沒有 HDD 的網站重構 WebForm to MVC

twMVC#31 2017-12-02 Kevin Tseng



個人簡介

- 微軟最有價值專家（MVP）2013~2017
- twMVC 核心成員及講師 （http://mvc.tw）
- SkillTree 兼任講師 （http://skilltree.my）
- 部落格「mrkt 的程式學習筆記」
  http://kevintsengtw.blogspot.tw
- Github
  https://github.com/kevintsengtw
- Facebook 社團「軟體架構與實務測試」

主題說明

- 「ASP.NET WebForm 重構為 ASP.NET MVC」

  這是個老掉牙的議題。

  如今 ASP.NET Core 都已經發佈 2.0 的時候，怎麼還會講 WebForm 轉為 MVC 的內容呢？

主題說明

- 這樣的內容一直到現在都還是無時無刻地到處上演

  很多公司都面臨這樣的處境

- 這次的講題內容就分享一個重構的實戰經驗

  讓大家瞭解在重構的過程裡會應用到的技術與做法

注意事項

- **禁止攝影、禁止錄影、禁止錄音**
- 請將手機設定為靜音或震動
- 簡報會在結束後會盡快發佈
- 感謝您的配合與諒解

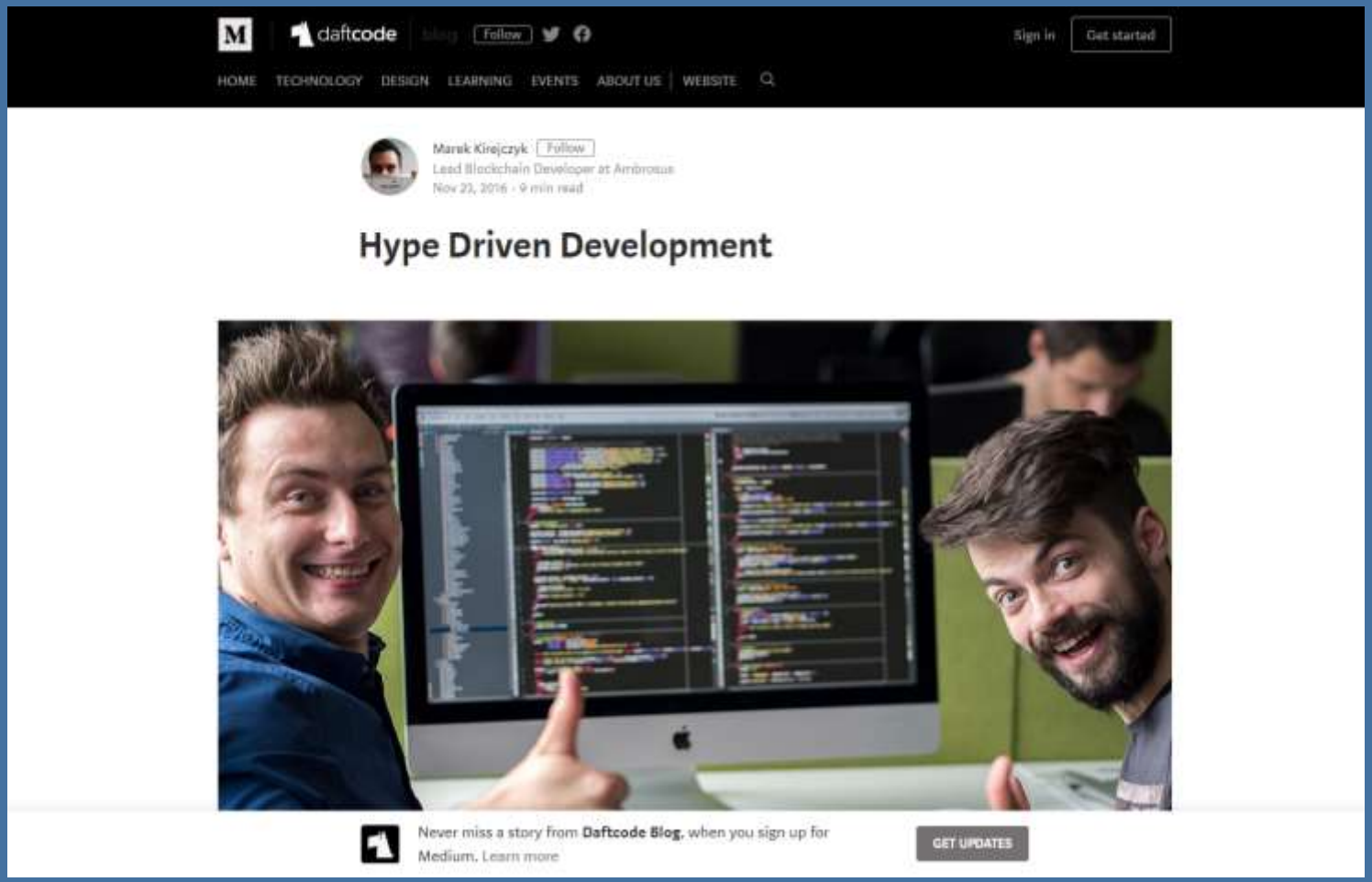


什麼是 HDD？

Hype Driven Development



Hype Driven Development



https://blog.daftcode.pl/hype-driven-development-3469fc2e9b22

中文翻譯「一窩蜂驅動開發」



https://blog.chunfuchao.com/?p=656

新技術彷彿就是那道光...

- 很多人喜歡關注新技術，也喜歡追新技術
  也勇於在任何的專案裡直接導入新技術
  不管這技術是否成熟或完整
- 網路上大家都在討論什麼就緊跟，
  反正用下去就對了，用下去就會一樣潮！
  不跟著用就落伍啦

然而...

- 新技術不見得適合應用在公司專案裡

  因為資源、文件、官方支援都還少之又少

- 既有的系統結構已經零零散散並且存在很多問題

  又再把新技術往上堆疊

- 要導入但是又無法掌握新技術

  也沒有人可以有那個時間投入研究

最後

- 原本已經用新技術寫好的部分，只能砍掉重練
- 開始鄙視這項新技術，到處說它的壞話
- 又看到更夯的新技術，把原本的翻掉，再導入再一次進入循環…

許多人都樂於當勇士
但更多時候都變成了烈士

什麼是重構？

重構

在不改變軟體外部行為的前提下
改變其內部結構
使其更容易理解且易於修改

Improving the Design of Existing Code
Martin Fowler

重構

- 小一點的情境：方法或類別的改寫、調整
- 大一點的情境：多個類別或類別庫的改寫、調整
- 更大一點的情境：整個系統的改寫、調整

然而...

- 很多人以為的重構只是單純的程式碼重寫

  但最後寫出來是對還是錯，連自己都不知道

- 更可怕的是，一邊改寫又一邊添加功能

  但往往最後都改成四不像

最後...

- 改寫後的程式比原本的還更糟糕
- 改變了對外部的行為與破壞了原有的邏輯

  得到了無限發散的錯誤、陷入無止境的修改循環
- 當無法收拾的時候，再一次進入重構的死亡行軍

這次的任務是...

ASP.NET WebForm

重構改寫為

ASP.NET MVC

沒有所謂「一鍵轉換」或「快速轉換」

- 不要再問啦！

  沒有所謂「一鍵轉換」或「快速轉換」的方法

- ASP to ASP.NET WebForm 或

  ASP.NET WebForm to ASP.NET MVC 或

  ASP to ASP.NET MVC

  框架與結構上有很大的差異

除非...

- WebForm 有確實分離前後端，無過度使用控制項
- 後端程式有做分層架構處理
- 除了 UI 層之外

  服務層與資料存取層是有很大機會是可以沿用

比較糟糕的作法

- 將 ASP.NET WebForm 後端程式直接套用到 MVC

  「反正都是 .NET 程式嘛，先讓它可以動」

  「MVC 語法就用 Google 查、看 MSDN 就好啦」

- 生吞活剝式的改寫，結局往往都是失敗

ASP.NET WebForm 版本的網站



好房網買屋

- .NET Framework 4.5
- VB.NET
- ASP.NET WebForm
- Web Site Project(網站專案)
- ADO.NET
- Microsoft SQL Server

ASP.NET WebForm 版本的網站

- 原網站歷史的足跡大概可以追溯到七年前(2010)
- 既有的網站程式碼包含了兩個階段的內容

  所以可以看到第二代與第三代的舊程式
- 除了網站程式外，還有許多共用的 Library

網站的狀況

- 以老闆和需求單位的角度來看網站

  該有的功能都有，其實沒有什麼大問題

- 以維護和開發者的角度來看網站

  當有需要加新功能或修改錯誤時，就是個大問題

- 不算問題的問題是...

