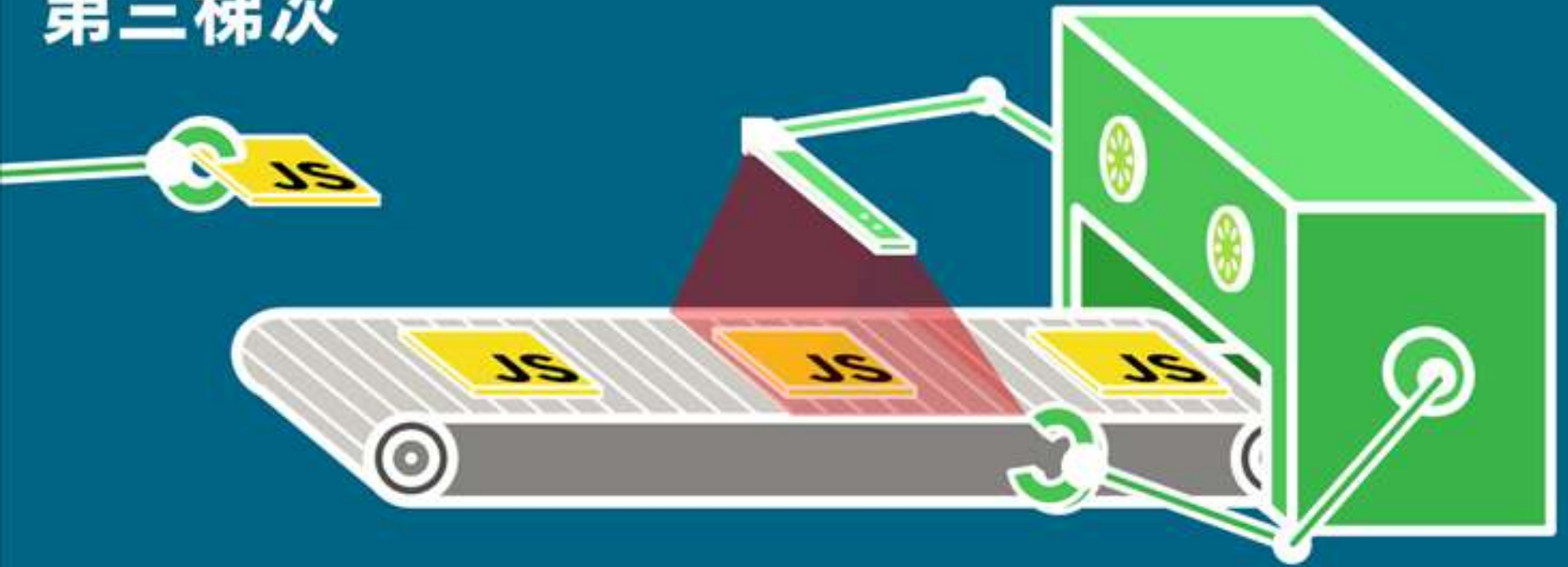


JavaScript 實務測試新手班

第三梯次



 SKILLTREE

2017/09/16 - 2017/09/23 14H

陳鋒逸 (陳小風)

自我介紹

- 陳鋒逸 (陳小風)
- 講師經歷
 - 微軟最有價值專家 (MVP)
 - SkillTree 兼任講師
 - Techday 講師 (2014)
 - JSDC 講師 (2013)
 - 社群研討會講師
 - AgileCommunity.tw
 - Javascript.tw
 - twMVC



粉絲團：愛流浪的小風

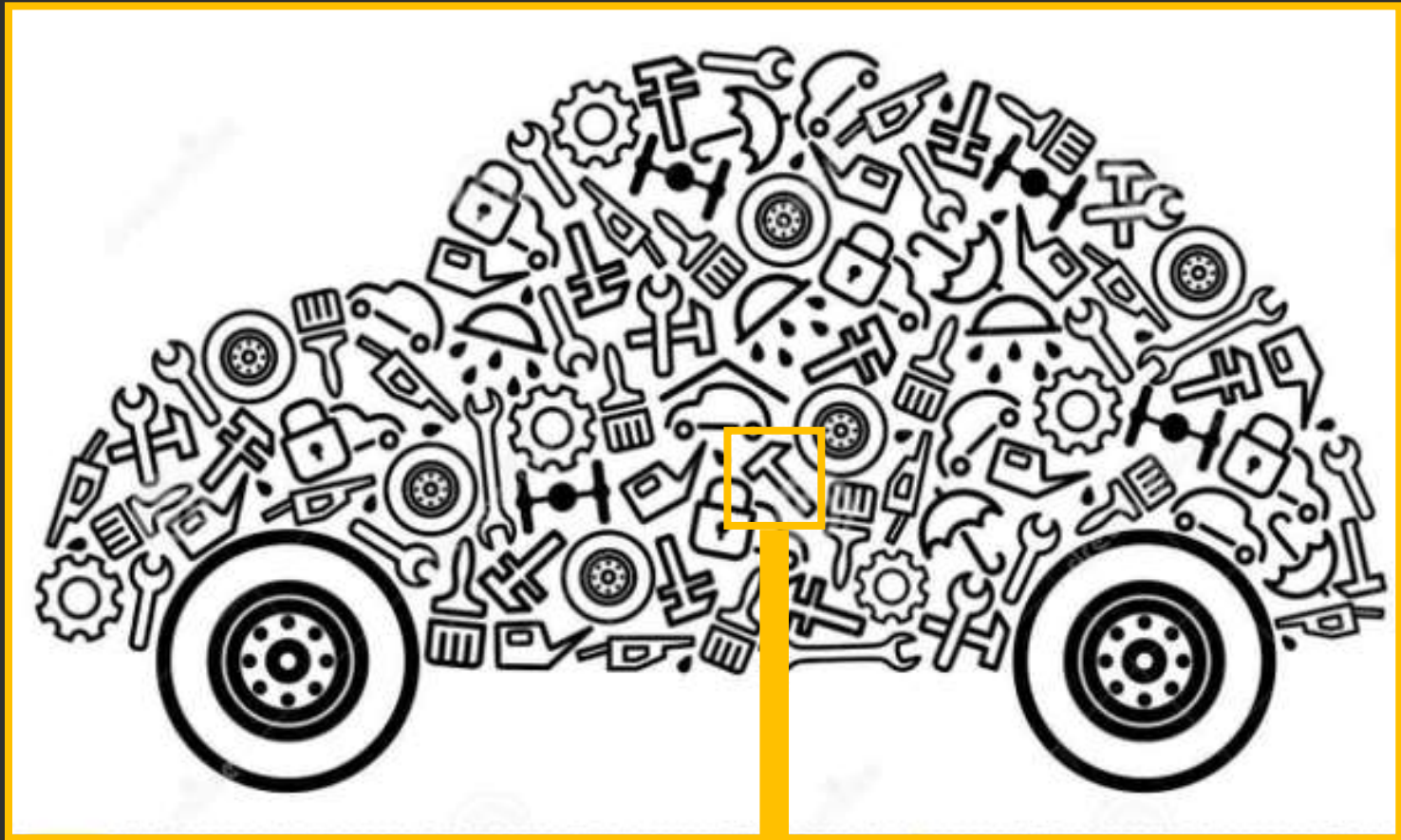


自動化測試

自動化測試

- 模擬人類操作的行為
- 最好的切入點
- 自動、重複執行預期的驗證行為
- 驗證預期行為正確

自動化測試



自動化測試

單元測試

Nightmare.js

- High-level browser automation library
- 使用 Electron
- 速度快
- 使用簡單
- 可顯示 UI

開始使用 Nightmare.js

- 安裝 Nightmare.js

```
npm install nightmare --save-dev
```

- 設定 Nightmare

```
const nightmare = Nightmare({  
  show: true  
});
```

搭配 mocha 使用

```
describe('Test Google', () => {  
  let nightmare;  
  before(() => {  
    nightmare = Nightmare({ show: true })  
  });  
  
  after(async () => {  
    await nightmare.end();  
  });  
  
  it('Google\'s title should be google', async function () {  
    let title = await nightmare  
      .goto('https://google.com')  
      .title();  
  
    title.should.be.equal('Google');  
  });  
});
```


