

# Property based testing in practice

Marek Třešňák

# Automated tests...

- ... are fast to execute.
- ... don't forget about specific use-cases.
- ... prevent regression.
- ... serve as documentation.

# Example based testing

1. Write down some input values.
2. Write down expected output values.
3. Execute code and check the result is as expected.

## Example - List Reverse

{ 1, 2, 3, 10 }



Reverse()

{ 10, 3, 2, 1 }

# Test with a collection of integer values

```
[Test]
public void ReversalOfListOfIntsWorks()
{
    var list = new List<int> { 1, 2, 3, 10 };
    var reversedList = Reverse(list);
    CollectionAssert.AreEqual(
        expected: new List<int> { 10, 3, 2, 1 },
        actual: reversedList
    );
}
```

## + Test with an empty collection

```
[Test]
public void ReversalOfListOfIntsWorks()
{
    var list = new List<int> { 1, 2, 3, 10 };
    var reversedList = Reverse(list);
    CollectionAssert.AreEqual(
        expected: new List<int> { 10, 3, 2, 1 },
        actual: reversedList
    );
}
```

```
[Test]
public void ReversalOfEmptyListOfIntsWorks()
{
    var list = new List<int>();
    var reversedList = Reverse(list);
    CollectionAssert.AreEqual(
        expected: new List<int>(),
        actual: reversedList
    );
}
```

# Refactoring

```
[Test]
public void ReversalOfListOfIntsWorks()
{
    TestListReversal(
        list: new List<int> { 1, 2, 3, 10 },
        expected: new List<int> { 10, 3, 2, 1 }
    );
}
```

```
[Test]
public void ReversalOfEmptyListOfIntsWorks()
{
    TestListReversal(
        list: new List<int>(),
        expected: new List<int>()
    );
}
```

2 usages

```
private void TestListReversal<T>(List<T> list, List<T> expected)
{
    var reversedList = Reverse(list);
    CollectionAssert.AreEqual(
        expected: expected,
        actual: reversedList
    );
}
```

# Property based testing

1. Describe the input.
2. Describe the properties of the code under test.
3. Run test with randomly generated values and verify the properties.



$\text{Reverse}(\text{Reverse}(\text{list})) = \text{list}$

{ 1, 2, 3, 10 }

Reverse()

{ 10, 3, 2, 1 }

Reverse()

{ 1, 2, 3, 10 }

# First generative test

```
[Test]
public void DoubleReversalOfListReturnsOriginalList()
{
    var random = new Random();
    for (var i = 0; i < 100; i++)
    {
        var list =
            Enumerable.Range(0, random.Next(1000))
                .Select(_ => random.Next())
                .ToList();
        var reversedList = Reverse(list);
        var doubleReverseList = Reverse(reversedList);
        CollectionAssert.AreEqual(
            expected: list,
            actual: doubleReverseList
        );
    }
}
```

# First generative test

```
DoubleReversalOfListReturnsOriginalList (.NETCoreApp, Version=v3.1) [34 ms] System.ArgumentOutOfRangeException: Index was out of range. Must be non-negative and less than the size of the collection. (Parameter 'index')
PropertyBasedTesting.ListReverse.DoubleReversalOfListReturnsOriginalList

System.ArgumentOutOfRangeException : Index was out of range. Must be non-negative and less
than the size of the collection. (Parameter 'index')
  at System.Collections.Generic.List`1.get_Item(Int32 index)
  at PropertyBasedTesting.ListReverse.Reverse[T](List`1 input) in
C:\Users\marek\OneDrive\mews\presentace\property-based-testing\samples
\PropertyBasedTesting\PropertyBasedTesting\ListReverse.cs:line 18
  at PropertyBasedTesting.ListReverse.DoubleReversalOfListReturnsOriginalList() in
C:\Users\marek\OneDrive\mews\presentace\property-based-testing\samples
\PropertyBasedTesting\PropertyBasedTesting\ListReverse.cs:line 71
```