  當初的開發團隊都已經不在了，也沒有文件

效能及系統維運問題

- 過多的資料庫讀取
- 沒有適當及有效使用快取
- 沒有確實做好輸入參數驗證
- 沒有應用程式端的監控與記錄機制

比較令人頭痛的問題

- 重複的程式碼

  一般網站、行動網站 aspx ashx

- 冗長、複雜的程式碼

  商業邏輯與資料存取邏輯交錯

  網頁顯示、商業邏輯、資料存取邏輯混雜

無法掌控的問題

- 開發 WebForm 網站時就已經實行前後端分離

  前端 JS 程式與 HTML 原始碼不屬於後端管控

- 原始版本的 JS 與 HTML 沒有版控

- 重構階段的前半年都還沒有看到新版 HTML Code

重構的範圍

- 好房網買屋網站(含行動版)以 ASP.NET MVC 改寫查詢列表頁、明細頁、主題屋
- 整合降價地圖(WebForm)與捷運降價宅(MVC)

重構的基礎

- 採用好房網買屋 APP 後端 WebAPI 專案架構基礎
- 初期有 80 % 的程式是直接複製使用

  預想在重構完成後可做到兩邊整合
- 好房網買屋 APP 後端 WebAPI 專案的功能

  只是原本網站功能的 1/5

twMVC#21 2015-12-12



https://goo.gl/yW1fHq

分析原有網站

- 分析原有網站各功能的執行流程、路徑、方法

  使用 XMind 記錄

- 不同情境下，輸入什麼樣的參數，執行了什麼方法或判斷，使用了哪些 Library，有多少種條件路徑，執行了哪些 T-SQL，全部都要記錄

拆解舊程式

- 先找出資料存取邏輯
- 完全摒除顯示邏輯的程式，抓出商業邏輯處理
- 最後再處理顯示邏輯

專案使用框架與技術

- .NET Framework 4.6.2
- ASP.NET MVC 5.2.3 / ASP.NET WebAPI 2.2.3
- Dapper
- MSTest
- NanoProfiler, Exceptionless

相關使用套件

- Unity
- AutoMapper
- FluentValidation
- NSubstitute
- AutoFixture
- FluentAssertions
- CsvHelper
- ExpectedObjects
- MSTestHacks
- MvcRouteUnitTester
- SpecFlow

專案架構


1.Core
HouseFun.Buy.ClassLibsProxy
HouseFun.Buy.Common
HouseFun.Buy.Repository
HouseFun.Buy.Service
2.Applicaion
HouseFun.Buy.WebApplication

TFS 2017

- 程式版本控管
- 開發工作管理
- 建置
- 發佈管理

twMVC#24 2016-10-01



https://goo.gl/wju4GL

測試是開發的一部份

- 先開發 Repository 與 Service 專案程式
  使用單元測試驗證程式的正確性
- Repository 使用 LocalDB 做整合測試
- Service 使用 CSV 與 AutoFixture
  做為單元測試的測試資料來源

Repository 測試使用 LocalDB

- Repository 測試使用 LocalDB - Part.1
- Repository 測試使用 LocalDB - Part.2
- Repository 測試使用 LocalDB - Part.3
- Repository 測試使用 LocalDB - Part.4
- Repository 測試使用 LocalDB - 範例程式碼 @ Github

