Тестирование

Что это вообще?

- **Тести́рование програ́ммного обеспе́че́ния** — процесс исследования, испытания программного продукта. (с) - Википедия

Для чего?

- Уверенность в работоспособности программы
- Выявление багов, конкретизации причин ошибок
- Документация кода
- Поддержка качества кода
- Упрощение рефакторинга

- TDD

Виды тестирования

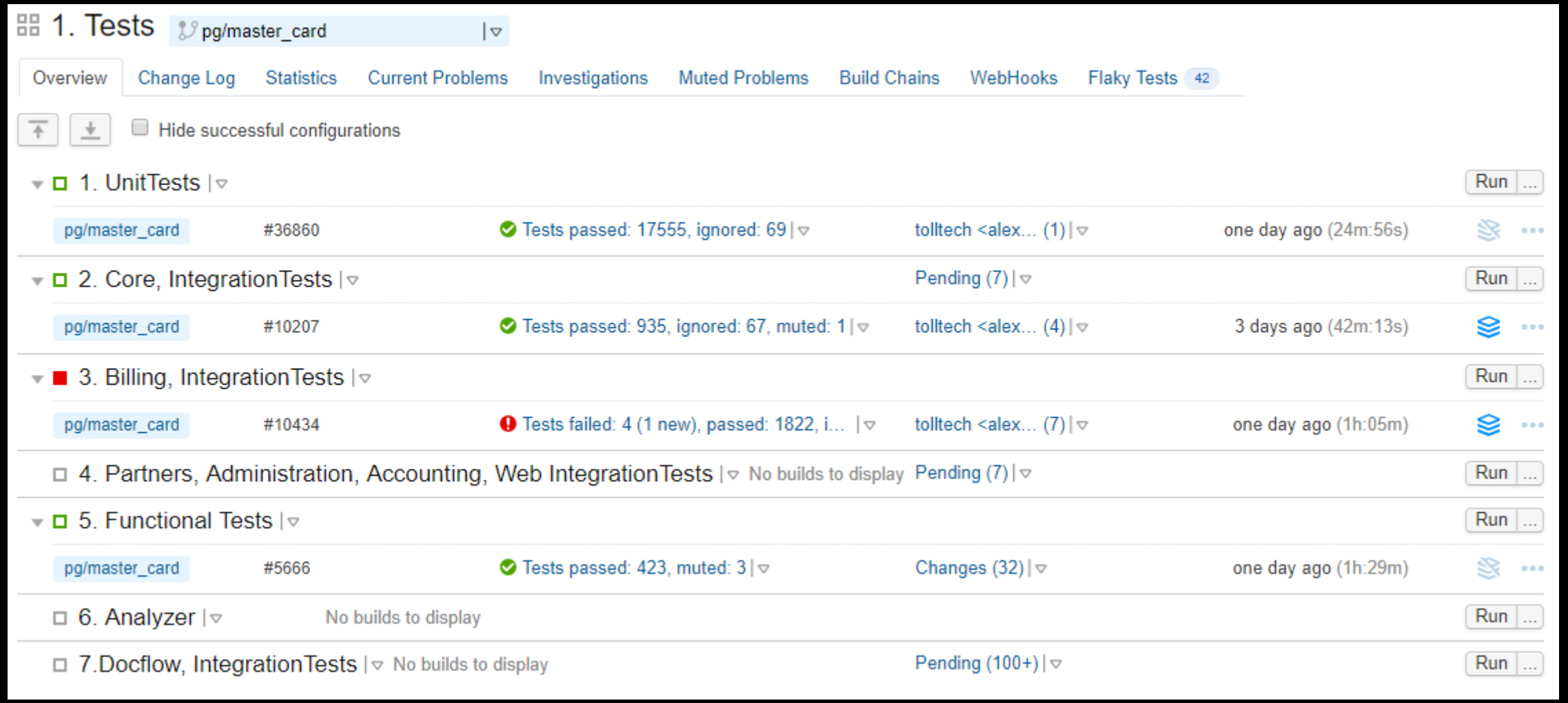
- Модульное
- Интеграционное
- Функциональное
- Нагрузочное
- И еще куча других

Способы тестирования

- Ручное

Способы тестирования

- Автоматизированное



Модульное

Тестирование отдельных компонентов кода.

