HMIWorks

TouchPAD 系列開發軟體

繁體中文使用手冊版本 1.5

ICP DAS CO., LTD.
service@icpdas.com

著作權

授權範圍

使用者僅能將本軟體與本公司 TouchPAD 搭配使用。

擔保

在相關法律所允許之最大範圍內，本公司不承擔任何瑕疵擔保責任與條件，不論其為明示或默示者，其中包括但不限於適售性、適合某特定用途以及不侵害他人權益之默示擔保責任。

在相關法律所允許之最大範圍內，本公司對於使用者因使用或不能使用本軟體而遭受之特別、附隨、間接或衍生性損害（包括，但不限於營業利益之損失、營業中斷、營業資訊之損失或其他金錢損失）不負任何損害賠償責任。此項規定不因使用者事先告知本公司該損害發生之可能性而有所不同。
除有其他約定之外，本產品並無容錯設計 (Fault-tolerant)，本產品不應使用於可能導致死亡或人身傷害或環境傷害之高風險性作業，例如核能操作、航空導航通訊控制、維生器材或武器系統等。

商標

本使用手冊提及之所有公司商標、商標名稱及產品名稱分別屬於該商標或名稱的擁有者所有。

支援

如有任何問題歡迎聯繫我們，我們將會為您提供完善的咨詢服務。
Email: service@icpdas.com
泓格科技關於 TouchPAD 網站：
https://www.icpdas.com/tw/product/guide+Panel_Products+TouchPAD+TPD_Series

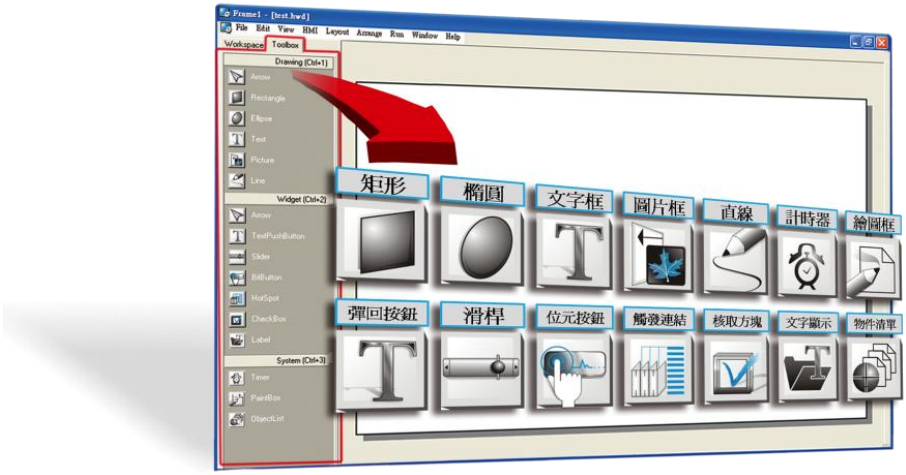
目錄

1. 簡介

HMIWorks 是泓格科技提供 TouchPAD 系列產品的免費開發軟體，支援大量的控制項，可縮短開發時間，內建可擴充的圖形庫，具有直覺式開發、支援 C 語言和階梯圖 (Ladder Diagram) 程式開發、完全整合 I/O 模組... 等多樣特色，搭配泓格 TouchPAD 系列 HMI 裝置設備，能提供兼具精密設計與成本效益的觸控解決方案。

支援大量的控制項，縮短開發時間

在 HMIWorks 中，支援各式各樣的控制項，包含矩形工具、橢圓工具、文字工具、圖片工具、直線工具、文字彈回式按鈕、滑桿、位元式按鈕、觸發連結框、核取方塊、標籤工具、計時器、繪圖框、物件清單。這些控制項提供大部分常用的功能，如繪製圖形、事件函式處理、時程控制管理等。綜上所述，使用如上工具可以有效地縮短開發時間。



C 語言和階梯圖 (Ladder Diagram) 程式開發

65536 色螢幕，明亮而清晰

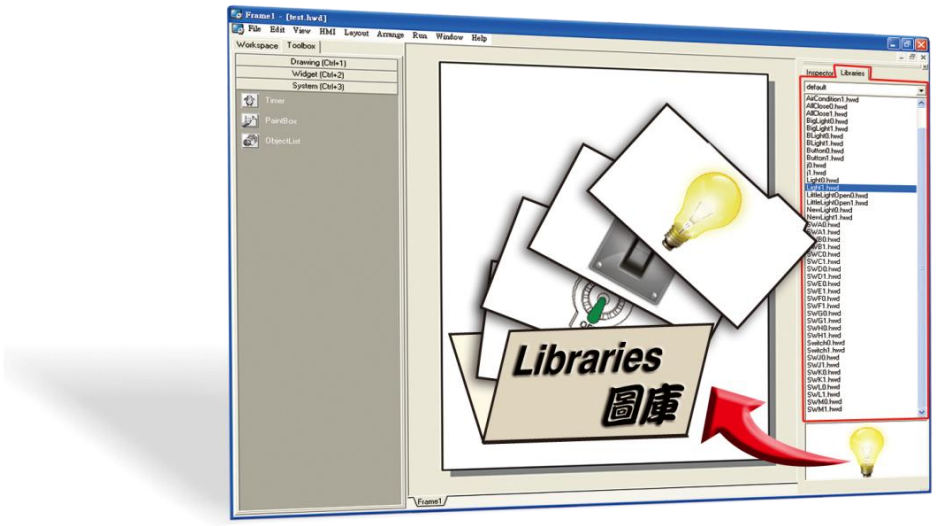
目前共有 2.8 吋、3.5 吋、4.3 吋、7 吋的觸控液晶螢幕，支援的解析度則有 240 x 320 、 480 x 272 、800 x 480。未來，泓格科技將提供更多的選擇。

直覺式開發

直覺式開發讓使用者可以專注於想要做什麼，而不是應該做什麼。把程式設計的瑣碎細節丟掉，只專注於設計本身，讓整個專案更容易完成。

內建可擴充的圖形庫

HMIWorks 支援一些簡單的圖形處理，並且針對一些常見的應用，內建大量的圖形庫。除此之外，使用者也可以新增由其他軟體所編輯的圖形到圖庫中，擴充應用更容易。(圖片格式支援 jpg、bmp、emf、wmf)



一次拖放，完全整合 I/O 模組

目前泓格科技支援各種 I/O 模組，如 ET-7000/PET-7000 系列的 Modbus TCP 模組、M-7000 系列的 Modbus RTU 模組、 I-7000 系列的 DCON 模組及其他廠家的 Modbus TCP/RTU 模組。 可以預期的，未來 HMIWorks 會支援更多的模組。

1.1 HMIWorks 的特色

- 完全免費 (僅供泓格科技 TouchPAD 使用)
- 兩種方法開發程式: 標準 C 語言和階梯圖(Ladder)
- 豐富的元件
- 大量的範例縮短開發時程
- 支援模組搜尋功能
- 詳細的錯誤訊息
- 簡單的編譯上傳步驟
- 建置應用頁面，編譯後會自動產生對應的程式碼
- 提供多頁面設計
- 將元件操控封裝成簡單的 API，不需要撰寫困難的程式
- 容易使用和學習的開發整合環境，可在短時間內提高生產力
- 資料交換功能,解決上位機與下位機之間不同通訊協定的問題

1.2 支援泓格產品

下表 HMIWorks 軟體所支援的泓格產品:

TPD 高速系列模組	TPD-280-H、TPD-280U-H、TPD-283-H、TPD-280-M1、TPD-280-M2、TPD-280-M3、, TPD-283-M1、TPD-283-M2、TPD-283-M3、TPD-283U-M1、TPD-283U-M2、 TPD-283U-M3、TPD-430-H、TPD-433-H、TPD-433F-H、TPD-432F-H、TPD-433-M2、 TPD-703、TPD-703-64
VPD 高速系列模組	VPD-130-H、VPD-130N-H、VPD-132-H、VPD-132N-H、VPD-133-H、VPD-133N-H、VPD-130-H2、VPD-130N-H2、VPD-133-H2、VPD-133N-H2、VPD-142-H、VPD-142N-H、VPD-143-H、VPD-143N-H、VPD-173N、VPD-173N-64、 VPD-173X、VPD-173X-64

下表的產品已經停產: 最後支援版本為 HMIWorks v2.10.61

TPD 系列模組	TPD-280、TPD-280U、TPD-283、TPD-283U、TPD-430、TPD-430-EU、TPD-433、 TPD-433-EU、TPD-432F、TPD-433F
VPD 系列模組	VPD-130、VPD-130N、VPD-132、VPD-132N、VPD-133、VPD-133N、VPD-142、 VPD-142N、VPD-143、VPD-143N

2. 軟體安裝

本章節將詳細介紹如何取得安裝執行檔、安裝及移除應用程式步驟…等資訊。

2.1 取得 HMIWorks 應用程式

注意: TPD-280/283/280U/238U、TPD-430/433/432F/433F、VPD-130(N)/132(N)/133(N) 及 VPD-142(N)/143(N)等停產模組最後支援版本為 HMIWorks v2.10.61，HMIWorks v2.30.xx 之後的版本已不再支援上述產品。

HMIWorks 應用程式安裝執行檔，可從泓格的網站中下載，詳細位置如下:

https://www.icpdas.com/en/download/show.php?num=944

下表 HMIWorks 所支援的 Microsoft Windows 作業系統:

32-bit(x86)	64-bit(x64)
Microsoft Windows 7	Microsoft Windows 7
Microsoft Windows 2008	Microsoft Windows 2008
Microsoft Windows 8	Microsoft Windows 8
Microsoft Windows 2012	Microsoft Windows 2012
Microsoft Windows 10	Microsoft Windows 10

2.2 安裝 HMIWorks 應用程式

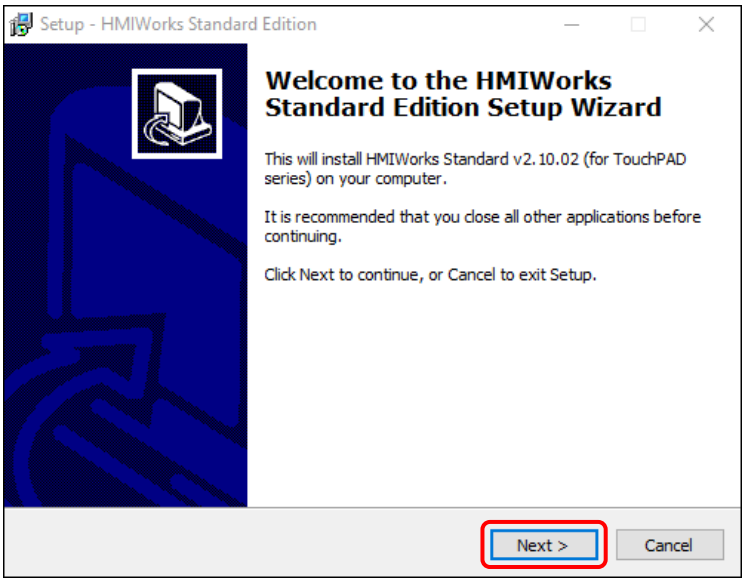
下面將在 64-bit Windows 10 作業系統安裝 HMIWorks 應用程式，詳細操作步驟如下:

步驟 1: 雙擊 **“HMIWorks_STD_vxxx_setup.exe”** 應用安裝程式執行檔。

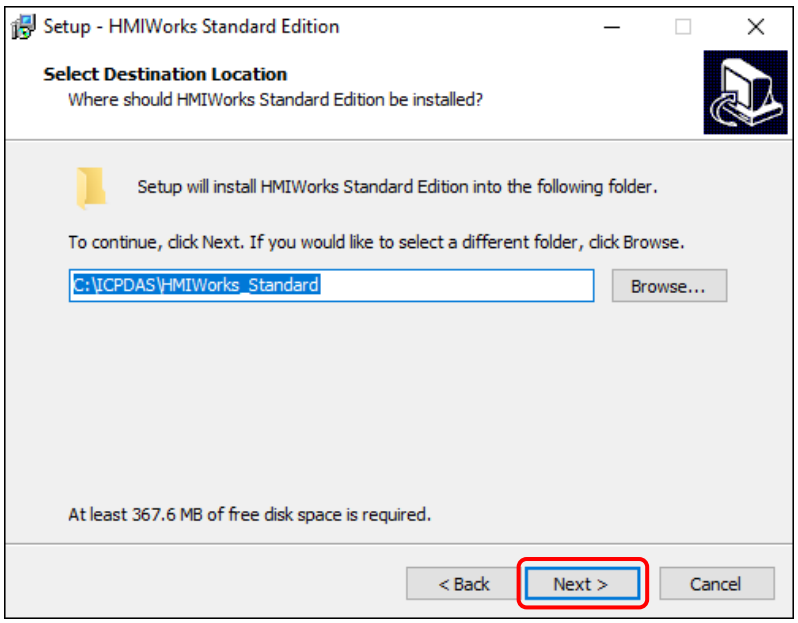
注意

某些作業系統中 (如: Windows 10、Windows 7…等) 在安裝過程中，將會跳出提示對話框來要求您確認您安裝的設備軟體，如左圖所示。請按 **“是(Y)”** 繼續下一步驟。

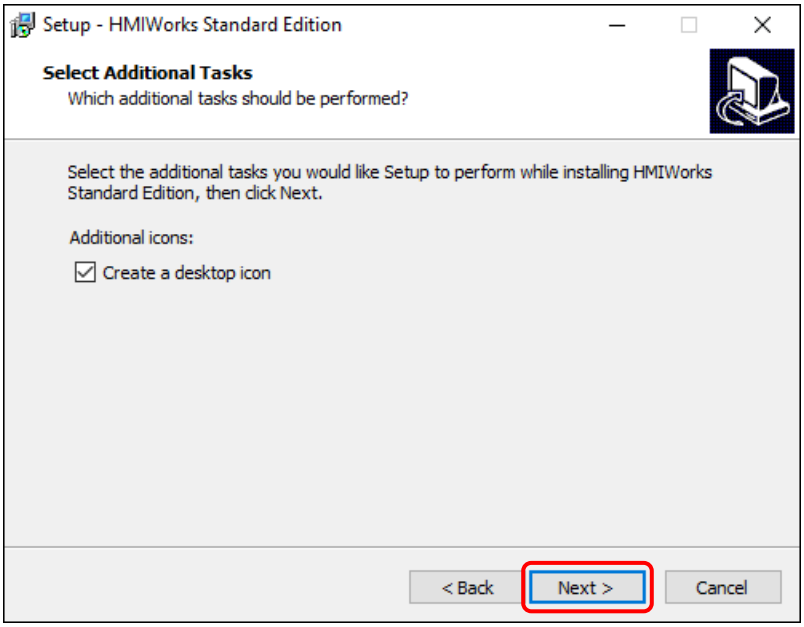
步驟 2: 單擊 **“Next>”** 到下一個安裝畫面。



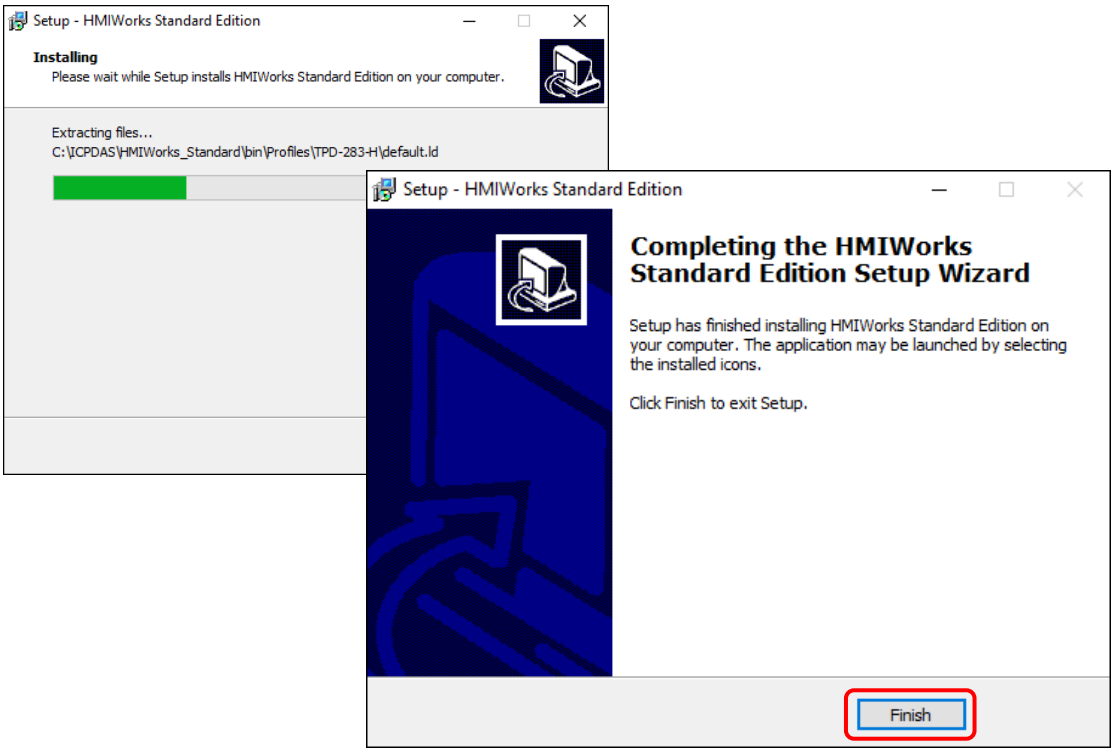
步驟 3: 選擇安裝目錄，**預設安裝路徑 C:\ICPDAS\HMIWorks_Standard**。確認後按 **“Next >”** 到下一個安裝畫面。



步驟 4: 勾選 **“Create a desktop icon”** 項目，在桌面建立捷徑圖示。按 **“Next >”** 到下一個安裝畫面。



步驟 5: 進行安裝中，完成後按下 **“Finish”** 結束安裝程序。



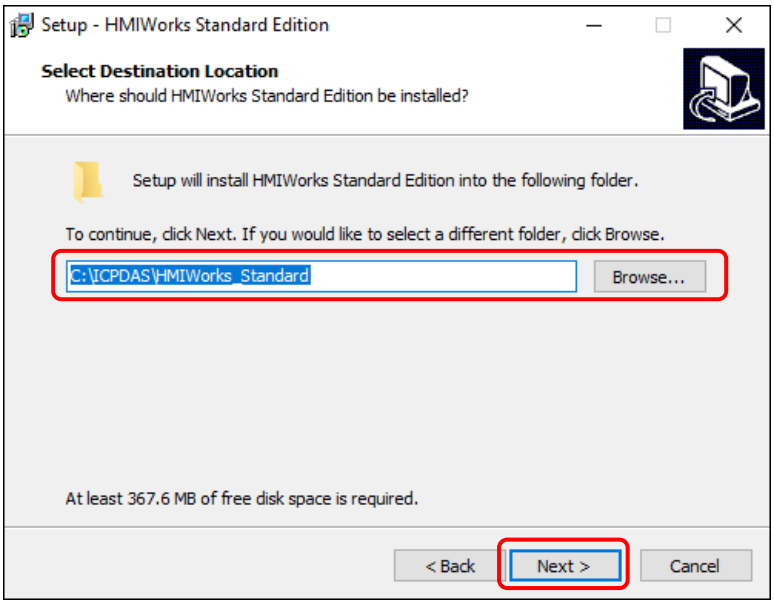
步驟 6: 一旦 HMIWorks 應用程式安裝完成後，再雙擊 **“HMIWorks_STD_vxxx_Update_xx.exe”** 執行檔來開始安裝 HMIWorks 更新程式。

HMIWorks_STD_
v2.10_Update_32.exe

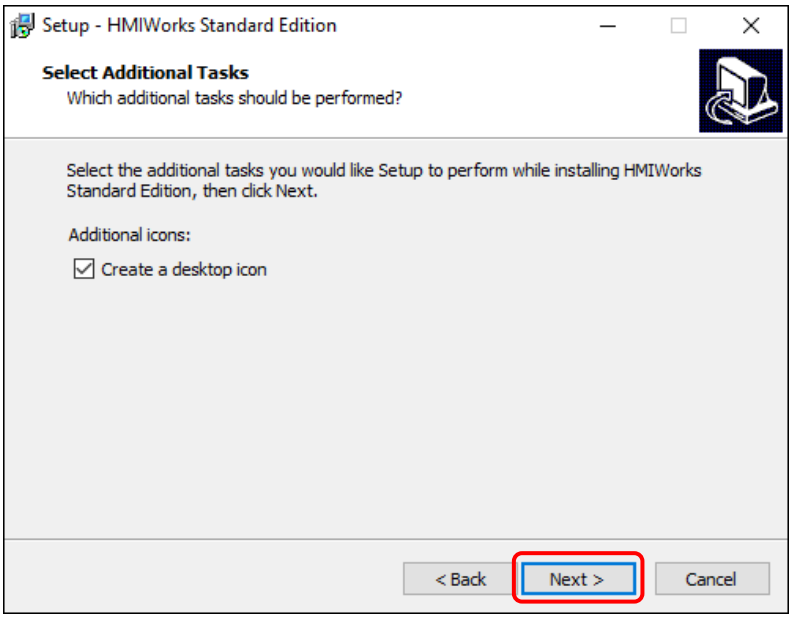
注意

某些作業系統中 (如: Windows 10、Windows 7…等) 在安裝過程中，將會跳出提示對話框來要求您確認您安裝的設備軟體，如右圖所示。請按 **“是(Y)”** 繼續下一步驟。

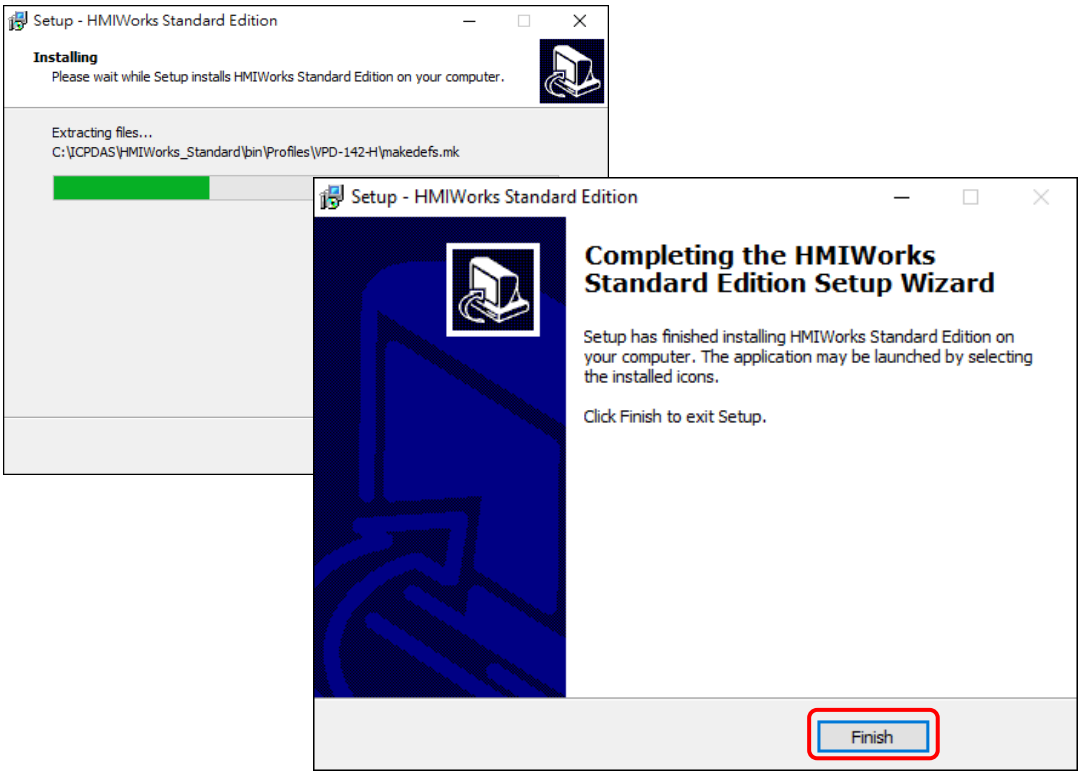
步驟 7: 選擇安裝目錄，**預設安裝路徑 C:\ICPDAS\HMIWorks_Standard**。確認後按 **“Next >”** 到下一個安裝畫面。



步驟 8: 勾選 **“Create a desktop icon”** 項目，在桌面建立捷徑圖示。按 **“Next >”** 到下一個安裝畫面。



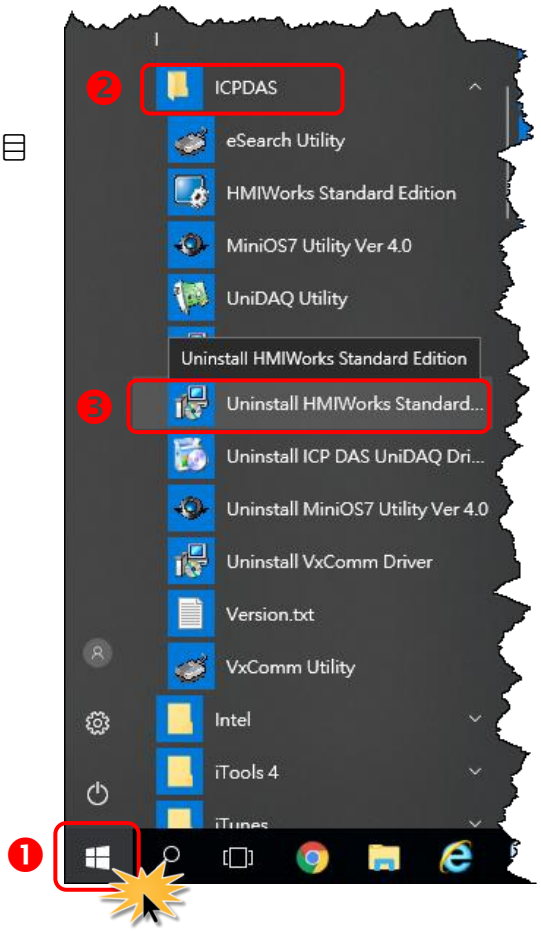
步驟 9: 進行安裝中，完成後按下 **“Finish”** 結束安裝程序。



2.3 移除 HMIWorks 應用程式

泓格應用程式包括反安裝工具來協助您從電腦上移除軟體，如果您想要移除軟體請完成下列的步驟來執行反安裝工具，下面將以 64-bit Windows 10 作業系統為操作範例。

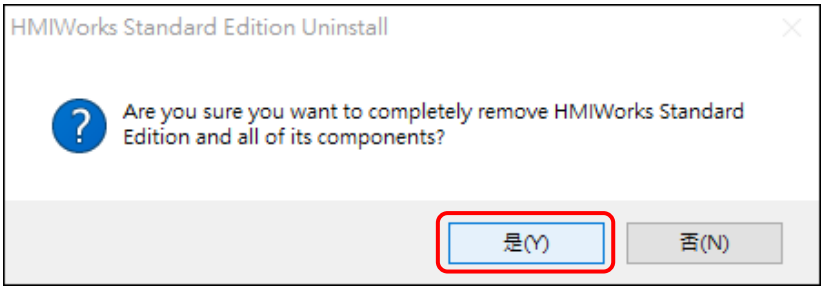
步驟 1: 單擊 **Windows “開始” 功能表**
→ 單擊 **“ICP DAS”**
→ 單擊 **“Uninstall HMIWorks Standard Edition”** 項目來移除應用程式。



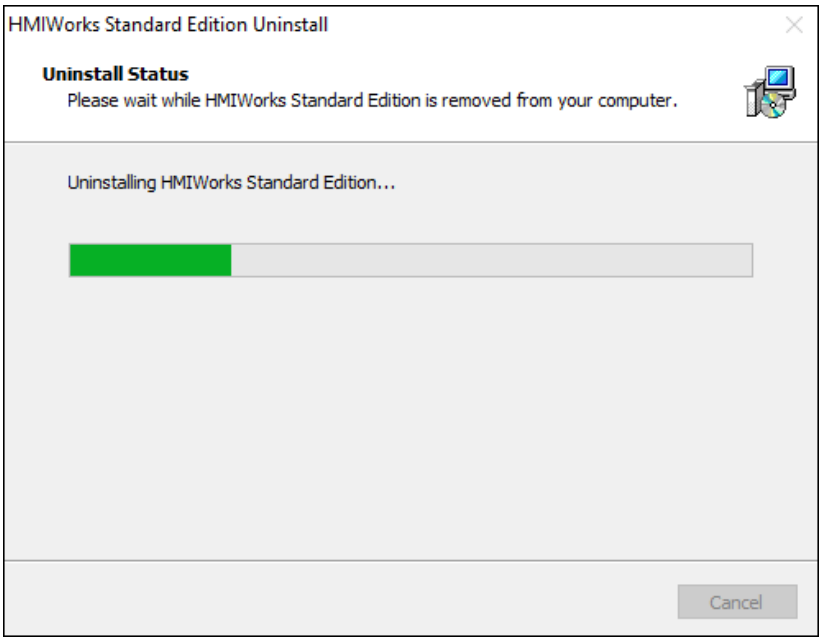
⚠注意

某些作業系統中 (如: Windows 10、Windows 7...等) 在安裝過程中，將會跳出提示對話框來要求您確認您安裝的設備軟體，如右圖所示。請按 **“是(Y)”** 繼續下一步驟。

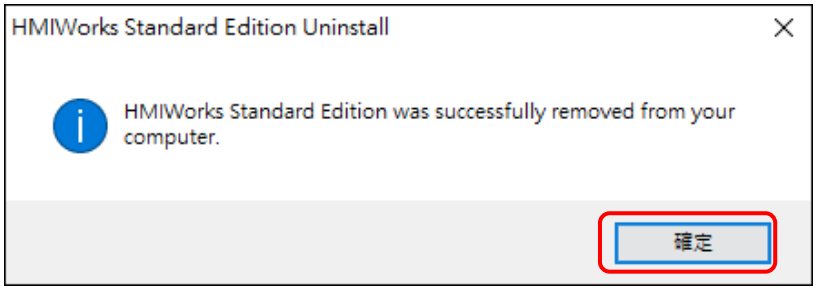
步驟 2: 將會跳出一個對話框來詢問是否確定要移除此軟體應用程式，請按下 **“是(Y)”** 開始執行反安裝。



步驟 3: 進行移除中。



步驟 4: 再按下 **“確定”** 後，確認已成功完成移除。

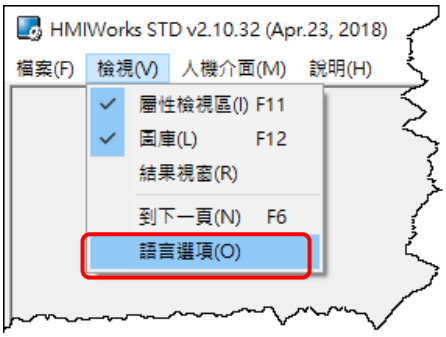


3. 開發軟體 HMIWorks

HMIWorks 應用程式安裝完成後，將在 Windows 桌面建立捷徑圖示。請雙擊 HMIWorks 捷徑圖示來啟動 HMIWorks。當您啟動 HMIWorks 時，就會出現以下歡迎對話框畫面，本章節將詳細介紹 HMIWorks 開發環境各項配置功能。

注意

HMIWorks 介面預設是 **“English”**，在下面各章節中我們將以 **“繁體中文”** 介面來介紹 HMIWorks 各項配置功能，您可從 **“檢視(V)”** 功能選單中，點選 **“語言選項(O)”** 來變更操作介面語言。



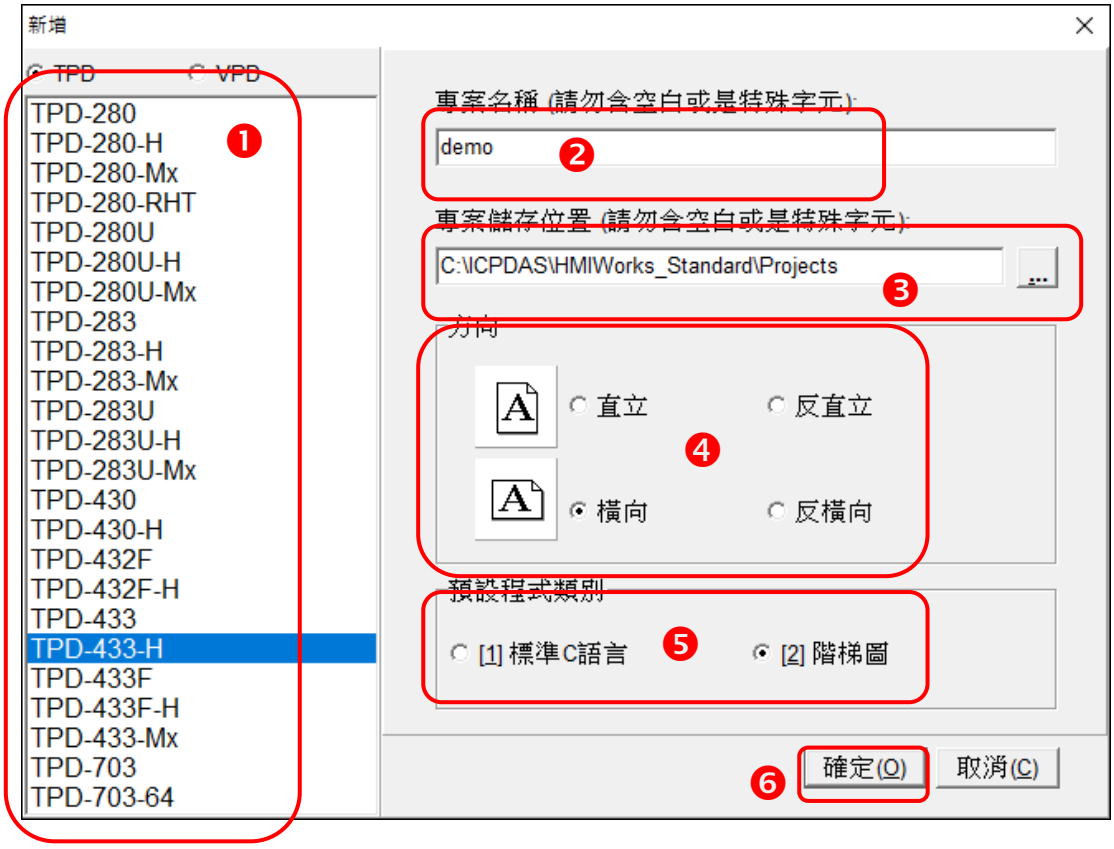
3.1 HMIWorks 的環境

在介紹 HMIWorks 開發環境之前，首先請新增一個專案，詳細操作如下:

步驟 1: 按下**“新增專案”** 項目來建立一個新專案。
或是關閉此歡迎對話框，從 **“檔案(F)”** 功能選單中，點選 **“新增...(N)”**來建新專案。

步驟 2: 在 **“新增”** 配置視窗中，設定新專案的參數如下:

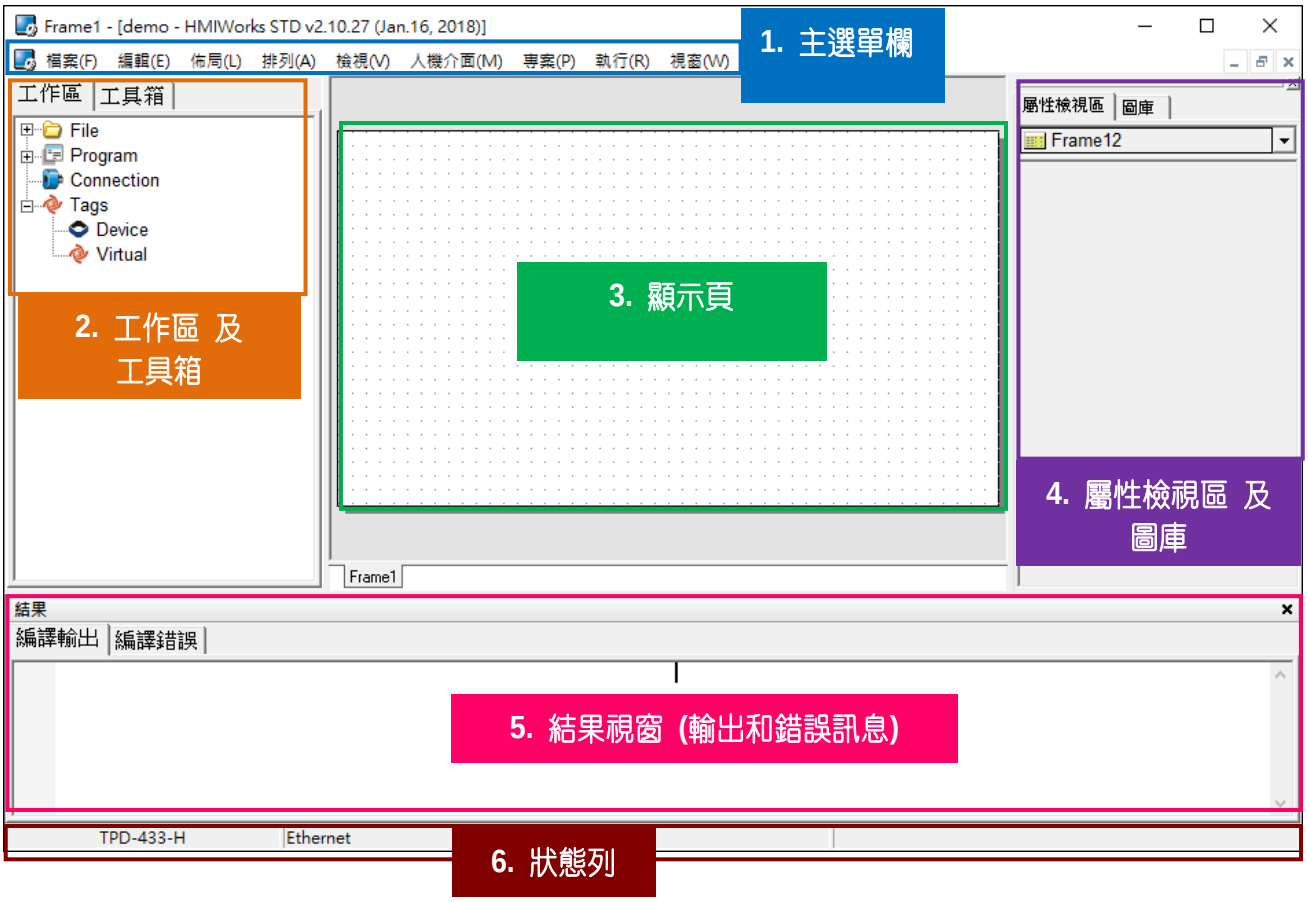
1. 選擇 TouchPAD 模組名稱 (此範例為 TPD-433-H 模組)。
2. 輸入專案名稱 (如: demo)。
3. 選擇專案儲存位置 (預設路徑)。
4. 選擇版面配置方向 (如: 橫向)。
5. 選擇預設程式類別 (如: [2] 階梯圖)。
6. 按 **“確定(O)”** 按鈕來完成新專案建立。



注意

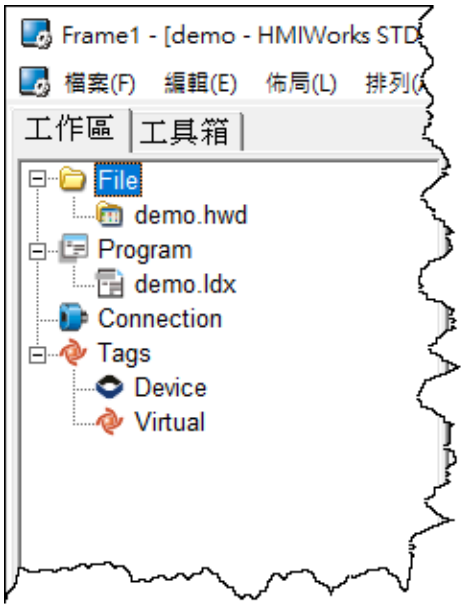
專案名稱必須由**字母**、**數字**和**下底線(_)** 所構成，且**第一個字元不能是數字**。此外，專案名稱有長度的限制，加上專案所在位置的路徑，建議長度是 100 個字元。

➢ HMIWorks 開發環境如下圖所示:



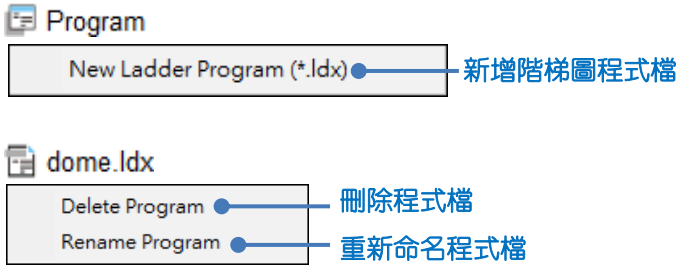
項目		說明
1.	主選單欄	此為 HMIWorks 的主功能選單，詳細說明請參考 第 3.5 節 選單。
2.	工作區	詳細工作區說明，請參考下一頁。
	工具箱	詳細工具箱說明，請參考 第 3.4 節 顯示頁和元件。
3.	顯示頁(Frame)	在此頁面中設計建立您的應用程式，詳細說明請參考 第 3.4 節 顯示頁和元件 。
4.	屬性檢視區	詳細元件屬性檢視區及圖庫說明，請參考 第 3.4 節 顯示頁和元件。
	圖庫	
5.	結果視窗 (輸出和錯誤訊息)	此視窗將顯示執行編譯和下載時輸出及錯誤的狀態訊息。
6.	狀態列	顯示 TouchPAD 裝置的狀態。

➢ 在“**工作區**”允許您配置 (新增、刪除、修改) 專案檔、程式檔、設備連線及關連標籤。您可在 **“File”** 按右鍵來開啟功能選單進行配置，如下圖所示:

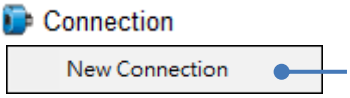


File	顯示和配置專案檔。

Program	顯示和配置程式檔。

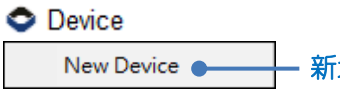


Connection	顯示和配置 Modbus TCP/RTU 設備的 TCP/IP 及序列連線。

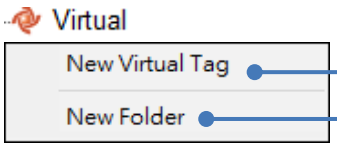


詳細關於配置設備的 TCP/IP 及序列連線，請參考 第 4.3 節 TouchPAD 與 I/O 模組整合。

Tags	顯示和配置虛擬標籤及 TCP/RTU 設備。



詳細說明請參考 第 3.3.6 節 使用工具和標籤產生關連 及 第 3.3.7 節使用者定義的 I/O 模組。



詳細說明請參考 第 3.3 節階梯圖設計家 (Ladder Designer)。

3.2 TouchPAD 的選項

3.2.1 語言選項

HMIWorks 介面**預設為 “English”**，可變更語言為 **“繁體中文”**，變更方式如下:

步驟 1: 從 **“檢視(V)”** 功能選單中，點選 **“語言選項(O)”** 項目來開啟 “語言選項” 配置視窗。

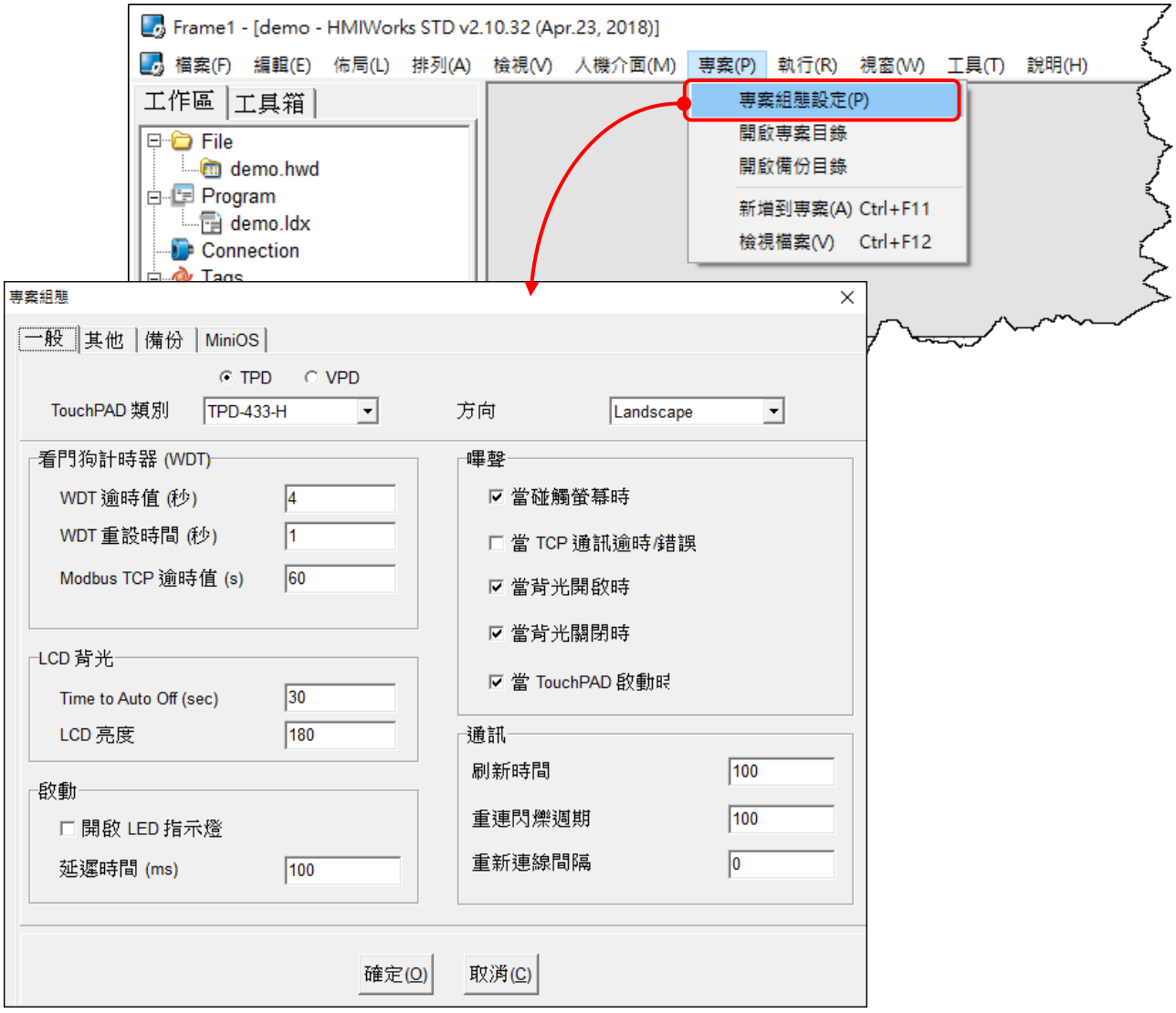
步驟 2: 從 **“語言對應檔”** 下拉式選單中，選擇 **“繁體中文 Traditional Chinese”** 項目，再按下 **“確定”** 來完成設定，此時 HMIWorks 將變更為 **“繁體中文”** 介面。

3.2.2 專案組態設定

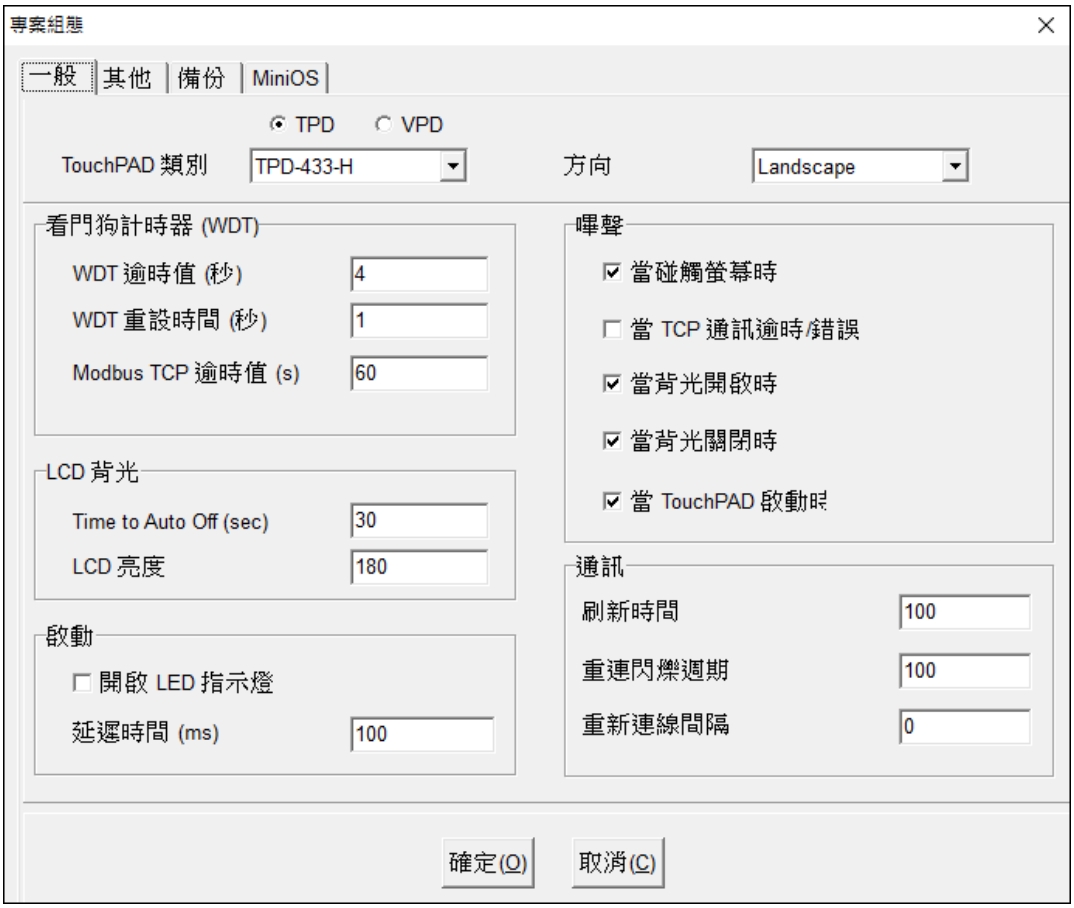
“專案組態設定”提供您配置看門狗計時器 (Watchdog)、LCD 背光、嗶聲、通訊及備分…等各項設定，詳細說明如下。

開啟專案組態配置

從**“專案(P)”**功能選單中，點選**“專案組態設定(P)”**項目來開啟“專案組態”配置視窗。



一般

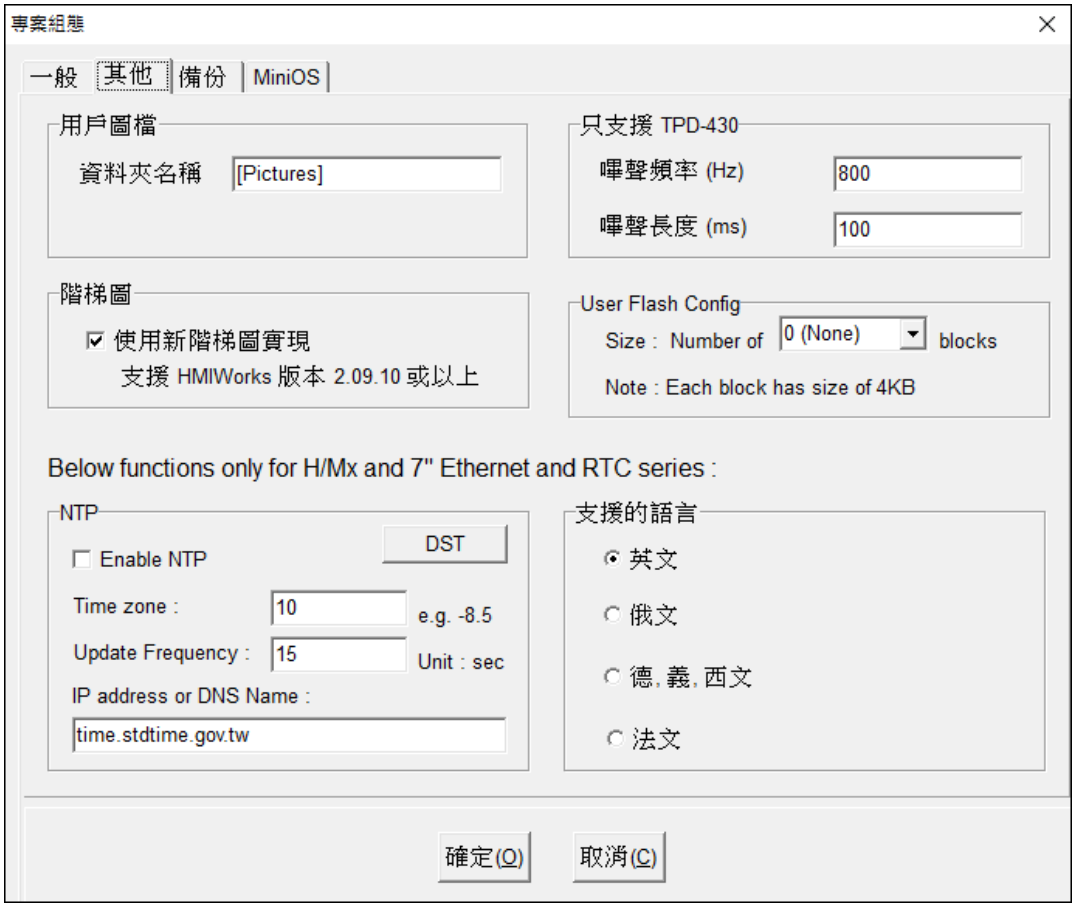


“一般” 設定項目詳細說明如下表:

