

ТЕСТИРОВАНИЕ

<https://github.com/kontur-csharp/testing>



ТЕСТЫ КАК СПЕЦИФИКАЦИЯ

ДОВЕРИЕ ТЕСТАМ

Будет ли тест понятен ревьюеру?

Сможет ли ревьюер быстро убедиться в корректности теста?

ТЕСТЫ КАК СПЕЦИФИКАЦИЯ

```
class Superman_should {  
    [Test]  
    public void save_kitten_from_tree(){  
        ...  
        superman.Act();  
        Assert.IsTrue(kitten.IsSaved());  
    }  
    [Test]  
    public void wear_redBlue_suit(){  
        ...  
    }  
    ...  
}
```

ПРАВИЛЬНАЯ СТРУКТУРА ТЕСТА

Arrange

Act

Assert



System
Under
Test



SAMPLES / AAA /
ZIP_TESTS.CS

ИМЯ ТЕСТА КАК СПЕЦИФИКАЦИЯ

Что должно быть в имени теста?

1. Conditions: preconditions, input, state
2. System Under Test: class name, method name
3. Expected behaviour / Requirement to check

<http://java.dzone.com/articles/7-popular-unit-test-naming>

ИМЯ ТЕСТА КАК СПЕЦИФИКАЦИЯ

ParserTests.TestParse?

ParserTests.Parse_Fails?

ParserTests.Parse_BigNumbers?

ParserTests.Parse_NumbersGreaterThanMaxInt?

ParserTests.Fail_OnNegativeNumbers?

ИМЯ ТЕСТА КАК СПЕЦИФИКАЦИЯ

`isAdult_AgeLessThan18_False`

`ParseInt_should.Fail_OnNonNumber`

`Stack_should.BeEmpty_AfterCreation`

`When_MandatoryFieldsAreMissing_Expect_StudentAdmissionToFail`



SAMPLES / SPECIFICATIONS /
STACK_SPECIFICATION.CS

АНТИПАТТЕРНЫ



SAMPLES /
ANTIPATTERNS

Local Hero

Loudmouth

Free Ride

Over specification

<http://blog.james-carr.org/2006/11/03/tdd-anti-patterns/>

```
▲ ✓ StringCalculator_should (9 tests) Success
  ▲ ✓ add_ComaSeparatedNumbers (4 tests) Success
    ✓ add_ComaSeparatedNumbers("") Success
    ✓ add_ComaSeparatedNumbers("1,2,3,4,5") Success
    ✓ add_ComaSeparatedNumbers("42") Success
    ✓ add_ComaSeparatedNumbers("42,13") Success
  ▲ ✓ add_NewlineSeparatedNumbers (1 test) Success
    ✓ add_NewlineSeparatedNumbers("1\n2,3") Success
  ✓ listAllNegatives_inExceptionMessage Success
  ✓ throwException_onNegativeNumbers Success
  ▲ ✓ useDelimiterFromFirstSpecialLine (2 tests) Success
    ✓ useDelimiterFromFirstSpecialLine("//;\n1;2;3") Success
    ✓ useDelimiterFromFirstSpecialLine("//|\n4|5|6|1") Success
```

ПРИМЕР СПЕЦИФИКАЦИИ ТЕСТАМИ



ТЕСТ НАПИСАТЬ – КАК ЧАЙ ПОПИТЬ

ПИШЕМ ТЕСТЫ ЛЕГКО

БОРЬБА С ДУБЛИРОВАНИЕМ

- SetUp, TearDown
- Comparer, Equal, ToString
- Собственные Assert-ы

PARAMETRIZED TESTS

Они же Data Driven



SAMPLES / SPECIFICATIONS /
DOUBLEPARSE_SHOULD.CS

ОГРАНИЧЕНИЕ ПО ВРЕМЕНИ

```
[Test, Timeout(1000)]  
public void Test()  
{  
    ...  
}
```

FLUENT ASSERTIONS

1. `Assert.AreEqual(expected, actual)` или `Assert.AreEqual(actual, expected)`?
2. `Assert` – корявая семантика
`(2+2).Should().Be(4)` – лучше!
3. Неинформативные исключения
«Expected True but was False»

`FluentAssertions` – доступны через NuGet

ФИШКИ RESHARPER

Resharper → Tools → Templates Explorer →
Import → tests-templates.DotSettings

tf — TestFixture

tt — Test

su — SetUp

Ctrl+T+R или Ctrl+U+R



CHALLENGE

CHALLENGE

В проекте **Challenge** в файле **WordsStatistics_Tests** напишите тесты:

1. **WordsStatistics** — должен проходить все тесты.
2. **WordStatisticsXXX** — некорректные реализации.
Должны падать хотя бы на одном тесте.

Запускайте по **Ctrl+F5**.

Не открывайте файл **DoNotOpen!**

CHALLENGE

Открываем DoNotOpen!

РАЗБОР CHALLENGE

Тесты по спецификации – это просто

Про взаимодействие разных пунктов спецификации подумать трудно (E3)

Про тесты на производительность вспомнить труднее (998, 999)

Тесты не заменяют Code Review (STA)

Code Review не заменяет тесты (CR)