- Безопасный рефакторинг
- Поддержка качества кода
- Документирование

Интеграционное

Одна из фаз тестирования программного обеспечения, при которой отдельные программные модули объединяются и тестируются в группе. Так же тестирование сторонних сборок, сервисов (приемочное)

- Уверенность в отдельных бизнес-процессах системы
- Уверенность в стороннем функционале

Функциональное

это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям.

Нагрузочное

подвид тестирования производительности, сбор показателей и определение производительности и времени отклика программно-технической системы или устройства в ответ на внешний запрос с целью установления соответствия требованиям, предъявляемым к данной системе (устройству)

- Приближение к реальным условиям
- SLA
- Формирование статистики, отчетов производительности и др. показателей ПО

Доверие тестам

- Будет ли тест понятен ревьюеру?
- Сможет ли ревьюер быстро убедиться в корректности теста?

Тесты как спецификация

```
public class Superman_Should
{
    [Test]
    public void SaveKittenFromTree()
    {
        var kitten = new Kitten();
        var superman = new Superman();
        //...
        superman.Act();
        Assert.IsTrue(kitten.IsSaved());
    }
    [Test]
    public void WearRedBlueSuit_WhenAtWork(){ /* ... */}
}
```

Правильная структура теста

- Arrange
- Act
- Assert

System
Under
Test

Тесты

```
[Test]
public void GiveResultOfSameSize_OnEqualSizeArrays()
{
    var arr1 = new[] { 1 };
    var arr2 = new[] { 2 };

    var result = arr1.Zip(arr2, Tuple.Create);

    CollectionAssert.AreEqual(new[] { Tuple.Create(1, 2) }, result);
}
```

Тесты

```
[Test]
public void BeEmpty_WhenBothInputsAreEmpty()
{
    var arr1 = new int[0];
    var arr2 = new int[0];

    var result = arr1.Zip(arr2, Tuple.Create);

    CollectionAssert.IsEmpty(result);
}
```

Тесты

```
[Test]
public void BeEmpty_WhenFirstIsEmpty()
{
    var arr1 = new int[0];
    var arr2 = new[] { 1, 2 };

    var result = arr1.Zip(arr2, Tuple.Create);

    CollectionAssert.IsEmpty(result);
}
```

Имя теста как спецификация

Что должно быть в имени теста?

- Conditions: preconditions, input, state
- System Under Test: class name, method name
- Expected behaviour / Requirement to check