項目		說明
TouchPAD 類別		在更改這兩個選項之後，HMIWorks 會自動調整每個頁面及元件的大小，以維持它們彼此之間的相對位置。 **註:** 恆定文字框 (Text) 元件是不調整的。
方向		
看門狗計時器 (WDT)	**WDT 逾時值 (秒)**	逾時值 (單位: 秒) 以重啟 TouchPAD。 有效範圍: 1 ~ 50 秒，其它值表示取消。 **註:** 建議開啟 WDT 功能。
	WDT 重設時間 (秒)	WDT 重設週期 (單位: 秒) 以避免看門狗計時器重啟 TouchPAD (建議: 逾時值的 25%)
	Modbus TCP 逾時值 (秒)	Modbus TCP 的逾時值 (單位: 秒) 以重啟 TouchPAD。 有效範圍: 10 ~ 10,000 秒，其它值表示取消。

項目		說明
LCD 背光	**Time to Auto Off (sec)**	指定當觸控螢幕閒置時自動關閉 LCD 背光的時間。 (單位: 秒; 預設: 30 秒) **註:** LCD 有壽命限制，建議開啟 LCD Auto oFF 功能。不僅可延長 LCD 壽命，也節約用電。
	LCD 亮度	指定液晶顯示器的亮度 (預設: 180) 有效範圍: 0 ~ 255。0: 最暗; 255: 最亮。
啟動	**開啟 LED 指示燈**	當 TouchPAD 啟動時，開啟 LED 指示燈。
	延遲時間 (ms)	指定延遲 TouchPAD 啟動的時間。 (單位: 毫秒; 預設: 100 ms)
嗶聲	**當碰觸螢幕時**	當螢幕被碰觸時，發出嗶一聲。 如果勾選「當碰觸螢幕時」，函式 **hmi_PlaySong** 就會變得無效。
	當 TCP 通訊逾時/錯誤時	當 TCP 通訊逾時或是有錯誤時，發出一嗶聲。
	當背光開啟時	當 LCD 的背光開啟時，發出一嗶聲。
	當背光關閉時	當 LCD 的背光關閉時，發出一嗶聲。
	當 TouchPAD 啟動時	當 TouchPAD 啟動時，發出一嗶聲。
通訊	**刷新時間**	I/O 的時間間隔和階梯圖的掃描時間。 (預設: 100 ms)
	重連閃爍週期	用於 TouchPAD 作為 Modbus TCP Master (主站) 通訊時，「重連閃爍週期」定義了「ERROR (錯誤)」標籤在重新連線時閃爍的週期，該標籤可以在工作區 (Workspace) 中的 device 內找到。
	重新連線間隔	兩組重新連線嘗試的間隔 (每組重連 7 次)
確定		按下此按鈕來儲存新設定值並關閉視窗。
取消		按下此按鈕將會停止並關閉視窗。

其他

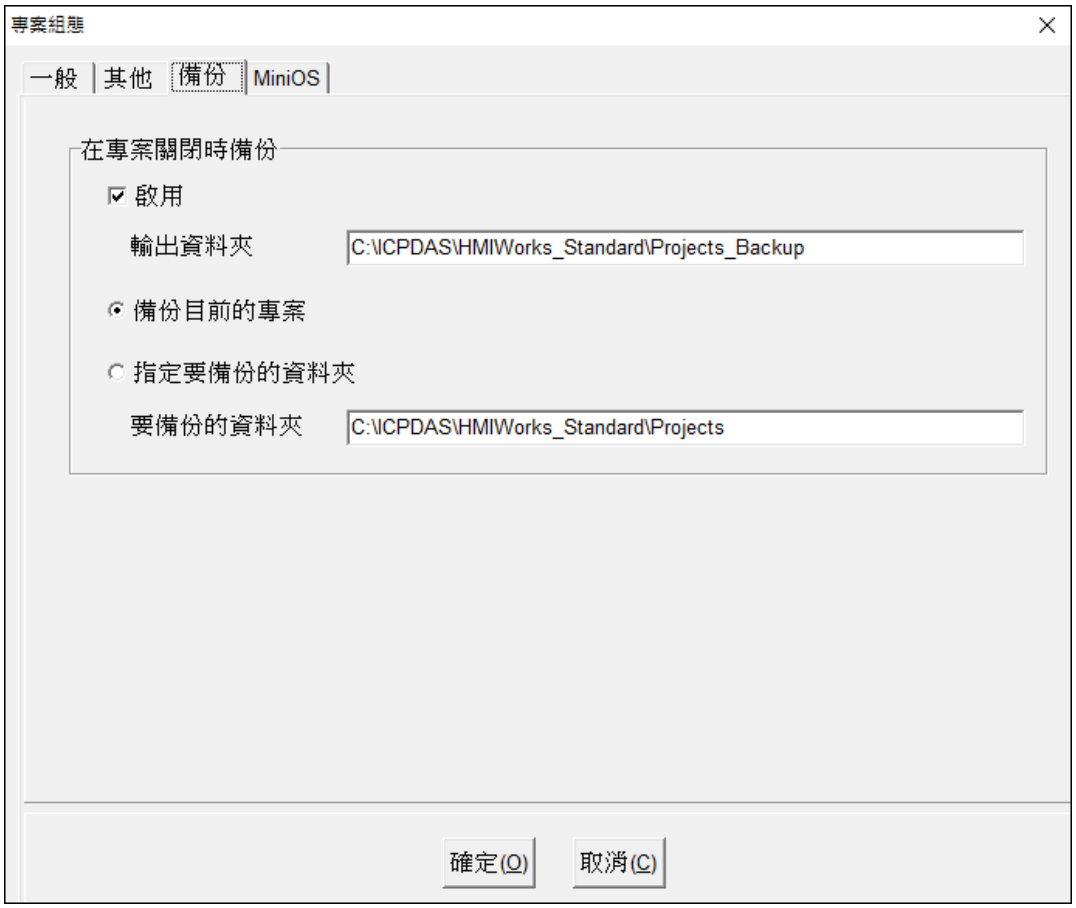


“其他” 設定項目詳細說明如下表:

項目		說明
用戶圖檔	**資料夾名稱**	於專案目錄下，儲存用戶圖檔的資料夾名稱 (相對路徑)。
階梯圖	**使用新階梯圖實現** **支援 HMIWorks 版本 2.09.10 或以上**	如您原來的專案是使用 HMIWorks v2.09.09 或更早版本的階梯圖程序所建立的，請取消勾選此項目來關閉 **新的階梯圖模式:** Coil-Set 及 Coil-Reset 可改變 Coil 狀態並且 Lock 住(工業標準)直到下一個 Coil-Reset 或 Coil-Set。 **舊的階梯圖模式:** 無 Lock 特色。

項目		說明
只支援 TPD-430	**嗶聲頻率 (Hz)**	指定嗶聲的頻率 (預設: 800 Hz)。 有效範圍: 30 ~ 4,000 Hz。
	嗶聲長度 (ms)	指定嗶聲的長度 (預設: 25 ms) 。
User Flash Config	**Size**	指定您需要的 Flash 大小 (預設: 0) 通常 Flash 用於存儲專案程式。使用者可以將部份 Flash 空間用於其它目的，如: 做資料記錄功能。 **註 1:** 此功能僅供 C 程式語言客戶使用，需搭配 API 函式。 **註 2:** 此項設定會減少可存儲專案檔的大小，且每個記憶體區塊具有 100,000 次寫入限制。
Below functions only for H/Mx and 7" Ethernet and RTC Series: **以下功能僅適用於 TouchPAD 的 H、Mx、乙太網路 7 吋及 RTC 系列**		
NTP	**Enable NTP**	啟動 NTP 後，可以自動從 NTP 服務器取得時間。
	Time Zone	依據您的位置設定時區。
	Update Frequency	設定更新率 (單位: 秒)。
	IP address or DNS Name	設定 NTP 服務器。
支援的語言		內建多語系功能，包含英文、俄文、德/義/西文 (歐洲語系) 及法文。預設是英文，如果要使用英語以外的語言，請參考 FAQ: 如何使用 HMIWorks 內建字型在 TouchPAD 上顯示多語言文字? 如所需的語言不在此選項中，如:CJK (中文/日文/韓文)...等，您可以安裝 ebFonts 來支援更多語言，詳細請參 FAQ: 如何在 TouchPAD 上使用 ebFont 顯示多語系文字?
確定		按下此按鈕來儲存新設定值並關閉視窗。
取消		按下此按鈕將會停止並關閉視窗。

備份



“備份” 設定項目詳細說明如下表:

項目		說明
在專案關閉時備份	**啟用**	HMIWorks 可以在專案結束時備份檔案。備份的檔案會壓縮成 **.7z** 的格式。
	輸出資料夾	放置備份之壓縮檔的地方
	備份目前的專案	-
	指定要備份的資料夾	要備份的資料夾。用分號 (;) 區隔不同路徑。
確定		按下此按鈕來儲存新設定值並關閉視窗。
取消		按下此按鈕將會停止並關閉視窗。

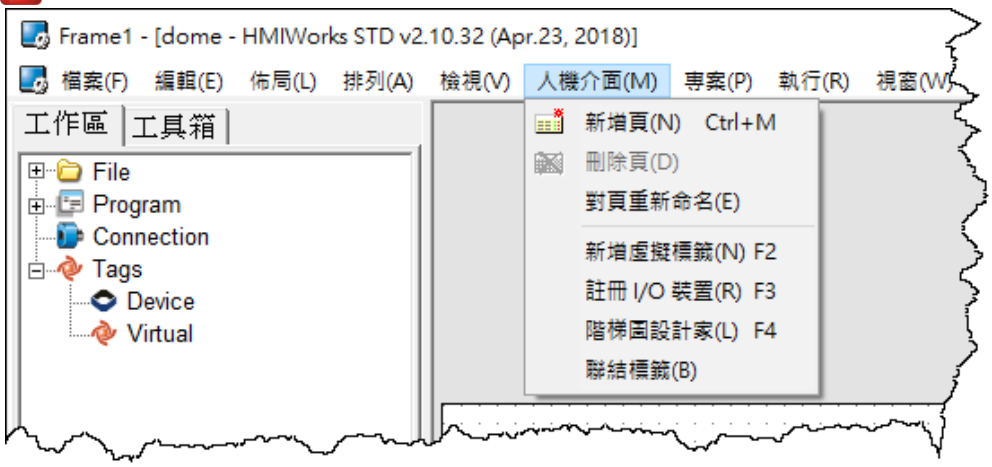
3.3 階梯圖設計家 (Ladder Designer)

階梯圖設計是 HMIWorks 所提供的一個重要功能。階梯圖邏輯由下列步驟定義：

1. 一幅階梯圖由數個階 (Rung) 所構成。
2. 每一階都可以對應到一個繼電器 (Relay) 電路。
3. 在每次掃描中，依次處理每一階。

接下來，點選 **“人機介面(M)”** 功能選單來開發階梯圖。

⚠ **注意:** 在 **“工作區”** 中，使用者可以管理階梯圖設計。



“人機介面(M)” 設定項目詳細說明如下表:

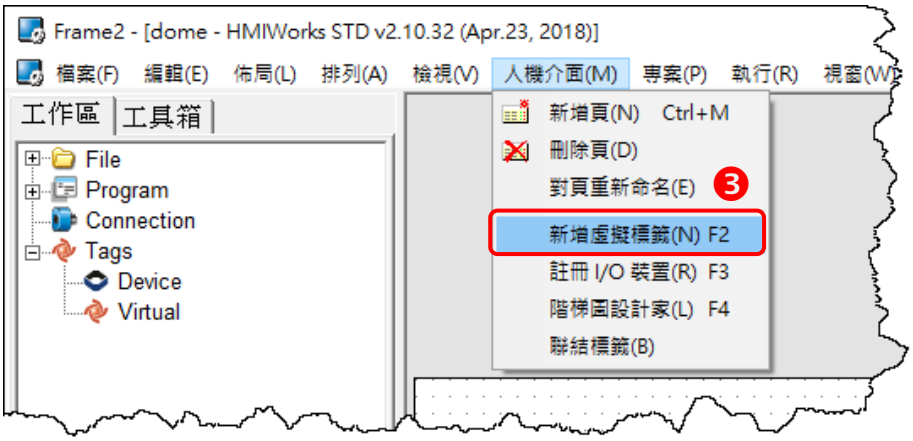
項目	快捷鍵	說明
新增頁(N)	**Ctrl + M**	新增一個新的顯示頁。
刪除頁(D)		刪除一個顯示頁。
對頁重新命名(E)		重新命名顯示頁。
新增虛擬標籤(N)	**F2**	定義階梯圖設計家使用的標籤。
註冊 I/O 裝置(R)	**F3**	使用同網路上由泓格科技提供的模組。
階梯圖設計家(L)	**F4**	設計階梯圖。
聯結標籤(B)		詳細說明請參考第 3.3.8 節 資料交換。

3.3.1 開始設計

步驟 1: 執行 **HMIWork_Standard.exe**來建立一個新專案。
步驟 2: 在 **“新增”** 配置視窗中，設定新專案的參數

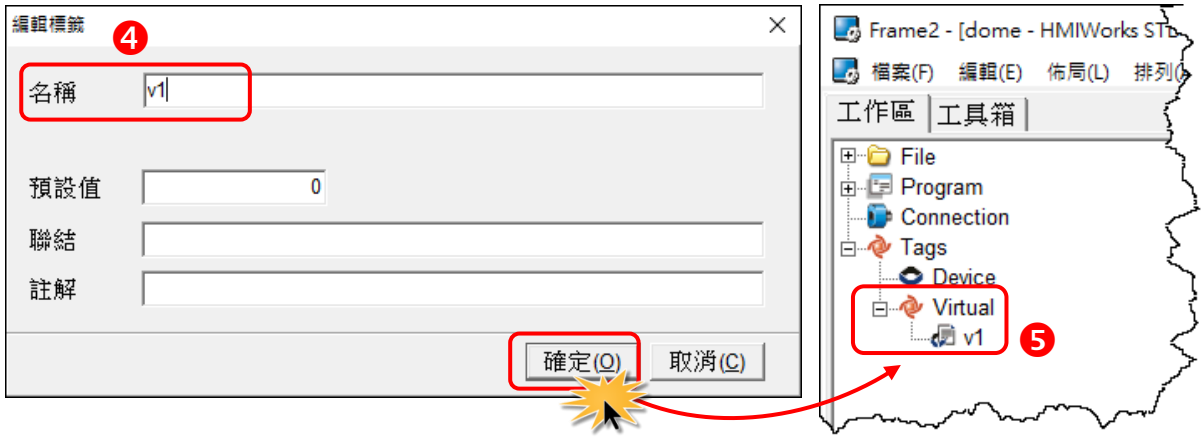
↱ **步驟 1 ~ 2** 詳細操作，請參考 第 3.1 節 HMIWorks 的環境。

下面我們將新增一個在階梯圖設計家中會使用的標籤。
步驟 3: 從 **“人機介面(M)”** 功能選單中，點選 **“新增虛擬標籤(N)”** 項目來開啟 “編輯標籤” 配置視窗。



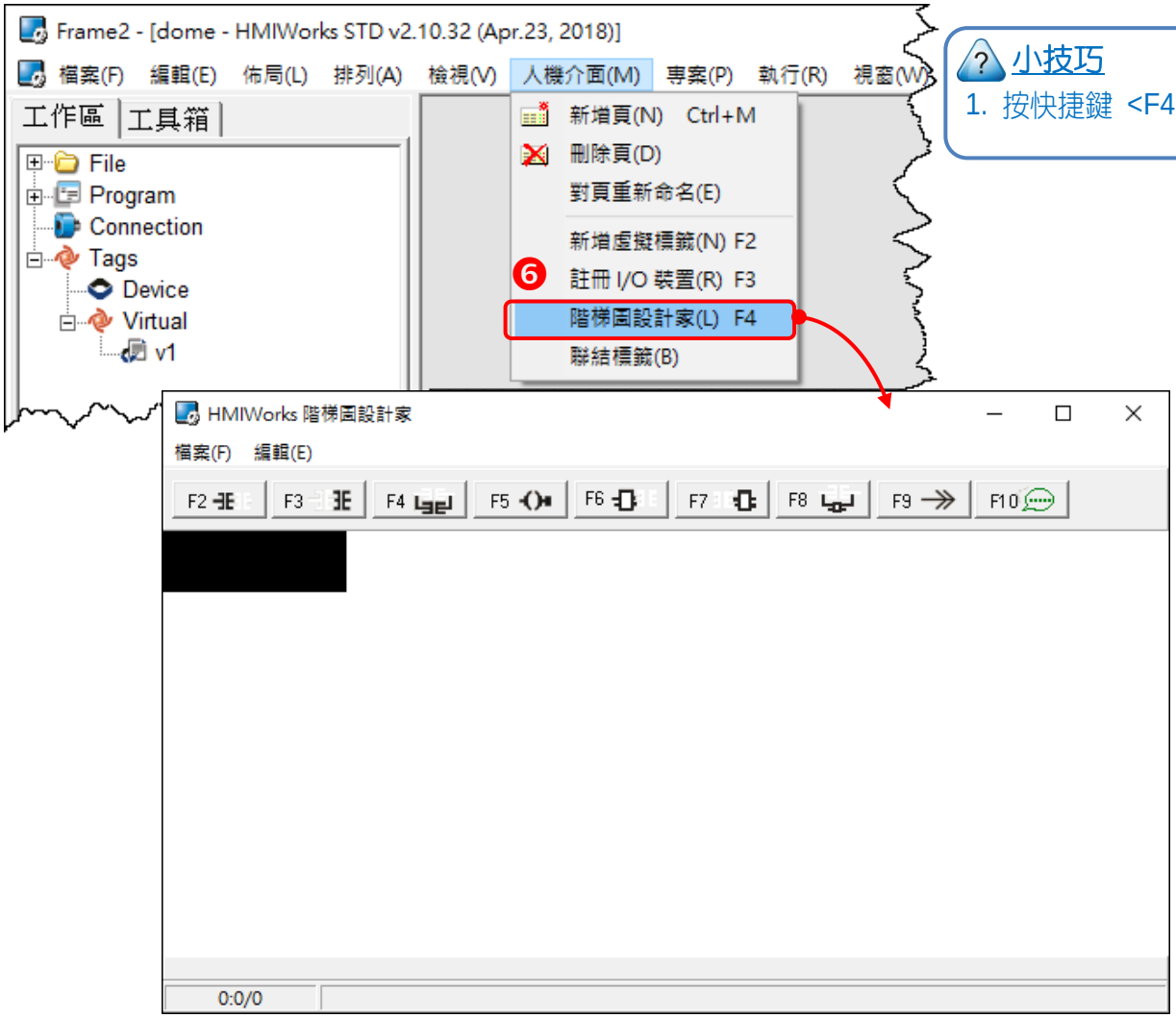
小技巧
1. 按快捷鍵 <F2> 。
2. “工作區”裡，在“Virtual” 項目上按右鍵，在彈出的功能選單中，點選“New Virtual Tag”。

步驟 4: 在**“名稱”** 欄位輸入標籤名稱 (如: v1)，在按下 **“確定(O)”** 按鈕。
步驟 5: 查看 **“工作區”** 中 **“Virtual”** 項目下將顯示剛新建的標籤 (如: v1)。



注意
詳細標籤使用方式，請參考 第 3.3.3 節 階梯圖設計家的操作。

步驟 6: 從 **“人機介面(M)”** 功能選單中，點選 **“階梯圖設計家(L)”** 項目來開啟 “HMIWorks 階梯圖設計家” 配置視窗。詳細關於階梯圖設計家的介面說明、使用方式及功能方塊介紹…等，請參考 第 3.3.2 節 階梯圖設計家的簡介 及 第 3.3.3 節 階梯圖設計家的操作。



3.3.2 階梯圖設計家的簡介

階梯圖設計家 (Ladder Designer) 是一個用來設計、實現使用者階梯圖邏輯的軟體。本章節將介紹階梯圖設計家的操作介面，包含選單欄、功能鍵、功能方塊及使用方式...等。

3.3.2.1 外觀

我們在上一章節 (第 3.3.1 開始設計) 已成功開啟階梯圖設計家配置視窗，顯示畫面如下圖所示。階梯圖設計家主要包含三個部分，選單欄、功能鍵列和編輯區。圖中黑色的反白區塊為游標所在位置。

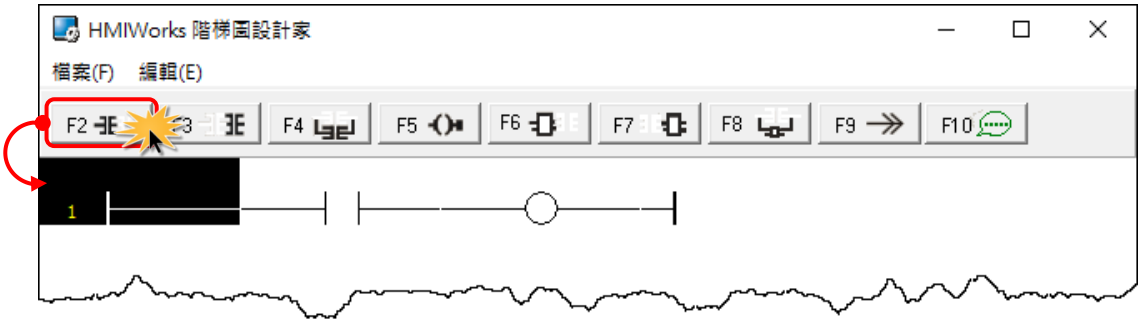
“選單欄” 的簡介:

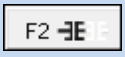
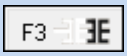
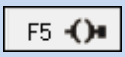
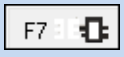
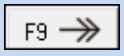
項目		快捷鍵	說明
檔案(F)	新增(N)	Ctrl + N	建立一個新的階梯圖設計家檔案。
	開啟(O)	Ctrl + O	選擇一個現有的階梯圖設計家檔案來開啟。
	儲存(S)	Ctrl + S	儲存階梯圖設計家檔案。
	另存新檔...(A)	Ctrl + A	將階梯圖設計家檔案儲存為另一個新的檔名。
	儲存後關閉(C)	Ctrl + K	儲存檔案後並關閉階梯圖設計家視窗。
	離開(X)	Ctrl + X	離開並關閉階梯圖設計家視窗。

項目			快捷鍵	說明
編　輯(E)	新增一階(N)	在之前插入(B)	Ctrl + I	在選定的階(Rung)之前插入一階(Rung)。
		在之後插入(A)	Ctrl + M	在選定的階(Rung)之後插入一階(Rung)。
	複製(D)		Ctrl + D	複製並貼上選定的階(Rung)。
	複製 (至剪貼簿)(C)		Ctrl + C	複製選定的階(Rung)到剪貼簿中。
	貼上(P)		Ctrl + V	從剪貼簿中貼上複製的階(Rung)。

“功能鍵列” 的簡介:

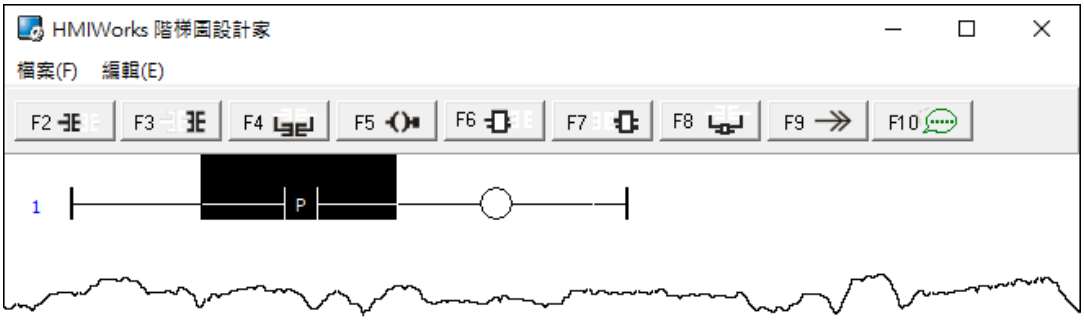
範例: 按下  按鈕來插入一階開關輸入 (Contact Input)，如下圖所示。



項目	快捷鍵	說明
	F2	在游標左側插入一階開關輸入(Contact Input)
	F3	在游標右側插入一階開關輸入(Contact Input)
	F4	在平行於游標處插入一階開關輸入(Contact Input)
	F5	插入一個線圈輸出 (Coil Output)
	F6	在游標左側插入一個功能方塊 (Function Block)
	F7	在游標右側插入一個功能方塊 (Function Block)
	F8	在平行於游標處插入一個功能方塊 (Function Block)
	F9	在平行於游標處插入一個跳出 (Jump)
	F10	註解

“開關輸入 (Contact Input)” 的簡介:

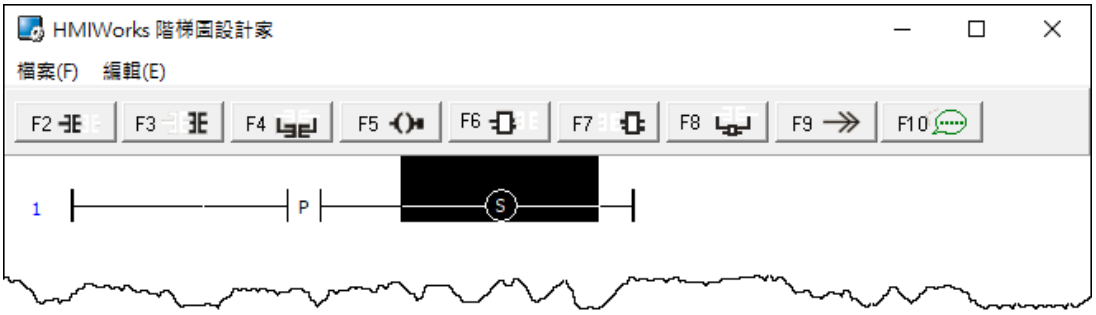
範例: 點選開關輸入，再按鍵盤 **<P>** 鍵 (或連續按 **<空白鍵>** 來調整輸入類型)，如下圖所示。



項目	快捷鍵	說明
┤ ├	空白鍵	常開開關輸入 (A normally-open contact input)
┤ \ ├	空白鍵 或 \	常關開關輸入 (A normally-closed contact input)
┤ P ├	空白鍵 或 P	正向開關輸入 (A positive transition contact input) 當狀態從關到開 (OFF → ON) 時，觸發一次
┤ N ├	空白鍵 或 N	負向開關輸入(A negative transition contact input) 當狀態從開到關 (ON → OFF) 時，觸發一次

“線圈輸出 (Coil Output)” 的簡介:

範例: 點選線圈輸出，再按鍵盤 **<S>** 鍵 (或連續按 **<空白鍵>** 來調整輸出類型)，如下圖所示。



項目	快捷鍵	說明
	**空白鍵**	常開線圈輸出 (A normally-open coil output)
	**空白鍵 或**	常關線圈輸出 (A normally-closed coil output)
	**空白鍵 或 S**	設定型線圈輸出 (A 「Set」 coil output) 一旦設定 (Set) 後，在重置 (Reset) 之前，線圈永遠保持開 (ON) 的狀態
	**空白鍵 或 R**	重置型線圈輸出 (A 「Reset」 coil output) 一旦重置 (Reset) 後，在設定 (Set) 之前，線圈永遠保持關 (OFF) 的狀態
	**空白鍵 或 P**	正向線圈輸出 (A positive transition coil output) 當狀態從關到開 (OFF → ON) 時，觸發一次
	**空白鍵 或 N**	負向線圈輸出 (A negative transition coil output) 當狀態從開到關 (ON → OFF) 時，觸發一次

3.3.3 階梯圖設計家的操作

本章節將介紹如何使用階梯圖設計家。

3.3.3.1 新增虛擬 (Virtual) 標籤 (F2)

使用階梯圖設計家 (Ladder Designer) 之前，首先要新增標籤，詳細步驟如下:

步驟 1: 從 **“人機介面(M)”** 功能選單中，點選 **“新增虛擬標籤(N)”** 項目來開啟 “編輯標籤” 配置視窗。

小技巧

1. 按快捷鍵 <F2> 。
2. “工作區”裡，在“Virtual” 項目上按右鍵，在彈出的功能選單中，點選“New Virtual Tag”。

步驟 2: 在 **“名稱”** 欄位輸入標籤名稱 (如: v1) 。

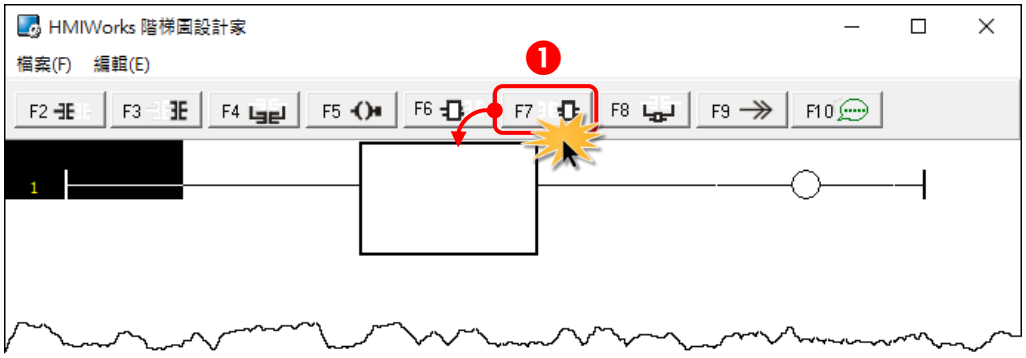
步驟 3: 按下 **“確定(O)”** 按鈕。

為了以下幾節的說明，在這裡先新增三個標籤，名稱分別是 v1、v2、v3。

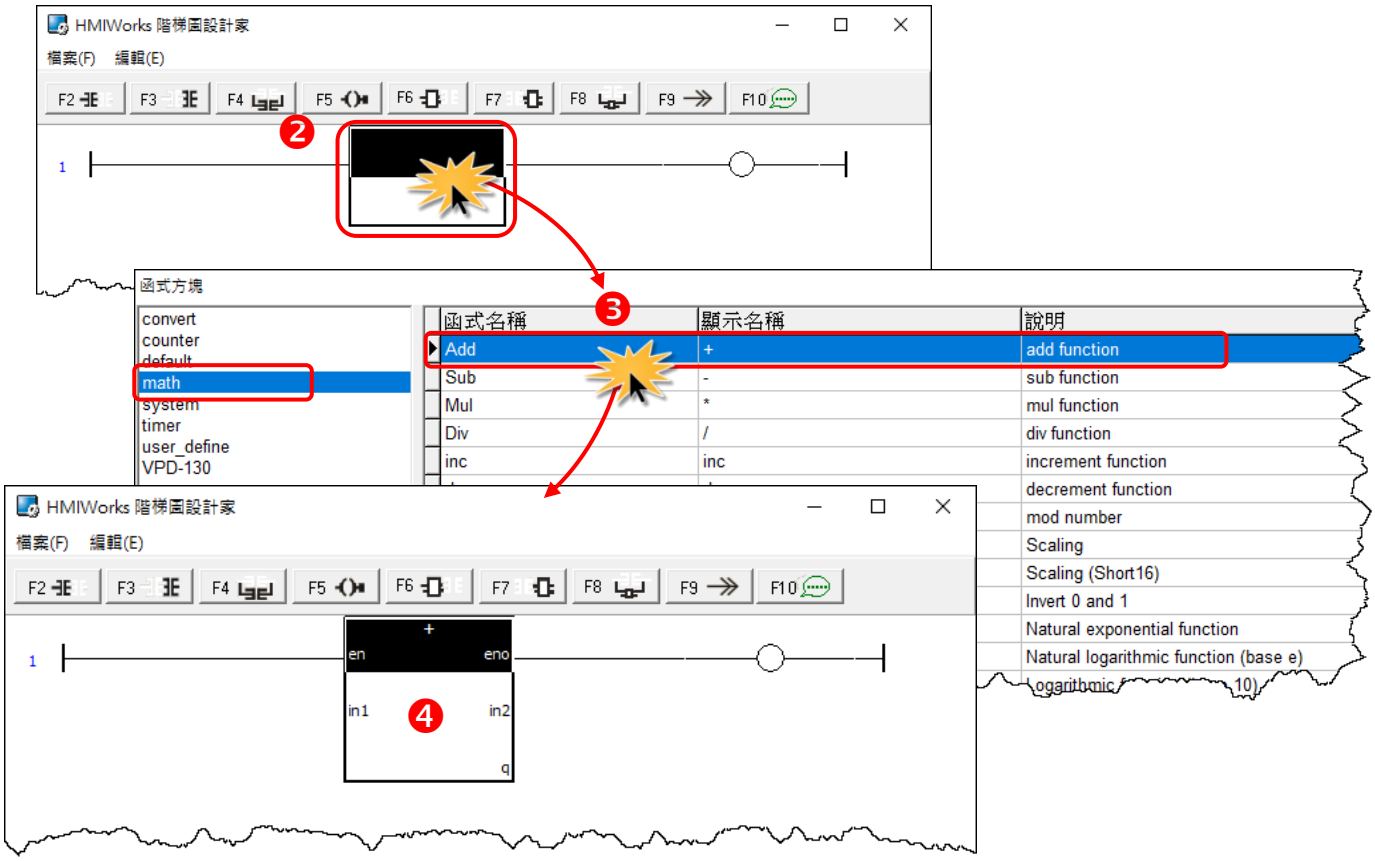
3.3.3.2 選擇標籤和輸入常數

下面將以 **數學算式: v3 = 1 + 2** 及 **v2 = v1** 為說明範例。

步驟 1: 按鍵盤 **<F4>** 來開啟階梯圖設計家，按 F7 按鈕來插入一階功能方塊。



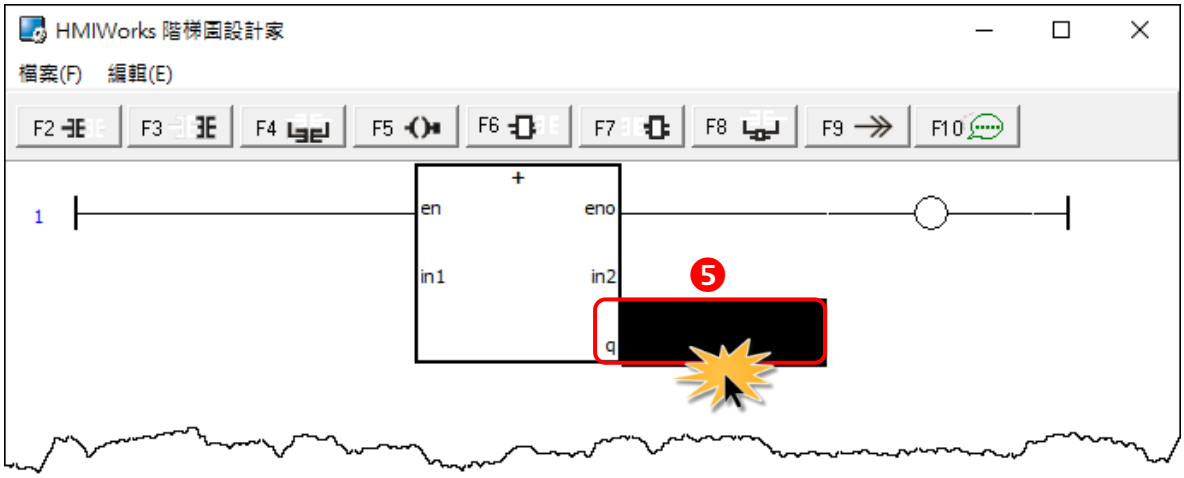
步驟 2: 雙擊此功能方塊來開啟“函式方塊”配置視窗。
步驟 3: 選擇 **“math”** 項目及雙擊 **“Add”** 函式名稱。
步驟 4: 將開啟 **“Add”** 功能方塊。



瀏覽標籤及輸入常數

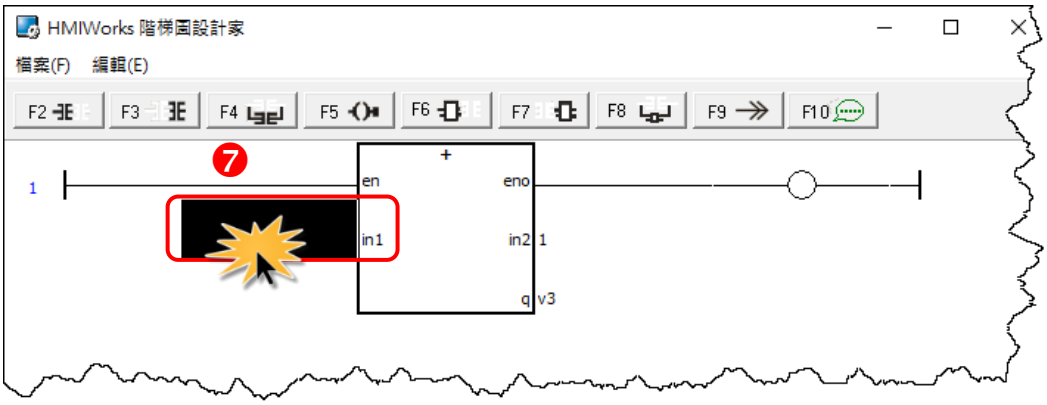
範例: 數學式 v3 = 1 + 2

步驟 5: 雙擊 **“q” 符號**來開啟 “選擇標籤” 視窗。

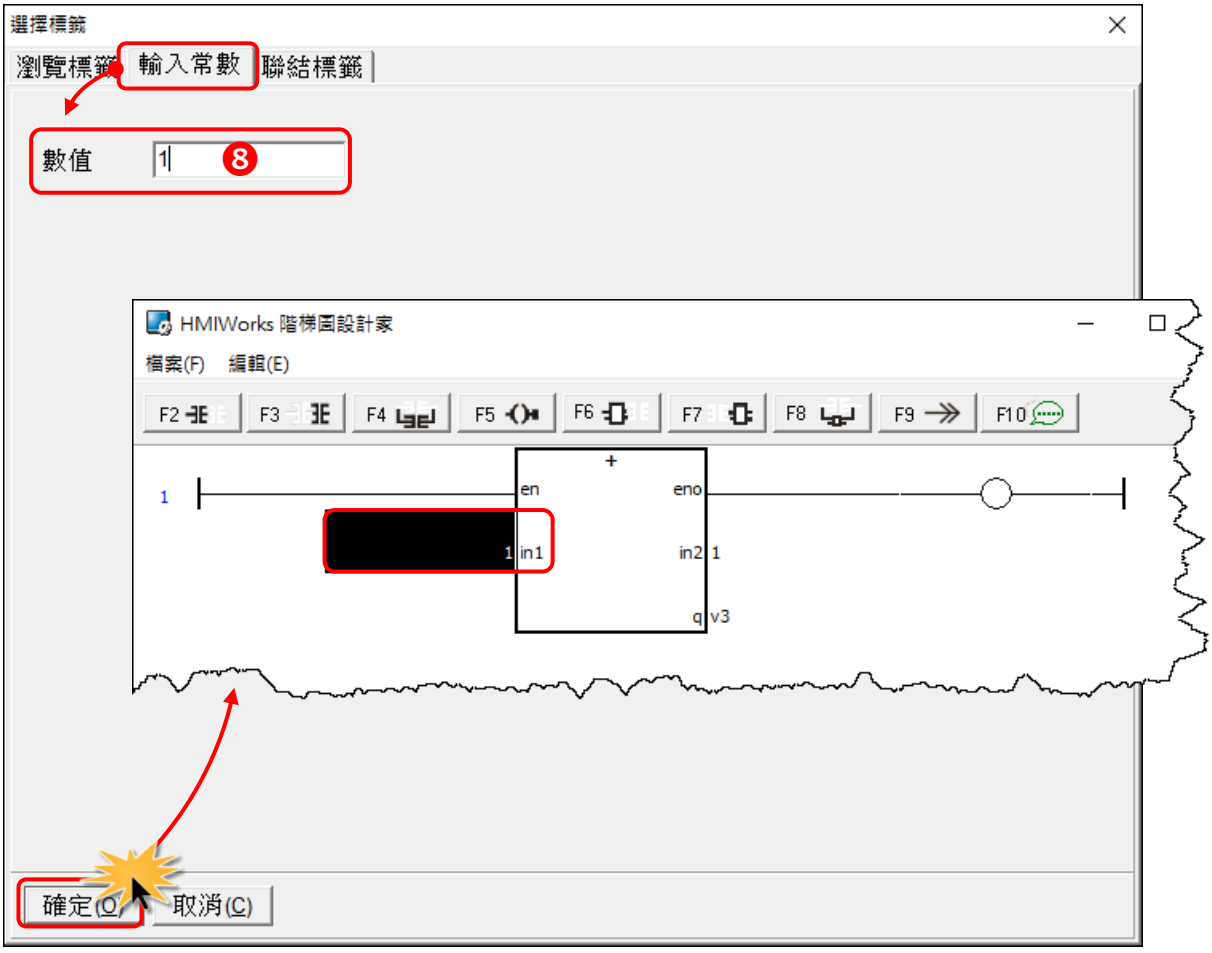


步驟 6: 點選 “瀏覽標籤” 項目，雙擊 **“v3”** 來選擇它。

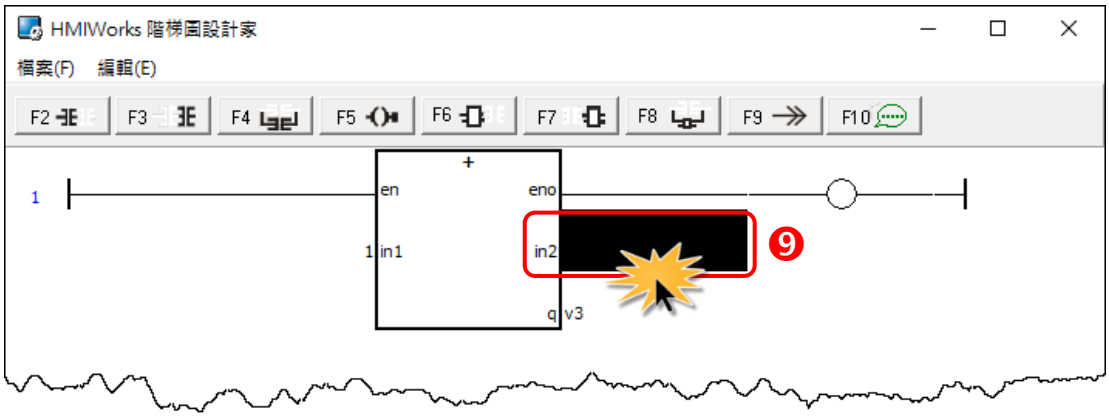
步驟 7: 雙擊 **“in1”符號**來開啟“選擇標籤”視窗。



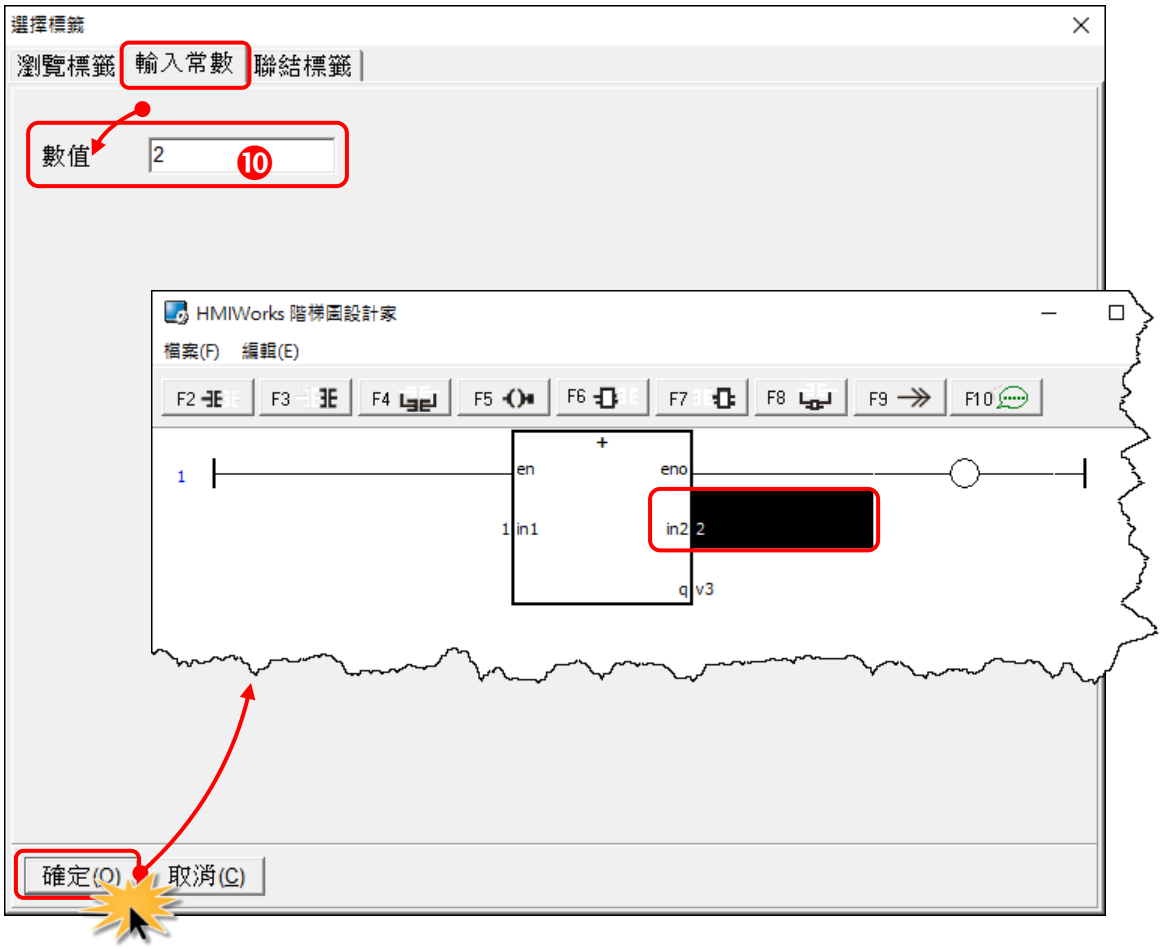
步驟 8: 點選 **“輸入常數”**項目，然後在 **“數值”**欄位輸入 **“1”** ，再按下 **“確定(O)”**按鈕。



步驟 9: 雙擊 **“in2” 符號**來開啟“選擇標籤”視窗。



步驟 10: 點選 **“輸入常數”** 項目，然後在 **“數值”** 欄位輸入 **“2”** ，再按下 **“確定(O)”** 按鈕。

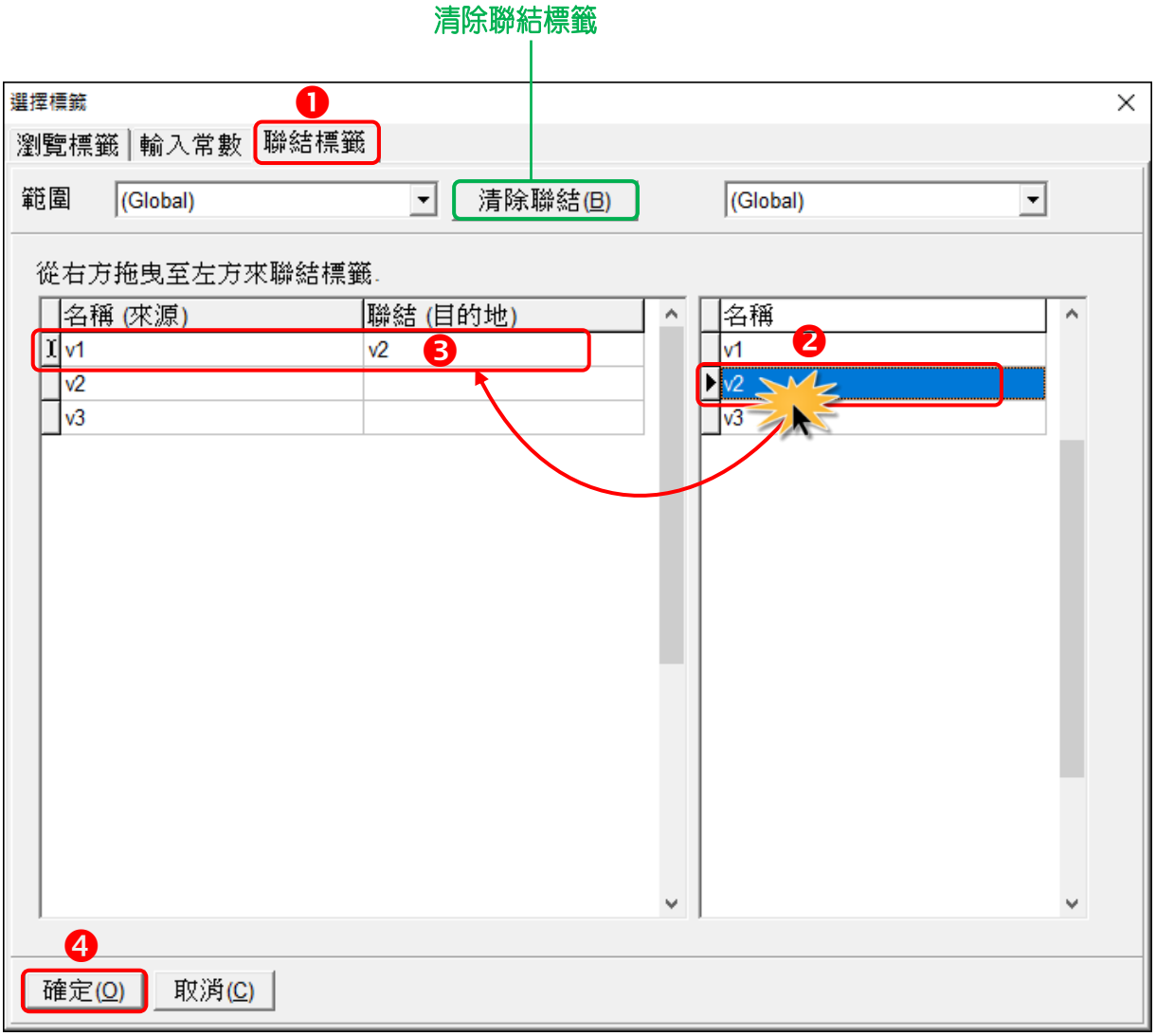


聯結標籤

範例 v2 = v1

從右側的標籤拖到左側的標籤以綁定標籤。例如: 當 v2 拖移到 v1 時，如 v1 變更，則 v2 = v1。詳細應用請參考 第 3.3.8 節 資料交換。

步驟 1: 點選 **“聯結標籤”** 項目。
步驟 2: 在右手邊標籤清單中點選 **“v2”** 。
步驟 3: 拖移 v2 到左手邊 **v1** 的 **“聯結 (目的地)”** 欄位。
步驟 5: 按下 **“確定(O)”** 按鈕。

