AS YOU MEANT IT…

NFluent: the craft you needed
to boost your TDD

Brown Bag Lunch - 2014

@tpierrain  @cyrdup

WHO ARE WE?

former webExpert
coding architect
curious
geek
DDD
usecase driven
NET
XP
lowlatency systems
TDD addict
blogger

Performance
68000
CPP Thread
IO Socket
XP TDD
threading
DDD distributed
NET blogger
functional RAFTiNG
Agile

use case driven (blog)

Many Cores (blog)

UNIT TESTS FROM THE TRENCHES…

COMMENTS?

IDEAL WORLD

HARD FACT #1

Tests assertions are **needlessly difficult to write**

```
[Test]
public void NUnit()
{
    var array = new int[] { 45, 43, 54, 666 };
    Assert.That(array, Is.Not.Property("Length").EqualTo(1));
    Assert.That(array, Has.No.Property("Length").GreaterThan(3));
```

```
(decimal expected, decimal actual):void
(decimal expected, decimal actual, string message):void
(decimal expected, decimal actual, string message, params object[] args):void
(double expected, double actual, double delta):void
(double expected, double actual, double delta, string message):void
(double expected, double actual, double delta, string message, params object[] args):void
(double expected, double? actual, double delta):void
(double expected, double? actual, double delta, string message):void
(double expected, double? actual, double delta, string message, params object[] args):
void
(int expected, int actual):void
(int expected, int actual, string message):void
(int expected, int actual, string message, params object[] args):void
  Verifies that two ints are equal. If they are not, then an AssertionException is thrown.
  expected: The expected value
(long expected, long actual):void
(long expected, long actual, string message):void
(long expected, long actual, string message, params object[] args):void
(object expected, object actual):void
(object expected, object actual, string message):void
(object expected, object actual, string message, params object[] args):void
(uint expected, uint actual):void
(uint expected, uint actual, string message):void
(uint expected, uint actual, string message, params object[] args):void
(ulong expected, ulong actual):void
(ulong expected, ulong actual, string message):void
```

```
[Test]
public void NUnitSu
{
    string heroes =
    Assert.AreEqual()
}
```

CONSEQUENCES

error-prone
reluctance to write tests
relevance of tests?

#1: WE DESERVE NO BRAINER ASSERTIONS!

