|  |  |
| --- | --- |
| aPPIUM  學習紀錄及錯誤點 | 摘要  受測對象：ZMoney  受測類型：Android 11 實體手機  測試架構：Appium + Python  鄭又寧  2024/03 開始 |

目錄

[ 自動化測試介紹及流程 1](#_Toc162017760)

[ Appium 架構與設計哲學 1](#_Toc162017761)

[ 各模組做的事 2](#_Toc162017762)

[【指定元素的資料】 2](#_Toc162017763)

[ 架設Appium測試環境 6](#_Toc162017764)

[ 驗證環境架設成功 6](#_Toc162017765)

[ 元素定位方式 6](#_Toc162017766)

# 自動化測試介紹及流程

## Appium 架構與設計哲學

* 不應該為了自動化而修改或重新編譯（recompile）應用程式。
* 不應該被鎖定在特定語言或框架之下撰寫、及進行測試。
* 自動化測試框架不應該在 APIs 重造輪子，重複已有的東西。
* 無論在名義上/精神上/實做上，自動化測試框架應該開源。

【優點】

1. 可以直接使用APK測試。

【缺點】

## 各模組做的事

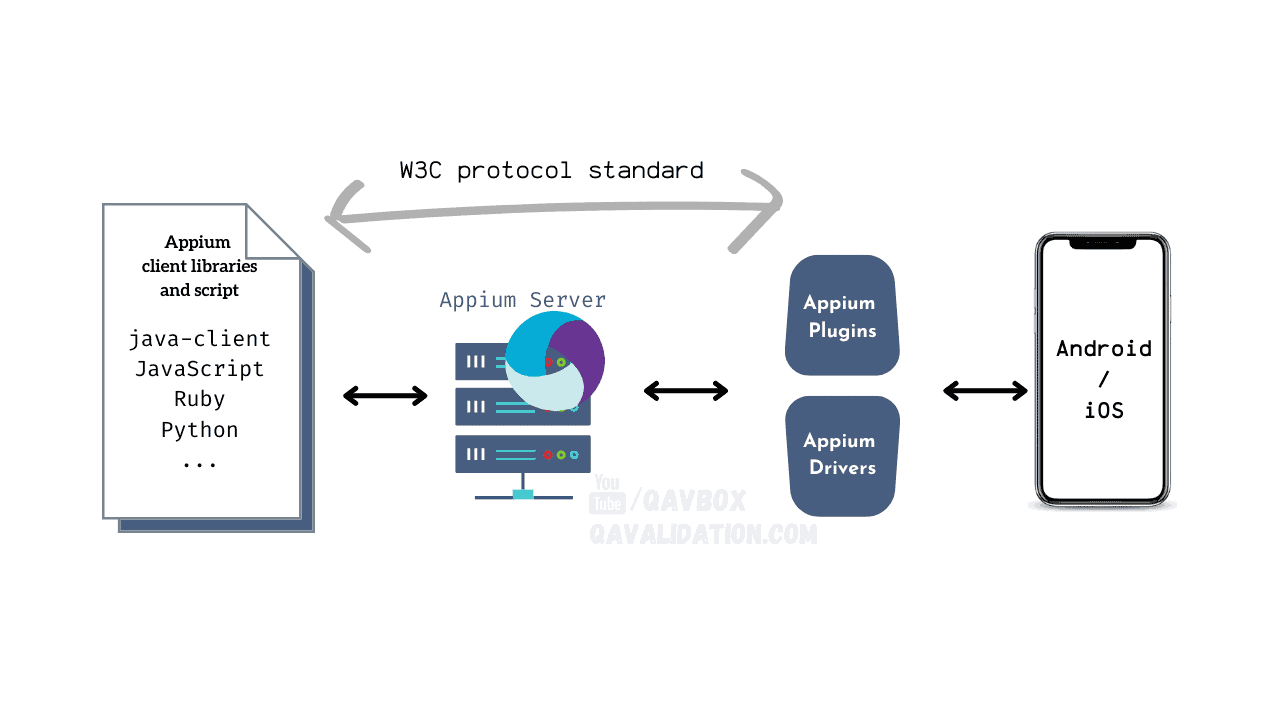
利用WEditor找UI指定元素的資料，用以定位。

利用ADB連結手機。

利用Appium-Python-Client開發測試內容。

利用UiAutomator2模擬點擊等動作。

利用Appium Server進行自動化。



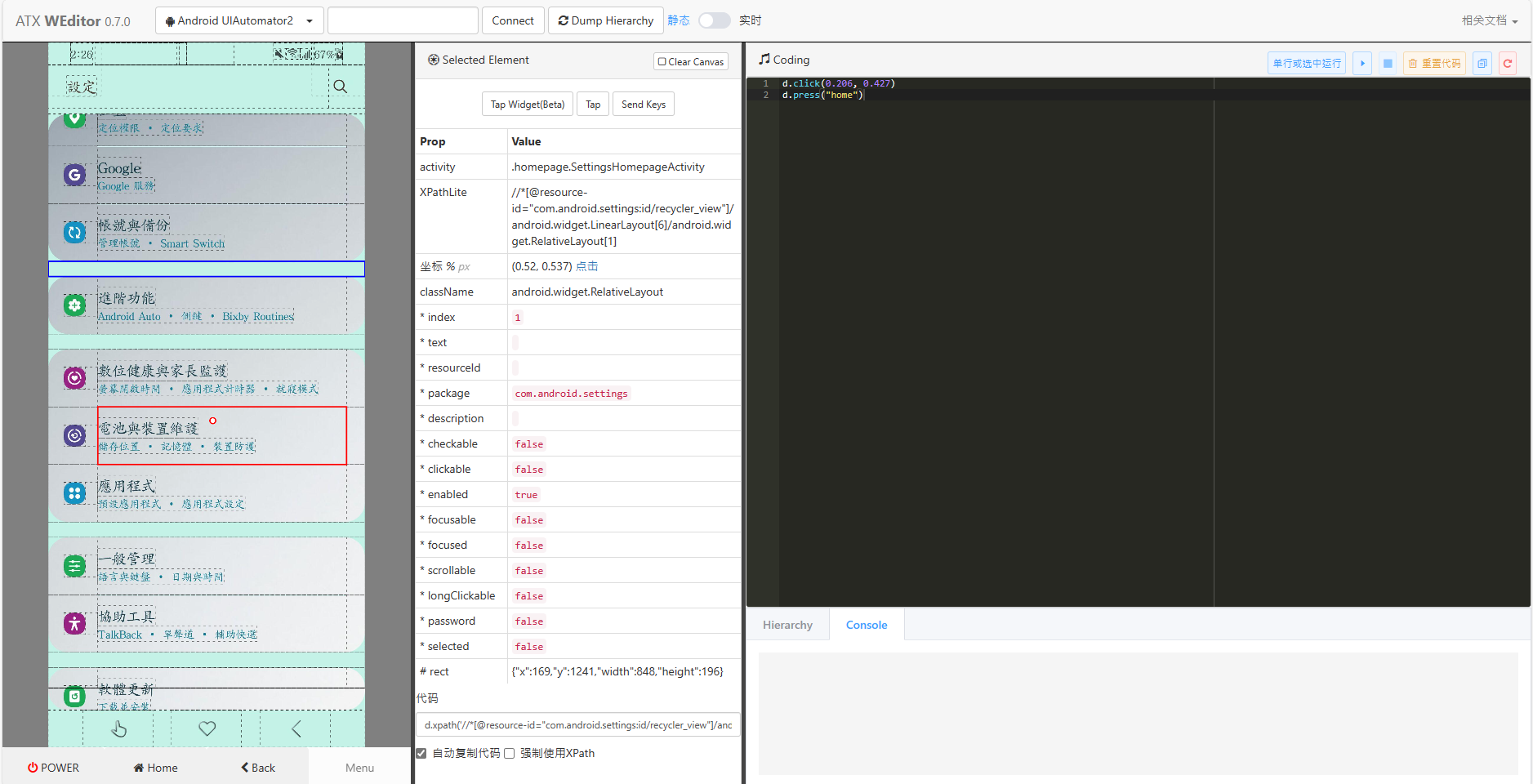
### 【指定元素的資料】

想要查看UI元素的資料有三種方法，按照效率及方便程度，選擇第一種方法。

1. (選擇)Python的WEditor模組

※安裝方法請見下一章節。

* 1. 使用UiAutomator2，因此可以與Appium Server共存
  2. 查詢介面為WEB，簡單明瞭，且同時提供package / activity / XPathLite / className



