

AEDT自動化腳本之GUI開發流程介紹

Mar 2024, Ansys Taiwan

要在 ANSYS AEDT 環境中開發 GUI 結合 Classical API 使用，使用微軟 .Net 框架的 Windows Forms 是一個最合適的選擇。AEDT 允許開發者使用 IronPython 來自動化 AEDT 的操作，包括 HFSS、Maxwell、Q3D Extractor 和 Icepak 等模擬工具的操作。由於 IronPython 是 Python 的 .NET 實現，它能夠無縫地與 .NET 框架集成，**包括使用 Windows Forms 來創建 GUI**。

因為 IronPython 是 Python 的一個實現，運行在 .NET 框架上，與 CPython (Python 的標準實現) 有所不同。因此，在一些庫和框架的支援上會有差異。一些直接依賴於 CPython 的庫，如 Tkinter 或 PyQt/PySide，無法直接在 IronPython 上運行。這是因為這些庫和框架在底層使用了 C 語言的擴展，而這些擴展並不與 .NET 框架兼容。

Windows Form簡介

Windows Forms 是一個 .NET 的 GUI (圖形用戶介面) 框架，用於開發 Windows 桌面應用程序。它提供了一組豐富的控件，如按鈕、文本框、標籤和復選框等，使開發者能夠構建功能豐富的用戶介面。Windows Forms 框架旨在簡化桌面應用程序的開發過程，通過拖放控件和事件驅動的程式設計模型來提高開發效率。

核心特點

- **豐富的控件庫**：Windows Forms 提供了各種預建的控件，包括但不限於文本框、按鈕、下拉列表、菜單、工具條和對話框。這些控件支持廣泛的客戶端應用程序開發需求。
- **事件驅動程式設計**：Windows Forms 使用一種事件驅動的程式設計模型，允許應用程序響應用戶操作（如點擊按鈕、輸入文本等）和其他形式的輸入。
- **易於使用的設計環境**：通過 Visual Studio 等集成開發環境 (IDE) 的支持，開發者可以輕鬆地通過拖放界面元素來設計窗體，並快速設定控件屬性和事件處理函數。

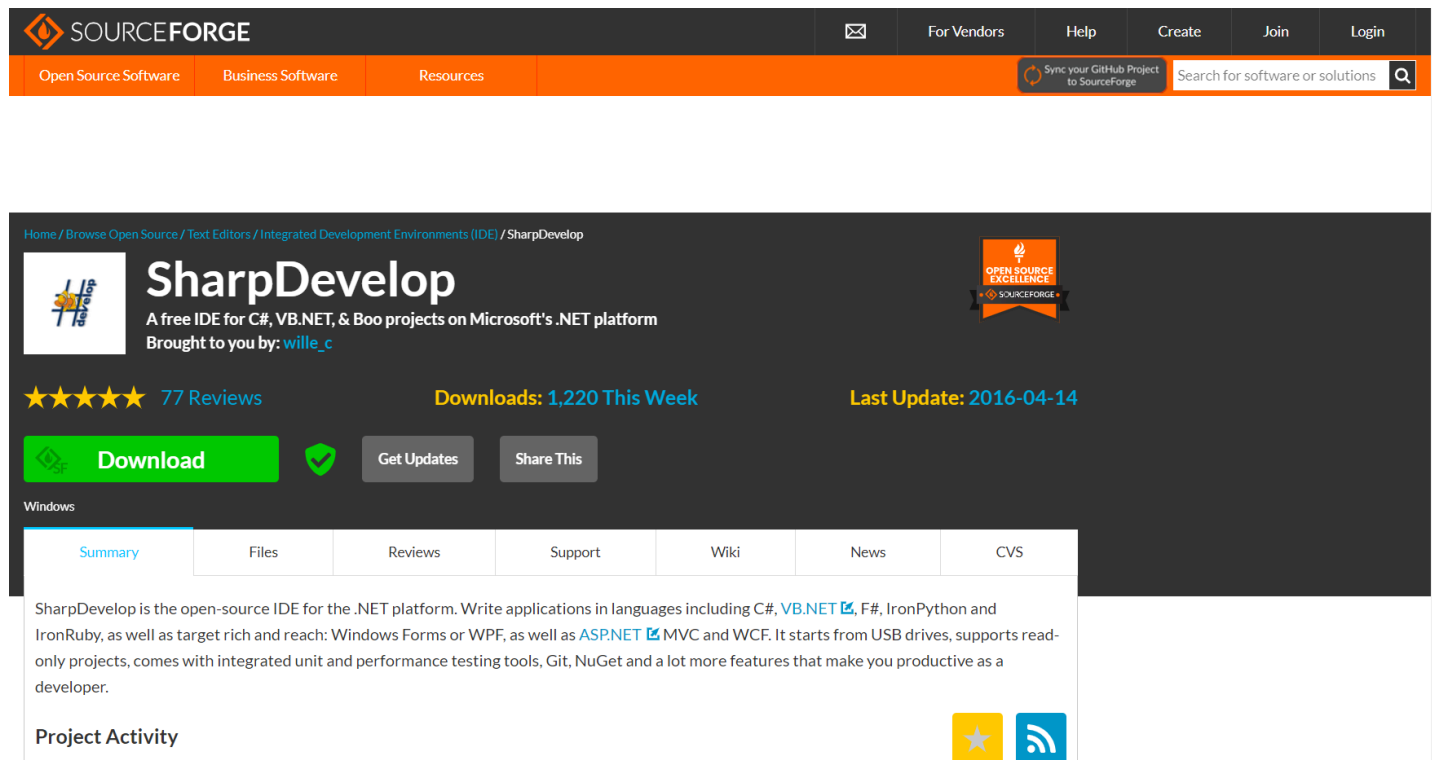
Windows Forms 適用於開發傳統的 Windows 桌面應用程式，尤其是業務和辦公應用程式。它廣泛應用於需要豐富用戶界面和複雜用戶互動的應用程式中。

