**測試流程說明書**

**1. 測試環境初始化**

* **測試框架載入**
  + 在執行所有測試之前，由 TestBase 中的 [AssemblyInitialize] 進行環境設置，包含設定日誌、目錄參數以及監控檔案變更的 watchers 等。
* **WinAppDriver 啟動**
  + 測試過程使用 Windows Application Driver 驅動目標應用程式。測試準備階段需確認 WinAppDriver 已啟動，以及目標應用程式（如 PP5 IDE 等）的路徑、工作目錄正確配置。

**2. 驅動連線與應用程式啟動**

* **驅動建立與附加**
  + 在 DriverBase 中檢查目標應用程式是否已在執行，如果存在則透過 AttachExistingDriver 取得相關視窗句柄並建立 PP5Driver 連線；否則則使用 CreateDriver 新建驅動連線。
  + 使用 AppiumOptions 設置日志等參數，並反覆嘗試連線以避免超時錯誤。
* **主視窗與 IDE 驗證**
  + 測試初始化中使用 TestBase.PP5IDEWindow，若視窗不存在或過時，就會重新透過 GetPP5Window 取得有效的應用程式視窗。

**3. 測試用例流程**

針對 TI Editor（及其他功能模組）測試，具體流程如下：

**3.1 操作流程**

* **新建文件/編輯操作**
  + 測試用例（如 TIEditor\_CreateNewTIByTypeSubTIRunTypeUUTTest）會點擊主介面上相關按鈕，利用 AutoUIExecutor 執行元素搜尋（依據 Xpath 或其他定位策略）。
  + 每個點擊動作都會對應一個 Console 輸出日誌，記錄「LeftClick」等操作步驟，並在無法找到元素時輸出錯誤訊息。
* **文件保存與加載**
  + 測試流程包含「SaveTI」和「SaveAsTI」操作，針對新建或已存在的測試項目進行不同方式保存。
  + 測試用例可能會驗證文件中描述內容或 TI 名稱是否正確更新，並採用斷言（Assert 或 ShouldEqualTo）比對預期與實際值。
* **組群與其他參數設定**
  + 部分測試用例會對文件的群組或描述資訊進行修改，再透過關閉與重新打開編輯器來驗證修改是否生效。

**3.2 元素互動與操作方法**

* **點擊操作**
  + 通過 PerformAction 模組中的 PerformClick 方法進行點擊，支援不同的點擊型態（如左擊、雙擊、右擊），並可指定點擊位置的偏移。
* **輸入操作**
  + 使用 PerformInput 方法針對文字框進行輸入，可以支援單鍵或組合鍵輸入，例如在名稱或備註欄位輸入新值。
* **DataGrid 操作**
  + 當操作表格中元素時，會依據元素定位字符串（如 ByDataGrid、ByRow、ByCell）分別進行操作，並針對不同格式的定位參數做出相應動作。
* **ToolBar操作**
* **MenuItem操作**
* **TreeViewItem操作**

**4. 日誌記錄與錯誤處理**

* **日誌記錄**
  + 測試過程中，所有操作均透過 Console 輸出詳細步驟，並由驅動日誌記錄更深入的 Appium 與 Driver 日誌（包括 Client、Driver、Server 的詳細資訊）。
* **錯誤處理**
  + 當定位不到元素或點擊操作失敗時，會在 Console 輸出錯誤訊息並提前結束該測試用例。
  + 驅動建立階段也設有例外捕捉機制，針對超時或無法附加已存在進程的情況進行重試。

**5. 測試結束與資源清理**

* **測試結果斷言**
  + 每個測試用例最後都會依據 bSuccess 標誌進行斷言，確保操作成功。如果斷言失敗，則將存儲截圖與日誌幫助後續問題追蹤。
* **清理操作**
  + 在每個測試用例結束後（TestCleanup），會釋放驅動資源或關閉多餘的視窗，確保每次測試執行前環境乾淨。
  + AssemblyCleanup 階段也會進行系統命令路徑、暫存檔案的清理操作。

**如何建立一個測試方法?**

1. **以管理員權限(admin)**開啟Visual Studio，並載入專案
2. 對於DataGrid/TreeView/Menu/Toolbar這類控鍵，沒有特定的ID/Name，需使用對應的方法；其餘所有可錄製控鍵，都可以UI Recorder直接進行錄製
3. 新增一個單元測試方法，複製錄製後的結果到該方法內並在測試總管中執行
4. 等候UI動作結束，可以查看測試紀錄Log

**輔助工具 : WinAppDriver UI Recorder**

**※ XPath 構建策略詳細解說**

* **XPath 路徑構建的基本原則**
  + 在 WinAppDriver UI Recorder 中，XPath 構建是透過 [GenerateXPathToUiElement](vscode-file://vscode-app/c:/Users/adam.chen/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ") 方法實現的。
  + 此方法採用精心設計的分層策略，以確保生成的 XPath 表達式既精確又穩定，能夠在應用程式的不同執行階段可靠地識別 UI 元素。
* **元素識別的優先順序**

