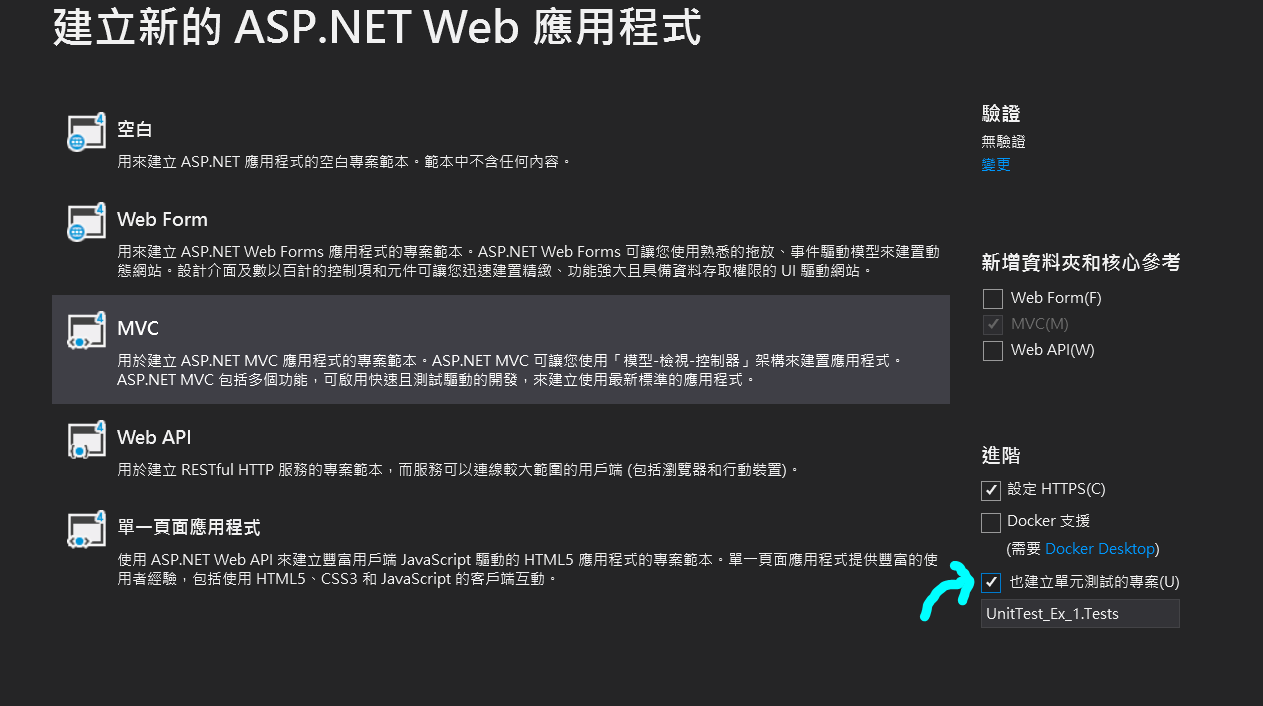
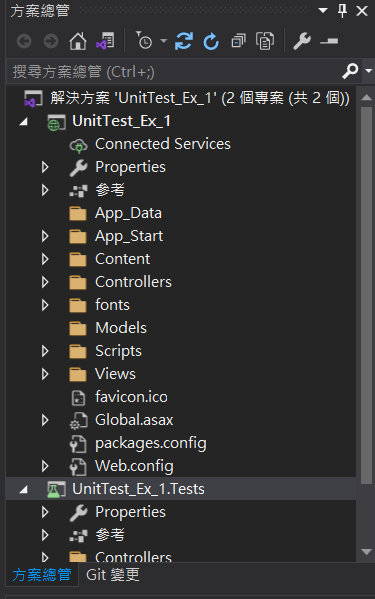