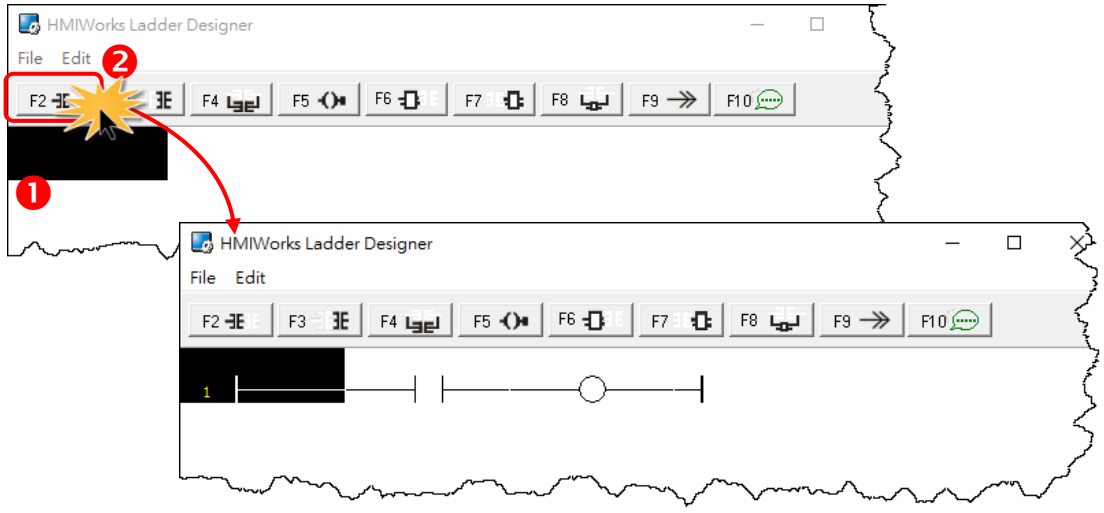


3.3.3.3 插入/刪除一階

插入一階

步驟 1: 移動游標到空白的地方。

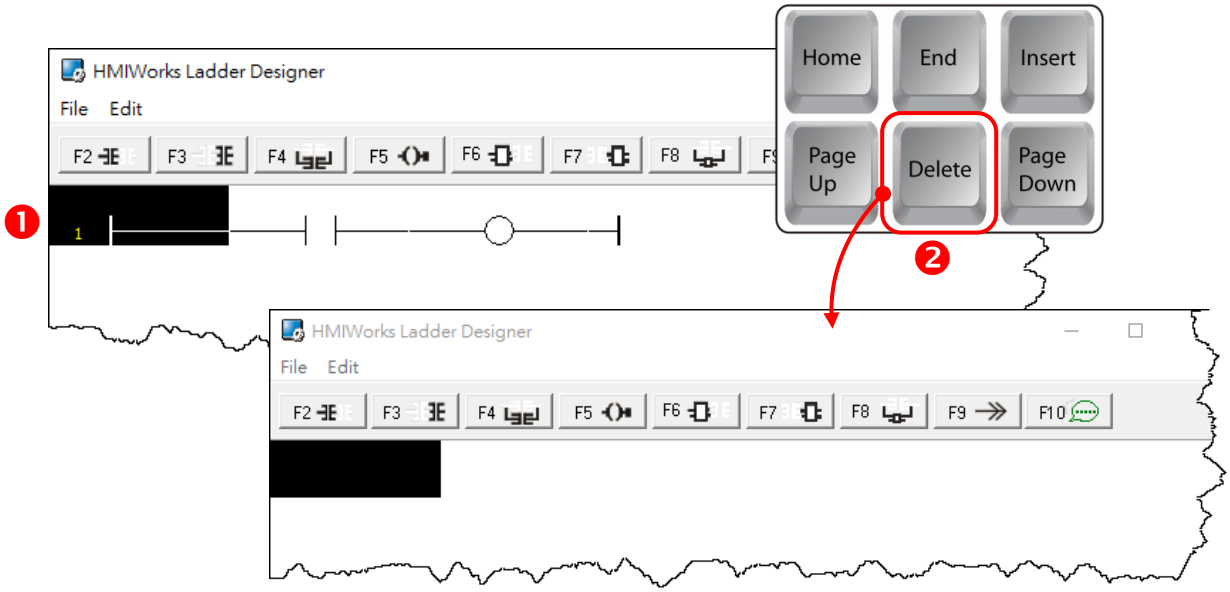
步驟 2: 按下 F2 按鈕 (或按 **<F2> 鍵**) 來插入一階。



刪除一階

步驟 1: 移動游標(反白區域)到每一階的起始點。

步驟 2: 按 **<Delete>** 鍵來刪除一階。



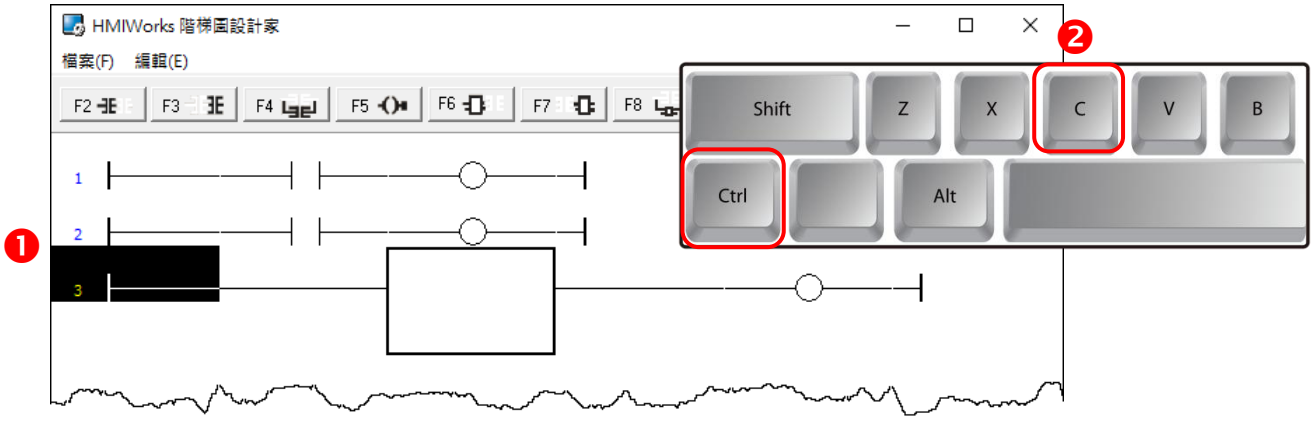
3.3.3.4 複製和貼上一階

假設現在共有三個階，而我們想要複製第三階，然後貼上於第一與第二階之間。

複製一階

步驟 1: 移動游標到第三階。

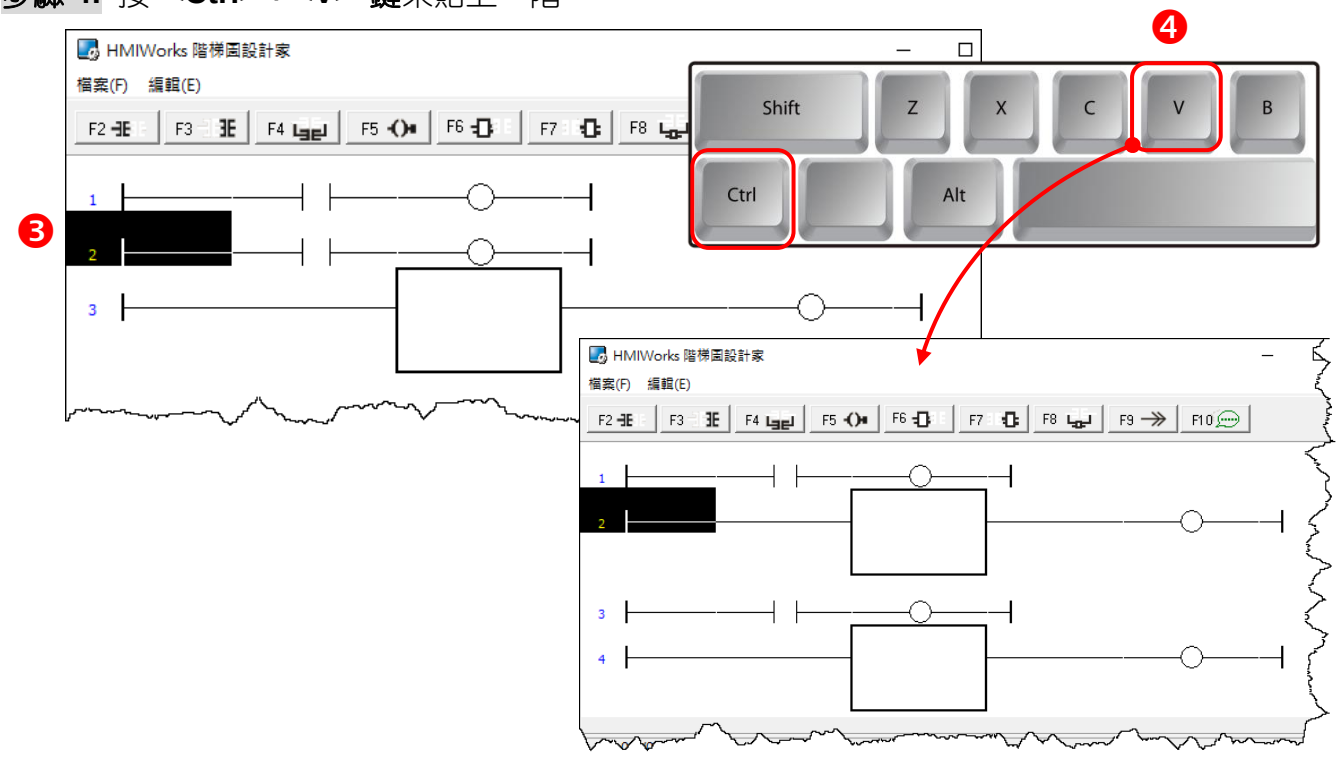
步驟 2: 按 **<Ctrl> + <C> 鍵**來複製一階。



貼上一階

步驟 3: 移動游標到第二階。

步驟 4: 按 **<Ctrl> + <V> 鍵**來貼上一階。

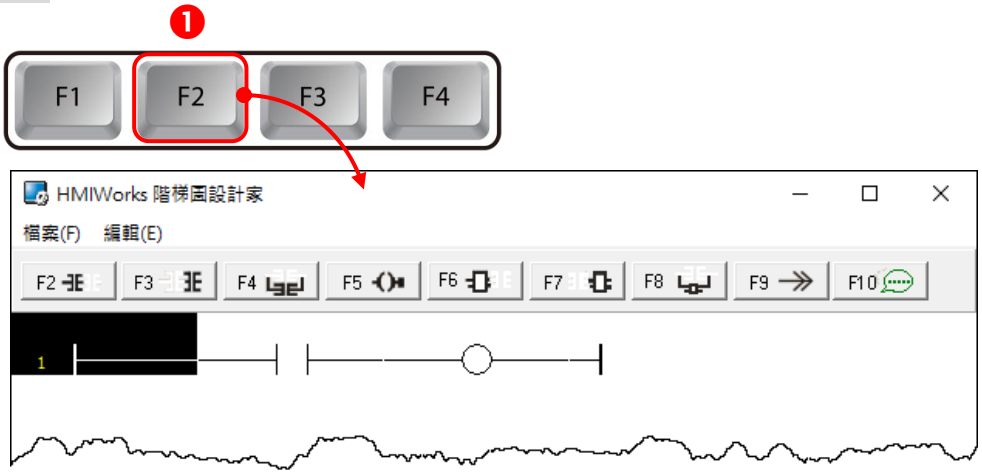


3.3.3.5 插入/刪除一個開關輸入

為了說明如何插入/刪除一個開關輸入 (Contact Input)，我們將於下面步驟中，說明每一項功能。

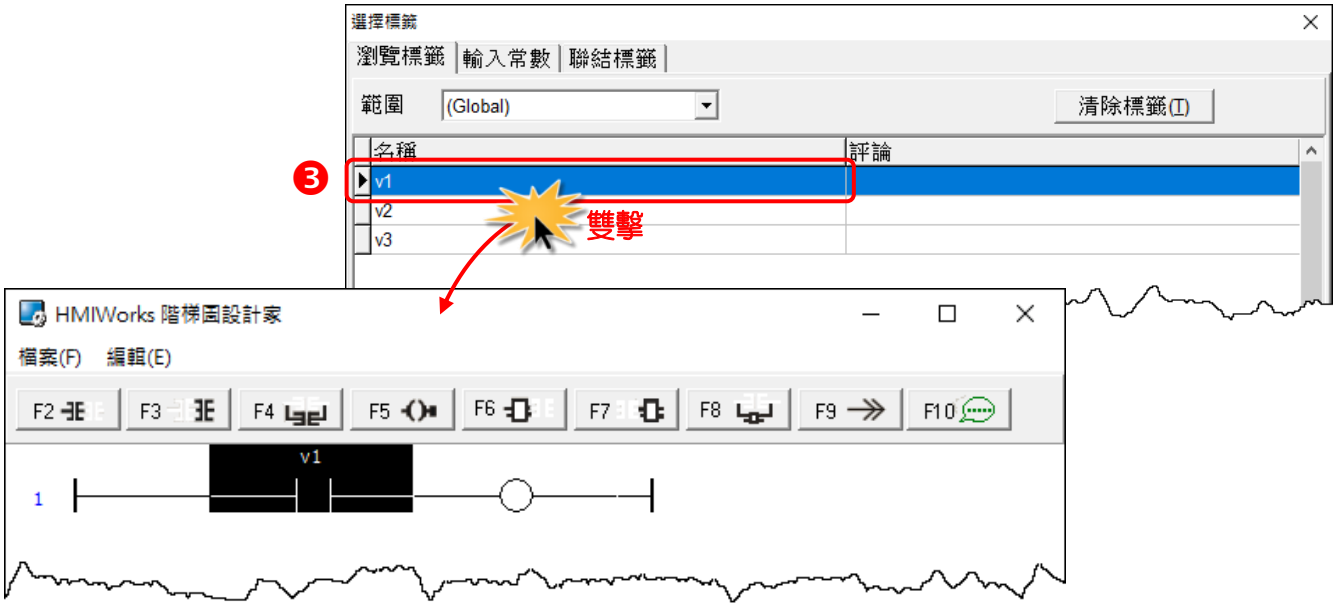
將一個標籤 (tag) 和該開關輸入建立關連

步驟 1: 按**<F2> 鍵**來插入一個含有一個開關輸入和一個線圈輸出的階。



步驟 2: 在新的一階中雙擊開關輸入來開啟出現“**選擇標籤**”視窗。

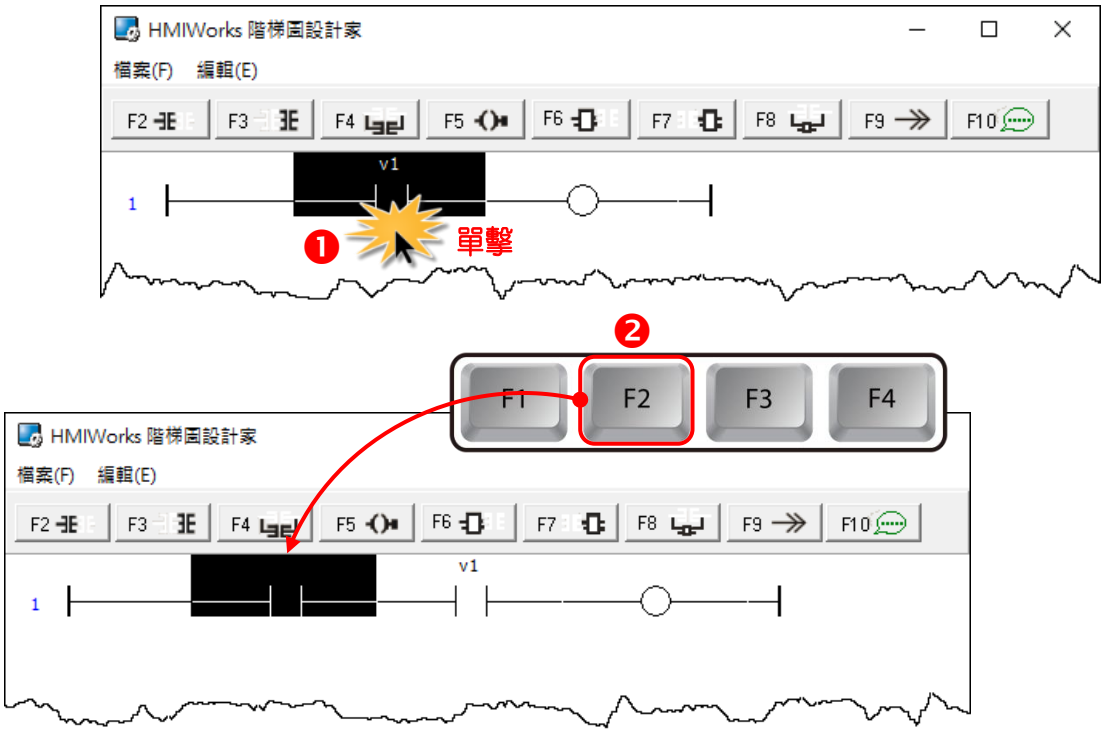
步驟 3: 雙擊 **“v1”** 標籤名稱來建立該標籤和開關輸入的關連。



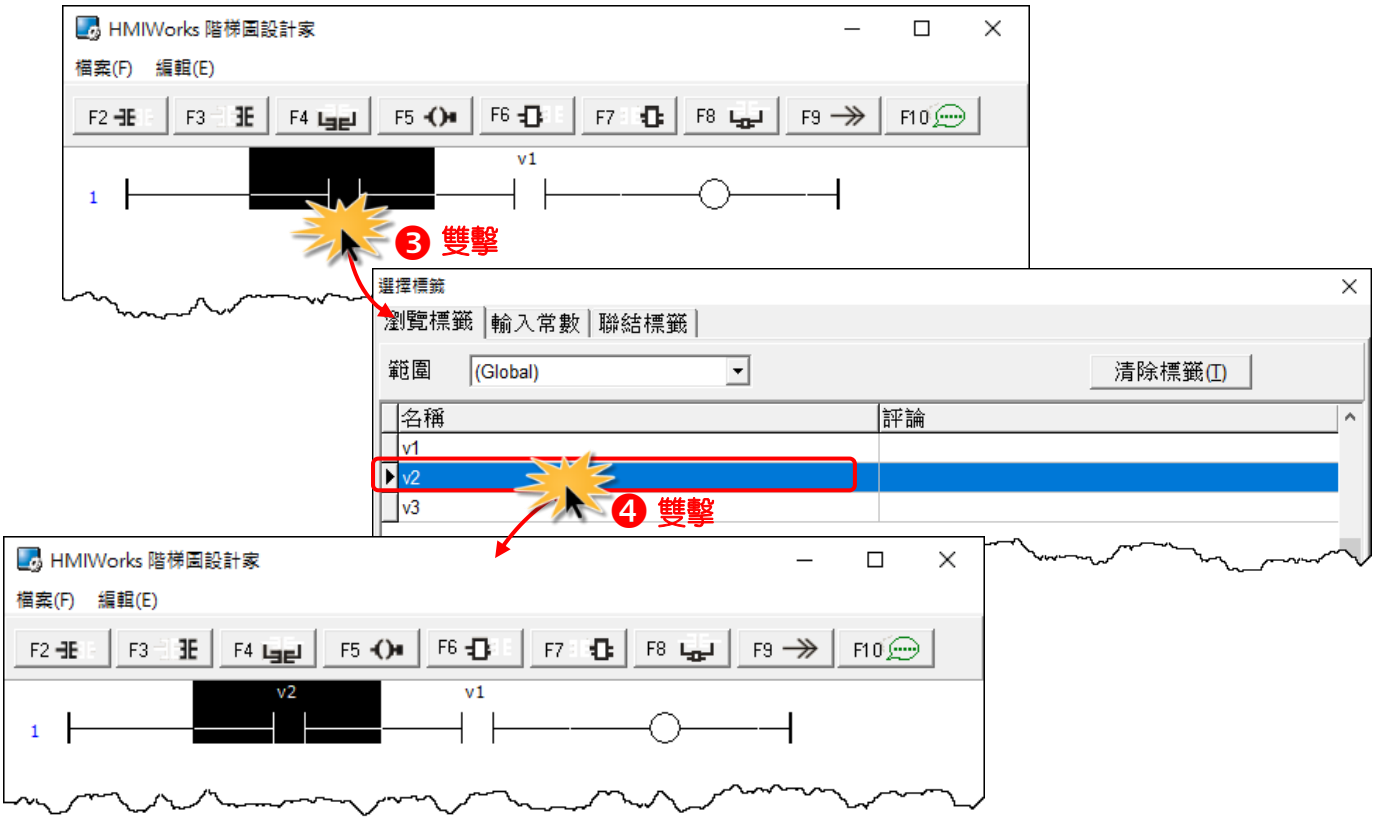
在游標左方插入一個新的開關輸入(F2)

步驟 1: 將游標移到 **“v1”** 的開關輸入上。

步驟 2: 按 **<F2> 鍵**。



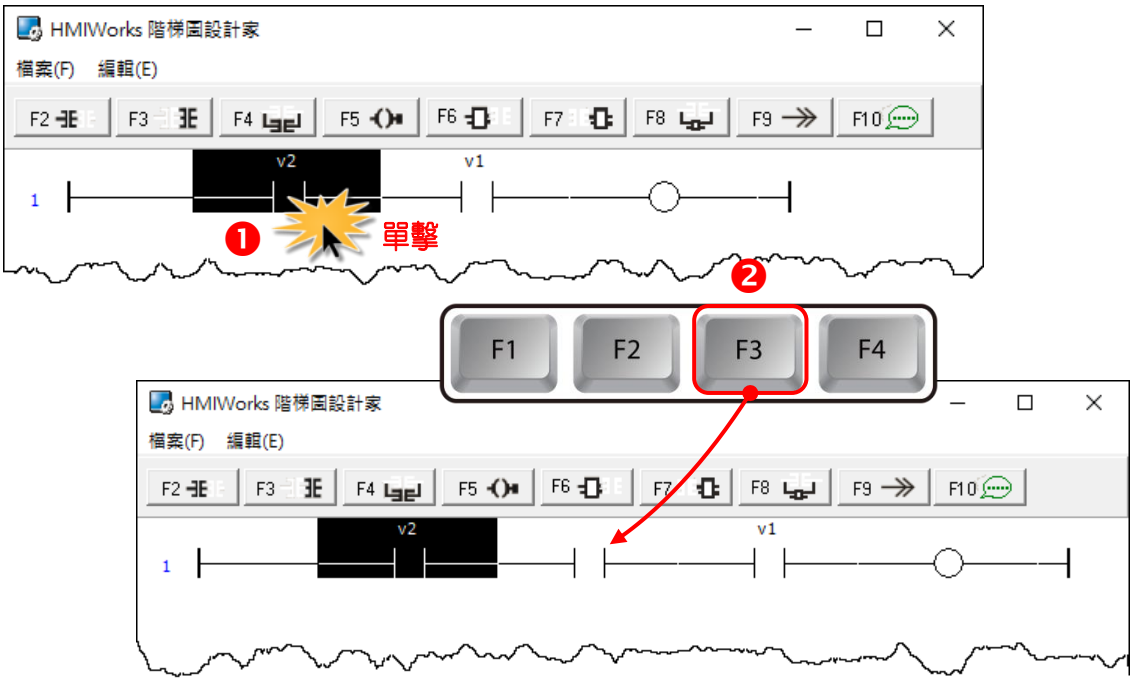
步驟 3: 將標籤 **“v2”** 和剛新插入的開關輸入建立關連。



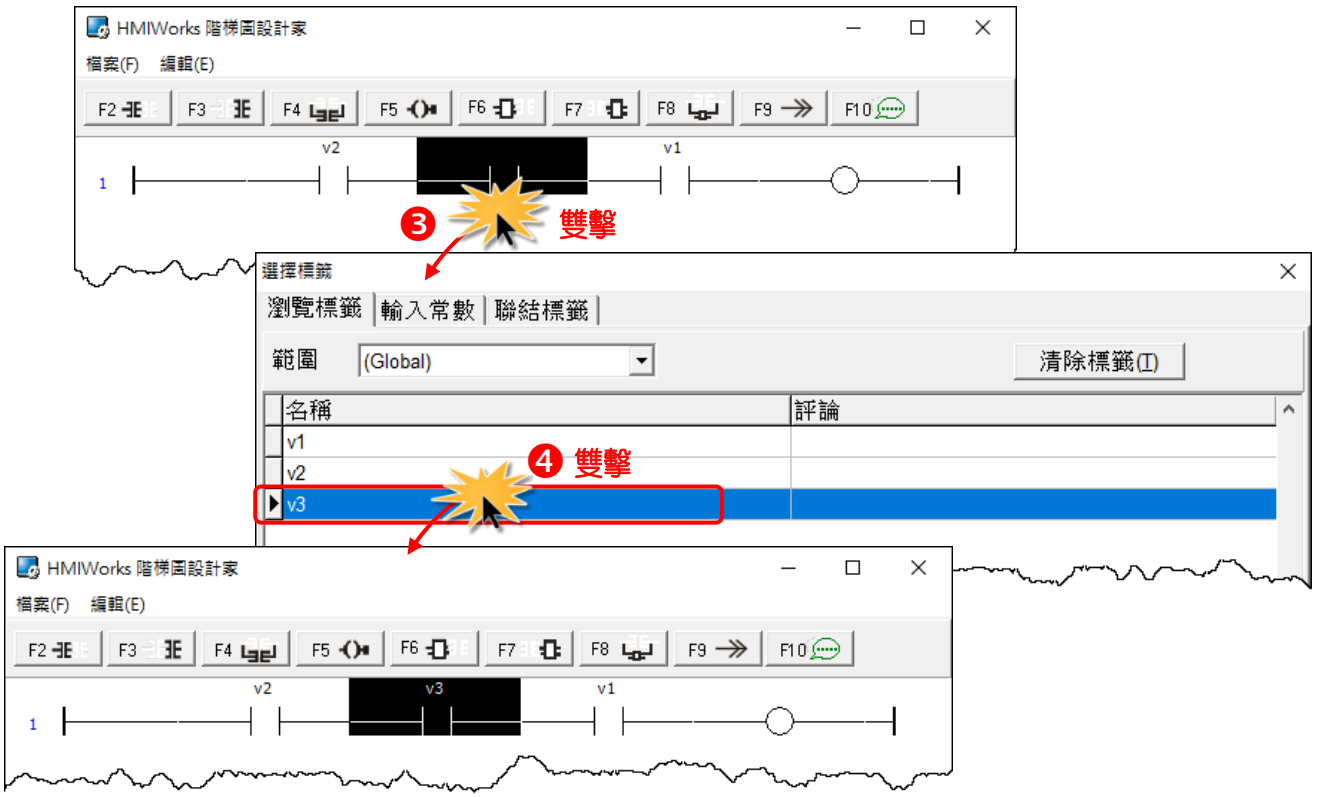
在游標右方插入一個新的開關輸入(F3)

步驟 1: 將游標移到 **“v2”** 的開關輸入上。

步驟 2: 按 **<F3> 鍵**。



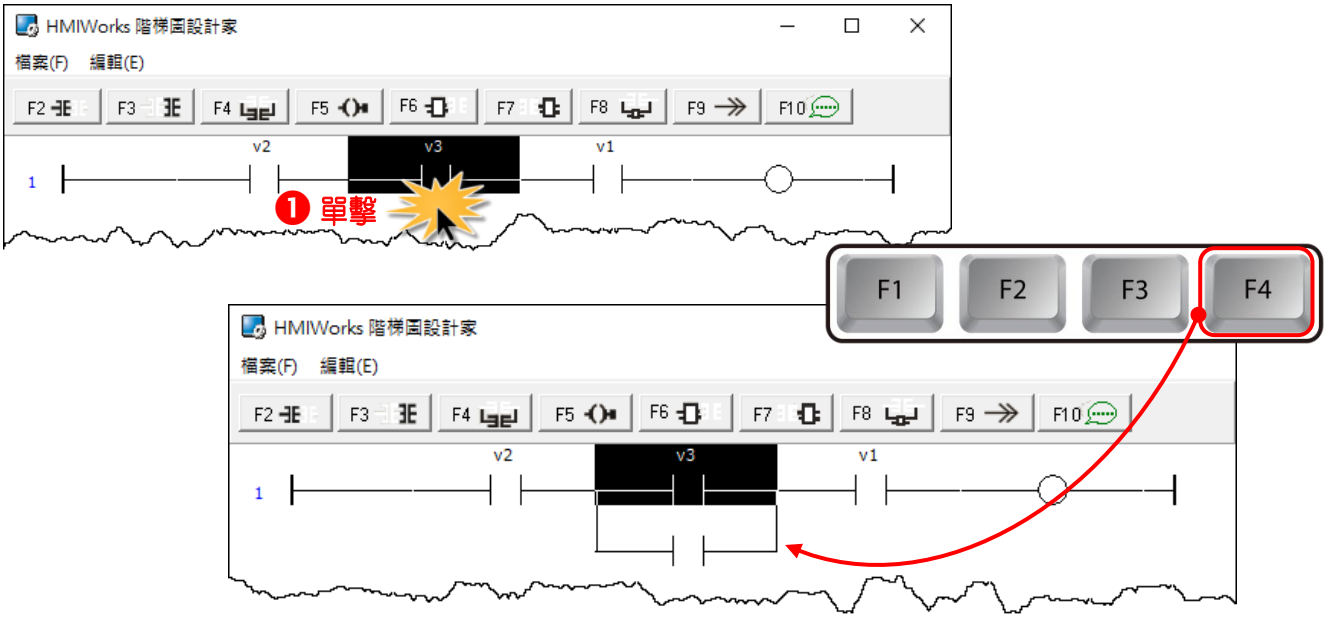
步驟 3: 將標籤 **“v3”** 和剛新插入的開關輸入建立關連。



在游標平行處插入一個新的開關輸入(F4)

步驟 1: 將游標移到 **“v3”** 的開關輸入上。

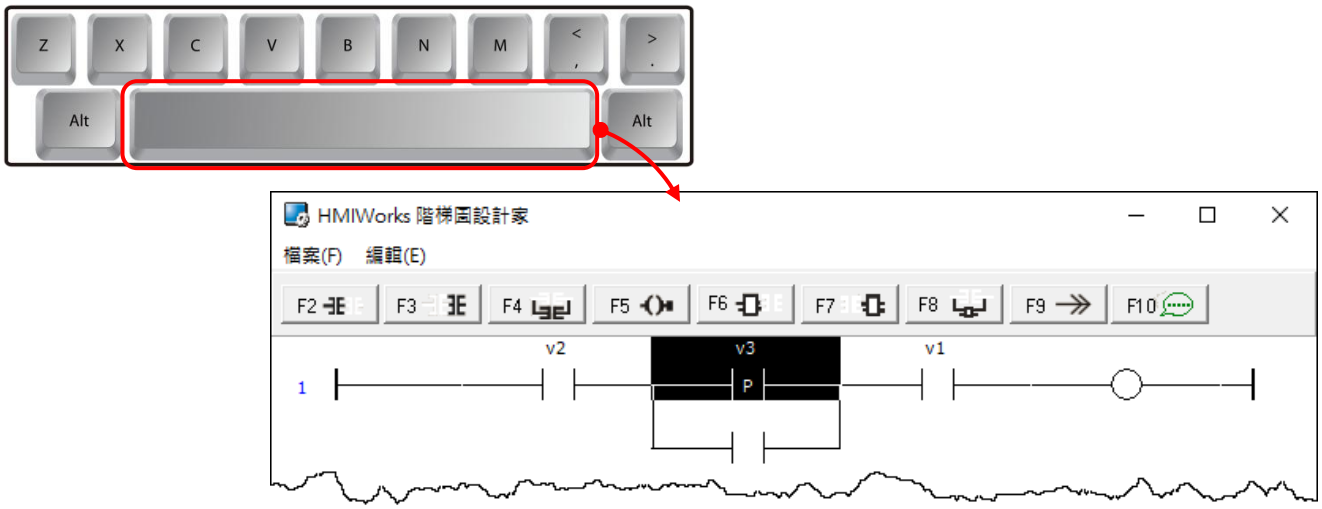
步驟 2: 按 **<F4> 鍵**。



選擇開關輸入的類別

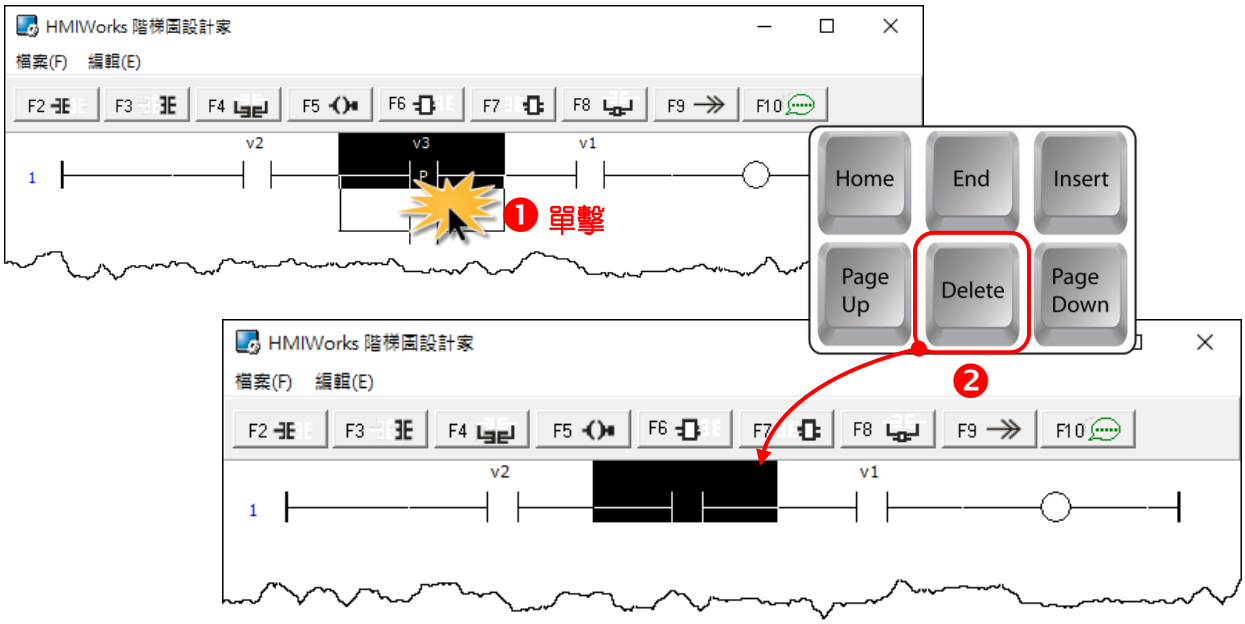
將游標移到開關輸入上，然後按下 **<空白鍵>** 來改變開關輸入的類別。

步驟 1: 將游標移到 **“v3”** 的開關輸入上。
步驟 2: 然後按 **<空白鍵>** 來改變開關輸入的類別為正向開關輸入。



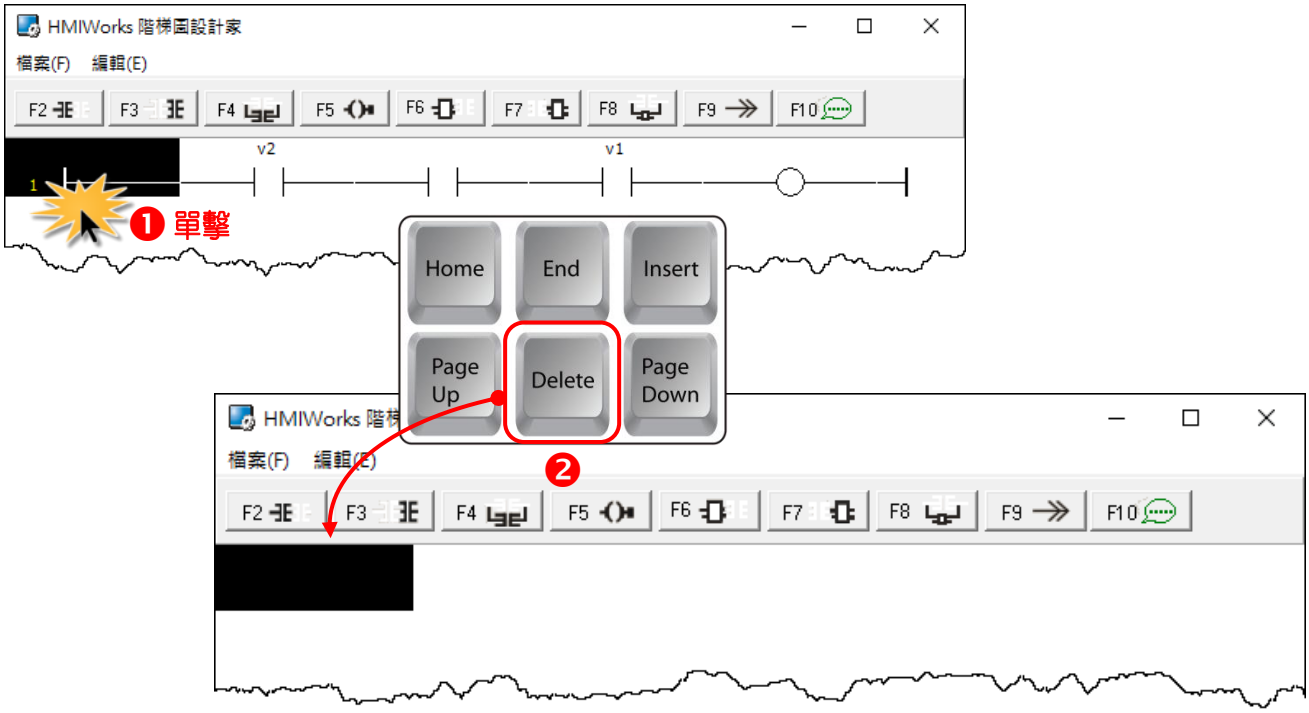
刪除一個開關輸入

移動游標到要刪除的開關輸入上，再按下 **<Delete> 鍵**。
例如: 將游標移動到 **“v3”** 的開關輸入上，再按下 **<Delete> 鍵**。



刪除一整個階

先將游標移動到該階的起始點，然後按下 **<Delete> 鍵**。

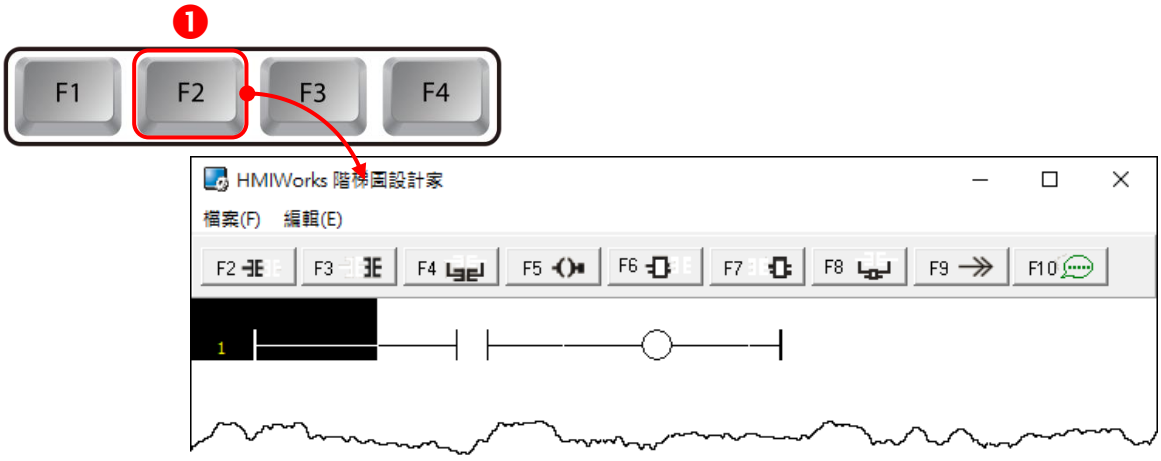


3.3.3.6 插入/刪除一個線圈輸出

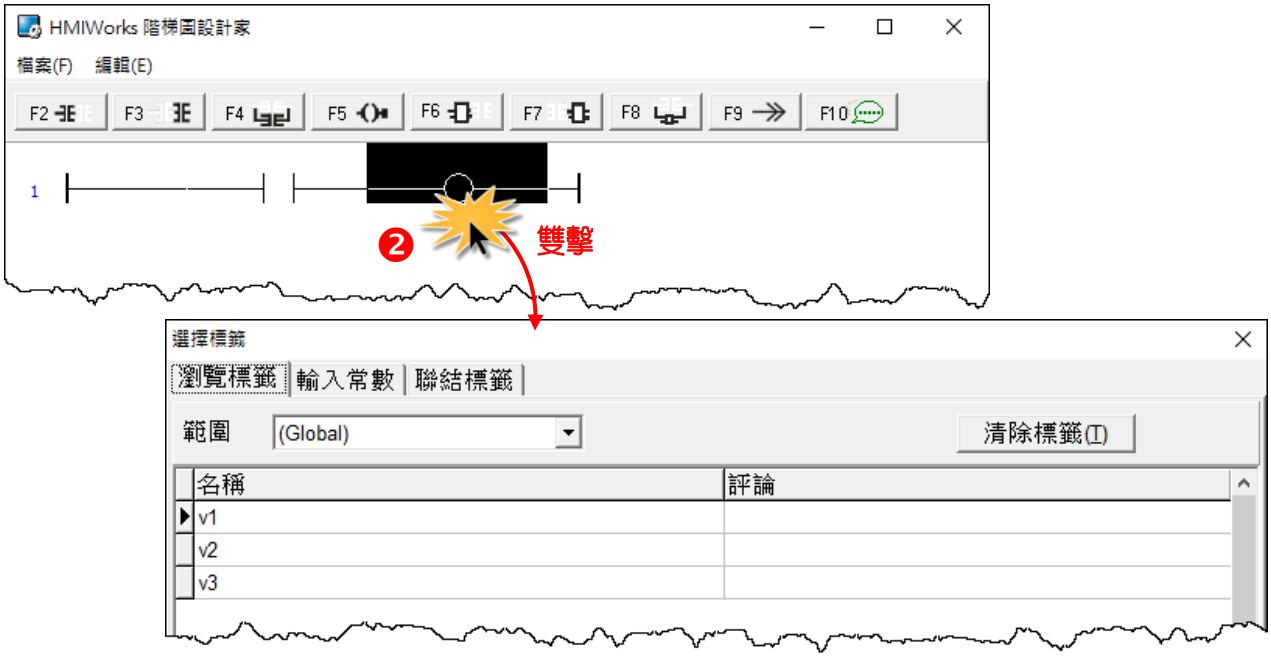
為了說明如何插入/刪除一個線圈輸出 (Coil Output)，我們將於下面一連串的步驟中，說明每一項功能。

將一個標籤 (tag) 和該線圈輸出建立關連

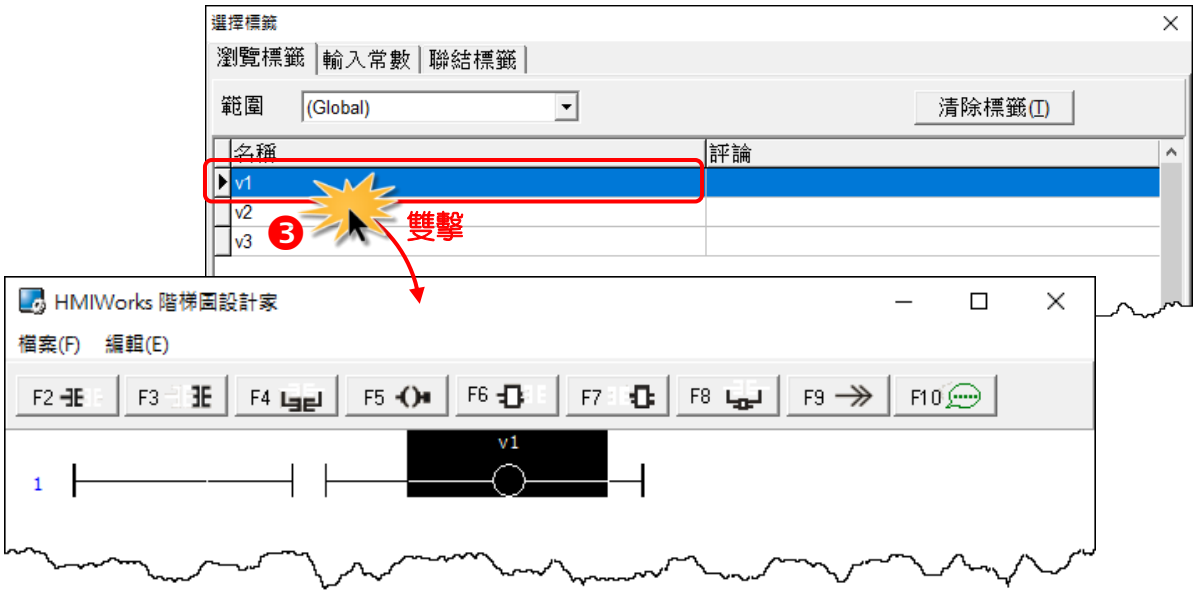
步驟 1: 按**<F2> 鍵**來插入一個含有一個開關輸入和一個線圈輸出的階。



步驟 2: 在新的一階中雙擊線圈輸出來開啟出現“**選擇標籤**”視窗。

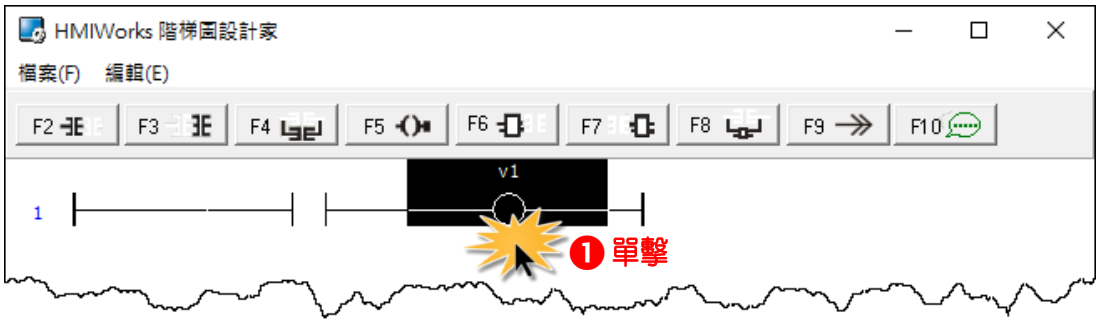


步驟 3: 雙擊 **“v1”** 標籤名稱來建立該標籤和線圈輸出的關連。

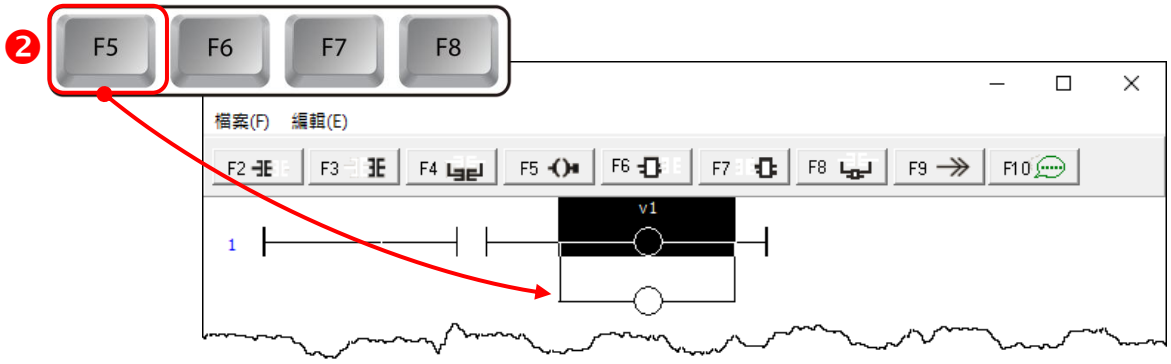


在游標平行處插入一個新的線圈輸出(F5)

