LUXOFT TRAINING

Разработка через тестирование Acceptance Tests

Test Driven Development

Ivan Dyachenko <IDyachenko@luxoft.com>

Module 8

Содержание

LUXOFT Training Center

Functional testing (Функциональное тестирование)

LUXOFT Training Center

- Тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям
- Тип “Black Box Testing”

Acceptance testing (Приемочное тестирование)

Формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится с целью:

- определения удовлетворяет ли система приемочным критериям
- вынесения решения заказчиком или другим уполномоченным лицом принимается приложение или нет.

Acceptance testing (Приемочное тестирование)

Приемочное тестирование выполняется на основании набора типичных тестовых случаев и сценариев, разработанных на основании требований к данному приложению.

- Тестовый случай (Test Case)
- Тестовый сценарий (Scenario)

Acceptance testing

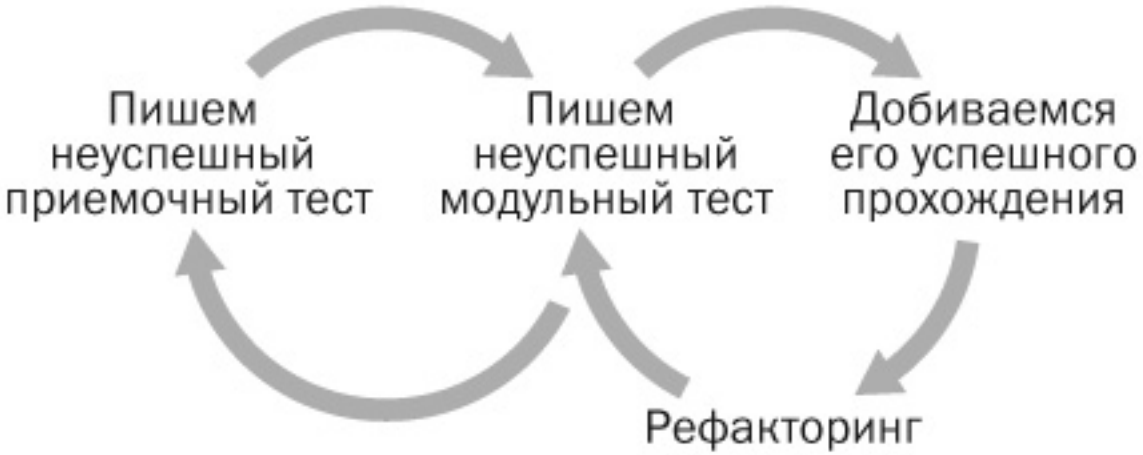
LUXOFT Training Center

При помощи чего производится функциональное тестирование:

- Продукты, эмулирующие поведение браузера httpUnit, JWebUnit, WebTester из SimpleTest и другие.
- Продукты реализованные на JavaScript и реализующие проверки непосредственно средствами браузера. Watir и Selenium

Цикл разработки в TDD

Цикл разработки в BDD



Flavors of BDD approach

BDD подход можно разделить на два вида: xSpec и xBehave.

- xSpec - это применение BDD на уровне модулей (unit level), то есть описание спецификаций по отношению к модулям
- xBehave - это применение BDD для описания высокоуровневых пользовательских историй (user stories) в виде приемочных критериев (acceptance criteria) при помощи given-when-then синтаксиса.