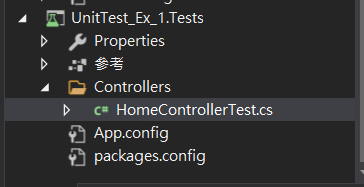
# MVC建立時一併建立UnitTest

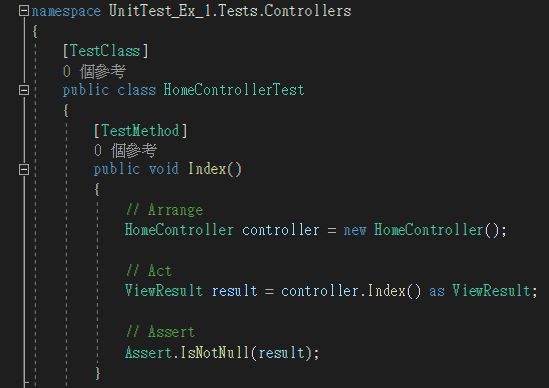




<-- 結果一開出來長這樣

一個方案，有兩個專案

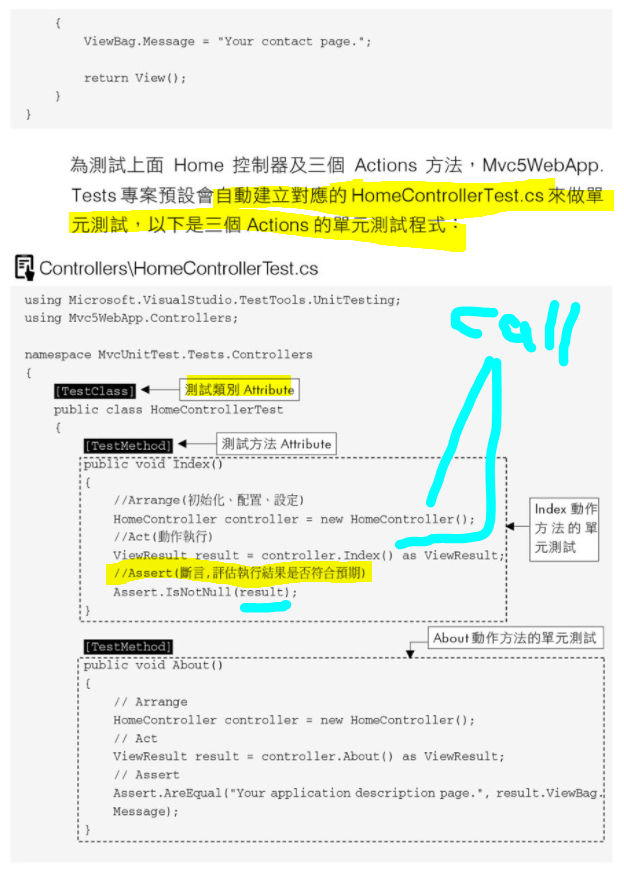
第二個專案是重點。





<- true

## 程式碼範本

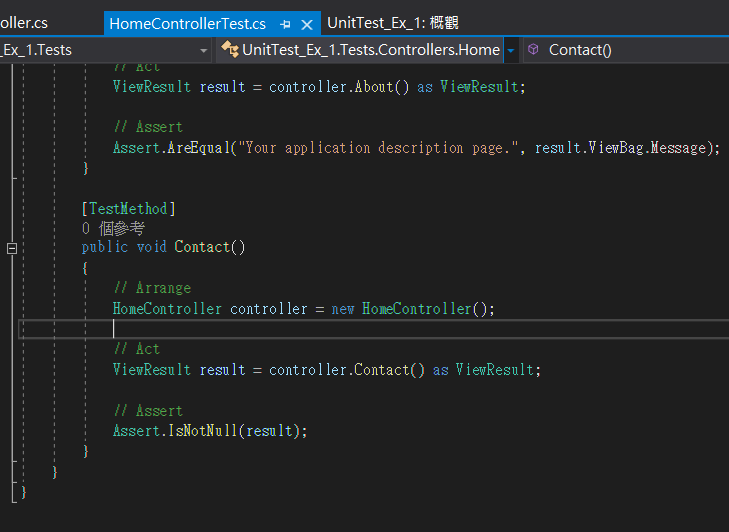


建立另一個專門的 class 和 method 來一個個去呼叫 Controller 和 Action

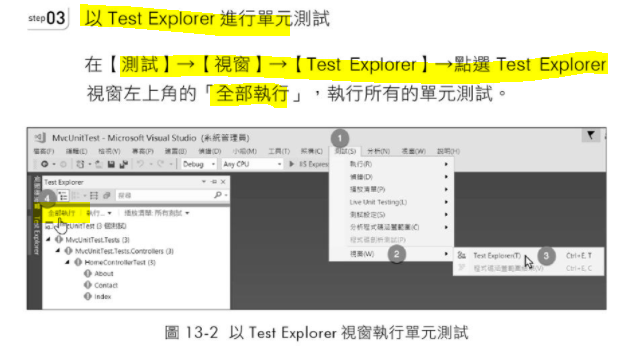
而這些專門的 method 會有 Assert 來結尾，判斷是否是我們預想的結果。