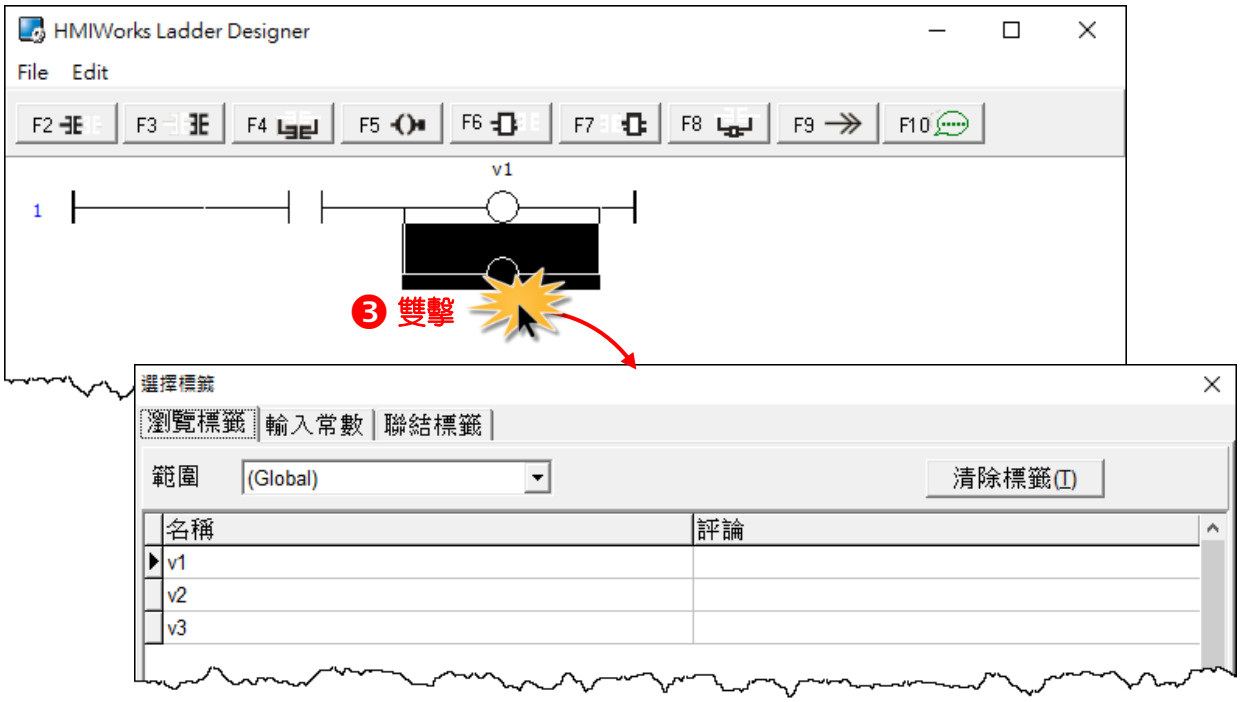
步驟 1: 將游標移到 **“v1”** 的線圈輸出上。



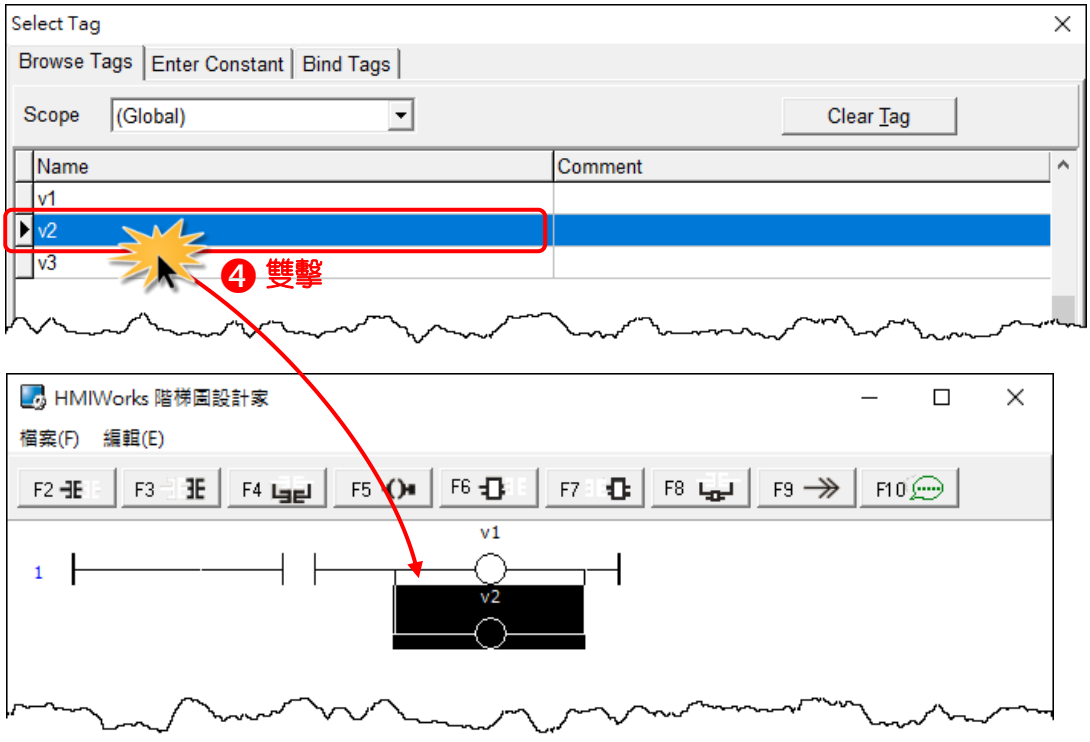
步驟 2: 按 **<F5> 鍵**。



步驟 3: 雙擊線圈輸出來開啟出現“**選擇標籤**”視窗。



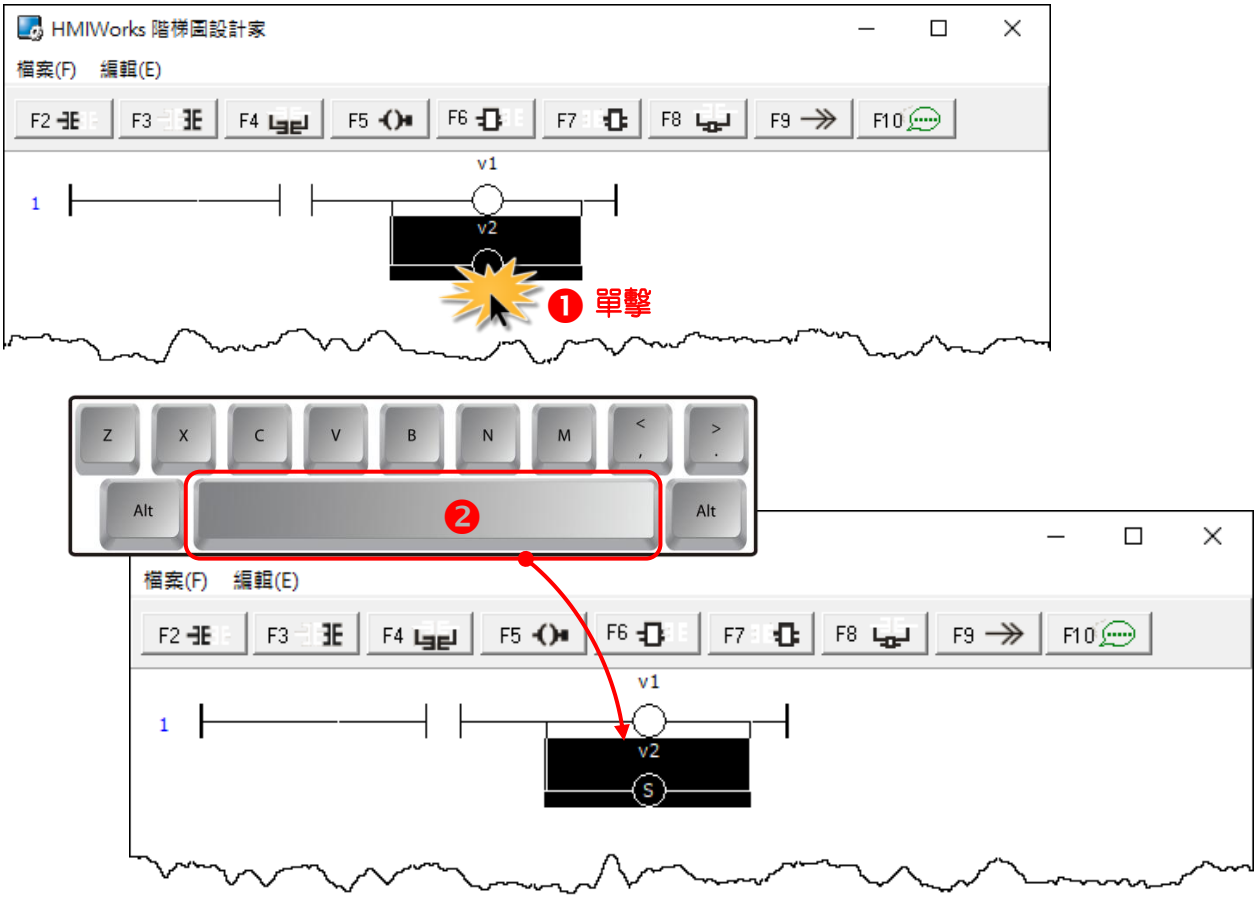
步驟 4: 雙擊 **“v2”** 標籤名稱來建立該標籤和線圈輸出的關連。



選擇線圈輸出的類別

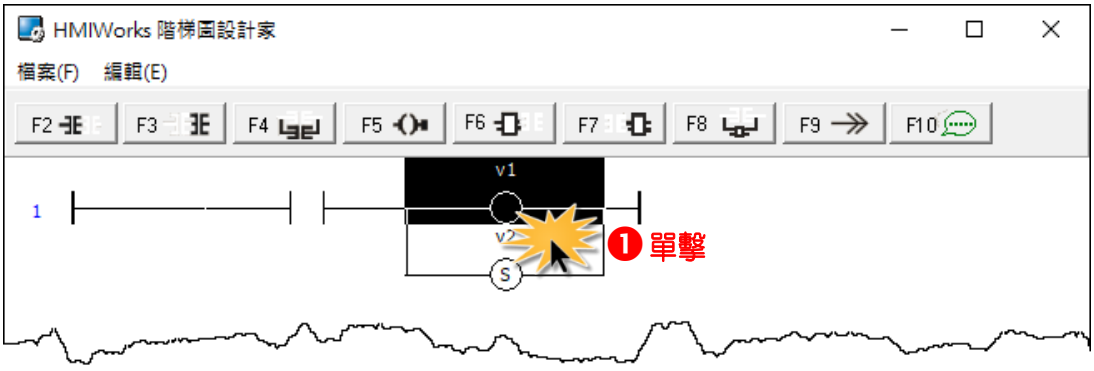
將游標移到線圈輸出上，然後按下 **<空白鍵>** 來改變線圈輸出的類別。

步驟 1: 將游標移到 **“v2”** 的線圈輸出上。
步驟 2: 然後按 **<空白鍵>** 來改變線圈輸出的類別為設定型線圈。

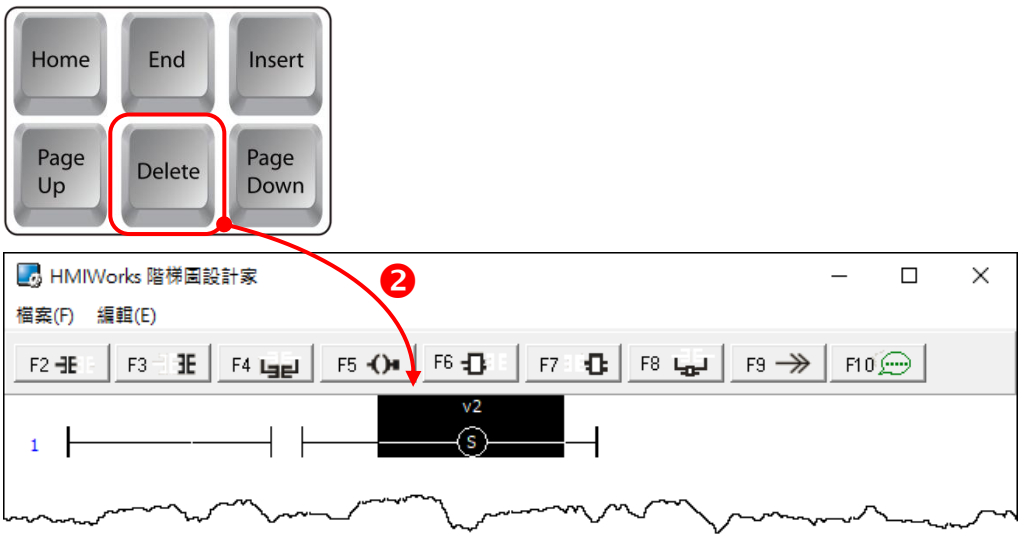


刪除一個開關輸入

步驟 1: 將游標移到 **“v1”** 的線圈輸出上。

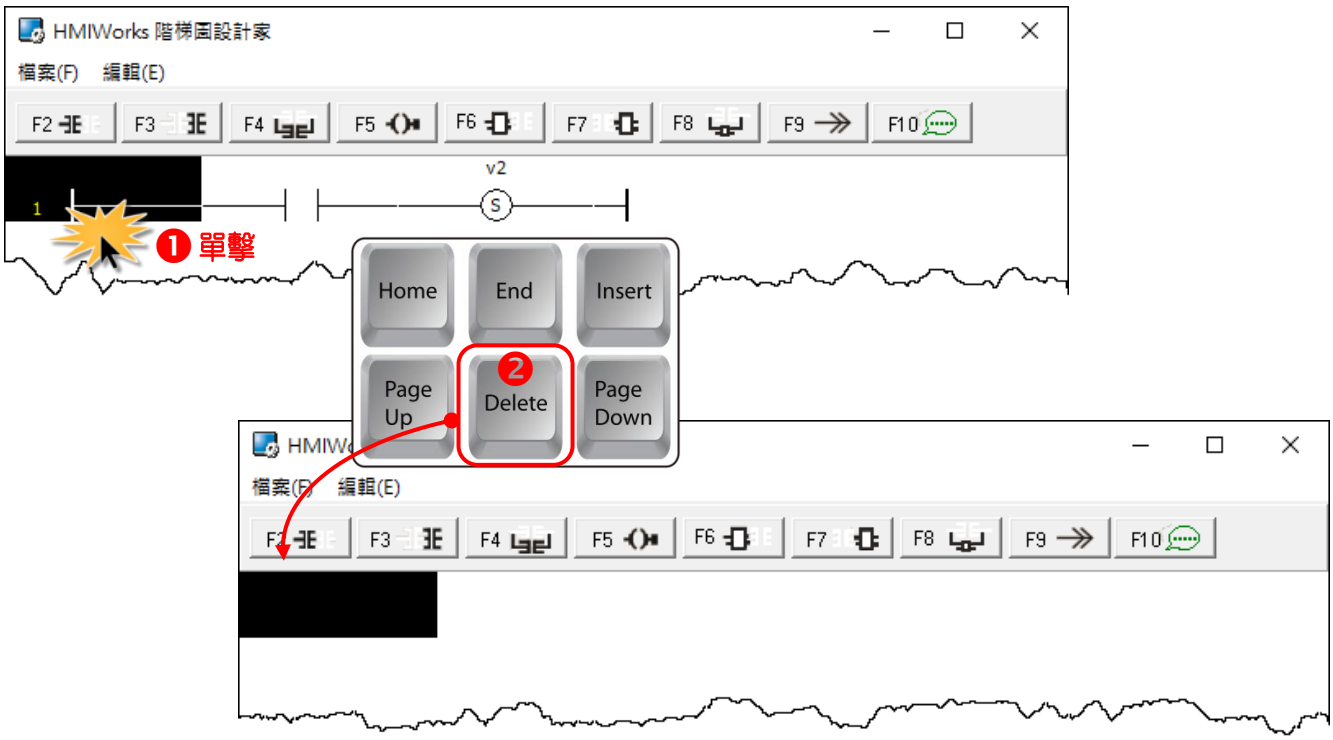


步驟 2: 按 **<Delete> 鍵**刪除 **“v1”** 的線圈輸出。



刪除一整個階

先將游標移動到該階的起始點，再按下 **<Delete> 鍵**。



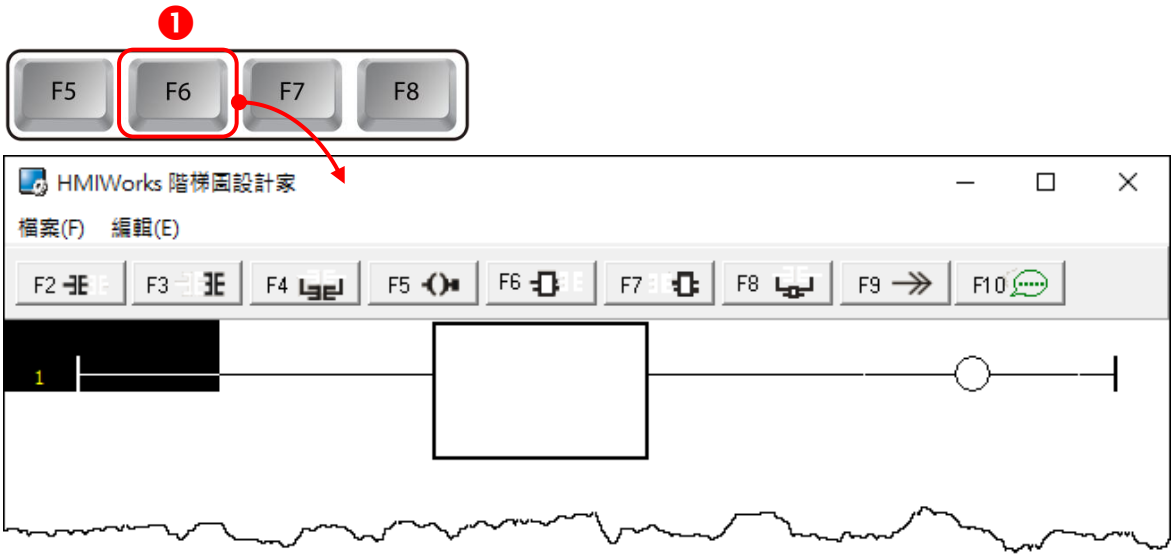
3.3.3.7 插入/刪除一個功能方塊

為了說明如何插入/刪除一個功能方塊 (Function Block)，我們將於下面一連串的步驟中，說明每一項功能。

選擇功能方塊的函數

1. 插入新的一階

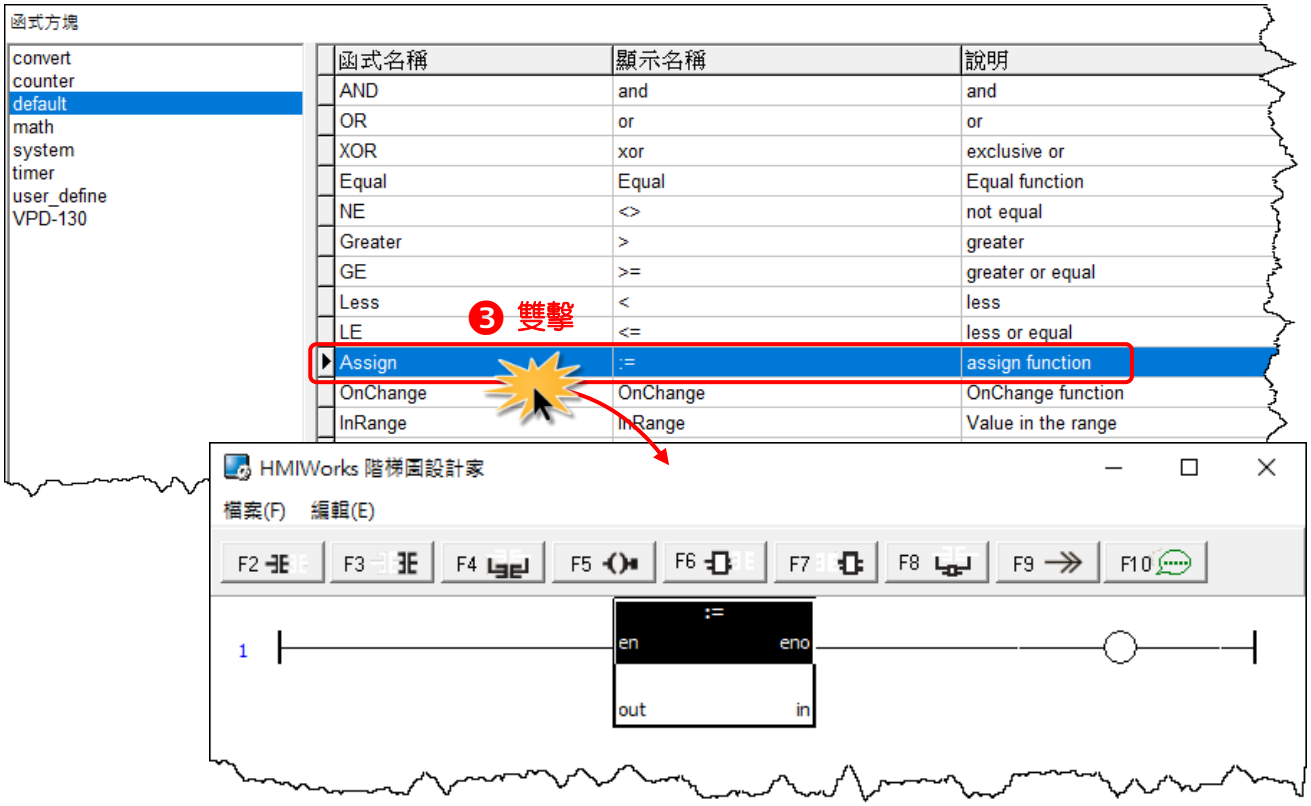
步驟 1: 按 **<F6>** **鍵**插入含一個功能方塊和一個線圈輸出的一階。



2. 選擇函數類別

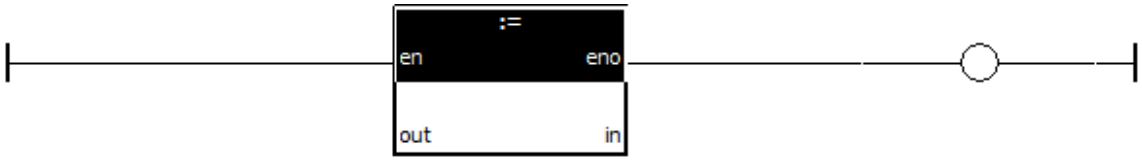
步驟 2: 雙擊功能方塊來開啟“函式方塊”配置視窗。

步驟 3: 雙擊您所需要的函式名稱。(如: 點選 **“default”** 群組後，再雙擊「**Assign (賦值)**」函式)



3. 設定該函數的變數

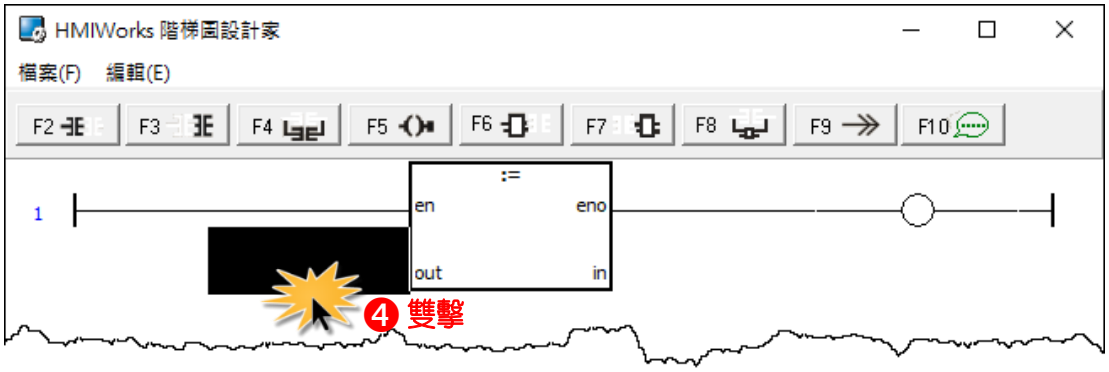
接著，應該設定函式「Assign (賦值)」的變數。如圖所示，共有四個變數 **en**、**eno**、**out**、**in**，說明如下：



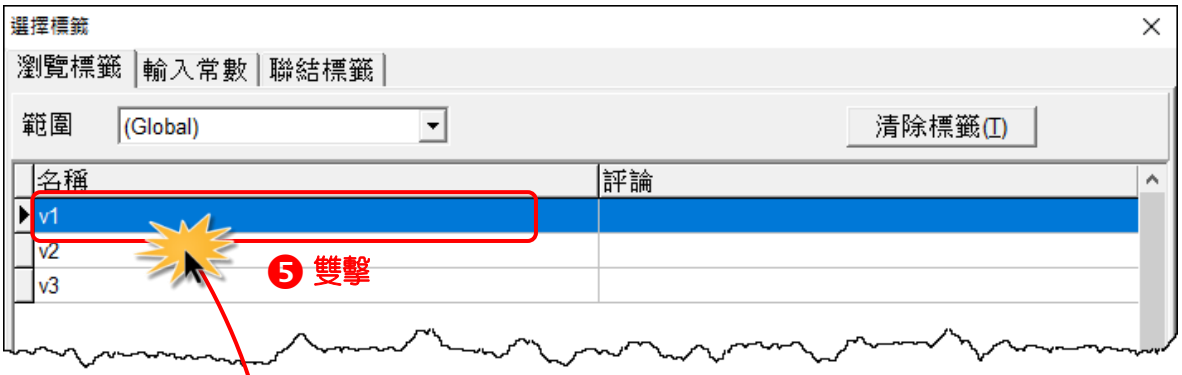
- **“en”** 和 **“eno”** 兩者都不可以和標籤建立關連。
- **“out”** 和 **“in”** 可以和標籤建立關連。用前面提過的 “新增虛擬標籤” 來建立標籤的關連。

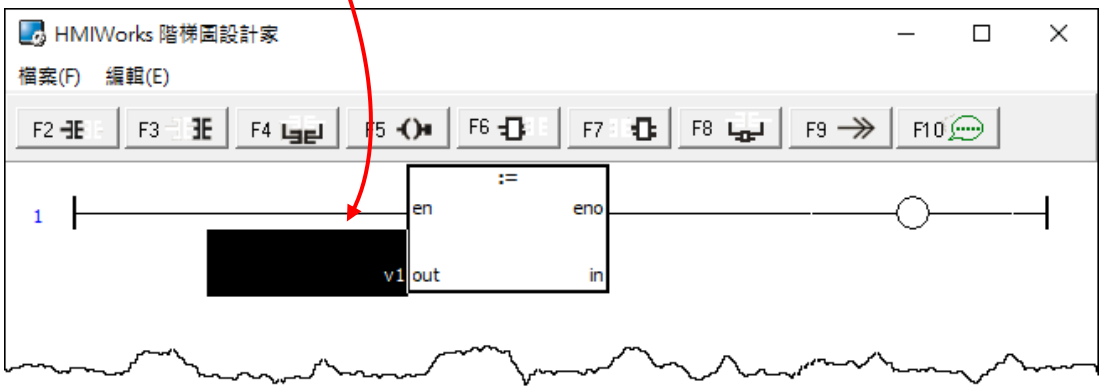
例如: 我們可以將 “v1”和 “out”、“v2” 和 “in” 建立關連。v1、v2 和 v3 是 第 3.3.3.1 節 新增虛擬標籤 (F2) 所新增的。

步驟 4: 移動游標到 **“out”** 的旁邊 (在功能方塊外，如下圖所示)。雙擊緊鄰 **“out”** 旁的空白處來開啟“選擇標籤”視窗。



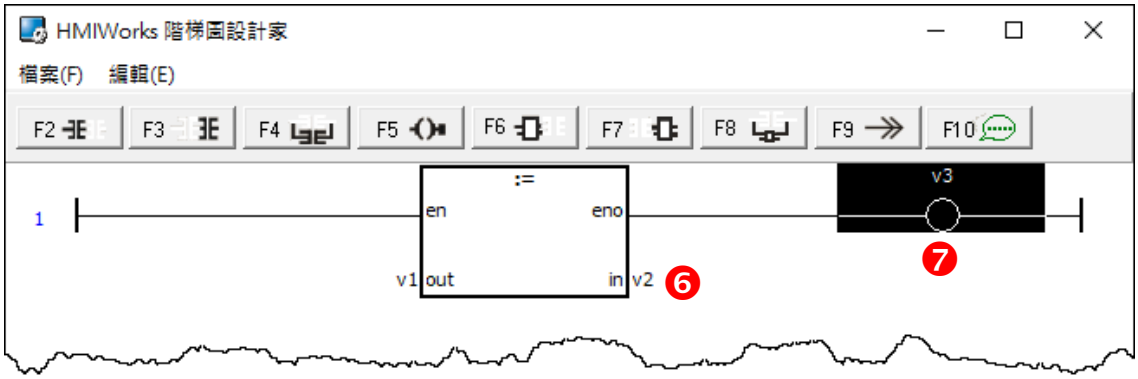
步驟 5: 雙擊點選清單中的標籤名稱來建立該標籤和 **“out”** 的關連。範例:雙擊點選標籤 **“v1”** 來建立和 **“out”** 的關連。 **“out”** 是「**Assign (賦值)**」函式的一個變數。





步驟 6: 用相同的方法來建立 **“v2”** 和 **“in”** 的關連。 **“in”** 是「**Assign (賦值)**」函式的一個變數。

步驟 7: 最後，建立 **“v3”** 和線圈輸出的關連。

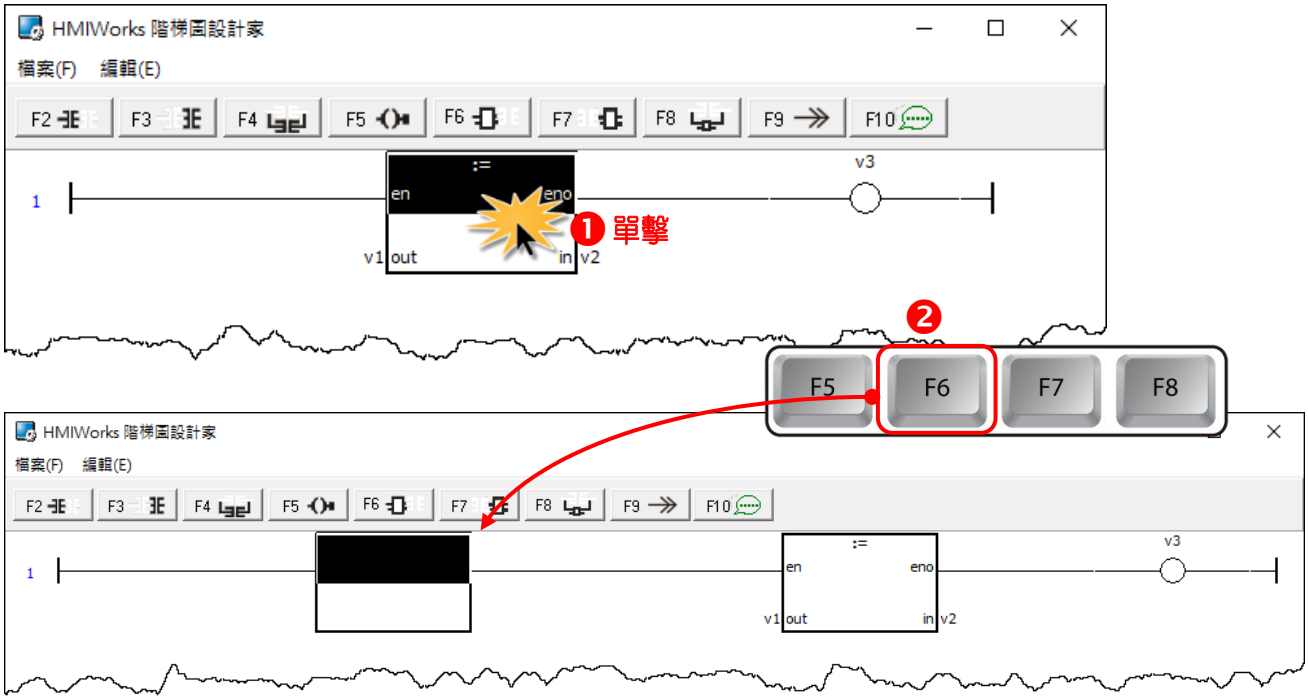


「Assign (賦值)」函式在當 **“en”** 是 **ON** 時，會將 **“v2”** 的值賦予 **“v1”**。
線圈輸出 **“v3”** 的狀態則決定於 **“eno”**。

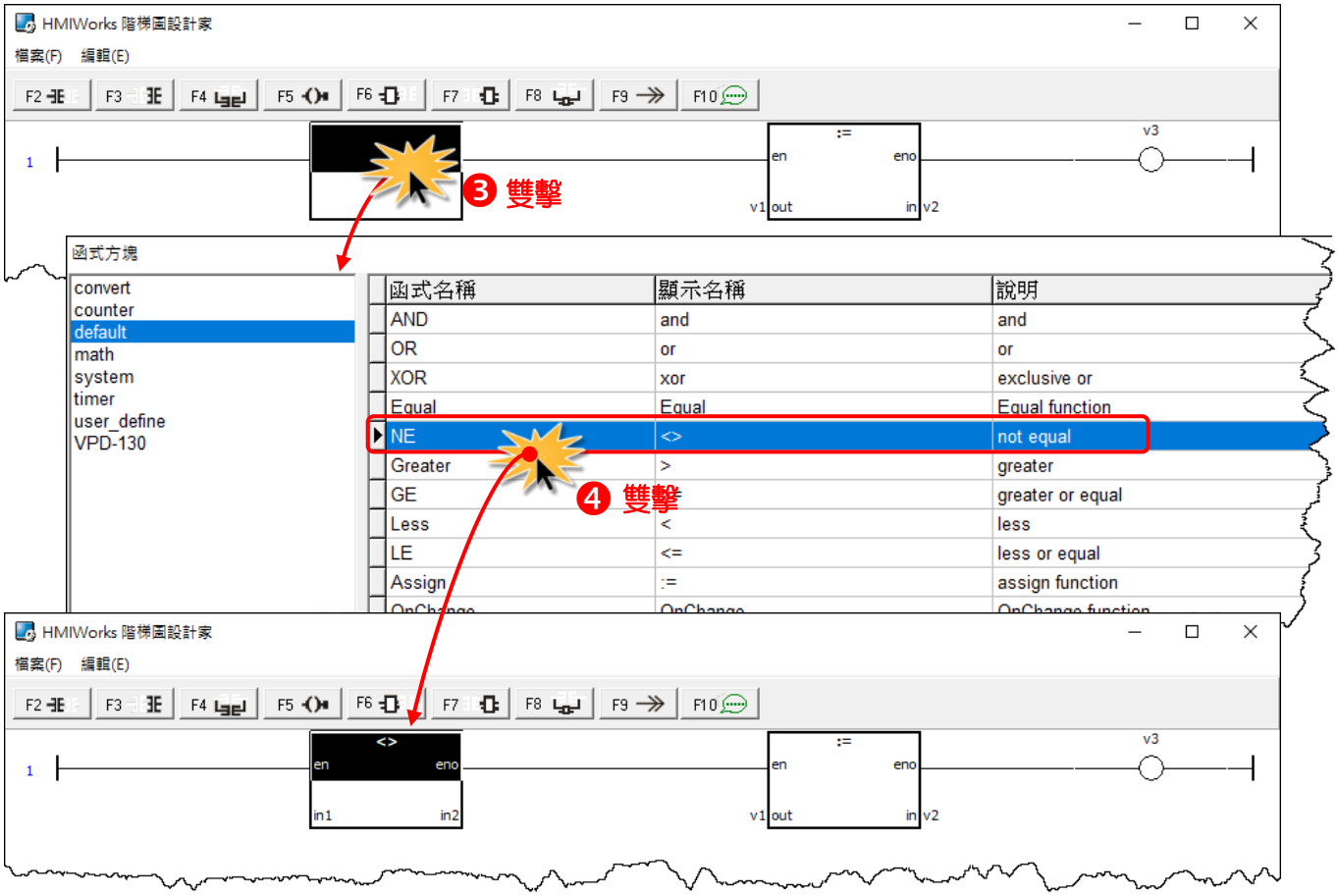
在游標左方插入一個新的功能方塊(F6)

步驟 1: 將游標移到「**Assign (賦值)**」的功能方塊上。

步驟 2: 按 **<F6> 鍵**。

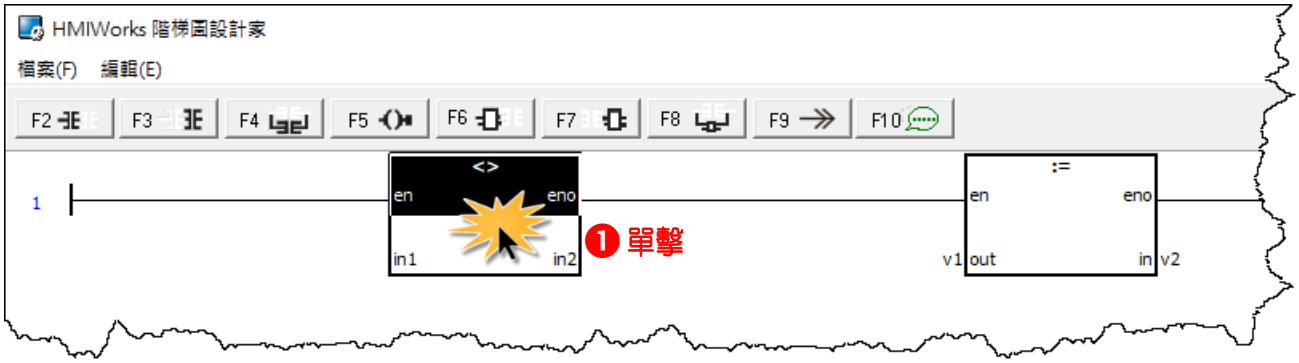


步驟 3: 將剛新插入的功能方塊函式的類別設為「**NE (不等於)**」。

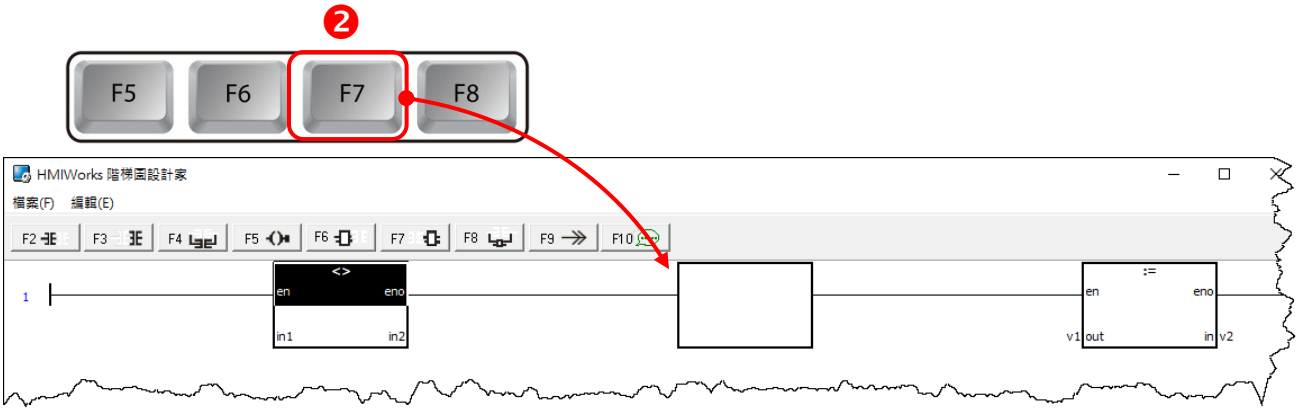


在游標右方插入一個新的功能方塊(F7)

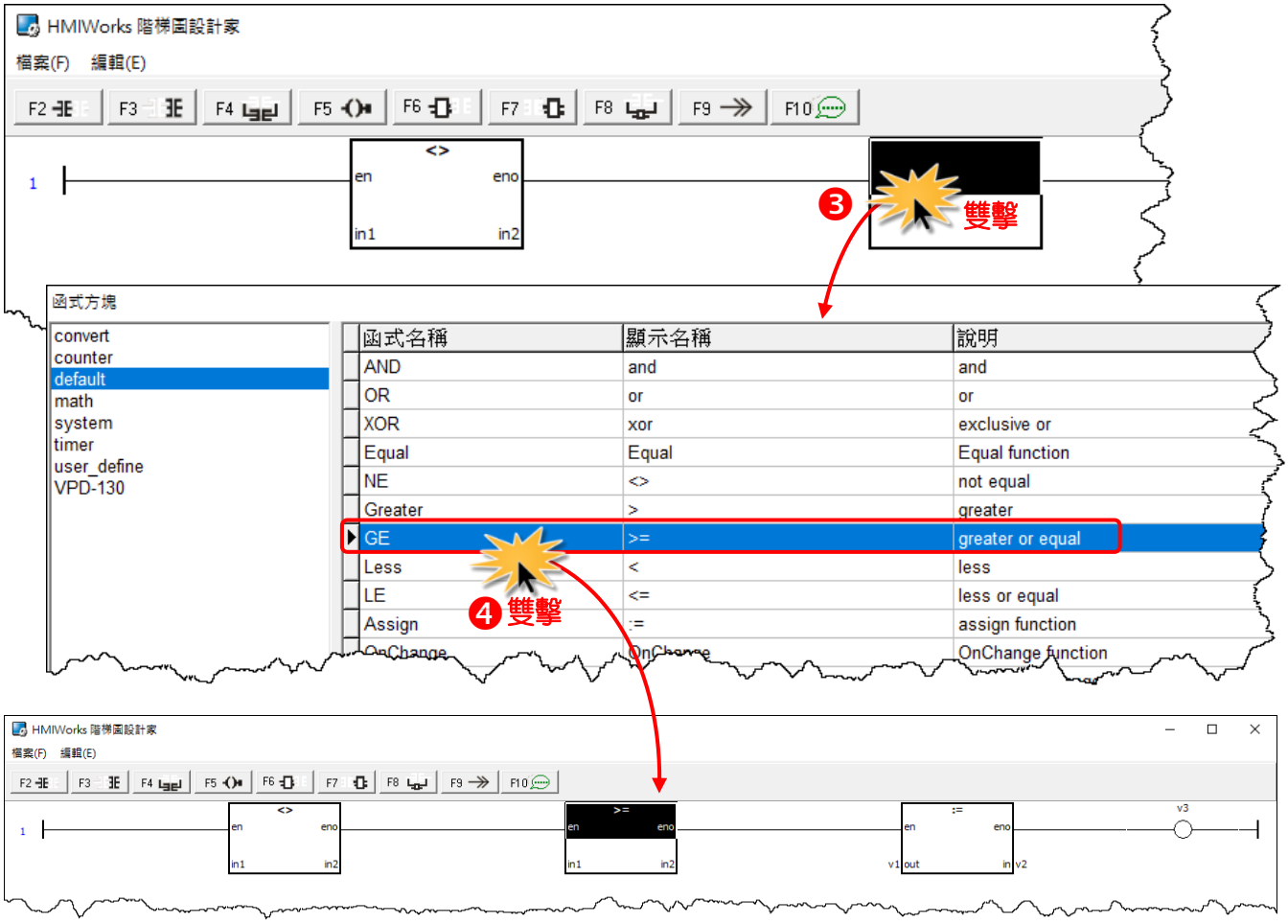
步驟 1: 將游標移到「**NE (不等於)**」的功能方塊上。



步驟 2: 按 **<F7> 鍵**。

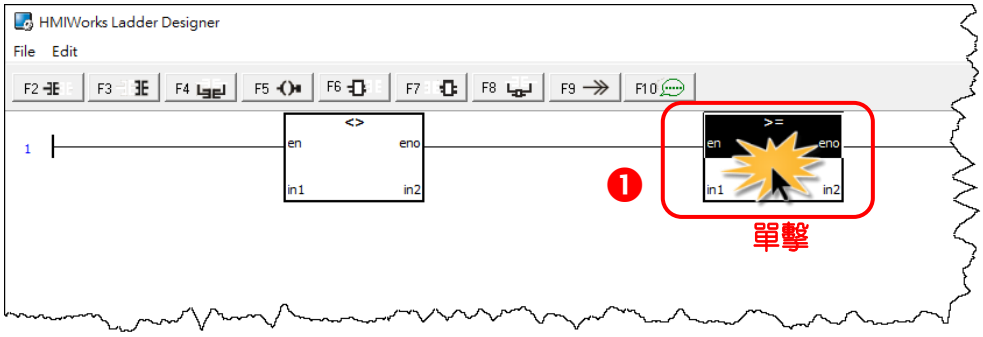


步驟 3: 將剛新插入的功能方塊函式的類別設為「**GE (大於或等於)**」。

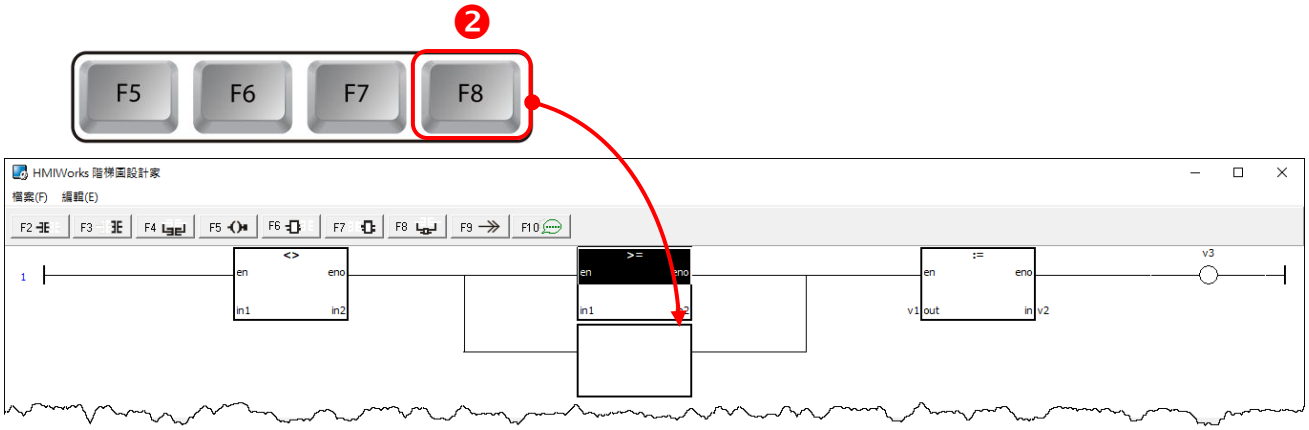


在游標平行處插入一個新的功能方塊(F8)

