Alea jacta test

Bienvenue dans le monde des tests !

Pierre-Yves Lapersonne

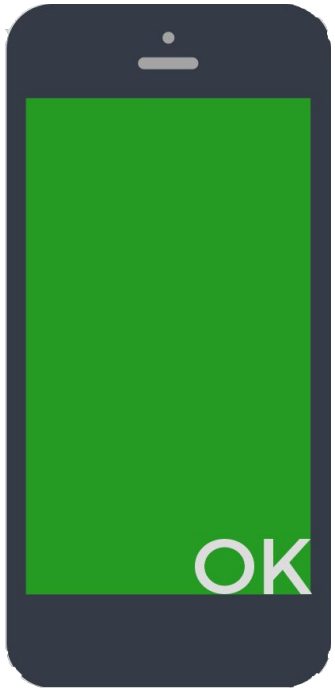
Version 14 - 18/12/2017

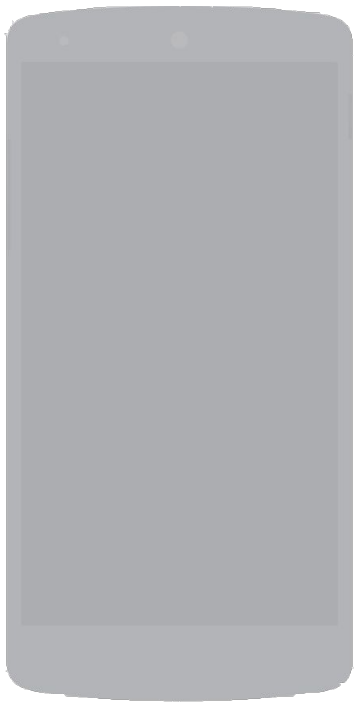
PY LAPERSONNE

Software developer

pylapp.github.io

Code d'Armor





Alea jacta test

Code d'Armor

Plan de test

- les clichés sur les tests
- exemples de tests râtés
- pourquoi en faire ?
- les tests en détails
- quelques outils

Code d'Armor

quelques... trucs.

Code d'Armor

- 9ème Journée Française des Tests Logiciels
  11 / 04 / 2017
  Montrouge, France

- CFTL
  Comité Français des Tests Logiciels
  Perros Guirrec, Bretagne

- ISTQB
  International Software Testing Qualifications Board
  Bruxelles, Belgique

Code d'Armor

quelques chiffres

Code d'Armor

- **40 %** du budget projet dédiés aux tests
- **39 %** des devs sont en mode TDD
- **46 %** des devs estiment ne pas avoir assez de temps pour les tests

Chiffres : 2016

les clichés sur les tests

Code d'Armor

Source : CommitStrip

ça ne sert à rien !

Code d'Armor

ça prend trop de temps !

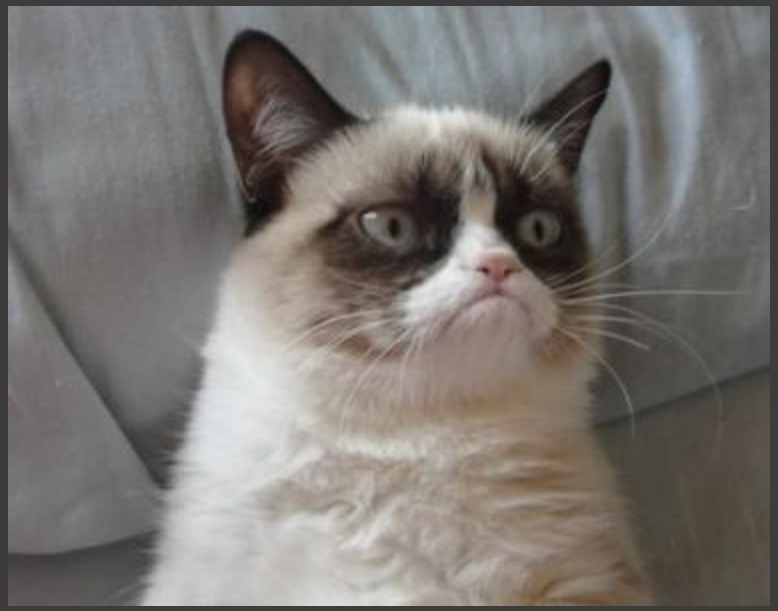
Code d'Armor

ça coûte trop cher !

Code d'Armor

le client est le meilleur testeur !

Code d'Armor

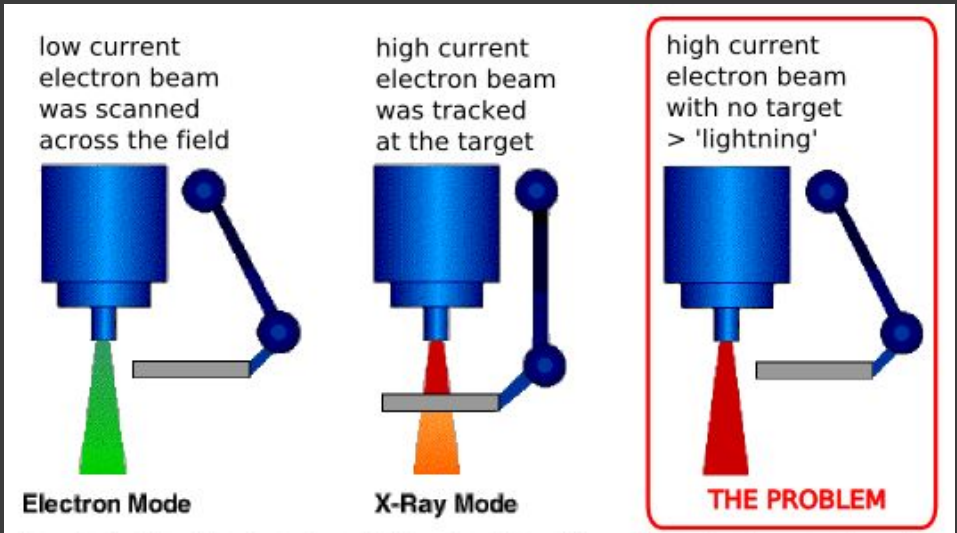


ça ajoute des sources à maintenir !

Code d'Armor

inutiles les tests ?

Code d'Armor



Therac 25

1980’

Image : Wikidot

- outil de traitement de cancers par radiations
- exposition prévue de 200 rad
- exposition réelle de 20 000 rad
- au moins 6 morts

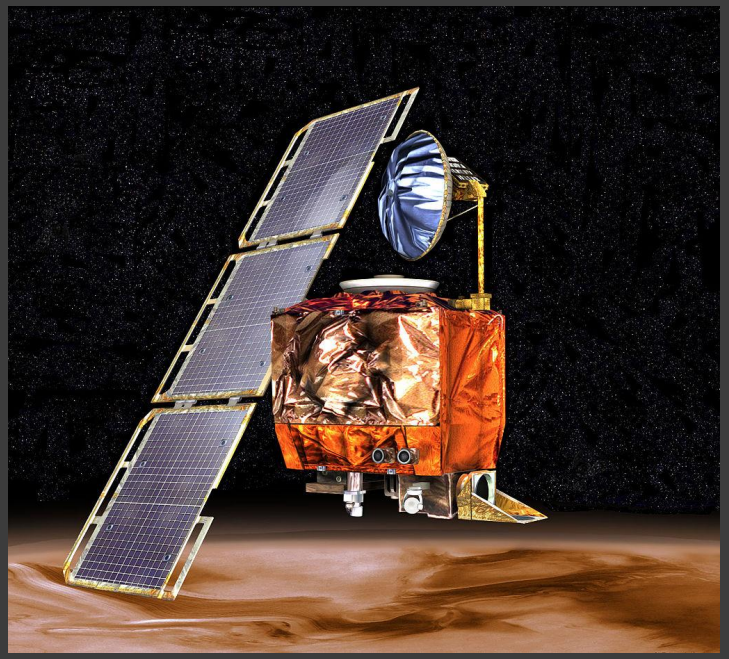
AT&T

15 Janvier 1990

Code d'Armor

- mise à jour du réseau téléphonique
- transmissions de mauvais messages entre les antennes
- 9h de panne
- 60 millions $ de pertes