High Level Api

- 網址操作

- goto
- back
- forward
- Refresh

- 狀態檢查

- wait
- exists
- visible

- 介面操作

- click
- mousedown
- mouseup
- type

- check
- select

- 額外技能

- evaluate
- screenshot

Browser 端執行程式碼

- `evaluate(fn(args...))`

```
const selector = 'h1';
nightmare
  .evaluate((selector) => {
    // browser 端執行
    return document.querySelector(selector).innerText;
  }, selector) // <-- 從 Node.js 端帶入變數
  .then((text) => {
    // ...
  })
```

自訂行為

▪ action

```
Nightmare.action('size', function(done) {  
  this.evaluate_now(() => {  
    var w = Math.max(document.documentElement.clientWidth, window.innerWidth ||  
0)  
    var h = Math.max(document.documentElement.clientHeight, window.innerHeight  
|| 0)  
    return {  
      height: h,  
      width: w  
    }  
  }, done)  
})
```

- 練習
 - 練習撰寫自動化測試

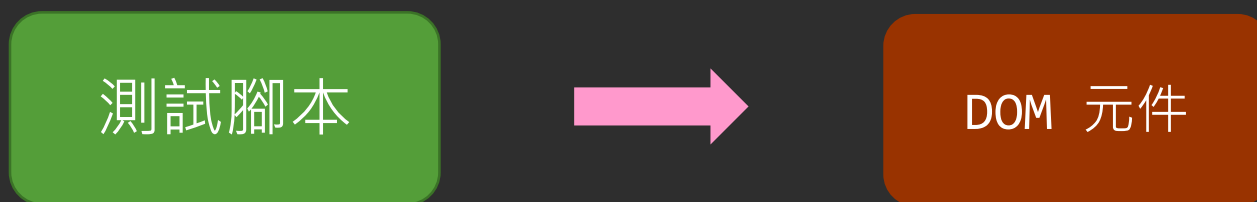


讓測試更好維護

- 畫面修改就要重寫測試
- 直接受到 **Selector** 的影響
- 改用 **PageObject**

<http://martinfowler.com/bliki/PageObject.html>

- 讓畫面修改時，成本減少

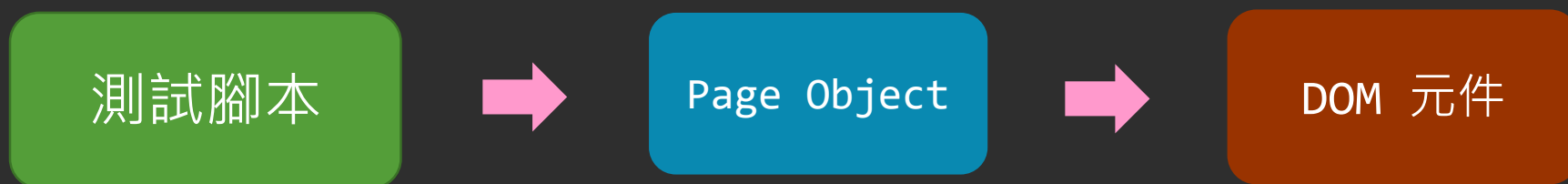


讓測試更好維護

- 畫面修改就要重寫測試
- 直接受到 **Selector** 的影響
- 改用 **PageObject**

<http://martinfowler.com/bliki/PageObject.html>

- 讓畫面修改時，成本減少



Page Object

■ 建立 ShoppingCartPage

```
function ShoppingCartPage(nightmare, rootUrl){  
  this.rootUrl = rootUrl;  
  this.nightmare = nightmare;  
  
  this.visit = async () => {  
    var url = `${this.rootUrl}/src/`;   
    await this.nightmare.goto(url);  
  }  
  
  this.selectLevel = async(level)=>{  
    await this.nightmare  
      .select('select[name=memberLevel]', level);  
  }  
  
  // Others  
}
```

Page Object

■ 改寫測試程式

```
it('VIP 會員購買 200 元商品 2 件, 結帳金額為 400 元', async () => {  
  // Arrange...  
  
  // Act  
  await shoppingCartPage.visit();  
  await shoppingCartPage.updateQty(productId, qty);  
  await shoppingCartPage.selectLevel(level);  
  
  actual = await shoppingCartPage.getPrice();  
  
  // Assert  
  actual.should.equal(expected);  
});
```


- 練習
 - 練習使用 PageObject 改寫程式碼



用測試來重構

什麼是重構?

- 減少 技術債
- 改善 Legacy Code
- 增加 擴充 的彈性
- 增進程式的 效能
- 讓程式碼更好維護

重構的困難點

- 遺失的 邏輯
- 沒有 文件
- 記憶 不明
- 線索只有 程式碼



前輩說...

真男人就是要看 Code !!!

重構起手式

撰寫測試

通過所有路徑

擷取邏輯

補強結構

重構起手式

- 撰寫 整合測試
- 確保 涵蓋率
- 重構程式碼
- 套用 物件導向、設計模式
- 補上 單元測試



- 練習
 - 練習重構



行為驅動開發

Behavior Driven Development

- 程式碼只有 開發人員 看得懂
- 使用 人類的語言 來描述程式行為
- 明確溝通需求
- 測試即文件
- 消除落差 (Gap)
- 留作紀錄 (歷史資料、邏輯演進)

使用 mocha 撰寫單元測試

```
describe('Calculator', function() {
  describe('#GetDiscountPrice(totalPrice)', function() {
    describe('if total price over 200, get 80% discount', function() {
      it('should return 200 if total price is 250', function() {
        var result = calculator.getDiscountPrice(250);

        result.should.equal(200);
      });
    });

    describe('if total price over 100 and less than 200, get 90% discount', function() {
      it('should return 135 if total price is 150', function() {
        var result = calculator.getDiscountPrice(150);

        result.should.equal(135);
      });
    });

    describe('if total price less than 100, no discount', function() {
      it('should return 80 if total price is 80', function() {
        var result = calculator.getDiscountPrice(80);

        result.should.equal(80);
      });
    });
  });
});
```