1. Android SDK的uiautomatorviewer.bat
   1. 使用UiAutomator，因此**不可以**與Appium Server共存！！！
   2. 只有提供className / package
   3. 為Android提供的工具，下載android studio 或Android SDK就可以獲得，但需要設定Java，設定上容易出現相容性問題。
   4. 會另開視窗

路徑：C:\Users\s7500\AppData\Local\Android\Sdk\tools\bin

相關系統變數：

|  |  |
| --- | --- |
| ANDROID\_SDK\_ROOT | C:\Users\s7500\AppData\Local\Android\Sdk |
| JAVA\_HOME | C:\Program Files\Android\jdk\jdk-8.0.302.8-hotspot\jdk8u302-b08 (備註：應該是安裝android studio時帶得JDK8，此版本有自動加裝JRE，新JDK聽說沒有，因此會出問題) |
| CLASSPATH | .;%JAVA\_HOME%\lib;%JAVA\_HOME%\lib\tools.jar |
| PATH | %ANDROID\_SDK\_ROOT%、%JAVA\_HOME%\bin、  %JAVA\_HOME%\jre\bin、  C:\Program Files (x86)\Android\android-sdk\platform-tools |

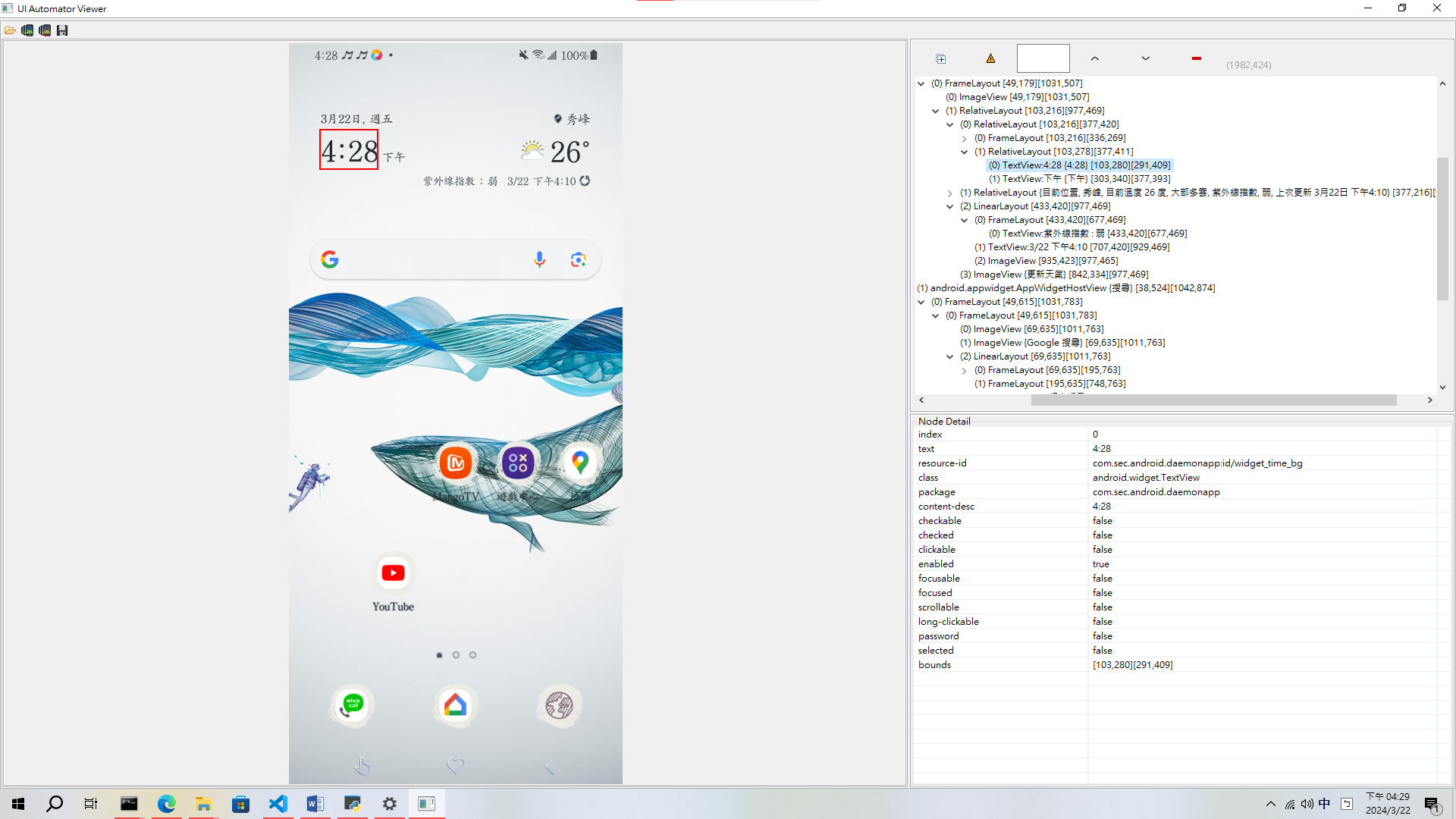
安裝時遇見的問題：

**Djava.ext.dirs=C:\\Users\\s7500\\AppData\\Local\\Android\\Sdk\\tools\\lib\\x86\_64;..\\lib is not supported. Use -classpath instead.**

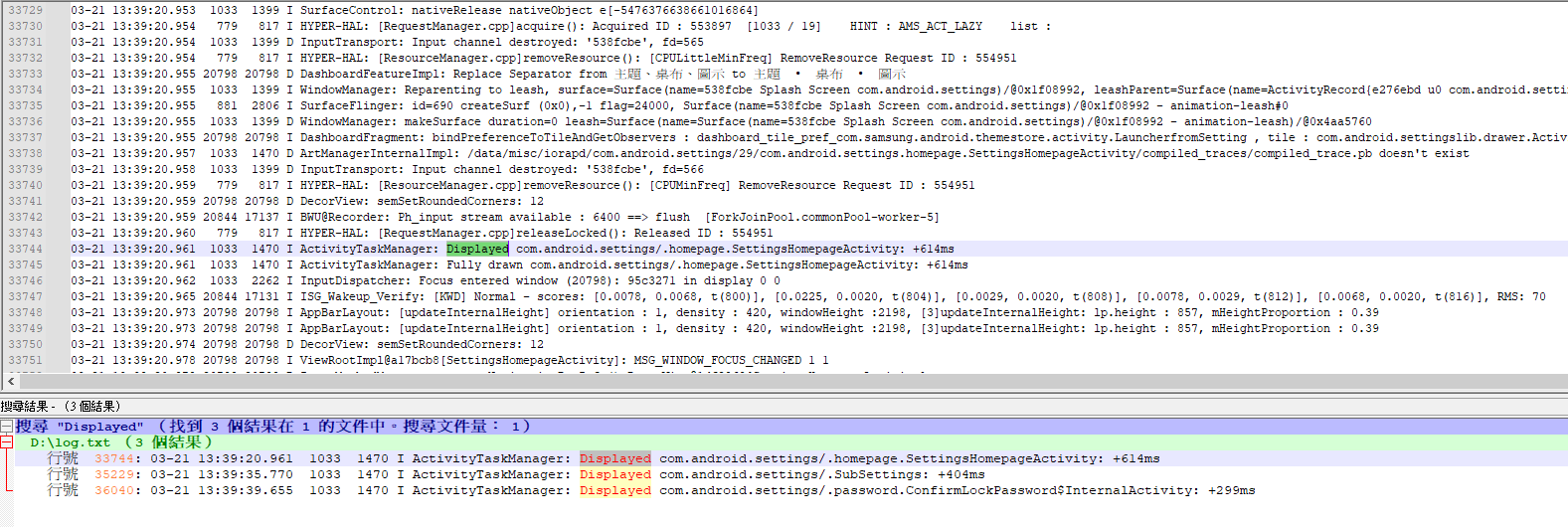
**Error: Could not create the Java Virtual Machine.**