Code d'Armor



Mars Climate Orbiter

23 Septembre 1999

Image : Wikipédia

- plusieurs équipes internationales...
- utilisant le système anglo-saxon...
- ... ET le système métrique
- navigation totalement défectueuse
- 900 millions $ perdus

vol 501 de Ariane 5

4 Juin 1996

Code d'Armor

- récupération d’éléments logiciels d’Ariane 4
- overflow dans les calculs de trajectoire
- problème de conversion
  float 64 bits ➡ unsigned 16 bits
- destruction de la fusée
- +370 millions $ ... à l’eau

Code d’Armor

retournement d’un F-18

Code d’Armor

- retournement une fois passé l’équateur
- mauvaise gestion des coordonnées

Code d’Armor

Louvois

2011

Image : LeMagIT

- retards de paiement
- mauvais soldes versés
- les familles des militaires en danger
- 465 millions d’euros d’erreur... juste en 2012
  selon la Cour des Comptes

Code d’Armor

Stagefright

2015

Code d'Armor

- AOSP, Firefox, Firefox OS
- librairie multimédia obsolète
- débordement mémoire
- execution de code arbitraire, etc.

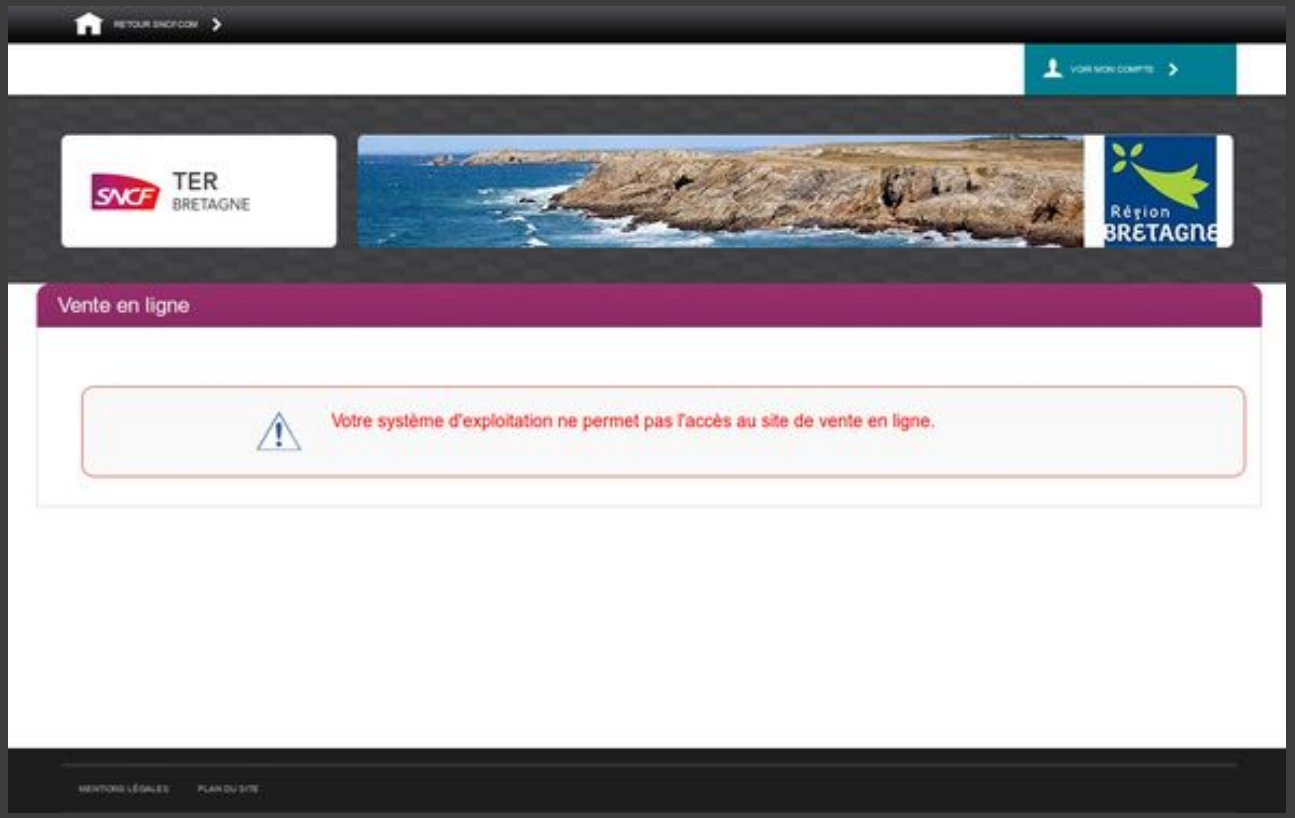
Code d'Armor

iOS et le killer text

2015

Code d'Armor

- réception d’un message particulier
- affichage dans les notifications
- débordement d’un buffer
- comportements dangereux de l’iPhone
  redémarrage, blocage, boot loop, ...



SNCF et Linux...

2015

Code d'Armor

- sur Windows ? Ça fonctionne.
- sur OS X ? Ça fonctionne.
- Linux ? Android ? Oups.

Castorama

8 Juin 2016

Image : Hitek

- suggestions douteuses selon les requêtes
- fermeture du site en catastrophe
  donc manque à gagner pour l'entreprise

Code d'Armor

SAIP

14 Juillet 2016

Image : Google Play

- notification des usagers 3h après la tragédie
- retard d'information par rapport à d'autres *Facebook*, *Twitter*, *QWIDAM*, les *SMS...*

pourquoi faire des tests ?

Code d'Armor

- prouver la qualité du produit
- identifier des comportements incohérents
- repérer d’éventuelles failles
- avoir des clients cont€nts
- éviter les surcharges liées aux bug$
  - et puis avoir la conscience tranquille aussi...

Code d’Armor

mais qu’est-ce qu’un test ?

Code d’Armor

mais qu’est-ce qu’un test ?

définitions

Code d’Armor

Testing

The process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.

Code d'Armor

Verification

Confirmation by examination and through provision of objective evidence that specified requirements have been fulfilled.

Are we building the product right?

Code d'Armor

Validation

Confirmation by examination and through provision of objective evidence that the requirements for a specific intended use or application have been fulfilled.

Are we building the right product?

Code d'Armor

Acceptance

The exit criteria that a component or system must satisfy in order to be accepted by a user, customer or other authorized entity.

Is the feature good enough?

Code d'Armor

mais qu’est-ce qu’un test ?

concrètement

Code d’Armor

- pré-conditions, post-conditions, invariants
- paramètres, valeurs de retour
- succès, échec, en cours, pas fait, planté

- tests statiques
  - vérification des sources
    dead / unreachable code, métriques, syntaxe, standards...
  - pas d’exécution de code

Code d’Armor

- tests dynamiques
  - black box
    se baser sur les spécifications, comportements macros, abstraction de la conception
  - white box
    se baser sur la conception, comportements micros

Code d'Armor

user

instrumented

smoke

regression

monkey

acceptance

usability

component

attack-based

integration

alpha

friendly user

unit

bottom-up

accessibility

integrity

portability

load

beta

stress

Source : http://glossary.istqb.org/search/testing/

les 6 commandements

Code d'Armor

1. Automatisable et rejouable
2. Facile à concevoir
3. Pérenne
4. Exécutable par tous
5. Facilement exécutable
6. Rapide d’exécution

Code d’Armor

et les bugs dans tout ça ?