雖然 Windows Forms 仍然是開發 Windows 桌面應用程式的一個可靠選擇，但 Microsoft 已經推出了其他技術，如 WPF (Windows Presentation Foundation) 和 UWP (Universal Windows Platform)，提供了更現代化的 UI 元素和更豐富的互動特性。這些新技術支持更高級的布局、動畫和觸摸功能，適合於開發現代 Windows 應用程式。

不過，對於許多傳統業務應用程式和那些尋求快速開發和部署桌面應用程式的項目來說，Windows Forms 依然提供了一個簡單而有效的解決方案。

Sharp Develop集成開發環境

"Sharp Develop" (也稱為#develop) 是一個免費的、開源的集成開發環境 (IDE)，主要用於.NET和Mono平台上的程式開發。它支持多種程式語言，如C#、Python等。SharpDevelop提供了許多功能，包括代碼編輯、界面設計、除錯、版本控制等，並且可以通過插件來擴展其功能。



The screenshot shows the SourceForge project page for SharpDevelop. The page header includes the SourceForge logo and navigation links like 'Open Source Software', 'Business Software', and 'Resources'. The main content area features the SharpDevelop logo, a description of it as a free IDE for .NET, and statistics such as '77 Reviews' and '1,220 Downloads This Week'. There are buttons for 'Download', 'Get Updates', and 'Share This'. Below this is a 'Windows' section with tabs for 'Summary', 'Files', 'Reviews', 'Support', 'Wiki', 'News', and 'CVS'. The 'Summary' tab is active, showing a brief description of the IDE and its features. At the bottom, there is a 'Project Activity' section with a star icon and a RSS feed icon.

SharpDevelop的開發始於2000年，由Mike Krüger啟動，目標是提供一個自由且功能豐富的.NET開發環境。雖然它在.NET開發社區中相對較為小眾，特別是與Visual Studio這類主流IDE相比，但它為開發者提供了一個無需付費即可使用的強大工具。

開發範例

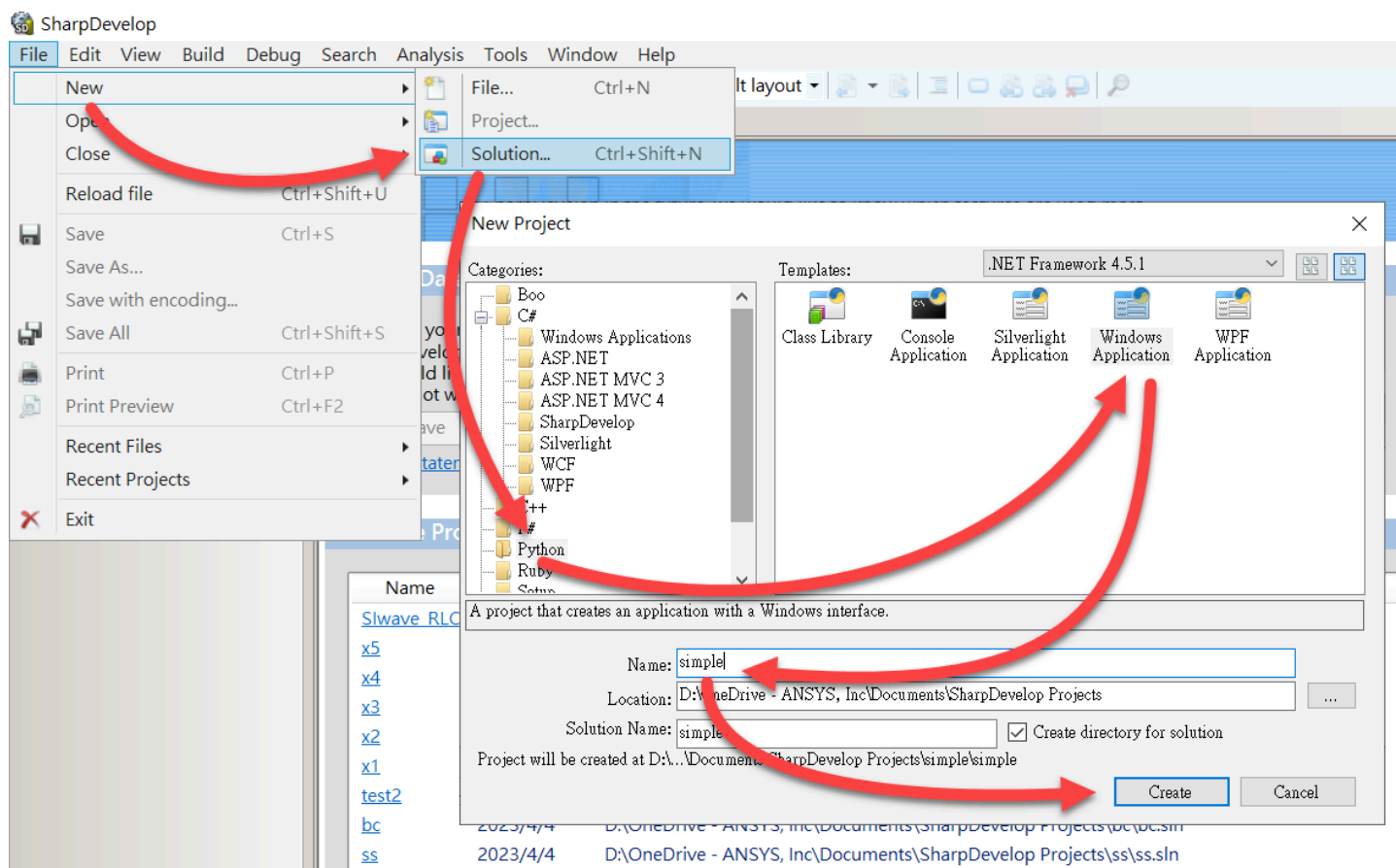
以下是一個簡單操作介面的設計流程，使用者輸入新專案數量，按下按鈕，便可以在 AEDT 自動生成所有新專案。

