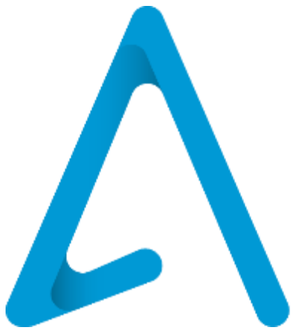


How To Practice TDD

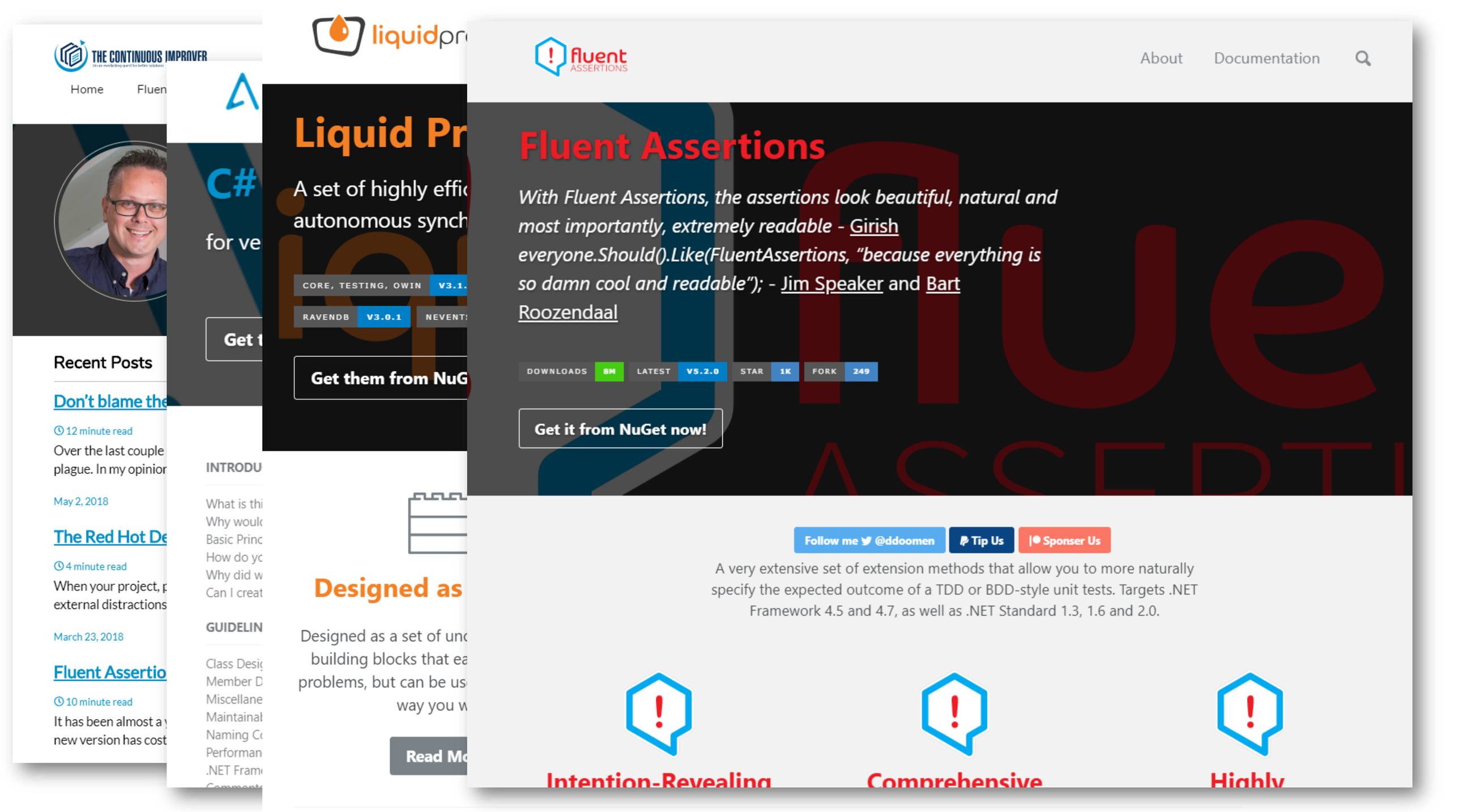
without shooting yourself in the foot

Dennis Doomen | @ddoomen | Aviva Solutions | The Continuous Improver

Dennis Doomen
The Continuous Improver

On an everlasting quest for knowledge that significantly improves the way you build your key systems in an agile world.