Code d'Armor

- niveaux de bugs

  new, open, assign, test, deferred, rejected, duplicate, verified, closed

- sévérités

  minor / low, average / medium, major / high, critical

Code d'Armor

Test-Driven Development

- méthodologie de projet

  Kent Beck, 2003

- héritage de *eXtreme Programming*

  revues de codes, tests unitaires, cycles très courts, *test-first*

- *Test Driven Development: By Example, Kent Beck*

  Addison-Wesley Longman, 2002, ISBN 0-321-14653-0, ISBN 978-0321146533

Code d'Armor

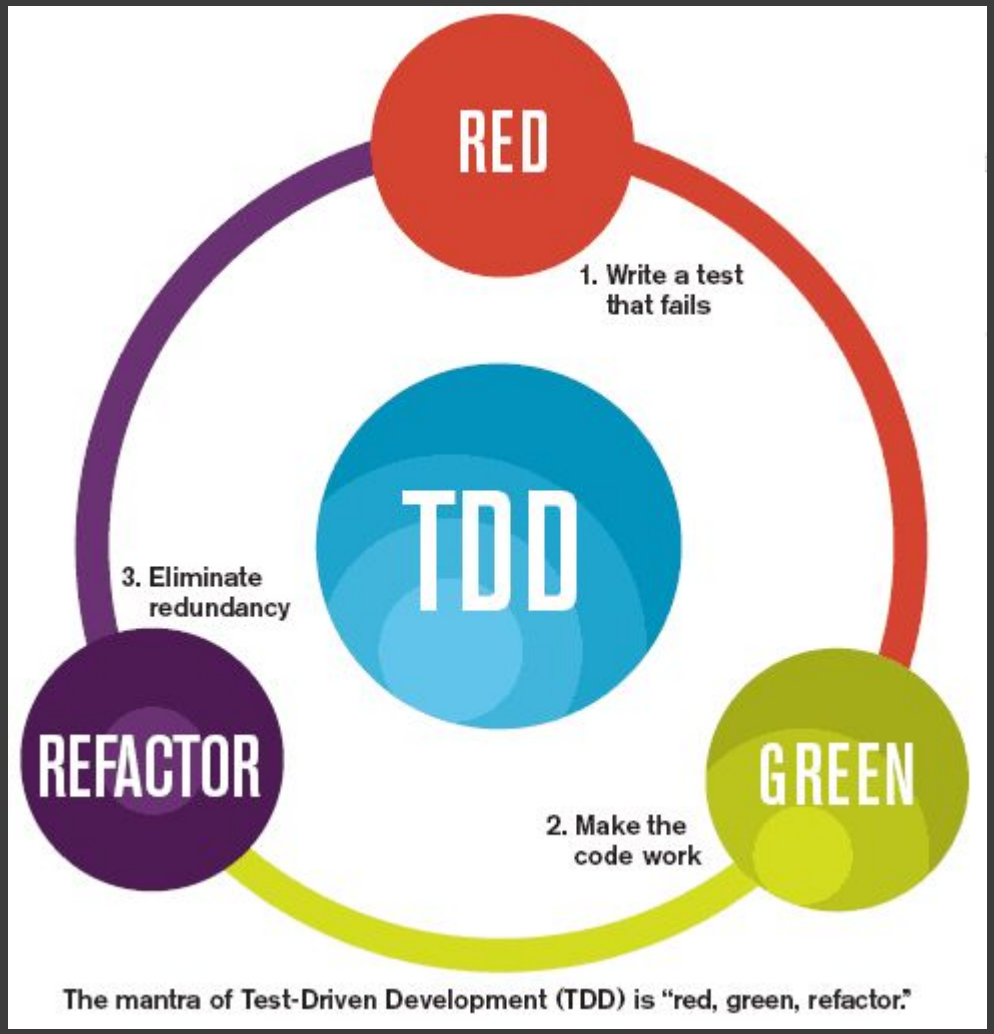
- proche du programme

  JUnit, Espresso, QUnit, UnitJS, Mocha, ...

- vise le code source, l'implémentation

  bas niveau, côté développeurs

Code d'Armor



The mantra of Test-Driven Development (TDD) is "red, green, refactor."