Step 1. 建立 SharpDevelop 專案

下圖顯示了在 SharpDevelop IDE 中創建新的 Windows Forms 應用程式專案的流程。下面是按照圖片中顯示的步驟進行的：

1. **打開 SharpDevelop**：啟動 SharpDevelop IDE。
2. **創建新專案**：在菜單欄選擇「File」(文件) -> 「New」(新建) -> 「Solution...」(專案)。
3. **選擇專案類型**：在彈出的「New Project」(新建專案) 對話框中，從左側的「Categories」(類別) 列表中選擇「Python」。
4. **選擇模板**：在右側的「Templates」(模板) 選項中，選擇「Windows Application」(Windows 應用程式)。
5. **設定專案名稱和位置**：
 - 在「Name」(名稱) 欄位中輸入專案的名稱，例如：`simple`。
 - 在「Location」(位置) 欄位中指定專案的儲存位置。
6. **創建專案**：點擊「Create」(創建) 按鈕，IDE 會在指定位置創建專案文件和資料夾。

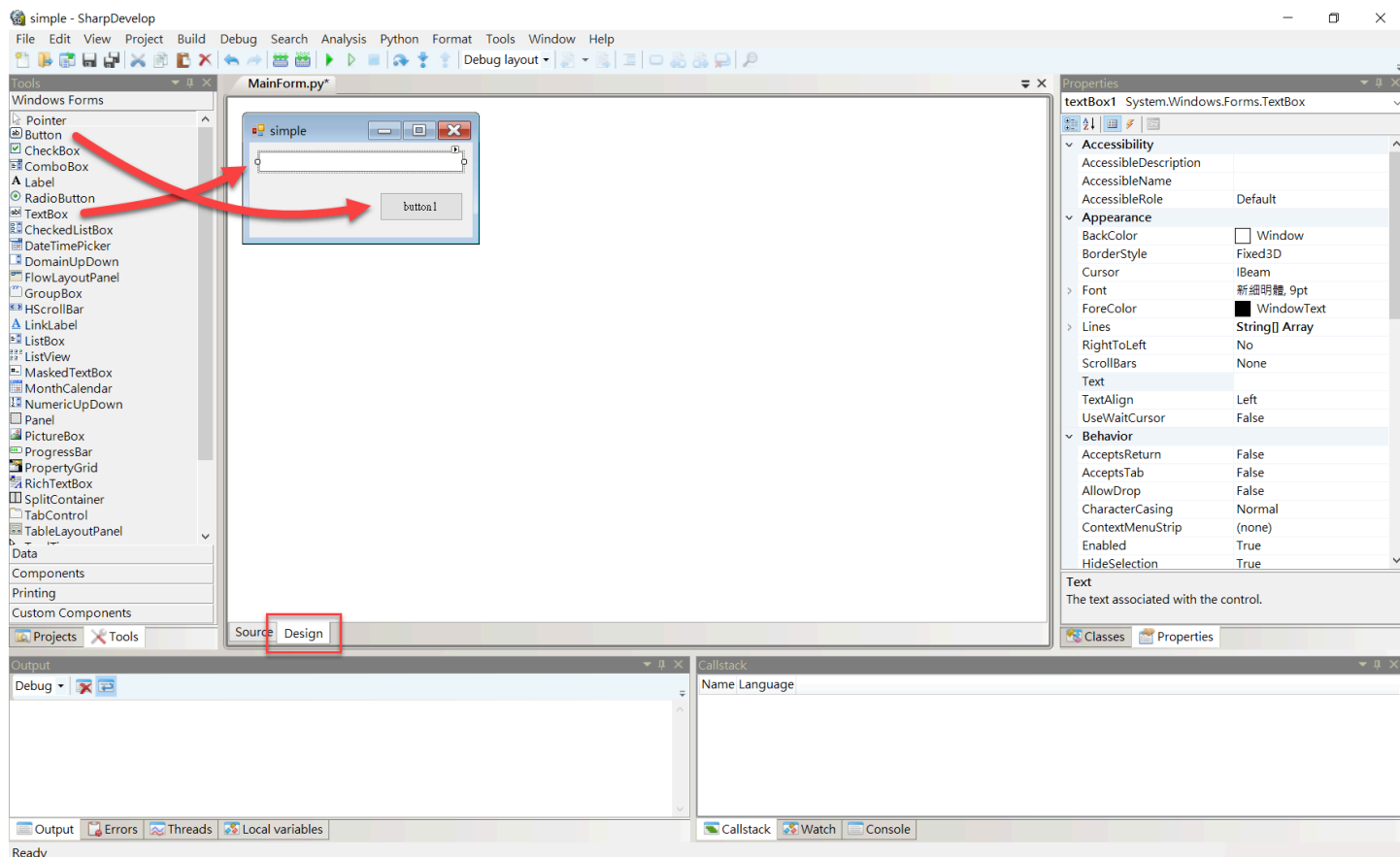
完成上述步驟後，SharpDevelop 會創建一個新的 Windows Forms 專案，並且你可以開始設計你的應用程式的界面，以及編寫和測試代碼。