HARD FACT #2

Test intentions may be **difficult to grasp**

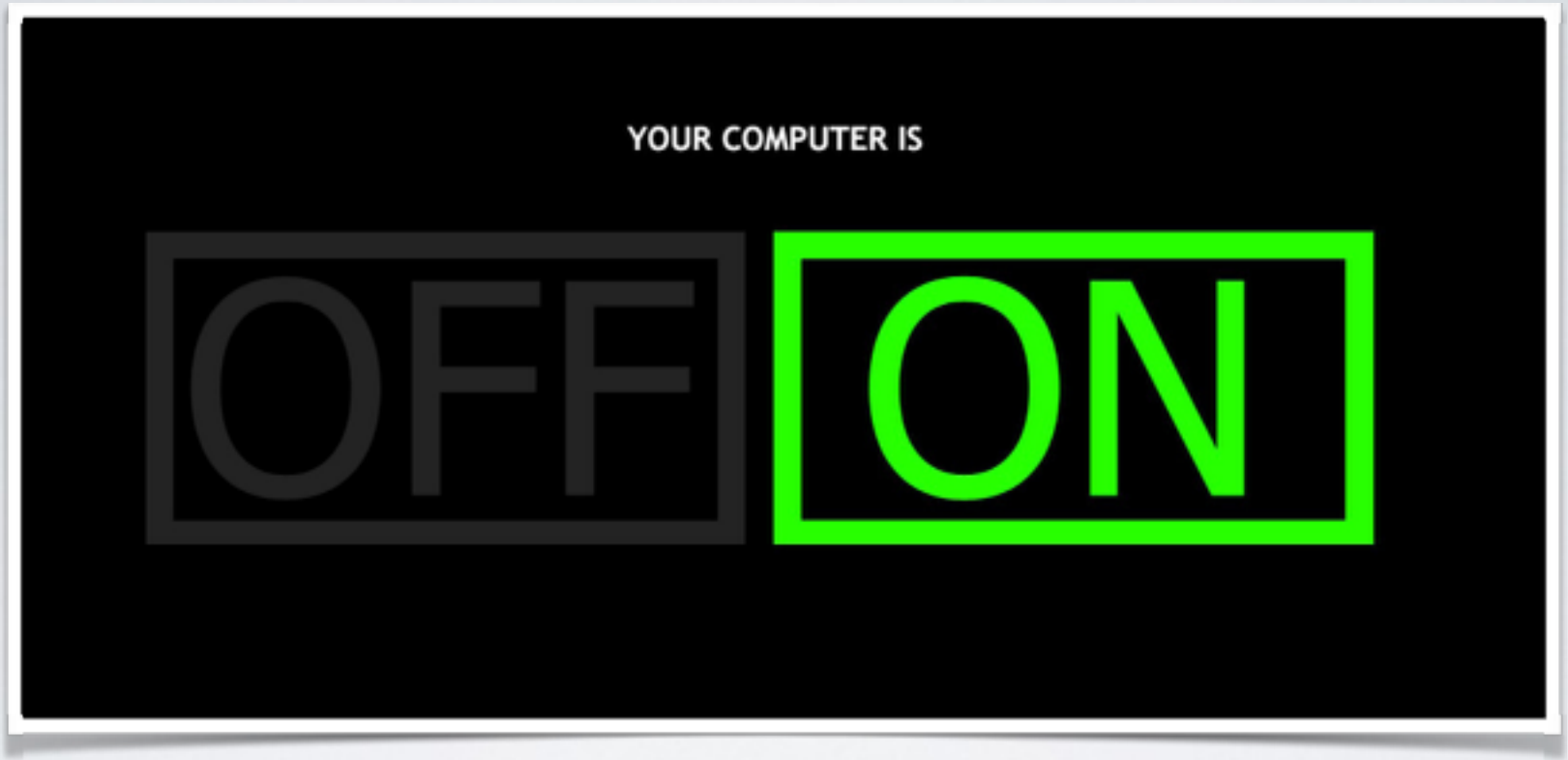
```
[Test]
public void Test3()
{
    var args = GenerateArgs();

    IDictionary<string, string> src = CommandLineArgsSettingSource.BuildParametersDico(args);

    Assert.IsTrue(src.ContainsKey("param1"));
    Assert.IsTrue(src.ContainsKey("param2"));
    Assert.IsTrue(src["param2"] == "value");
}
```

CONSEQUENCES

tests are **hard to maintain**

loss of trust in existing tests

Removal of failing tests?



#2: TESTS INTENTIONS SHOULD BE OBVIOUS!

HARD FACT #3

When reliable… **error messages are** often **unhelpful**

```
Expected: not property Length greater than 3
But was:  4

(was thrown here when running 'EnumerableRelatedTests.NUnitSucks')
Execution time is 0ms
(Left-click to show tests, Right-click for options)
```

CONSEQUENCES

Need to debug the failing test to understand
slowwww… TDD **feedback loop**
Poor efficiency?!?

```
[Test]
//[ExpectedException(typeof(FluentCheckException), ExpectedMessage = "\nThe chec
public void WeCanSeeTheDifferenceBewteenTwoDifferentObjectsThatHaveTheSameToStri
{
    Person dad = new Person { Name = "John" };
    Person son = new Child { Name = "John" };
    Check.That(son).IsEqualTo(dad);
```

```
NFluent.FluentCheckException :
The checked value is different from the expected one.
The checked value:
        [John] of type: [NFluent.Tests.Child]
The expected value:
        [John] of type: [NFluent.Tests.Person]
(was thrown 5 l
Execution time
(Left-click to sh
```

```
[Test]
//[ExpectedException(typeof(FluentCheckException), MatchType = MessageMatch.Regex, Expecte
public void WeCanAlsoSeeTheDifferenceBetweenTwoDifferentInstancesOfTheSameTypeWhithIdentica
{
    Person dad = new Person { Name = "John John" };
    Person uncle = new Person { Name = "John John" };

    Check.That(uncle).IsEqualTo(dad);
```

```
NFluent.FluentCheckException :
The checked value is different from the expected one.
The checked value:
        [John John] with HashCode: [35410979]
The expected value:
        [John John] with HashCode: [57416410]
(was thrown 5 levels deeper when running 'EqualRelatedTests.WeCanAlsoSeeTheDifferenceBetweenTwoDifferentInstancesOfTheSameTypeWhithIdenticalToString')
Execution time is 15ms
(Left-click to show tests, Right-click for options)
```

#3: WE WANT TRUSTWORTHY & HELPFUL ERROR MESSAGES!