# Different flavors of QuickCheck


- |                       |                 |                 |
|-----------------------|-----------------|-----------------|
| 1. C                  | 15. Go          | 29. Python      |
| 2. C++                | 16. Io          | 30. R           |
| 3. Chicken            | 17. Java        | 31. Racket      |
| 4. Clojure            | 18. JavaScript  | 32. Ruby        |
| 5. Common Lisp        | 19. Julia       | 33. Rust        |
| 6. Coq                | 20. Logtalk     | 34. Scala       |
| 7. D                  | 21. Lua         | 35. Scheme      |
| 8. Elm                | 22. Mathematica | 36. Smalltalk   |
| 9. Elixir             | 23. Objective-C | 37. Standard ML |
| 10. Erlang            | 24. OCaml       | 38. Swift       |
| 11. F#                | 25. Perl        | 39. TypeScript  |
| 12. C#                | 26. Prolog      | 40. Whiley      |
| 13. Visual Basic .NET | 27. PHP         |                 |
| 14. Factor            | 28. Pony        |                 |

# Generative test using the library

```
[Test]
public void DoubleReversalOfListReturnsOriginalList()
{
    Prop.ForAll<List<int>>(list =>
    {
    });
}
```

# Generative test using the library

```
[Test]
public void DoubleReversalOfListReturnsOriginalList()
{
    Prop.ForAll<List<int>>(list =>
    {
    });
}
```



✓ Output (.NETCoreApp, Version=v3.1) [173 ms]

97:  
seq [0; 0; 0; 0; ...]

98:  
seq [3; 0; -3; -5; ...]

99:  
seq [0; 1; 34; 0; ...]

# Generative test using the library

```
[Test]
public void DoubleReversalOfListReturnsOriginalList()
{
    Prop.ForAll<List<int>>(list =>
    {
        var reversedList = Reverse(list);
        var doubleReversedList = Reverse(reversedList);
        return list.SequenceEqual(doubleReversedList);
    });
}
```

# Generative test using the library

```
[Test]
public void DoubleReversalOfListReturnsOriginalList()
{
    Prop.ForAll<List<int>>(list =>
    {
        var reversedList = Reverse(list);
        var doubleReversedList = Reverse(reversedList);
        return list.SequenceEqual(doubleReversedList);
    }).QuickCheckThrowOnFailure();
}
```



# Generative test using the library

```
[Test]
public void DoubleReversalOfListReturnsOriginalList()
{
    Prop.ForAll<List<int>>(list =>
    {
        var reversedList = Reverse(list);
        var doubleReversedList = Reverse(reversedList);
        return list.SequenceEqual(doubleReversedList);
    }).QuickCheckThrowOnFailure();
}
```

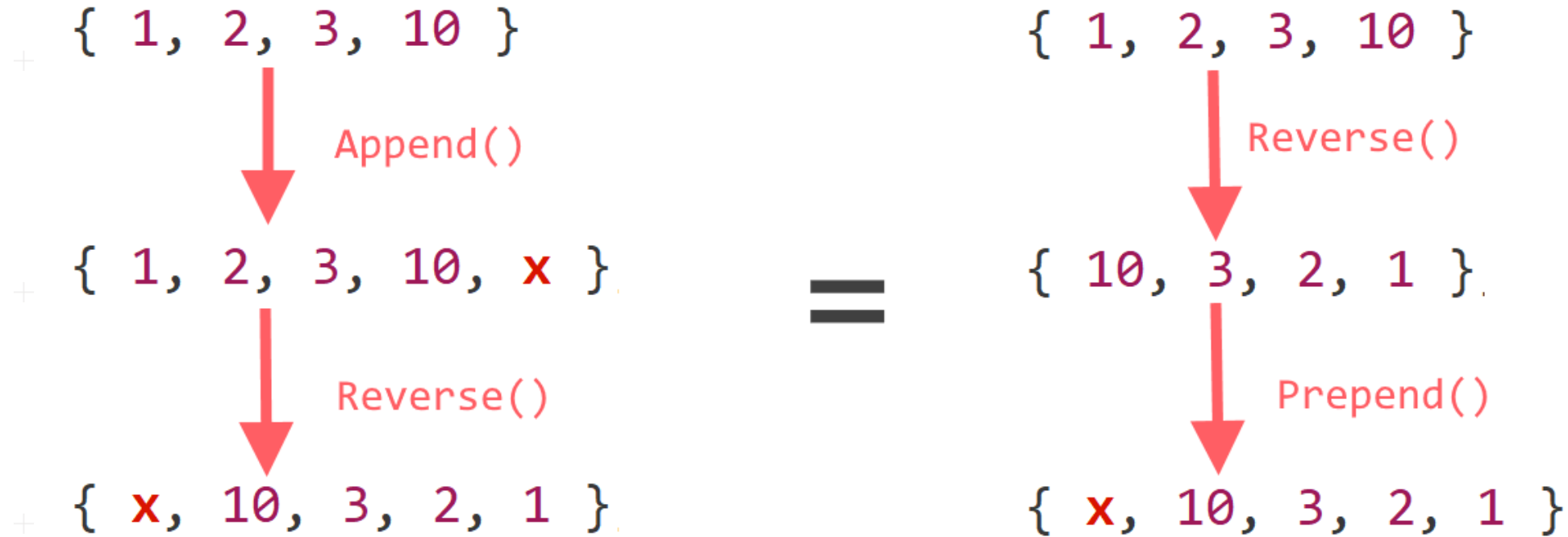
✓ DoubleReversalOfListReturnsOriginalList [173 ms] Success