Step 2. 建立視窗外觀

在 SharpDevelop 的 Design 頁面上拖拉一個 TextBox 和一個 Button 到窗體上，並調整它們的位置和大小。詳細步驟如下：

1. **選擇控件**：在工具箱 (Toolbox) 中選擇你需要的控件。在這個情況下，你已經選擇了 TextBox 和 Button。
2. **拖拉控件**：將控件拖拉到窗體上。你可以通過點擊然後拖動來調整控件的位置。
3. **調整大小**：選中控件後，可以在其周圍看到一系列的調整點 (小方塊)。點擊並拖動這些點可以改變控件的大小。
4. **設計窗體外觀**：除了單個控件之外，你還可以設定整個窗體的屬性，例如大小、背景顏色、標題等。

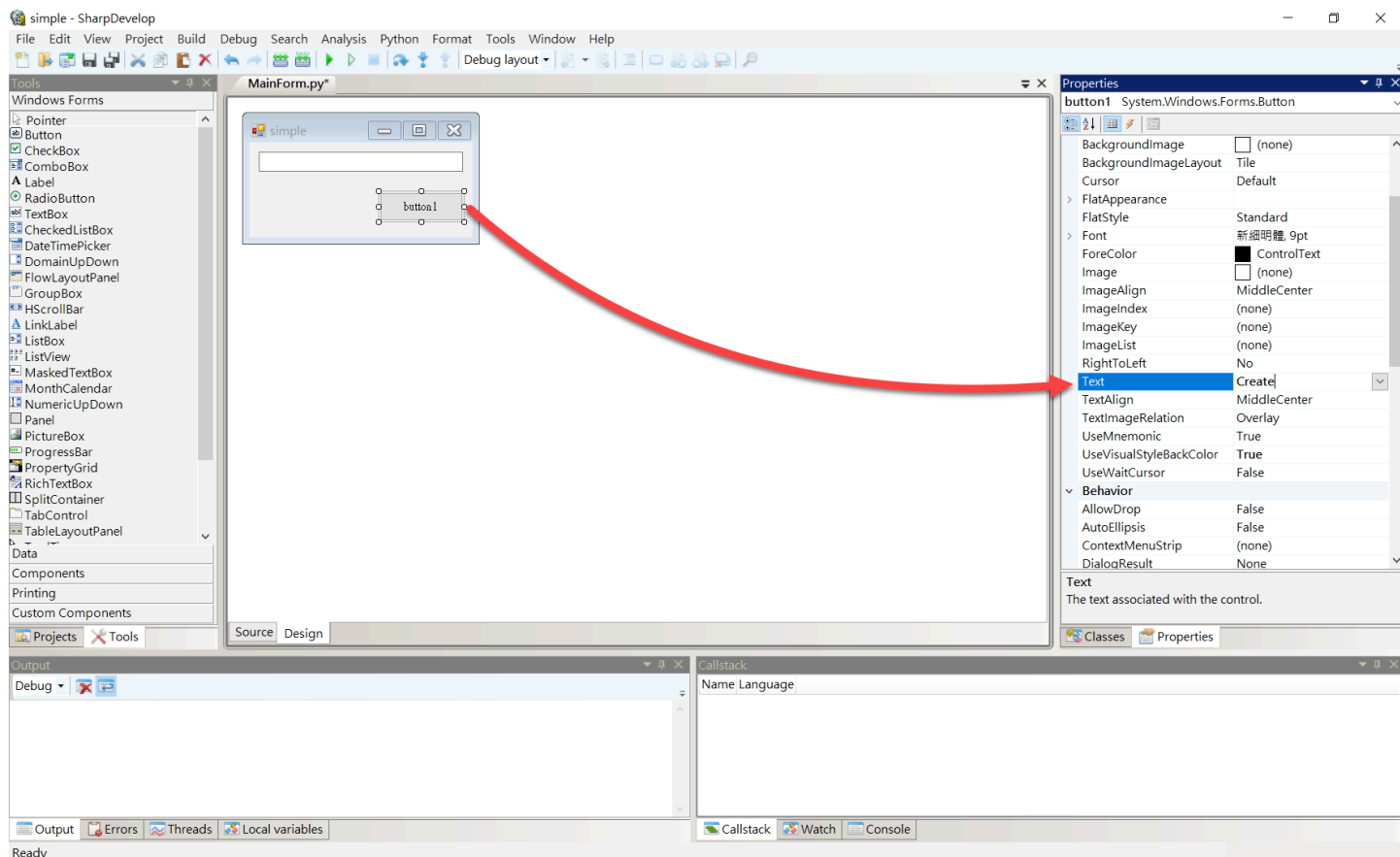


Step 3. 設定屬性與事件函式

選擇 Button 控件。設置Button控件屬性和建立事件處理：

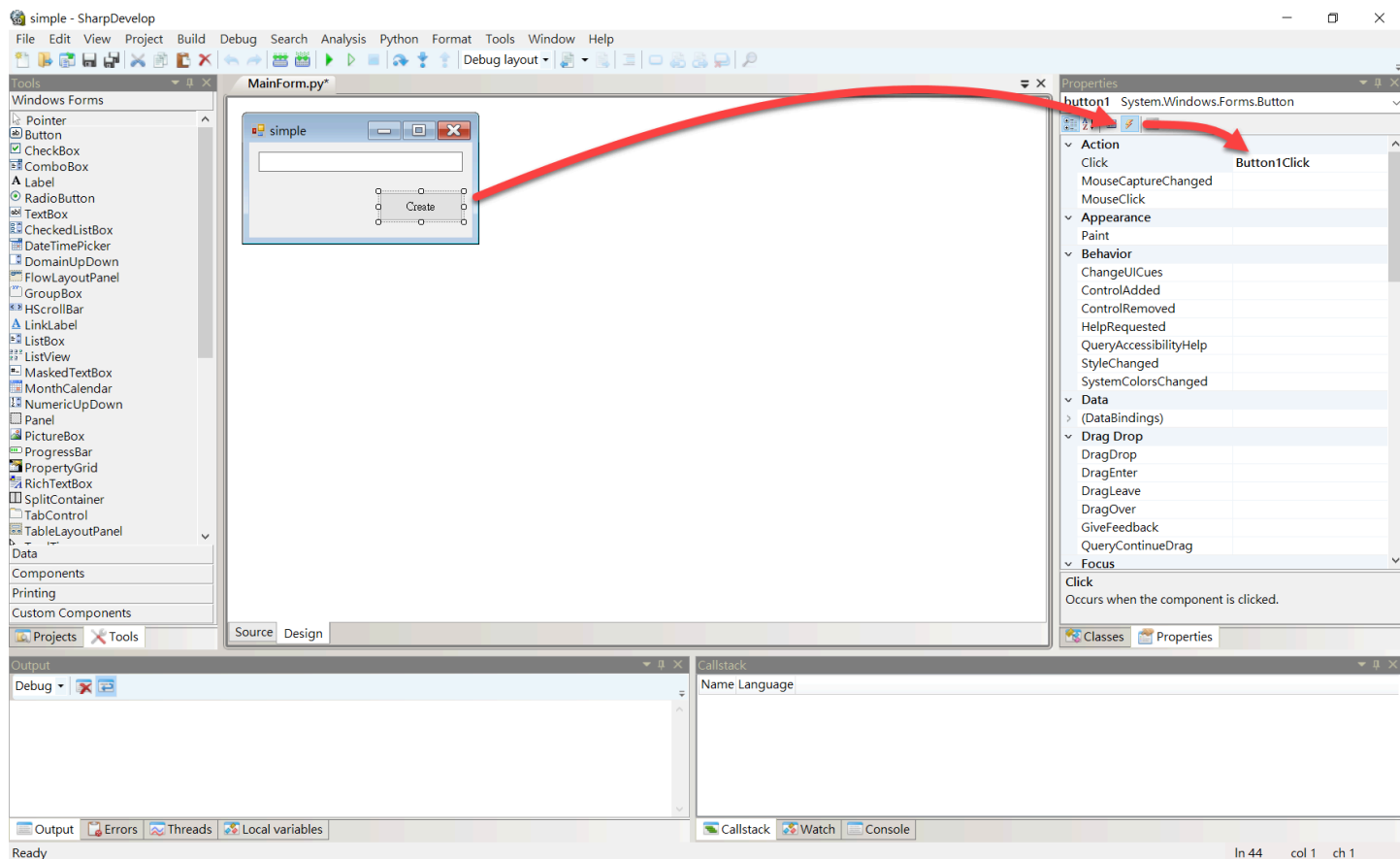
修改 Button 的屬性

1. 在屬性視窗 (Properties) 中選擇 Button 控件。
2. 找到 **Text** 屬性，並將其更改為您希望按鈕顯示的文字，例如 **"Create"**。
3. 您還可以修改其他屬性，如 **Font**、**ForeColor**、**BackColor** 等，以便自定義按鈕的外觀。



