BDD - Testing - TestCafe

Manuel Armillas Hernández
Pablo Bande Sánchez-Girón

Contents

Contents

Contents

Test Types

- Unit tests: test individual units
- Integration tests: test that a set of units works well together
- End-to-end tests: analyze the behaviour of the application

Functional Testing

1. Functional testing

- Validates functional specifications
- Black box testing
- Simulate user behavior

TestCafé®

1.1 What do you test?

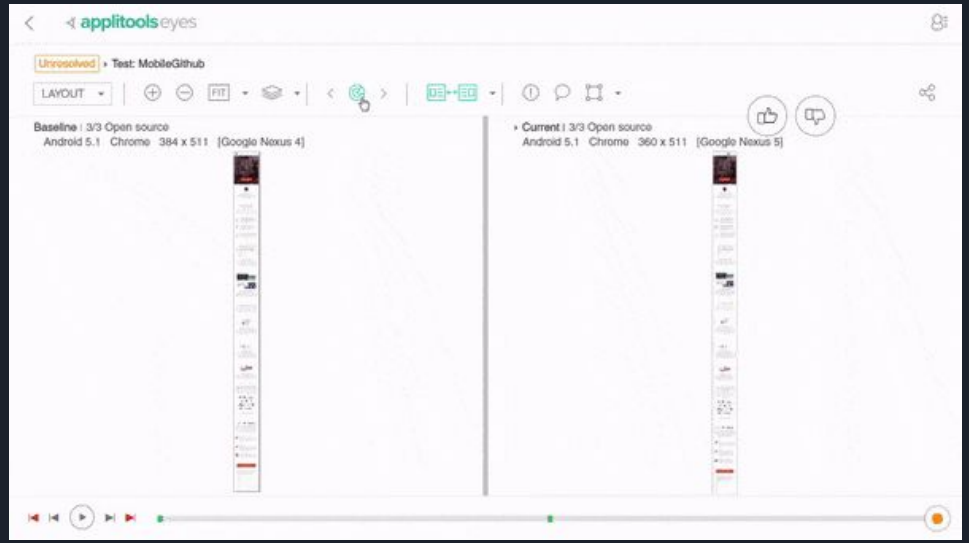
- Mainline functions
- Basic usability
- Basic accessibility
- Error conditions

1.2 Steps

1. User requirements
2. Test case
    2.1. Test input
    2.2. Test output
3. Execute test cases and validate results
4. Log defects and get them fixed

1.3 Visual Regression Testing

- Validate appearance
- Front end
- Detect changes



TestCafé®

2. TestCafé

- E2E testing tool for websites and app´s
- Parallel Testing
- Supports many browsers
  - Internet Explorer and Microsoft Edge
  - Google Chrome
  - Mozilla Firefox

TestCafé®

2.1 Install TestCafé

Global

```
npm install -g testcafe
```

Local

```
npm install --save-dev testcafe
```

2.2 Simple test

```
import { Selector } from 'testcafe';

fixture `Getting Started`

    .page `http://devexpress.github.io/testcafe/example`;

test('My first test', async t => {

    await t

        .typeText('#developer-name', 'John Smith')

        .click('#submit-button')

        .expect(Selector('#article-header').innerText).eql('Thank you, John Smith!');

});
```

2.3 Test results

```
tests — ⌘2
→  tests testcafe chrome test1.js
 Running tests in:
 - Chrome 54.0.2840 / Mac OS X 10.11.5

 Getting Started
 ✓ My first test


 1 passed (1s)
→  tests
```

2.4 Fixture and Test: before and after

.after()

.before()

.aftereach()

.before.each()

```
import { Selector } from 'testcafe';

fixture `My fixture`

    .beforeEach( async t => {

        await t

            .useRole(admin)

            .click('#open-management-console');


test

       .after( async t => {

             await t.click('#delete-data');

       })
```

2.4 Fixture and Test: skip and only

.skip()

.only()

```
import { Selector } from 'testcafe';

fixture.only `My fixture`

    .page `http://example.com`;

test('My fixture - Test', () => {});



fixture.skip `My Fixture 2`;

test.skip('My fixture 2 - Test 1', () => {});

test.only('My fixture 2 - Test 2', () => {});

test('My fixture 2 - Test 3', () => {});
```