使用 Gherkins 撰寫需求

功能: Calculator

場景: 當金額高於 200 元時，可享有 80% 折扣

假設 顧客消費總金額為 "250" 元

當 計算折扣後金額

那麼 折扣後金額應該為 "200" 元

場景: 當金額高於 100 元，但不高於 200 元時，可享有 90% 折扣

假設 顧客消費總金額為 "150" 元

當 計算折扣後金額

那麼 折扣後金額應該為 "135" 元

場景: 當金額小於 100 元，沒有折扣

假設 顧客消費總金額為 "80" 元

當 計算折扣後金額

那麼 折扣後金額應該為 "80" 元

Cucumber

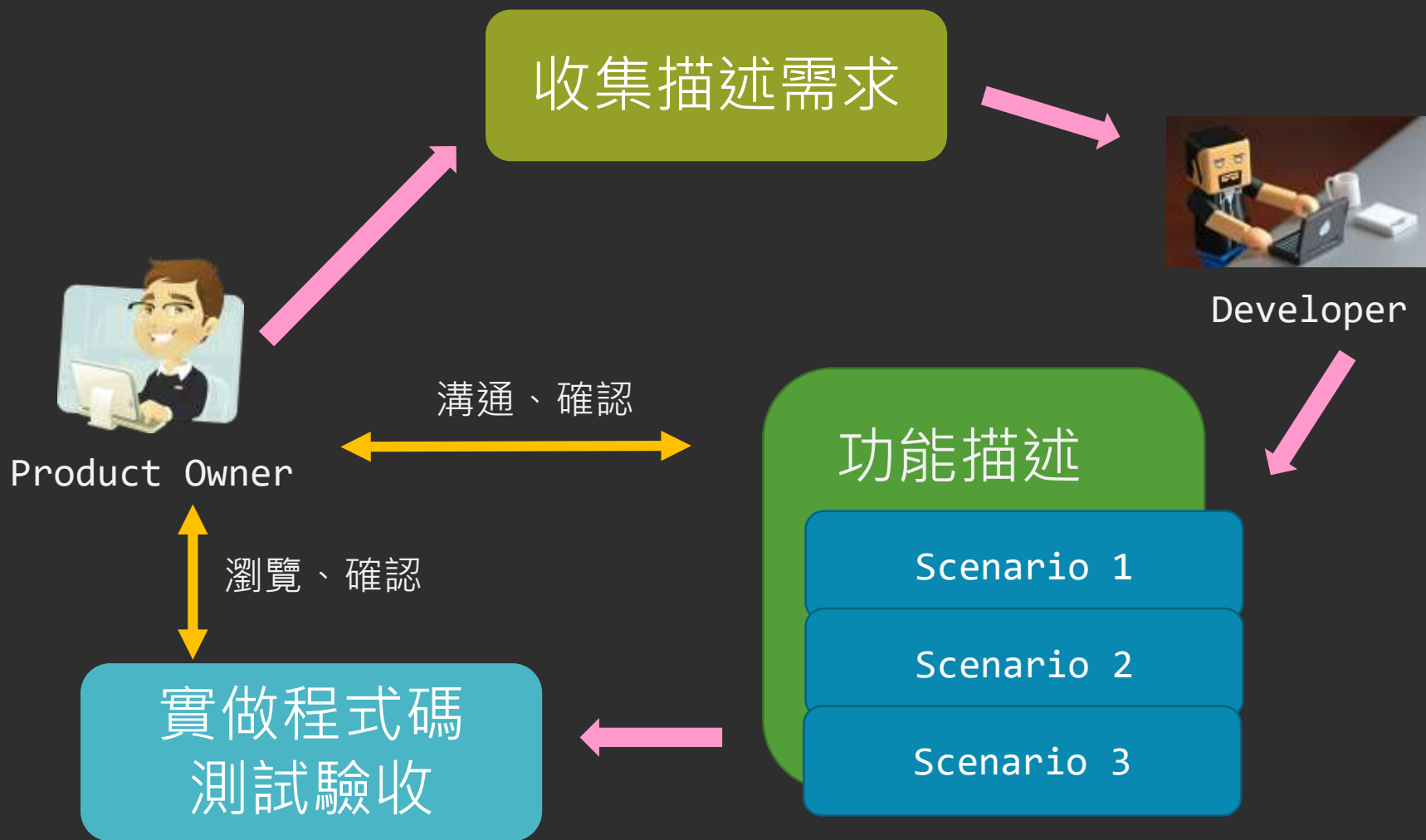
- BDD 工具
- 很多程式語言都有實作的 Framework
<https://cucumber.io/docs>
- Cucumber.js – JavaScript 的 BDD 工具
<http://cucumber.github.io/cucumber-js/>



Gherkin

- Cucumber 所使用的描述方式
 - 描述 商業邏輯、程式行為
 - 一種 Domain Specific Languages
- <http://huan-lin.blogspot.com/2008/05/domain-specific-languages.html>
- 支援多國語言 （包括 中文）

改善開發流程



開發前整理完整的需求

Tree: f428f4adc · cucumber-js-sample / features / calculator.feature

Find file Copy path

kirkchen Add member level feature f428f4f an hour ago

1 contributor

30 lines (24 sloc) 1.03 KB

Raw Blame History

```
1 #language: zh-TW
2 功能: Calculator
3
4 場景: 當金額高於 200 元時, 可享受 80% 折扣
5     假設 顧客消費總金額為 "250" 元
6     當 計算折扣後金額
7     那麼 折扣後金額應該為 "200" 元
8
9 場景: 當金額高於 100 元, 但不高於 200 元時, 可享受 90% 折扣
10    假設 顧客消費總金額為 "150" 元
11    當 計算折扣後金額
12    那麼 折扣後金額應該為 "135" 元
13
14 場景: 當金額小於 100 元, 沒有折扣
15    假設 顧客消費總金額為 "80" 元
16    當 計算折扣後金額
17    那麼 折扣後金額應該為 "80" 元
18
19 場景: 當顧客是 VIP 時, 可享受 20 元折扣
20    假設 顧客消費總金額為 "90" 元
21    並且 顧客會員等級是 "VIP"
22    當 計算折扣後金額
23    那麼 折扣後金額應該為 "70" 元
24
25 場景: 當顧客是一般會員 (Normal) 時, 可享受 10 元折扣
26    假設 顧客消費總金額為 "90" 元
27    並且 顧客會員等級是 "Normal"
28    當 計算折扣後金額
29    那麼 折扣後金額應該為 "80" 元
```


<https://github.com/kirkchen/cucumber-js-sample/pull/1>

用明確的案例討論程式邏輯


新增會員折扣邏輯 #1 Edit


Merged kirkchen merged 2 commits into master from feature/add_new_feature an hour ago

Conversation 1 Commits 2 Files changed 4 +44 -4

 kirkchen commented an hour ago Owner + 🗨️ ✎️


Add new member only discount logic

◊  Add member level feature ✓ f428f4f

<>  kirkchen commented on the diff an hour ago

features/calculator.feature View full changes

+++	+++	00 -15,3 +15,15
15	15	假設 顧客消費總金額為 "80" 元
16	16	當 計算折扣後金額
17	17	那麼 折扣後金額應該為 "80" 元
	18	+
	19	+ 場景: 當顧客是 VIP 時, 可享有 20 元折扣

 kirkchen added a note an hour ago Owner + 🗨️ ✎️ ✕

VIP 折扣金額會不會太高?

👍 1


Add a line note

Labels ⚙️
None yet

Milestone ⚙️
No milestone

Assignees ⚙️
No one—assign yourself

1 participant



Notifications

🔊 Unsubscribe

You're receiving notifications because you modified the open/close state.