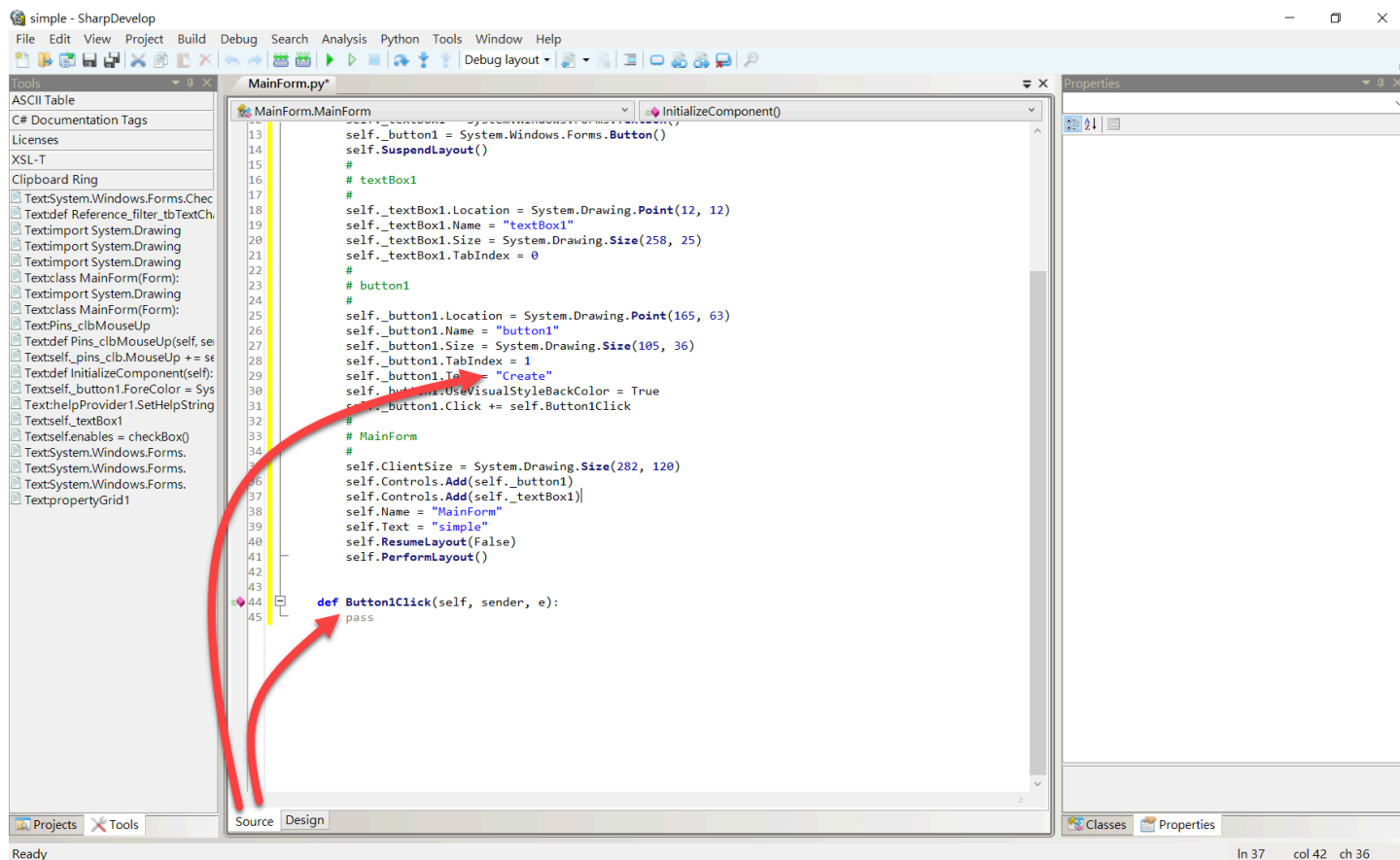
建立 Click 事件

1. 在屬性視窗的頂部，有一個閃電圖標的按鈕，這代表事件（Events）。
2. 點擊這個按鈕，會顯示所有可用的事件。
3. 找到 `Click` 事件，並在它旁邊雙擊空白的區域。這樣會在您的 `MainForm.py` 的源代碼中生成一個事件處理器函數，如 `button1_Click`。
4. 在自動生成的 `button1_Click` 函數中，您可以添加 Python 代碼來定義按鈕點擊時應該發生什麼行為。



檢視Python程式碼

切換到 **Source** 編輯頁面，這裡顯示了 **MainForm** 的 **Python** 代碼。在這個代碼中，您可以看到窗體控件的屬性設定，以及為 **button1** 控件綁定的事件處理函數 **Button1Click**。



Step 4. 建立Python檔案並測試

從 SharpDevelop 環境中將程式碼復制到一個新的文件中，並將該文件保存為 `create.py`。接著，在文件的最開始處加入以下導入模組的語句：

```
import clr
clr.AddReference("System.Windows.Forms")
clr.AddReference("System.Drawing")
```