2.4 Fixture and Test: page and meta

```
.page()

.meta()
```

```
import { Selector } from 'testcafe';

fixture `My fixture`

    .page `http://example.com`;

test

    .page `http://devexpress.github.io/testcafe/blog/`

    .meta('device', 'mobile');

    (Mobile test', async t => {

    });

test

    .meta('device', 'pc');

    ('PC test', async t => {

    });
```

$ testcafe chrome sample.js --fixture-meta device=mobile

2.5 Interactions

click( )

typeText( )

PressKey( )

hover( )

drag( )

2.5 Interactions: Click

Click

Double-Click

Right-Click

```
import { Selector } from 'testcafe';


fixture `Example`

    .page `https://devexpress.github.io/testcafe/example/`;

test('Click test', async t => {

    const selectBasedOnText = Selector('label').withText('I have tried TestCafe');

    await t

        .click(selectBasedOnText);

});
```

2.5 Interactions: Type text

```
import { Selector } from 'testcafe';


fixture `Example`

    .page `https://devexpress.github.io/testcafe/example/`;


test('Type Text test', async t => {

    await t

        .typeText('#developer-name', 'John Doe');

});
```

2.5 Interactions: Press Key

```
import { Selector } from 'testcafe';



fixture `Example`

    .page `https://devexpress.github.io/testcafe/example/`;

test('Press Key test', async t => {

    await t

        .click('#tried-test-cafe')

        // pressing 'Space' imitates clicking the checkbox again

        .pressKey('space')

});
```

2.5 Interactions: Hover

```
import { Selector } from 'testcafe';


fixture `Example`

    .page `https://js.devexpress.com`;


test('Hover test', async t => {

    await t

        .hover('.map-container');

});
```

2.5 Interactions: Drag elements

```
import { Selector } from 'testcafe';



fixture `Example`

    .page `https://devexpress.github.io/testcafe/example/`;

test('Drag test', async t => {

    const triedCheckbox = Selector('label').withText('I have tried TestCafe');

    await t

        .click(triedCheckbox)

        .drag('#slider', 360, 0, { offsetX: 10, offsetY: 10 });

});
```

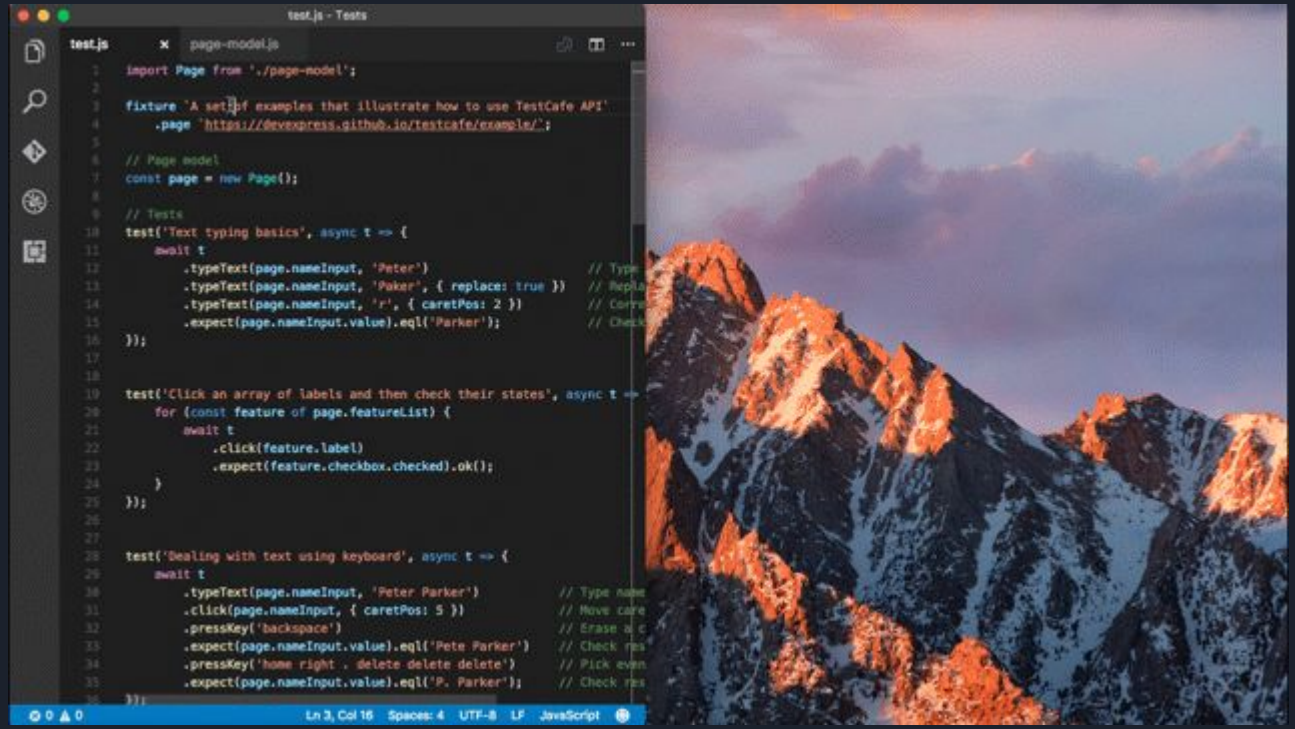
2.6 Assertions

```
.expect(Selector('#text').innerText).eql('Thank you, John Smith!');

.expect(Selector('#text').innerText).contains('Thank you');

.expect(Selector('#element').offsetHeight).gt(400);

.expect(Selector('#element').exists).ok();
```