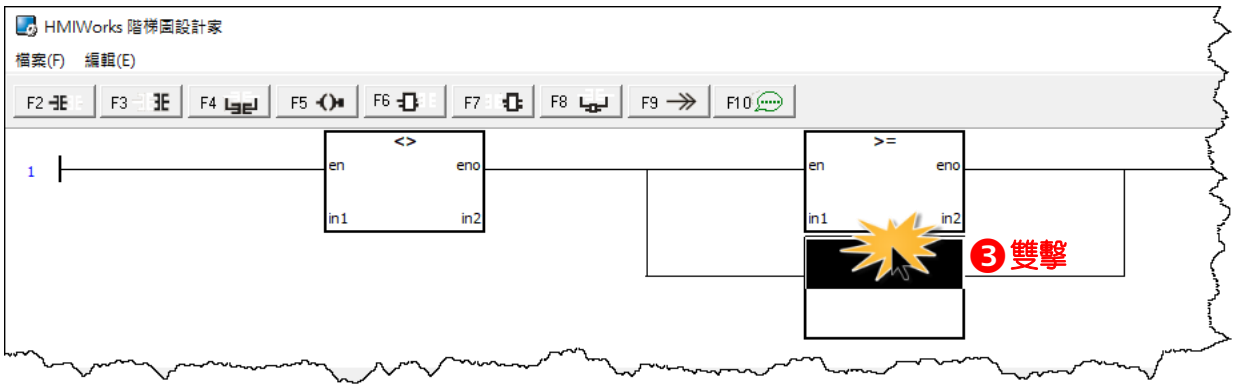
步驟 1: 將游標移到「**GE (大於或等於)**」的功能方塊上。

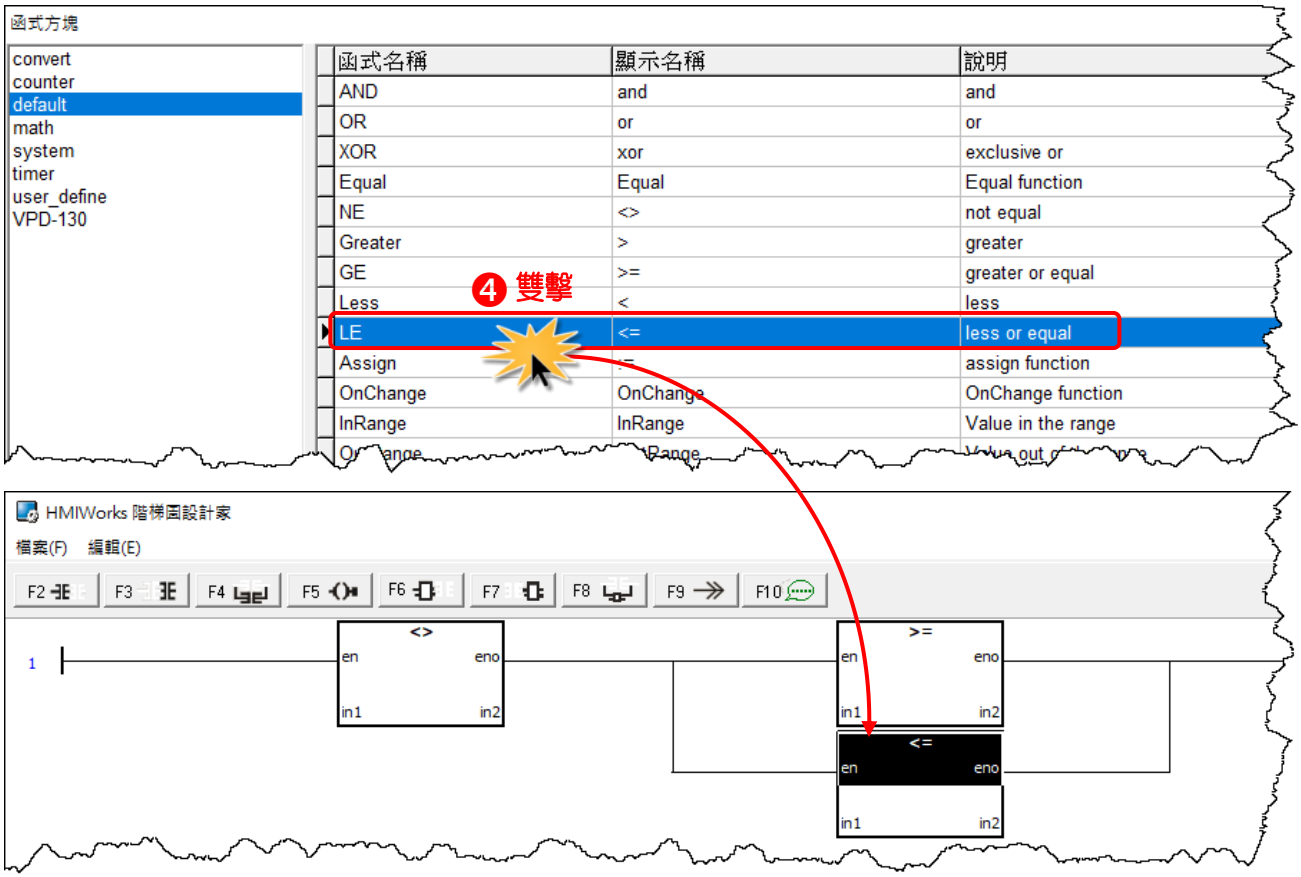


步驟 2: 按 **<F8> 鍵**。



步驟 3: 將剛新插入的功能方塊函數的類別設為「**LE (小於或等於)**」。

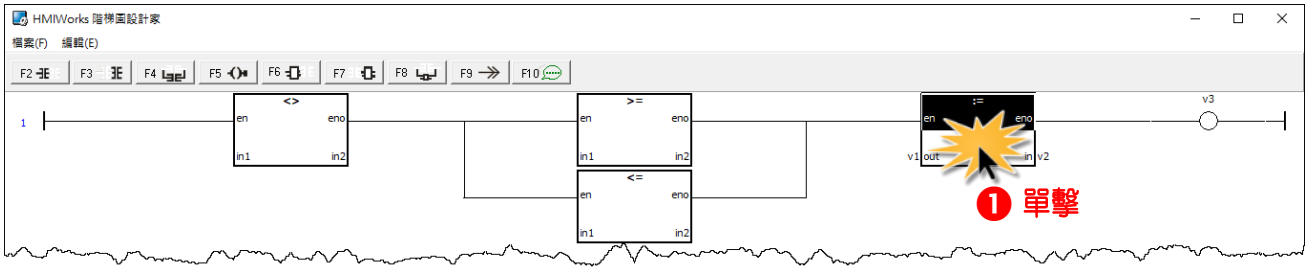




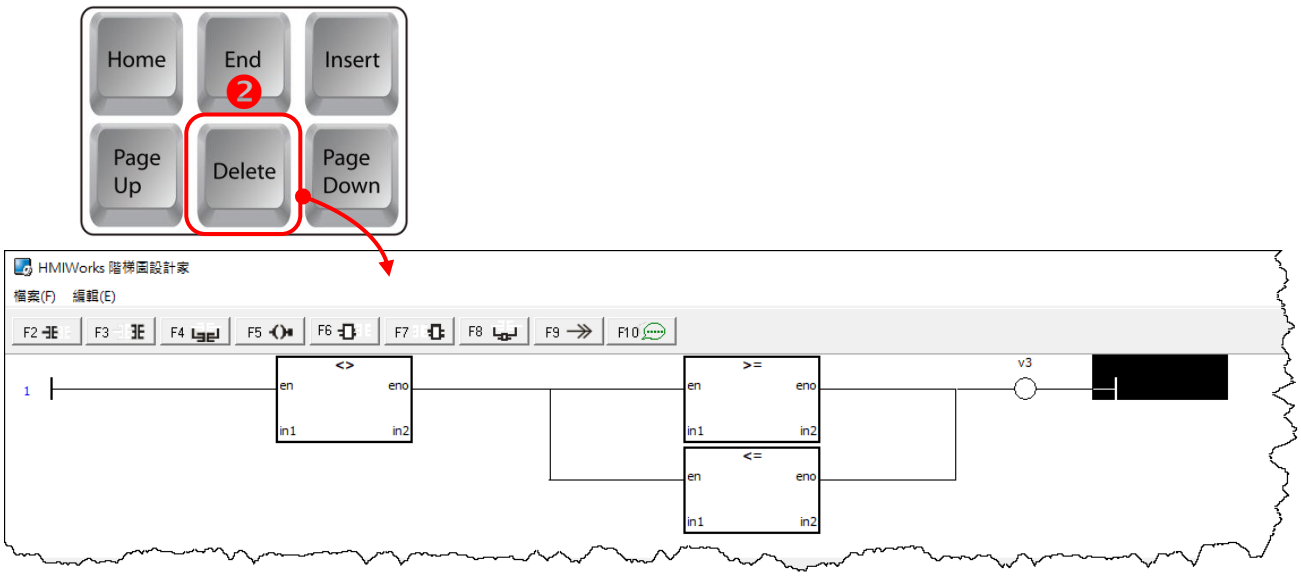
刪除一個功能方塊

移動游標到要刪除的功能方塊上，然後按 **<Delete> 鍵**。

步驟 1: 將游標移動到「**Assign (賦值)**」的功能方塊上。

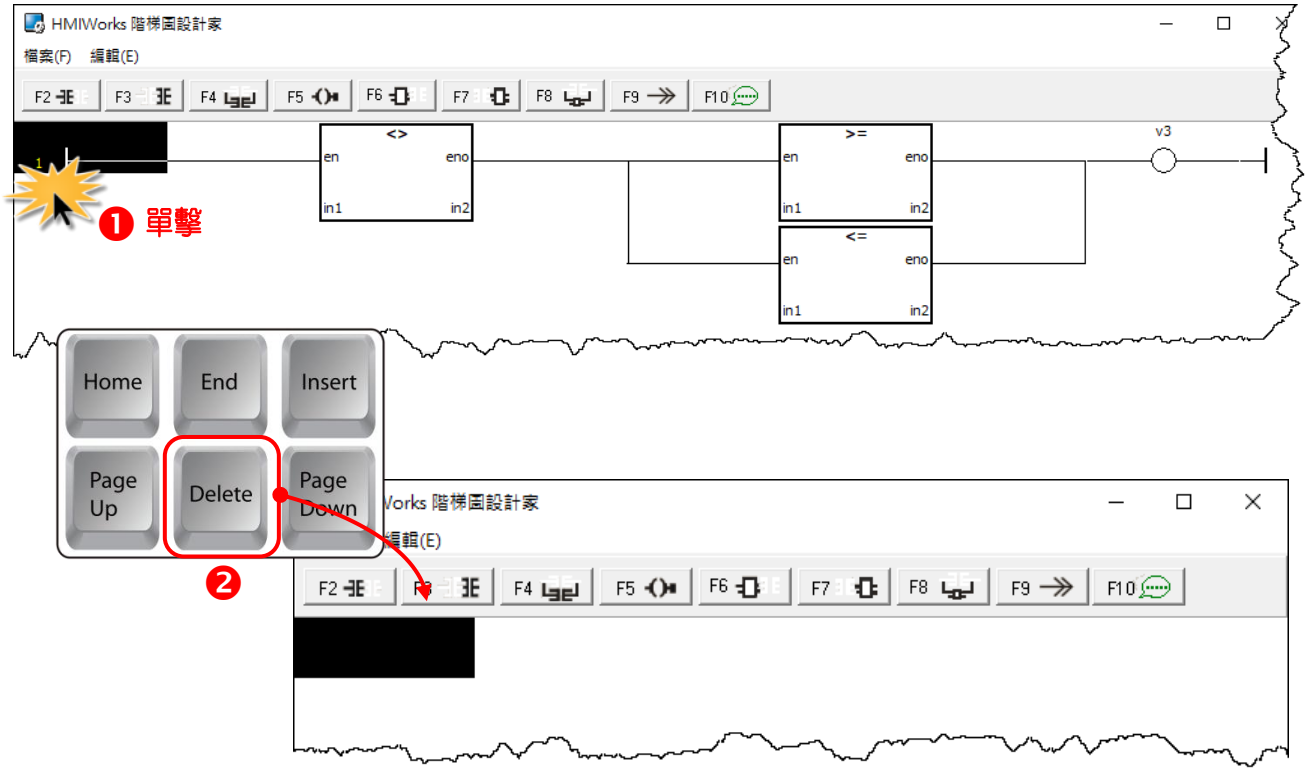


步驟 2: 按下 **<Delete> 鍵**。



刪除一個階

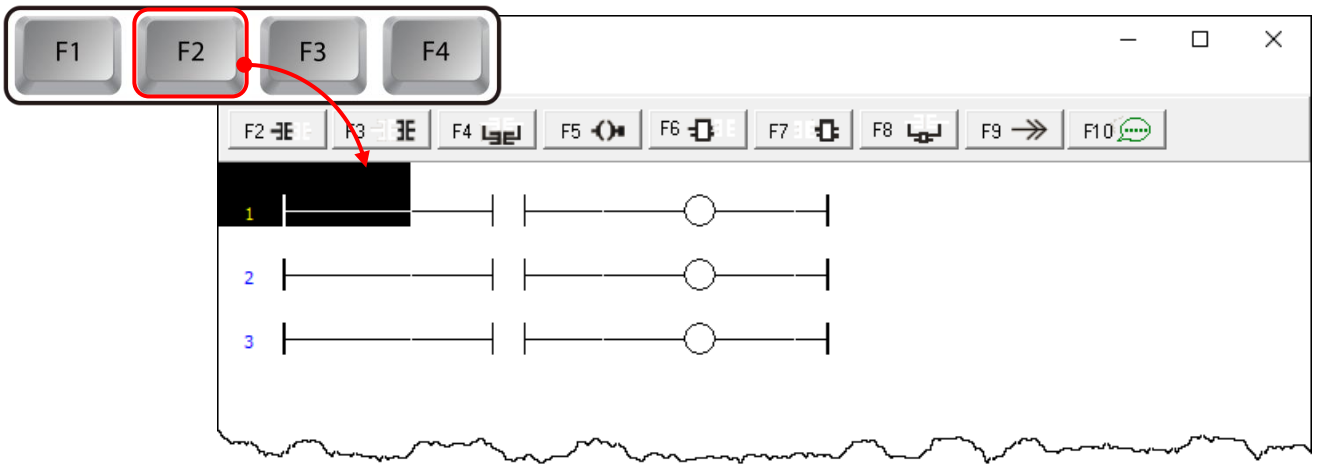
先將游標移動到該階的起始點，再按下 **<Delete> 鍵**。



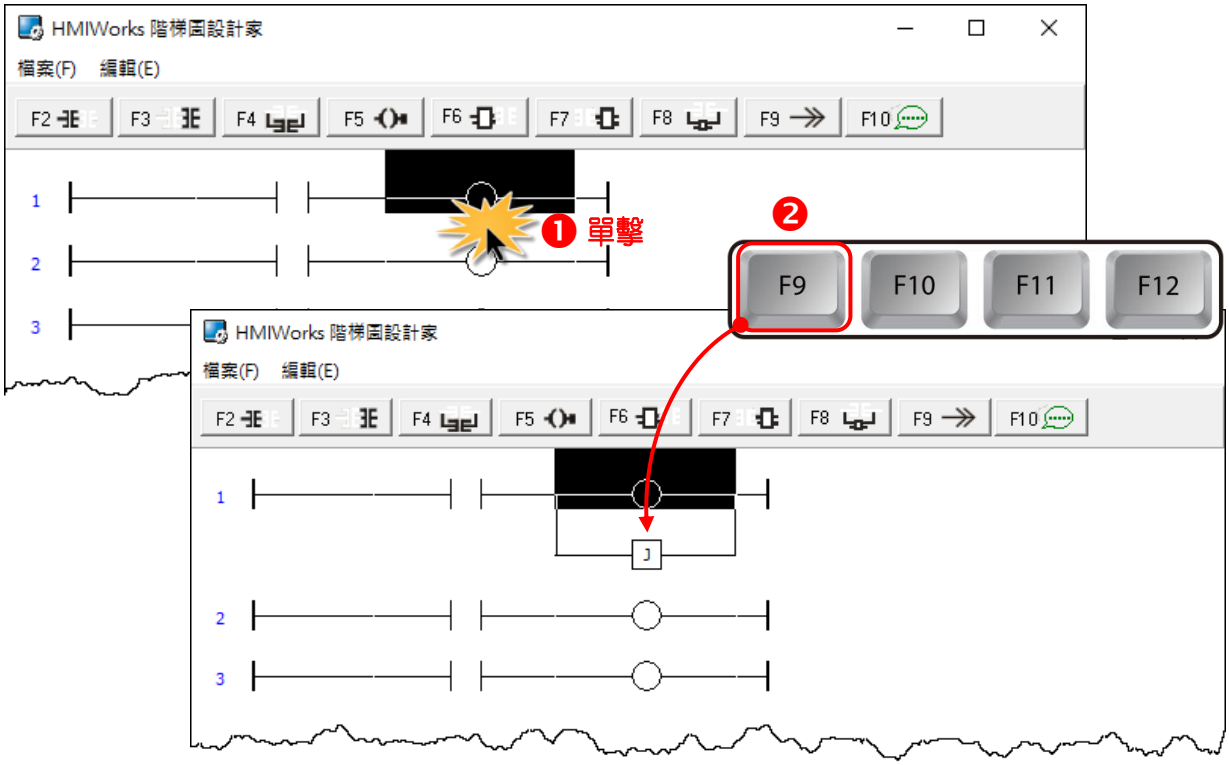
3.3.3.8 跳出至另一階

為了說明如何跳出至另一階，首先先畫出三個階，然後說明如何作出跳過第二階的方法。

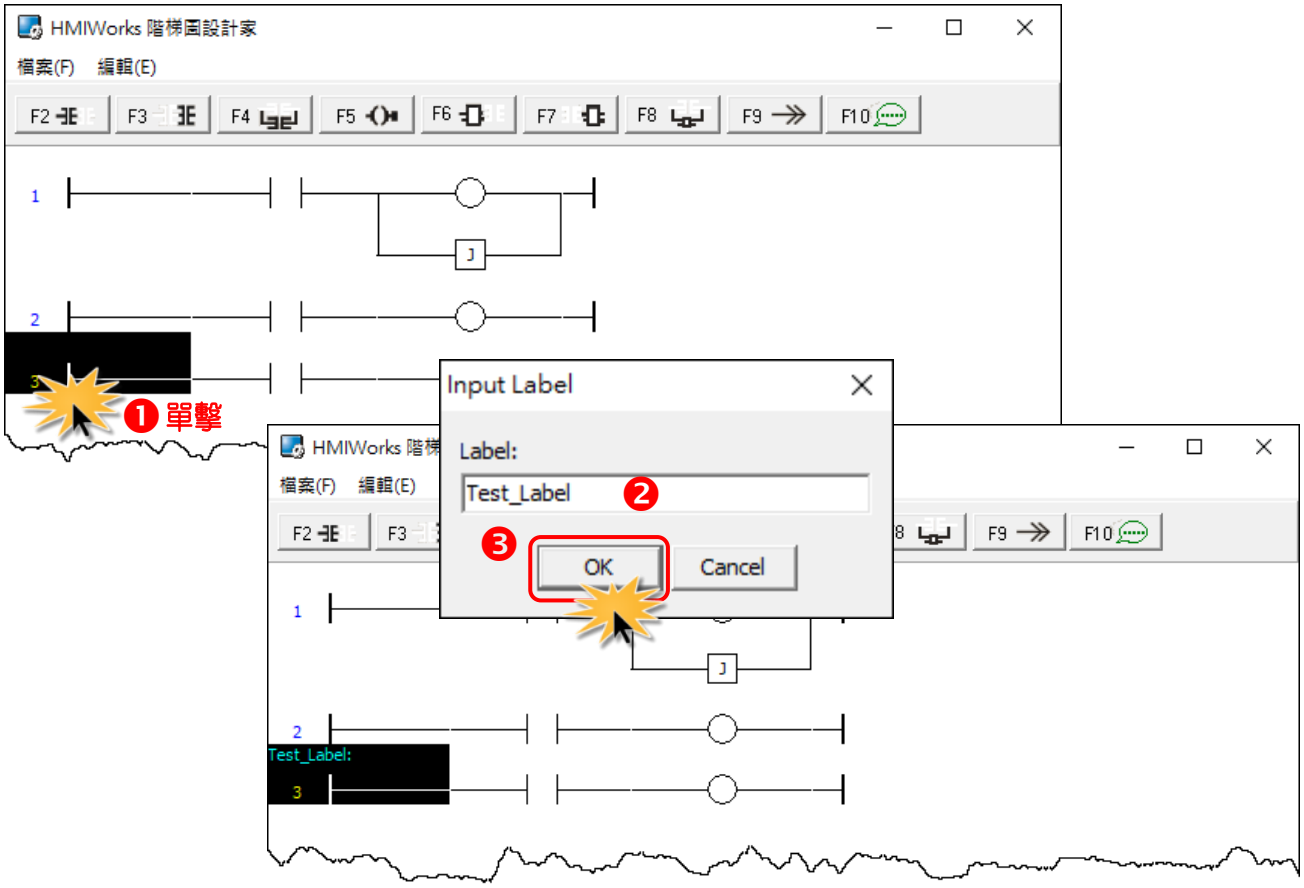
步驟 1: 按 **<F2> 鍵**三次以產生三個階。



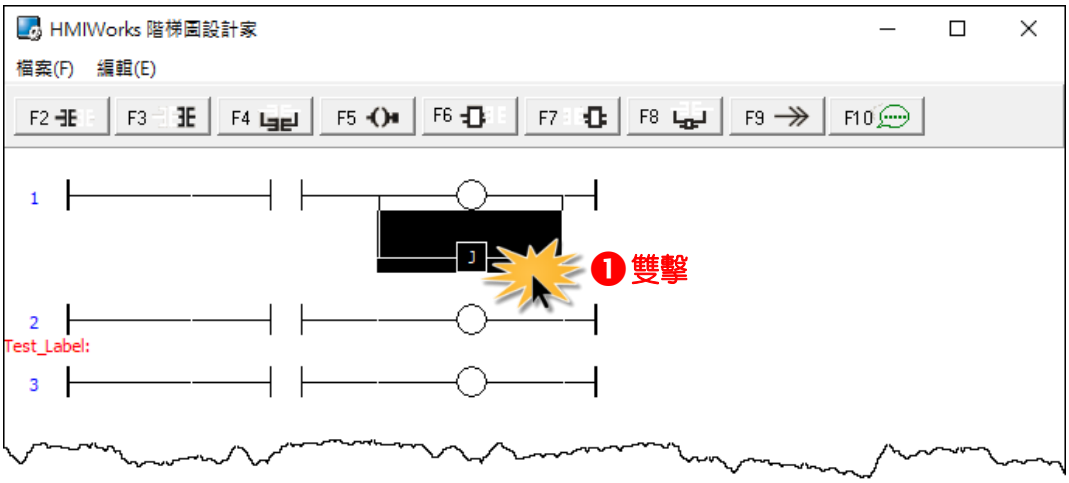
步驟 2: 將游標移到第一階的線圈輸出 (Coil Output)，再按 **<F9> 鍵**產生一個平行線圈輸出的「 **J (跳出)**」。

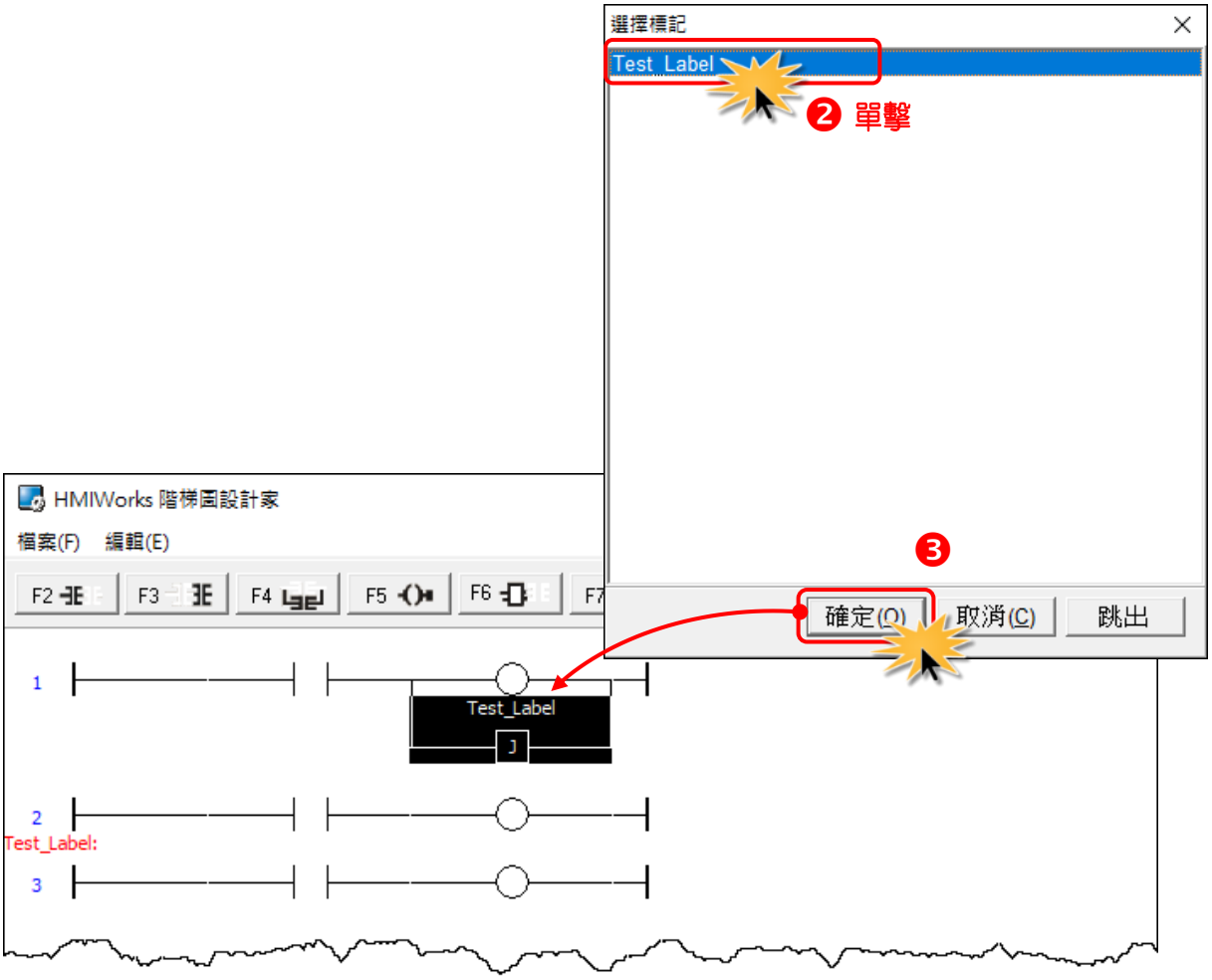


步驟 3: 雙擊第三階的起始點，並給第三階命名新名字「**Test_Label**」。



步驟 4: 雙擊第一階線圈輸出平行的「**J (跳出)**」，並把它關連到第三階的名字「Test_Label」。





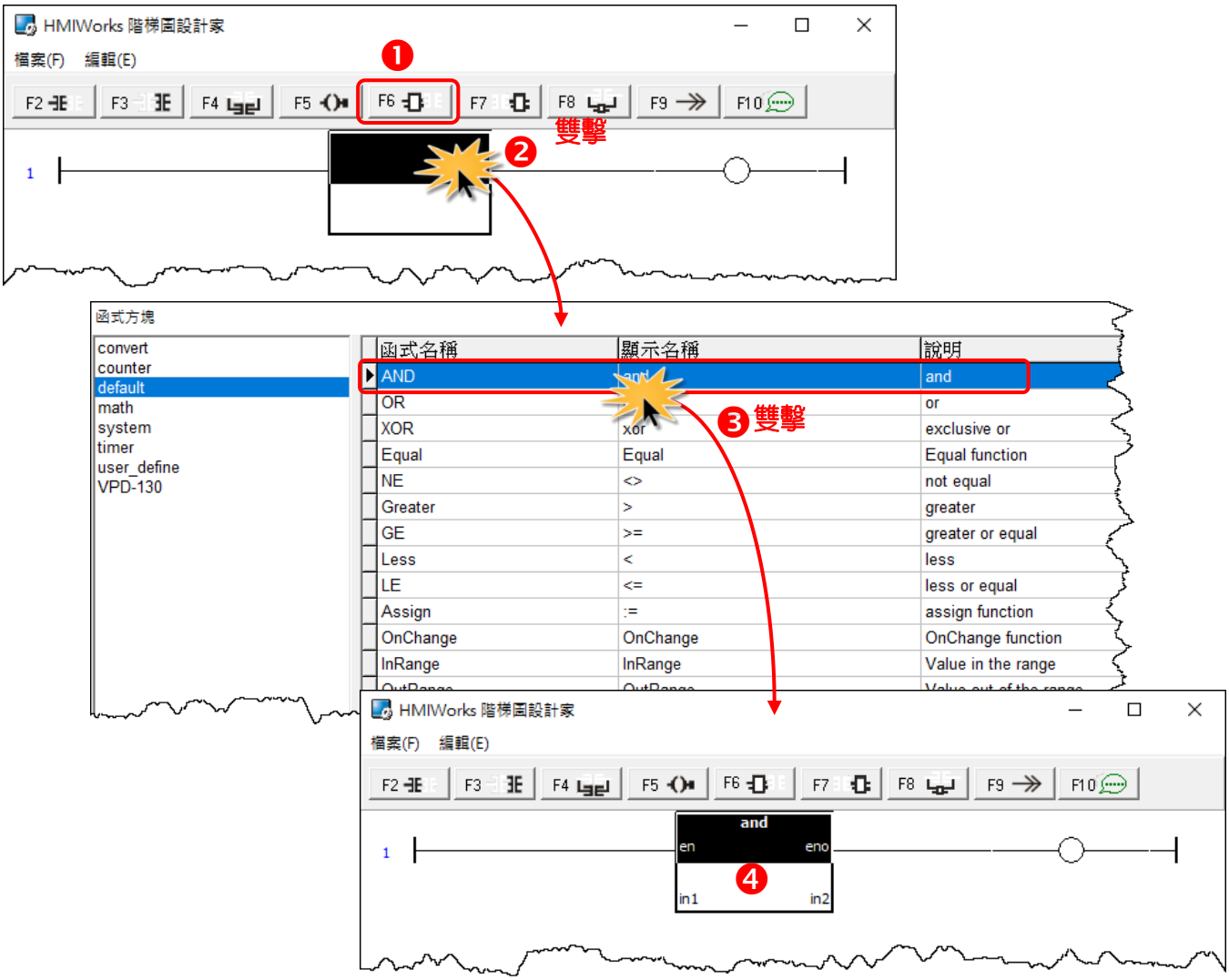
步驟 5: 當階梯圖被執行時，如第一階的開關輸入是關上的，那第一階的線圈輸出就設為 High，然後跳過第二階，直接執行第三階。

3.3.4 功能方塊 (Function Block)

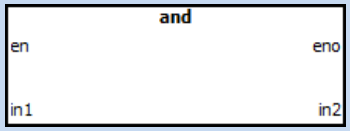
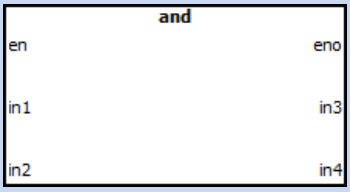
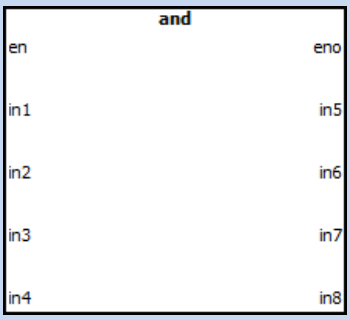
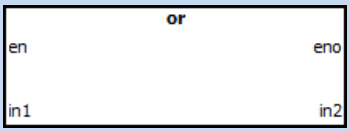
階梯圖設計家提供各種功能方塊，包含數學、轉換、計數器，定時器、系統功能及程式碼…等，讓使用者方便、快速應用。
更多詳細內容請參考 “**C:\ICPDAS\HMIWorks_Standard\bin\FunctionBlock**”。

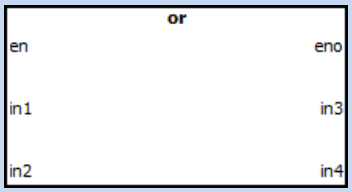
範例:

步驟 1: 按下 F6 按鈕來新增一個功能方塊。
步驟 2: 雙擊此功能方塊來開啟“函式方塊”配置視窗。
步驟 3: 雙擊您所需要的函式名稱 (如: AND)。
步驟 4: 將開啟 **“AND”** 功能方塊。



3.3.4.1 Default 群組

<table>
<tr><th>函式方塊</th><th>說明</th></tr>
<tr><td> 
</td><td>
AND (邏輯運算: 且)
➢ 參數:
in1: [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤
➢ 範例:
<table>
<tr><th colspan="2">Input</th><th>Output</th></tr>
<tr><th>in1</th><th>in2</th><th>eno</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
➢ 虛擬碼:
If en == 1
eno = in1 & in2;
Else eno = 0;
</td></tr>
<tr><td> 
</td><td>
AND4
➢ 參數:
in1: [Input] 輸入值/標籤 in3: [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤 in4: [Input] 輸入值/標籤
➢ 虛擬碼:
If en == 1
eno = in1 & in2 & in3 & in4;
Else eno = 0;
</td></tr>
<tr><td> 
</td><td>
AND8
➢ 參數:
in1: [Input] 輸入值/標籤 in5: [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤 in6: [Input] 輸入值/標籤
in3: [Input] 輸入值/標籤 in7: [Input] 輸入值/標籤
in4: [Input] 輸入值/標籤 in8: [Input] 輸入值/標籤
➢ 虛擬碼:
If en == 1
eno = in1 & in2 & in3 & in4 & in5 & in6 & in7 & in8;
Else eno = 0;
</td></tr>
<tr><td> 
</td><td>
OR (邏輯運算: 或)
➢ 參數:
in1: [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤
➢ 範例:
<table>
<tr><th colspan="2">Input</th><th>Output</th></tr>
<tr><th>in1</th><th>in2</th><th>eno</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
➢ 虛擬碼:
If en == 1
eno = in1 | in2;
Else eno = 0;
</td></tr>
</table>

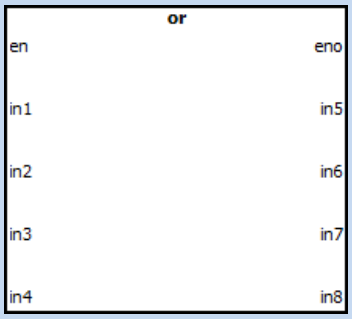
<table>
<tr><th>函式方塊</th><th>說明</th></tr>
<tr><td> 
</td><td>

OR4

➢ **參數:**

in1: [Input] 輸入值/標籤 **in3:** [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤 **in4:** [Input] 輸入值/標籤

➢ **虛擬碼:**

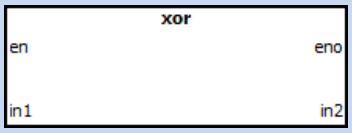
If en == 1
  eno = in1 | in2 | in3 | in4;
Else eno = 0;
</td></tr>
<tr><td> 
</td><td>

OR8

➢ **參數:**

in1: [Input] 輸入值/標籤 **in5:** [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤 **in6:** [Input] 輸入值/標籤
in3: [Input] 輸入值/標籤 **in7:** [Input] 輸入值/標籤
in4: [Input] 輸入值/標籤 **in8:** [Input] 輸入值/標籤

➢ **虛擬碼:**

If en == 1
  eno = in1 | in2 | in3 | in4 | in5 | in6 | in7 | in8;
Else eno = 0;
</td></tr>
<tr><td> 
</td><td>

XOR (邏輯運算: 互斥或)

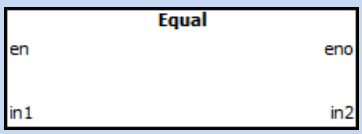
➢ **參數:**

in1: [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤

➢ **範例:**

Input		Output
in1	in2	eno
0	0	0
0	1	1
1	0	1
1	1	0

➢ **虛擬碼:**

If en == 1
  eno = in1 ^ in2;
Else eno = 0;
</td></tr>
<tr><td> 
</td><td>

Equal (數學符號: 等於)

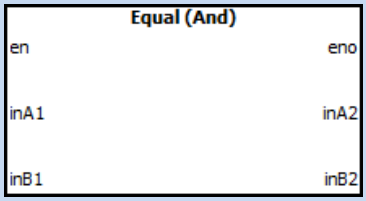
➢ **參數:**

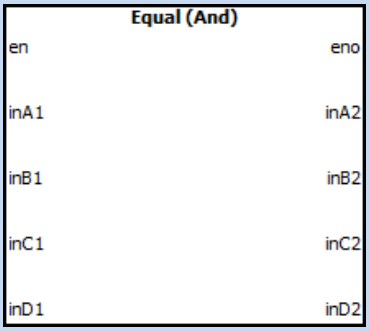
in1: [Input] 輸入值/標籤
in2: [Input] 輸入值/標籤

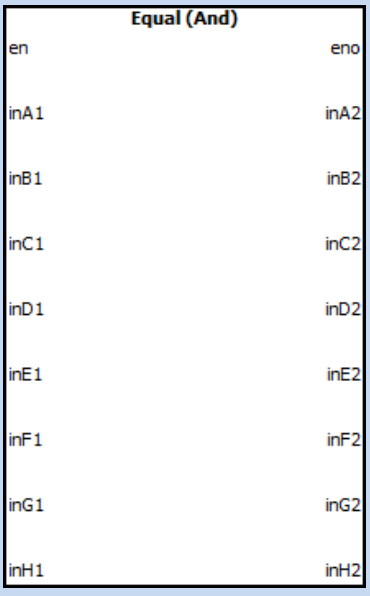
➢ **虛擬碼:**

If (en == 1 and in1 is equal to in2)
  eno = 1;
Else eno = 0;
</td></tr>
</table>

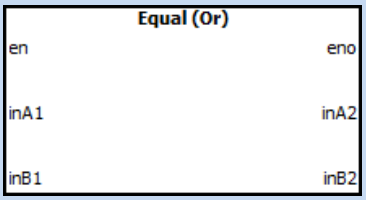
函式方塊	說明

	**Equal_And2**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	➢ **虛擬碼**:
	If (en == 1 and
	inA1 is equal to inA2 and inB1 is equal to inB2)
	eno = 1;
	Else eno = 0;

	**Equal_And4**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
	inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
	➢ **虛擬碼**:
	If (en == 1 and
	inA1 is equal to inA2 and inB1 is equal to inB2 and
	inC1 is equal to inC2 and inD1 is equal to inD2)
	eno = 1;
	Else eno = 0;

	**Equal_And8**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
	inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
	inE1: [Input] 輸入值/標籤 **inE2:** [Input] 輸入值/標籤
	inF1: [Input] 輸入值/標籤 **inF2:** [Input] 輸入值/標籤
	inG1: [Input] 輸入值/標籤 **inG2:** [Input] 輸入值/標籤
	inH1: [Input] 輸入值/標籤 **inH2:** [Input] 輸入值/標籤
	➢ **虛擬碼**:
	If (en == 1 and
	inA1 is equal to inA2 and inB1 is equal to inB2 and
	inC1 is equal to inC2 and inD1 is equal to inD2 and
	inE1 is equal to inE2 and inF1 is equal to inF2 and
	inG1 is equal to inG2 and inH1 is equal to inH2)
	eno = 1;
	Else eno = 0;

函式方塊	說明



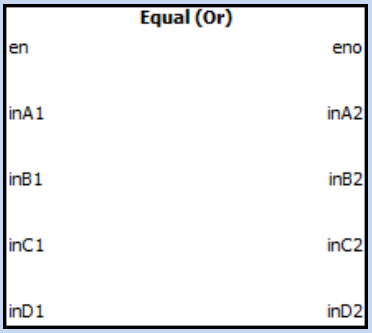
Equal_Or2

- **參數:**

inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤

- **虛擬碼:**

```
If (en == 1 and
   inA1 is equal to inA2 or inB1 is equal to inB2)
      eno = 1;
Else eno = 0;
```



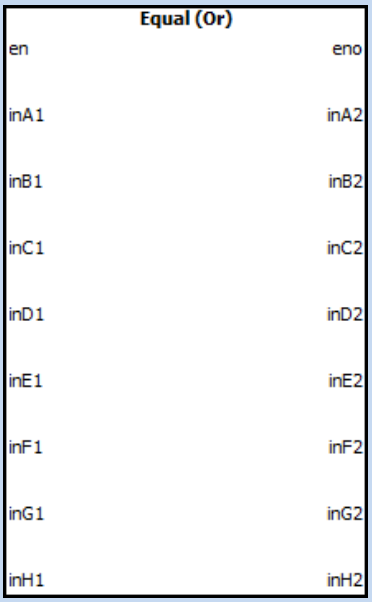
Equal_Or4

- **參數:**

inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤

- **虛擬碼:**

```
If (en == 1 and
   inA1 is equal to inA2 or inB1 is equal to inB2 or
   inC1 is equal to inC2 or inD1 is equal to inD2)
      eno = 1;
Else eno = 0;
```



Equal_Or8

- **參數:**

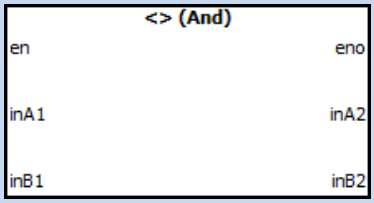
inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
inE1: [Input] 輸入值/標籤 **inE2:** [Input] 輸入值/標籤
inF1: [Input] 輸入值/標籤 **inF2:** [Input] 輸入值/標籤
inG1: [Input] 輸入值/標籤 **inG2:** [Input] 輸入值/標籤
inH1: [Input] 輸入值/標籤 **inH2:** [Input] 輸入值/標籤

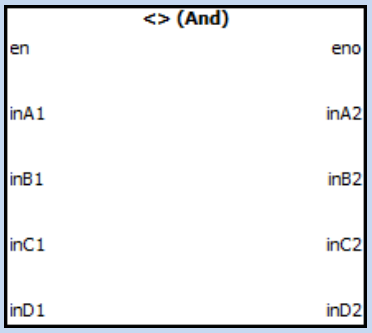
- **虛擬碼:**

```
If (en == 1 and
   inA1 is equal to inA2 or inB1 is equal to inB2 or
   inC1 is equal to inC2 or inD1 is equal to inD2 or
   inE1 is equal to inE2 or inF1 is equal to inF2 or
   inG1 is equal to inG2 or inH1 is equal to inH2)
      eno = 1;
Else eno = 0;
```

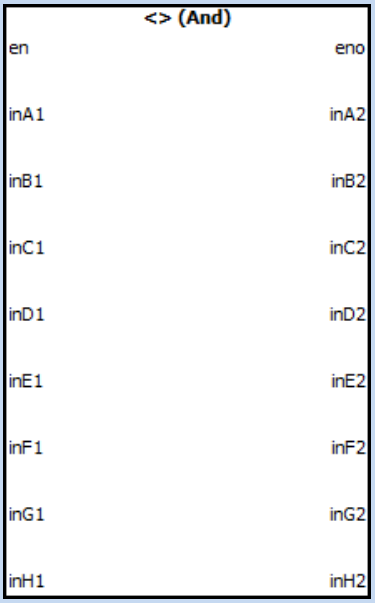
函式方塊	說明

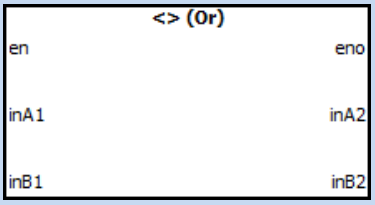
	**NE** (數學符號: 不等於)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and in1 is not equal to in2)
	eno = 1;
	Else eno = 0;

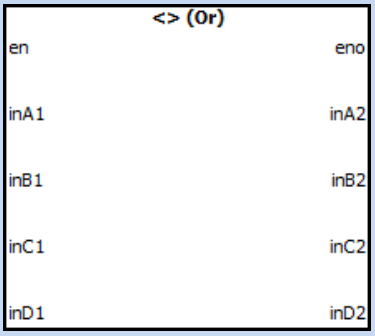
	**NE_And2**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and
	inA1 is not equal to inA2 and
	inB1 is not equal to inB2)
	eno = 1;
	Else eno = 0;

	**NE_And4**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
	inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and
	inA1 is not equal to inA2 and
	inB1 is not equal to inB2 and
	inC1 is not equal to inC2 and
	inD1 is not equal to inD2)
	eno = 1;
	Else eno = 0;

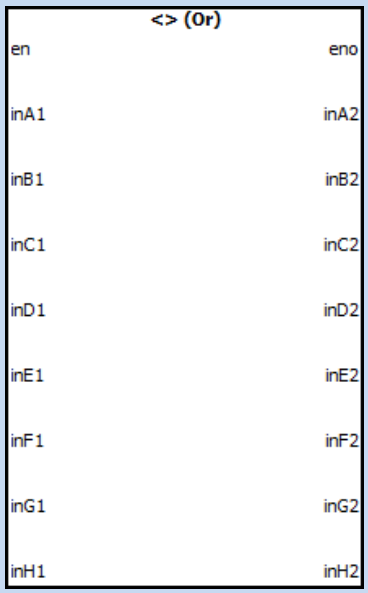
函式方塊	說明

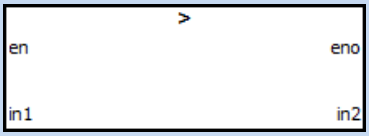
	**NE_And8**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
	inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
	inE1: [Input] 輸入值/標籤 **inE2:** [Input] 輸入值/標籤
	inF1: [Input] 輸入值/標籤 **inF2:** [Input] 輸入值/標籤
	inG1: [Input] 輸入值/標籤 **inG2:** [Input] 輸入值/標籤
	inH1: [Input] 輸入值/標籤 **inH2:** [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and
	inA1 is not equal to inA2 and inB1 is not equal to inB2 and
	inC1 is not equal to inC2 and inD1 is not equal to inD2 and
	inE1 is not equal to inE2 and inF1 is not equal to inF2 and
	inG1 is not equal to inG2 and inH1 is not equal to inH2)
	eno = 1;
	Else eno = 0;

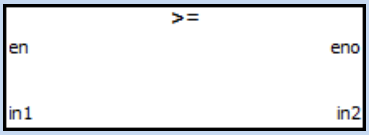
	**NE_Or2**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and
	inA1 is not equal to inA2 or inB1 is not equal to inB2)
	eno = 1;
	Else eno = 0;

	**NE_Or4**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
	inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and
	inA1 is not equal to inA2 or inB1 is not equal to inB2 or
	inC1 is not equal to inC2 or inD1 is not equal to inD2)
	eno = 1;
	Else eno = 0;

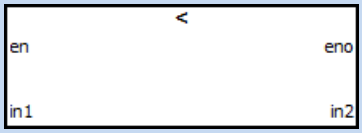
函式方塊	說明

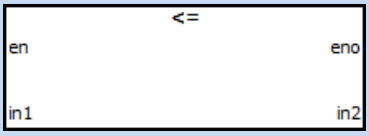
	**NE_Or8**
	➢ **參數:**
	inA1: [Input] 輸入值/標籤 **inA2:** [Input] 輸入值/標籤
	inB1: [Input] 輸入值/標籤 **inB2:** [Input] 輸入值/標籤
	inC1: [Input] 輸入值/標籤 **inC2:** [Input] 輸入值/標籤
	inD1: [Input] 輸入值/標籤 **inD2:** [Input] 輸入值/標籤
	inE1: [Input] 輸入值/標籤 **inE2:** [Input] 輸入值/標籤
	inF1: [Input] 輸入值/標籤 **inF2:** [Input] 輸入值/標籤
	inG1: [Input] 輸入值/標籤 **inG2:** [Input] 輸入值/標籤
	inH1: [Input] 輸入值/標籤 **inH2:** [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and
	inA1 is not equal to inA2 or
	inB1 is not equal to inB2 or
	inC1 is not equal to inC2 or
	inD1 is not equal to inD2 or
	inE1 is not equal to inE2 or
	inF1 is not equal to inF2 or
	inG1 is not equal to inG2 or
	inH1 is not equal to inH2)
	eno = 1;
	Else eno = 0;

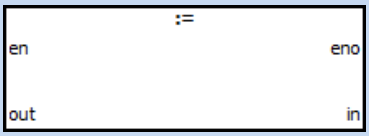
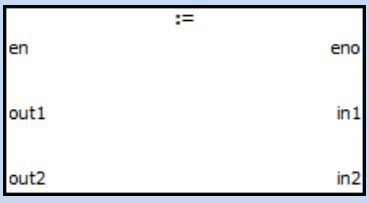
	**Greater** (數學符號: 大於)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and in1 > in2)
	eno = 1;
	Else eno = 0;

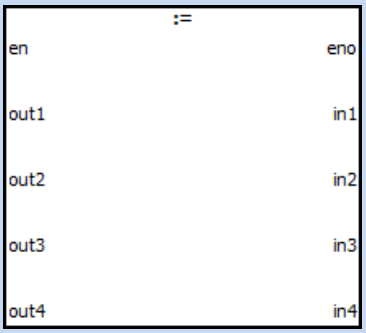
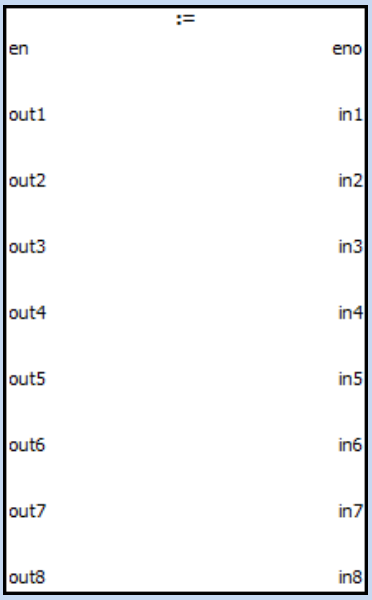
	**GE** (數學符號: 大於或等於)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and in1 >= in2)
	eno = 1;
	Else eno = 0;

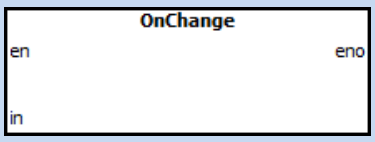
函式方塊	說明

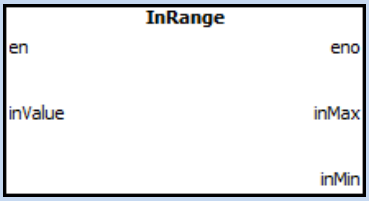
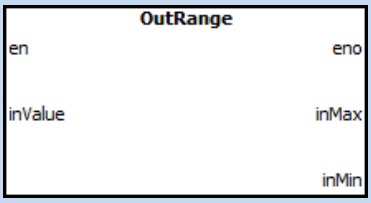
	**Less** (數學符號: 小於)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and in1 < in2)
	eno = 1;
	Else eno = 0;

	**LE** (數學符號: 小於或等於)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If (en == 1 and in1 <= in2)
	eno = 1;
	Else eno = 0;

	**Assign** (數學符號: 賦值，如下: 將 “in” 的值指定給 “out”) ➢ **參數:** **out:** [Output] 標籤 **in:** [Input] 值/標籤 ➢ **虛擬碼:** If en == 1 “out” is assigned with “in” eno = 1; Else eno = 0;
	**Assign2** ➢ **參數:** **Out1:** [Output] 標籤 **in1:** [Input] 值/標籤 **Out2:** [Output] 標籤 **in2:** [Input] 值/標籤 ➢ **虛擬碼:** If en == 1 “out1” is assigned with “in1” “out2” is assigned with “in2” eno = 1; Else eno = 0;

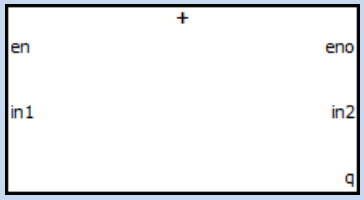
函式方塊	說明
	**Assign4** ➢ **參數:** **Out1:** [Output] 標籤 **in1:** [Input] 值/標籤 **Out2:** [Output] 標籤 **in2:** [Input] 值/標籤 **Out3:** [Output] 標籤 **in3:** [Input] 值/標籤 **Out4:** [Output] 標籤 **in4:** [Input] 值/標籤 ➢ **虛擬碼:** If en == 1 “out1” is assigned with “in1” “out2” is assigned with “in2” “out3” is assigned with “in3” “out4” is assigned with “in4” eno = 1; Else eno = 0;
	**Assign8** ➢ **參數:** **Out1:** [Output] 標籤 **in1:** [Input] 值/標籤 **Out2:** [Output] 標籤 **in2:** [Input] 值/標籤 **Out3:** [Output] 標籤 **in3:** [Input] 值/標籤 **Out4:** [Output] 標籤 **in4:** [Input] 值/標籤 **Out5:** [Output] 標籤 **in5:** [Input] 值/標籤 **Out6:** [Output] 標籤 **in6:** [Input] 值/標籤 **Out7:** [Output] 標籤 **in7:** [Input] 值/標籤 **Out8:** [Output] 標籤 **in8:** [Input] 值/標籤 ➢ **虛擬碼:** If en == 1 “out1” is assigned with “in1” “out2” is assigned with “in2” “out3” is assigned with “in3” “out4” is assigned with “in4” “out5” is assigned with “in5” “out6” is assigned with “in6” “out7” is assigned with “in7” “out8” is assigned with “in8” eno = 1; Else eno = 0;

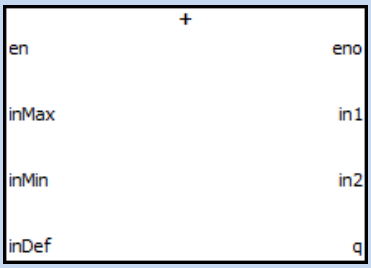
	**OnChange** (有變化時)
	➢ **參數:**
	in: [Input] 輸入標籤
	➢ **虛擬碼:**
	If (en == 1 and “in” is changed)
	eno = 1;
	Else eno = 0;

函式方塊	說明
	**InRange** (檢查輸入值是否在範圍內) ➢ **參數:** **inValue:** [Input] 輸入值/標籤 **inMax:** [Input] 輸入範圍的最大值/標籤 **inMin:** [Input] 輸入範圍的最小值/標籤 ➢ **虛擬碼:** If en == 1 and (inMin <= inValu<=inMax) eno = 1; Else eno = 0;
	**OutRange** (檢查輸入值是否超出圍) ➢ **參數:** **inValue:** [Input] 輸入值/標籤 **inMax:** [Input] 輸入範圍的最大值/標籤 **inMin:** [Input] 輸入範圍的最小值/標籤 ➢ **虛擬碼:** If en == 1 and (inValue < inMin or inValue >inMax) eno = 1; Else eno = 0;

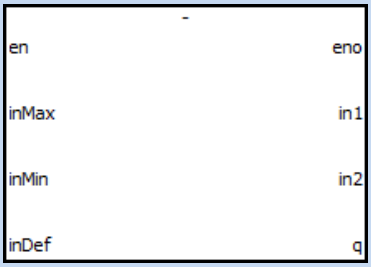
3.3.4.2 Math 群組

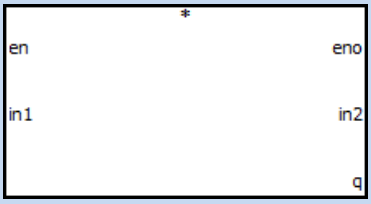
函式方塊	說明

	**Add** (運算符號: 加法)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 + in2;
	eno = 1;
	Else eno = 0;

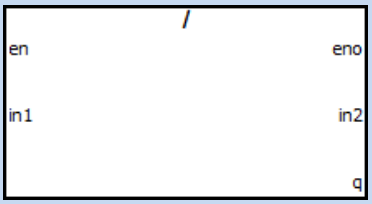
	**AddRange**
	➢ **參數:**
	inMax: [Input] 輸入值/標籤
	inMin: [Input] 輸入值/標籤
	inDef: [Input] 輸入值/標籤
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 + in2;
	if q < inMin or q >inMax
	q = inDef;
	eno = 1;
	Else eno = 0;

	**Sub** (運算符號: 減法)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 - in2;
	eno = 1;
	Else eno = 0;

	**SubRange**
	➢ **參數:**
	inMax: [Input] 輸入值/標籤
	inMin: [Input] 輸入值/標籤
	inDef: [Input] 輸入值/標籤
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 - in2;
	if q < inMin or q >inMax
	q = inDef;
	eno = 1;
	Else eno = 0;

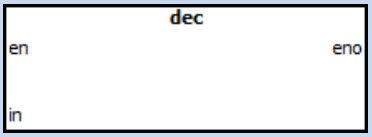
	**Mul** (運算符號: 乘法)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 * in2;
	eno = 1;
	Else eno = 0;

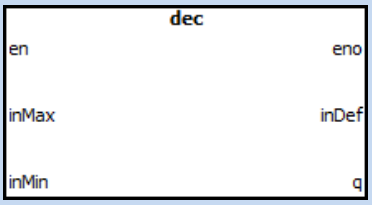
函式方塊	說明

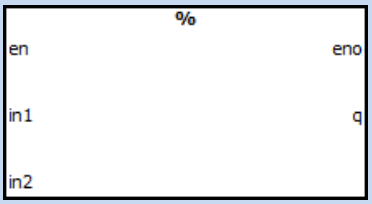
	**Div** (運算符號: 除法)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 / in2;
	eno = 1;
	Else eno = 0;

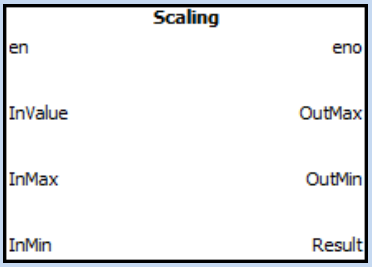
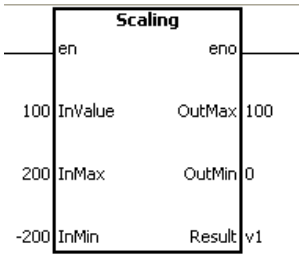
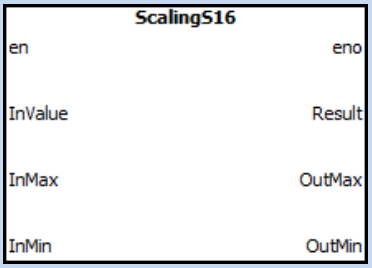
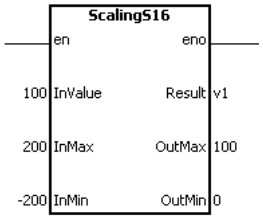
	**inc** (運算符號: 加一) ➢ **參數:** **in:**[Input/Output] 輸入標籤 ➢ **虛擬碼:** If en == 1 increment “in” by 1; eno = 1; Else eno = 0;

	**incRange**
	➢ **參數:**
	inMax: [Input] 輸入值/標籤
	inMin: [Input] 輸入值/標籤
	inDef: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = q + 1;
	if q < inMin or q >inMax
	q = inDef;
	eno = 1;
	Else eno = 0;

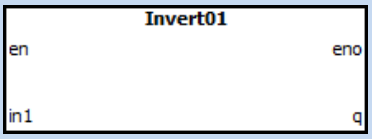
	**dec** (運算符號: 減一) ➢ **參數:** **in:** [Input/Output] 輸入標籤 ➢ **虛擬碼:** If en == 1 decrement “in” by 1; eno=1; Else eno = 0;

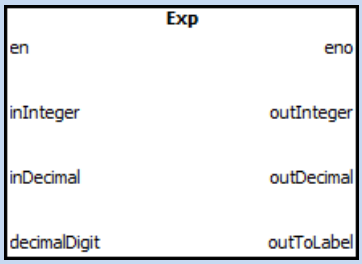
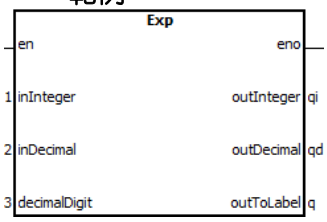
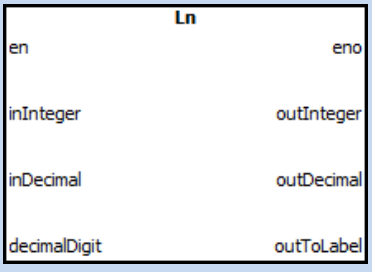
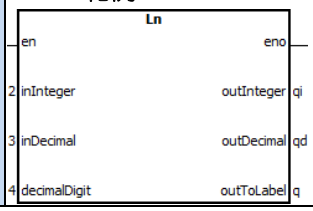
	**decRange**
	➢ **參數:**
	inMax: [Input] 輸入值/標籤
	inMin: [Input] 輸入值/標籤
	inDef: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = q - 1;
	if q < inMin or q >inMax
	q = inDef;
	eno = 1;
	Else eno = 0;

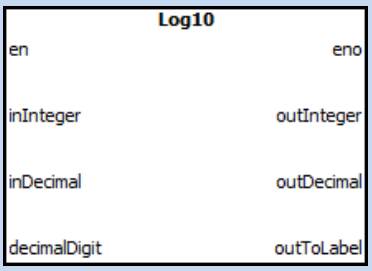
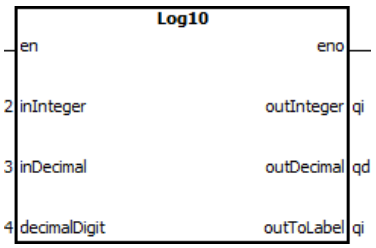
	**Mod** (運算符號: 同餘)
	➢ **參數:**
	in1: [Input] 輸入值/標籤
	in2: [Input] 輸入值/標籤
	q: [Output] 標籤
	➢ **虛擬碼:**
	If en == 1
	q = in1 % in2;
	eno = 1;
	Else eno = 0;

函式方塊	說明
	**Scaling** (類型: int) ➢ **參數:** **InValue:** [Input] 輸入值/標籤 **InMax:** [Input] 輸入範圍的最大值/標籤 **InMin:** [Input] 輸入範圍的最小值/標籤 **OutMax:** [Input] 輸出範圍的最大值/標籤 **OutMin:** [Input] 輸出範圍的最小值/標籤 **Result:** [Output] 縮放標籤 ➢ **範例:**  (InValue - InMin) / (InMax - InMin) = 0.75 Result(v1) = 0.75*(OutMax-OutMin)+OutMin = 75 ➢ **虛擬碼:** If en == 1 Do the conversion function; eno = 1; Else eno = 0;
	**ScalingS16** (類型: short 16) ➢ **參數:** **InValue:** [Input] 輸入值/標籤 **InMax:** [Input] 輸入範圍的最大值/標籤 **InMin:** [Input] 輸入範圍的最小值/標籤 **OutMax:** [Input] 輸出範圍的最大值/標籤 **OutMin:** [Input] 輸出範圍的最小值/標籤 **Result:** [Output] 縮放標籤 ➢ **範例:**  (InValue - InMin) / (InMax - InMin) = 0.75 Result(v1) = 0.75*(OutMax-OutMin)+OutMin = 75 ➢ **虛擬碼:** If en == 1 Do the conversion function; eno = 1; Else eno = 0;

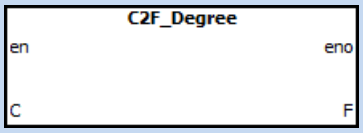
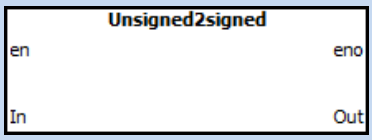
函式方塊	說明

	**Invert01** (邏輯運算: 相反)
	➢ **參數:**
	in1: [Input] 輸入值/標籤，0 和 1 的邏輯相反。
	q: [Output] 標籤
	➢ **範例:**
	If"in1"is True => q = 0;
	If"in1"is False => q = 1;
	➢ **虛擬碼:**
	If en == 1,
	q = invert(in1);
	eno = 1;
	Else eno = 0;

	**Exp** (e^x) ➢ **參數:** **inInteger:** [Input] x 的整數位 **inDecimal:** [Input] x 的小數位 **decimalDigit:** [Input] 小數部分位數(0~5) **outInteger:** [Output] 輸出整數位 **outDecimal:** [Output] 輸出小數位 **outToLabel:** [Output] 輸出顯示於 Label 的值(搭配 DecimalDigits 屬性使用) ➢ **範例:**  e^1.002 = 2.723 qi = 2 qd = 723 q =2723 ➢ **虛擬碼:** If en == 1, Do the exponential function; eno = 1; Else eno = 0;
	**Ln** (ln(x)) ➢ **參數:** **inInteger:** [Input] x 的整數位 **inDecimal:** [Input] x 的小數位 **decimalDigit:** [Input] 小數部分位數(0~5) **outInteger:** [Output] 輸出整數位 **outDecimal:** [Output] 輸出小數位 **outToLabel:** [Output] 輸出顯示於 Label 的值(搭配 DecimalDigits 屬性使用) ➢ **範例:**  ln(2.0003)=0.6932 qi = 0 qd = 6932 q =6932 ➢ **虛擬碼:** If en == 1, Do the natural logarithm function; eno = 1; Else eno = 0;