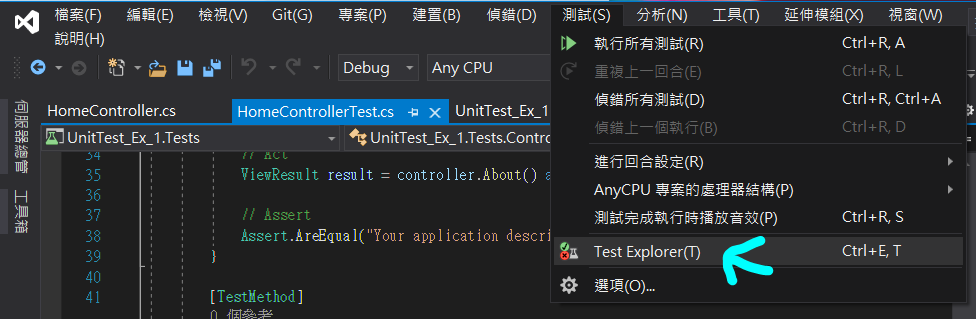
但是要怎麼去呼叫這些 method 呢？

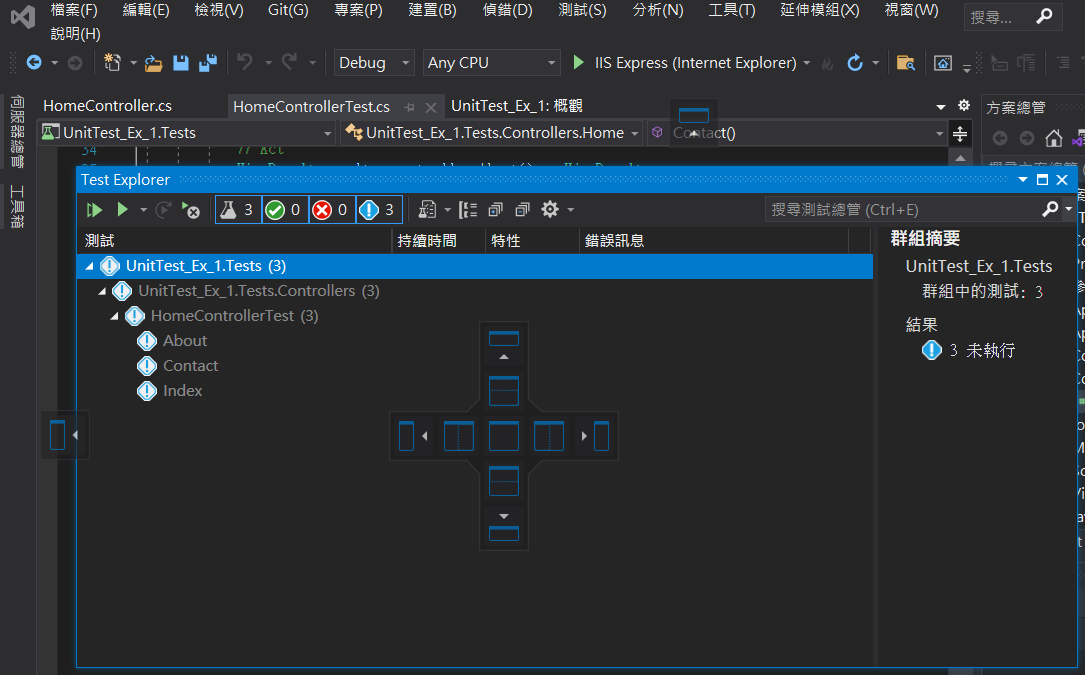


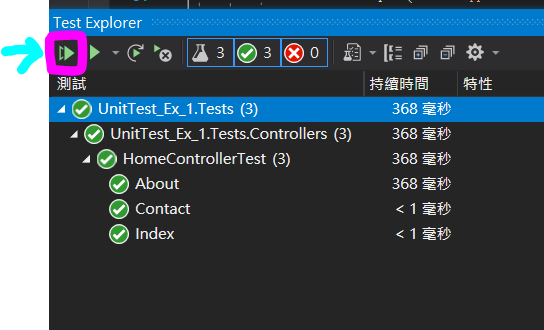
<- true

## ! 以TestExplorer進行測試 !

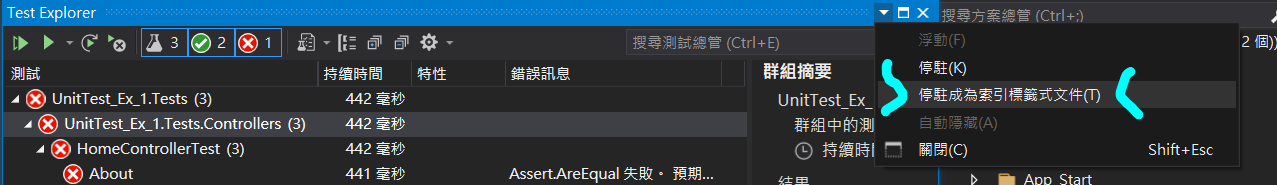




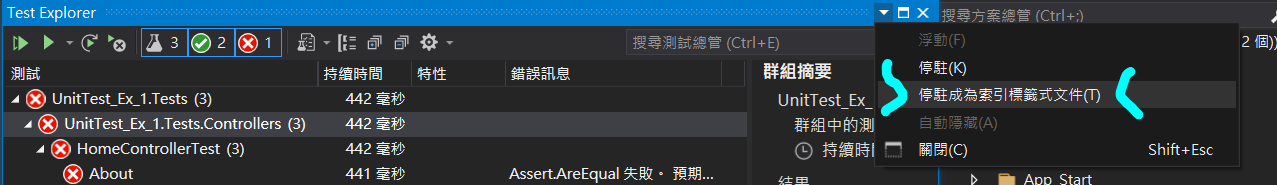
<-出現視窗



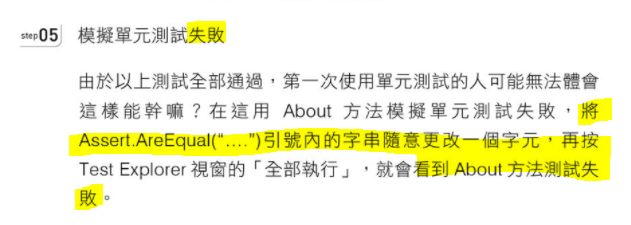
在紫色的地方按下後，就全綠了。

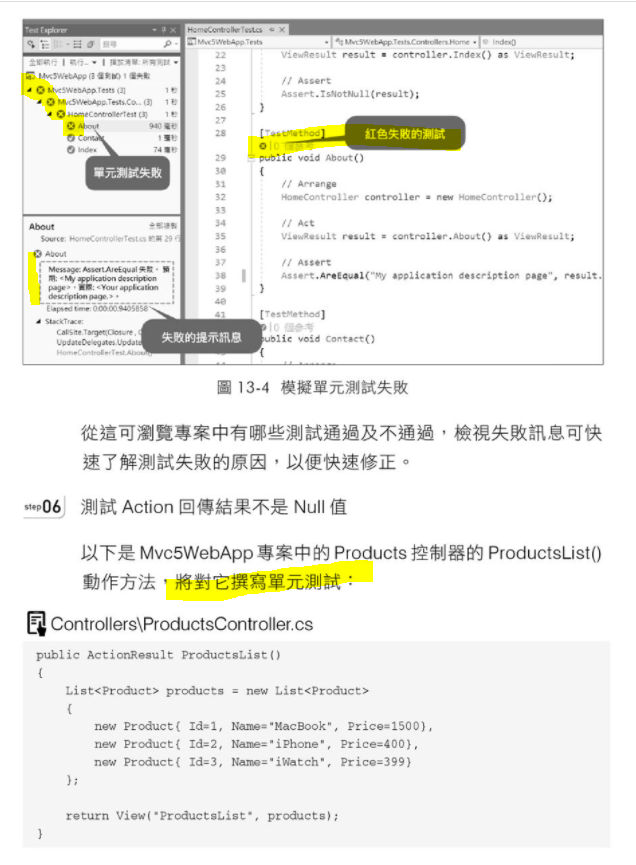


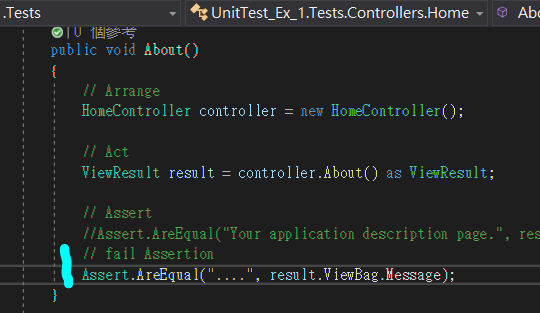
可以固定視窗

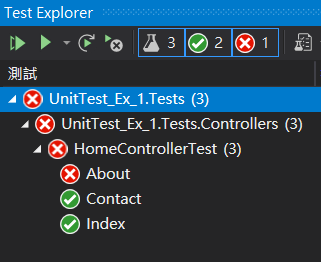