TDD IS NOT EASY…

… but **NFluent will greatly help you**

UBIQUITOUS LANGUAGE

Fluent [ˈfluːənt]

adj

1. spoken or written with ease: *fluent French*.
2. able to speak or write smoothly, easily, or readily: *fluent in three languages*.
3. **smooth; easy; graceful**: *fluent motion*
4. flowing or capable of flowing; ***fluid***

[from Latin: *fluere* to flow, stream]

(source: http://www.thefreedictionary.com/fluent)

NFLUENT'S ANSWERS

#1: NO BRAINER ASSERTIONS!

A unique entry point for assertions (*Check.That*)

IntelliSense guided writing (a.k.a. the *super-duper-happy* dot xp)

#2: OBVIOUS TESTS INTENTIONS!

Natural language assertions

#3: TRUSTWORTHY & HELPFUL ERROR MESSAGES!

Non ambiguous, contextualized and explicit error messages

COMBINING THOSE 3 WISHES,

WE CAN REACH OUR IDEAL:

EFFICIENT TDD!

NFLUENT SAMPLES

```
[Test]
public void BetterOuBien()
{
    var alphabet = "abcdefghijklmnopqrstuvwxyz";
    Check.That(alphabet).Contains("i").And.StartsWith("abcd").And.IsInstanceOf<string>();

    var integers = new int[] { 1, 2, 3, 4, 5, 666 };
    Check.That(integers).Contains(3, 5, 666);

    var guitarHeroes = new[] { "Hendrix", "Paco de Lucia", "Django Reinhardt" };
    Check.That(guitarHeroes).ContainsExactly("Hendrix", "Paco de Lucia", "Django Reinhardt");

    var camus = new Person() { Name = "Camus" };
    var sartre = new Person() { Name = "Sartre" };
    Check.That(camus).IsInstanceOf<Person>().And.IsNotEqualTo(sartre);

    var heroes = "Batman and Robin";
    Check.That(heroes).StartsWith("Bat").And.Contains("Robin");

    var motivationalSaying = "Failure is mother of success.";
    Check.That(motivationalSaying).IsNotInstanceOf<int>();
}
```

DEMO

IN A FEW WORDS

- NFluent is
  - An OSS .NET library for fluent assertions
  - Independent and compliant with all major .NET test runners
  - Directly inspired by the awesome Java FEST assert library
  - Available on nuget gallery

BONUS FEATURES

- **A fully extensible model**
  - No need to wait the NFluent team to add your own assertions on any kind of 'value'
  - The opportunity to use your own ubiquitous language (DSL) from within your tests
- Some extra candies 
  - Like the *Properties* extension method for *IEnumerable*
  - The *HasFieldsWithSameValues(anonymous class)*