2.7 Interesting TestCafé options

```
testcafe --list-browsers

testcafe chrome,path:/applications/safari.app tests/sample-fixture.js

testcafe all tests/sample-fixture.js

testcafe  “firefox:headless” tests/sample-fixture.js

testcafe “chrome:emulation:device=iphone X” tests/sample-fixture.js
```

2.8 VSCode: TestCafe Test Runner



Best Practices

Smart assertions

```
import { Selector } from 'testcafe';

fixture `My fixture`
    .page `http://devexpress.github.io/testcafe/example/`;

test('Assertion with Selector', async t => {
    const developerNameInput = Selector('#developer-name');

    await t.typeText(developerNameInput, 'Peter');

    //the selector prefixed with the "await" operator doesn't update and produces unstable test results. Avoi
    const developerName = await Selector('#developer-name').value;

    await t
            .expect(developerName).eql('Peter')
            .typeText(developerNameInput, 'Jack')
            .expect(developerName).eql('Jack'); // fails
});
```

File structure

```
.
├── .testcaferc.json
└── tests
    ├── |- test_group1/
    |   └── |-test1.js
    |       |-test2.js
    ├── |- test_group2/
    |   └── |-test1.js
    |       |-test2.js
    ├── |- page_model/
    |   └── |- page1.js
    |       |- page2.js
    ├── |- helpers/
    |   └── |- helper1.js
    |       |- helper2.js
    ├── |- roles/
    |   └── |- roles.js
    └── |-data
```

Use only one Fixture

```
fixture `Another fixture`
    .page `http://example.com`
    .beforeEach( async t => {
        await setupFileSystem();
    });

test('Another test', async t => {
    //test actions
});
```

Behaviour Driven Development

3.1 What is BDD?

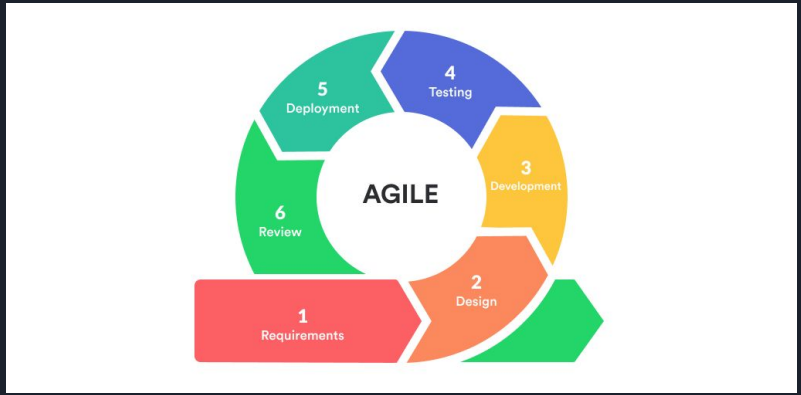
- Behaviour Driven Development
- Development strategy
- Branch of Test Driven Development (TDD)
- Defines a common vocabulary between stakeholders and engineers.