Image : apollossc.com

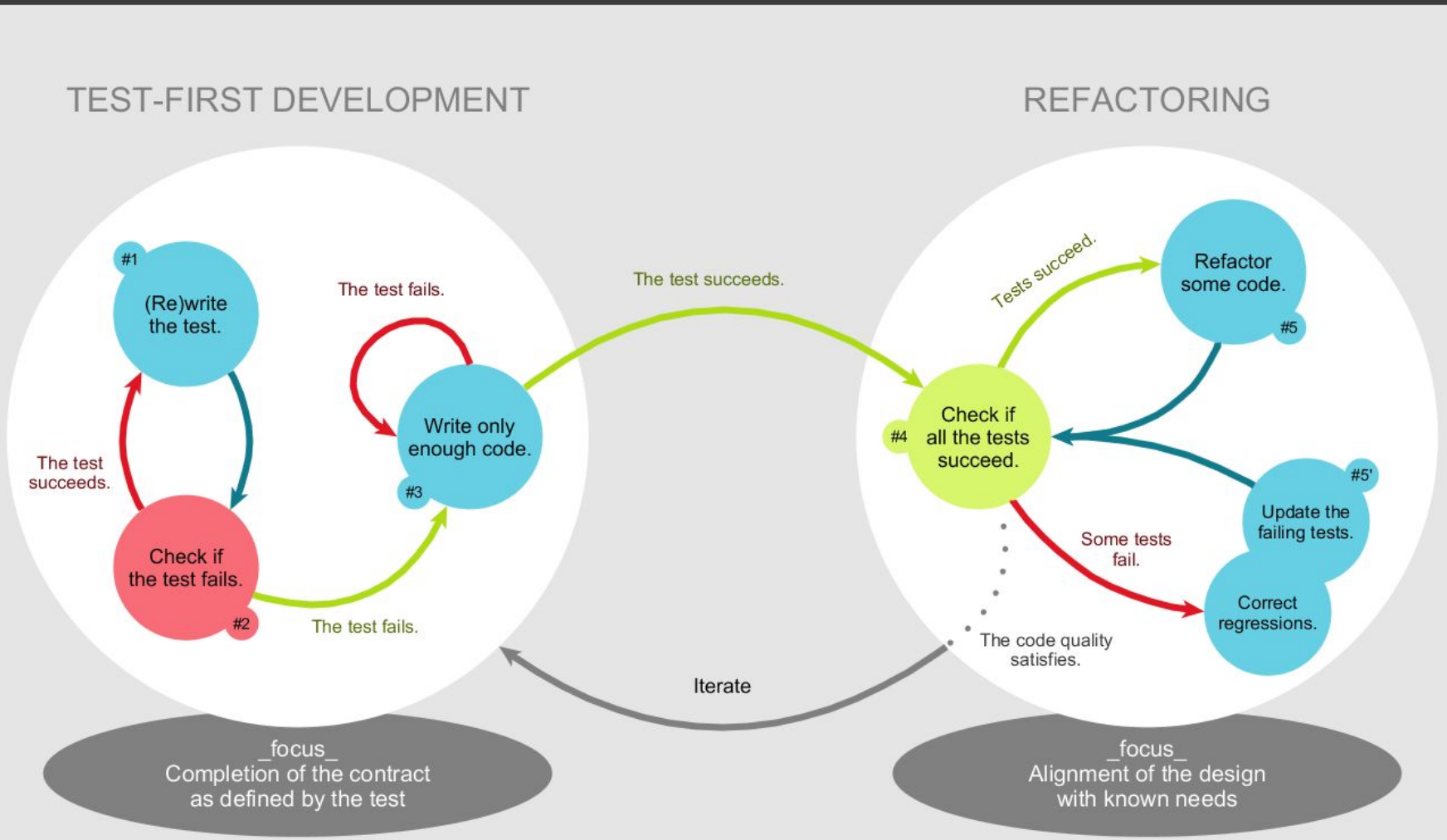


Image : Wikipédia

Behaviour-Driven Development

Code d'Armor

- méthodologie de projet

  Dan North, 2003

- héritage de *Test-Driven Development*

  mélange de *XP*, *TDD*, *YAGNI*, *domain driven design, DSL*

- basé sur des *stories* et vise les specifications

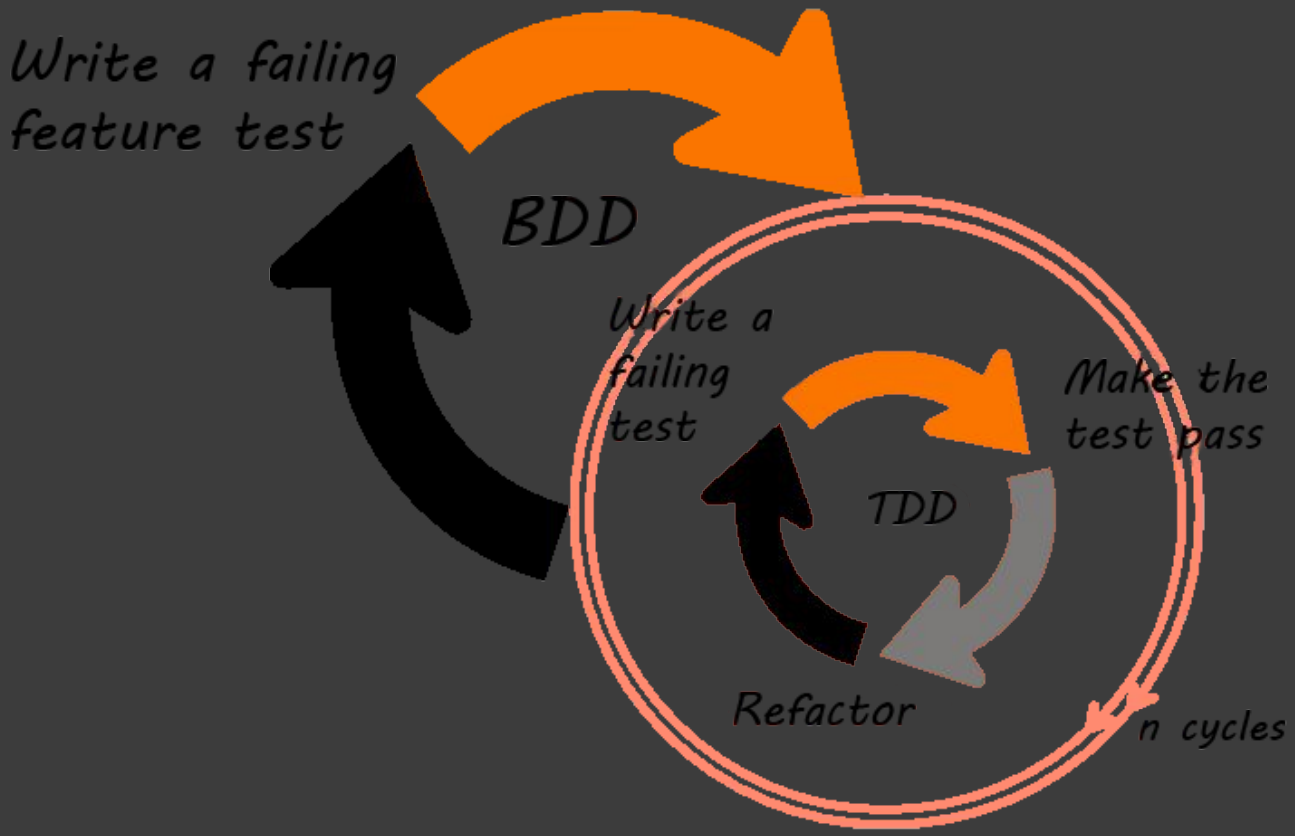
  Cucumber, Gherkin

Code d'Armor

- proche des fonctionnalités
- vise le comportement du produit

  et ne considère pas d'abord l'implémentation

Code d'Armor



Source : http://agilelucero.com/agile-points/bdd/

Acceptance Test-Driven Development

Code d'Armor

- méthodologie de projet
- héritage de *Behaviour-Driven Development*

  mélange de *BDD*, *TDD*
- basé sur des *stories*

  Robot Framework

Code d'Armor

- proche de l’utilisateur
- vise l’*acceptance* du produit
- permet de guider le développement

