行動裝置 App 自動化測試工具—Appium 之研究

專題編號：110-CSIE-S001
執行期限：109 年第 1 學期至 110 年第 1 學期
指導教授：郭忠義
專題參與人員： 107590012 陳志榮
107590039 歐陽文立
107590045 潘榮祥

一、摘要

本專題規劃研究一個能以較接近自然語言的「測試行為劇本」為輸入，優化傳統行動裝置測試缺點，產生「測試行為劇本之半自動化程式指令」的測試工具。

二、緣由與目的

行動裝置程式現今蓬勃發展，為了提升軟體系統品質，需要針對其龐大的程式做測試，且需要解決跨平台及版本相容問題，因此需要大量的測試程式以及測試劇本。由於測試程式其操作具高度重複性，為了簡化高度的重複動作，本專題規劃發展一套系統，可以達到使用者只要設計符合格式的測試行為與資料的劇本文件，便能產生測試行為步驟之指令程式碼的元件導向半自動介面測試工具並可兼容不同平台使用。

三、研究範圍

本研究主要為了能將手機本身自動化測試流程優化。

傳統系統功能測試，其測試為先將測試內容與程式撰寫完畢並封包之安裝檔中，在執行時便可對 App 進行測試，導致新增測試困難，並且平台之間因撰寫安裝檔的語法不同無法兼容，導致測試無法並用之缺點。

本專題將改進其對於更新測試時須將安裝檔進行拆包再進行撰寫的冗長動作，利用 Appium 對已下載 App 不需要了解其內部邏輯或是重新編譯的優點進行測試，來達到方便且快速的新增所需測試且 Appium 為基於 Webdriver 協定來進行 App 測試便可兼容各平台，達到在各平台測試的兼容之優點並進行優化針對前述行動裝置程式系統功能自動化測試，改善重複且可自動化步驟，實現介面測試元件的半自動化建置，著重於寫好腳本後自動套用已建構好的測試程式 Keyword 以實施自動化測試，達到以更友善更直觀的方法，使用更接近自然語言的形式達到使用者想要執行的測試動作。

四、使用技術方法

本專題目的是把寫行動裝置測試步驟的部分自動化。使用 Appium 工具及命名原則來進行測試及增加可讀性。以下描述本專題使用到的技術。

1. Appium 基於 webdriver 協定新增對移動裝置自動化 api 擴充套件而成的，所以具有和 webdriver 一樣的特性，比如多語言支援，提出自動化、精實、快速反應等技術提升軟體品質與系統可靠與可用。
2. Webdriver 是基於 http 協定，第一連線會建立一個 session 對談，並通過 post 傳送一個 json 告知伺服器端相關測試資訊。
3. Keyword 是比較接近自然語言的函數。通常使用者看到 keyword 就可以大概知道其功能與目的。以下描述網路元件與對應 keyword 對照。

Term	description	Keyword
Button	按鈕	Click Element
Input Text	輸入框	Click Element
Application	開啟應用程式	Open Application
Notifications	拉下通知欄	Open Notifications
Background	將程式放置後台運行	Background App
Context	如同網頁的 Frame	Switch To Context

Home	行動裝置主頁面	Press Keycode 3
App Switch	進入 App 切換器(可做為切換 App 的方式)	Press Keycode 187
Swipe	從一個點滑動到另一點	Swipe
Zoom	放大行動裝置畫面(可用於測試 RWD)	Zoom
Lock	鎖定行動裝置	Lock

常用的Keyword多為已存在原生Keyword的組合，或是輔助的 Python code。以下列出幾個常用的 keyword。

(1) Get_webelement()：參數放 locator，透過 locator 在頁面尋找對應網頁元件。

(2) Close_Application()：關掉開啟應用程式。

(3) Capture_page_screenshot()：截圖所在畫面。通常利用此 keyword 在 fail 的時候將畫面截圖，以了解出現錯誤時畫面。

(4) Wait_until_page_contains_element()：有時執行動作，app element 載入畫面需一段時間，此 keyword 作用是等到該元件出現後才繼續進行，否則若畫面該元件還沒出現就對其執行動作會出現錯誤。

3. Appium 元素定位方式：

app 自動化測試過程中最重要一個環節就是元素定位，只有準確定位到了元素才能進行相關元素的操作，如輸入、點擊、拖拽、滑動等，appium 也提供了許多元素定位的方法，如 id 定位、name 定位、class 定位、xpath 定位等等。

方法	說明
find_element_by_id(id)	傳回符合指定 id 名稱之元素
find_element_ by_class_name(name)	傳回符合指定 class 名稱之元素
find_element_by_xpath (xpath)	傳回符合指定 xpath 之元素
find_element_ by_android_uiautomator ('new UiSelector(). attribute("%s"))	利用 uiautomator 定位指定屬性並傳回符合指定屬性名稱之元素 attribute 為欲搜尋之屬性 %s 為指定屬性名稱

定位元素後，也可以透過 get_attribute("%s")獲取該元素屬性資料，常獲取屬性資訊有：

方法	說明
get_attribute("id")	傳回定位元素之 id
get_attribute("class")	傳回定位元素之 class
get_attribute("name")	傳回定位元素之 content-desc
get_attribute("text")	傳回定位元素之 text
get_attribute("size")	傳回定位元素之 size 字典
get_attribute("location")	傳回定位元素之 location 字典
get_attribute("clickable")	傳回定位元素之 clickable 布林值
get_attribute("checkable")	傳回定位元素之 checkable 布林值

五、工具說明

1. Robot framework：劇本實做部分。透過程式碼模擬人工測試，達自動化測試。
2. Appium API:行動裝置自動化測試工具。
3. Python：無法由 Appium 原生 Keyword 完成，以 python 輔助。
4. RED, VScode: Integrated development environment

六、研究結果

(一) 根據需求文件將測試流程整理為 Appium 可執行指令。

(二) 將指令以符合 Robot Framework 的規範進行撰寫。

(三) 使用 Jenkins 對完成腳本進行持續整合(Continuous Integration)

七、參考文獻

[1] Appium Library
https://serhatbolsu.github.io/robotframework-appiumlibrary/AppiumLibrary.html
[2] Robot Framework
https://robotframework.org/#/
[3] Red IDE
https://github.com/nokia/RED