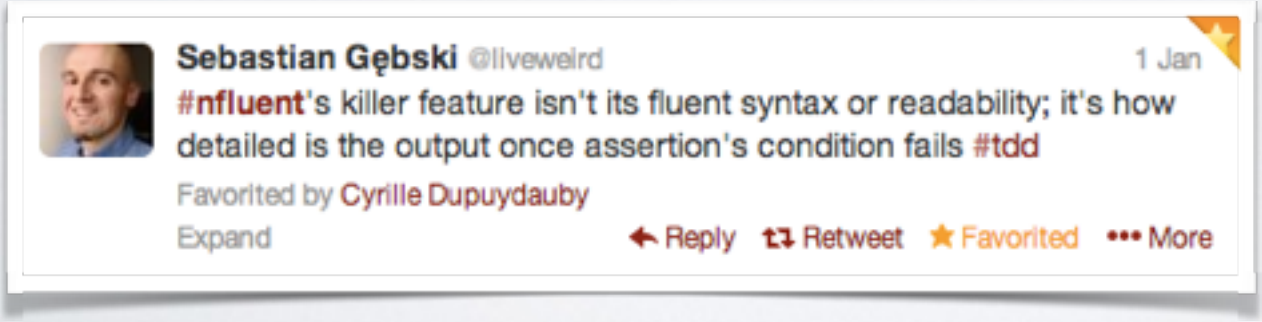
IN THE WILD

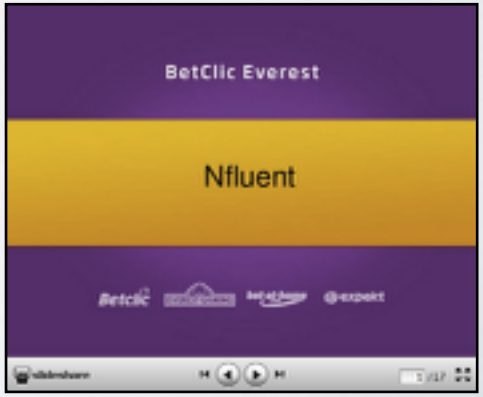
tpierrain

dupdob

rhwy

malat





FINAL QUESTION

xUnit.net

Nunit

mb·unit

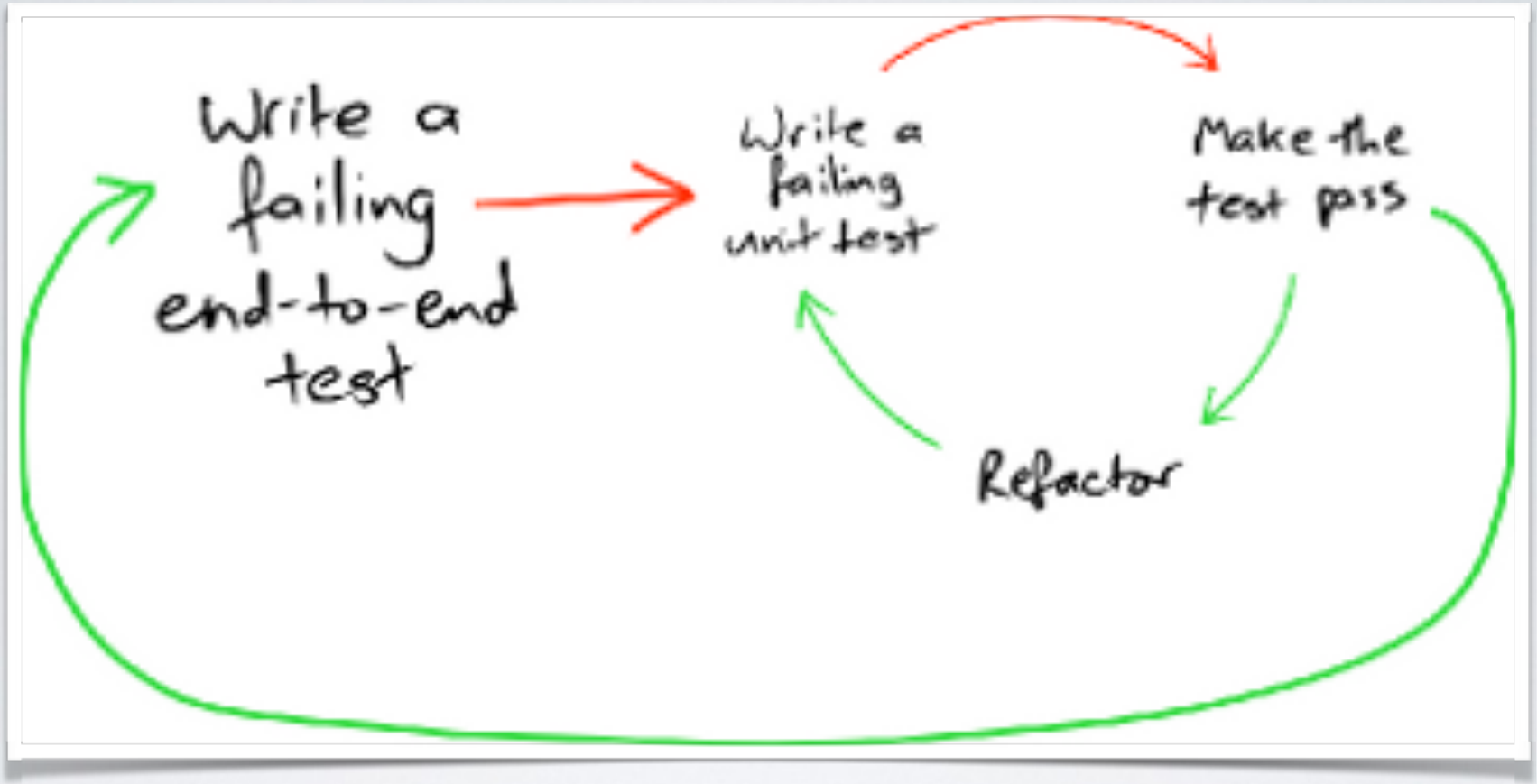
Does your test framework dictate the way you work, or is it the other way around?