  implique tous les membres du projet qui savent ce qui doit être fait

Code d’Armor

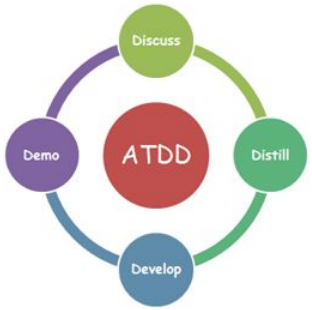
{T | B | AT} DD

products

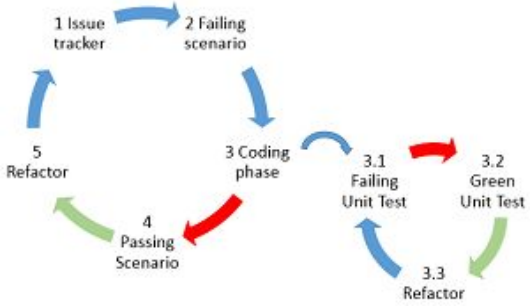
programs

abstraction

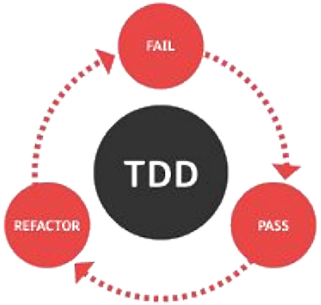
ATDD - acceptance



BDD - features



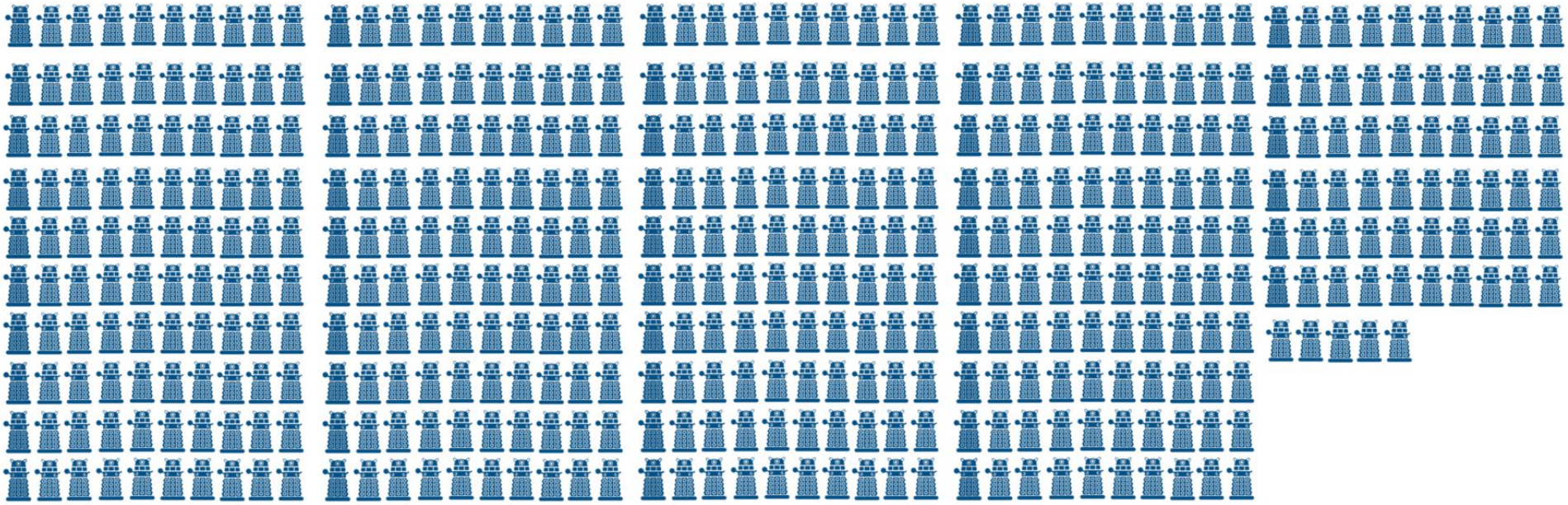
TDD - source code



Images : https://www.linkedin.com/pulse/tdd-vs-atdd-bdd-vahid-farahmandian

la (grosse) boîte à outils

Code d'Armor







Robotium




UI TESTING FOR ANDROID
espresso



appium

MOCHA

selendroid
SELENIUM FOR ANDROID
JUnit

Android
uiautomator
catch

QUnit
js unit testing
QUiCK
cucumber
et bien d'autres encore...

project lifecycle

tasks scheduler

Acceptance Test-Driven Development

instrumented tests

unit tests

Code d'Armor

JUnit

```java
import static org.junit.Assert.assertEquals;

import org.junit.Test;

public class MyTests {

  @Test
  public void multiplicationOfZeroIntegersShouldReturnZero() {

    // MyClass is tested
    MyClass tester = new MyClass();

    // assert statements
    assertEquals("10 x 0 must be 0", 0, tester.multiply(10, 0));
    assertEquals("0 x 10 must be 0", 0, tester.multiply(0, 10));
    assertEquals("0 x 0 must be 0", 0, tester.multiply(0, 0));
  }

}
```

```java
@RunWith(RobolectricTestRunner.class)
public class WelcomeActivityTest {

    // Robolectric is a unit-testing frameworkn we can check intentsn the LoginActivity will not start really
    @Test
    public void clickingLogin_shouldStartLoginActivity() {
        WelcomeActivity activity = Robolectric.setupActivity(WelcomeActivity.class);
        activity.findViewById(R.id.login).performClick();
        Intent expectedIntent = new Intent(activity, LoginActivity.class);
        assertThat(shadowOf(activity).getNextStartedActivity()).isEqualTo(expectedIntent);
    }

}
```


```java
@RunWith(AndroidJUnit4.class)
public class MainActivityEspressoTest {


    @Rule
    public ActivityTestRule<MainActivity> mActivityRule =
      new ActivityTestRule<>(MainActivity.class);

    @Test
    public void ensureTextChangesWork() {
        // Type text and then press the button.
        onView(withId(R.id.inputField))
                .perform(typeText("HELLO"), closeSoftKeyboard());
        onView(withId(R.id.changeText)).perform(click());

        // Check that the text was changed.
        onView(withId(R.id.inputField)).check(matches(withText("Lalala")));
    }

    @Test
    public void changeText_newActivity() {
        // Type text and then press the button.
        onView(withId(R.id.inputField)).perform(typeText("NewText"),
                closeSoftKeyboard());
        onView(withId(R.id.switchActivity)).perform(click());

        // This view is in a different Activity, no need to tell Espresso.
        onView(withId(R.id.resultView)).check(matches(withText("NewText")));
    }
}
```

Android
uiautomator

```
private UiDevice mDevice;

@Before
public void startMainActivityFromHomeScreen() {
    // Initialize UiDevice instance
    mDevice = UiDevice.getInstance(getInstrumentation());
    // Start from the home screen
    mDevice.pressHome();
    // Wait for launcher
    final String launcherPackage = mDevice.getLauncherPackageName();
    assertThat(launcherPackage, notNullValue());
    mDevice.wait(Until.hasObject(By.pkg(launcherPackage).depth(0)), LAUNCH_TIMEOUT_MS);
    // Launch the app
    Context context = InstrumentationRegistry.getContext();
    final Intent intent = context.getPackageManager().getLaunchIntentForPackage(PACKAGE_APP_PATH);
    // Clear out any previous instances
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
    context.startActivity(intent);
    // Wait for the app to appear
    mDevice.wait(Until.hasObject(By.pkg(PACKAGE_APP_PATH).depth(0)), LAUNCH_TIMEOUT_MS);
}

@Test
public void testClicksAndCheckResult() throws UiObjectNotFoundException {

    // Different ways to get components...
    UiObject bt5 = mDevice.findObject(new UiSelector().text("5"));
    UiObject btAdd = mDevice.findObject(new UiSelector().text("+").className("android.widget.Button"));
    UiObject bt7 = mDevice.findObject(new UiSelector().text("7"));
    UiObject btEq = mDevice.findObject(new UiSelector().resourceId(PACKAGE_APP_PATH+":id/buttonEqual"));

    bt5.click();
    btAdd.longClick();
    bt7.click();

    // Some controls are available on picked components
    if ( btEq.exists() && btEq.isEnabled() && btEq.isClickable() ) btEq.click();

    UiObject result = mDevice.findObject(new UiSelector().resourceId(PACKAGE_APP_PATH + ":id/tvScore"));
    assertEquals("12.0", result.getText()); // Thanks JUnit !

}
```