完成後，在程式碼的末尾添加以下行以啟動應用程式：

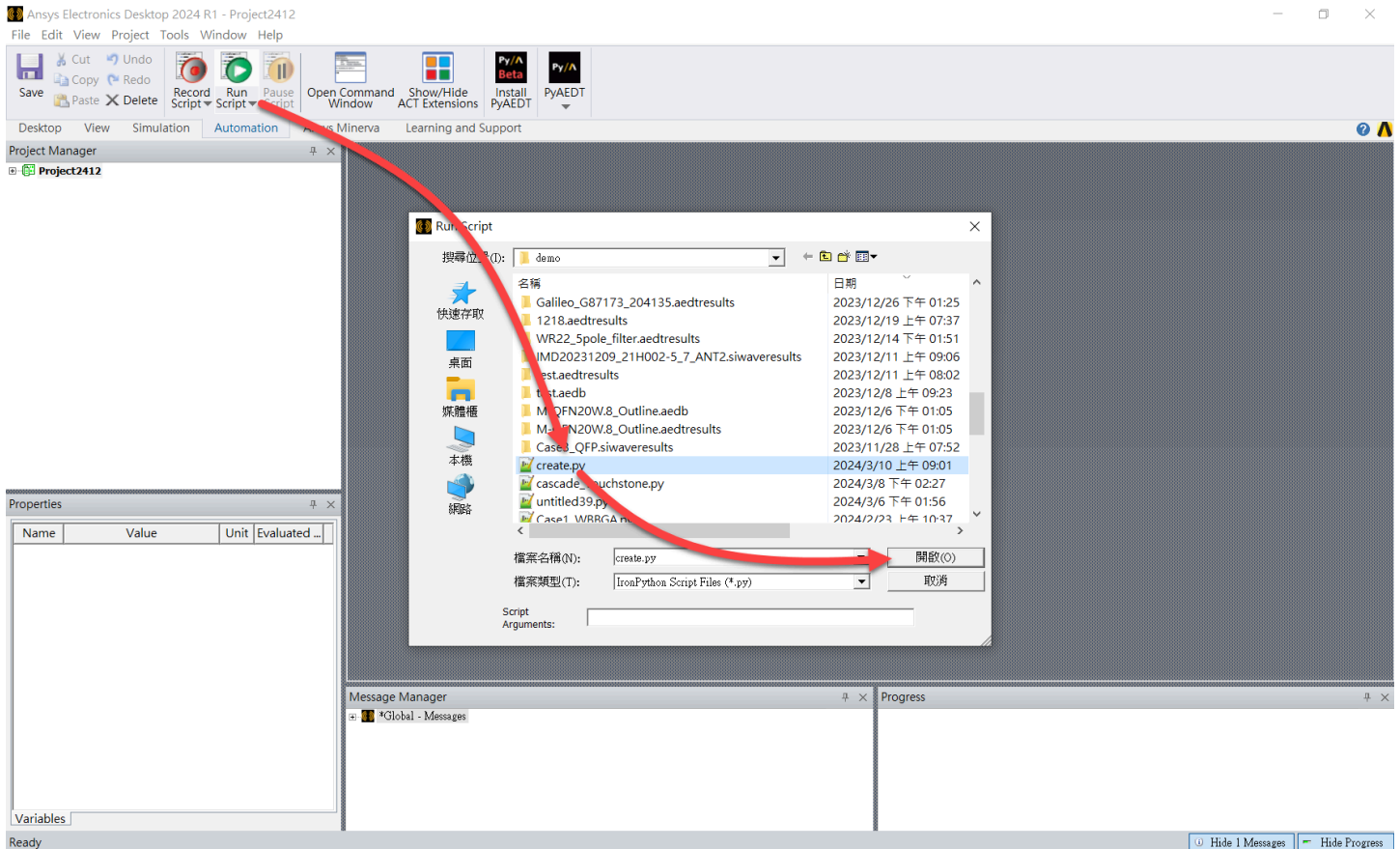
```
Application.Run(MainForm())
```

這樣做可以確保程式碼正確地導入必要的 .NET 組件並在運行時能夠建立和顯示主窗體。

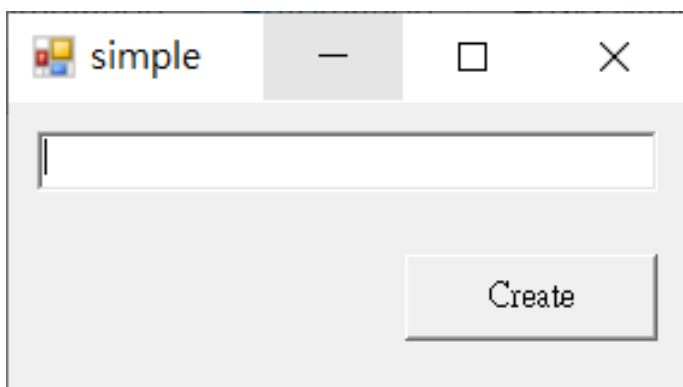
測試GUI

1. 打開腳本管理器：在 AEDT 頂部的工具列中，點擊「Run Script」按鈕（有一個 Python 腳本的圖示）。

2. **導航到腳本位置**：在彈出的對話框中，瀏覽到包含您的 Python 腳本（如 `create.py`）的文件夾位置。
3. **選擇腳本文件**：在文件列表中，選擇您想要運行的腳本。在此例中，腳本名稱是 `create.py`。
4. **運行腳本**：選擇腳本後，點擊「開啟(O)」按鈕來執行腳本。