@ddoomen

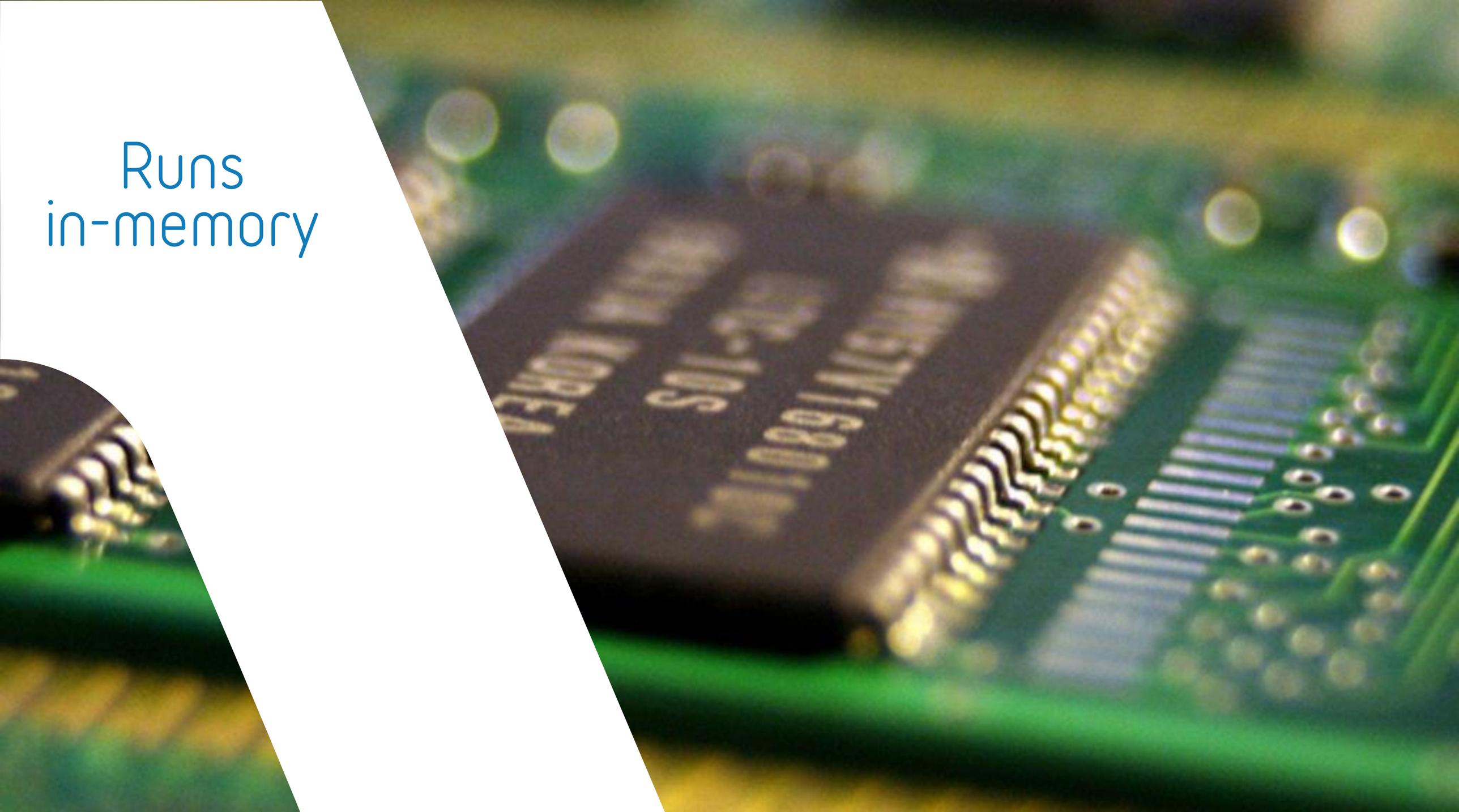

THE CONTINUOUS IMPROVER
Home
Recent Posts
Don't blame the
12 minute read
May 2, 2018
The Red Hot De
4 minute read
March 23, 2018
Fluent Assertio
10 minute read
C#
for ve
Get t
liquidpr
Liquid Pr
A set of highly effi
autonomous synch
CORE, TESTING, OWIN
RAVENDB V3.0.1
Get them from NuG
INTRODU
What is thi
Why woul
Basic Princ
How do yc
Why did w
Can I creat
GUIDELIN
Class Desi
Member D
Miscellane
Maintainal
Naming C
Performan
.NET Fram
Designed as
Designed as a set of unc
building blocks that ea
problems, but can be us
way you w
Read Mo
fluent ASSERTIONS
About
Documentation
Fluent Assertions
With Fluent Assertions, the assertions look beautiful, natural and most importantly, extremely readable - Girish
everyone.Should().Like(FluentAssertions, "because everything is so damn cool and readable"); - Jim Speaker and Bart Roozendaal
DOWNLOADS 8M
LATEST V5.2.0
STAR 1K
FORK 249
Get it from NuGet now!
Follow me @ddoomen
Tip Us
Sponser Us
A very extensive set of extension methods that allow you to more naturally specify the expected outcome of a TDD or BDD-style unit tests. Targets .NET Framework 4.5 and 4.7, as well as .NET Standard 1.3, 1.6 and 2.0.
Intention-Revealing
Comprehensive
Highly

A unit test is....

An opinionated definition

Is fast
(< 50ms)

Runs in-memory



Has no I/O-bound dependencies

Has no static dependencies

Causes no side-effects

Should be parallelizable

Can be run in any order

Test Driven Development means...

(in my definition)

1. Design the class responsibilities
2. Write a first unit test
3. Generate stubs using R#, Rider, etc
4. Ensure test fails for the right reason
5. Implement the real deal
6. Ensure test succeeds
7. Identify alternative scenarios
8. Repeat twice
9. Refactor.

The principles of great unit tests

No black box.
No white box.