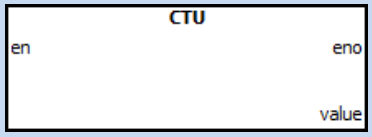
函式方塊	說明
	**Log** (log(x)) ➢ **參數:** **inInteger:** [Input] x 的整數位 **inDecimal:** [Input] x 的小數位 **decimalDigit:** [Input] 小數部分位數(0~5) **outInteger:** [Output] 輸出整數位 **outDecimal:** [Output] 輸出小數位 **outToLabel:** [Output] 輸出顯示於 Label 的值(搭配 DecimalDigits 屬性使用) ➢ **範例:**  log(2.0003) = 0.3010 qi = 0 qd = 3010 q =3010 ➢ **虛擬碼:** If en == 1, Do the log(x) function; eno = 1; Else eno = 0;

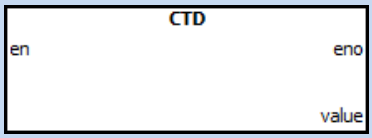
3.3.4.3 Convert 群組

函式方塊	說明
	**C2F_Degree** (攝氏溫度轉為華氏溫度) ➢ **參數:** **C:** [Input] 輸入值/標籤 (攝氏度) **F:** [Output] 輸出標籤 (華氏度) ➢ **虛擬碼:** If en == 1, F = (9/5)*C + 32; eno = 1; Else eno = 0;
	**Unsigned2signed** (將值從無符號轉為有符號) ➢ **參數:** **In:** [Input] 輸入值/標籤 **Out:** [Output] 輸出標籤 ➢ **虛擬碼:** If en == 1, Do the conversion function; eno = 1; Else eno = 0;

3.3.4.4 Counter 群組

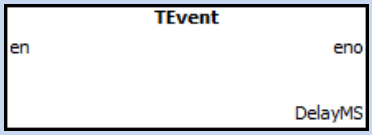
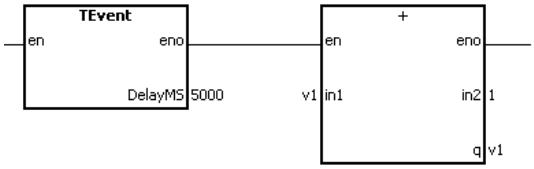
函式方塊	說明

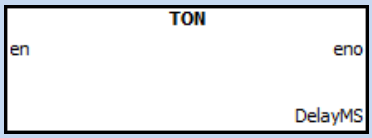
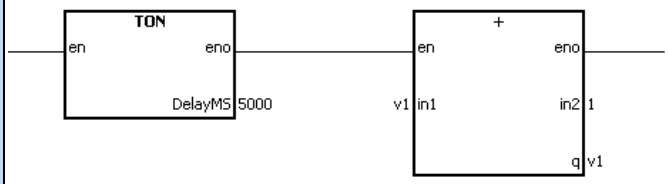
	**CTU** (Count Up，向上計數)
	➢ **參數:**
	value: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If en == 1,
	Count=0;
	Loop: Count up until count>=value;
	During counting, eno = 0;
	When End, eno = 1;
	Else
	Reset count to 0,
	eno = 0;
	註: 計數間隔與階 (Rung) 的數目有關。

	**CTD** (Count Down，向下計數)
	➢ **參數:**
	value: [Input] 輸入值/標籤
	➢ **虛擬碼:**
	If en == 1,
	Count=value;
	Loop: Count down until count<=0;
	During counting, eno = 0,
	When End, eno = 1;
	Else
	Reset count to value,
	eno = 0;
	註: 計數間隔與階 (Rung) 的數目有關。

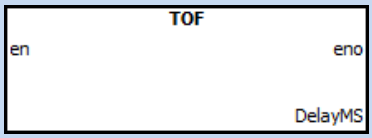
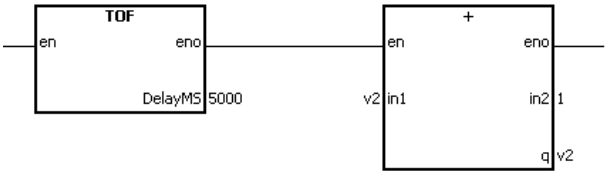
3.3.4.5 Timer 群組

函式方塊	說明

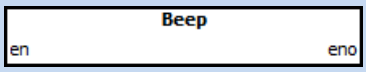
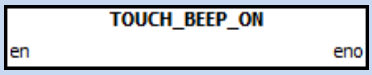
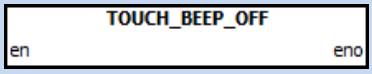
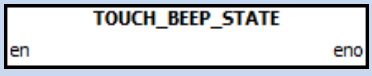
	**TimerEventH**
	(週期定時器事件，在每個定時器(Timer)之後觸發一次，單位:ms)
	➢ **參數:**
	DelayMS: [Input] 設定間隔時間。最小設定為 10 ms。
	➢ **範例:**
	每 5 秒循環的計算 v1 = v1 + 1
	 
	➢ **虛擬碼:**
	If en == 1
	If timer < DelayMS
	Start the timer;
	eno = 0;
	Else
	Timer = 0;
	eno = 1;
	Else
	Reset the timer;
	eno = 0;

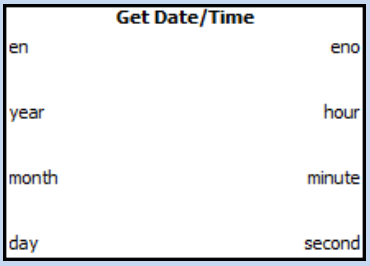
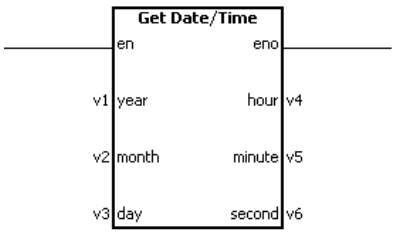
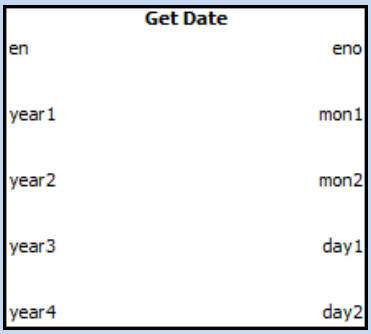
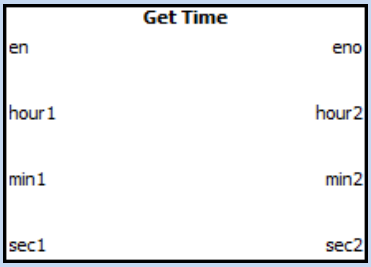
	**TON** (Timer On，一段時間後開啟，單位:ms)
	➢ **參數:**
	DelayMS: [Input] 設定間隔時間。最小設定為 10 ms。
	➢ **範例:**
	5 秒後，計算 v1 = v1 + 1
	 
	➢ **虛擬碼:**
	If en == 1
	Start the timer if not;
	Stop the timer when timer>=DelayMS;
	If timer runs
	eno = 0;
	else
	eno = 1;
	Else
	Reset the timer;
	eno = 0;

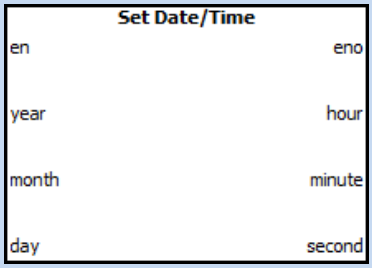
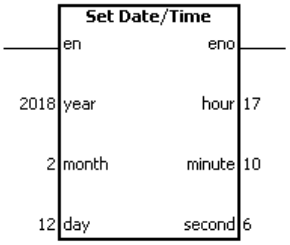
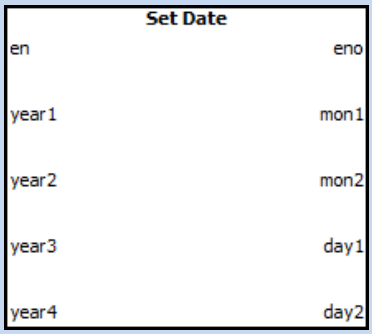
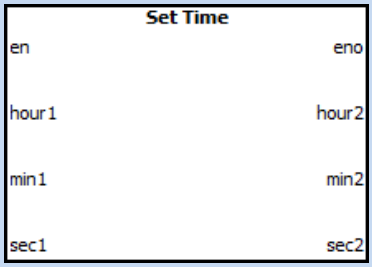
函式方塊	說明

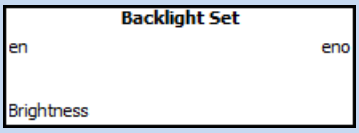
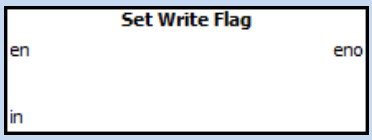
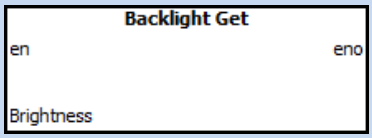
	**TOF** (Timer Off，一段時間後關閉，單位:ms)
	➢ **參數:**
	DelayMS: [Input] 設定間隔時間。最小設定為 10 ms。
	➢ **範例:**
	5 秒內， 計算 v2 = v2 + 1
	 
	➢ **虛擬碼:**
	If en == 1,
	Start the timer if not;
	Stop the timer when timer>=DelayMS;
	If timer runs
	Eno=1;
	Else
	Eno=0;
	Else
	Reset the timer;
	eno = 0;

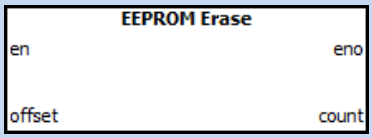
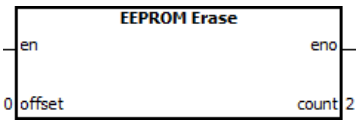
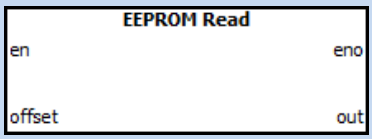
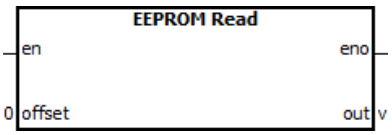
3.3.4.6 System 群組

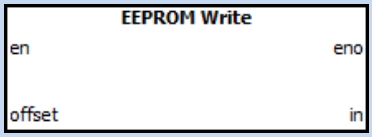
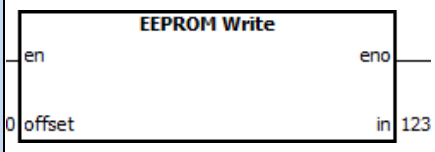
函式方塊	說明
	**Beep** (發出嗶聲) ➢ **虛擬碼:** If en == 1, beep and eno = 1; Else eno = 0;
	**TOUCH_BEEP_ON** (當點擊螢幕時發出嗶聲) ➢ **虛擬碼:** If en == 1, Set the beep function to ON; eno = 1; Else eno = 0;
	**TOUCH_BEEP_OFF** (關閉嗶聲功能) ➢ **虛擬碼:** If en == 1, Set the beep function to OFF; eno = 1; Else eno = 0;
	**TOUCH_BEEP_STATE** (檢查嗶聲功能狀態) ➢ **虛擬碼:** If en == 1, Beep function is ON, eno =1; Beep function if OFF, eno =0; Else eno = 0;

函式方塊	說明
	**Get_Date_Time** (從 TouchPAD 裝置上的 RTC chip 取得日期和時間) ➢ **參數:** **year:** [Output] 年 **hour:** [Output] 時 **month:** [Output] 月 **minute:** [Output] 分 **day:** [Output] 日 **second:** [Output] 秒 ➢ **範例:**  v1= 年 v2= 月 v3= 日 v4= 時 v5= 分 v6= 秒 ➢ **虛擬碼:** If en == 1, Get the RTC's date and time; eno =1; Else eno = 0;
	**Get_Date_Digit** (從 TouchPAD 裝置上的 RTC chip 取得日期) ➢ **參數:** **year1:** [Output] 年(千位) **mon1:** [Output] 月(十位) **year2:** [Output] 年(百位) **mon2:** [Output] 月(個位) **year3:** [Output] 年(十位) **day1:** [Output] 日(十位) **year4:** [Output] 年(個位) **day2:** [Output] 日(個位) ➢ **虛擬碼:** If en == 1, Get the RTC's date; eno =1; Else eno = 0;
	**Get_Time_Digit** (從 TouchPAD 裝置上的 RTC chip 取得時間) ➢ **參數:** **hour1:** [Output] 時(十位) **hour2:** [Output] 時(個位) **min1:** [Output] 分(十位) **min2:** [Output] 分(個位) **sec1:** [Output] 秒(十位) **sec2:** [Output] 秒(個位) ➢ **虛擬碼:** If en == 1, Get the RTC's time; eno =1; Else eno = 0;

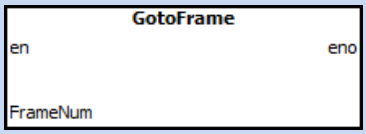
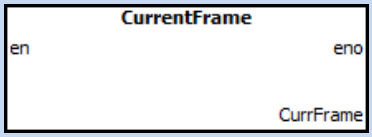
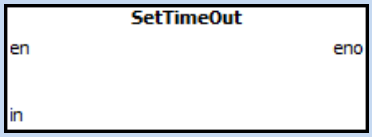
函式方塊	說明
	**Set_Date_Time** (在 TouchPAD 裝置上設定 RTC chip 的日期和時間) ➢ **參數:** **year:** [Input] 年 **hour:** [Input] 時 **month:** [Input] 月 **minute:** [Input] 分 **day:** [Input] 日 **second:** [Input] 秒 ➢ **範例:**  設定日期和時間: 2018-2-12，17:10:06 ➢ **虛擬碼:** If en == 1, Set the date and time to RTC; eno =1; Else eno = 0;
	**Set_Date_Digit** (在 TouchPAD 裝置上設定 RTC chip 的日期) ➢ **參數:** **year1:** [Input] 年(千位) **mon1:** [Input] 月(十位) **year2:** [Input] 年(百位) **mon2:** [Input] 月(個位) **year3:** [Input] 年(十位) **day1:** [Input] 日(十位) **year4:** [Input] 年(個位) **day2:** [Input] 日(個位) ➢ **虛擬碼** If en == 1, Set the date to RTC; eno =1; Else eno = 0;
	**Set_Time_Digit** (在 TouchPAD 裝置上設定 RTC chip 的時間) ➢ **參數:** **hour1:** [Input] 時(十位) **hour2:** [Input] 時(個位) **min1:** [Input] 分(十位) **min2:** [Input] 分(個位) **sec1:** [Input] 秒(十位) **sec2:** [Input] 秒(個位) ➢ **虛擬碼** If en == 1, Set the time to RTC; eno =1; Else eno = 0;

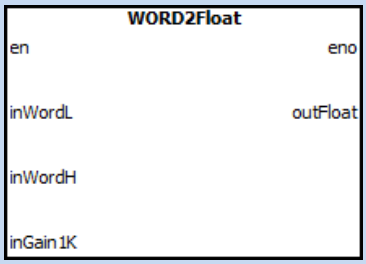
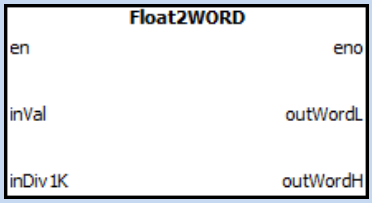
函式方塊	說明
	**BacklightSet** (設定 TouchPAD 裝置的亮度) ➢ **參數:** **Brightness:** [Input] 指定 TouchPAD 亮度。 範圍: 0 ~ 255，0: 最暗，255: 最亮。 ➢ **虛擬碼:** If en == 1 Set the brightness value as “Brightness”; eno =1; Else eno = 0;
	**Set_Write_Flag** (設定 I/O 標籤的寫入標籤，因此 I/O 標籤應在下一次 I/O 掃描時更新到遠端設備) ➢ **參數:** **in:** [Input/Output] I/O 標籤 ➢ **虛擬碼:** If en == 1 Set the “write” flag of a I/O tag; eno =1; Else eno = 0;
	**BacklightGet** (取得 TouchPAD 裝置的亮度) ➢ **參數:** **Brightness:** [Output] 取得 TouchPAD 亮度。 ➢ **虛擬碼:** If en == 1 “Brightness”=brightness value; eno =1; Else eno = 0;

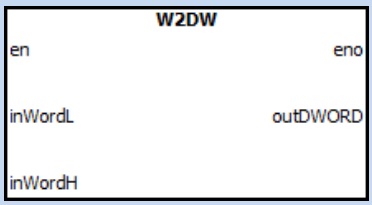
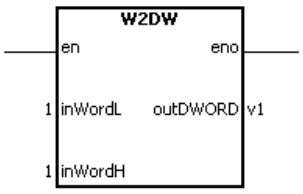
函式方塊	說明
	**EepromErase** (清除 EEPROM 內資料) ➢ **參數:** **offset:** [Input] 指定位移的位址，來清除資料。範圍: 0 ~ 511。 **註:** offset + count 不能大於 512。 **count:** [Input] 指定要清除的 32-bit 空間數量。範圍: 1 ~ 512。 **註:** offset + count 不能大於 512。 ➢ **範例:**  從區域 ”0” 開始清除 2 個 32-bit 空間的值 ➢ **虛擬碼:** If en == 1 Erase data of the EEPROM; eno =1; Else eno = 0; **註:** EEPROM 寫入限制為 100,000 次。如頻繁寫入會造成 EEPROM 損壞。
	**EepromRead** (從 EEPROM 取得資料) ➢ **參數:** **offset:** [Input] 指定位移的位址，來讀取資料。 範圍: 0 ~ 511。 **out:** [Output] 指定存儲來自參數區域的值。 ➢ **範例:**  v = 讀取區域 “0” 的值(32-bit)。 ➢ **虛擬碼:** If en == 1 Read data from the EEPROM; eno =1; Else eno = 0;

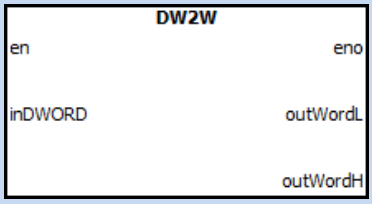
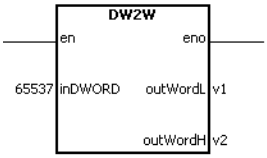
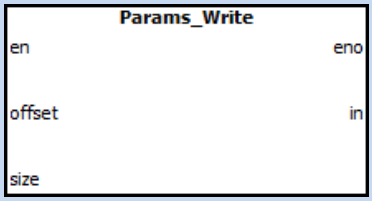
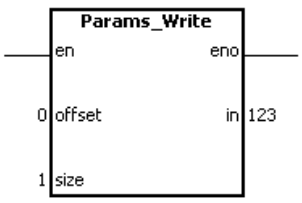
函式方塊	說明
	**EepromWrite** (設定資料到 EEPROM) ➢ **參數:** **offset:** [Input] 指定位移的位址，來寫入資料。範圍: 0 ~ 511。 **in:** [Input] 指定寫入的值。 ➢ **範例:**  將 "123" 值寫入區域 “0”。 ➢ **虛擬碼:** If en == 1, Write data to the EEPROM; eno = 1; Else eno = 0; **註:** Flash 寫入限制為 100,000 次。如頻繁寫入會造成 Flash 損壞。

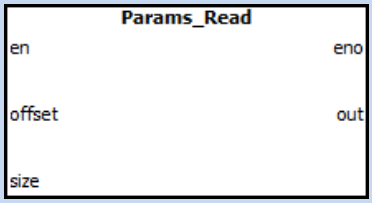
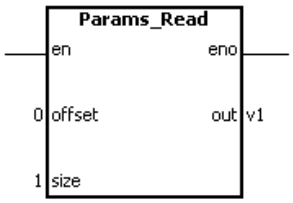
3.3.4.7 User_define 群組

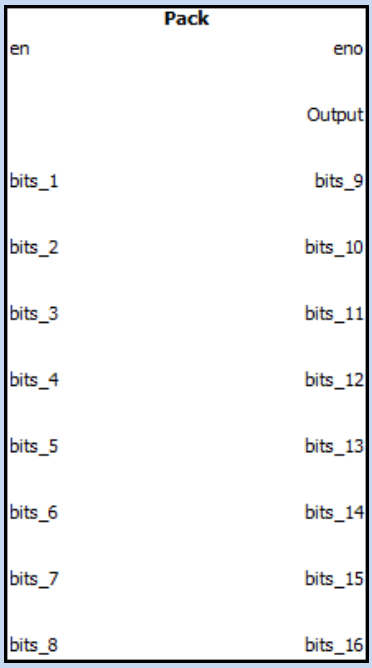
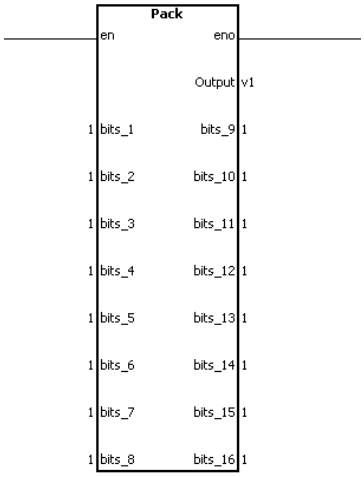
函式方塊	說明
	**GotoFrame** (到設計頁面(Frame)號碼) ➢ **參數:** **FrameNum:** [Input] 設定 Frame 號碼。Frame 號碼是從 1 開始索引，不是 Fame 屬性檢視區中的 ID 碼。 ➢ **虛擬碼:** If en == 1 Go to the frame number; eno =1; Else eno = 0;
	**CurrentFrame** (取得目前設計頁面(Frame)號碼) ➢ **參數:** **CurrFrame:** [Output] 目前 Frame 號碼 ➢ **虛擬碼:** If en == 1 CurrFrame= current frame number; eno =1; Else eno = 0;
	**SetTimeOut** (設定 uart fucntions 定時器(Timer)的超時) ➢ **參數:** **in:** [input] 超時值/標籤 ➢ **虛擬碼:** If en == 1 TimeOut value = “in”; eno =1; Else eno = 0;

函式方塊	說明
	**WORD2Float** (將 2 WORD 轉換為單一 float) ➢ **參數:** **inWordL:** [Input] Low word 值 **inWordH:** [Input] High word 值 **inGain1K:** [Input] 結果是否要*1000 **outFloat:** [Output] float 標籤 ➢ **虛擬碼:** If en == 1, Do the conversion function; eno = 1; Else eno = 0;
	**Float2WORD** (將單一 float 轉換為 2 WORD) ➢ **參數:** **inVal:** [Input] float 值 **inDiv1K:** [Input] 結果是否要/1000 **outWordL:** [Output] Low word 標籤 **outWordH:** [Output] High word 標籤 ➢ **虛擬碼:** If en == 1, Do the conversion function; eno = 1; Else eno = 0;

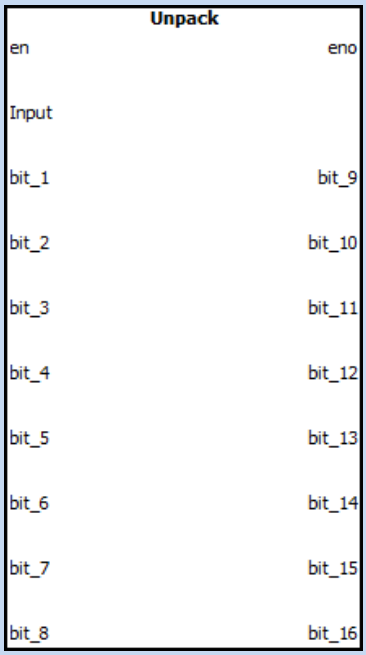
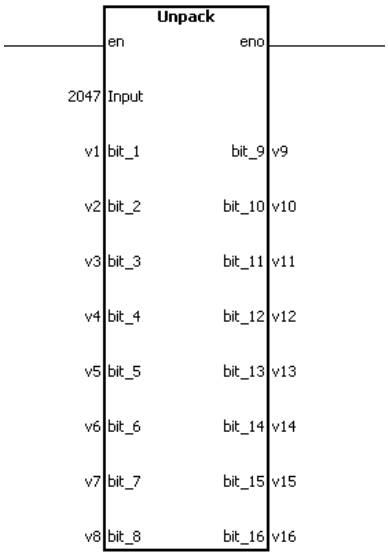
	**WORD2DWORD** (將 2 WORD 轉換為單一 DWORD)
	➢ **參數:**
	inWordL: [Input] Low word 值
	inWordH: [Input] High word 值
	outDWORD: [Output] DWORD 標籤
	➢ **範例:**
	 
	unsigned char v1[4];
	v1[1]=(inWordL>>8) & 0xFF;
	v1[0]=inWordL & 0xFF;
	v1[3]=(inWordH>>8) & 0xFF;
	v1[2]=inWordH & 0xFF;
	取得 v1 = 65537;
	➢ **虛擬碼:**
	If en == 1,
	Do the conversion function;
	eno = 1;
	Else eno = 0;

函式方塊	說明
	**DWORD2WORD** (將單一 DWORD 轉換為 2 WORD) ➢ **參數:** **inDWORD:** [Input] DWORD 值/標籤 **outWordL:** [Output] Low-word 標籤 **outWordH:** [Output] High-word 標籤 ➢ **範例:**  V1 = inDWORD & 0xFFFF = 1 V2=(inDWORD>>16) & 0xFFFF =1 ➢ **虛擬碼:** If en == 1, Do the conversion function; eno = 1; Else eno = 0;
	**Params_Write** (設定資料到 MCU (MicroController Unit) 內部 Flash 的 256-byte 參數區域) ➢ **參數:** **offset:** [Input] 指定參數區域位移的位址，來寫入資料。範圍: 0 ~ 255。**註:** iOffset + iSize 不能大於 256。 **size:** [Input] 指定要寫入參數區域 byte 數的大小。範圍: 1 ~ 256。**註:** iOffset + iSize 不能大於 256。 **in:** [Input] 指定寫入的值。 ➢ **範例:**  將 "123" 值寫入區域 "0"，空間大小為"1"。 ➢ **虛擬碼:** If en == 1, Write data to internal flash; eno = 1; Else eno = 0; **註:** Flash 寫入限制為 100,000 次。如頻繁寫入會造成 Flash 損壞。

函式方塊	說明
	**Params_Read** (從 MCU (MicroController Unit) 內部 Flash 的 256-byte 參數區域取得資料) ➢ **參數:** **offset:** [Input] 指定參數區域位移的位址，來讀取資料。 範圍: 0 ~ 255。**註:** iOffset + iSize 不能大於 256。 **size:** [Input] 指定要從參數區域讀取的資料大小。 範圍: 1 ~ 256。**註:** iOffset + iSize 不能大於 256。 **out:** [Output] 指定存儲來自參數區域的值。 ➢ **範例:**  v1 = 從區域 “0” 讀取空間大小 “1” 的值。 **虛擬碼:** If en == 1, Read data from internal flash; eno = 1; Else eno = 0;

	**Pack** (二進制轉十進制)
	將 16 個標籤值轉換為一個擁有 16-bit 資料的單一標籤。
	➢ **參數:**
	Bits_1: [Input] 輸入值/標籤
	Bits_2: [Input] 輸入值/標籤
	……
	Bits_15: [Input] 輸入值/標籤
	Bits_16: [Input] 輸入值/標籤
	Output: [Output] 標籤
	➢ **範例:**
	for(i=0 ; i<16 ; i++)
	v1= v1+ (bits_[i]<<i);
	➔ v1=65535
	➢ **虛擬碼:**
	If en == 1,
	for(i=0 ; i<16 ; i++)
	Output= Output+ (bits_[i]<<i);
	eno = 1;
	Else eno = 0;
	 

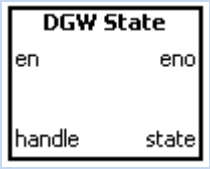
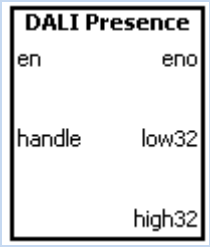
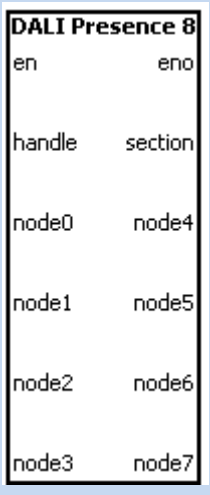
函式方塊	說明

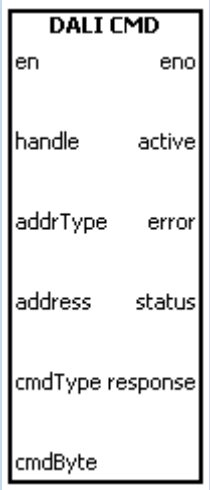
	**UnPack** (十進制轉二進制)
	將 16-bit 資料的單一標籤轉換為 16 個標籤值。
	➢ **參數:**
	Input: [Input] 輸入值/標籤
	Bits_1: [Output] 輸出標籤
	Bits_2: [Output] 輸出標籤
	……
	Bits_15: [Output] 輸出標籤
	Bits_16: [Output] 輸出標籤
	➢ **範例:**
	 
	v1=v2=v3=v4=v5=v6=v7=v8=v9=v10=v11=1;
	v12=v13=v14=v15=v16=0
	➢ **虛擬碼:**
	If en == 1,
	bit_1 = (input>>0)&1;
	bit_2 = (input>>1)&1;
	bit_3 = (input>>2)&1;
	…..
	bit_16 = (input>>15)&1;
	eno = 1;
	Else eno = 0;

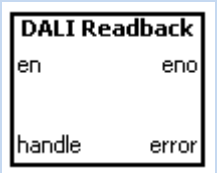
3.3.4.8 VPD-130 群組

函式方塊	說明
	**GetPanelKey** (只支援 VPD 系列) ➢ **參數:** **out:** [Output] 用於存儲按下塑膠鍵盤的值。 ➢ **虛擬碼:** If en == 1, “out” = the panel key number; eno = 1; Else eno = 0;
	**ShowPanelLed** (設定 LED 狀態) ➢ **虛擬碼:** LED state = en If en == 1, LED state = ON; If en == 0, LED state = OFF;

3.3.4.9 DGW-521 群組

函式方塊	說明
	**DGW State**(取得 DGW-521 模組的狀態) ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **state:** [Output] 用於存儲 DGW-521 的狀態。 >= 0, DGW-521 的各種狀態。 < 0, error code。
	**DALI Scan** (掃描連接在 DGW-521 上的 DALI 燈具) ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **active:** [Input] 設定為 1 後開始執行，完成後自動清除為 0。 **error:** [Output] 錯誤代碼。 eno(1) = 完成並且 $active 被自動清除。 eno(0) = 空閒、失敗或正在處理。
	**DALI Presence**(將所連接的 DALI 燈具數量儲存在兩個 32bit 標籤內) ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **Low32:** [Output] DALI 燈具的連接狀態。 **High32:** [Output] DALI 燈具的連接狀態。
	**DALI Presence 8**(將連接的 DALI 燈具數量儲存在各個區間內) ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **section:** [Input] 設定區間 Section 0 = address 0 - 7, Section 1 = address 8 - 15. Section 2 = address 16 - 23, Section 3 = address 24 - 31. Section 4 = address 32 - 39, Section 5 = address 40 - 47. Section 6 = address 48 - 55, Section 7 = address 56 - 63. node0 ~ node7: [Output] DALI 燈具的連接狀態 1 = 存在。 0 = 不存在。

函式方塊	說明
	**DALI Scene**(發送 DALI goto-scene 的指令) 設置 $active = 1 後應持續調用此 Function Block，直到 $active 回到 0（完成/空閒）。 ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **addrType:** [Input] 設定地址類型，0=Lamp, 1=Group, 2=Broadcast。 **addrss:** [Input]DALI 地址，0~63 為 Lamp 地址, 0~15 為 Group 地址 **scene:** [Input] 0-15。 **active:** [Input] 設定為 1 後開始執行，完成後自動清除為 0。 **error:** [Output] 錯誤代碼。 **status:** [Output] 指令處理狀態。 **response:** [Output] DALI 燈具回傳值。 有關$scene, $status, $response 的詳細資訊請參考 DGW-521 的使用手冊。
	**DALI CMD**(發送 DALI 指令) 設置 $active = 1 後應持續調用此 Function Block，直到 $active 回到 0（完成/空閒）。 ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **addrType:** [Input] 設定地址類型，0=Lamp, 1=Group, 2=Broadcast。 **addrss:** [Input]DALI 地址，0~63 為 Lamp 地址, 0~15 為 Group 地址 **cmdType:** [Input] 指令類型， 0=燈具功率, 1=DALI 命令。 **cmdByte:** [Input] 取決於 cmdType 決定是燈具功率的數值或者是 DALI 命令。 **active:** [Input] 設定為 1 後開始執行，完成後自動清除為 0。 **error:** [Output] 錯誤代碼。。 **status:** [Output] 指令處理狀態。 **response:** [Output] DALI 燈具回傳值。 有關$cmdByte, $status, $response 的詳細資訊請參考 DGW-521 的使用手冊。

函式方塊	說明
	**DALI Raw**(發送原始 DALI 指令) 設置 $active = 1 後應持續調用此 Function Block，直到 $active 回到 0（完成/空閒）。 ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **active:** [Input] 設定為 1 後開始執行，完成後自動清除為 0。 **addrByte:** [Input] DALI 命令的地址字節（包括地址、類型、廣播設置）。 **cmdByte:** [Input] 取決於 addrByte 決定是燈具功率的數值或者是 DALI 命令。 **error:** [Output] 錯誤代碼。 **status:** [Output] 指令處理狀態 **response:** [Output] DALI 燈具回傳值 有關$addrByte, $cmdByte, $status, $response 的詳細資訊請參考 DGW-521 的使用手冊。
	**DALI Readback**(發送 DALI 命令以獲取當前燈具的數值並且存儲在 DGW-521 中) ➢ **參數:** **handle:** [Intput] DGW-521 的 handle 數值。 **error:** [Output] 錯誤代碼。

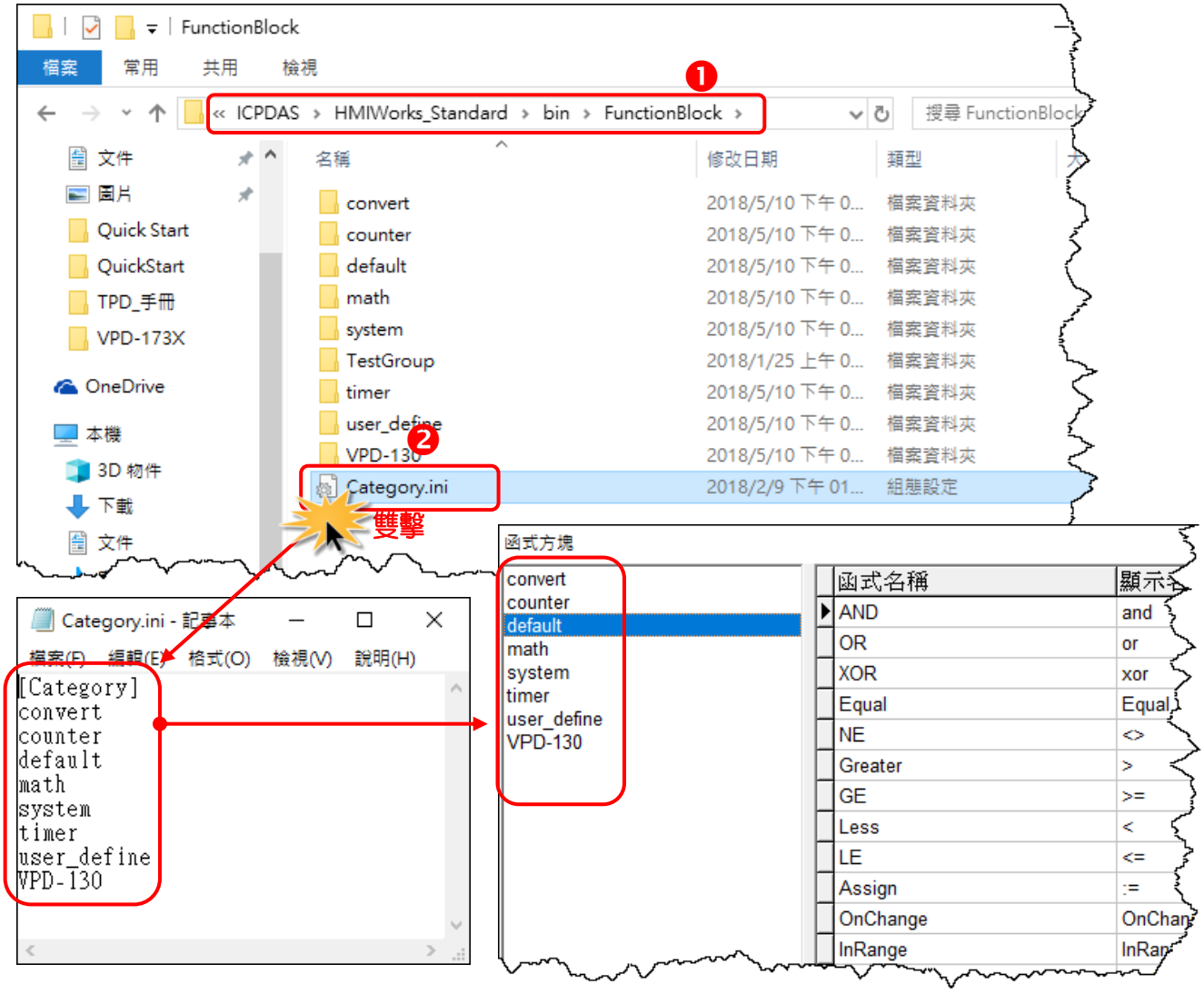
3.3.5 使用者定義的功能方塊

為何要使用功能方塊 (Function Block)?
若是只使用階梯圖的話，遇到複雜的情形時，可能會很困難。這時階梯圖就是很好的選擇了。

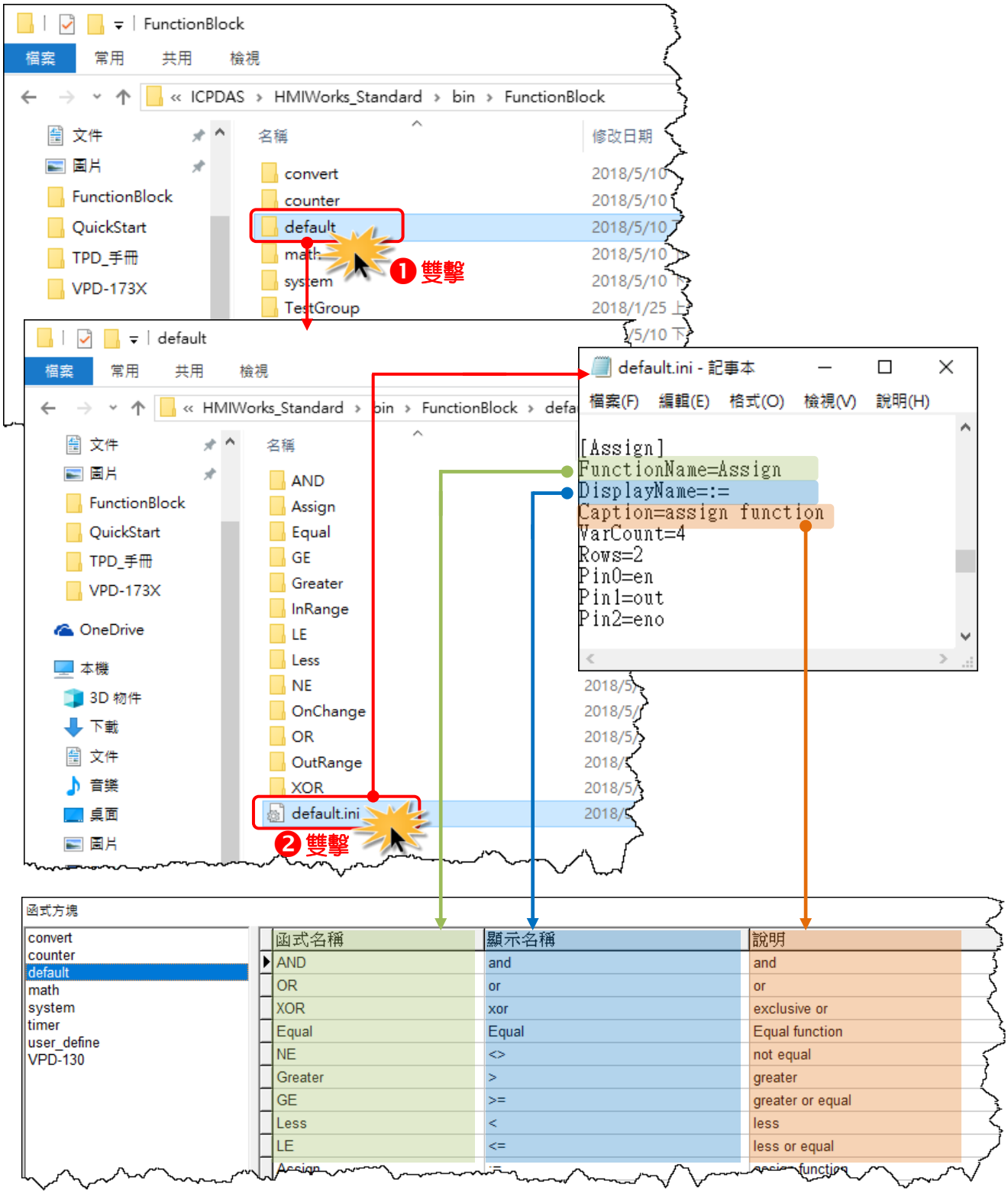
想要知道如何自行定義一個功能方塊，首先要先了解 HMIWorks 是如何使用功能方塊的。以「賦值 (Assign)」功能方塊為例，說明如下。

3.3.5.1 HMIWorks 如何使用功能方塊

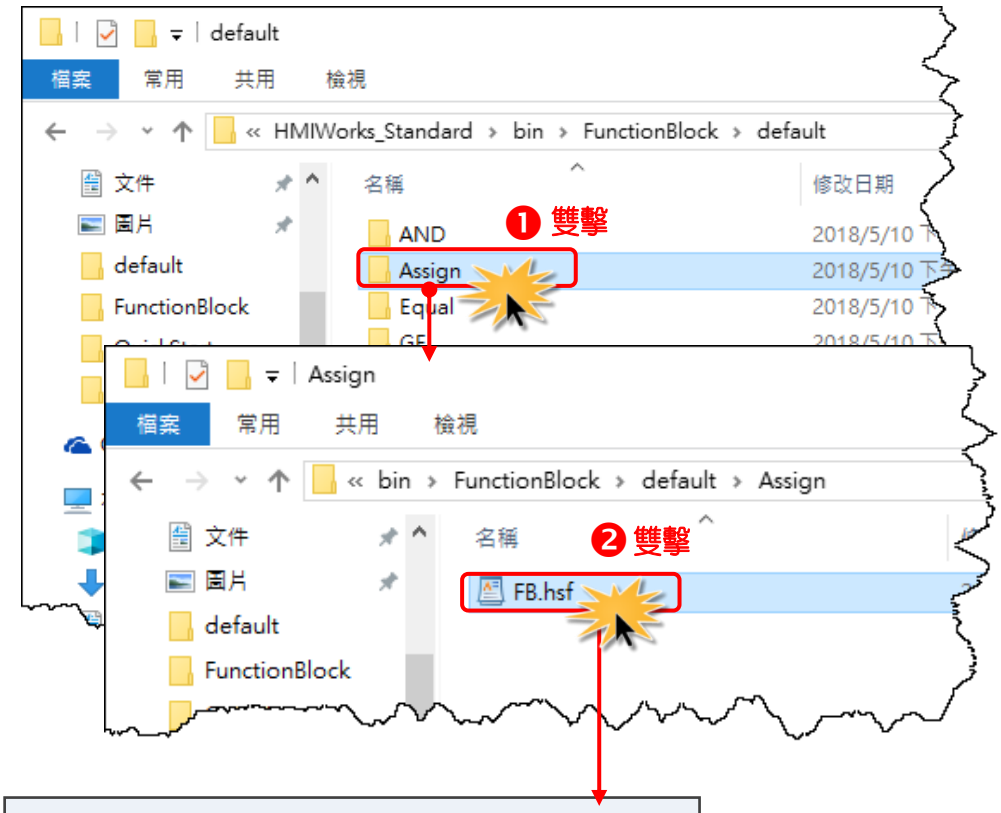
1. 到 HMIWorks 的安裝路徑下，找到「**bin\FunctionBlock**」的子目錄，在該子目錄下開啟「**Category.ini**」這個檔案，並依檔案內容載入功能方塊的群組，如圖所示。



2. 舉例而言，若選擇群組「default」，HMIWorks 會在有著相同名字的子目錄，開啟有著同樣名字的 “.ini” 檔，然後從檔案中載入資料。即在「default」這個目錄下開啟 “default.ini” 這個檔案來載入資料。



3. 雙擊點選「賦值 (Assign)」，將功能方塊的函式設為賦值。階梯圖設計家會執行在「Assign」子目錄下的 “FB.hsf” 這個檔案所定義的邏輯。 “FB.hsf” 是 C 語言格式。
下圖解釋了「Assign」的 “FB.hsf” 做了什麼。

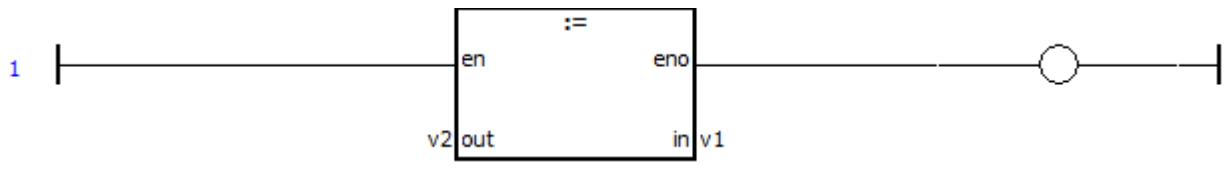



```
if (!en) return 0;

if (VAR_VALUE($out) == VAR_VALUE($in)) return 1;

#ifdef LAD_MODS_VER_1
VAR_SET($out, VAR_VALUE($in));
#else
VAR_VALUE($out) = VAR_VALUE($in);
#endif

VAR_SET_DIRTY($out);
VAR_FLAG_SET_WRITE($out);
return 1;
```

若 **en** 設為 0，則 **eno** 也設為 0 並返回。

若 **v1** 等於 **v2**，**eno** 設為 1 並返回。
v2 給定 **v1 的值。**

將 **v2** 的值顯示於螢幕上與其關連的控制項 (widget)。
eno 設為 1 並返回。



3.3.5.2 如何新增使用者定義之功能方塊

下面將詳細說明如何的新增一個自己定義的功能方塊。

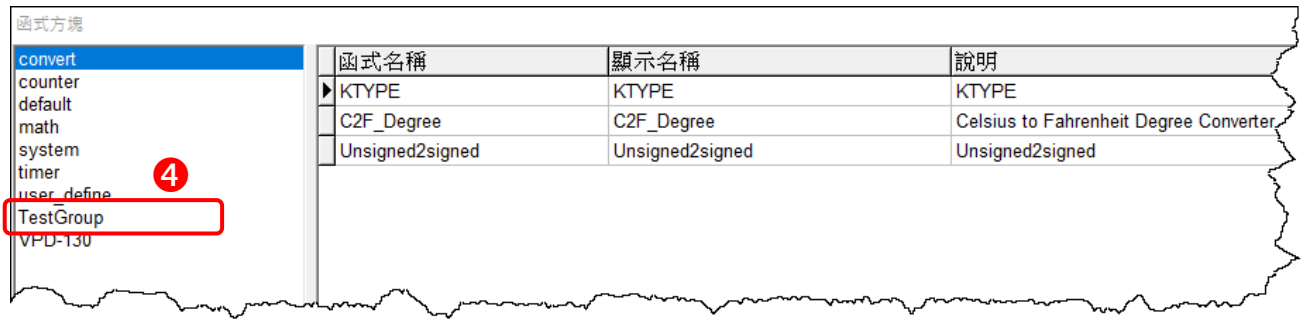
步驟 1: 到 HMIWorks 的安裝路徑下，在子目錄「**bin\FunctionBlock**」中，新增一個資料夾，例如取名為「**TestGroup**」，這是新群組的名稱。

步驟 2: 雙擊開啟 **“Category.ini”**，新增代表新群組的一個項目 (例如: TestGroup)。

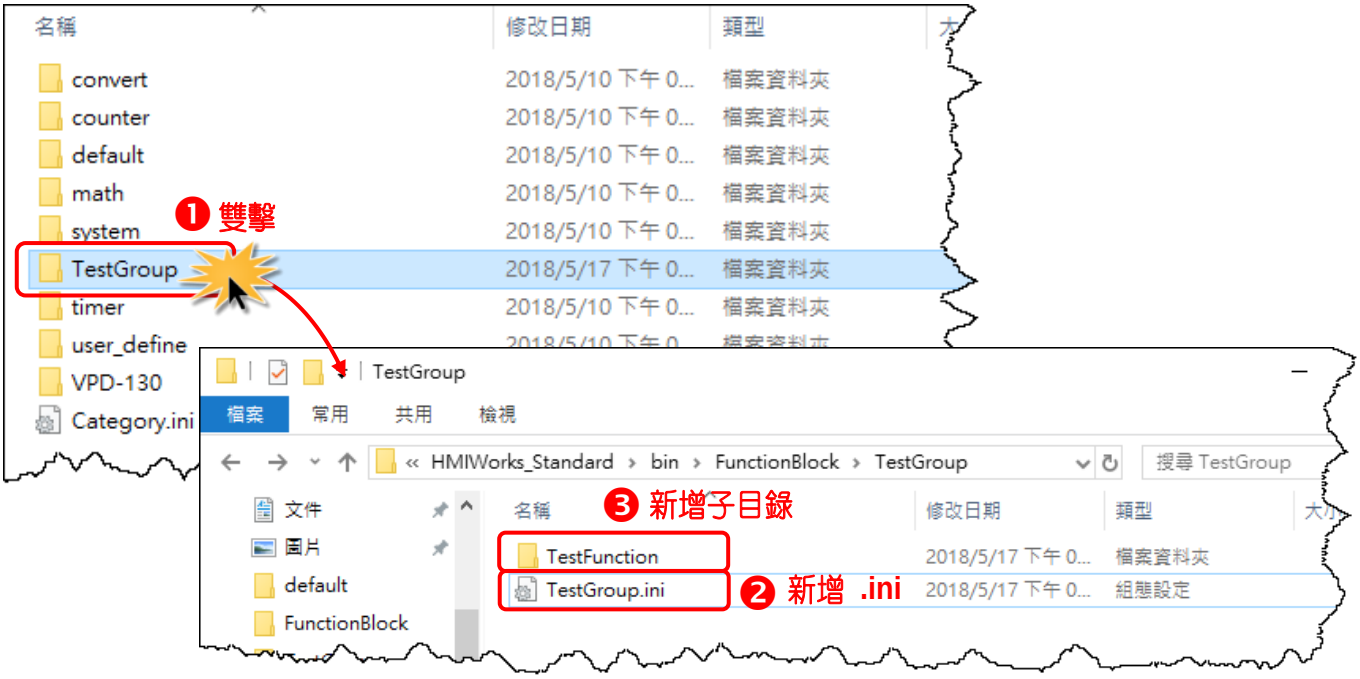
注意

在 **Category.ini** 新增項目的名稱 **“一定”** 要和新增資料夾的名稱一模一樣。

您可以開啟 **“函式方塊”** 視窗來檢查剛新增完成的新群組名稱 (如: TestGroup)。

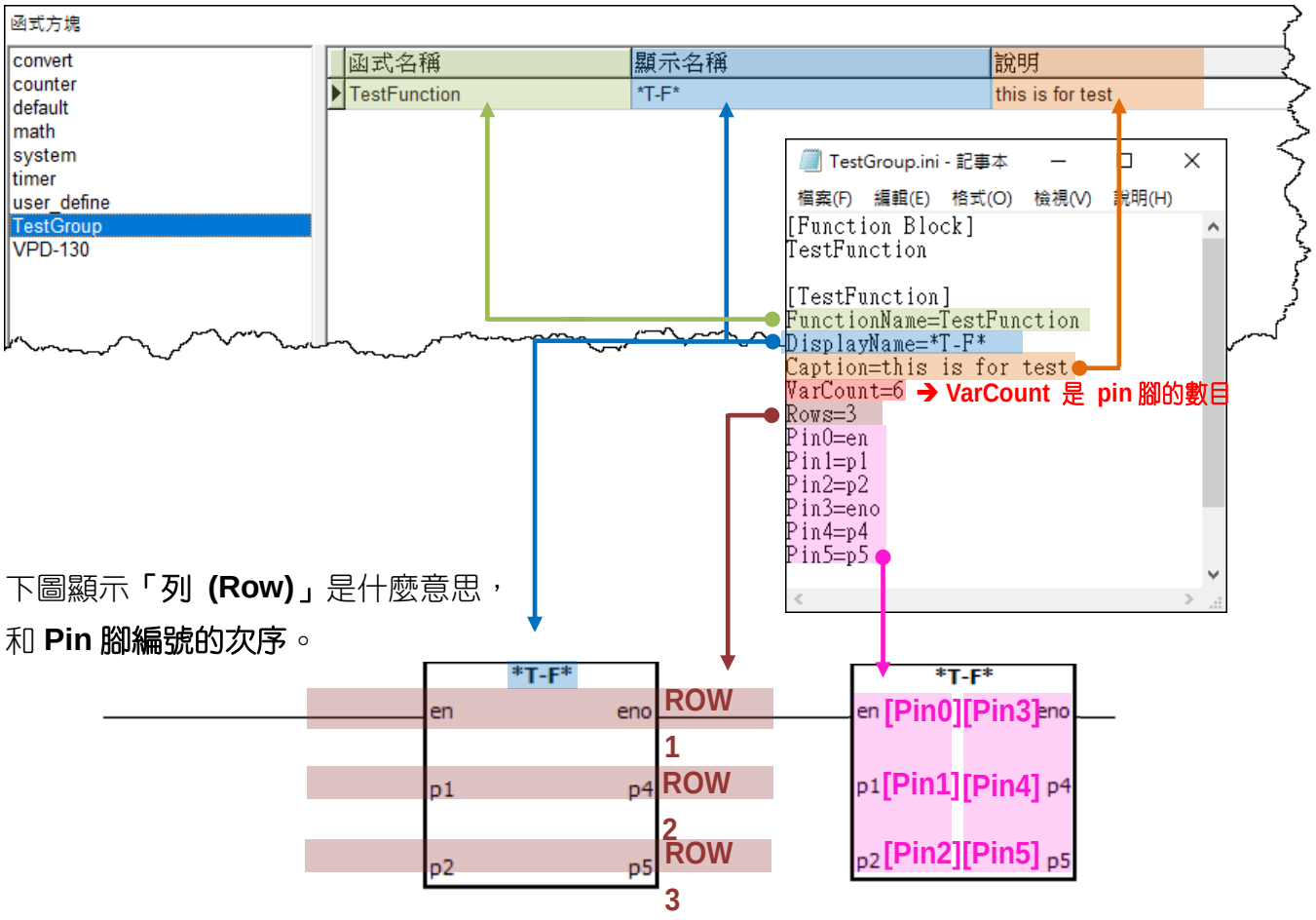


步驟 3: 到資料夾「**TestGroup**」下，新增一個 **“.ini”** 檔，新的 **“.ini”** 檔一定要和群組的名稱一模一樣，即名稱是「**TestGroup**」。然後在資料夾「**TestGroup**」下，新增一個資料夾，舉例而言叫「**TestFunction**」，新增資料夾的名稱必需是即將新增函數的名稱。