開發過程中比較大的挑戰

一、重寫資料查詢核心程式

二、舊網址轉換新網址

開發過程中比較大的挑戰

- **重寫資料查詢核心程式**

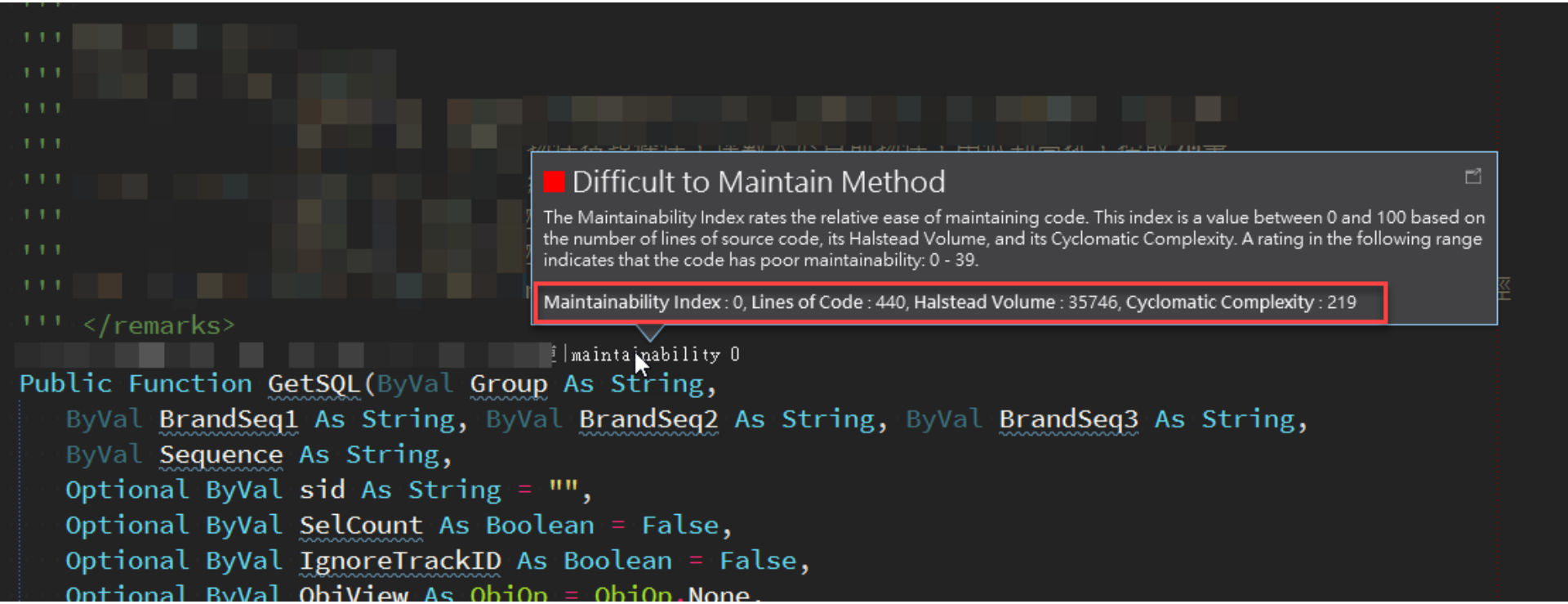
  大約耗費五週的時間，包含開發與單元測試

- **舊網址轉換新網址**

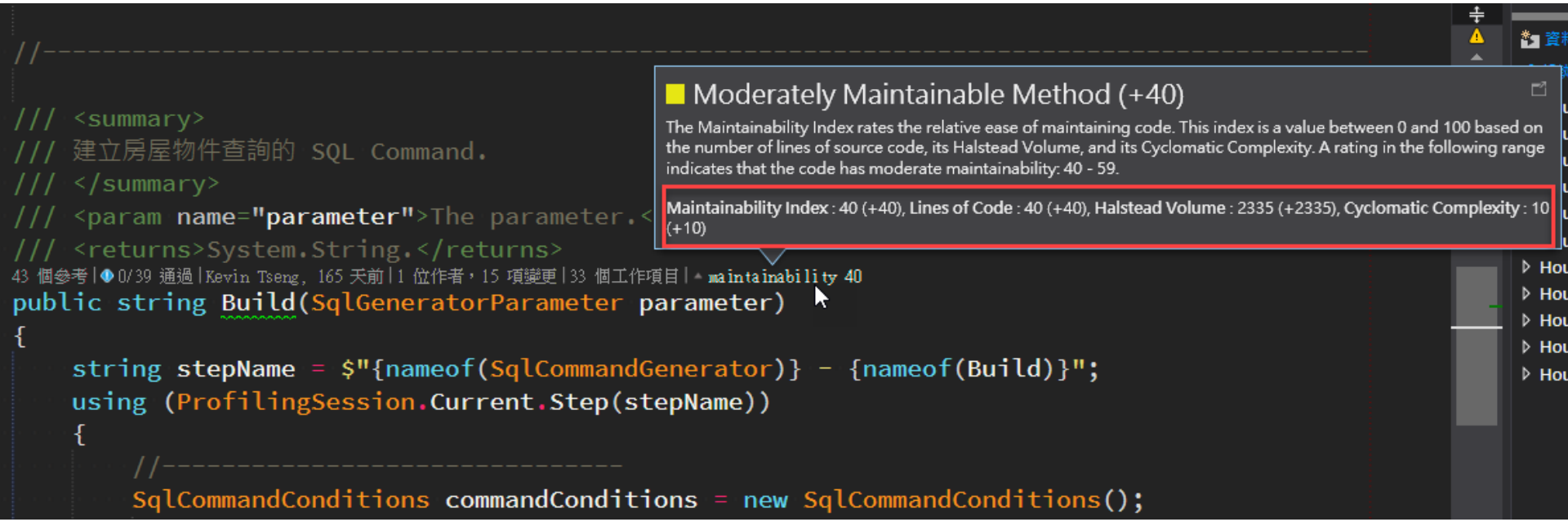
  前後大約歷經三個月時間斷斷續續的開發與測試

  最後的整合測試耗費四週

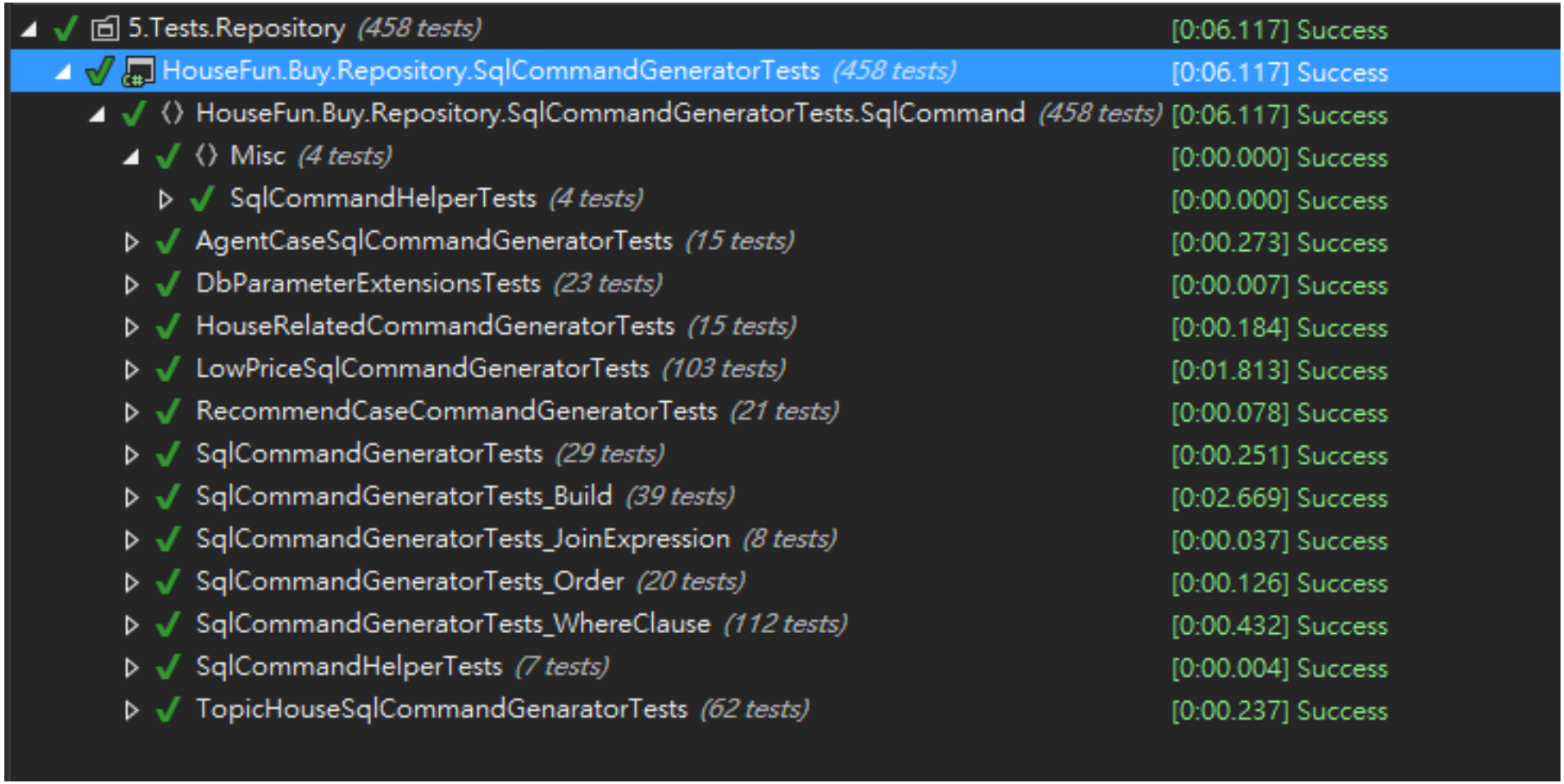
原本的資料查詢核心程式



重構之後的資料查詢核心程式



資料查詢核心程式的相關測試



舊網址轉換

- 已知的 UrlRewrite 設定有 50 個
- 而這 50 個設定裡有 12 個是變化題

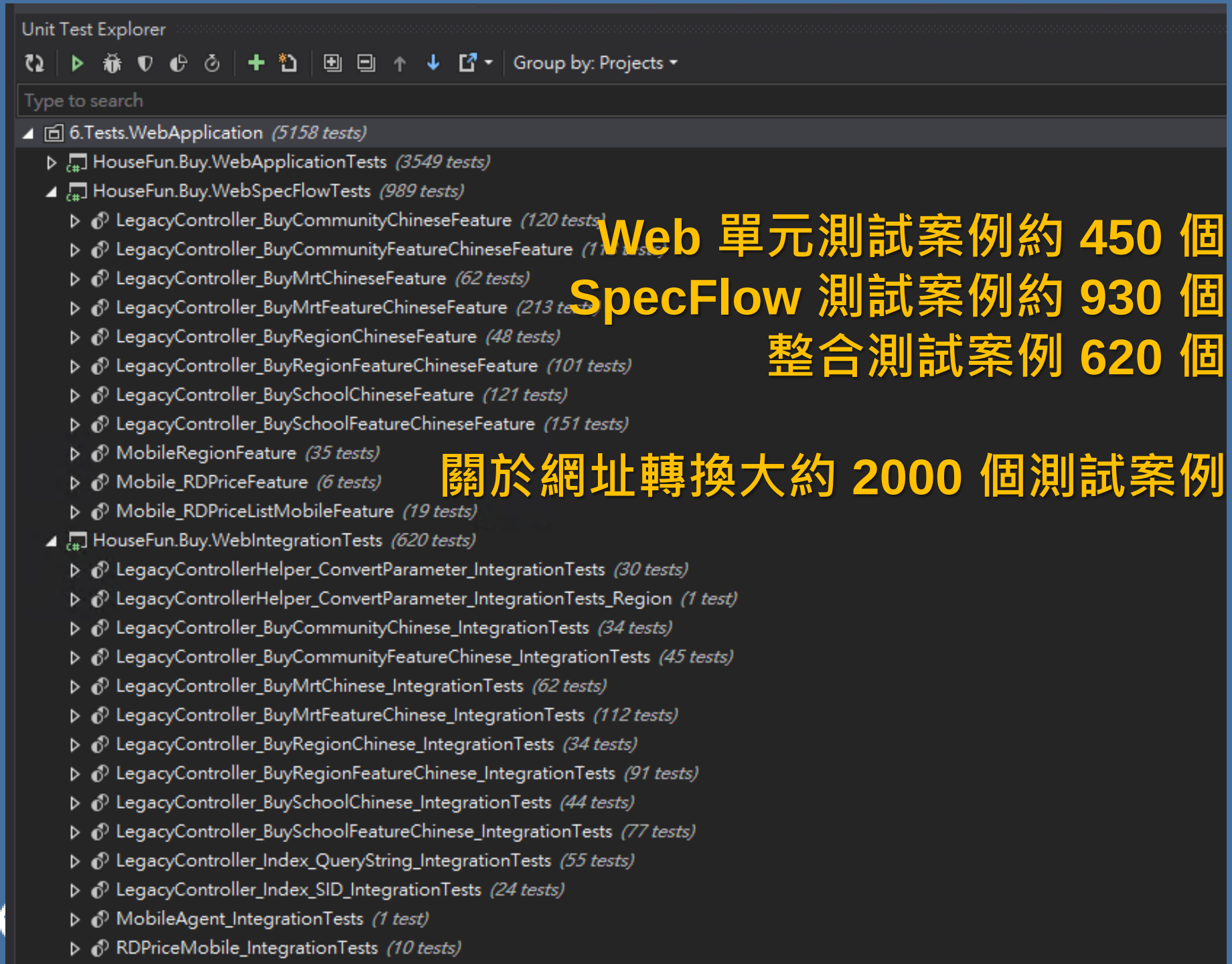
  都還可以再變化 4 種，也就是 38 + 48 種設定
- 另外還有藏在 QueryString 裡的一些隱藏規則

  輸入某神奇變數以及 1,2,3 顯示特定規則資料

  變數組合並且帶特定變數要能夠讓後台可以追蹤

轉換案例

- /買屋/台北市/降價?hd_CityID=0000&hd_Tag=1&hd_Purpose=K&hd_CaseType=L&hd_Sequence=Sequp&hd_SearchGroup=Group01&hd_PM=1&hd_Tab=1
- /region/台北市_c/降價_tag/電梯大樓_type/住宅_use/?od=Sequp



Web 單元測試案例約 450 個

SpecFlow 測試案例約 930 個

整合測試案例 620 個

關於網址轉換大約 2000 個測試案例

使用 SpecFlow Feature 寫下測試案例

```
Scenario Outline: 學校找房(特色房屋)_輸入舊網址URL_取得轉換後的新網址格式_有縣市區域名稱_無學校資料_特色房屋_降價屋_有QueryString(不含SID)
    Given 輸入縣市區域 <region>
    And 輸入學校名稱 <school>
    And 輸入特色房屋 <feature>
    And 輸入網址 <url>
    When 執行LegacyController的BuySchoolFeatureChinese方法
    Then 取得轉換後的URL路徑應包含 <UrlAddress>

    Examples:
    | SID       | region   | school | feature | url
    | 188622399 | 新北市_中和區 |        | 降價屋     | http://buy.housefun.com.tw/買屋/學校/新北市_中和區/降價屋?hd_Tag=1&hd_TGType=5&hd_Purpose=K&hd_RoomID=3&hd_Room
    | 116368935 | 桃園市_中壢區 |        | 降價屋     | http://buy.housefun.com.tw/買屋/學校/桃園市_中壢區/降價屋?hd_Tag=1&hd_Purpose=K&hd_CaseType=s&hd_priceID=4&hd_pu
    |           | 桃園市_中壢區 |        | 降價       | http://buy.housefun.com.tw/買屋/學校/桃園市_中壢區/降價?hd_CityID=0008&hd_AreaID=034&hd_Tag=1&hd_Purpose=K&hd_Cas


Scenario Outline: 學校找房(特色房屋)_輸入舊網址URL_取得轉換後的新網址格式_有縣市區域名稱_學校資料_特色房屋_降價屋_有QueryString(不含SID)
    Given 輸入縣市區域 <region>
    And 輸入學校名稱 <school>
    And 輸入特色房屋 <feature>
    And 輸入網址 <url>
    When 執行LegacyController的BuySchoolFeatureChinese方法
    Then 取得轉換後的URL路徑應包含 <UrlAddress>

    Examples:
    | SID       | region   | school             | feature | url
    | 214452634 | 台中市_大里區 | 光榮國中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台中市_大里區/光榮國中/降價屋?hd_Tag=1&hd_TGSN=127602&hd_TGTy
    | 214452617 | 高雄市_前鎮區 | 瑞豐國中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/高雄市_前鎮區/瑞豐國中/降價屋?hd_Tag=1&hd_TGSN=125354&hd_TGTy
    | 214343111 | 台北市_文山區 | 政治大學               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台北市_文山區/政治大學/降價屋?hd_Tag=1&hd_TGSN=128078&hd_TGTy
    | 214351574 | 台中市_西屯區 | 西苑高中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台中市_西屯區/西苑高中/降價屋?hd_Tag=1&hd_TGSN=127775&hd_TGTy
    | 213737547 | 高雄市_鳳山區 | 鎮北國小               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/高雄市_鳳山區/鎮北國小/降價屋?hd_Tag=1&hd_TGSN=125151&hd_TGTy
    | 213738384 | 台北市_松山區 | 敦化國中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台北市_松山區/敦化國中/降價屋?hd_Tag=1&hd_TGSN=127999&hd_TGTy
    | 213715450 | 台南市_歸仁區 | 成功大學歸仁分校航空太空科技研究中心 | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台南市_歸仁區/成功大學歸仁分校航空太空科技研究中心/降
    | 191626746 | 台中市_西屯區 | 漢口國中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台中市_西屯區/漢口國中/降價屋?hd_Tag=1&hd_TGSN=127757&hd_TGTy
    | 191625614 | 台北市      | 大業國中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台北市/大業國中/降價屋?hd_Tag=1&hd_TGSN=127763&hd_TGType=3&hd_
    | 191144827 | 台中市_東區   | 東峰國中               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/台中市_東區/東峰國中/降價屋?hd_Tag=1&hd_TGSN=127761&hd_TGType
    | 188125934 | 高雄市_鼓山區 | 內惟國小               | 降價屋     | http://buy.housefun.com.tw/買屋/學校/高雄市_鼓山區/內惟國小/降價屋?hd_Tag=1&hd_TGSN=125338&hd_TGTy
```