🔒 Lock conversation

<https://github.com/kirkchen/cucumber-js-sample/pull/1>

文件也是測試程式

The screenshot displays a GitHub pull request interface. At the top, a comment from 'kirkchen' indicates a change to the file 'features/calculator.feature'. The diff shows several lines of text in Chinese, with line 18 being added and line 19 being modified. Below the diff, a note from 'kirkchen' asks 'VIP 折扣金額會不會太高?' (Will the VIP discount amount be too high?). The note has one thumbs-up reaction. A red box highlights a status message: 'All checks have passed' with '2 successful checks'. The checks listed are 'continuous-integration/travis-ci/pr' and 'continuous-integration/travis-ci/push', both marked as successful. On the right side, there are sections for 'Milestone' (No milestone), 'Assignees' (No one—assign yourself), '1 participant' (with a profile picture), and 'Notifications' (with an 'Unsubscribe' button). At the bottom right, there is a 'Lock conversation' option.

kirkchen commented on the diff an hour ago

features/calculator.feature [View full changes](#)

...	...	@@ -15,3 +15,15 @@
15	15	假設 顧客消費總金額為 "80" 元
16	16	當 計算折扣後金額
17	17	那麼 折扣後金額應該為 "80" 元
	18	+
	19	+ 場景: 當顧客是 VIP 時, 可享受 20 元折扣

kirkchen added a note an hour ago Owner + 😊 ✎ ✕

VIP 折扣金額會不會太高?

👍 1

[Add a line note](#)

Implement member di

kirkchen merged comm
2 checks passed

All checks have passed
2 successful checks:

- 🟢 [continuous-integration/travis-ci/pr](#) — The Travi... [Details](#)
- 🟢 [continuous-integration/travis-ci/push](#) — The T... [Details](#)

kirkchen deleted the feature/add_new_feature branch an hour ago [Restore branch](#)

Milestone ⚙️
No milestone

Assignees ⚙️
No one—assign yourself

1 participant

Notifications
[🔊 Unsubscribe](#)
You're receiving notifications because you modified the open/close state.

[🔒 Lock conversation](#)

搭配 Continuous Integration

kirkchen / cucumber-js-sample  build passing

Current Branches Build History Pull Requests

More options 

✓ feature/implement_coupon_discount Add calculator description

→ #12 passed 

Commit d178d75

Compare 3e0e857..d178d75

⌚ Elapsed time 31 sec

📅 about an hour ago

 Kirk Chen authored and committed

Remove log Raw log

```
1 Using worker: worker-linux-docker-00ab0245.prod.travis-ci.org:travis-linux-16
2
3 Build system information system_info
67
68 $ export DEBIAN_FRONTEND=noninteractive
69 $ git clone --depth=50 --branch=feature/implement_coupon_discount https://github.com/kirkchen/cucumber-js-sample kirkchen/cucumber-js-sample Fix CVE-2015-7547 git_checkout 0.78s
117
118 This job is running on container-based infrastructure, which does not allow use of 'sudo', setuid and setgid executables.
119 If you require sudo, add 'sudo: required' to your .travis.yml
120 See https://docs.travis-ci.com/user/workers/container-based-infrastructure/ for details.
121 Updating nvm to v0.31.0
122 $ nvm install 6.2.0
123 Downloading https://nodejs.org/dist/v6.2.0/node-v6.2.0-linux-x64.tar.xz...
124 ##### 100.0%
125 Now using node v6.2.0 (npm v3.8.9)
126
127 Starting with io.js 3 and Node.js 4, building native extensions requires C++11-compatible compiler, which seems unavailable on this VM. Please read https://docs.travis-ci.com/user/languages/javascript-with-nodejs#Node.js-v4-(or-io.js-v3)-compiler-requirements.
128 $ node --version
129 v6.2.0
130
```

<https://travis-ci.org/kirkchen/cucumber-js-sample>

測試即文件

Browse documentation

Calculator

Calculator

為了根據顧客消費金額，會員等級和折價券資料
提供專門計算金額的 Calculator
用來計算折扣後的金額

Scenarios

- 當金額高於 200 元時，可享有 80% 折扣
- 當金額高於 100 元，但不高於 200 元時，可享有 90% 折扣
- 當金額小於 100 元，沒有折扣
- 當顧客是 VIP 時，可享有 20 元折扣
- 當顧客是一般會員（Normal）時，可享有 10 元折扣
- 當顧客使用折價券時，可享有折價券折扣