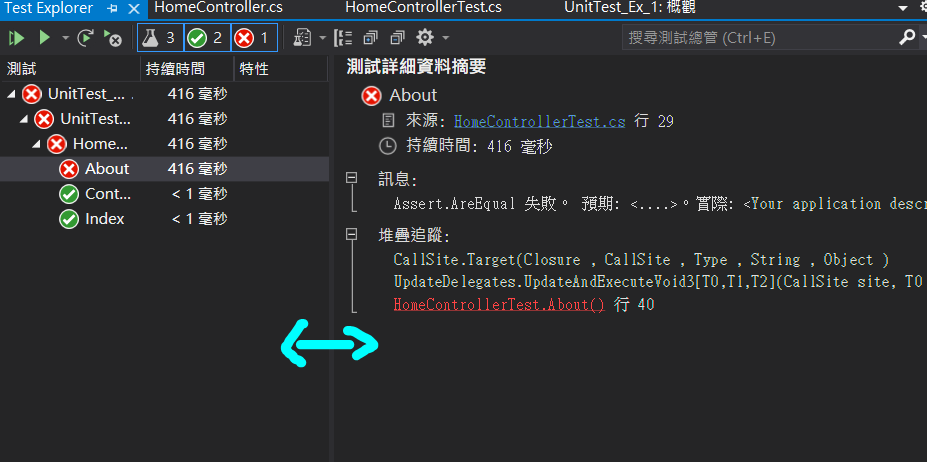
## 解讀結果

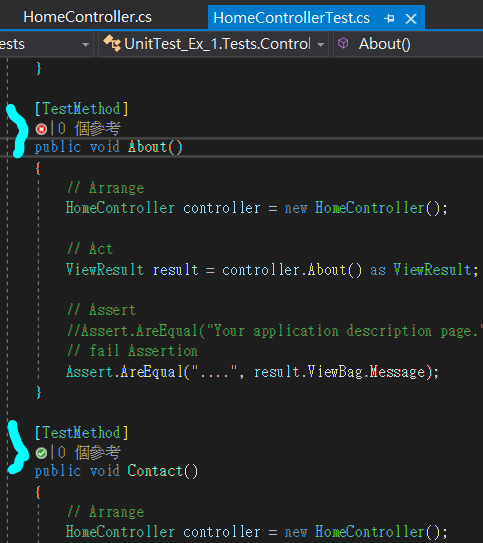


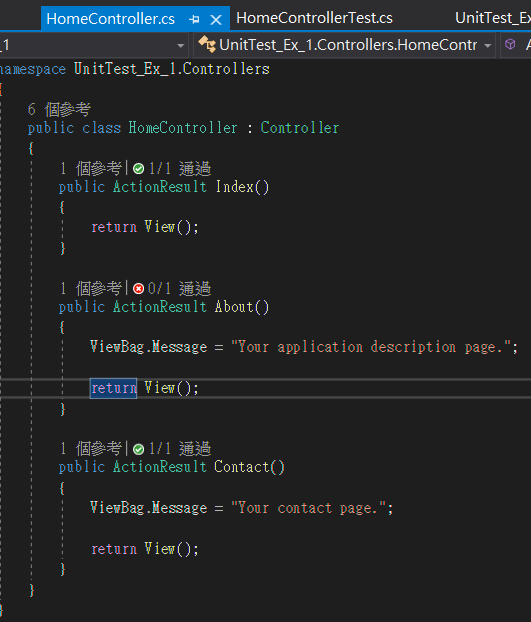








摘要要從右邊拉出來。

有這種小圖示在程式碼上。

## 對比 null 值



### Assert.IsNotNull(result)



### 

實作這個之前，先開一個新controller﹑model。在test的專案，controller

### 

Pass

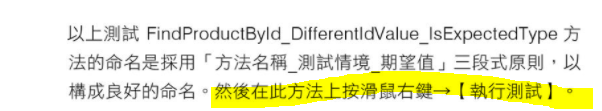
## 一次測試多種結果

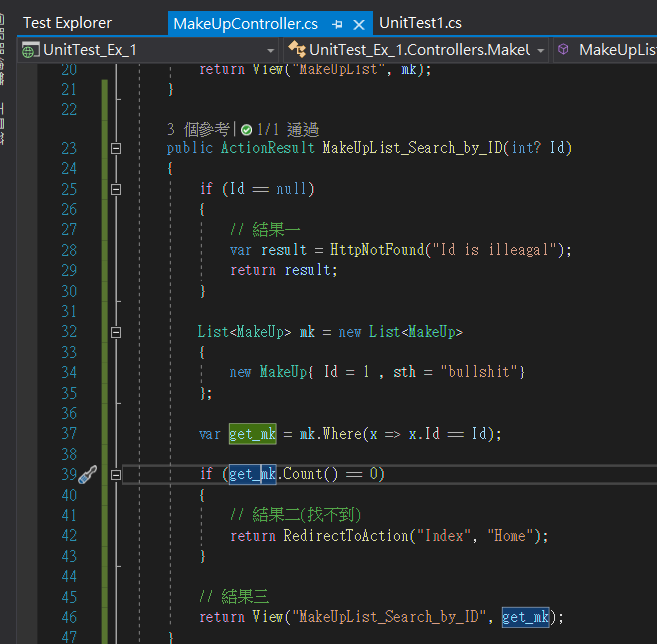


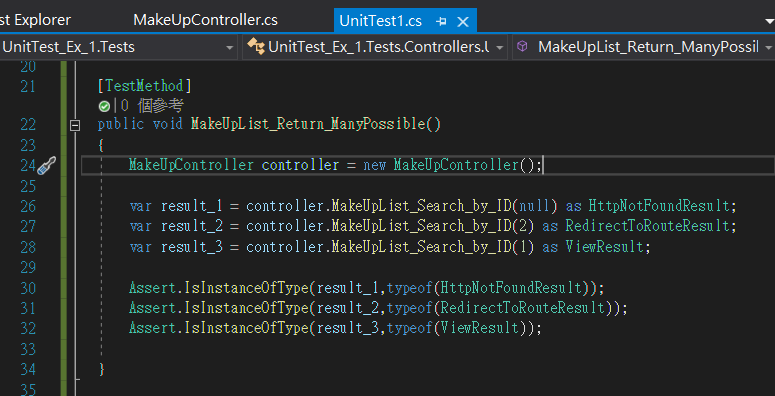