Route 測試

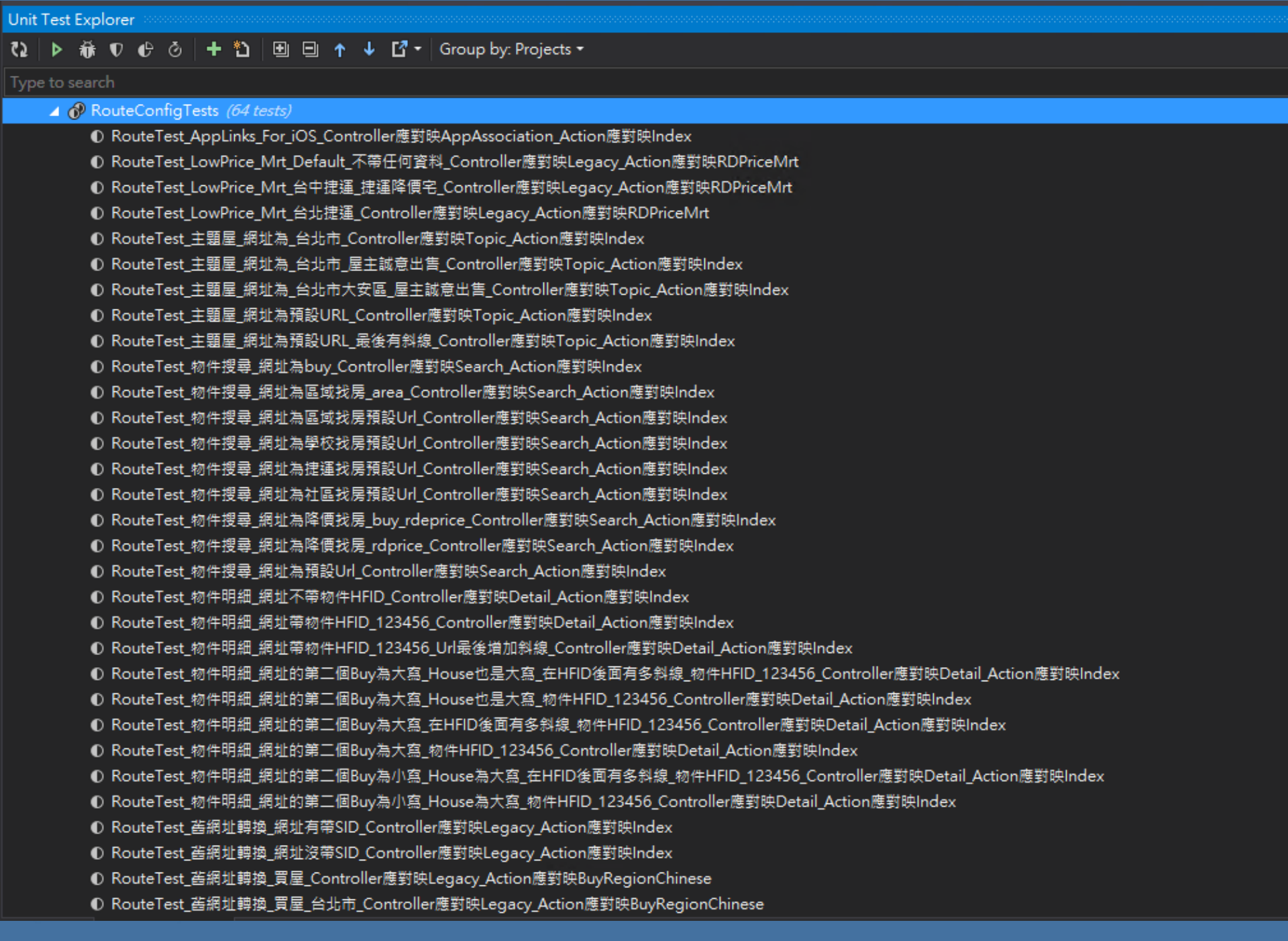
- 因為太多的網址轉換，有許多的 Route 設定

  所以有必要做 Route 的測試

- 使用 MvcRouteUnitTester

  阿磊的學習手寫版

  ASP.NET MVC Route Unit Test Part 1 ~ 4


Unit Test Explorer
Group by: Projects
Type to search
RouteConfigTests (64 tests)
RouteTest_AppLinks_For_iOS_Controller應對映AppAssociation_Action應對映Index
RouteTest_LowPrice_Mrt_Default_不帶任何資料_Controller應對映Legacy_Action應對映RDPriceMrt
RouteTest_LowPrice_Mrt_台中捷運_捷運降價宅_Controller應對映Legacy_Action應對映RDPriceMrt
RouteTest_LowPrice_Mrt_台北捷運_Controller應對映Legacy_Action應對映RDPriceMrt
RouteTest_主題屋_網址為_台北市_Controller應對映Topic_Action應對映Index
RouteTest_主題屋_網址為_台北市_屋主誠意出售_Controller應對映Topic_Action應對映Index
RouteTest_主題屋_網址為_台北市大安區_屋主誠意出售_Controller應對映Topic_Action應對映Index
RouteTest_主題屋_網址為預設URL_Controller應對映Topic_Action應對映Index
RouteTest_主題屋_網址為預設URL_最後有斜線_Controller應對映Topic_Action應對映Index
RouteTest_物件搜尋_網址為buy_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為區域找房_area_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為區域找房預設Url_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為學校找房預設Url_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為捷運找房預設Url_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為社區找房預設Url_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為降價找房_buy_rdeprice_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為降價找房_rdprice_Controller應對映Search_Action應對映Index
RouteTest_物件搜尋_網址為預設Url_Controller應對映Search_Action應對映Index
RouteTest_物件明細_網址不帶物件HFID_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址帶物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址帶物件HFID_123456_Url最後增加斜線_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址的第二個Buy為大寫_House也是大寫_在HFID後面有多斜線_物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址的第二個Buy為大寫_House也是大寫_物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址的第二個Buy為大寫_在HFID後面有多斜線_物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址的第二個Buy為大寫_物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址的第二個Buy為小寫_House為大寫_在HFID後面有多斜線_物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_物件明細_網址的第二個Buy為小寫_House為大寫_物件HFID_123456_Controller應對映Detail_Action應對映Index
RouteTest_舊網址轉換_網址有帶SID_Controller應對映Legacy_Action應對映Index
RouteTest_舊網址轉換_網址沒帶SID_Controller應對映Legacy_Action應對映Index
RouteTest_舊網址轉換_買屋_Controller應對映Legacy_Action應對映BuyRegionChinese
RouteTest_舊網址轉換_買屋_台北市_Controller應對映Legacy_Action應對映BuyRegionChinese

