Let’s create some mutants!

Let’s change

```
if (s.Length > 3)
```

to

```
if (s.Length < 3)  // <- this is a "MUTANT"

if (s.Length >= 3) // <- this is another "MUTANT"

if (s.Length <= 3) // <- ...and another "MUTANT"
```

Do we still have the same code coverage?

Concept

- Production code is modified (by the mutation testing framework)
- Test suite is run

Did any mutants survive?

- If all mutants die, the test suite is fine 👌
- But if some mutants survive, the tests are not covering all cases 👀
  - 👉 take a closer look

Many mutation frameworks generate an interactive html report

Example mutations

https://stryker-mutator.io/docs/stryker-net/mutations/

Mutations

Stryker supports a variety of mutators, which are listed below. In parentheses the names of correspondent mutations are specified, which you might need for the `exclude-mutations` section of the configuration.

Do you have a suggestion for a (new) mutator? Feel free to create an issue!

Arithmetic Operators (*arithmetic*)


Live Demo

Other languages

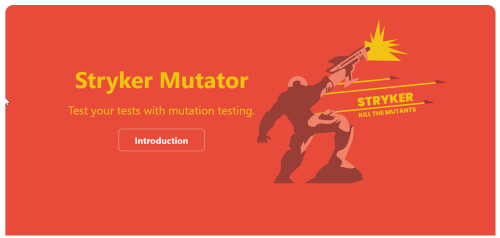
Frameworks are available for many languages:

- Java (using PIT)
- Scala (using Stryker4s)
- C# (using Stryker.NET)
- Javascript/Typescript (using StrykerJS)
- Python (using Cosmic Ray or mutmut)
- C/C++ (exist, I was to lazy to research...)
- Haskell (using MuCheck)
- ...

: Example project in the repo

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.



Getting started with Stryker

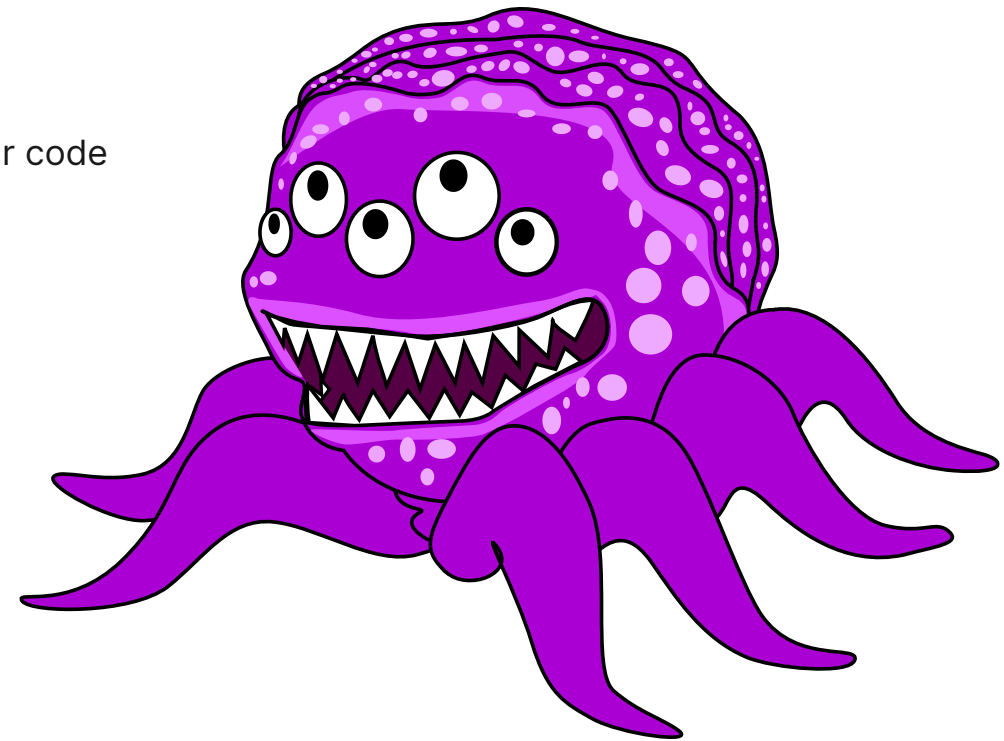
mutmut - python mutation tester

Hackage :: [Package]

MuCheck: Automated Mutation Testing

Mutation Testing: Summary

- requires decent test coverage
- modifies production code to find corner cases
- requires a lot of resources: use deliberately!
- Don’t include it in your CI/CD pipeline
- Use it as an exploratory tool to find bugs in your code
- Use it to find critical bugs in your code



The End

Any Questions?