Test at the right scope

If it’s not important for the test, it’s very important to hide it

aviva solutions

Crystal-clear cause and effect

aviva solutions

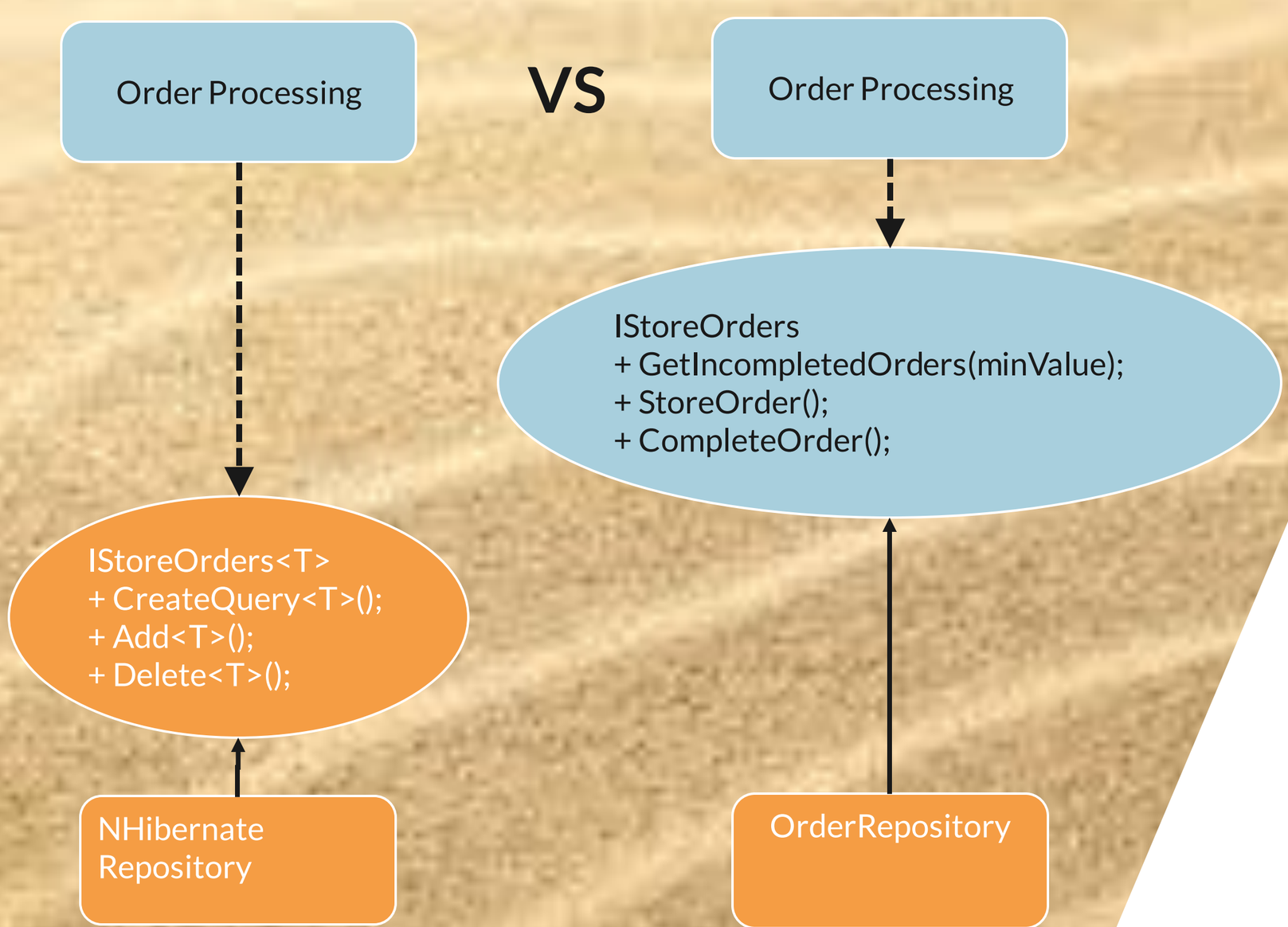
Reads like a book

aviva solutions

Copy three times before refactoring

aviva solutions

Avoids DRY



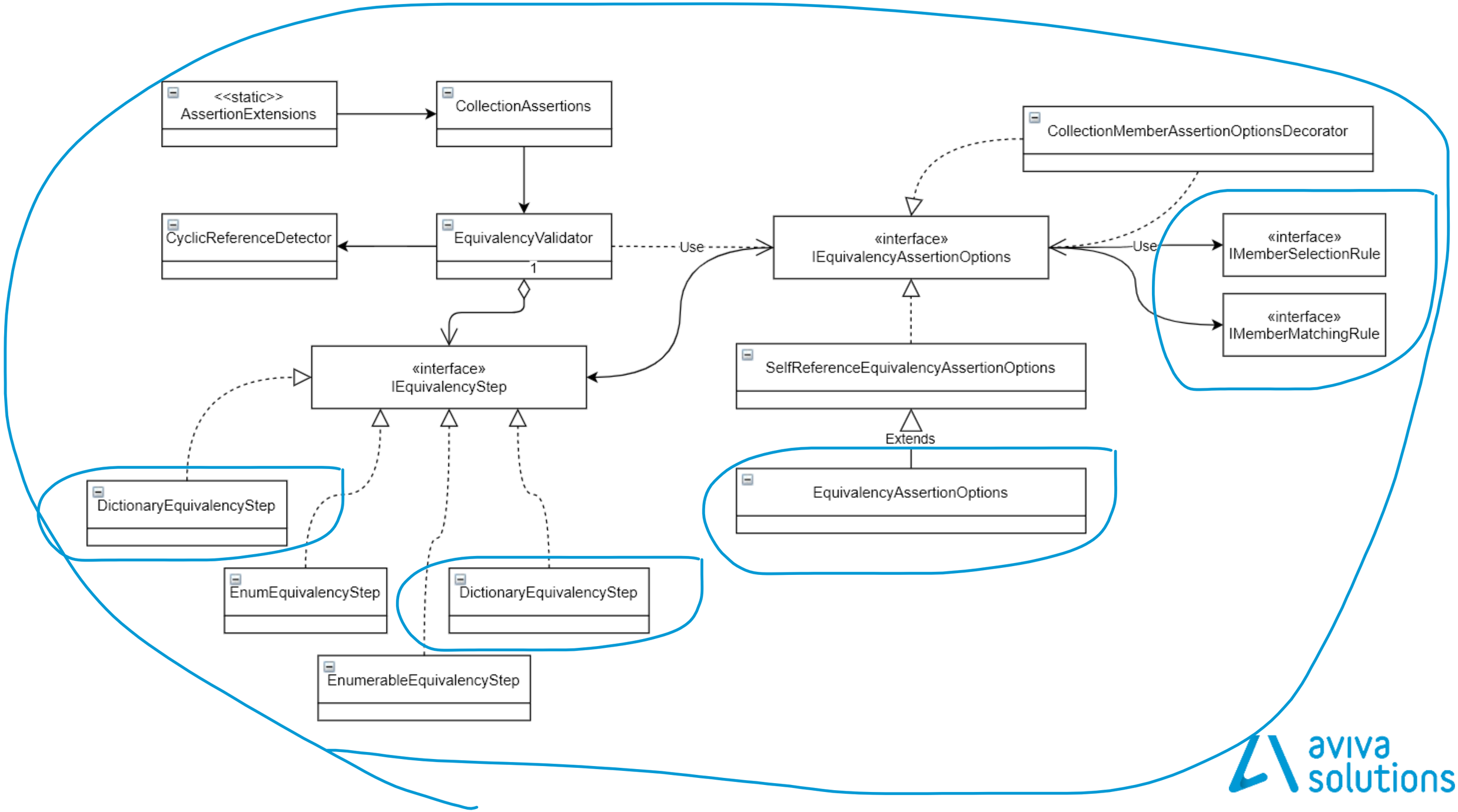
Acknowledges DIP

aviva solutions

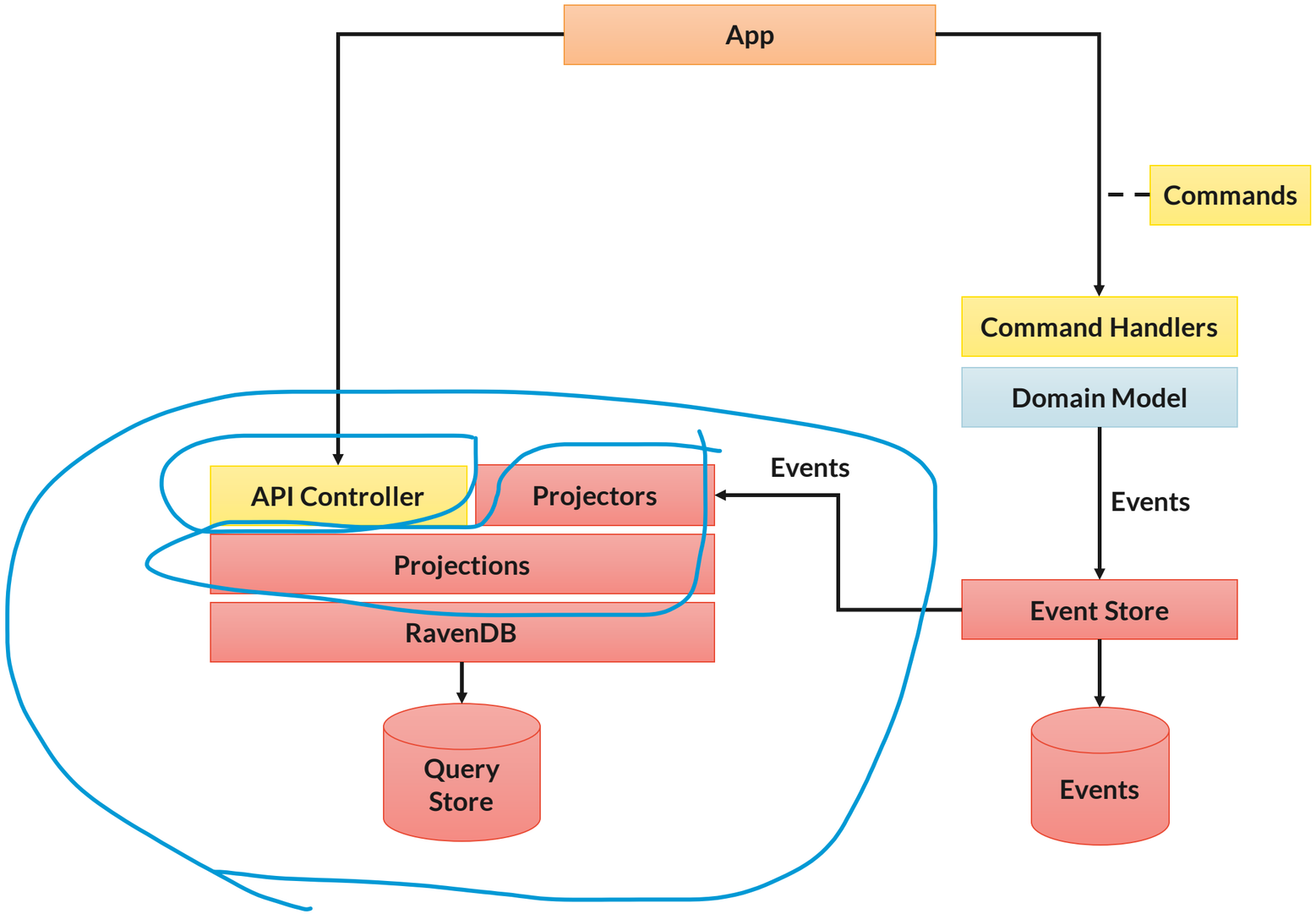
Defining the scope of testing

```
eventMonitor.OccurredEvents.Should().BeEquivalentTo(new[]
{
    new
    {
        EventName = "PropertyChanged",
        TimestampUtc = utcNow - 1.Hours(),