Repository 測試使用 LocalDB 的問題

- 當團隊的 CI/CD pipeline 只有一條線時

  同時間只有一個專案做建置或發佈

  做什麼都必須排隊，這個時候不會有問題

- 但 CI/CD pipeline 有多條線時

  同時間會有多個專案做建置或發佈，問題就出現了

Repository 測試使用 LocalDB 的問題

- 當很多專案會用到相同的資料庫

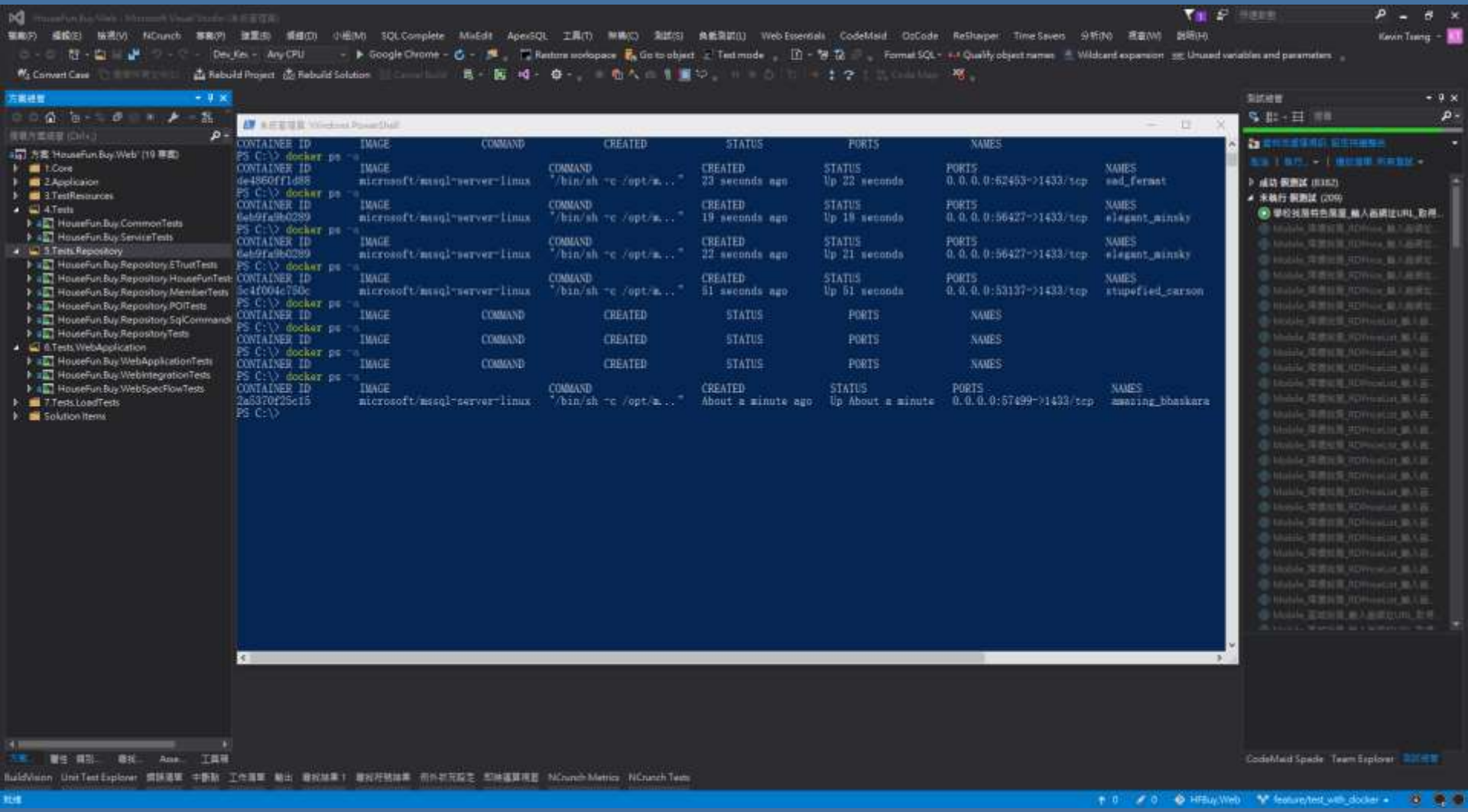
  RepositoryTests 會建立名稱一樣的 LocalDB

  就會出現衝突，容易造成專案測試失敗

- 尤其當一個測試很多的專案正在執行時

  就會排擠到其他專案的建置與發佈工作

解決方式 – 使用 Docker



```
1 reference
private static void CreateDockerContainer()
{
    string machineName = Environment.MachineName;

    // 建立測試用測試資料庫的 container
    string databaseIp;
    string containerId;

    DockerSupport.CreateContainer(machineName, out databaseIp, out containerId);

    DatabaseIp = databaseIp;
    ContainerId = containerId;

    // 於 container 裡的 sql-server 建立測試用的 database
    var connectionString = $"Data Source={DatabaseIp};Initial Catalog=master;Persist Security Info=True;Use
    DatabaseCommand.CreateDatabase(connectionString, "HOUSEFUN");
}

1 reference
private static void DeleteDockerContainer()
{
    // 移除測試資料庫
    DockerSupport.StopContainer(ContainerId);
}
```

```
var runSpacePool = RunspaceFactory.CreateRunspacePool(1, 5);
runSpacePool.Open();

using (runSpacePool)
{
    var powerShellCommands = new List<PowerShell>();
    var powerShellCommandResults = new List<IAsyncResult>();

    // 1.先停止並移除之前所建立的測試資料庫 sql-server
    var powerShellInstance1 = PowerShell.Create();
    powerShellInstance1.RunspacePool = runSpacePool;
    powerShellInstance1.AddScript($"docker stop {ContainerId}");
    powerShellCommands.Add(powerShellInstance1);
    powerShellCommandResults.Add(powerShellInstance1.BeginInvoke());

    // 2.使用 microsoft/sql-server-linux 建立測試資料庫 sql-server
    var powerShellInstance2 = PowerShell.Create();
    powerShellInstance2.RunspacePool = runSpacePool;
    powerShellInstance2.AddScript($"docker run --rm -d -e SA_PASSWORD=      -e ACCEPT_EULA=Y -ti
    powerShellCommands.Add(powerShellInstance2);
    powerShellCommandResults.Add(powerShellInstance2.BeginInvoke());

    int i = 0;
    string message = string.Empty;
```

使用 Docker 要注意的事項

- 建置環境必須要能夠支援容器
- 要在測試專案裡以程式的方式啟始 Container

  要使用 System.Management.Automation
- 執行時間不會比使用 LocalDB 來得快

使用 Docker 要注意的事項

- Windows、Linux 的 SQL Server Container

  兩者在使用上會有一些差異

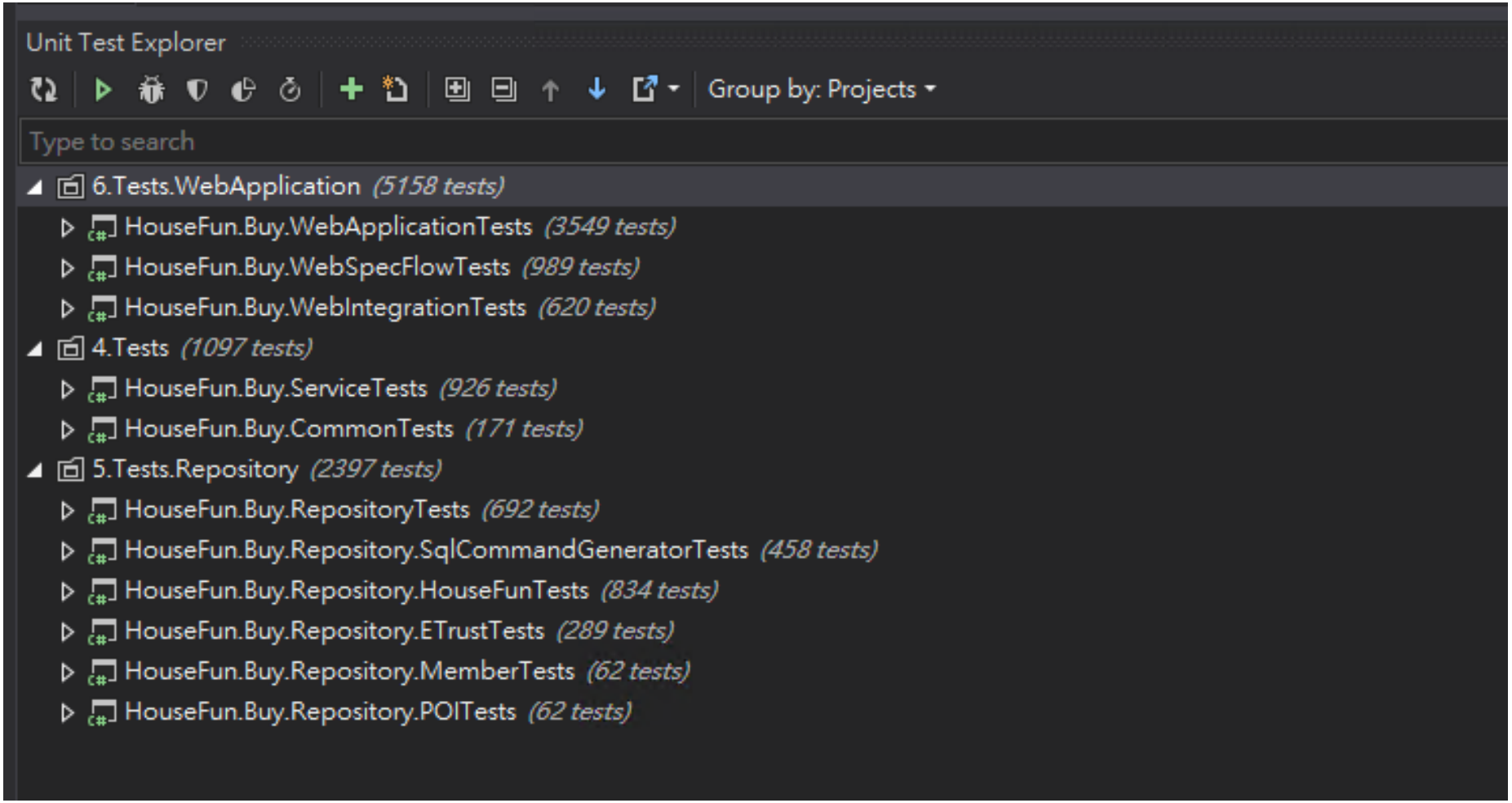
  如：IP 位置、服務啟始時間

一個依賴於自動化測試的團隊
自動化測試就是開發團隊的心跳
團隊需要一絲不苟的關心和注意
去維持它的健康

The automated tests are the heartbeat of the team that relies on them, and they need meticulous care and attention to keep them healthy.

- from The Cucumber Book

專案所有的測試


Unit Test Explorer
Group by: Projects
Type to search
6.Tests.WebApplication (5158 tests)
HouseFun.Buy.WebApplicationTests (3549 tests)
HouseFun.Buy.WebSpecFlowTests (989 tests)
HouseFun.Buy.WebIntegrationTests (620 tests)
4.Tests (1097 tests)
HouseFun.Buy.ServiceTests (926 tests)
HouseFun.Buy.CommonTests (171 tests)
5.Tests.Repository (2397 tests)
HouseFun.Buy.RepositoryTests (692 tests)
HouseFun.Buy.Repository.SqlCommandGeneratorTests (458 tests)
HouseFun.Buy.Repository.HouseFunTests (834 tests)
HouseFun.Buy.Repository.ETrustTests (289 tests)
HouseFun.Buy.Repository.MemberTests (62 tests)
HouseFun.Buy.Repository.POITests (62 tests)