○ 當金額高於 200 元時，可享有 80% 折扣

假設 顧客消費總金額為 "250" 元
當 計算折扣後金額
那麼 折扣後金額應該為 "200" 元

<http://www.relishapp.com/kirkchen/cucumber-js-sample/docs/calculator>

完整循環

確認需求

撰寫程式

測試邏輯正確

產生文件

- 練習
 - 練習使用 Gherkins 撰寫需求



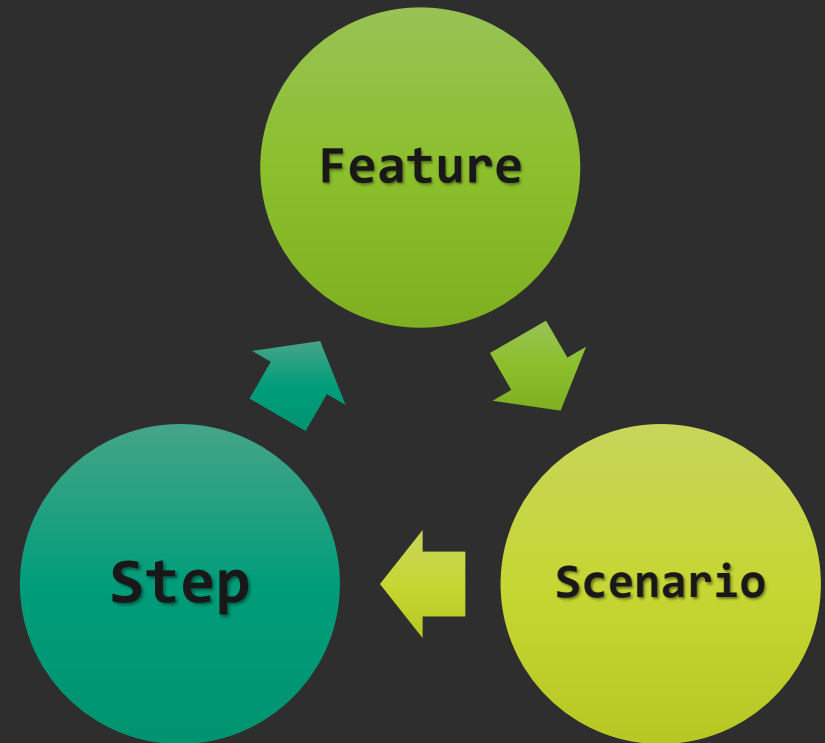
小結

- BDD
- 建立循環
- 有效溝通

會說話的測試程式

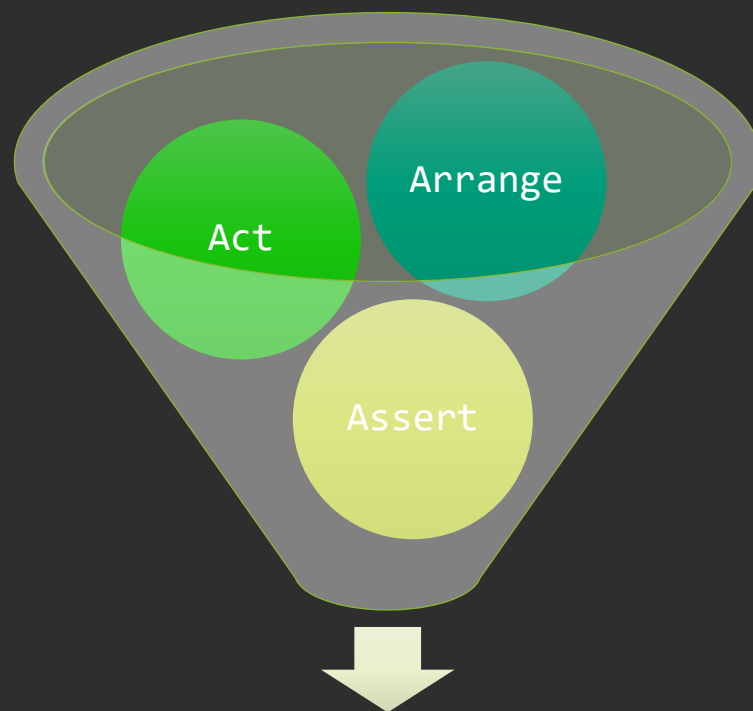
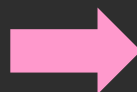
描述行為

- **Feature** (功能)
功能說明、使用情境
- **Scenario** (場景、劇本)
操作的方式、特定流程
- **Step** (步驟)
流程的步驟



操作步驟

- **Given (假設)**
前情提要、初始化
- **When (當)**
執行邏輯、呼叫物件
- **Then (那麼)**
檢查結果



Unit Test 3A

使用 `Cucumber.js` 撰寫測試

- 建立 `features/Calculator.feature`

功能: `Calculator`

為了根據顧客消費金額，
提供專門計算金額的 `Calculator` 用來計算折扣後的金額

場景：當金額高於 200 元時，可享有 80% 折扣

假設 顧客消費總金額為 "250" 元

當 計算折扣後金額

那麼 折扣後金額應該為 "200" 元

使用 Cucumber.js 撰寫測試

■ 產生測試程式框架 `npm test`

功能：Calculator

為了根據顧客消費金額，會員等級和折價券資料
提供專門計算金額的 Calculator
用來計算折扣後的金額

場景,劇本：當金額高於 200 元時，可享有 80% 折扣
假設顧客消費總金額為 "250" 元
當計算折扣後金額
那麼折扣後金額應該為 "200" 元

Warnings:

1) Scenario: 當金額高於 200 元時，可享有 80% 折扣 - features/calculator.feature:7
Step: 假設顧客消費總金額為 "250" 元 - features/calculator.feature:8

Message:

Undefined. Implement with the following snippet:

```
this.Given(/^顧客消費總金額為 "([^"]*)" 元$/, function (arg1, callback) {  
  // Write code here that turns the phrase above into concrete actions  
  callback(null, 'pending');  
});
```

使用 Cucumber.js 撰寫測試

- 實做測試至 features/CalculatorSteps.js

```
var { defineSupportCode } = require('cucumber');
```

```
defineSupportCode(function ({ Given, When, Then }) {  
  Given(/^顧客消費總金額為 "([^"]*)" 元$/, function(totalPrice) {  
    this.totalPrice = parseInt(totalPrice);  
  });
```

```
  When(/^計算折扣後金額$/, function() {  
    this.result = calculator.getDiscountPrice(this.totalPrice);  
  });
```

```
  Then(/^折扣後金額應該為 "([^"]*)" 元$/, function(result) {  
    this.result.should.equal(parseInt(result));  
  });  
})
```

使用 Cucumber.js 撰寫測試

■ 執行測試 `npm test`

```
kirkchen ~/Codes/cucumber-js-sample feature/implement_coupon_discount npm test
```

```
> cucumber-js-sample@1.0.0 test /Users/kirkchen/Codes/cucumber-js-sample
> cucumber-js
```

功能：Calculator

為了根據顧客消費金額，會員等級和折價券資料
提供專門計算金額的 Calculator
用來計算折扣後的金額

場景，劇本：當金額高於 200 元時，可享有 80% 折扣
假設顧客消費總金額為 "250" 元
當計算折扣後金額
那麼折扣後金額應該為 "200" 元

```
1 scenario (1 passed)
3 steps (3 passed)
0m00.003s
```

Cucumber.js

- Cucumber 在 JavaScript 的版本
- 由 **Feature** 和 **Step** 所組合而成

Calculator.feature

+

CalculatorStep.js

Cucumber.js

Calculator.feature

CalculatorStep.js

功能: Calculator


為了根據顧客消費金額，會員等級和折價券資料
提供專門計算金額的 Calculator
用來計算折扣後的金額

場景: 當金額高於 200 元時，可享有 80% 折扣

假設 顧客消費總金額為 "250" 元

當 計算折扣後金額

那麼 折扣後金額應該為 "200" 元



```
this.Given(/^(顧客消費總金額為 "([^\"]*)" 元$/,  
  function(totalPrice) {  
    this.totalPrice = parseInt(totalPrice);  
  }  
);
```

透過 Regular Expression 解析參數

Cucumber.js

CalculatorStep.js

```
this.Given(/^顧客消費總金額為 "([^"]*)" 元$/, function(totalPrice) {  
    this.totalPrice = parseInt(totalPrice);  
});  
  
this.When(/^計算折扣後金額$/, function() {  
    this.result = calculator.getDiscountPrice(this.totalPrice);  
});  
  
this.Then(/^折扣後金額應該為 "([^"]*)" 元$/, function(result) {  
    this.result.should.equal(parseInt(result));  
});
```



Assertion Library - 驗證結果與預期一致

- 練習
 - 練習根據需求撰寫測試



小結

- 描述行為
- 操作步驟
- 執行測試

Cucumber 使用心法

將測試分組

- 使用 **Tag** 將測試案例分組

@ShoppingCart

假設 購物車中有商品如下

name	price
Item 1	50
Item 2	100

當 計算總價格

那麼 總價格應該為 "150"

- 只執行特定 **Tag** 的測試

cucumber.js --tags @ShoppingCart	With @ShoppingCart
cucumber.js --tags ~@ShoppingCart	Not @ShoppingCart
cucumber.js --tags @ShoppingCart,@Product	Or
cucumber.js --tags @ShoppingCart --tags @Product	And

全域事件

- Before / BeforeAll

```
Before(function(scenario){  
    ///// 設定全域物件  
})
```

- After / AfterAll

```
After(function(scenario){  
    ///// 關閉釋放資源  
})
```

Tags

- 在分類前後執行程式

```
defineSupportCode(function({After, Before}) {  
  Before({tags: "@foo"}, function () {  
    // 在 @foo 的 Feature 前執行  
  });  
});
```

```
Before({tags: "@foo and @bar"}, function () {  
  // 在有 @foo 和 @bar 的 Feature 前執行  
});
```

```
Before({tags: "@foo or @bar"}, function () {  
  // 在有 @foo 或 @bar 的 Feature 前執行});  
});
```

表示複雜物件的時候

- 用 **Json** 表示購物車資料

假設 購物車中有商品

```
""  
[  
  {  
    "name": "Item 1",  
    "price":50  
  },  
  {  
    "name": "Item 2",  
    "price":100  
  }  
]  
""
```