Monkey

adb shell monkey -p com.package.myApp --throttle 300 -s 123456 10

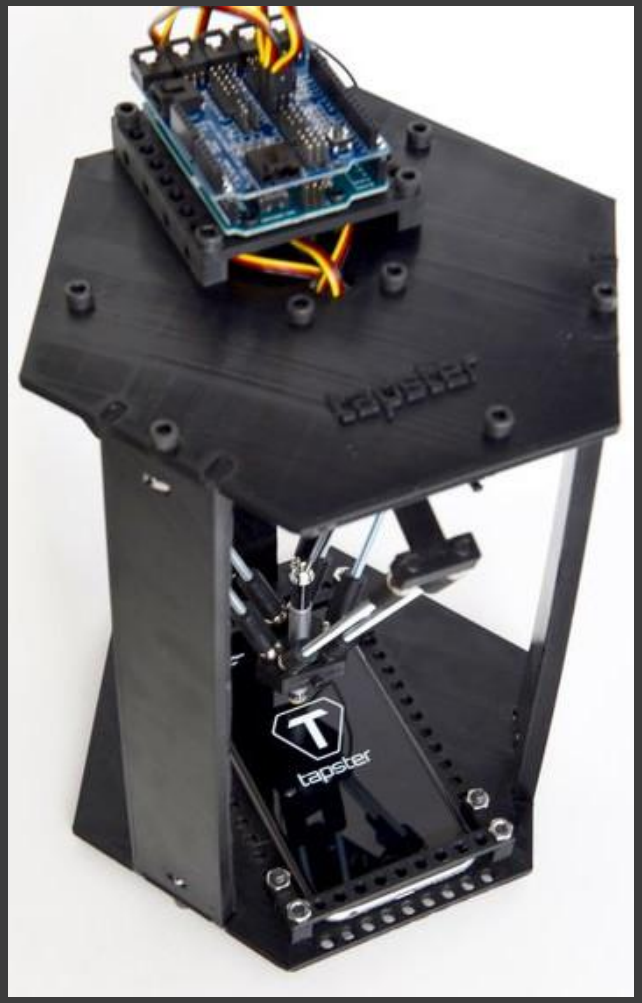
```
:Monkey: seed=123456 count=10
:AllowPackage:
:IncludeCategory: android.intent.category.LAUNCHER
:IncludeCategory: android.intent.category.MONKEY
// Event percentages:
//   0: 15.0%
//   1: 10.0%
//   2: 2.0%
//   3: 15.0%
//   4: -0.0%
//   5: 25.0%
//   6: 15.0%
//   7: 2.0%
//   8: 2.0%
//   9: 1.0%
//   10: 13.0%
:Switch: #Intent;action=android.intent.action.MAIN;category=android.intent.category.LAUNCHER;launchFlags=0x10200000;component=
                    /.views                    ;end
    // Allowing start of Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=
    /.views                    } in package
:Sending Touch (ACTION_DOWN): 0:(177.0,736.0)
:Sending Touch (ACTION_UP): 0:(165.56677,739.5863)
:Sending Trackball (ACTION_MOVE): 0:(-1.0,1.0)
Events injected: 10
:Sending rotation degree=0, persist=false
:Dropped: keys=0 pointers=0 trackballs=0 flips=0 rotations=0
## Network stats: elapsed time=1287ms (0ms mobile, 0ms wifi, 1287ms not connected)
// Monkey finished
```

adb shell monkey -p com.package.myApp --pct-touch 50 -pct-motion 25 --pct-trackball 25 10000

Monkeyrunner

```
# Imports the monkeyrunner modules used by this program
from com.android.monkeyrunner import MonkeyRunner, MonkeyDevice

# Connects to the current device, returning a MonkeyDevice object
device = MonkeyRunner.waitForConnection()

# Installs the Android package. Notice that this method returns a boolean, so you can test
# to see if the installation worked.
device.installPackage('myproject/bin/MyApplication.apk')

# sets a variable with the package's internal name
package = 'com.example.android.myapplication'

# sets a variable with the name of an Activity in the package
activity = 'com.example.android.myapplication.MainActivity'

# sets the name of the component to start
runComponent = package + '/' + activity

# Runs the component
device.startActivity(component=runComponent)

# Presses the Menu button
device.press('KEYCODE_MENU', MonkeyDevice.DOWN_AND_UP)

# Takes a screenshot
result = device.takeSnapshot()

# Writes the screenshot to a file
result.writeToFile('myproject/shot1.png','png')
```


```
public class TestSample extends AppiumConfigTestSample /* <-- custom class with configuration */ {


        /**
         * Tests the swipe : from right to left and left to right
         */
        @org.junit.Test
        public void doSomeTests() throws Exception {

                // Swipe from right to left
                Dimension size = driver.manage().window().getSize();
                int startx = (int) (size.width * 0.9);
                int endx = (int) (size.width * 0.1);
                int starty = size.height / 2;
                driver.swipe(startx, starty, endx, starty, 1000 /*duration*/);

                try { Thread.sleep(1000); } catch ( InterruptedException ie ){}

                // Get items and click on them
                List<String> cell_names = new ArrayList<String>();
                for (WebElement cell : tags("android.widget.TextView")) {
                        cell_names.add(cell.getAttribute("text"));
                }
                for (String cell_name : cell_names) {
                        wait(for_text(cell_name)).click();
                }

                // A simple click
                new TouchAction(driver).tap(element(By.id("mypackage.myapp:id/fab"))).perform();
                wait(for_text("A dummy text to wait for"));

                // A long press
                new TouchAction(driver).longPress(element(By.id("mypackage.myapp:id/fab"))).perform();

                // Find the activity "Dummy Activity", android:exported should be to true in the manifest
                driver.startActivity("mypackage.myapp","mypackage.myapp.MyActivity");

                new TouchAction(driver).tap(element(for_find("An item's text !"))).perform();

                // Plays with seekbars...
                new TouchAction(driver).longPress(element(By.id("aleajactatest.appiumcalculator:id/seekBar"))).moveTo(100, 0).perform();

                // Open the menu
                driver.sendKeyEvent(AndroidKeyCode.MENU);
                // Open the notifications panel
                driver.openNotifications();
                // Locks the screen
                driver.lockScreen(3);

        }
}
```


tapster

cucumber

```
# features/link_click.feature
Feature: Link Click

@javascript
Scenario: User clicks the link
Given I am on the homepage
When I click the provided link
Then I should see the link click confirmation

# features/step_definitions/link_click_steps.rb
Given(/^I am on the homepage$/) do
  visit root_path
end

When(/^I click the provided link$/) do
  click_on "js-click-me"
end

Then(/^I should see the link click confirmation$/) do
  expect(page).to have_content("Link Clicked")
end
```

```
*** Settings ***
Documentation     Example test case using the gherkin syntax.
...
...               This test has a workflow similar to the keyword-driven
...               examples. The difference is that the keywords use higher
...               abstraction level and their arguments are embedded into
...               the keyword names.
...
...               This kind of _gherkin_ syntax has been made popular by
...               [http://cukes.info|Cucumber]. It works well especially when
...               tests act as examples that need to be easily understood also
...               by the business people.
Library           CalculatorLibrary.py

*** Test Cases ***
Addition
    Given calculator has been cleared
    When user types "1 + 1"
    and user pushes equals
    Then result is "2"

*** Keywords ***
Calculator has been cleared
    Push button    C

User types "${expression}"
    Push buttons    ${expression}

User pushes equals
    Push button    =

Result is "${result}"
    Result should be    ${result}
```