如果順利運行了 `create.py` 腳本，一個簡單的 GUI 窗體將隨之出現。



Step 5: 加入AEDT程式碼並測試

以下是完整的函數代碼，包括了對使用者輸入的基本錯誤處理：

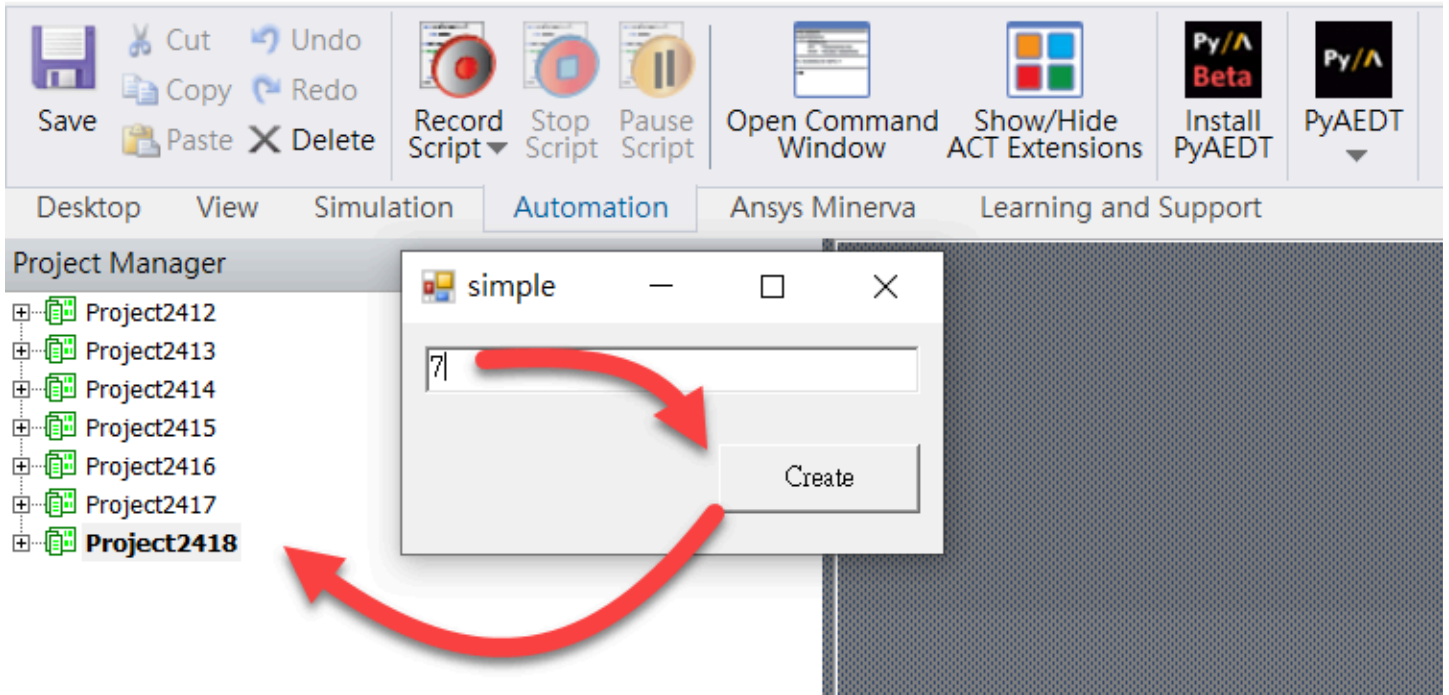
```
def Button1Click(self, sender, e):  
    try:  
        number = int(self._textBox1.Text)  
        for i in range(number):  
            oProject = oDesktop.NewProject()  
  
    except ValueError:  
        System.Windows.Forms.MessageBox.Show("Please enter a valid integer.")
```

`Button1Click` 事件處理函數將從文本框（其名稱是 `_textBox1`）中獲取一個數字，然後使用這個數字來控制創建新專案的次數。在這段代碼中，如果 `_textBox1` 中的文本不能轉換為整數，`int()` 函數將拋出一個 `ValueError` 異常。這段代碼捕捉到異常後，會顯示一個消息框提示使用者輸入有效的整數。這樣可以避免程序因為無效輸入而崩潰。

測試完整程式碼

使用這個界面，用戶可以輸入一個數字，然後點擊“**Create**”按鈕來創建相應數量的新項目。流程和結果如下：

1. **用戶輸入**：在 GUI 的文本框中，用戶輸入了希望創建的項目數量，例如數字“7”。
2. **執行操作**：點擊“**Create**”按鈕後，相關的事件處理函數被觸發，根據用戶輸入的數值執行創建項目的操作。
3. **AEDT 中新項目創建**：在 AEDT 的項目管理器中，現在顯示出新創建的項目，從“Project2412”到“Project2418”，共計 7 個新項目，這與用戶在文本框中輸入的數值相匹配。



這個過程展示了如何使用自定義 Python 腳本與 AEDT API 進行交互，進一步實現工程自動化和提高效率。這種方法可以靈活地根據用戶的需求定製 AEDT 的功能，使複雜或重複的任務變得更加簡單。

總結

結合 AEDT 腳本和自定義 GUI 的方法大大增強了工具的用戶友好性和互動性，使得非編程背景的工程師也能輕鬆實現複雜操作，從而提高了 AEDT 工具的可接近性和效率。