不容易閱讀

當 計算總價格

那麼 總價格應該為 "150"

表示複雜物件的時候

- 改用 **Table** 表示購物車資料

假設 購物車中有商品如下

name	price
Item 1	50
Item 2	100



可讀性較高

當 計算總價格

那麼 總價格應該為 "150"

使用 Table 表示物件

- Cucumber 支援 4 種 Table 轉換格式

- `table.raw()`

Item 1	50
Item 2	100



```
[  
  ['Item 1', 50],  
  ['Item 2', 100]  
]
```

- `table.rows()`

name	price
Item 1	50
Item 2	100



```
[  
  ['Item 1', 50],  
  ['Item 2', 100]  
]
```

使用 Table 表示物件

- Cucumber 支援 4 種 Table 轉換格式

- `table.rowsHash()`

Item 1	50
Item 2	100



```
{  
  'Item 1': 50,  
  'Item 2': 100  
}
```

- `table.hashes()`

name	price
Item 1	50
Item 2	100



```
[  
  {  
    "name": "Item 1",  
    "price": 50  
  },  
  {  
    "name": "Item 2",  
    "price": 100  
  }  
]
```

- 練習
 - 練習使用 Tags
 - 練習使用 Hooks
 - 練習使用 Table



將需求變成自動化測試

轉換需求

實例化需求

自動化測試

實做程式碼

單元測試

重構

使用 Cucumber 撰寫自動化測試

■ 撰寫 Feature

language: zh-TW

功能: 購物車

建立一個 購物車 應用程式，
必須要能夠根據會員的等級，提供不同的折扣方式。

- * 如果是 VIP 會員，只要購物滿 500 元，就一律有 8 折優惠
- * 如果是一般會員 (Normal)，除了購物必須要滿 1000 元，
而且購買超過 3 件商品才能擁有 85 折優惠

場景: VIP 會員, 購買 200 元商品 3 件, 結帳金額為 480 元
假設 進入購物車頁面
並且 選擇第 "1" 個商品，價格為 "200" 元，數量 "3" 件
並且 選擇會員等級為 "VIP"
當 選擇完畢，計算價格
那麼 折扣後價格為 "480" 元

使用 Cucumber 撰寫自動化測試

- 使用 World 設定共用工具

```
var { defineSupportCode } = require('cucumber');  
var Nightmare = require('nightmare');  
const nightmare = Nightmare({ show: true });
```

```
function CustomWorld({ attach }) {  
  this.driver = nightmare;  
}
```

```
defineSupportCode(function ({ setWorldConstructor, AfterAll }) {  
  setWorldConstructor(CustomWorld)
```

```
  AfterAll(async function () {  
    await nightmare.end()  
  })  
})
```


使用 Cucumber 撰寫自動化測試

■ 撰寫測試

```
Given('進入購物車頁面', async function () {  
  await this.driver.goto(url)  
  .viewport(1000, 760);  
});
```

```
Given('選擇第 {string} 個商品，價格為 {string} 元，數量 {string} 件', async function  
(index, price, qty) {  
  await this.driver  
    .insert(`div.product:nth-child(${index}) input[name=qty]`, false)  
    .insert(`div.product:nth-child(${index}) input[name=qty]`, +qty);  
});
```

```
// Others
```

- 練習
 - 練習使用 Cucumber 撰寫自動化測試



小結

- Tags
- Hook
- Table
- 自動化測試

多瀏覽器自動化測試

整合測試

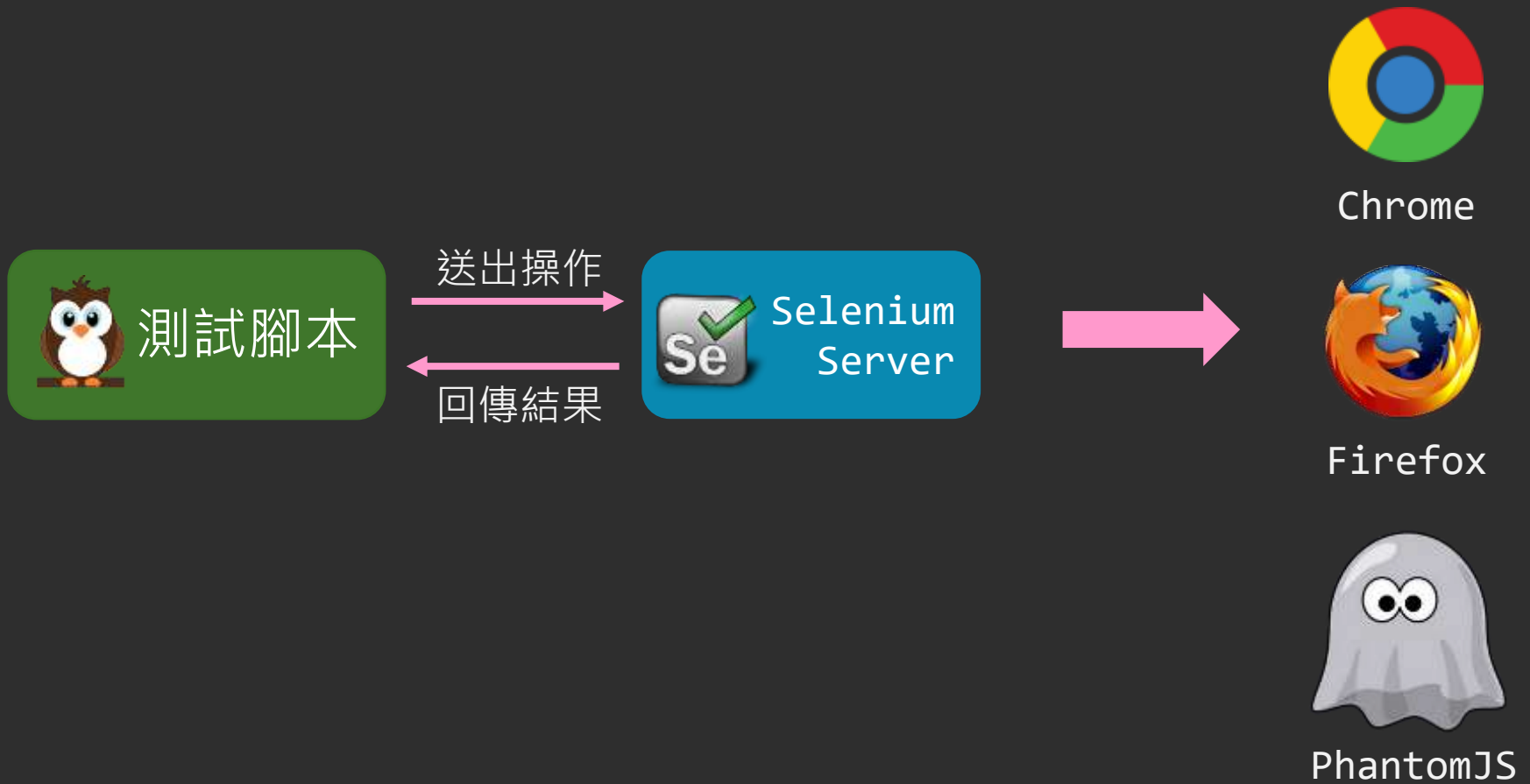
- 軟體測試最後的防火牆
- 包含外部資源的使用
(例如存取檔案、資料庫)
- 可以直接從 UAT 轉換
- 效益最高