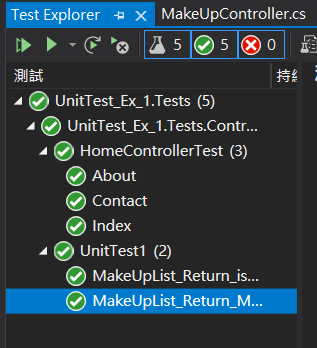


### Assert.IsInstanceofType(result, typeof( X ));

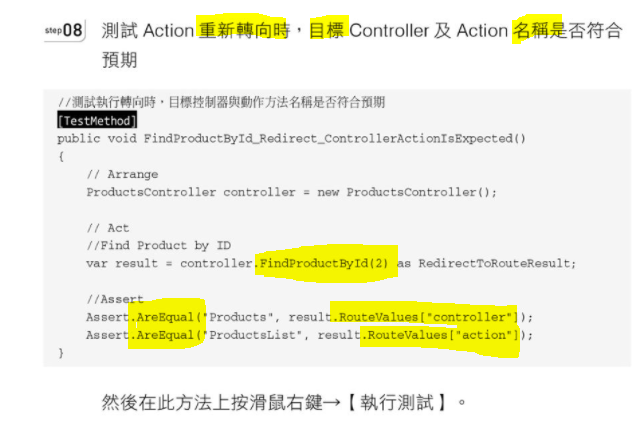








## 對比Redirect的Controller和Action的名稱



### Assert.AreEqual(“X”, result.RouteValues[“X”]);

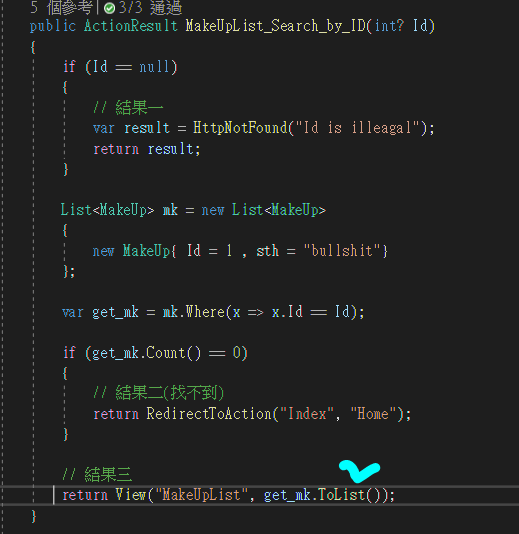


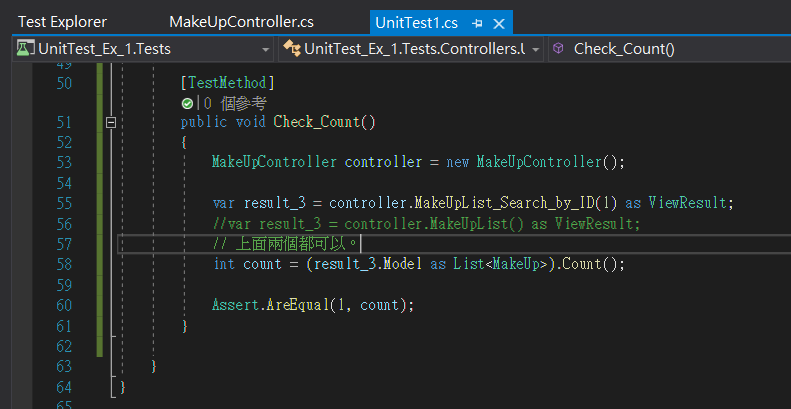


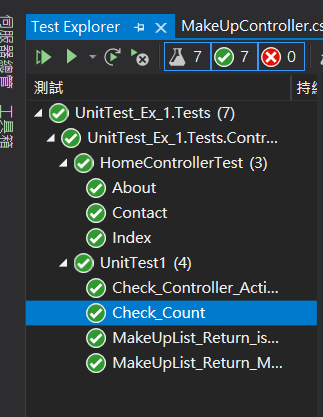
## 資料筆數是否合預期



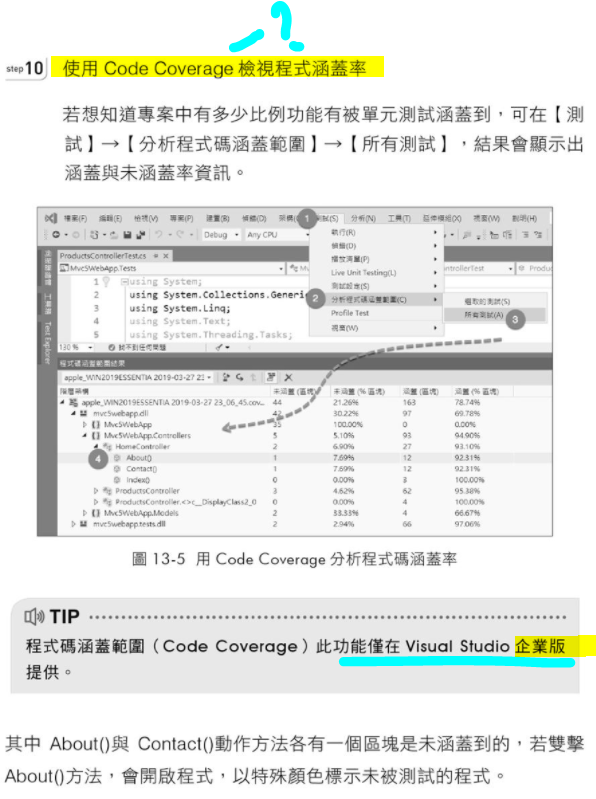
### (result.Model as List<X>).Count()