## Implementation of Reverse()

```
private List<T> Reverse<T>(List<T> input)
{
    return input;
}
```

✓ DoubleReversalOfListReturnsOriginalList [173 ms] Success

$\text{Reverse}(\text{Append}(\text{list})) = \text{Prepend}(\text{Reverse}(\text{list}))$



# Multiple auto-generated values

```
[Test]
public void AppendThenReverseIsEqualToReverseThenPrepend()
{
    Prop.ForAll(Arb.From<int>(), Arb.From<List<int>>(), (element, list) =>
    {
    });
}
```

# Test Reverse(Append) = Prepend(Reverse)

```
[Test]
public void AppendThenReverseIsEqualToReverseThenPrepend()
{
    Prop.ForAll(Arb.From<int>(), Arb.From<List<int>>(), (element, list) =>
    {
        var appendThenReverse = Reverse(Append(list, element));
        var reverseThenPrepend = Prepend(Reverse(list), element);
        return appendThenReverse.SequenceEqual(reverseThenPrepend);
    }).QuickCheckThrowOnFailure();
}
```

# Test Reverse(Append) = Prepend(Reverse)

```
[Test]
public void AppendThenReverseIsEqualToReverseThenPrepend()
{
    Prop.ForAll(Arb.From<int>(), Arb.From<List<int>>(), (element, list) =>
    {
        var appendThenReverse = Reverse(Append(list, element));
        var reverseThenPrepend = Prepend(Reverse(list), element);
        return appendThenReverse.SequenceEqual(reverseThenPrepend);
    }).QuickCheckThrowOnFailure();
}
```

❌ AppendThenReverseIsEqualToReverseThenPrepend [203 ms] Failed: System.Exception : Falsifiable,

## Working implementation of Reverse()

```
private List<T> Reverse<T>(List<T> input)
{
    var output = new List<T>(input);
    output.Reverse();
    return output;
}
```

- ✓ AppendThenReverselsEqualToReverseThenPrepend [130 ms] Success
- ✓ DoubleReversalOfListReturnsOriginalList [71 ms] Success

# Test Reverse(Append) = Prepend(Reverse)

AppendThenReverseIsEqualToReverseThenPrepend (.NETCoreApp,Version=v3.1) [199 ms] System.Exception : Falsifiable, after 3 tests (3 shrinks) (4277081541671057107,14413507071265457431)

PropertyBasedTesting.ListReverse.AppendThenReverseIsEqualToReverseThenPrepend

System.Exception : Falsifiable, after 3 tests (3 shrinks) (4277081541671057107, 14413507071265457431)

Last step was invoked with size of 4 and seed of (8928640929221858429, 1429398716816633359):

Original:

-3

seq [0]

Shrunk:

1

seq [0]



# Shrinker

AppendThenReversesEqualToReverseThenPrepend (.NETCoreApp,Version=v3.1) [199 ms] System.Exception : Falsifiable, after 3 tests (3 shrinks) (4277081541671057107,14413507071265457431)