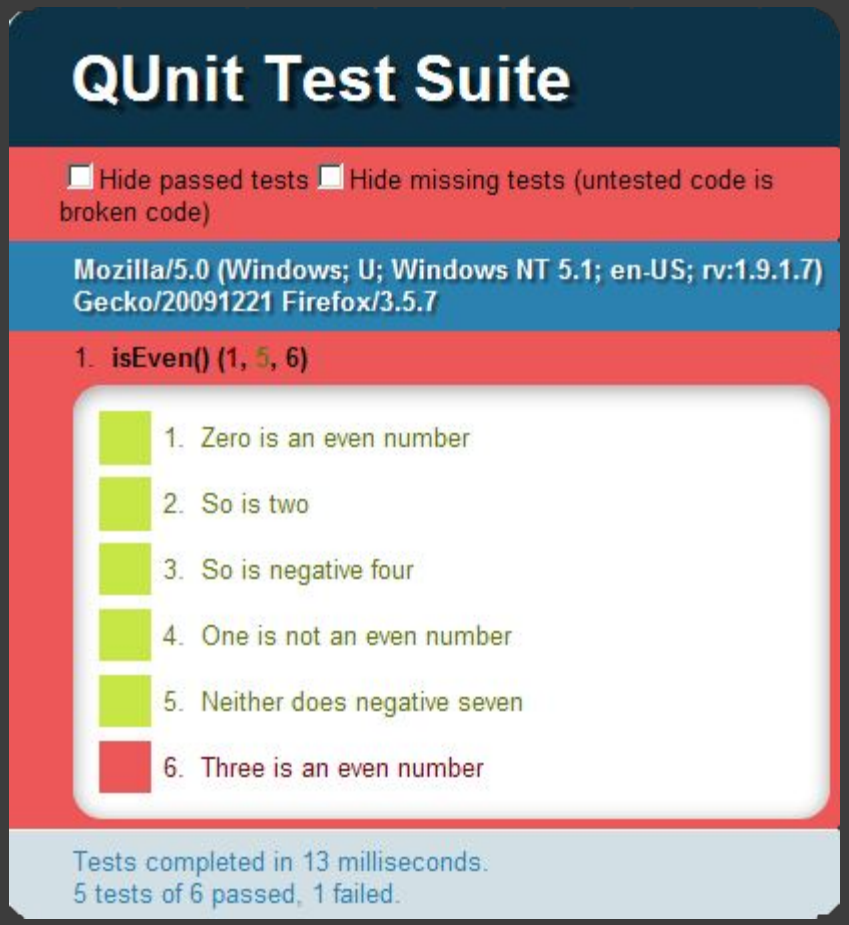
```
class CalculatorLibrary(object):
    """Test library for testing *Calculator* business logic.

    Interacts with the calculator directly using its ``push`` method.
    """

    def __init__(self):
        self._calc = Calculator()
        self._result = ''

    def push_button(self, button):
        """Pushes the specified ``button``.

        The given value is passed to the calculator directly. Valid buttons
        are everything that the calculator accepts.

        Examples:
        | Push Button | 1 |
        | Push Button | C |

        Use `Push Buttons` if you need to input longer expressions.
        """
        self._result = self._calc.push(button)

    def push_buttons(self, buttons):
        """Pushes the specified ``buttons``.

        Uses `Push Button` to push all the buttons that must be given as
        a single string. Possible spaces are ignored.

        Example:
        | Push Buttons | 1 + 2 = |
        """
        for button in buttons.replace(' ', ''):
            self.push_button(button)

    def result_should_be(self, expected):
        """Verifies that the current result is ``expected``.

        Example:
        | Push Buttons     | 1 + 2 = |
        | Result Should Be | 3       |
        """
        if self._result != expected:
            raise AssertionError('%s != %s' % (self._result, expected))
```


Source : https://bitbucket.org/robotframework/robotdemo

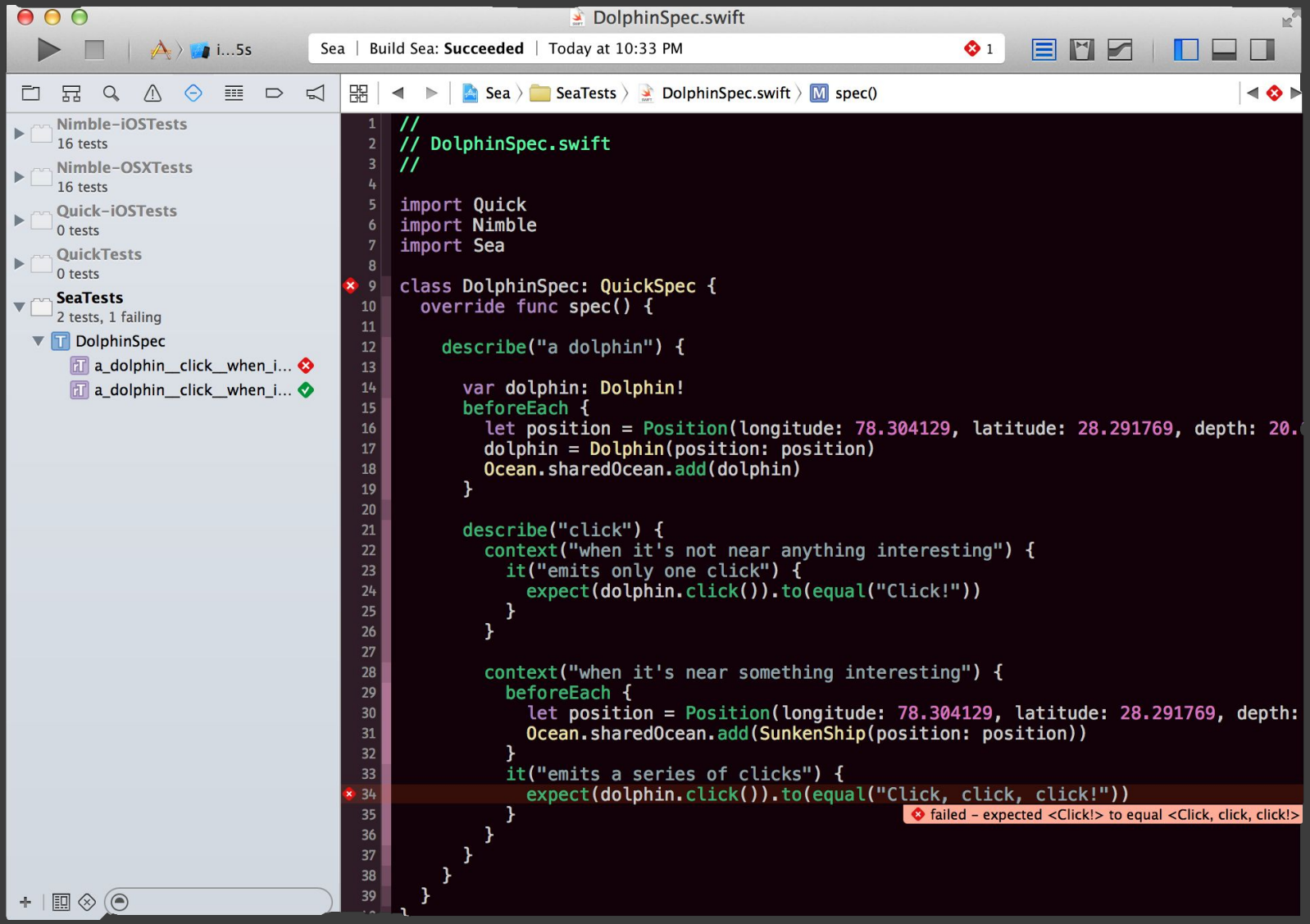
```
describe("A spec (with setup and tear-down)", function() {
  var foo;

  beforeEach(function() {
    foo = 0;
    foo += 1;
  });

  afterEach(function() {
    foo = 0;
  });

  it("is just a function, so it can contain any code", function() {
    expect(foo).toEqual(1);
  });

  it("can have more than one expectation", function() {
    expect(foo).toEqual(1);
    expect(true).toEqual(true);
  });
});
```