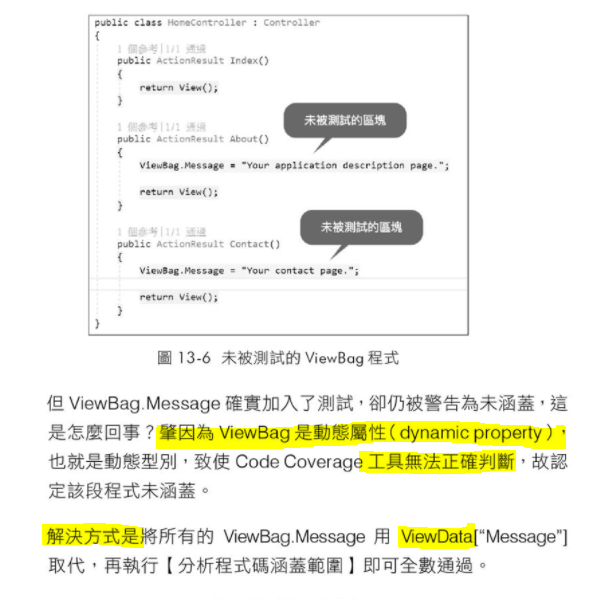


## Code Coverage

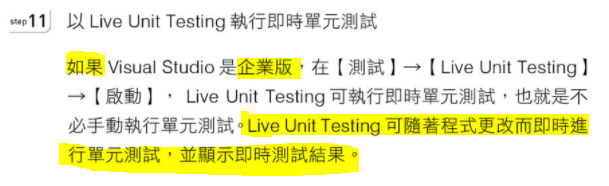




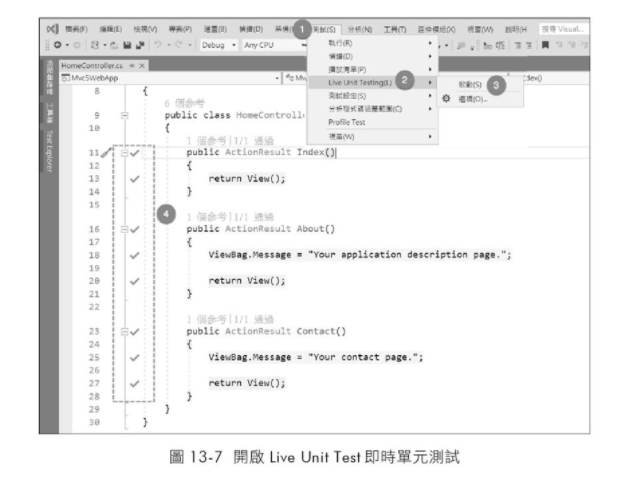
### 企業版…



關於「傳遞資料容器」的區別，我開了另一個獨立篇章來研究。

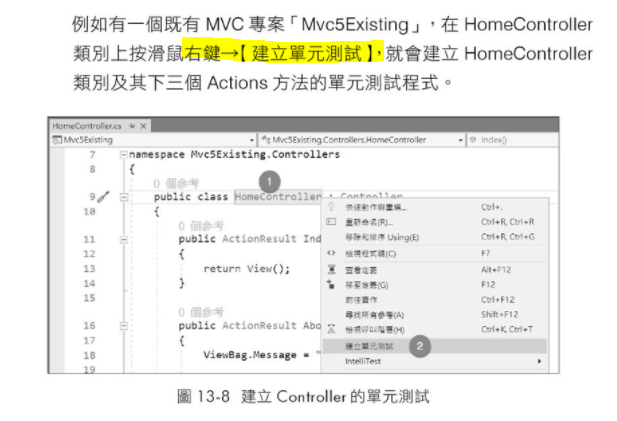


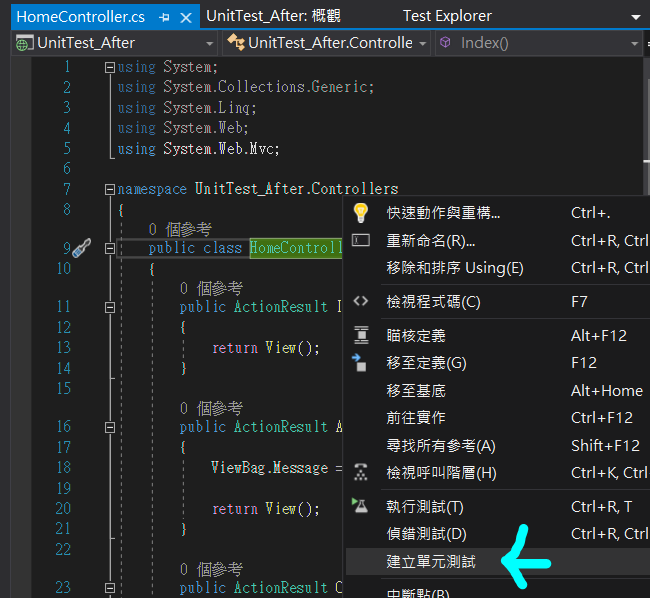
它一直替企業版打廣告真的好嗎？

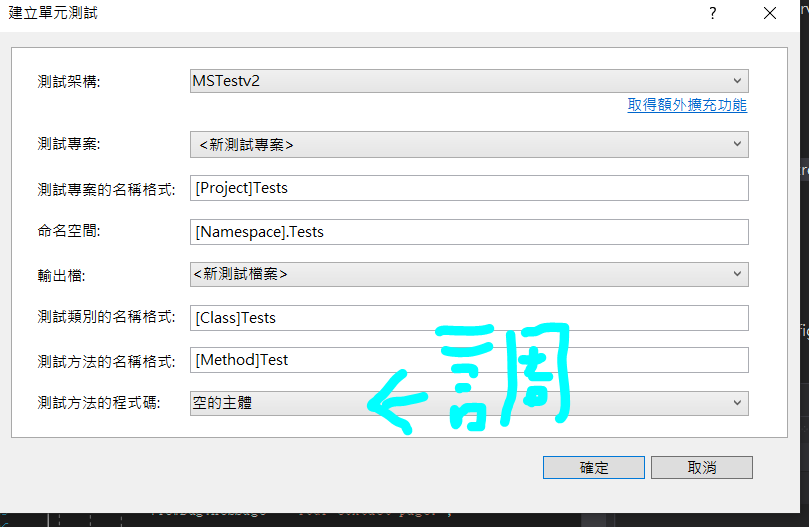