NightWatch

- 基於 Node.js 的 E2E 測試工具
- 包裝 Selenium 而成
- 支援雲端測試服務
<https://saucelabs.com/>
- 可搭配 Cucumber 使用



NightWatch



支援 Cucumber

- 描述你操作的瀏覽器的行為

功能： Google Search
實做搜尋功能

場景： 搜尋 Google
假設 打開 Google 搜尋頁面
那麼 標題應該是 "Google"
並且 存在 Google 搜尋框

支援 Cucumber

- 撰寫操作步驟的程式碼

<http://nightwatchjs.org/api>

```
this.Given(/^打開 Google 搜尋頁面$/, function () {  
    this  
        .url('http://google.com')  
        .waitForElementVisible('body', 1000)  
})
```

```
this.Then(/^標題應該是 "([^"]*)"$/, function (title) {  
    this.assert.title(title)  
})
```

```
this.Then(/^存在 Google 搜尋框$/, function () {  
    this.assert.visible('input[name="q"]')  
})
```

支援 Cucumber

■ 執行測試

```
1. kirkchen@ChenFengVideoM8P: ~/Codes/nightwatch-cucumber-test (zsh)
Starting selenium server... started - PID: 15968

[Google] Test Suite
=====
功能：Google Search

    實做搜尋功能

Running: 搜尋 Google
    場景,劇本：搜尋 Google
    ✓ Element <body> was visible after 145 milliseconds.
      假設打開 Google 搜尋頁面
    ✓ Testing if the page title equals "Google".
      那麼標題應該是 "Google"
    ✓ Testing if element <input[name="q"]> is visible.
      並且存在 Google 搜尋框

OK. 3 assertions passed. (1.78s)

1 scenario (1 passed)
3 steps (3 passed)
0m01.778s
kirkchen ~/Codes/nightwatch-cucumber-test 07:19:26
```

- 練習
 - 練習使用 `nightwatch` 撰寫測試



小結

- 整合測試
- Nightwatch

測試文件再進化

寫完測試了...然後呢？

- 測試，不只是測試
- 產生測試報告
- 可以瀏覽的文件
- 有效產出
- 保持系統穩定

產生報表

- 讓測試的結果圖表化

- 使用 `cucumber-html-report` 套件

<https://github.com/leinonen/cucumber-html-report>

- 匯出 `cucumber` 結果

```
cucumber-js -f json:cucumber_report.json
```

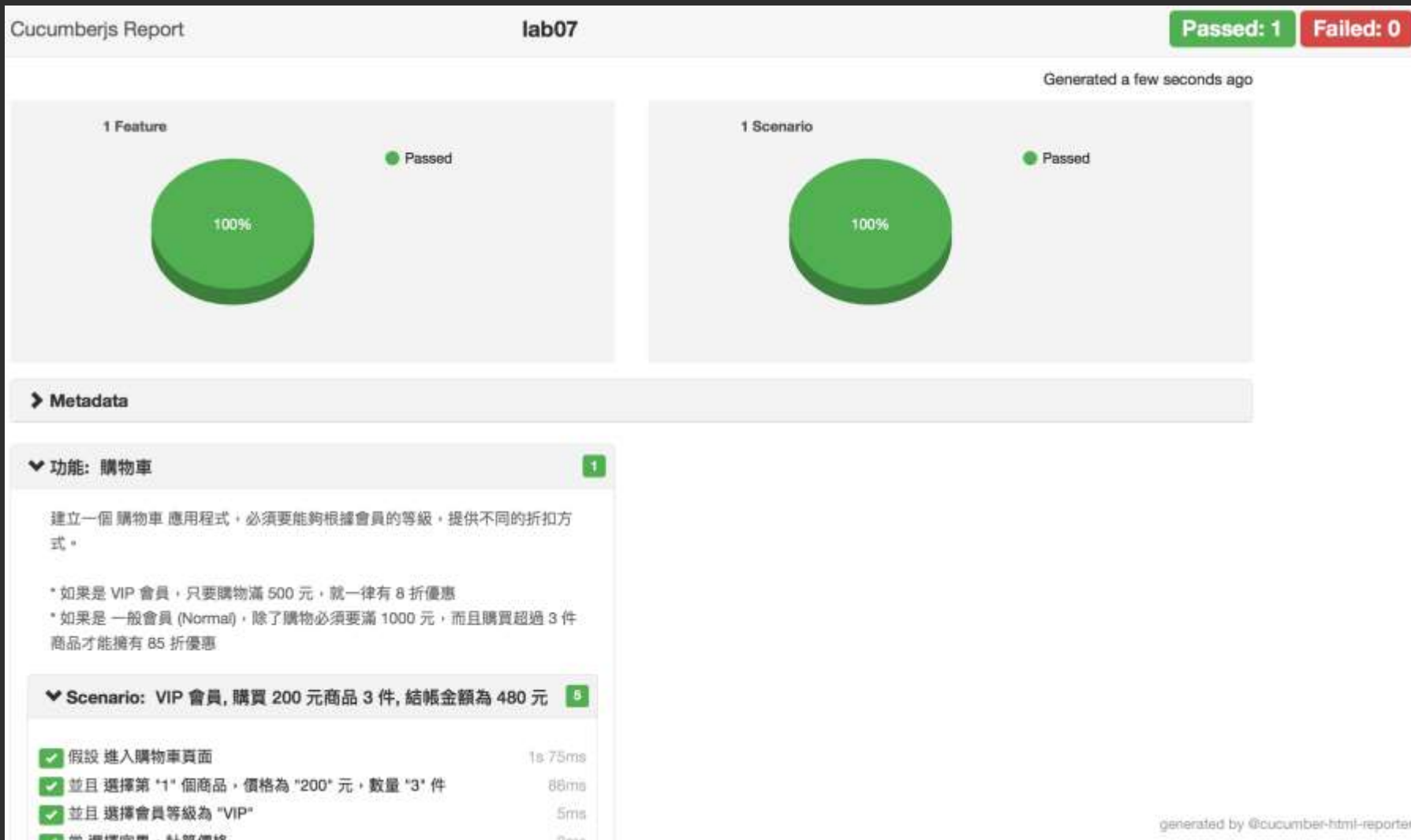
- 新增 `report.js`

```
var reporter = require('cucumber-html-reporter');
```

```
var options = {  
  // Settings  
};
```

```
reporter.generate(options);
```

產生報表



- 練習
 - 練習產生報表

測試涵蓋率

- Istanbul – 測試涵蓋率計算工具
<https://github.com/gotwarlost/istanbul>
- 同時執行測試並計算測試涵蓋率
`istanbul cover _mocha -- tests/`
- 維持涵蓋率，確保程式穩定
- 產生報表檢查是否有遺漏測試

測試涵蓋率

[all files](#) / [app/](#) calculator.js

100% Statements 13/13

100% Branches 10/10

100% Functions 2/2

100% Lines 13/13

```
1 1x function calculator() {
2 1x   this.getDiscountPrice = function(totalPrice, memberLevel, couponCode) {
3 5x     var finalPrice = totalPrice;
4
5 5x     if (totalPrice >= 200) {
6 1x       finalPrice = totalPrice * 0.8;
7 4x     } else if (totalPrice >= 100 && totalPrice < 200) {
8 1x       finalPrice = totalPrice * 0.9
9     }
10
11 5x    if (memberLevel === 'VIP') {
12 1x      finalPrice = finalPrice - 20;
13 4x    } else if (memberLevel === 'Normal') {
14 1x      finalPrice = finalPrice - 10;
15    }
16
17 5x    return finalPrice;
18  }
19 }
20
21 1x module.exports = calculator;
22
```