專案測試數量的變化

- 2016-09-18

  637 個測試案例，測試執行時間 1 分鐘 7 秒

  覆蓋率 35.69 %

- 2016-11-14

  1572 個測試案例，測試執行時間 2 分鐘 16 秒

  覆蓋率 57.59 %

專案測試數量的變化

- 2016-12-09

  2152 個測試案例，測試執行時間 7 分鐘 50 秒

  覆蓋率 61.51 %

- 2017-01-13

  2866 個測試案例，測試執行時間 5 分鐘 34 秒

  覆蓋率 67.66 %

專案測試數量的變化

- 2017-01-26

  3038 個測試案例，測試執行時間 5 分鐘 56 秒

  覆蓋率 68.83 %

- 2017-03-20

  4321 個測試案例，測試執行時間 8 分鐘 18 秒

  覆蓋率 73.89 %

專案測試數量的變化

- 2017-05-10

  5097 個測試案例，測試執行時間 9 分鐘 55 秒

  覆蓋率 71.84 %

- 2017-06-13

  6052 個測試案例，測試執行時間 11 分鐘 14 秒

  覆蓋率 72.96 %

專案測試數量的變化

- 2017-07-03

  7004 個測試案例，測試執行時間 14 分鐘 55 秒

  覆蓋率 74.24 %

- 2016-07-07

  7900 個測試案例，測試執行時間 22 分鐘 51 秒

  覆蓋率 74.86 %

專案測試數量的變化

- 2017-07-19

  8115 個測試案例，測試執行時間 30 分鐘 53 秒

  覆蓋率 74.19 %

- 2017-09-05

  8505 個測試案例，測試執行時間 22 分鐘 18 秒

  覆蓋率 83.67 %

專案測試數量的變化

- 2017-11-02

  8872 個測試案例，測試執行時間 23 分鐘 57 秒

  覆蓋率 85.08 %

注：
前面有些測試案例比前面多，但執行時間卻反而比較少
這是因為 Build 主機更換了比較快的硬體而造成的

測試案例數量過多

- 組建時間過長，會影響其他專案
- 專案組建時間過長，影響開發與整合節奏
- 影響上線發佈的時間

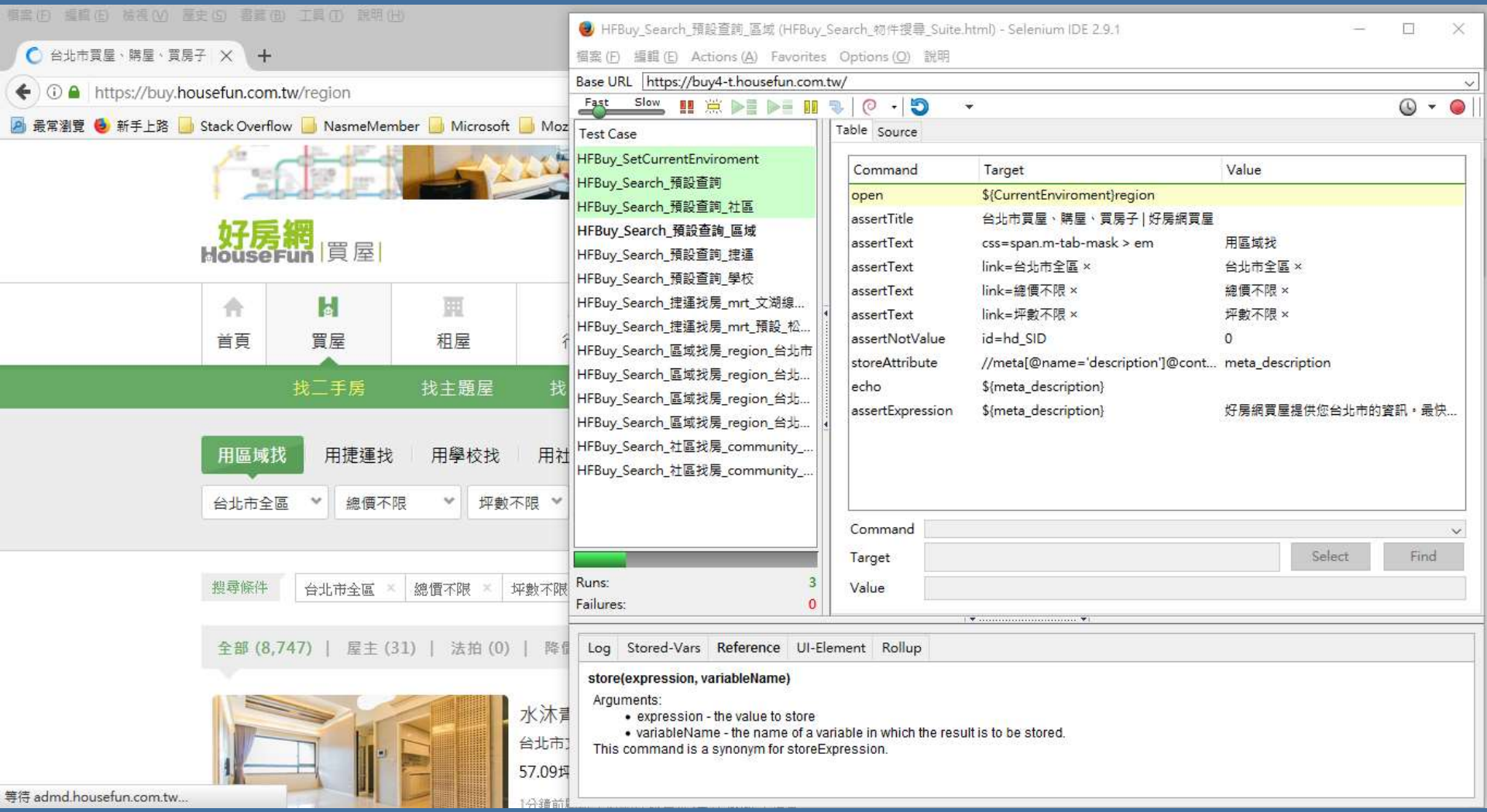
解決方式

- 分出哪些範圍是每次組建時必要執行的
- 將整合測試改為排程定時執行
- 定出那些測試是在上線前為必要執行的

等等...
好像都沒有提到 Web UI 測試？

關於 Web UI Test

- 有使用 Selenium IDE 錄製 Test Case
  但並沒有轉換為 MSTest 程式
- 前後端分離，前端變化過快，後端開發負擔也重
  無法再抽出額外的時間去處理
- 做為另一種整合測試的方式


台北市買屋、購屋、買房子
https://buy.housefun.com.tw/region
好房網 HouseFun 買屋
首頁
買屋
租屋
找二手房
找主題屋
用區域找
用捷運找
用學校找
台北市全區
總價不限
坪數不限
搜尋條件
全部 (8,747) | 屋主 (31) | 法拍 (0)
等待 admd.housefun.com.tw...
HFBuy_Search_預設查詢_區域 (HFBuy_Search_物件搜尋_Suite.html) - Selenium IDE 2.9.1
Base URL https://buy4-t.housefun.com.tw/
Fast Slow
Test Case
HFBuy_SetCurrentEnviroment
HFBuy_Search_預設查詢
HFBuy_Search_預設查詢_社區
HFBuy_Search_預設查詢_區域
HFBuy_Search_預設查詢_捷運
HFBuy_Search_預設查詢_學校
HFBuy_Search_捷運找房_mrt_文湖線...
HFBuy_Search_捷運找房_mrt_預設_松...
HFBuy_Search_區域找房_region_台北市
HFBuy_Search_區域找房_region_台北...
HFBuy_Search_區域找房_region_台北...
HFBuy_Search_區域找房_region_台北...
HFBuy_Search_社區找房_community_...
HFBuy_Search_社區找房_community_...
Runs: 3
Failures: 0
Table Source
Command Target Value
open ${CurrentEnviroment}region
assertTitle 台北市買屋、購屋、買房子 | 好房網買屋
assertText css=span.m-tab-mask > em 用區域找
assertText link=台北市全區 × 台北市全區 ×
assertText link=總價不限 × 總價不限 ×
assertText link=坪數不限 × 坪數不限 ×
assertNotValue id=hd_SID 0
storeAttribute //meta[@name='description']@cont... meta_description
echo ${meta_description}
assertExpression ${meta_description} 好房網買屋提供您台北市的資訊，最快...
Command
Target
Value
Select
Find
Log Stored-Vars Reference UI-Element Rollup
store(expression, variableName)
Arguments:
expression - the value to store
variableName - the name of a variable in which the result is to be stored.
This command is a synonym for storeExpression.

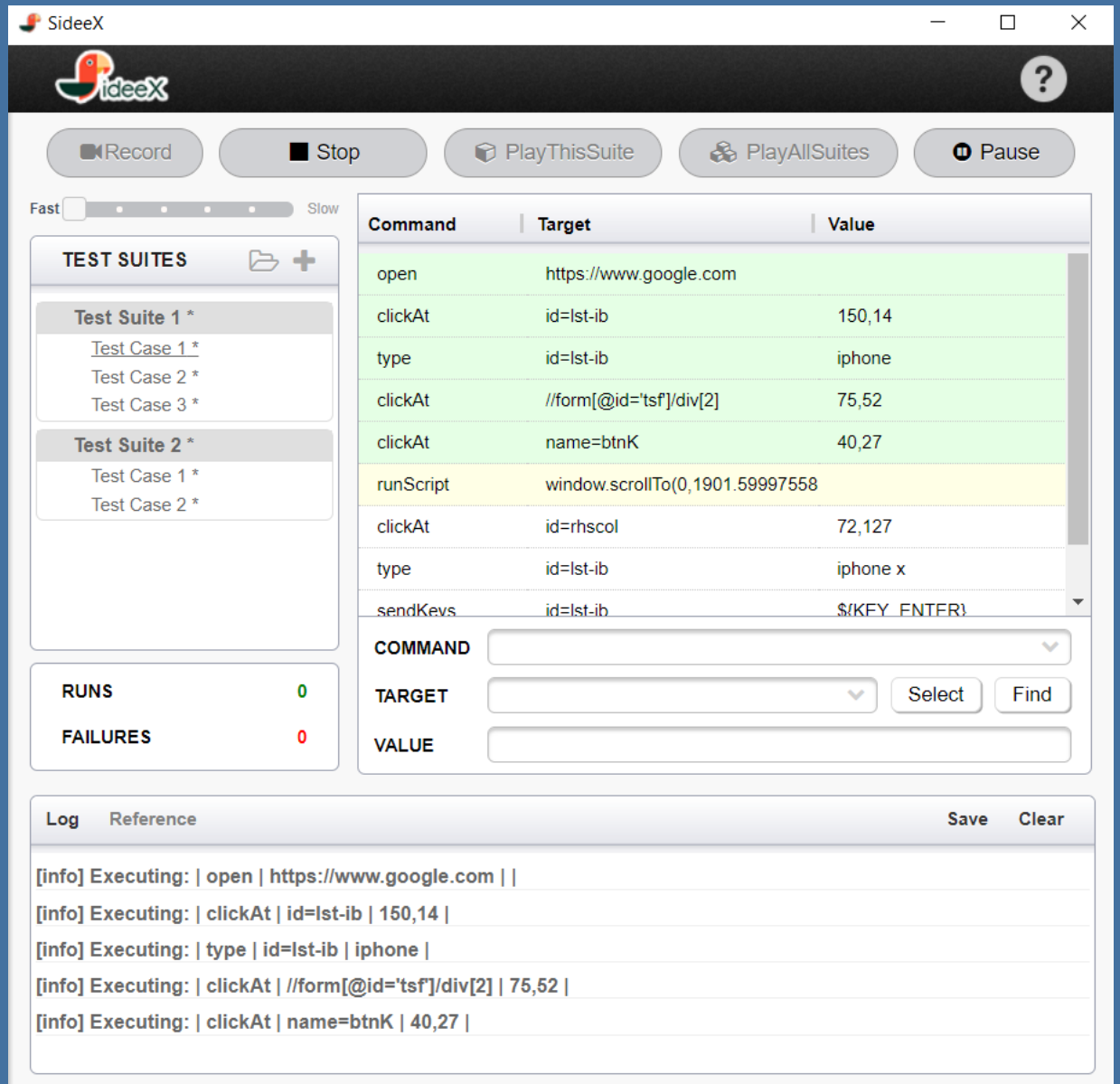
更新 Firefox Quantum 的因應

- 因為 Firefox Quantum 不再支援以往擴充套件
  原有的 Selenium IDE 無法繼續使用
- 解決的方式只有讓 Firefox 停在 54.01 版
- 另外的解決替代方案