xBehave

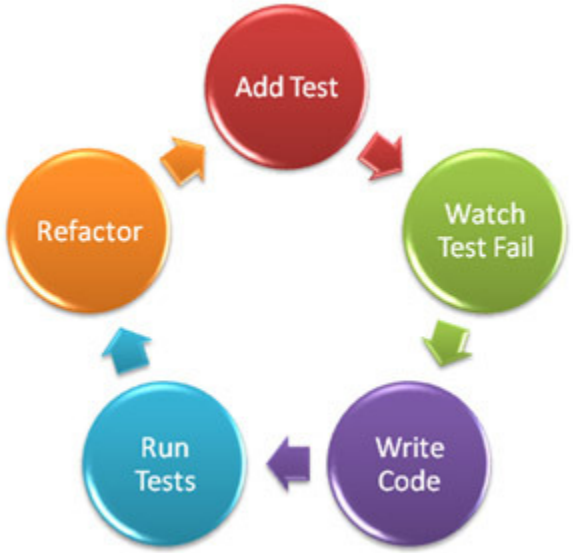
LUXOFT Training Center

Concordion

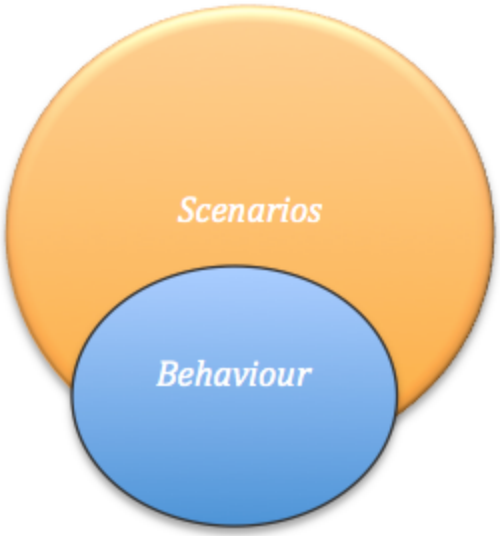
Difference between TDD, BDD & ATDD

LUXOFT Training Center

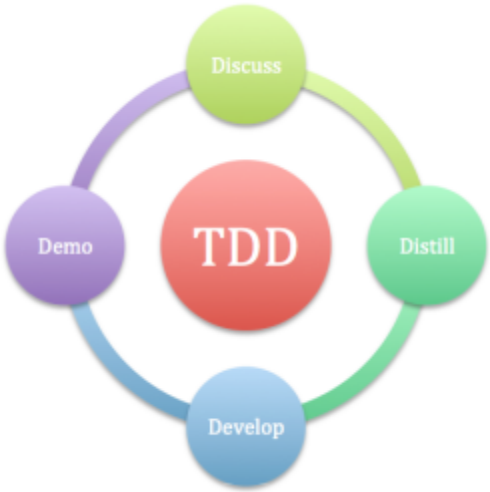
TDD



Behaviour Driven Development



ATDD



LUXOFT Training Center

behaviour driven development
with elegance and joy

http://cukes.info

LUXOFT Training Center

Gherkin

gherkin |ˈgərkin|

Gherkin is the language that Cucumber understands. It is a Business Readable, Domain Specific Language that lets you describe software's behaviour without detailing how that behaviour is implemented.

Cucumber

LUXOFT Training Center

Cucumber syntax is readable by business guys

*.feature

```
# language: en
Feature: Calculation

    Scenario: Add two numbers
    Given I have entered 5
      And I have entered 7
     When I press add
     Then the result should be 12
```

```
VS: assertEquals(12, add(5, 7))
```

*.feature

```
# language: ru
Функционал: Авторизация пользователей

    Чтобы указывать себя автором снипетов, голосовать за
    снипеты и нарабатывать карму, пользователи должны иметь
    возможность регистрироваться

    Сценарий: Успешная авторизация с указываемыми логином и паролем
    Допустим я зарегистрированный пользователь "User12" с паролем "User1212"
    И я на странице Авторизация
    Если ввожу в поле Логин "User12"
    И ввожу в поле Пароль "User1212"
    И кликаю кнопку "Войти"
    То я должен увидеть уведомление "Добро пожаловать!"
    И я должен оказаться на странице Страница пользователя
```

Concordion

LUXOFT Training Center

- Very flexible
- Very pretty report output
- Nice plugin framework
- Poorly documented. I had to read the source to figure it out (luckily its extremely good quality).
- Fixtures seemed likely to end up tightly coupled to the html.

EasyB

LUXOFT Training Center

- Very shallow learning curve (even for non-Groovy Developers)
- Extremely powerful DBUnit integration
- Apparently no support for parameters (leads to either very vague stories or duplication between text and code (edit: actually there is but the documentation for it was very well hidden.)
- Story and Code are very tightly coupled (same file)
- Very basic report output
- Couldn't get IntelliJ plugin to work
- Inactive community (Maven plugin seems to have been broken for three months - not many code examples to draw on).

JBehave

LUXOFT Training Center

- Extremely powerful and flexible (eg reduction of boiler-plate through composition of stories as pre-requisites)
- Extensive (if fragmented) documentation and examples
- Extensive (if overwhelming) support for different frameworks and environments
- Excellent separation of story files from code
- Looks to have a pretty active community and much more examples and discussion of it on web.
- Quite a steep learning curve (took me 3-4 times longer to figure out than Concordion/EasyB)

JBehave

LUXOFT Training Center

JBehave

JBehave – это BDD-инфраструктура для платформы Java, основанная на принципах xUnit. Как естественно предположить, в JBehave делается упор на слово должен, а не на тест.

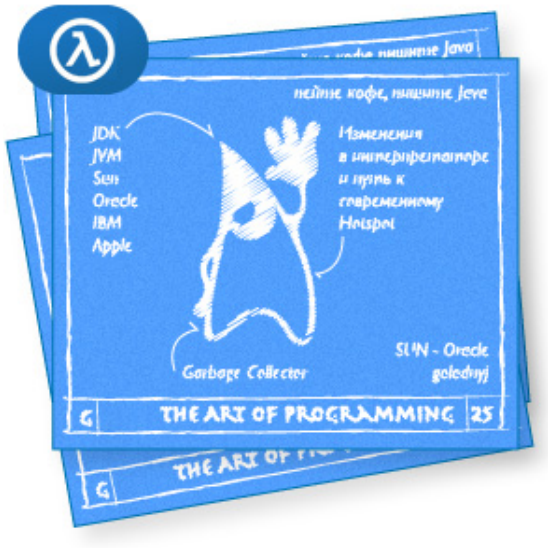
Как и в случае JUnit, классы JBehave можно запускать в вашей стандартной среде разработки или на предпочитаемой вами платформе для сборки проекта, такой как Ant.

JBehave

JBehave позволяет создавать классы для проверки функциональности, почти так же как и в JUnit; однако в случае с JBehave нет необходимости производить наследование от какого-либо базового класса, и все методы для проверки должны начинаться с should, а не с test

Пишем код

LUXOFT Training Center



Вопросы ?

Разработка через тестирование

IDyachenko@luxoft.com

git clone git://github.com/ivan-dyachenko/Trainings.git

https://github.com/ivan-dyachenko/Trainings