PropertyBasedTesting.ListReverse.AppendThenReverseIsEqualToReverseThenPrepend

System.Exception : Falsifiable, after 3 tests (3 shrinks) (4277081541671057107,  
14413507071265457431)

Last step was invoked with size of 4 and seed of (8928640929221858429,  
1429398716816633359):

Original:

-3

seq [0]

Shrunk:

1

seq [0]

# Seed

AppendThenReverseIsEqualToReverseThenPrepend (.NETCoreApp, Version=v3.1) [199 ms] System.Exception : Falsifiable, after 3 tests (3 shrinks) (4277081541671057107,14413507071265457431)

PropertyBasedTesting.ListReverse.AppendThenReverseIsEqualToReverseThenPrepend

System.Exception : Falsifiable, after 3 tests (3 shrinks) (4277081541671057107,  
14413507071265457431)

Last step was invoked with size of 4 and seed of (8928640929221858429,  
1429398716816633359):

Original:

-3

seq [0]

Shrunk:

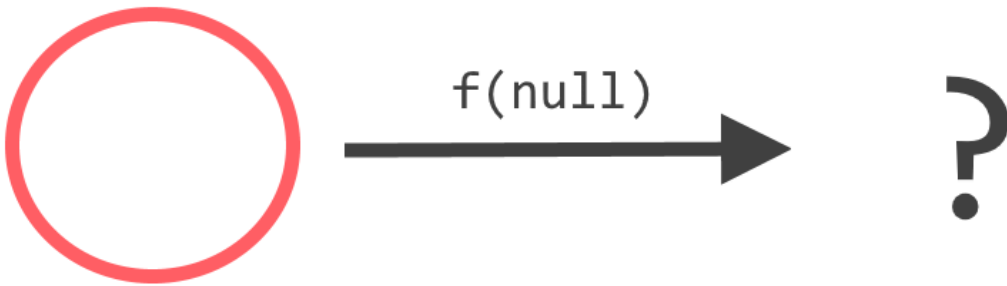
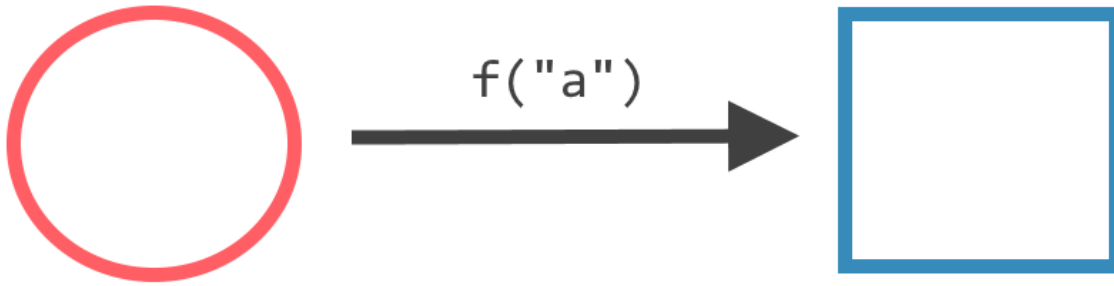
1

seq [0]