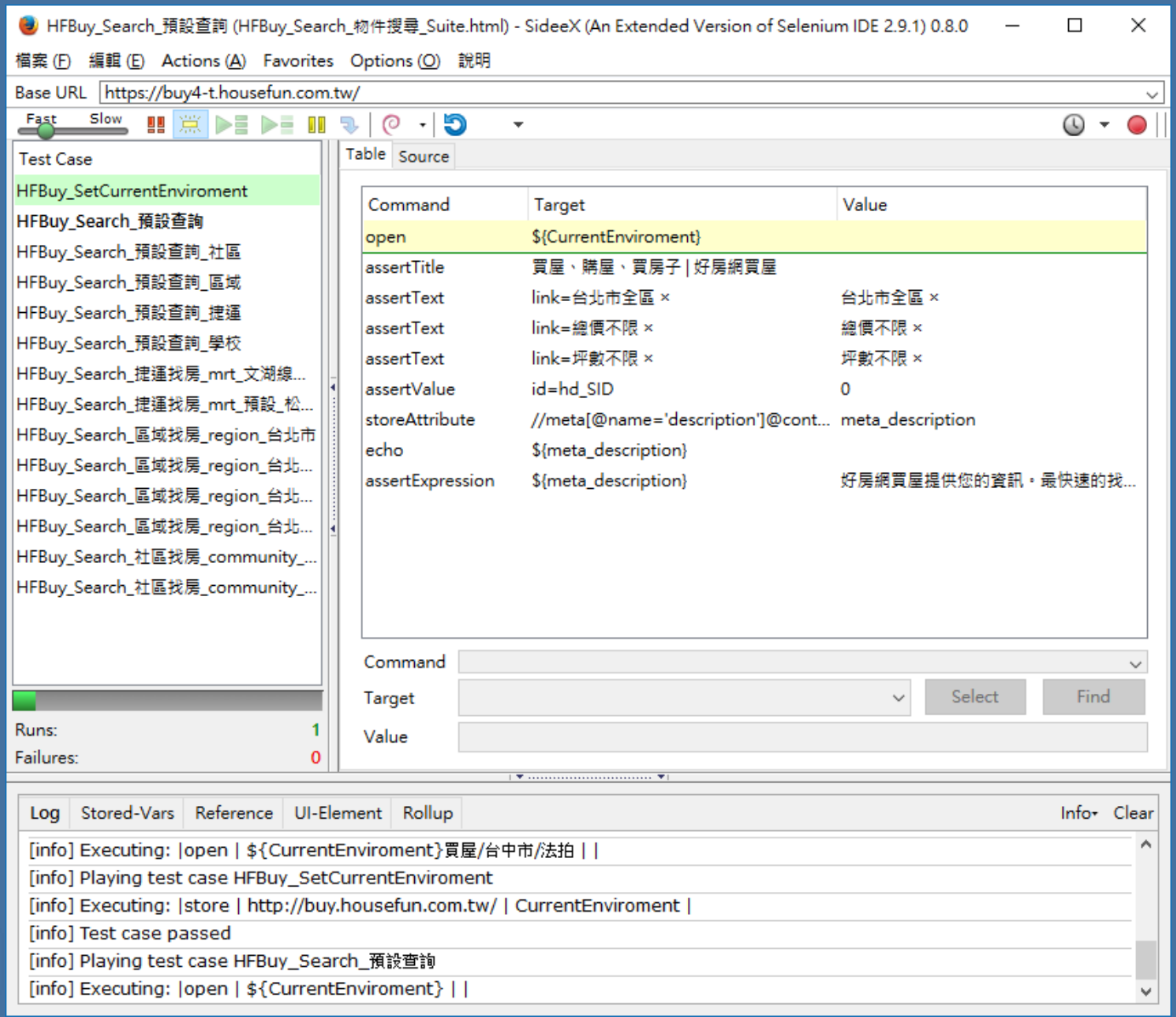
  SideeX

  Katalon Automation Recorder

SideeX Chrome/Firefox



SideeX for Firefox 33-54

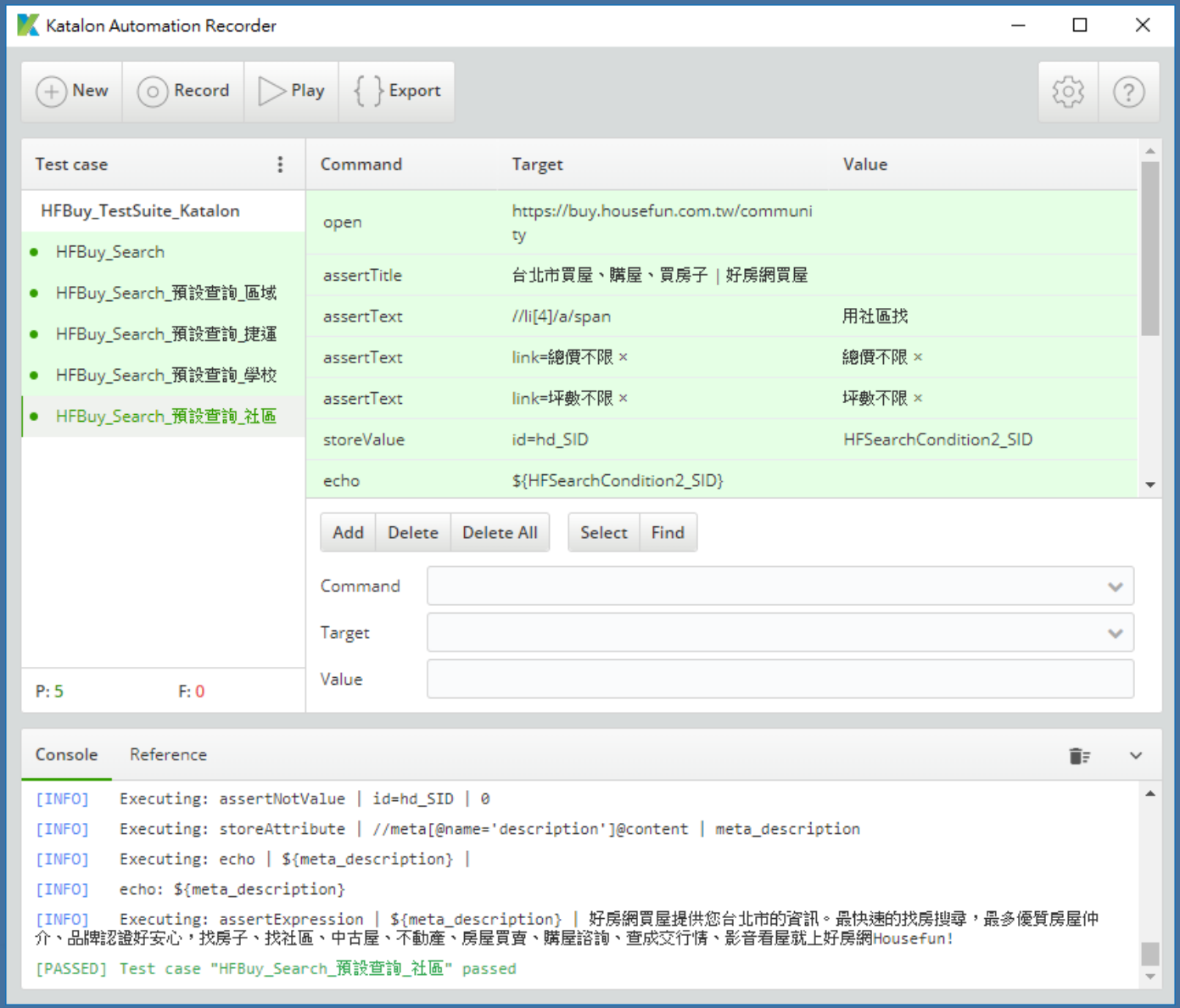


Sideex

- The Next Generation of Official Selenium IDE
- 提供 Chrome / Firefox 兩種版本的擴充功能
- 介面相近於 Selenium IDE
- Test Suite 與 Test Case 為同一個檔案
- 中文處理有些問題
- http://sideex.org/