# 既有專案建立單元測試

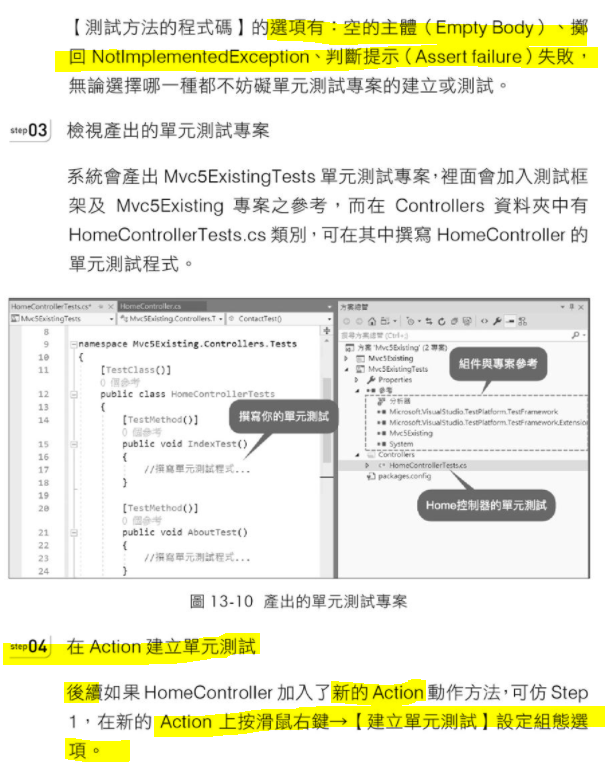
## 針對Controller

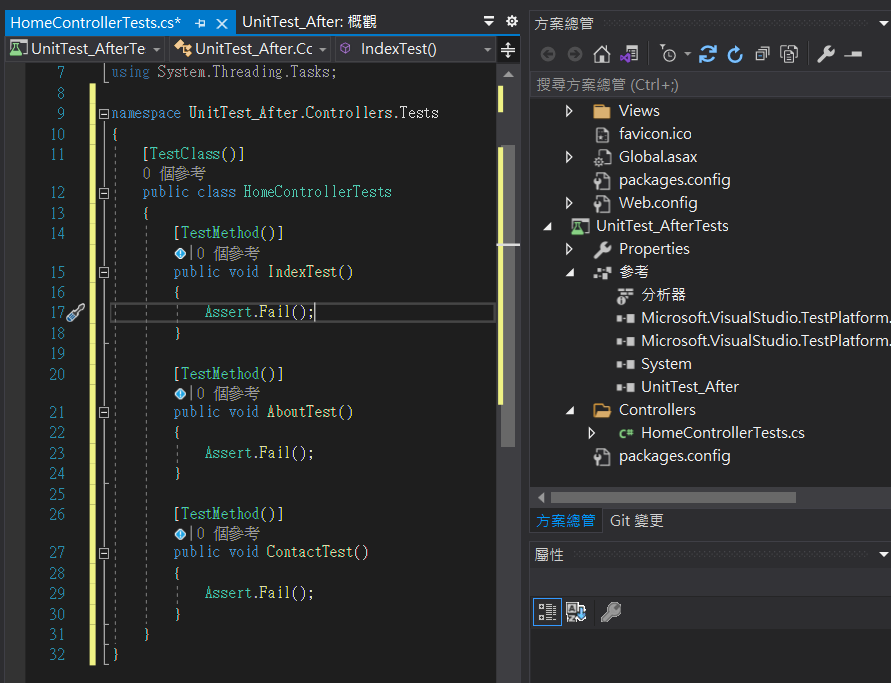






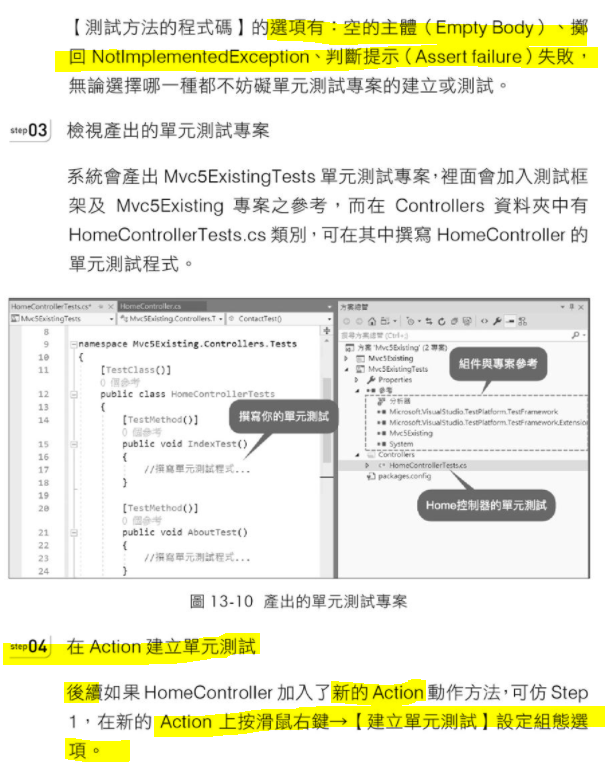
但下面說其實選什麼都沒差？沒差又為什麼要選？





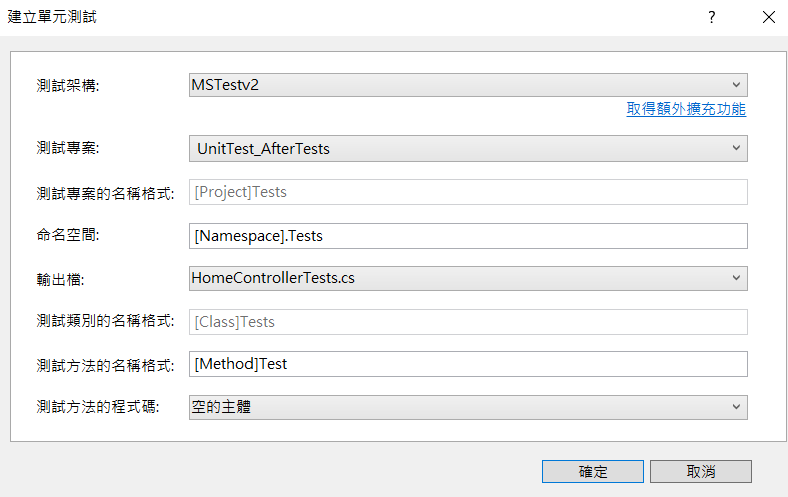
自動出現的。

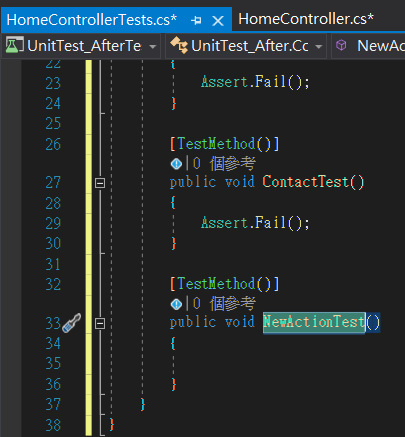
## 針對新的個別Action





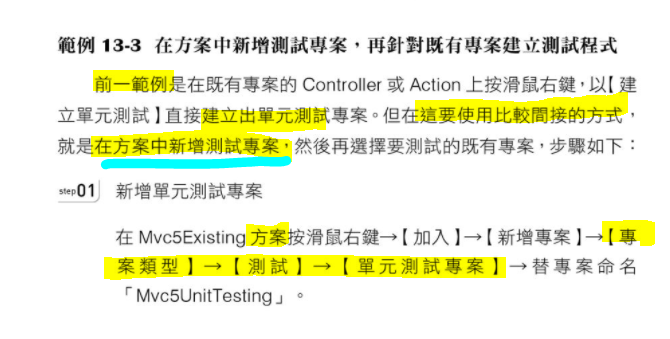




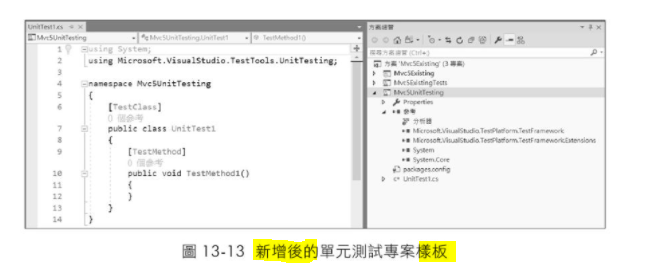