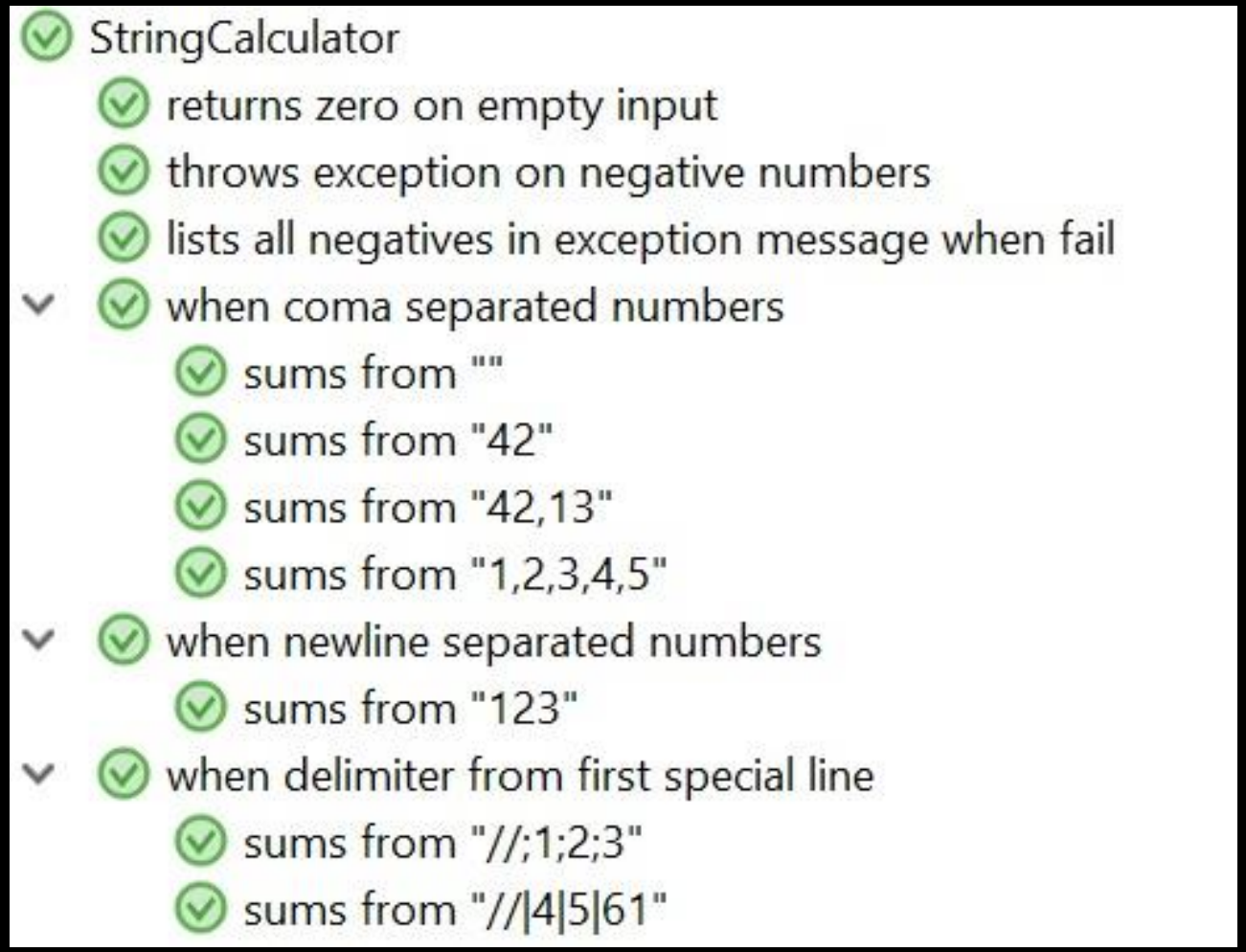
Имя теста как спецификация

- 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

Имя теста как спецификация



TDD

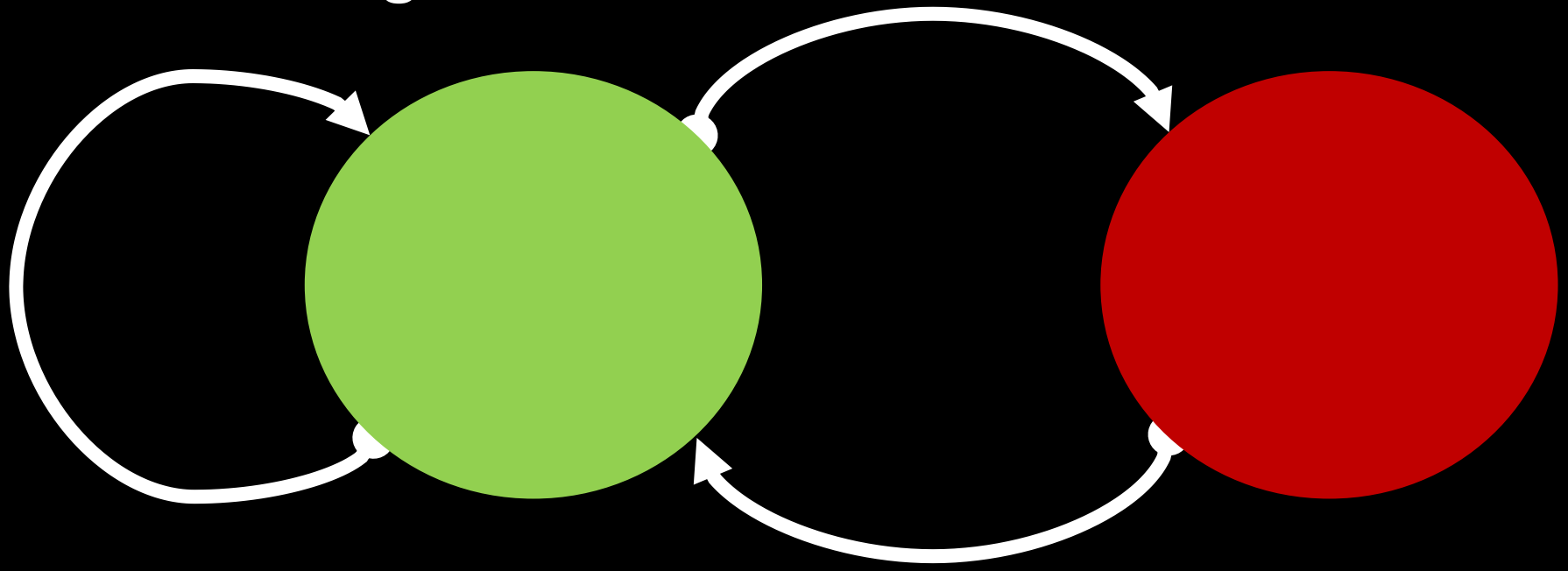
Test Driven Development

Сначала тесты, потом код, потом опять тесты, потом опять код

TDD

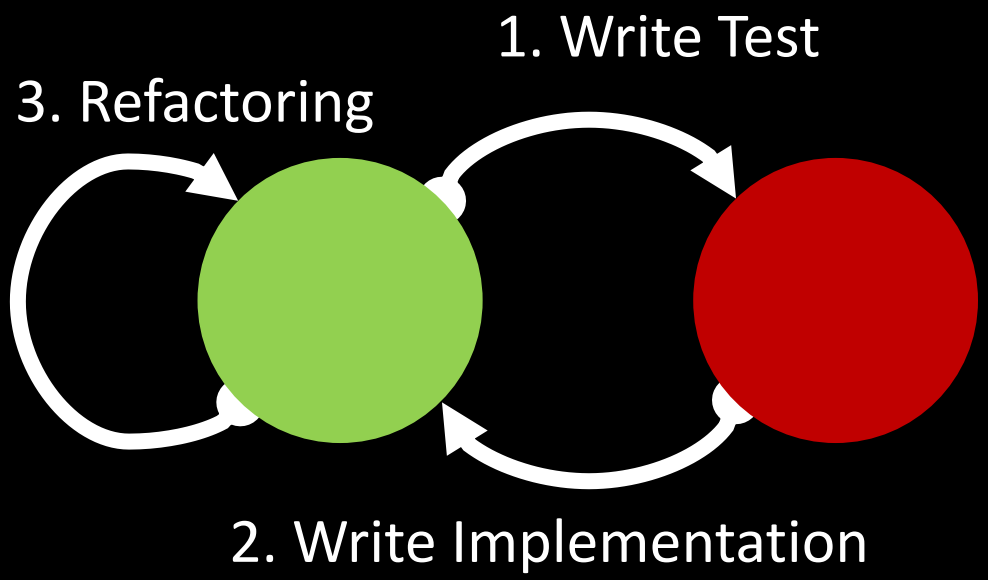