步驟 4: 最後，在 **“TestGroup.ini”** 檔案中，定義新的函式「**TestFunction**」。

您可以開啟 **“函式方塊”** 視窗來檢查剛新增完成的新函式 (如: TestFunction)。



下圖顯示「**列 (Row)**」是什麼意思，和 **Pin 腳編號的次序**。

步驟 5: 在 **“TestFunction”** 資料夾中，新增一個檔案 **“FB.hsf”**。 **“FB.hsf”** 就是寫使用者定義的函式的檔案。

3.3.6 使用工具和標籤產生關連

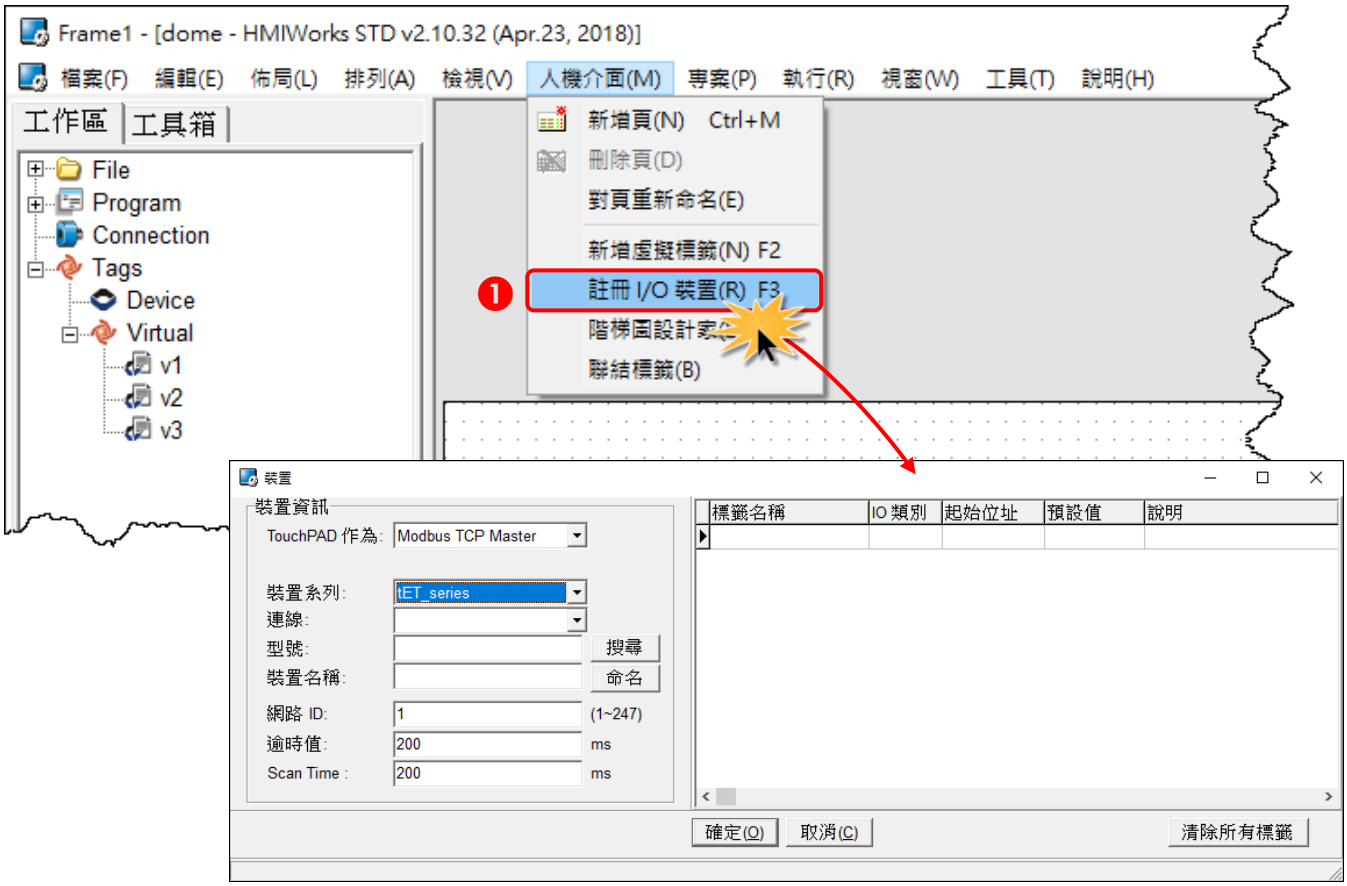
欲使**階梯圖設計家**設計的邏輯可以和 TouchPAD 的人機介面產生關連，即欲使**階梯圖設計家**可以和控制觸控螢幕上的工具 (Tool) 產生互動，就必須使工具和標籤 (Tag) 產生關連。

3.3.6.1 新增註冊 I/O 裝置 (F3)

要將工具和標籤 (Tag) 產生關連，首先新增 「**裝置(Device)**」 標籤，詳細步驟如下:

步驟 1: 從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(R)”** 項目來開啟 “裝置” 配置視窗。

> 小技巧
>
> 1. 按快捷鍵 <F3> 。
> 2. “工作區” 裡，在 “Device” 項目上按右鍵，在彈出的功能選單中，點選 “New Device”。



步驟 2: 在 **“裝置”** 配置視窗，依序選擇或填入相關裝置資訊，然後按下 **“確定(O)”** 來匯入標籤。這裡我們以 M-7060 模組為範例。

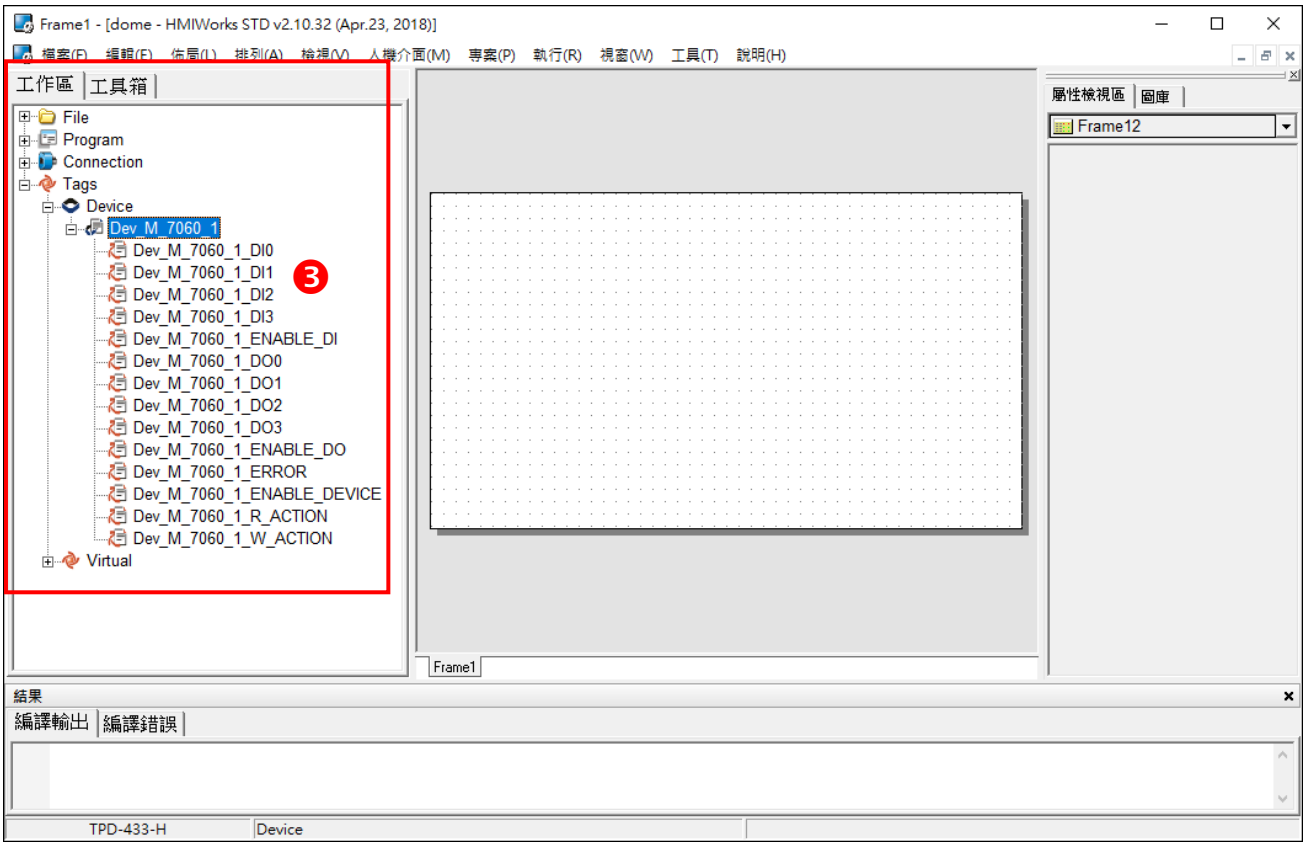
“裝置資訊” 設定項目詳細說明如下表:

項目	說明
TouchPAD 作為	設定 TouchPAD 是主裝置 (Master) 或 從裝置 (Slave) 及通訊協定，詳細請參考 表 3-1。
裝置系列	設定裝置類型，詳細請參考 表 3-1。
連線	設定連線到 I/O 模組的通訊方法 (TCP/IP、XVboard 或 COM Port)。 註: 當 **TouchPAD** 作為 **Modbus TCP Slave** 時，在 “**工作區**” 中找到使用的連線項目，按滑鼠右鍵叫出 “**新增/編輯連線**” 的對話框，並勾選 “**作為伺服器**” 的選項。
型號	設定想要連接的 I/O 模組的型號。
裝置名稱	指定想要連接的 I/O 模組的名稱。該名稱使用者可自定。
網路 ID	當 **TouchPAD 作為主站裝置 (Master)** 時 (主從式架構下)，網路 ID 是想要連接的 I/O 模組在網路中的 ID。 當 **TouchPAD 作為從站裝置 (Slave)** 時 (主從式架構下)，網路 ID 是 TouchPAD 自身在網路中的 ID。
逾時值	設定通訊時的逾時值 (預設: 200 ms)。
Scan Time	設定裝置標籤的更新時間 (預設: 200 ms)。

表 3-1: **“TouchPAD 作為”** 及 **“裝置系列”** 設定項目詳細說明如下:

TouchPAD 作為	裝置系列	裝置系列說明
Modbus RTU Master	**M-7000**	遠端 I/O 模組 (Modbus RTU 通訊協定)
	DL_series_MRTUM	遠端溫濕度模組 (Modbus RTU 通訊協定)
	tM_series	遠端精簡型 I/O 模組 (Modbus RTU 通訊協定)
	LC_series	燈控模組 (Modbus RTU 通訊協定)
	PM_series	電力量測模組 (Modbus RTU 通訊協定)
	IR_series	紅外線學習遙控模組 (Modbus RTU 通訊協定)
	PIR_series	被動式人體紅外線感測與溫度感測模組 (Modbus RTU 通訊協定)
	XVBoard	VPD 系列 IO 擴充卡
	User_Define(MRTUM)	使用者自定 非本公司 (泓格) 的遠端 Modbus RTU I/O 模組
	GateWay(MRTUM)	DALI 閘道器 (Modbus RTU 通訊協定)
	Example(MRTUM)	其它例外模組 (客製化模組)
Modbus RTU Slave	**Profiles(MRTUS)**	TouchPAD 被當成從端 (Modbus RTU Slave) 裝置，等待主從式的主端來控制
Modbus ASCII Master	**User_Define(MASCM)**	使用者自定 非本公司 (泓格) 的遠端 Modbus ASCII I/O 模組
Modbus ASCII Slave	**Profiles(MASCS)**	TouchPAD 被當成從端 (Modbus ASCII Slave) 裝置，等待主從式的主端來控制
DCON Master	**I-7000**	遠端 I/O 模組 (DCON 通訊協定)
	DL_series_DCON	遠端溫濕度模組 (DCON 通訊協定)
	tM_series_DCON	遠端精簡型 I/O 模組 (DCON 通訊協定)
Modbus TCP Master	**tET_series**	遠端精簡型 I/O 模組 (Modbus TCP 通訊協定)
	PET-7000	遠端 I/O 模組 (Modbus TCP 通訊協定)
	WISE-7000	WISE 控制器 (Modbus TCP 通訊協定)
	User_Define(MTCPM)	使用者自定 非本公司 (泓格) 的遠端 Modbus TCP I/O 模組
	Example(MTCPM)	其它例外模組 (客製化模組)
Modbus TCP Slave	**Profiles(MTCPS)**	TouchPAD 被當成從端 (Modbus TCP Slave) 裝置，等待主從式的主端來控制

步驟 3: 完成一個裝置的新增，可以在 **“工作區”** 中看見新匯入的 **“Dev_M_7060_1”** 裝置標籤。

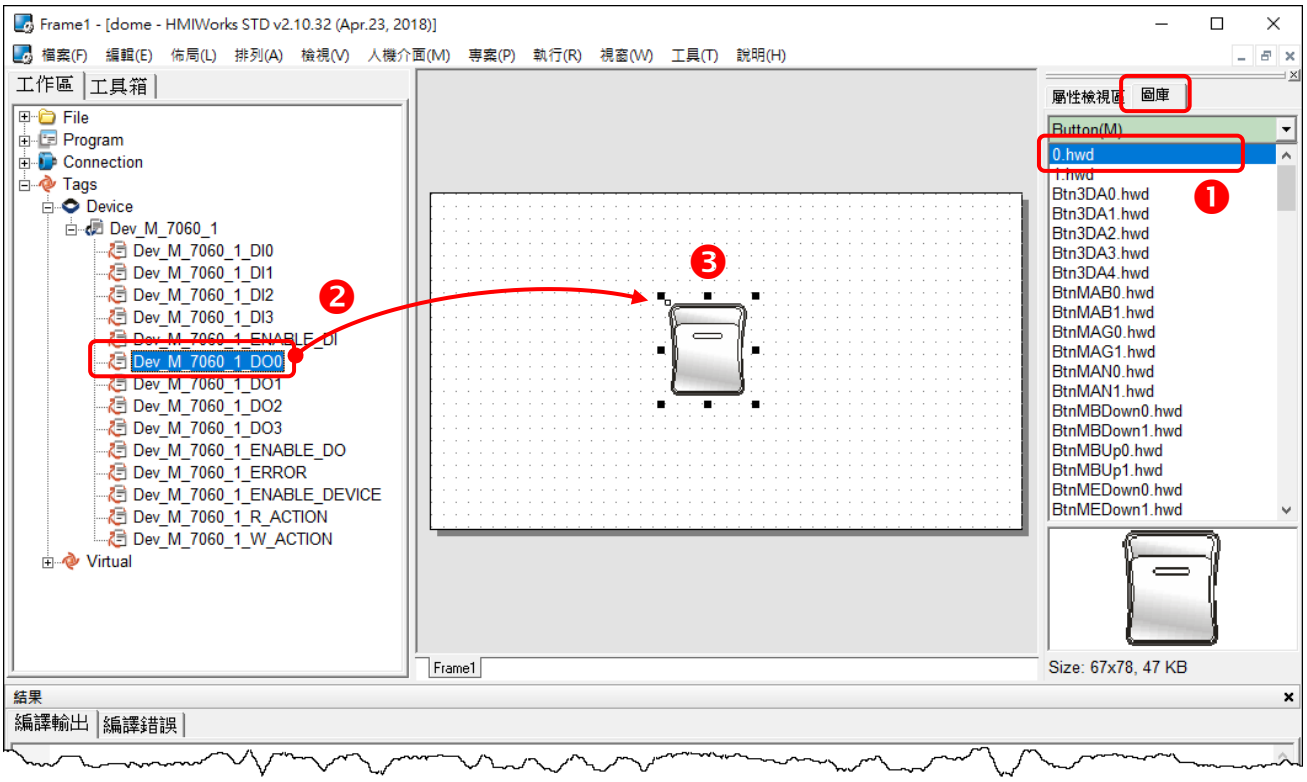


3.3.6.2 三種方式使工具和標籤產生關連

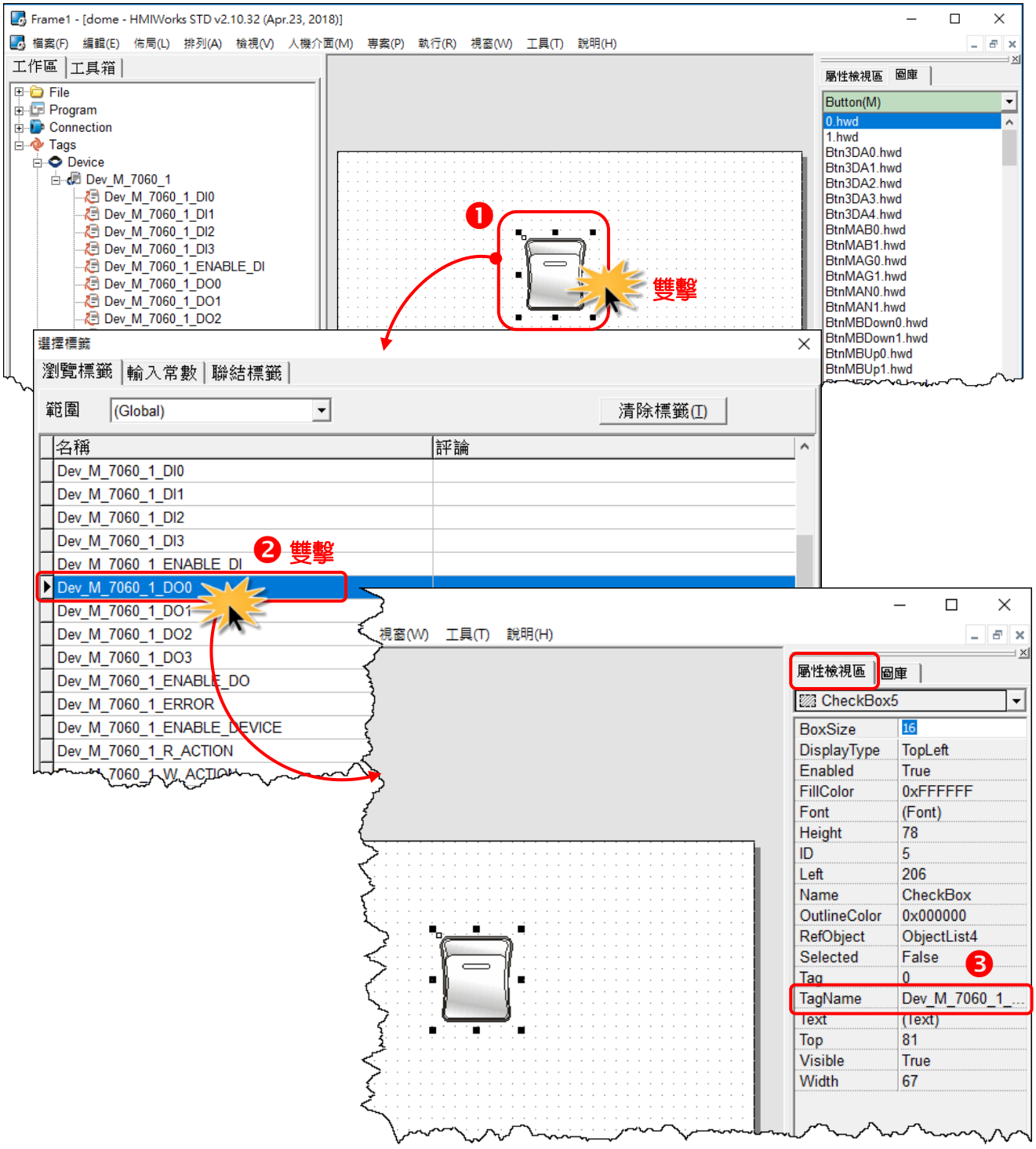
方法一: 把標籤從“**工作區**”拖放到設計頁面區，該標籤會和自動產生的 CheckBox 產生關連，如下圖所示。

注意: 自動拖放只會產生核取方塊。

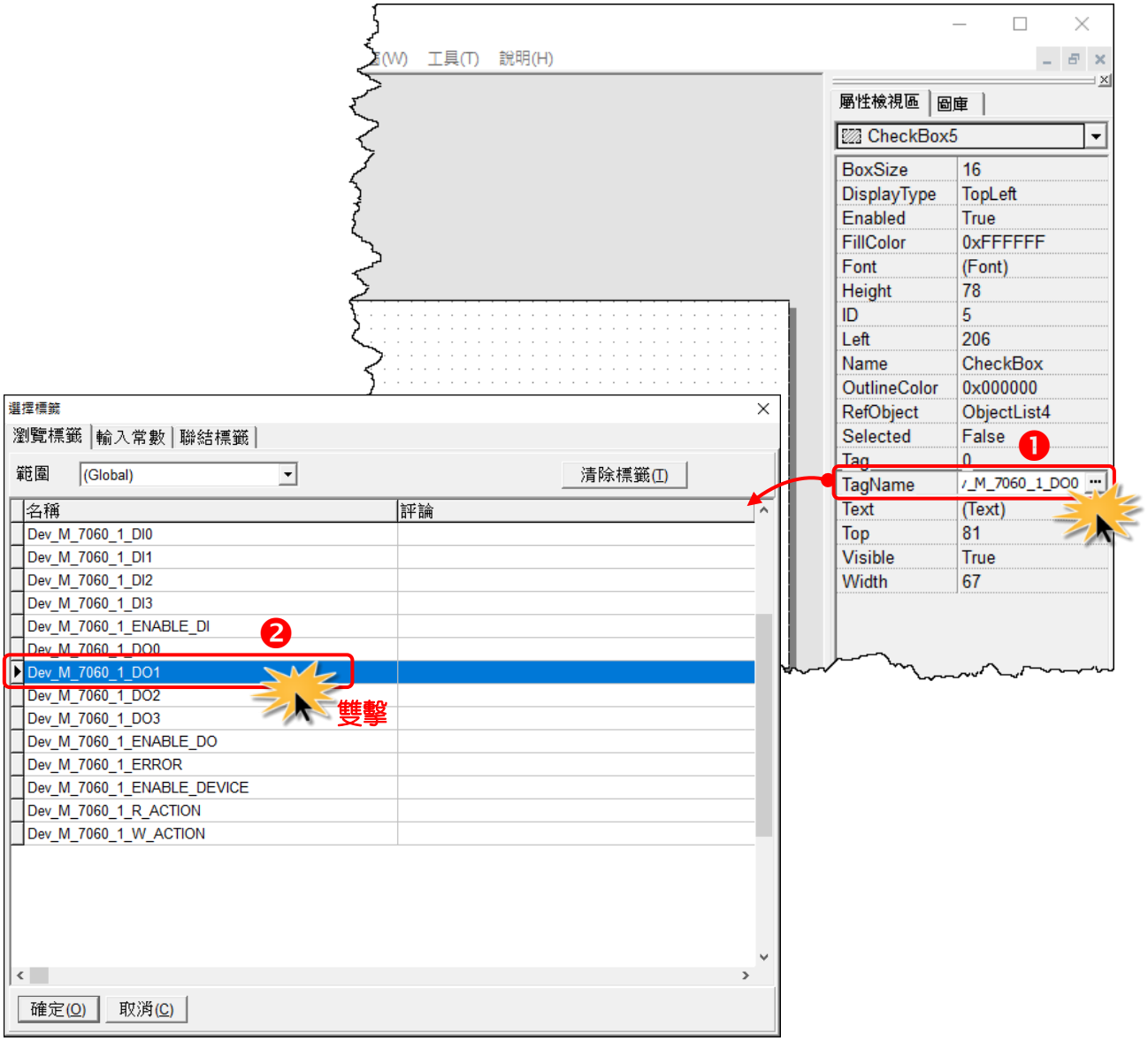
1. 在“**圖庫**”中挑一個圖案來代表標籤。
2. 在“**工作區**”中選擇一個標籤。
3. 拖放標籤 (I/O 通道) 到設計頁面區。



方法二: 雙擊位於設計頁面區的 CheckBox。接著出現 **“選擇標籤”** 的配置視窗，雙擊您所需的標籤，如此，關連就完成了。接著可以從 **“屬性檢視區”** 的 **“TagName”** 欄位查看到剛所設定的標籤名稱，確認關連的產生。



方法三: 點選 **“屬性檢視區”** 中，屬性 **“TagName”** 右側的 **“…”** 鈕來開啟 **“選擇標籤”**的視窗。接著如同上一步驟的做法就可以完成關連了。

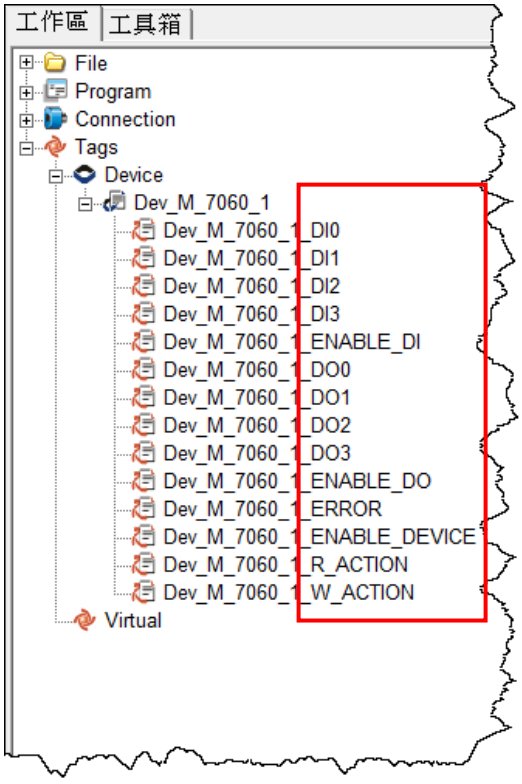


注意

參考 第 3.4.17 節 ObjectList。 將 **CheckBox** 的 **RefObject** 屬性設定為某一個物件清單，會使 **CheckBox** 在觸控螢幕上選取和不選取的兩個狀態，被替換成物件清單中的兩張圖片。接著再把階梯圖設計家中的標籤和 **CheckBox** 產生關連，就可以產生只要標籤的值一變動，觸控螢幕的圖也會跟著變換的效果了。這項功能是專門為了在觸控螢幕上顯示開關輸入所設計的。

3.3.6.3 裝置 (Device) 標籤說明

當完成裝置標籤建立，將會顯示該裝置相關的 Device 標籤，包含 **ENABLE_DO**、**R_ACTION**、**W_ACTION**、**ERROR** 及 **ENABLE_DEVICE** …等，此章節將詳細說明這些 Device 標籤的功用。



項目	說明
DIn	Digital Input 通道
DOn	Digital Output 通道
AIn	Analog Input 通道
AOn	Analog Output 通道
ENABLE_DI	可用來關閉及啟用 DI Group **1:** 啟用; **0:** 關閉
ENABLE_DO	可用來關閉及啟用 DO Group **1:** 啟用; **0:** 關閉
ENABLE_AI	可用來關閉及啟用 AI Group **1:** 啟用; **0:** 關閉
ENABLE_AO	可用來關閉及啟用 AO Group **1:** 啟用; **0:** 關閉
ENABLE_DEVICE	可用來關閉及啟用在此設備上所有讀取和寫入動作 **1:** 啟用; **0:** 關閉

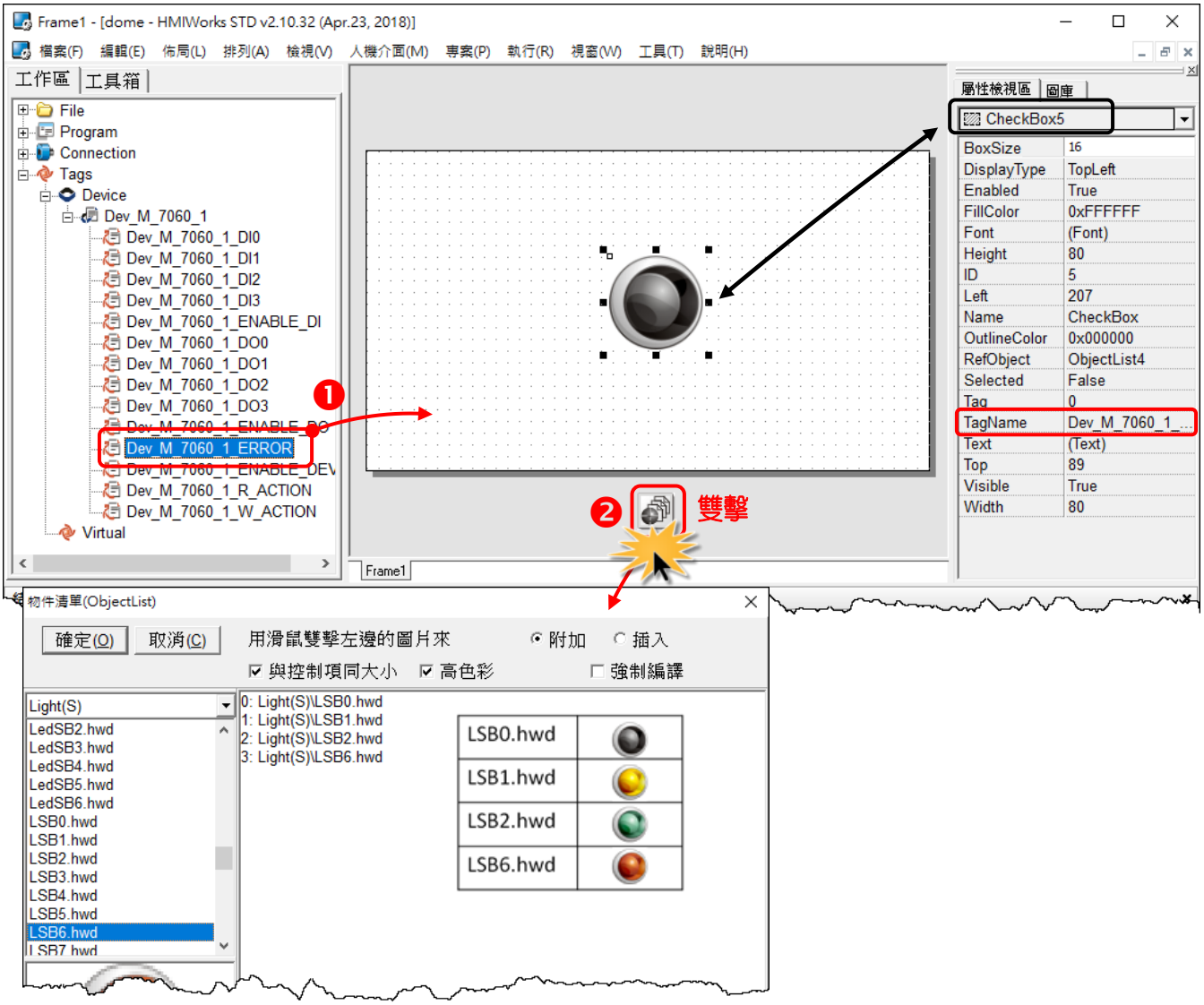
項目	說明
R_ACTION	可用來關閉及啟用 Read Action (包含 DI/DO/AI/AO Groups) **1:** 啟用; **0:** 關閉
W_ACTION	可用來關閉及啟用 Write Action (包含 DO 或 AO Group) **1:** 啟用; **0:** 關閉
ADDR_BASE	可用來配置 Base Address 的移位。因為 Modbus 協議對此沒有標準，某些設備的 Base Address 是使用 0，其它設備使用 1，因此使用者必需參考各自設備手冊來分配正確的位址
ERROR	確認通訊狀態，**1:** 通訊錯誤; **0:** 通訊成功 在下一頁中，我們將介紹 “ERROR” 標籤的閃爍週期應用

重連閃爍週期

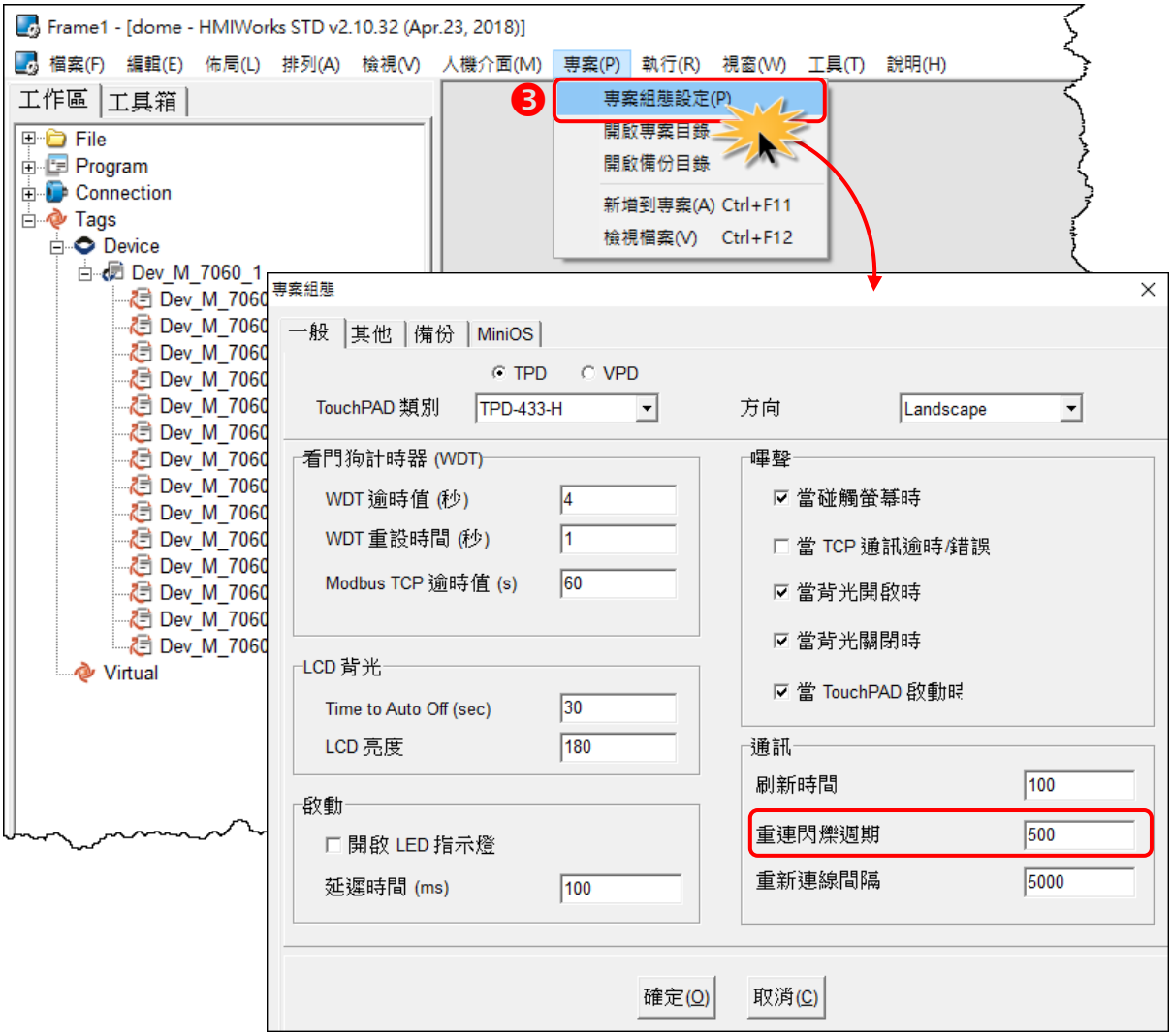
只用於 TouchPAD 作為 Modbus TCP Master (主站) 輪詢 (polling) 時，「**重連閃爍週期**」定義了 **“ERROR”** 標籤在重新連線時閃爍的週期，該標籤可以在 **“工作區”** 中的 **“Device”** 內找到。

設定「重連閃爍週期」的步驟如下:

1. 拖放 **“ERROR”** 標籤到設計頁面區。
   使用 CheckBox 作為與遠端 Modbus TCP Slave (從站) 通訊狀態的圖示，而 TouchPAD 是 Master (主站)。
2. 雙擊 **“ObjectList”** 來開啟 **“物件清單 (ObjectList)”** 的配置視窗，並指定 4 張圖片。
   該 CheckBox 被指定了一個含有 4 張圖片的物件清單。為與舊版相容，**前兩張圖片**仍是代表「**通訊正常**」及「**通訊錯誤**」，而第三和第四張則會在 TouchPAD 重新連線時交叉出現形成閃爍效果。



3. 從 **“專案(P)”** 功能選單中，點選 **“專案組態設定(P)”** 項目來開啟 “專案組態” 配置視窗來設定 **“重連閃爍週期”**。



3.3.7 使用者定義的 I/O 模組

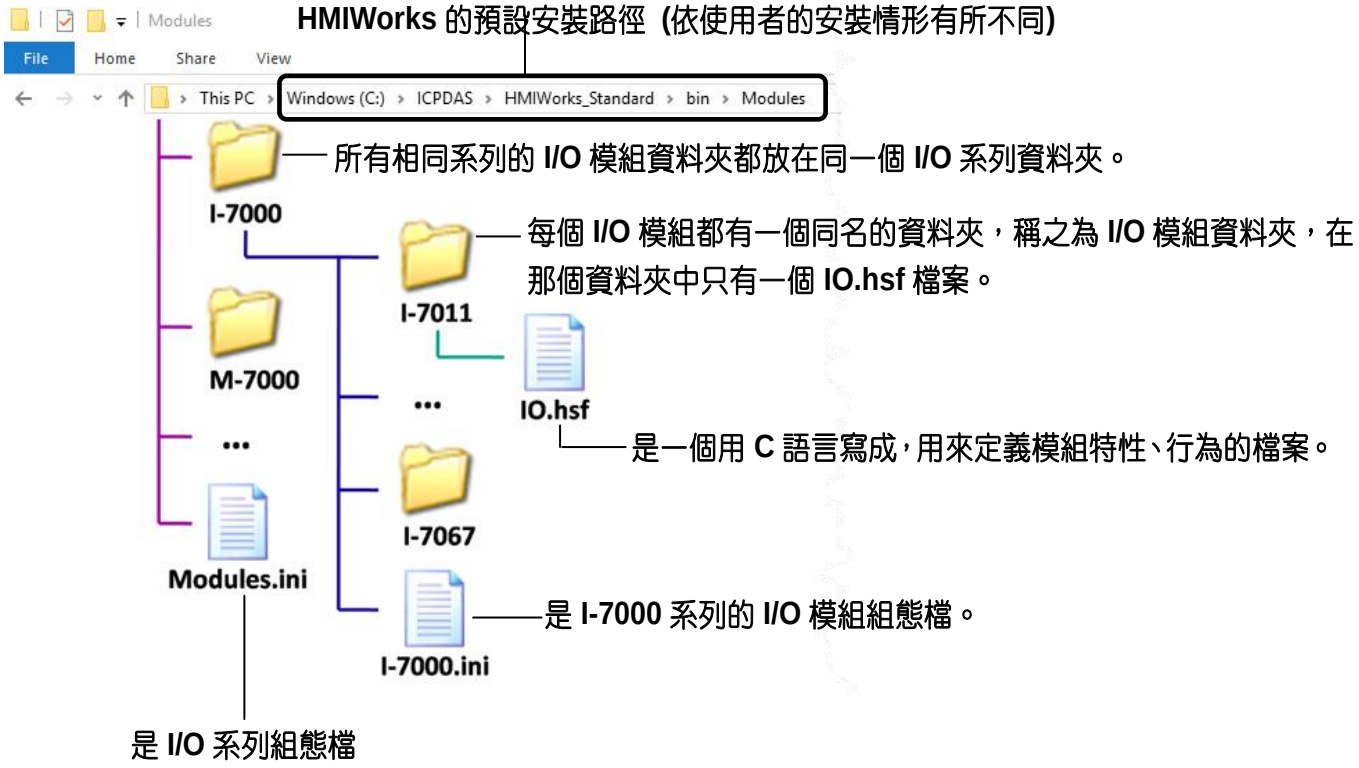
為了說明如何新增一個使用者定義的 I/O 模組，我們先說明 HMIWorks 如何處理 I/O 模組的。

泓格 I/O 模組可分為好幾種，主要有下面三大項:

型號	說明
I-7000 系列	DCON I/O 模組 WebSite:https://www.icpdas.com/en/product/guide+Remote_I_O_Module_and_Unit+RS-485_I_O_Modules+I-7000
M-7000 系列	Modbus RTU I/O 模組 WebSite:https://www.icpdas.com/en/product/guide+Remote_I_O_Module_and_Unit+RS-485_I_O_Modules+I-7000
ET/PET-7000 系列	Modbus TCP I/O 模組 WebSite:https://www.icpdas.com/en/product/guide+Remote_I_O_Module_and_Unit+Ethernet_I_O_Modules+ET-7000

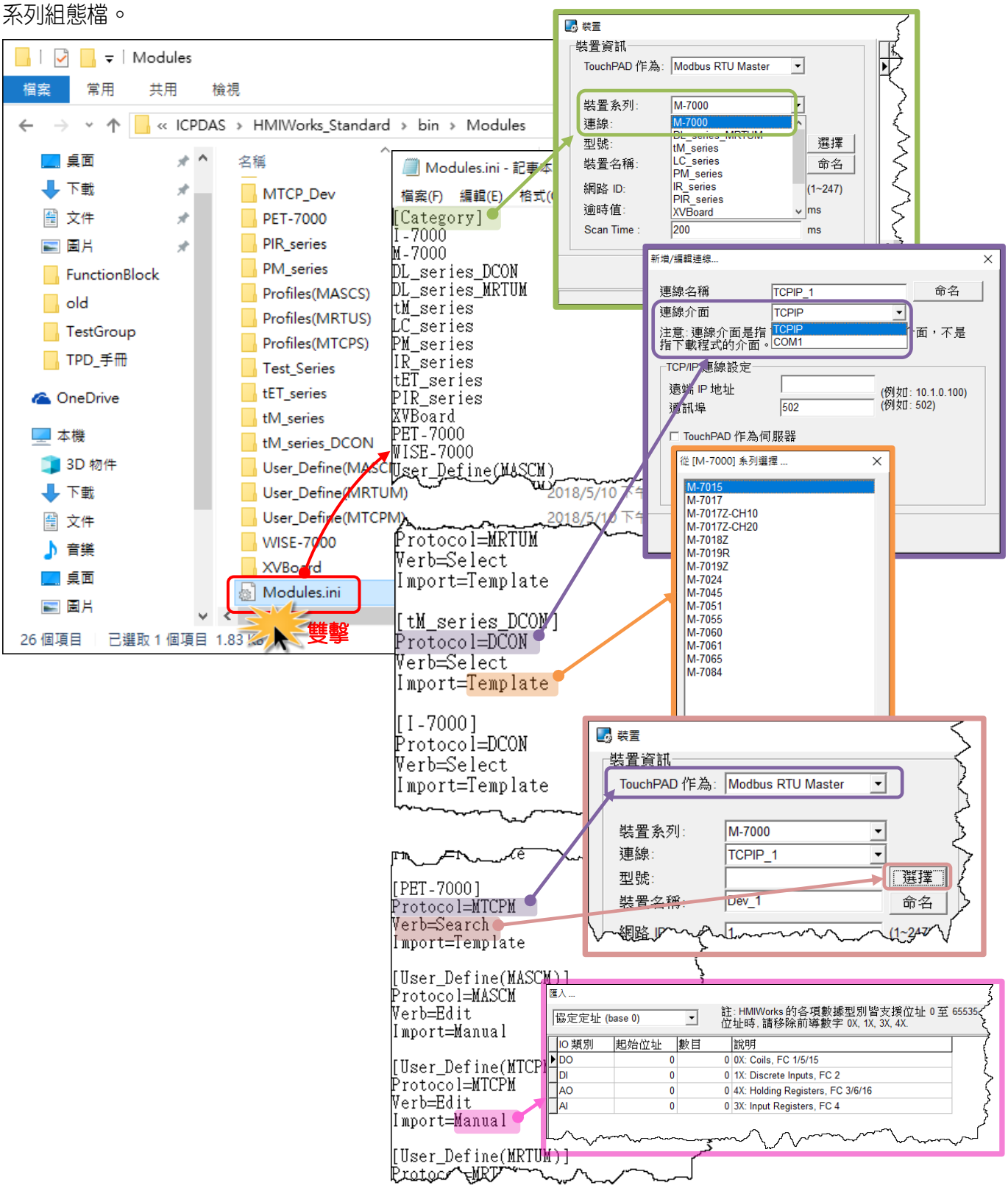
3.3.7.1 I/O 裝置的資訊放在什麼地方?

HMIWorks 將 I/O 模組的資訊放在如下圖所示的位置裡。



Module.ini 描述了什麼?