        Parameters = new object[] { “third”, “first”, 123 }
    },
    new
    {
        EventName = "NonConventionalEvent",
        TimestampUtc = utcNow,
        Parameters = new object[] { "first", 123, "third" }
    }
}, o => o.WithStrictOrdering());
```

aviva solutions



aviva solutions



aviva solutions

Show me some code

The Practices

- Define the boundary carefully
- Use BDD or AAA when applicable
- Don’t repeat the context in names
- Don’t use technical names
- Avoid constants
- Use Test Data Builders for the irrelevant parts
- Show relevant dependencies
- Only assert the relevant parts
- Keep refactoring
- Keep out of the debugger hell.

aviva solutions

@ddoomen
dennis.doomen@avivasolutions.nl
www.continuousimprover.com

www.fluentassertions.com
www.csharpcodingguidelines.com

Recommended Resources

- Example code
  https://github.com/dennisdoomen/EffectiveTddDemo
- Chill
  https://github.com/ChillBdd/Chill
- Fluent Assertions
  https://www.fluentassertions.com
- Laws of Jeremy D. Miller
  http://codebetter.com/jeremymiller/2005/10/21/haacked-on-tdd-and-jeremys-first-rule-of-tdd/
- Xunit Patterns
  http://xunitpatterns.com/
- Test data builders
  http://www.natpryce.com/articles/000714.html

aviva solutions