Use NFluent!

Use NFluent!

http://www.n-fluent.net

APPENDIX



TDD: A WORKFLOW

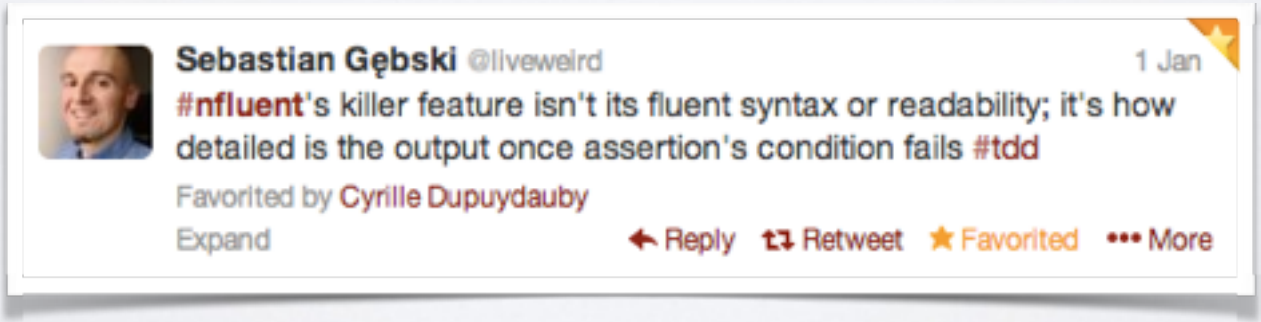
NFLUENT vs. OTHER ASSERTION LIBRARIES?

(like: FluentAssertion, Sharp Tests Ex, etc.)

- Their usage of lambda expression predicates hurts the *fluent experience*

```
new[] { 1, 2, 3 }.Should().Contain(item => item > 3, "at least {0} item should be larger than 3", 1);
```

- *(Red is dead,* but) Should is weak…
- We aim to be more ambitious in term of fluentness



DEFINITION OF DONE

- **For NFluent contributors**
  - No warning (all warn as ERROR)
  - No StyleCop warning (full rules)
  - **100% of test coverage** - all test passed of course!
  - The entire **build lasts less than a minute** (including all unit tests)

BY TEST DEVELOPMENT DRIVEN IS?

xUnit.net

```
[Fact]
public void InsideTheListDevilIs()
{
    var list = new List<int> { 1, 2, 3, 42, 5, 666 };

    Assert.Contains(666, list);
}

[Fact]
public void AgreeWithYouTheCouncilDoes()
{
    var charactersList = new List<string> { "Luke", "Vador", "Leila", "Yoda", "Chewie" };

    Assert.Contains("Yoda", charactersList);
}
```

WRITE VALUABLE TESTS

Use NFluent!

```
// Assert is dead!
Check.That(TDD).With(NFluent).IsAnInstanceOf<Awesomeness>();
```

http://www.n-fluent.net

Thank you!

(NFluent logo © **rhwy**)