3.3 Some advantages of BDD

1. Defines behaviour and not tests
2. Improve communication between the members of the team
3. Sort learning curve

3.4 More advantages of BDD

1. It´s not technical
2. Let you accept functionalities before development
3. Fits well in agile methodologies



3.5 How to implement the BDD approach?

Gherkin: ‘Given-When-Then’

Feature: Refund item

Scenario: Jeff returns a faulty microwave

Given Jeff has bought a microwave for $100

And he has a receipt

When he returns the microwave

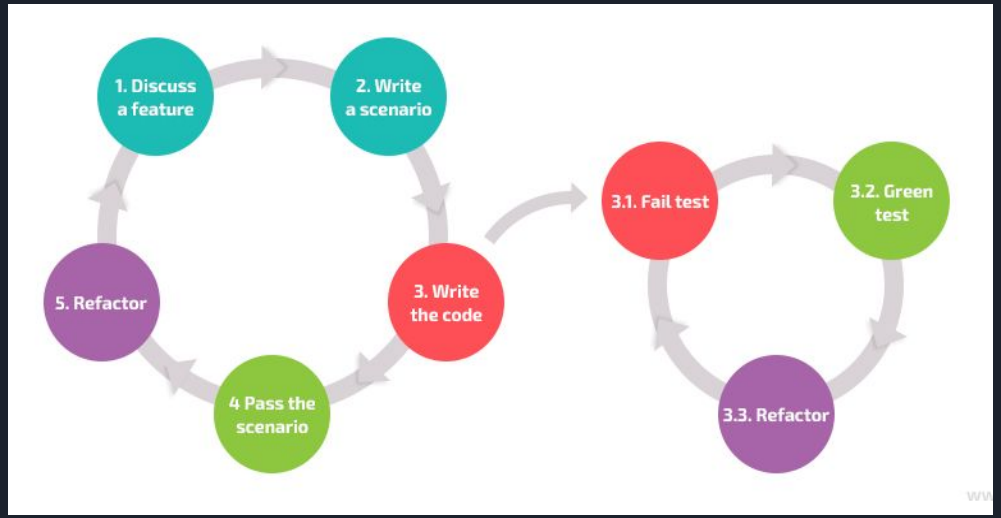
Then Jeff should be refunded $100

‘Role - Feature - Reason’

As a seller, **I want to** receive notification for order creation **so I can** arrange delivery as fast as I can.

3.6 TDD or BDD?

- Different scope
- TDD: development practice
- PDD: team methodology
- Combine both



Questions

Bibliography

BDD

BDD ¿Qué es BDD (Behavior Driven Development)? BDD es una estrategia de desarrollo dirigido por comportamiento

Gherkin Reference

How to describe user stories using Gherkin language

What is the Role-Feature-Reason Template? Examples Using Role-Feature-Reason Template

Writing User Stories With Gherkin | by Nic Werner

11 comentarios en "Entendiendo qué es BDD (Behavior-Driven Development) (I)"

TDD TDD vs BDD. Expectativas de calidad de tus desarrollos Es primordial enfoque de desarrollo en

TDD vs BDD: What's the Difference?

Should we use TDD, BDD… or both?. Comparison between Test-Driven… | by Maya Novarini | Walmart Global Tech Blog

Functional Test

What is Functional Testing? Types & Examples (Complete Tutorial)

What is Functional Testing? Explained with Test Cases & Example (Updated)

Different Functional Testing Types Explained in Detail

TestCafé

How to check whether an element is visible?

#FeatureFriday: End-to-End Testing with TestCafe

Introduction to TestCafe - Tutorial - Part 1

https://testcafe.io

Contact us

Manuel Armillas Hernández
alu0101243498@ull.edu.es

Pablo Bande Sánchez - Girón
alu0101225296@ull.edu.es