XPath 表達式的構建遵循以下優先順序：

1. **元素標籤（Tag）**：永遠作為 XPath 的基礎元素，例如 /Button、/TextBox
2. **AutomationId**：如果存在且非動態生成，將作為首選識別屬性
3. **Name 屬性**：通常是使用者可見的文字，如按鈕文字或標籤文字
4. **ClassName**：當 AutomationId 不可用時使用
5. **位置索引**：僅當以上所有屬性都不可用時，才使用 position() 作為最後手段

**※ 如何使用 UI Recorder**

* **開始錄製UI動作**

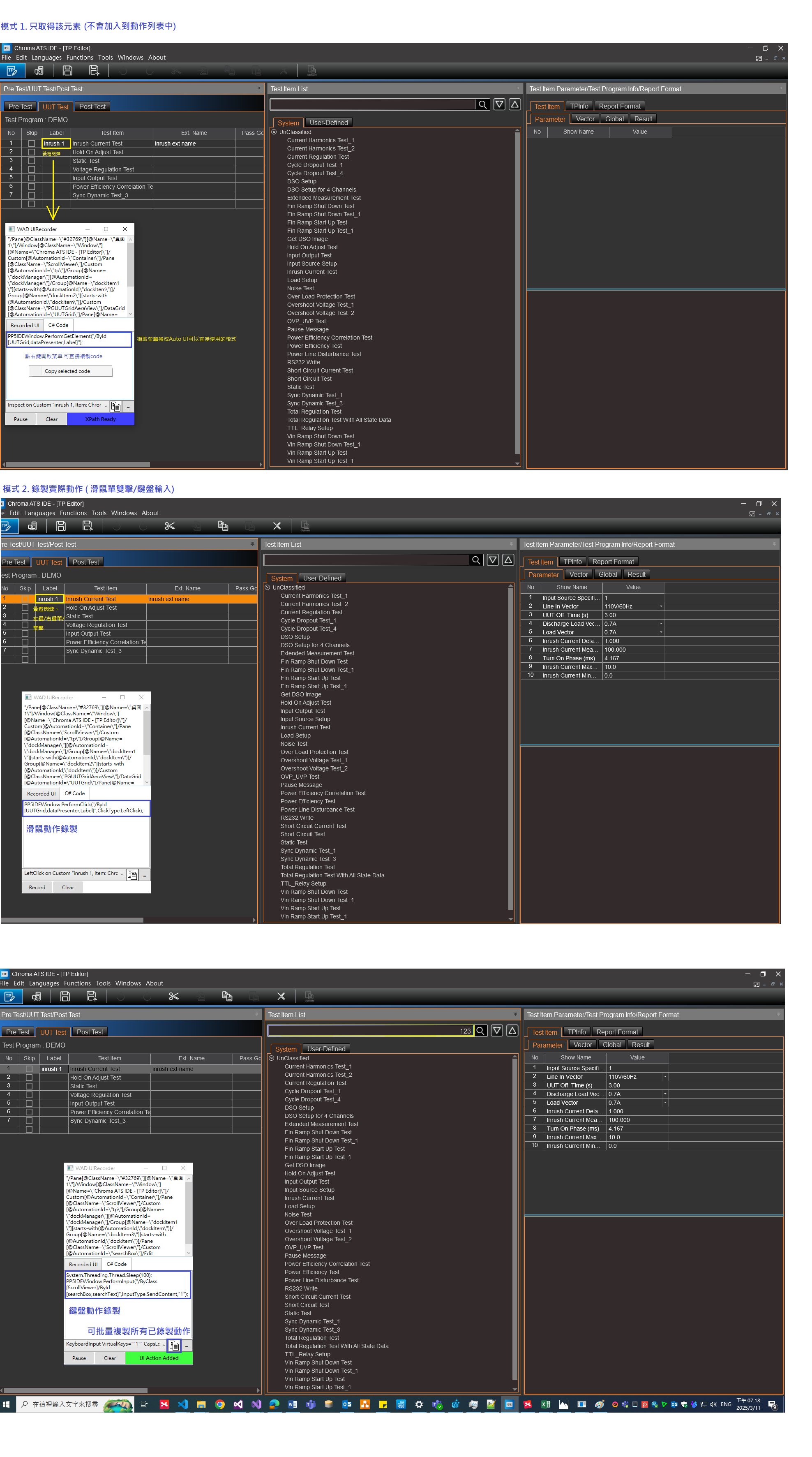
1. 以**管理員權限(admin)** 啟動 WinAppDriver UI Recorder 應用程式
2. 選擇您要錄製的目標應用程式視窗 (Ex: PP5)
3. 點擊"Record"以開始捕獲 UI 動作
4. 錄製完畢後，點擊"Pause"停止錄製

* **支援的 UI 動作**

UI Recorder可以捕獲以下動作:

1. **滑鼠動作：**
   * 左/右/雙擊
   * 拖放
   * 滑鼠滾輪滾動
   * 懸停
2. **鍵盤動作：**
   * 文字輸入
   * 特殊鍵（Enter、Tab、方向鍵）
   * 鍵盤快捷鍵（Ctrl+C、Alt+F4 等）
3. **檢查：**
   * 單純獲取該元素

* **實際操作:**

****

****