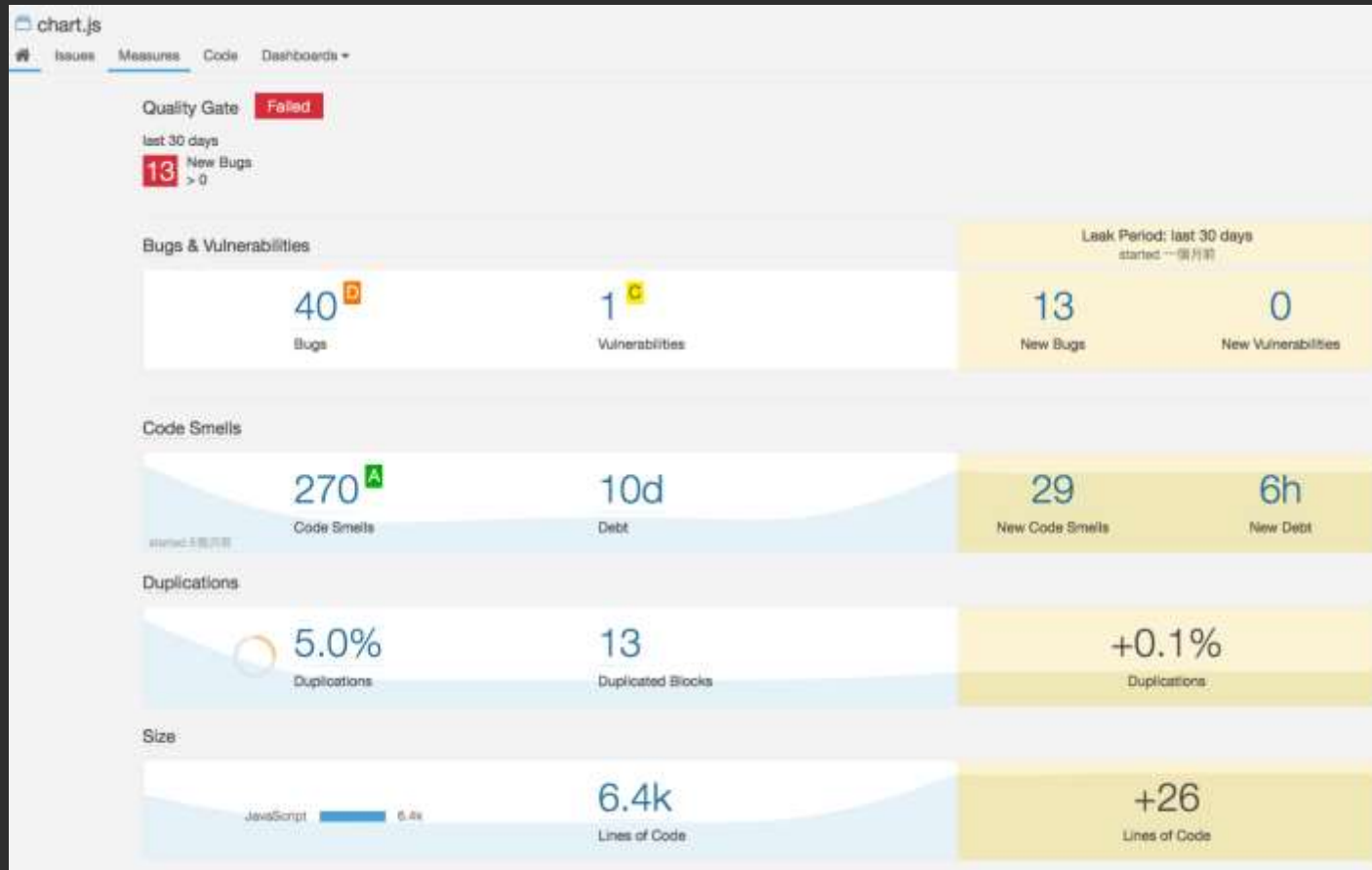
程式碼的健康檢查

- 檢查程式碼的品質
- 量化數據
- 計算測試程式碼涵蓋率

程式碼的健康檢查

■ SonarQube

<https://sonarcloud.io/projects>



- 練習
 - 練習產生涵蓋率報告



小結

- 產生報表
- 測試涵蓋率
- 軟體品質檢查

Homework

哈利波特一到五冊熱潮正席捲全球，世界各地的孩子都為之瘋狂。#出版社為了慶祝Agile Tour第一次在台灣舉辦，決定訂出極大的優惠，來回饋給為了小孩四處奔波買書的父母親們。

定價的方式如下：

1. 一到五集的哈利波特，每一本都是賣100元
2. 如果你從這個系列買了兩本不同的書，則會有5%的折扣
3. 如果你買了三本不同的書，則會有10%的折扣
4. 如果是四本不同的書，則會有20%的折扣
5. 如果你一次買了整套一到五集，恭喜你將享有25%的折扣
6. 需要留意的是，如果你買了四本書，其中三本不同，第四本則是重複的，那麼那三本將享有10%的折扣，但重複的那一本，則仍須100元。

你的任務是，設計一個哈利波特的購物車，能提供最便宜的價格給這些爸爸媽媽。

Blog 是記錄知識的最佳平台



<https://dotblogs.com.tw>

OzCode

Your Road to Magical Debugging



```
float CalculateCost( Customer customer, string restaurant)
{
    float courseCost = GetCourseCost(restaurant);
    bool shouldTip = waiter.IsNice && courseCost > COSTLY_MEAL;
```

Exceptions Trail:



ReservationException HotelException **IndexOutOfRangeException**

Exception:

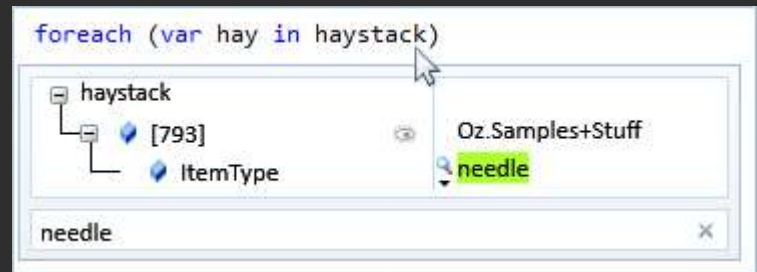
System.IndexOutOfRangeException

Message:

"Invalid customer index"

[Go to where exception was thrown](#) [Go to where exception was handled](#)

<http://www.oz-code.com/>



學員可使用 Yammer 取得優惠價

謝謝各位

<https://skilltree.my>

-
- 本投影片所包含的商標與文字皆屬原作者所有，僅供教學之用。
 - 本投影片的內容包括標誌、設計、文字、圖像、影片、聲音...等著作財產權均屬電魔小鋪有限公司所有，受到中華民國著作權法及國際著作權法律的保障。對本投影內容進行任何形式的引用、轉載、重製前，請務必取得電魔小鋪有限公司的"書面授權"，否則請勿使用，以免侵權。