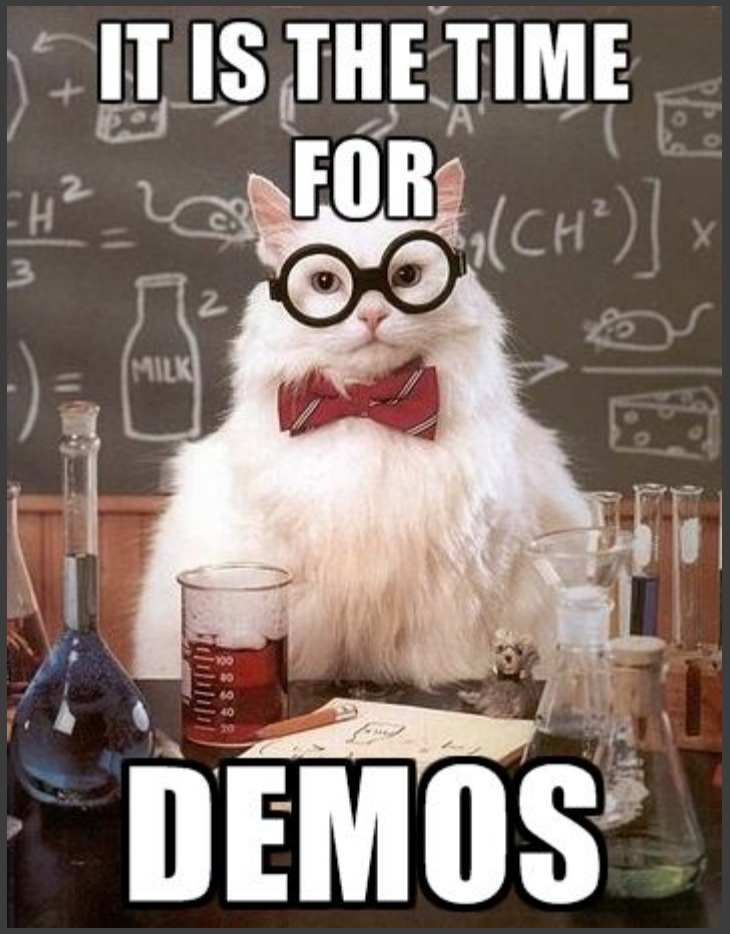
QUnit
js unit testing

```
// The dummy function to test
function isEven(val) {
    return val % 2 === 0;
}

// Let's test this function
test('isEven()', function() {
    ok(isEven(0), 'Zero is an even number');
    ok(isEven(2), 'So is two');
    ok(isEven(-4), 'So is negative four');
    ok(!isEven(1), 'One is not an even number');
    ok(!isEven(-7), 'Neither does negative seven');

    // Fails
    ok(isEven(3), 'Three is an even number');
})
```



Quick


DolphinSpec.swift
Sea | Build Sea: Succeeded | Today at 10:33 PM
Sea 〉 SeaTests 〉 DolphinSpec.swift 〉 spec()
Nimble-iOSTests
16 tests
Nimble-OSXTests
16 tests
Quick-iOSTests
0 tests
QuickTests
0 tests
SeaTests
2 tests, 1 failing
DolphinSpec
a_dolphin__click__when_i...
a_dolphin__click__when_i...
//
// DolphinSpec.swift
//
import Quick
import Nimble
import Sea
class DolphinSpec: QuickSpec {
override func spec() {
describe("a dolphin") {
var dolphin: Dolphin!
beforeEach {
let position = Position(longitude: 78.304129, latitude: 28.291769, depth: 20.
dolphin = Dolphin(position: position)
Ocean.sharedOcean.add(dolphin)
}
describe("click") {
context("when it's not near anything interesting") {
it("emits only one click") {
expect(dolphin.click()).to(equal("Click!"))
}
}
context("when it's near something interesting") {
beforeEach {
let position = Position(longitude: 78.304129, latitude: 28.291769, depth:
Ocean.sharedOcean.add(SunkenShip(position: position))
}
it("emits a series of clicks") {
expect(dolphin.click()).to(equal("Click, click, click!"))
}
failed - expected <Click!> to equal <Click, click, click!>



Code d'Armor

one more slide

- les tests sont indispensables
- les développeurs sont de bons testeurs du code
- les utilisateurs sont de bons testeurs du produit
- ce ne sont pas les outils qui manquent...


Merci !
pylapp.github.io
Dura test, sed test. Alea jacta test !
Code
d'Armor

Code d'Armor

- Des frameworks de tests
  - Appium
    - http://appium.io/
  - Catch
    - https://github.com/philsquared/Catch
  - Espresso
    - https://google.github.io/android-testing-support-library/docs/espresso/
  - Jasmine
    - http://jasmine.github.io/
  - JSUnit
    - http://www.jsunit.net/
  - JUnit
    - http://junit.org/
  - Mocha
    - https://mochajs.org/
  - Quick
    - https://github.com/Quick/Quick
  - QUnit
    - https://qunitjs.com/
  - Selendroid
    - http://selendroid.io/
  - Selenium
    - http://www.seleniumhq.org/
  - Robot Framework
    - http://robotframework.org/
  - Robolectric
    - http://robolectric.org/
  - Robotium
    - http://robotium.com/
  - UI Automator
    - http://developer.android.com/tools/testing-support-library/index.html#UIAutomator
  - UnitJS
    - http://unitjs.com/

- Des plateformes de tests
  - Google Cloud Test Lab
    - https://developers.google.com/cloud-test-lab/
  - Sauce Labs
    - http://saucelabs.com/
- Des robots de tests
  - Chrome Touch Bot
    - http://www.frandroid.com/marques/google/291985_chrome-touchbot-robot-teste-reactivite-appareils-de-google
  - Rob5X
    - http://www.keolabs.com/automation.html
  - Tapster
    - http://www.tapster.io/
- Des ordonnanceurs
  - Jenkins
    - https://jenkins-ci.org/
  - Hudson
    - http://hudson-ci.org/
- D'autres outils
  - HP Quality Center
    - https://saas.hpe.com/fr-fr/software/quality-center
  - HP Application Lifecycle Management
    - https://saas.hpe.com/fr-fr/software/application-lifecycle-management

- D’autres liens chouettes
  - http://fr.slideshare.net/tfrommen/an-introduction-to-software-testing
  - http://www.slideshare.net/UdayaSree/software-testing-life-cycle-presentation
  - http://blog.hubstaff.com/why-you-should-write-unit-tests/
  - http://blog.hubstaff.com/survey-many-developers-write-unit-tests/
  - http://artofunittesting.com/
  - http://programmers.stackexchange.com/questions/21133/how-to-write-good-unit-tests
  - http://blog.stevensanderson.com/2009/08/24/writing-great-unit-tests-best-and-worst-practises/
  - https://en.wikipedia.org/wiki/List_of_unit_testing_frameworks
  - http://android-developers.blogspot.fr/2015/12/leveraging-product-flavors-in-android.html
  - http://www.journaldugeek.com/2015/05/28/un-bug-qui-permet-a-un-sms-de-redemarrer-votre-iphone/
  - http://www.numerama.com/tech/135593-sncf-empeche-achat-de-billets-train-ter-sous-linux.html
  - https://fr.wikipedia.org/wiki/Logiciel_unique_%C3%A0_vocation_interarm%C3%A9es_de_la_solde
  - http://www.marmiton.org/recettes/recette_pate-a-crepes-des-plus-raffinees_49665.aspx
  - http://radonc.wikidot.com/radiation-accident-therac25
  - http://users.csc.calpoly.edu/~jdalbey/SWE/Papers/att_collapse.html
  - https://www.youtube.com/watch?v=_p3Qxl4736A
  - https://www.nirgal.net/mco_end.html
  - https://google.github.io/android-testing-support-library/
  - https://google.github.io/android-testing-support-library/docs/androidjunitrunner-guide/index.html
  - https://github.com/googlesamples/android-testing
  - https://developer.android.com/tools/testing-support-library/index.html
  - https://medium.com/@nileshjarad/how-to-do-tdd-in-android-90f013d91d7f#.dhvrjw8ug
  - https://medium.com/@nileshjarad/why-developers-scared-to-refactor-code-47efd1b854e7#.ojtyerioc
  - https://developer.android.com/studio/test/monkey.html
  - https://developer.android.com/studio/test/monkeyrunner/index.html
  - https://cucumber.io/
  - http://blog.soat.fr/2011/06/introduction-au-behavior-driven-development/
  - https://dannorth.net/introducing-bdd/
  - https://www.linkedin.com/pulse/tdd-vs-atdd-bdd-vahid-farahmandian
  - https://www.linkedin.com/pulse/agile-development-difference-between-tddatddbdd-komal-sureka
  - https://gaboesquivel.com/blog/2014/differences-between-tdd-atdd-and-bdd/


About being nice to your neighbors
SPEED LIMIT 80
Code d'Armor