**Error: A fatal exception has occurred. Program will exit.**

* + 1. 系統變數需加上CLASSPATH
    2. Java 虛擬機配置有問題，可能原因(準確原因不知道)：
       1. 系統變數設定錯誤
       2. JAVA版本不相容
       3. 沒有安裝JRE
       4. 空間不足



1. 透過系統log查看appPackage/appActivity(書上的方法)
   1. Android無時無刻都在刷新log，資料量很龐大，以關鍵字Displayed搜尋，也會有很多個，很難鎖定確切的appPackage/appActivity。
   2. 資料量過於龐大，ADB容易卡住，造成電腦當機。
   3. 不能針對頁面元素。



# 架設測試環境

## 安裝Node.js

系統變數

|  |  |
| --- | --- |
| NODE\_PATH | D:\Janny\nodeJs |
| Path | %NODE\_PATH% |

## 安裝JDK

## 安裝Appium Server

※備註：2023年Appium-desktop已被棄用，現在為Appium2.5，使用命令功能視窗操作。

1. 以系統管理員身分執行CMD，並輸入npm i -g appium
2. appium driver install uiautomator2
3. 以系統管理員身分執行CMD，直接輸入appium，會執行程式並出現[Appium] Welcome to Appium v2.4.1及Available drivers: - uiautomator2@3.0.2 (automationName 'UiAutomator2')等資訊。
4. 執行第一次測試時會在手機裡自動安裝對應Appium Setting。

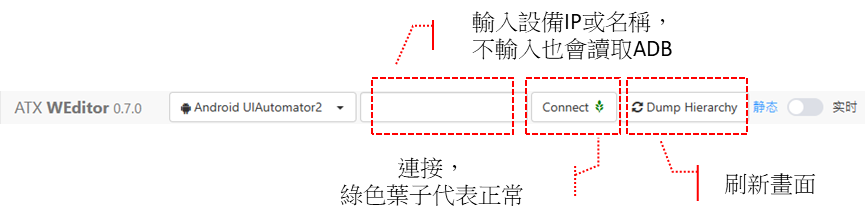
## 安裝Python(省略)

## 安裝Appium-Python-Client

1. 打開CMD，輸入pip install Appium-Python-Client

## 安裝UiAutomator2及WEditor

1. 打開CMD，輸入set PYTHONUTF8=1 (如果沒有，後面會跳錯誤訊息CP950編碼異常)
2. 輸入 pip install –U weditor==0.7.0 (UiAutomator2會一起被安裝)
3. (請確定ADB系統變數已被正確辨認)使用USB線連接手機
4. 輸入 系統uiautomator2 init，並確認success訊息  
   該動作會在手機裡安裝ATX APP，若沒有執行，會一直跳下列錯誤  
   **Local server not started, start with python -m weditor**  
   ※備註：網路上查都是輸入python -m uiautomator2 init，但不會有反應。
5. 輸入 weditor –shortcut  
   此指令會在桌面設定捷徑，可以點擊開啟。



## 設定系統環境變數

|  |  |
| --- | --- |
| 使用者變數 | |
| ANDROID\_SWT | C:\Users\s7500\AppData\Local\Android\Sdk\tools\lib\x86\_64 |
| NODE\_PATH | D:\Janny\nodeJs |
| PATH | %NODE\_PATH%、%ANDROID\_HOME% |
| 系統變數 | |
| PATH | %NODE\_PATH%、%ANDROID\_HOME%、C:\Program Files\Common Files\Oracle\Java\javapath、C:\Program Files (x86)\Android\android-sdk\platform-tools (ADB) |