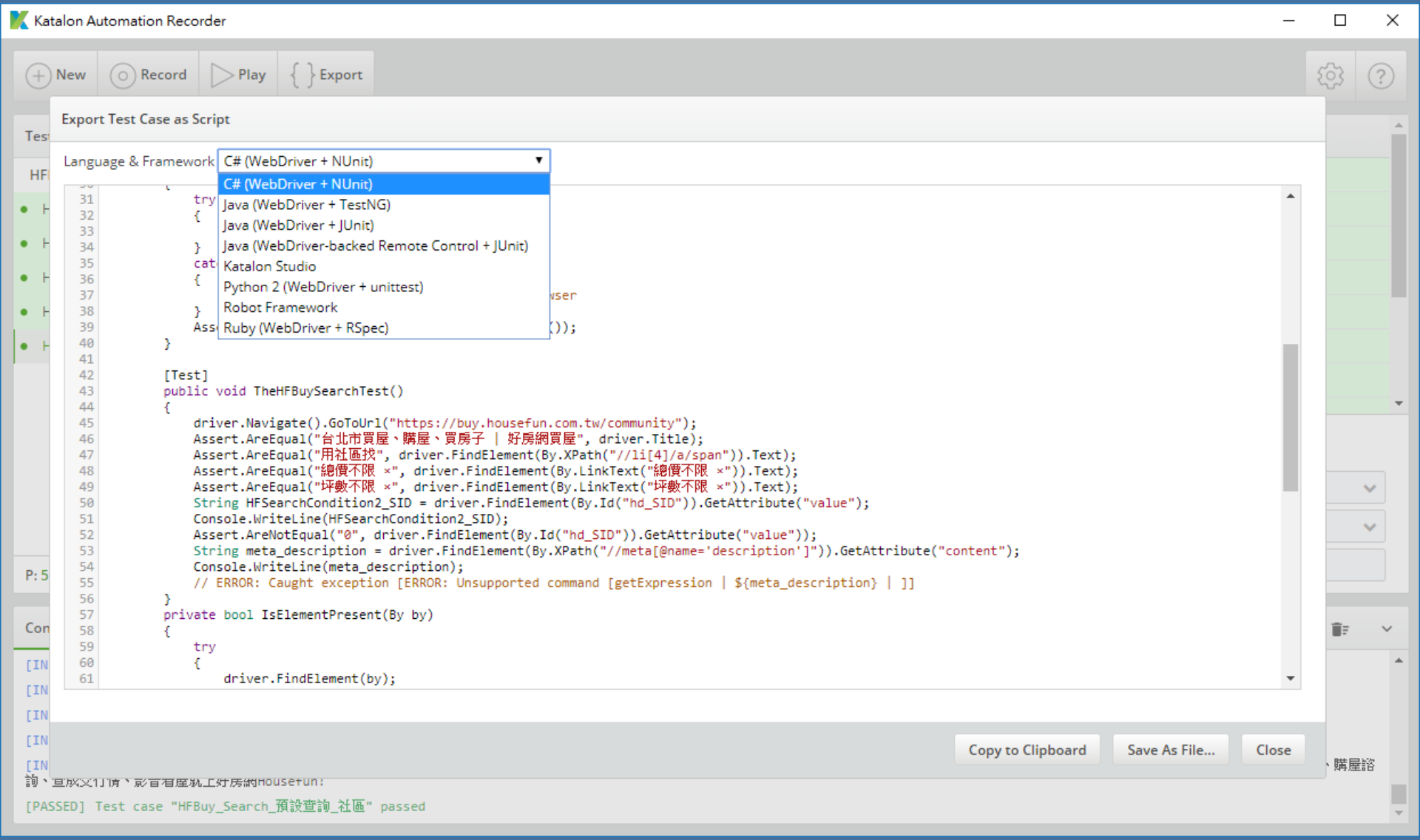
Katalon Automation Recorder



Katalon Automation Recorder

- 與 SieedX 有相似的介面，但比較陽春
- 提供 Chrome / Firefox 兩種版本的擴充功能
- 介面相近於 Selenium IDE
- Test Suite 與 Test Case 為同一個檔案
- 可以匯出 C# NUnit Script
- https://goo.gl/JhzBmQ




Katalon Automation Recorder
New
Record
Play
Export
Export Test Case as Script
Language & Framework
C# (WebDriver + NUnit)
C# (WebDriver + NUnit)
Java (WebDriver + TestNG)
Java (WebDriver + JUnit)
Java (WebDriver-backed Remote Control + JUnit)
Katalon Studio
Python 2 (WebDriver + unittest)
Robot Framework
Ruby (WebDriver + RSpec)
[Test]
public void TheHFBuySearchTest()
{
driver.Navigate().GoToUrl("https://buy.housefun.com.tw/community");
Assert.AreEqual("台北市買屋、購屋、買房子 | 好房網買屋", driver.Title);
Assert.AreEqual("用社區找", driver.FindElement(By.XPath("//li[4]/a/span")).Text);
Assert.AreEqual("總價不限 ×", driver.FindElement(By.LinkText("總價不限 ×")).Text);
Assert.AreEqual("坪數不限 ×", driver.FindElement(By.LinkText("坪數不限 ×")).Text);
String HFSearchCondition2_SID = driver.FindElement(By.Id("hd_SID")).GetAttribute("value");
Console.WriteLine(HFSearchCondition2_SID);
Assert.AreNotEqual("0", driver.FindElement(By.Id("hd_SID")).GetAttribute("value"));
String meta_description = driver.FindElement(By.XPath("//meta[@name='description']")).GetAttribute("content");
Console.WriteLine(meta_description);
// ERROR: Caught exception [ERROR: Unsupported command [getExpression | ${meta_description} |]]
}
private bool IsElementPresent(By by)
{
try
{
driver.FindElement(by);
Copy to Clipboard
Save As File...
Close
[PASSED] Test case "HFBuy_Search_預設查詢_社區" passed

但是...

- Katalon Recorder 無法自訂 Format
- 不管是 SieedX 或 Katalon Recorder 還不及於原有 Selenium IDE 的操作和功能
- 以往使用 Selenium IDE 所做的 Test Suite 無法直接使用，不過 Test Case 可以使用

Selenium IDE

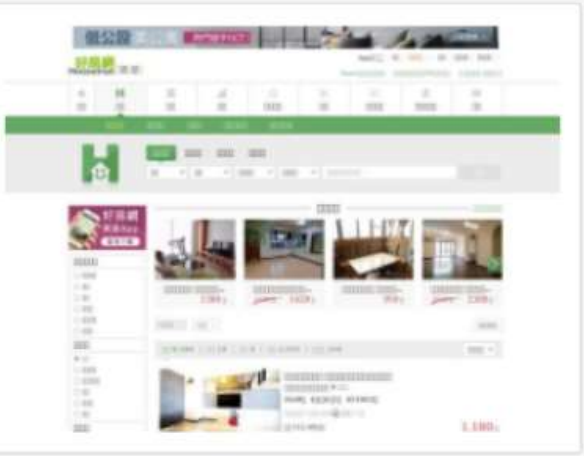
- 不管是 SieedX 或 Katalon Recorder 還正在發展中，所以可以持續關注
- 新版本的 Selenium IDE 已經在開發中 讓我們期待發佈的那一天

https://github.com/SeleniumHQ/selenium-ide

網站重構後的效益

舊版網站

Summary



Performance grade

D 68

Load time

7.43 s

Faster than

19 %

of tested sites

Page size

1.0 MB

Requests

163

Tested from

San Jose

on Sep 8 at 16:35

pingdom

新版網站

Summary



Performance grade	Load time	Faster than
C 70	4.63 s	34 % of tested sites

Page size	Requests	Tested from
1.1 MB	86	San Jose on Nov 30 at 13:40

pingdom

前端開發人員不再受限

- WebForm 受制於 Web Server Control 限制

  前端開發人員無法做任何調整

  必須透過後端程式開發人員

- MVC View 可以讓前端開發人員協同開發

  受限程度遠小於 WebForm

網站維護人員不會限縮於特定條件

- 使用 VB.NET 與 ASP.NET WebForm 的開發人員已經逐年大幅下降
- 現在會 ASP.NET WebForm 及 VB.NET 的開發者大部分都是年資比較資深的

專案架構採用分層架構

- 區分資料存取處理、商業運算邏輯、網站應用
  各層專案的程式不交錯使用
- 在維護上可以清楚區分程式的功能是處理那一塊的
  邏輯
- 減少重複程式碼、提高程式可維護性

開發過程加入單元測試、整合測試

- 讓單元測試成為開發過程的一部份

  測試案例就是專案文件

- 測試案例會依據需求及變化而增加

  讓程式品質可以有效提升，減少改東壞西的情況

- 降低錯誤的發生、減少無謂的人工檢測

  快速驗證程式的品質、提升程式的正確率

版本控管、持續整合與部署管理

- 以 Git Flow 的方式去管控，讓多人同時作業時依循共同的版控原則作業，有效管裡程式版本
- 使用 TFS 的 CI/CD （組建與發佈力）

  讓組建、測試、發佈能夠到自動化處理

  降低人工介入，減少錯誤發生

減少對資料庫的直接查詢頻率及降低負載

- 針對資料庫的存取邏輯做重新改寫

  增加快取處理，降低資料庫的直接讀取

  提升資料處理效率，加快 Request 回應速度

最後還是要說真相

專案重構的實際面

- 原本預估以三個開發人員的人力

  專案開發時間大約四個月就可以完成

- 初估過於樂觀，完全忽略舊有 Library 的影響

  沒有估算到重寫 Library 程式的時間

- 最後重構耗費了 15 個月才完成上線

增加專案維護的難度

- 因為採用分層架構，強調物件導向設計與開發
  對於熟悉的開發人員是可以很快進入狀況
  但對於新手開發人員來說，提高了進入門檻

需求單位與開發單位認知差異

- 開發單位對於重構的認知，程式改寫，提升效益
  但是對於外部使用與功能並不會改變
- 對於需求單位而言，花了成本，最後沒有改變什麼
  因為內在的改變對於使用者來說是無感的

最後...

- 我已經把專案交接出去了

  承接的開發人員要如何維護專案、如何在新的框架下繼續開發新功能而不讓專案成為另一塊棕土（Brownfield），這將是新專案維護者的課題

SkillTree 課程



https://skilltree.my/events/7mact

Blog 是記錄知識的最佳平台



https://dotblogs.com.tw

Jetbrains 重構必備工具



ReSharper

https://www.jetbrains.com/resharper/

OzCode 偵錯的魔法師



業界師資、實戰教學



https://skilltree.my

謝謝各位

https://mvc.tw

- 本投影片所包含的商標與文字皆屬原著作者所有。
- 本投影片使用的圖片皆從網路搜尋。
- 本著作係採用姓名標示-非商業性-相同方式分享 3.0 台灣授權。閱讀本授權條款，請到 http://creativecommons.org/licenses/by-nc-sa/3.0/tw/，或寫信至Creative Commons, 444 Castro Street, Suite 900, Mountain View, California, 94041, USA.