1. Write Test

3. Refactoring



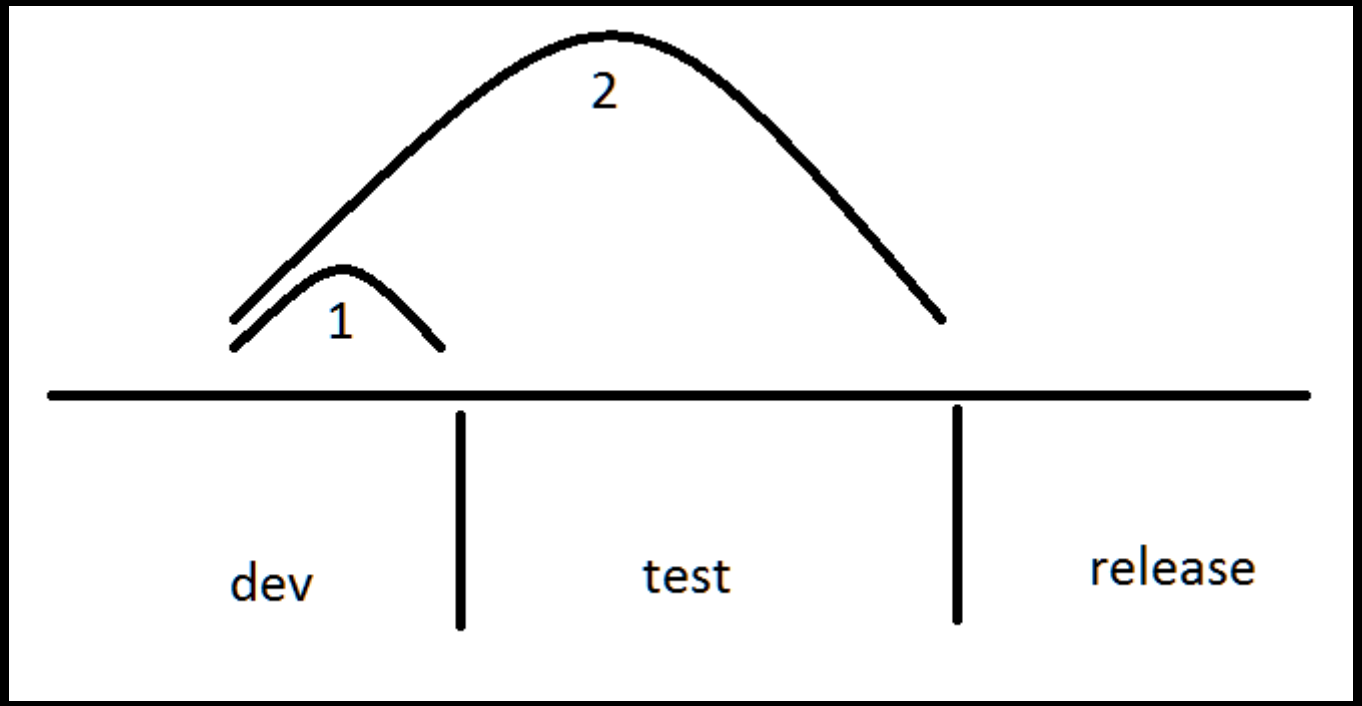
2. Write Implementation

Принципы



- Начинайте с теста
- Двигайтесь маленькими шажками
  - Добавьте простейший красный тест
  - AAA и один Assert на тест
  - Добавьте простейший код, проходящий тест.
    Например, заглушку
- Один красный тест в каждый момент
- Планируйте шажки наперед
- Не забывайте про рефакторинг
  - Перед рефакторингом тесты должны проходить

TIMELINE



1. N + N + N + N/2 = 3.5 N

2. N + N + N + N + N/2 = 4.5 N

Парное TDD

- Ping pong
- Devil's advocate
- 3 min timeframe

Ссылки, библиотеки

- Nunit
- Rhino Mocks
- Selenium

Спасибо за внимание