在CMD分別輸入 adb version 、java –version 、node –v，確認正確取得版本號。

# 驗證環境架設成功

※以官網上的程式碼做修改，目的是利用Appium測試開啟Setting裡的連接頁面，目的是測試整個測試環境是否架設成功。

## 【程式碼】

*import unittest #單元測試的標準python函式庫*

*from appium import webdriver #appium的驅動程式*

*from appium.webdriver.common.appiumby import AppiumBy #定位元素*

*#設置和配置 UiAutomator2 測試引擎的選項*

*from appium.options.android import UiAutomator2Options*

*#告訴伺服器app執行的環境。*

*capabilities = dict(*

*platformName="Android", #使用的行動平台*

*platformVersion="11.0", #指定平台的系統版本*

*#使用的自動化引擎，用appium伺服器執行時提供的資訊*

*automationName="UiAutomator2",*

*#啟動的裝置、實機、模擬機(adb devices 查詢名稱)*

*deviceName="R58MC1QYRTM",*

*#被測試APP的Package名，取自WEditor*

*appPackage="com.android.settings",*

*#被測試APP的Activity名，取自WEditor*

*appActivity=".homepage.SettingsHomepageActivity"*

*)*

*#Appium Server的Ip位置，取自執行中的Appium*

*appium\_server\_url = "http://127.0.0.1:4723"*

*#使用 UiAutomator2Options 加載設定的選項*

*capabilities\_options = UiAutomator2Options().load\_capabilities(capabilities)*

*class TestAppium(unittest.TestCase): # class A (class B) A繼承至B*

*"""*

*利用python標準庫unittest建立測試類別。*

*繼承至unittest.TestCase類別(必要);*

*目的：標準化的測試結果。*

*"""*

*def setUp(self) -> None: #回傳None*

*"""*

*定義測試使用案例的前置動作:建立與AppiumServer的聯繫。*

*command\_executor:指向 Appium 服務器的 URL。*

*options:app執行的環境設定, 已使用 UiAutomator2Options 加載。*

*"""*

*self.driver = webdriver.Remote(command\_executor=appium\_server\_url, options=capabilities\_options)*

*def tearDown(self) -> None: #回傳None*

*"""*

*定義測試使用案例的結束動作。*

*檢查是否有 Appium 驅動程式實例存在，如果有則退出。*

*"""*

*if self.driver:*

*self.driver.quit()*

*def test\_find\_wifi(self) -> None: #回傳None*

*"""*

*測試案例：找到WIFI設定頁面。*

*unittest的規則：測試方法一律以test\_開頭。*

*"""*

*el = self.driver.find\_element(by=AppiumBy.XPATH, value="//\*[@text='連接']")*

*el.click()*

*#執行unittest的main方法。*

*if \_\_name\_\_ == "\_\_main\_\_":*

*unittest.main(exit=False)*

*#若無exit=False，會一直跳SystemExit，需再研究。*

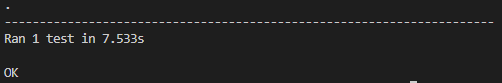
1. 原計畫是按照官網尋找Battery，因手機介面為中文，因此以「電池與裝置維護」為關鍵字，但會一直報錯。

原因：當開啟設定時，顯示下方畫面，driver也只會找此範圍的元素，因此不論是用text或其他定位法都無法準確找到電池，需要額外下滑動指令。

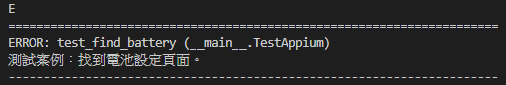


1. 引用unittest模組需要遵守他的規則。(擷取自至尊王者selenium3一書)
   1. 建立一個測試類別並一定要繼承至unittest.TestCase
   2. 建立測試方法一定要以test\_開頭。
2. 以unittest.main()執行測試時，unittest會依照方式名稱排序並依序執行，先執行test\_axx再執行test\_bxx。若要自訂順序，需使用TextTestRunner。

# 測試結果檢視

【成功結果】  


【失敗結果】



【說明】

“ . ”：第n項測試執行通過

“ F ”：第n項測試執行失敗

“ E ”：第n項測試執行錯誤，會有errors

“S ”：第n項測試跳過執行

# 元素定位方式