<-- 結果會自動附加出這個方法，但是內容也還是要自己寫

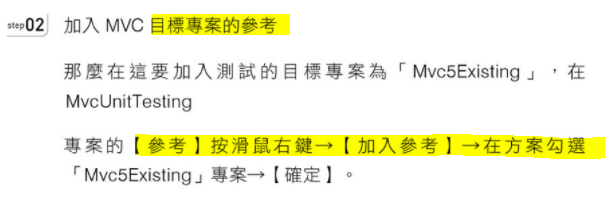
# 方案中加另一個測試專案！





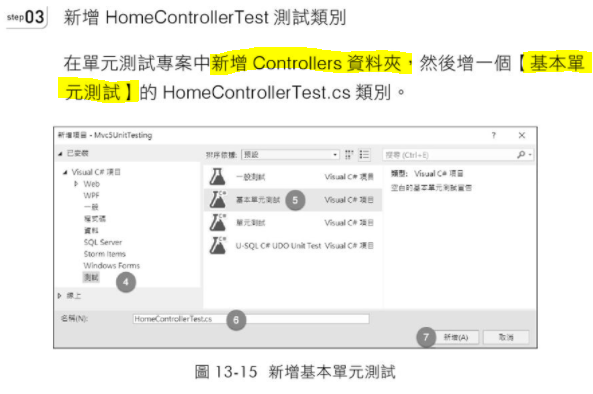


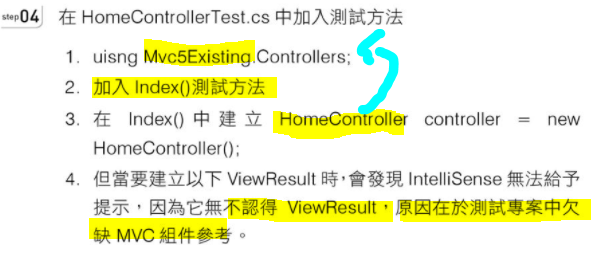
## 參考方案中的另一個專案





## 新增基本單元測試類別

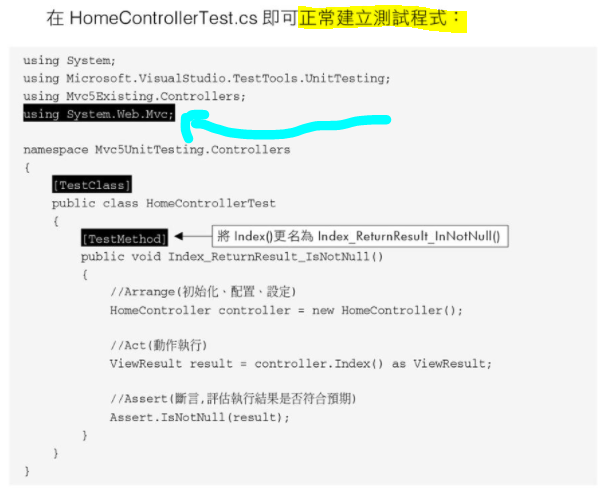


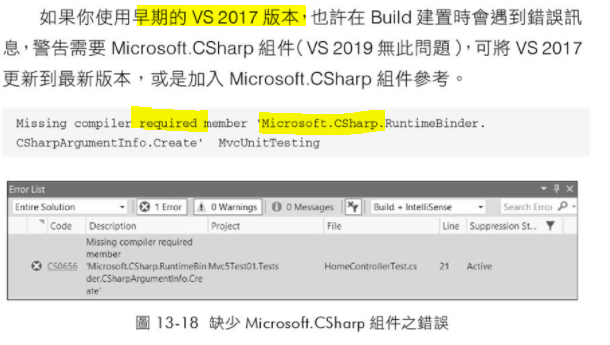




## 參考另一個專案的mvc組件

不是 nuget





not likely to happen