# Useful testing patterns

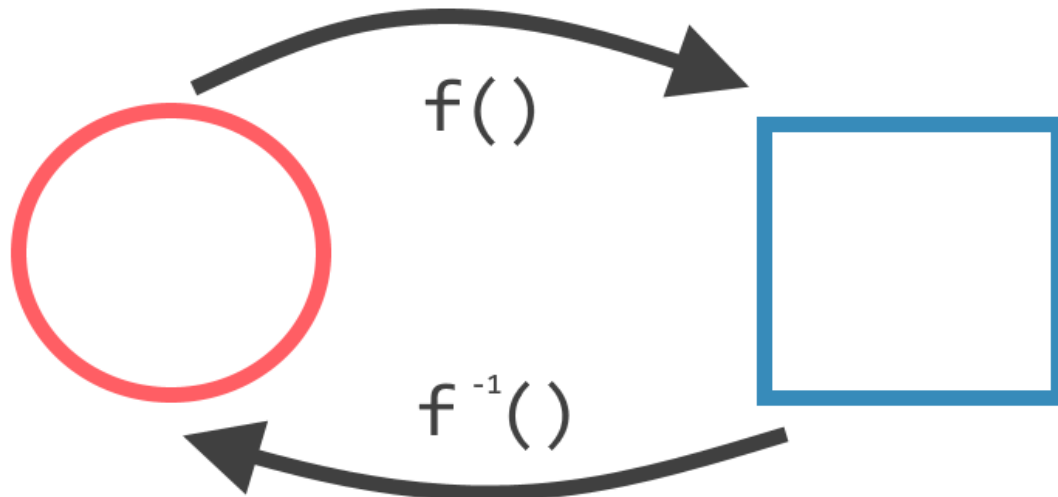
# Fuzz testing

- Checking that the code doesn't crash.



# Inverse function

- Encode / Decode
- Serialize / Deserialize
- Write / Read



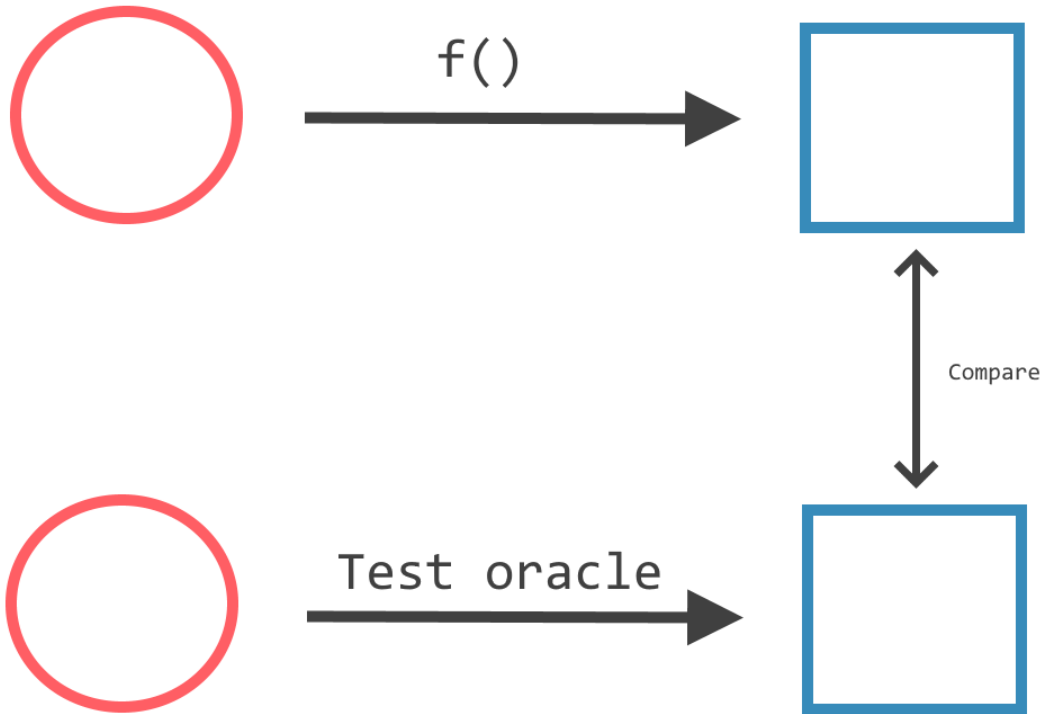
# Idempotence

- `List.Distinct()`
- `String.Trim()`



# Test oracle

- New vs old implementation.
- Naive vs optimized implementation.



# Summary

- It's often beneficial to combine EBT and PBT.
- Example based tests are usually easier to understand.
- Few generative test can replace many example based tests.
- Generative tests are more likely to discover edge-cases.



# Useful resources

- [FSharpForFunAndProfit.com](https://FSharpForFunAndProfit.com)
- [en.wikipedia.org/wiki/QuickCheck](https://en.wikipedia.org/wiki/QuickCheck)
- [github.com/marektresnak/talks](https://github.com/marektresnak/talks)



Questions?

- [marek.tresnak@outlook.com](mailto:marek.tresnak@outlook.com)
- [github.com/MewsSystems/developers](https://github.com/MewsSystems/developers)

MEWS