到 HMIWorks 的安裝路徑下，在子目錄「**bin\Modules**」中，雙擊 **“Module.ini”** 來開啟 I/O 系列組態檔。

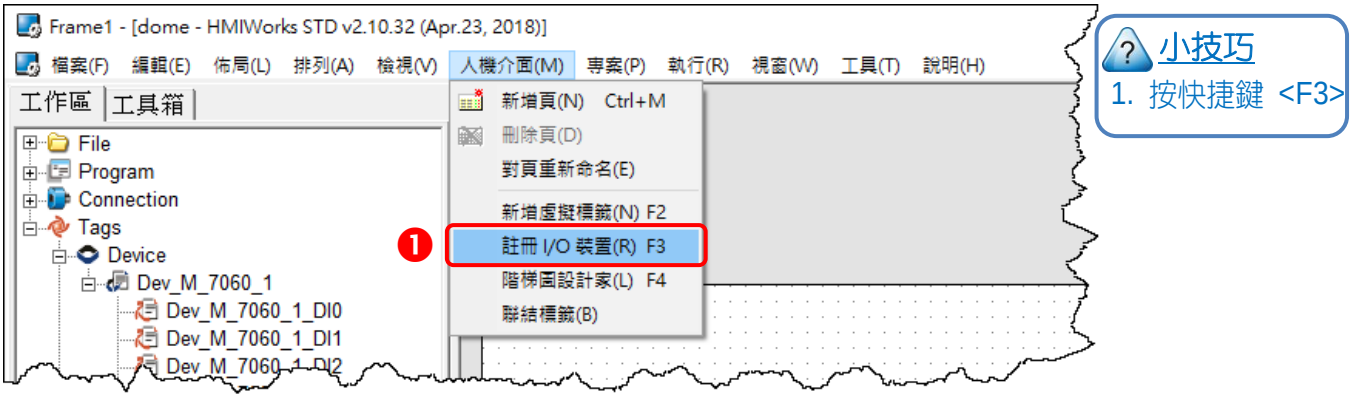


“Module.ini” 設定項目詳細說明如下表:

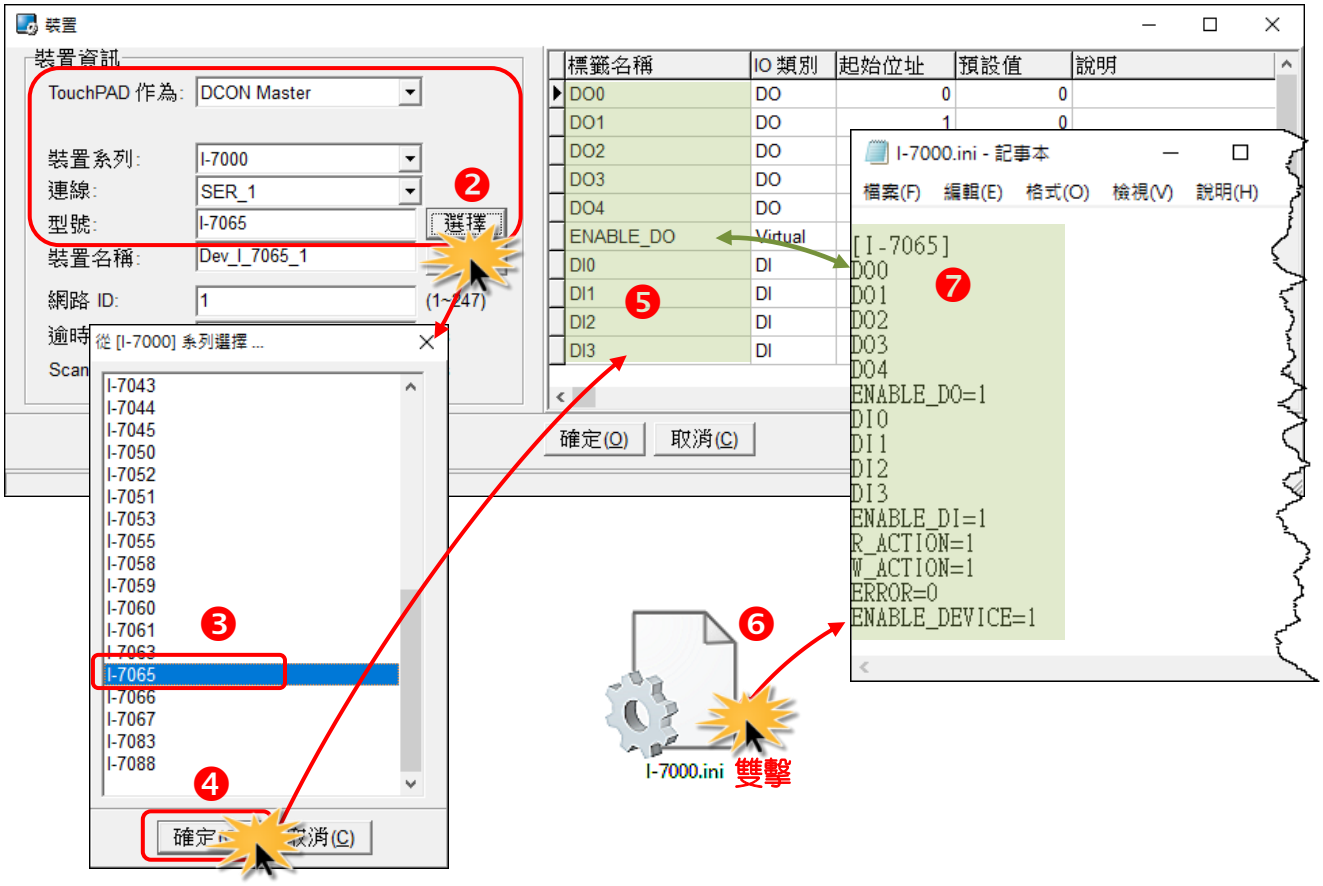
項目		說明
類別 (Category)		這個區塊描述了 HMIWorks 支援那些 I/O 系列。當**註冊裝置(F3)**時，“**裝置”** 視窗會從這裡讀取 I/O 系列的訊息。
通訊協定 (Protocol)	**MTCPM**	在 **Module.ini** 中的 **“Protocol=MTCPM”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: Modbus TCP Master**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = TCPIP”**。
	MTCPS	在 **Module.ini** 中的 **“Protocol=MTCPS”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: Modbus TCP Slave**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = TCPIP”**。
	MASCM	在 **Module.ini** 中的 **“Protocol=MASCM”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: Modbus ASCII Master**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = COM Port”**。
	MRTUM	在 **Module.ini** 中的 **“Protocol=MRTUM”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: Modbus RTU Master**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = COM Port”**。
	MASCS	在 **Module.ini** 中的 **“Protocol=MASCS”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: Modbus ASCII Salve**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = COM Port”**。
	MRTUS	在 **Module.ini** 中的 **“Protocol= MRTUS”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: Modbus RTU Salve**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = COM Port”**。
	DCON	在 **Module.ini** 中的 **“Protocol= DCON”** 對應到 “裝置” 視窗的 “**TouchPAD 作為: DCON Master**” 及 “新增/編輯連線…” 視窗的 **“連線介面 = COM Port”**。
動作 (Verb)	**搜尋 (Search)**	HMIWorks 掃描整個網路以找出 I/O 模組。
	選擇 (Select)	HMIWorks 顯示 I/O 模組的清單，讓使用者選擇。這個 I/O 模組清單是從 **[I/O 系列名稱].ini** 這個檔案載入的。
	編輯 (Edit)	HMIWorks 叫出 **“匯入”** 視窗，讓使用者決定 I/O 模組的 I/O 點數。
匯入 (Import)	**樣板 (Template)**	HMIWorks 從 I/O 模組組態檔匯入標籤。 例如: HMIWorks 從 **I-7000.ini** 的樣板中，匯入 I-7011 的標籤。
	自訂 (Manual)	HMIWorks 由使用者手動設定的 I/O 點數來匯入標籤。

“註冊 I/O 裝置(R)” 如何產生標籤

從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(R)”** 項目來開啟“裝置”配置視窗。



I/O 模組組態檔內有該所屬 I/O 系列中的所有 I/O 模組的樣板 (Templates)，如 **I-7000.ini** 是 I-7000 I/O 系列的 I/O 模組組態檔。下面我們將以 I-7000 I/O 系列中的 I-7065 為例來說明。



“IO.hsf” 如何定義 I/O 的特性和行為

以 I-7065 為例 (I-7000 系列 I/O 裝置)，在 **“[HMIWorks 安裝路徑]\bin\Modules\I-7000\I-7065\”** 資料夾中，開啟 “**IO.hsf**”。 **“IO.hsf”** 是用 C 語言寫成的，如下程式碼。

```
BEGIN_FUNCTION_BLOCK(); //this line is necessary

DWORD v_do = 0;
DWORD v_di = 0;
int      gWriteCount = 0;

uart_SetTimeout($DEVICE, $TIMEOUT);

//$W_ACTION: a tag used in Ladder to enable/disable writing actions
//$ENABLE_DO: a tag used in Ladder to enable/disable the part of DOs
if ( VAR_VALUE($ENABLE_DO) && VAR_VALUE($W_ACTION))
{
     int iWrite = 0; //To decide if there's a need to write any DO channel
     v_do      = 0;

     // Update the status for each channel if it has been changed.
     iWrite += VAR_GET_WRITE_U32(&v_do, $DO0, 0);
     iWrite += VAR_GET_WRITE_U32(&v_do, $DO1, 1);
     iWrite += VAR_GET_WRITE_U32(&v_do, $DO2, 2);
     iWrite += VAR_GET_WRITE_U32(&v_do, $DO3, 3);
     iWrite += VAR_GET_WRITE_U32(&v_do, $DO4, 4);

     if ( iWrite )    // Write only when need
     {
          gWriteCount++;
          if ( ! dcon_WriteDO($DEVICE, $NETID, 5, v_do & 0xFF) )
            // dcon_WriteDO: the DO writing API function of I-7000 I/O series.
            // I-7000 I/O series uses the DCON protocol.
                return HMI_ERROR;
     }
}

if ( gWriteCount ) return HMI_OK;
// Skip reading to reduce the device loading

if ( (VAR_VALUE($ENABLE_DO) || VAR_VALUE($ENABLE_DI)) && VAR_VALUE($R_ACTION)) {
//$R_ACTION: a tag used in Ladder to enable/disable reading actions
//$ENABLE_DO: a tag used in Ladder to enable/disable the part of DOs
//$ENABLE_DI: a tag used in Ladder to enable/disable the part of Dis
if (dcon_ReadDIO($DEVICE, $NETID, 4, 5, &v_di, &v_do))
```

```
// dcon_ReadDIO: the DI/DO reading API function of I-7000 I/O series.
// I-7000 I/O series uses the DCON protocol.
{
        VAR_SET($DI0, v_di & (1<<0));
       // VAR_SET: used to set the value of this channel to its tag
        VAR_SET($DI1, v_di & (1<<1));
        VAR_SET($DI2, v_di & (1<<2));
        VAR_SET($DI3, v_di & (1<<3));

        VAR_SET($DO0, v_do & (1<<0));
        VAR_SET($DO1, v_do & (1<<1));
        VAR_SET($DO2, v_do & (1<<2));
        VAR_SET($DO3, v_do & (1<<3));
        VAR_SET($DO4, v_do & (1<<4));
} else
      return HMI_ERROR;
}

END_FUNCTION_BLOCK(); //this line is necessary
```

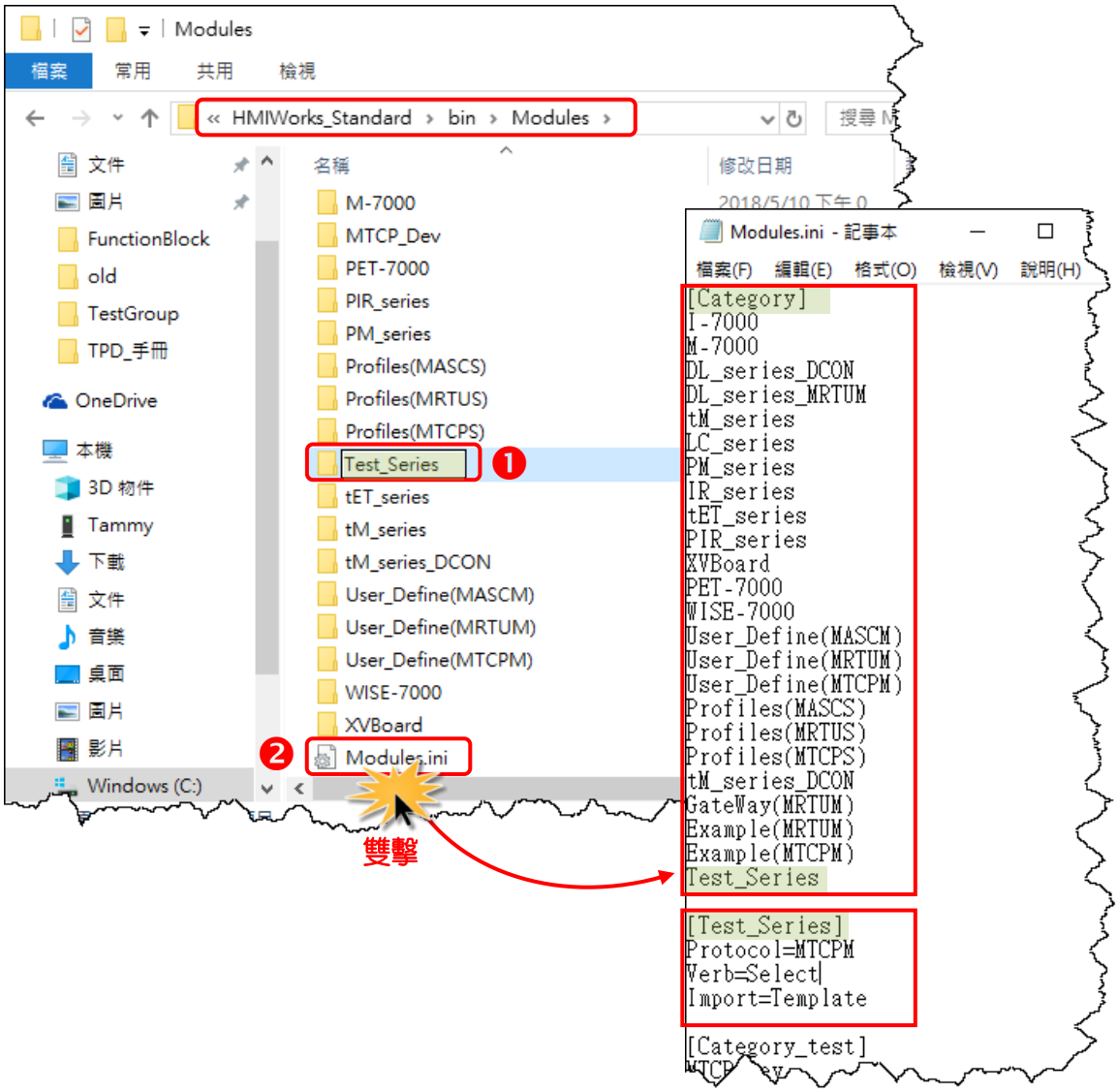
3.3.7.2 建立使用者定義 I/O 模組

下面我們將說明使用者如何來新增自定的 I/O 模組。

步驟 1: 在 **“[HMIWorks 安裝路徑]\bin\Modules\”** 下，建立一個新的 I/O 系列資料夾，並命名為 **“Test_Series”**。

步驟 2: 雙擊 **“Modules.ini”** 來新增一個新項目 **“Test_Series”**，然後儲存新設定，這樣 HMIWorks 才知道有一個新建立的 I/O 系列，名叫 **“Test_Series”**。

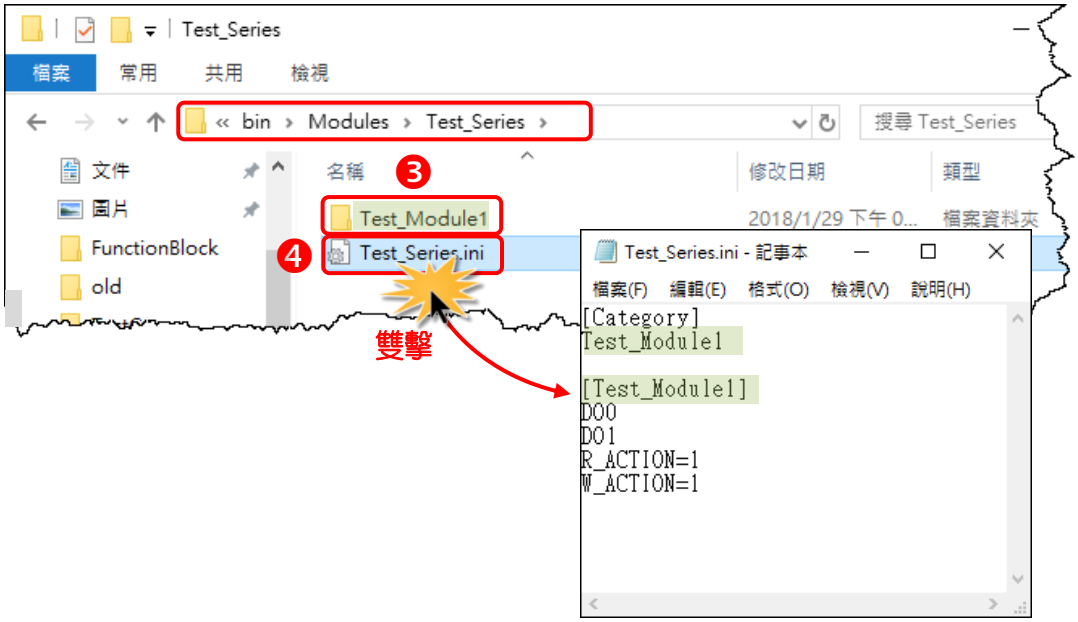
⚠ **注意: 系列資料夾和 Modules.ini 中的名稱，必須相同。**



步驟 3: 在 I/O 系列資料夾 **“Test_Series”** 中，建立一個新的 I/O 模組資料夾，並命名為 **“Test_Module1”**。

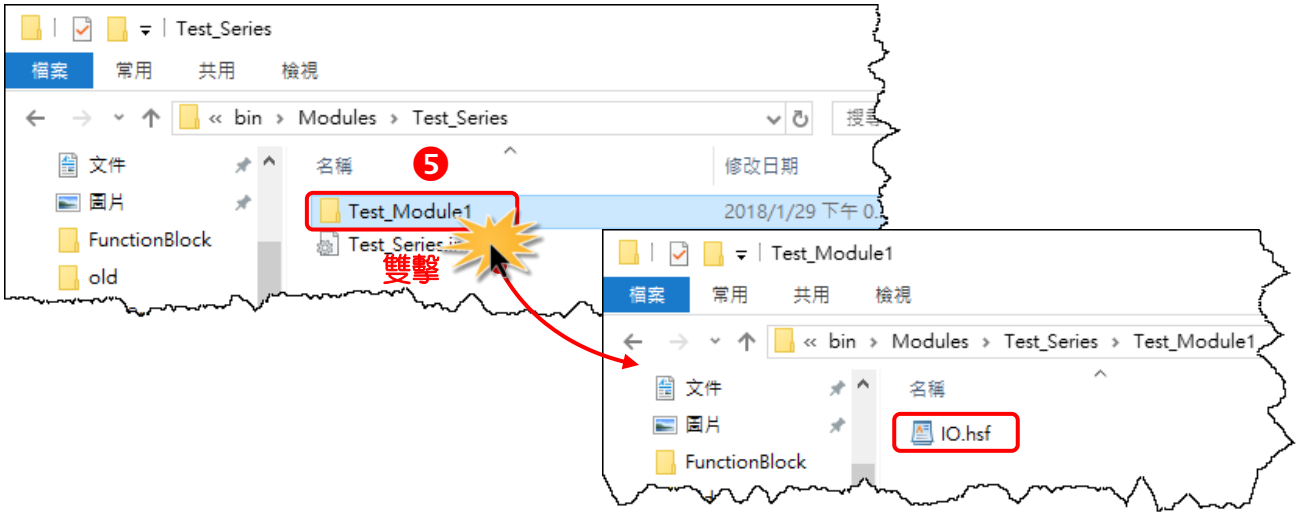
步驟 4: 建立 I/O 模組組態檔，並命名為 **“Test_Series.ini”**，該檔案是用來記載新建立的模組 **Test_Module1** 的樣板 (I/O 的數目)。

⚠注意: 模組資料夾的名稱必須和 Test_Series.ini 中的名稱相同。



步驟 5: 在 I/O 模組資料夾 **Test_Module1** 中的 **IO.hsf**，以描述 **Test_Module1** 的特性及行為。

- 若是 Modbus TCP 通訊協定的話，參考 PET-7000 系列的模組。
- 若是 Modbus RTU 通訊協定的話，參考 M-7000 系列的模組。
- 若是 DCON 通訊協定的話，參考 I-7000 系列的模組。
- 所有的 **IO.hsf** 都很相似，例如前述的 I-7065。

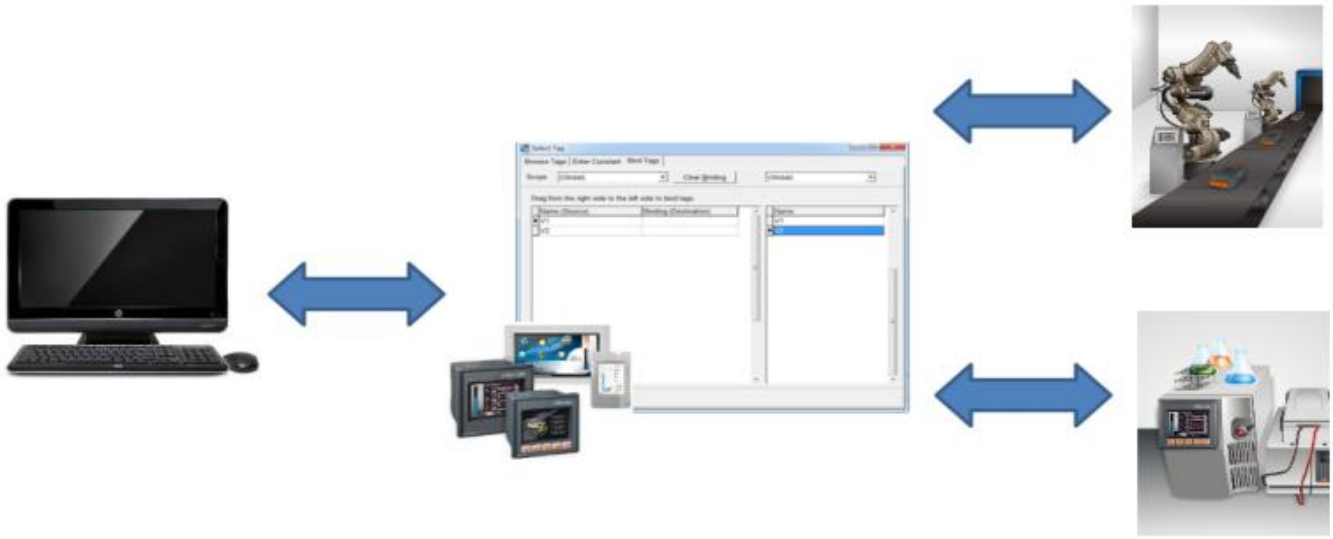


3.3.8 資料交換

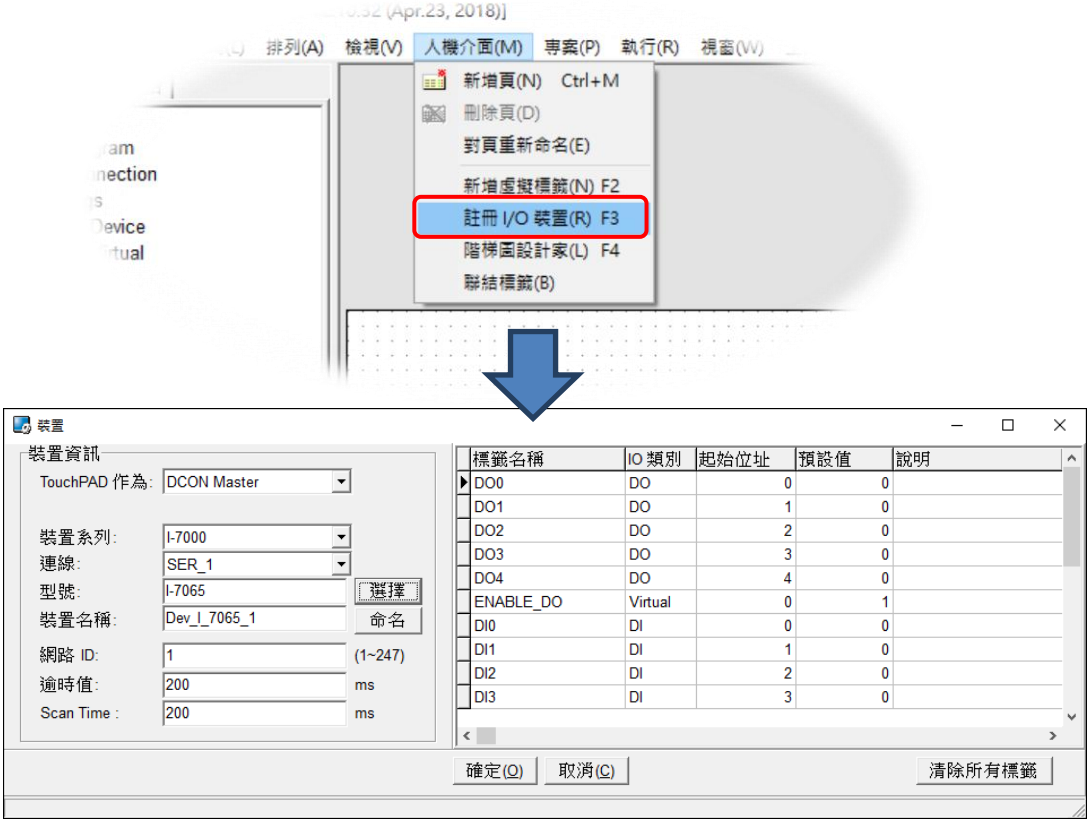
由 TouchPAD 統一標準資料格式並擔任通訊協定轉換的角色，達到不同協定之間的資料交換並解決上位機與下位機之間的資料交換問題，使裝置之間的資料傳輸能夠自動『了解』、『處理』和『回應』，並且讓現場應用更加彈性。

- 無此功能前，使用 C 語言需繁複的程式設計做轉換。

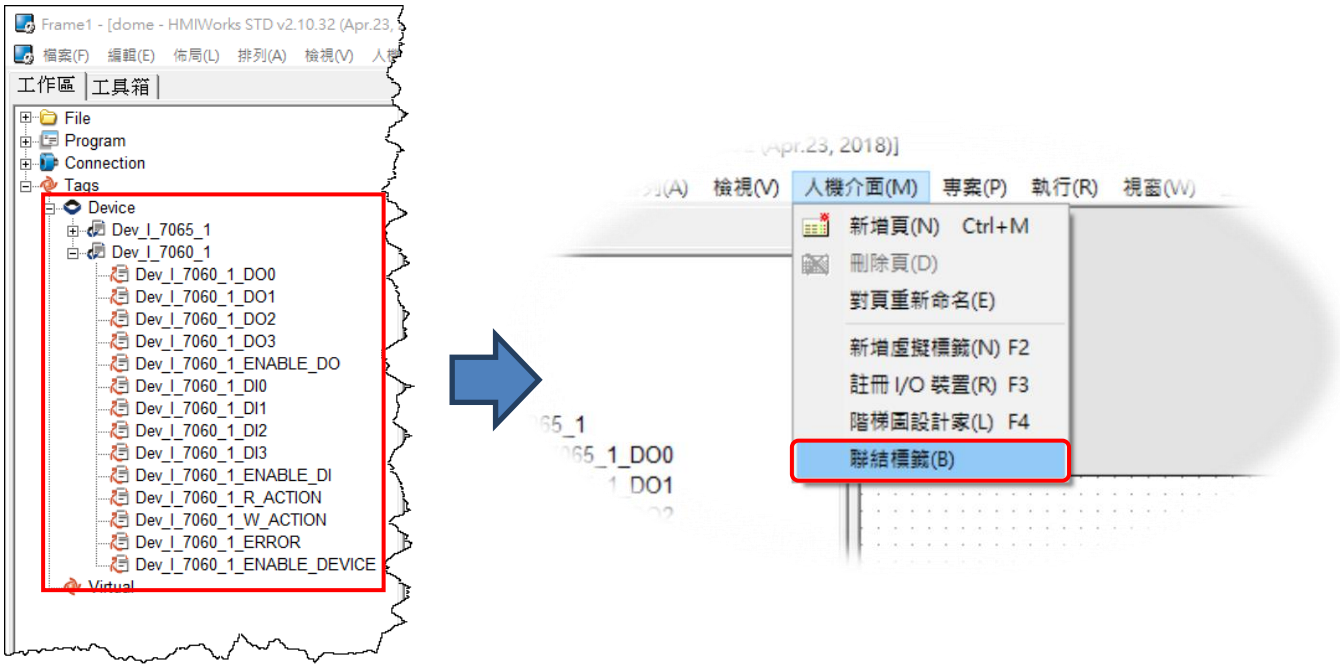
- 使用此功能後，可省去許多使用 C 語言做轉換的動作，也更方便於使用者作業，拖曳標籤後即可實現資料交換。



步驟 1: 從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(R)”** 項目來開啟 “裝置” 配置視窗(或按快捷鍵 **<F3>**)。這裡我們將建立 I-7065 及 I-7060 為範例。



步驟 2: 建立多個裝置後，即可在左方 **“工作區”** 中查看各裝置標籤，並從 **“人機介面(M)”** 功能選單中，點選 **“聯結標籤(B)”** 項目。

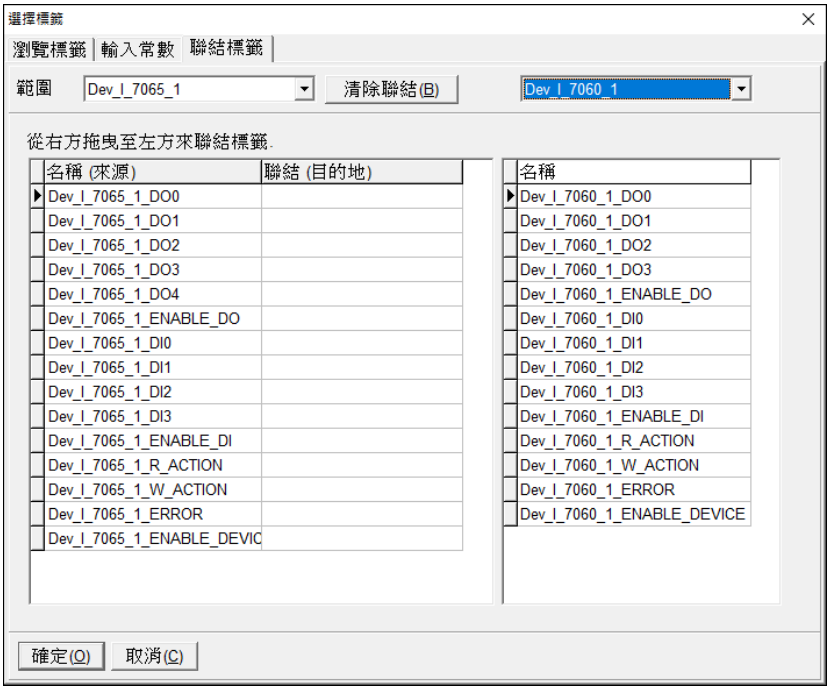


步驟 3: 拖曳標籤使標籤與標籤之間產生關聯及使用說明如下範例。

1. 從右方拖曳至左方。
2. 當 **“Dev_I_7060_1DO0”** 拖曳到 **“Dev_ I _7065_1_DO0”** 時:

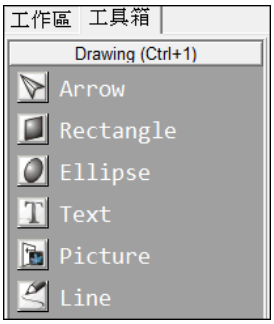
   如果 **“Dev_ I _7065_1_DO0”** 變動, 則**“Dev_ I _7060_1DO0”** 的值會更新為 **“Dev_ I _7065_1_DO0”** 的值，即 **Dev_ I _7060_1DO0 = Dev_ I _7065_1_DO0**。
3. 以此類推，當 B 拖曳到 A，C 拖曳到 B:

   如果 A 變動，則 B = A ， C = A 。



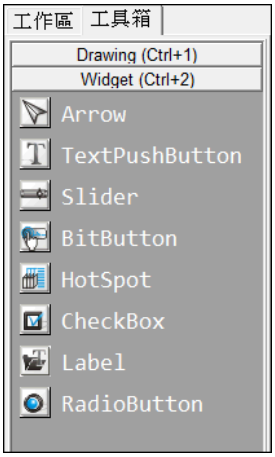
3.4 顯示頁和元件

本章節將詳細介紹顯示頁 (Frame) 及元件的屬性和用法。您可以在 “**工具箱 (Toolbox)**“ 區域中的分頁看到「**繪圖元件 (Drawing)**」、「**控制項元件 (Widget)**」和「**系統元件 (System)**」這三類元件，詳細說明如下:



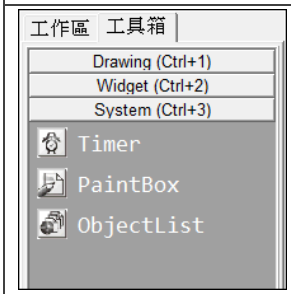
繪圖元件 Drawing (Ctrl+1):

1. **Rectangle (矩形工具)**: 用來畫一個矩形。
2. **Ellipse (橢圓工具)**: 用來畫一個橢圓。
3. **Text (恆定文字框)**: 用來顯示一個在執行時無法改變的文字。
4. **Picture (圖片框)**: 用來載入圖片於設定的範圍裡。
5. **Line (直線工具)**: 用來畫一條直線。



控制項元件 Widget (Ctrl+2):

1. **TextPushButton (文字彈回式按鈕)**: 用來產生一個上面有文字，且按下鬆開後，狀態會彈回原狀的按鈕。
2. **Slider (滑桿)**: 用來產生一個滑動拉桿可以決定大小或者是百分比的工具。
3. **BitButton (位元圖按鈕)**: 用來產生一個按下觸發後會自動彈回原狀且以圖形顯示的按鈕。
4. **HotSpot (觸發連結框)**: 用來產生一個區域，在該區域觸碰可以產生一個 **“點擊事件 (On-Click event)”**。
5. **CheckBox (核取方塊)**: 用來產生一個提供是非選擇的工具。
6. **Label (文字顯示框)**: 用來產生一個執行時可以更新文字的顯示文字區域。
7. **RadioButton (單選按鈕)**: 用來產生一個提供多選一的選擇之工具。



系統元件 System (Ctrl+3):

1. **Timer (計時器)**: 用來週期性執行一段程式碼。
2. **PaintBox (繪圖框)**: 用來在執行時繪圖於設定區域的工具。
3. **ObjectList (物件清單)**: 用來紀錄從**圖庫 (Libraries)** 中選取出來之物件的清單。必需和文字彈回式按鈕和核取方塊的“RefObject” 屬性共用。

⚠ 注意:

1. 請勿重疊控制項，否則在觸控時可能會產生無法預期的行為。
2. 兩個元件的最小間隔是 12 像素 (Pixel)。若小於 12，則可能會有按一元件而觸發相鄰元件之事件函式的錯誤。這是因為校正精確度造成的問題。

3.4.1 共同項目

本章節介紹顯示頁及 **“工具箱”** 元件的共同性質。

把元件放置於顯示頁上

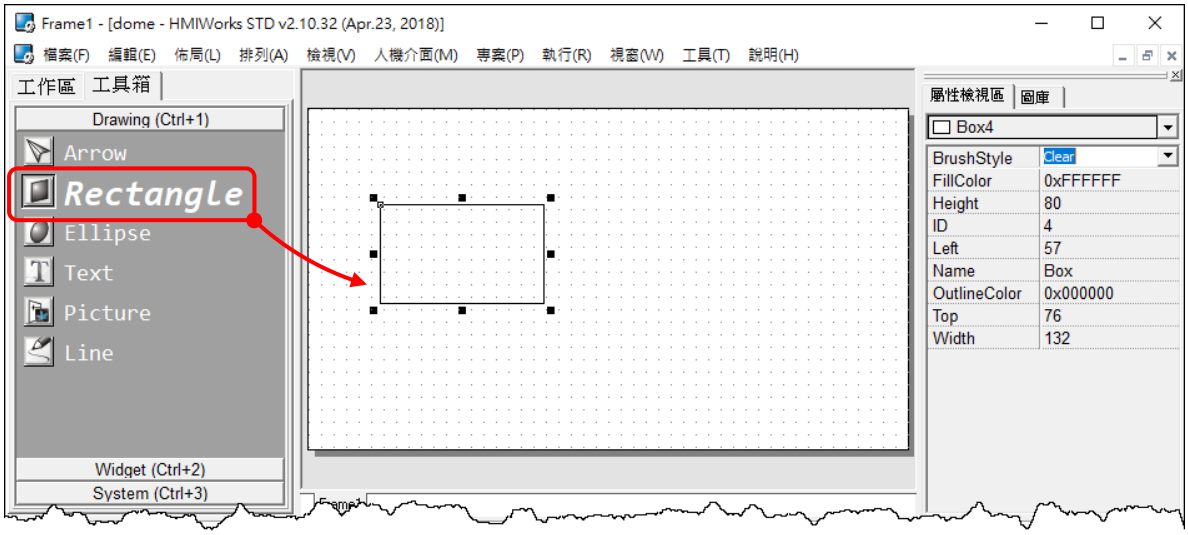
下面有兩種方法可以把元件放置於顯示頁上：

1. 拖一個適當大小的矩形
2. 選擇元件後，在頁面上用滑鼠點一下

若要拖一個適當大小的矩形，以矩形元件為例：

步驟 1: 在 **“工具箱”** 中，點選 **“繪圖元件 (Drawing)”** 的 **“Rectangle (矩形圖示)”**。

步驟 2: 將游標移至顯示頁上方，點選而拖曳出一個適當大小的矩形。



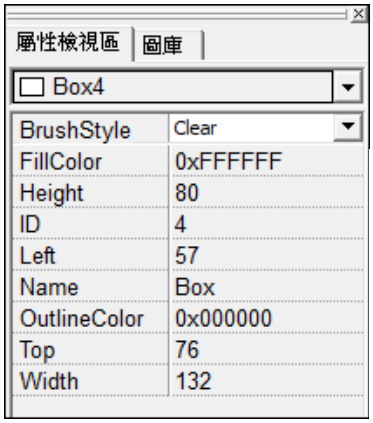
小技巧: 如何畫一個正方形?

在**步驟 2** 中，畫矩形時，按住 **<Ctrl> 鍵**不放。

小技巧: 如何畫一個圓形?

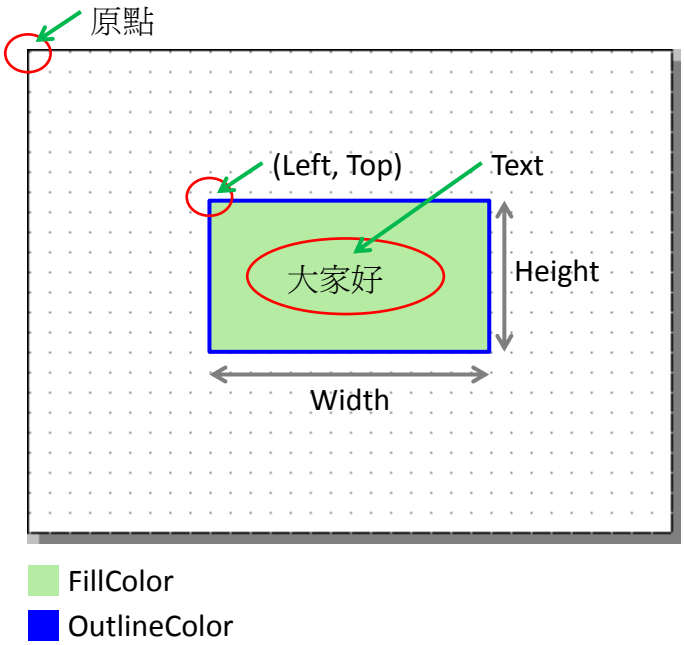
在**步驟 2** 中，在畫橢圓形時，按住 **<Ctrl> 鍵**不放。

共同屬性



小技巧: 那裡可以設定元件的屬性?

點選元件 (或顯示頁 Frame) 後，可在 “**屬性檢視區**” 設定屬性。



“**屬性檢視區**” 設定項目詳細說明如下表:

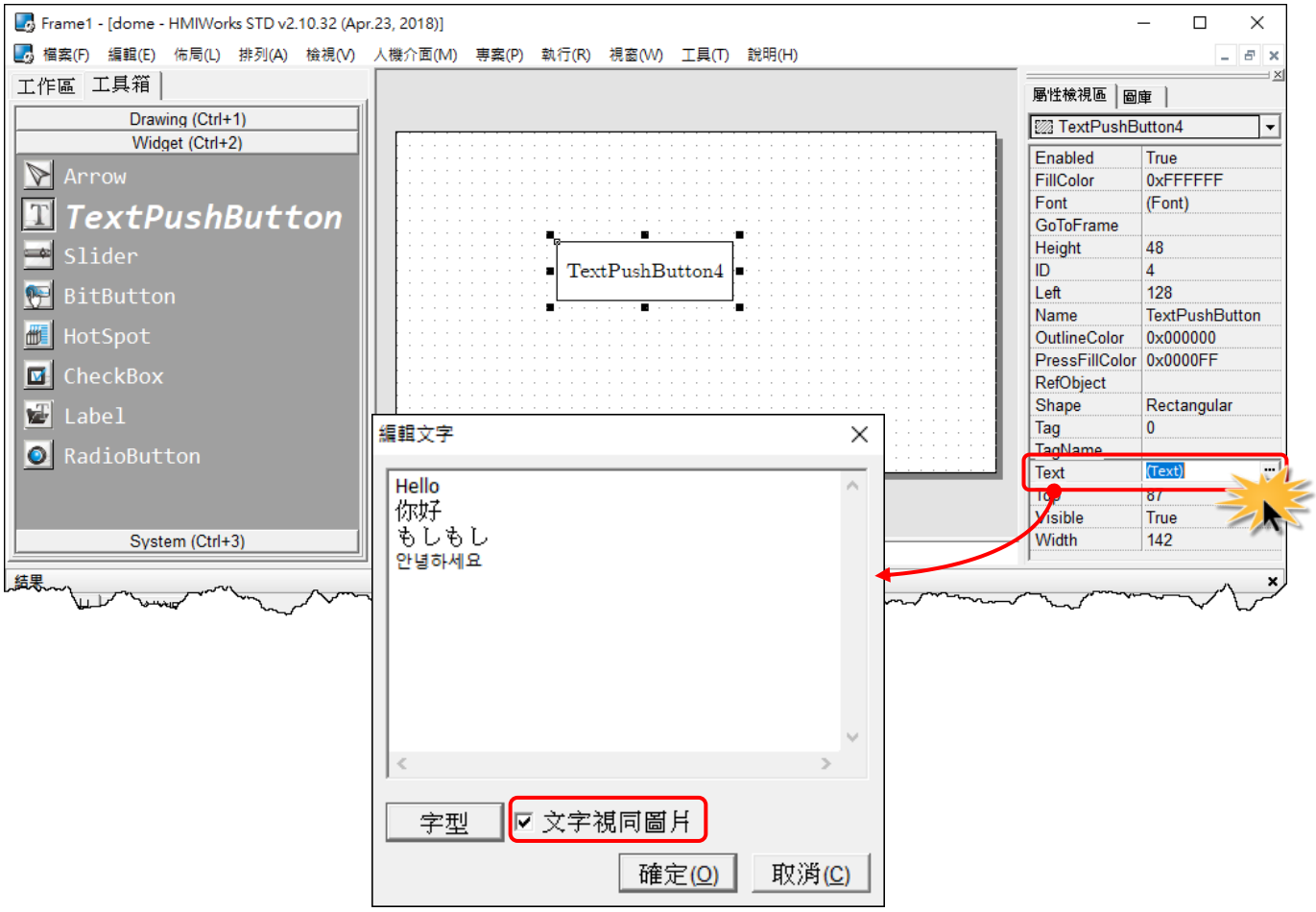
屬性	說明
FillColor (填滿色彩)	用來填滿元件中間的顏色。顏色是用三個位元組 (byte) 以十六進位表示。從最高的到最低的排列，位元組依序代表：藍色、綠色、紅色。
OutlineColor (外框色彩)	包住元件的矩形外框的顏色。
Height (高度)	包住元件的矩形的垂直長度。
Width (寬度)	包住元件的矩形的水平長度。
Left (左頂點)	包住元件的矩形左上角的 x 座標。
Top (上頂點)	包住元件的矩形左上角的 y 座標。
Name (名稱)	名字。
ID (流水號)	流水序號，是用來辨識同類別卻不同實體用的元件或顯示頁。 每個元件或是顯示頁都會有一個唯一的流水號。
Font (字型)	**Text (文字)** **屬性**的字型。

屬性	說明
GoToFrame (轉至頁)	指定顯示頁 (Frame)。換句話說，當按鈕按下時，會跳到由本屬性所指定的那一頁。 **註: "轉至頁 (GoToFrame)"** 優先於 **"點擊事件(OnClick)"**。也就是說，設定了轉至頁，點擊事件屬性就沒有作用了。
RefObject (參考物件)	指定您所要參考的 **"ObjectList (物件清單)"**。 **ObjectList** 是 **"工具箱"** 中的一個元件，用來紀錄從 **"圖庫"** 中選取出來的物件。詳細使用方法，請參閱 第 3.4.17 節 ObjectList。
Tag (標籤)	為了程式方便而存在的變數。例如，可以把每個 **TextPushButton** 的 **Tag (標籤) 屬性**都指定一個唯一的號碼以辨別 **TextPushButton** 彼此之間的不同。 參閱 <<HMIWorks API Reference>> 以使用函式來存取 **Tag (標籤) 屬性**。 **註:** 這個標籤與「標籤名稱」屬性所指的標籤無任何關聯。
TagName (標籤名稱)	與階梯圖設計家 (Ladder Designer) 中產生關連的標籤的名稱。 **註:** 這項屬性只有在**程式類別選「階梯圖 (Ladder)」**時才會生效。
Enabled (啟用)	該元件是否可以使用。
Visible (可見)	該元件是否可以被看見。

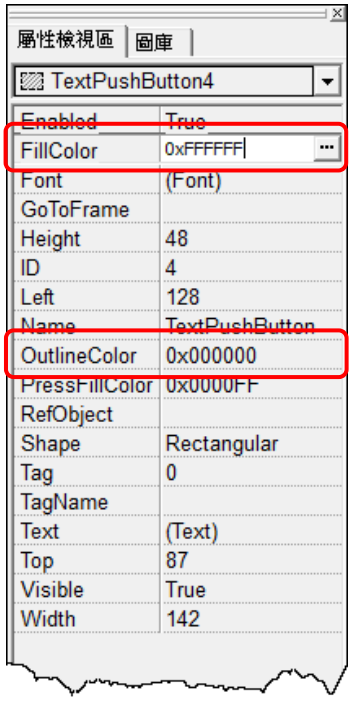
文字轉圖片和多國語言顯示

有三個元件: **TextPushButton (文字彈回式按鈕)**、**CheckBox (核取方塊)** 和 **RadioButton (單選按鈕)** 的 **Text (文字) 屬性**不同其他的元件，可以透過把文字轉成圖片，來作為多國語言顯示。

1. 勾選“**文字視同圖片**”選項，若勾選，則可以有個多字串。
2. 勾選後，其 **Text (文字) 屬性**的每個字串都會被轉成一張圖，而每一張圖則對應到元件的一個狀態。參閱下方的「使用參考物件 (RefObject) 屬性」以獲得更多資訊。

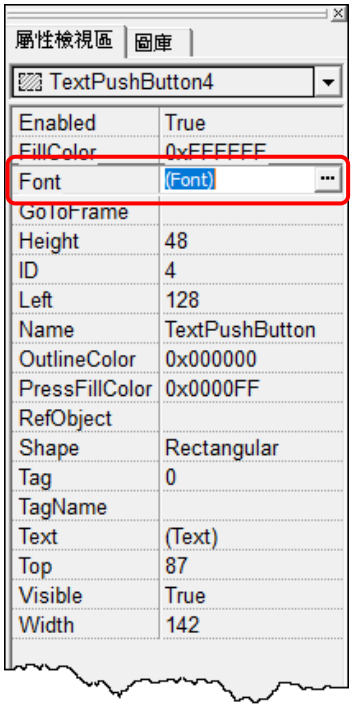


改變色彩

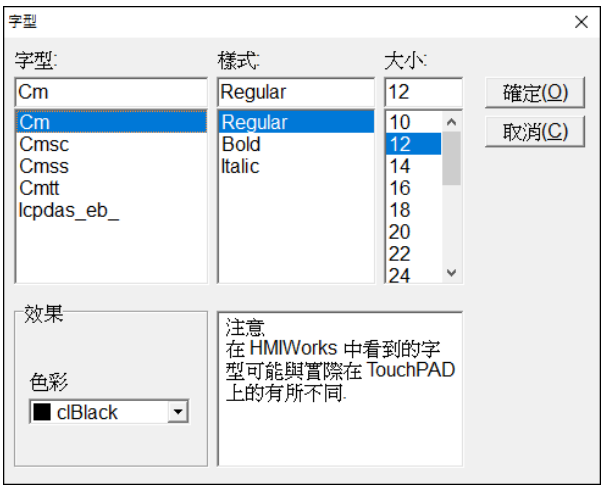


在“**屬性檢視區**”中的 **FillColor (填滿色彩) 屬性**欄中，點選「**...**」鈕，會出現“色彩”視窗，選取顏色然後按 **“確定”** 按鈕。也以相同的方式改變 **OutlineColor (外框色彩)**。

改變文字字型



在“**屬性檢視區**”中的 **Fon t (字型) 屬性**欄中，點選「**...**」鈕，會出現“字型”視窗，選取字型然後按 **“確定”** 按鈕。



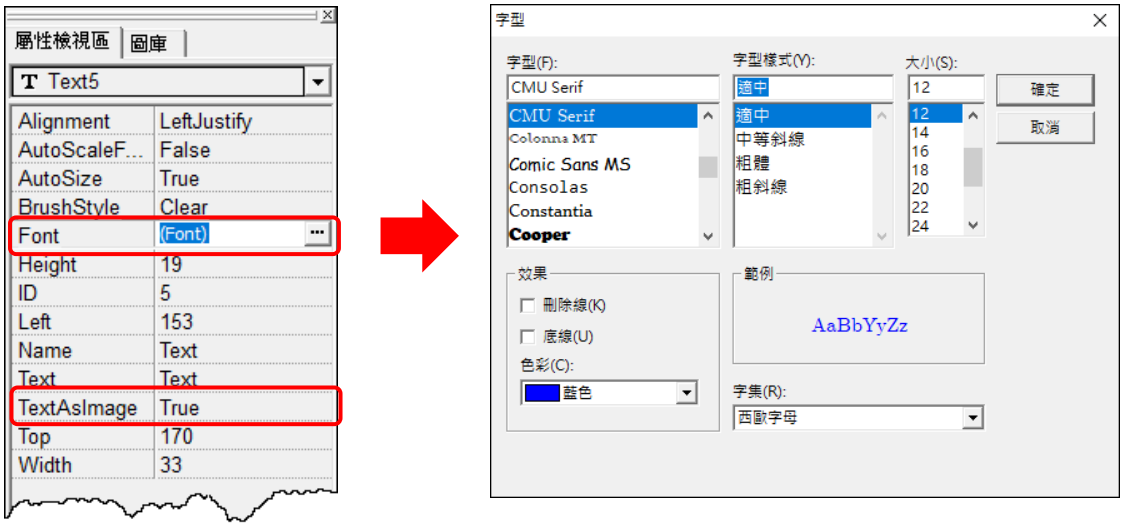
改變文字字型時，有兩種字型對話框：

- **和電腦相同的字型對話框。**
  1. 如果開啟的是這類的字型對話框，當程式下載至 TouchPAD 之後，文字是以圖片的方式存在程式中，所以比較占空間。
  2. 支援這個字型對話框的元件有：**Text (恆定文字框)**、**BitButton (位元圖按鈕)**。

- **TouchPAD 專用字型對話框。**
  1. 文字不是以圖片方式儲存。兩個相同字母，如: A 與 A，只占一個 A 的字型空間。
  2. 支援這個字型對話框的元件有：**Text (恆定文字框)**、 **TextPushButton (文字彈回式按鈕)**、**Slider (滑桿)**、**Checkbox (核取方塊)**、 **Label (文字顯示框)**、**RadioButton (多選按鈕)**。
  3. 支援除了英文以外的語言，可參考:
     FAQ: 如何使用 HMIWorks 內建字型在 TouchPAD 上顯示多語言文字?
     FAQ: 如何在 TouchPAD 上使用 ebFont 顯示多語系文字?

注意:

對 **Text (恆定文字框)** 元件來說，若要支援全部字型功能，須將其 **「TextAsImage (文字視同圖片)」** 屬性設為 **True (真)**。



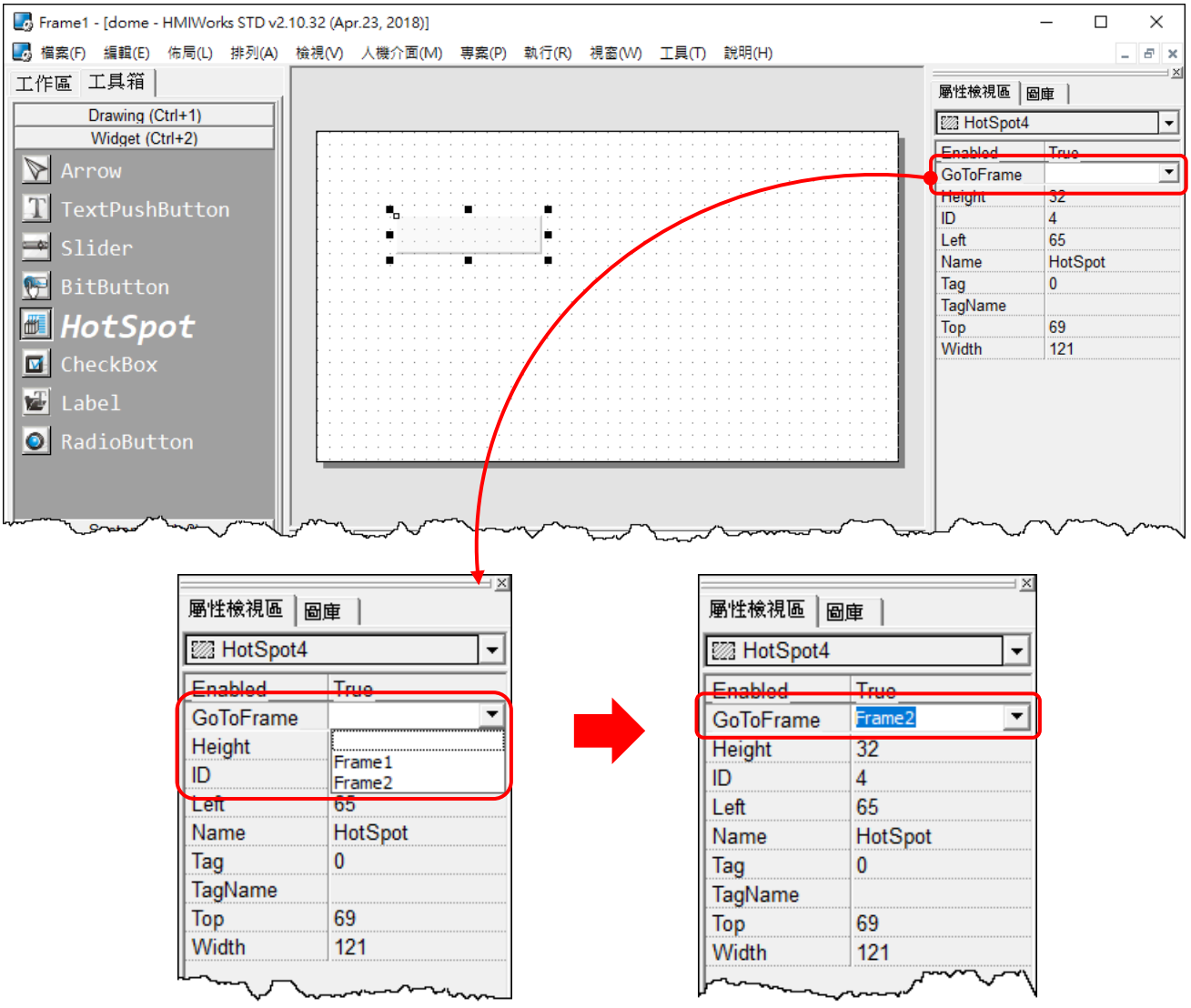
使用 GoToFrame 到另一顯示頁

GoToFrame (轉至頁) 屬性可以看做是 **“前往第幾顯示頁”** 的事件函數。
GoToFrame (轉至頁) 優先於各種事件函數。換句話說，設定了 **GoToFrame (轉至頁)**，事件函數就沒有作用了。

小技巧: 如何新增一個新的顯示頁?

1. 按快捷鍵 **<Ctrl> + <M>** 來開啟 “選擇程式類別” 視窗。
   或是從 **“人機介面(M)”** 功能選單中，點選 **“新增頁(N)”** 項目。
2. 選擇程式類別 **“標準 C 語言”** 或 **“階梯圖”**。

設定 **GoToFrame (轉至頁)** 非常簡單。在 **“屬性檢視區”** 中的 **GoToFrame 屬性**欄中，點選要指向的顯示頁即可。



使用參考物件 (RefObject) 屬性

RefObject 屬性可用來指定物件清單，而該物件清單中的圖片物件就可用來替換 **TextPushButton (文字彈回式按鈕)**、**Slider (滑桿)**、**Checkbox (核取方塊)**、 **Label (文字顯示框)** 和 **RadioButton (單選按鈕)** 的外觀。元件的狀態 (或值) 可作為索引，用來決定在 **RefObject** 屬性指定的物件清單之中的那一張圖片要顯示。元件的狀態可以被人的觸碰改變、被函式 (如: CheckBoxValueSet) 改變和被 TagName (標籤名稱) 屬性所指定的 Tag 改變。假設，有一物件清單叫 OL，且它被指定給元件 (如: 核取方塊) 的參考物件屬性。該物件清單總共有 n 個圖片物件，分別為 OL[0]、OL[1]、…、OL[n-1]。

元件/頁	圖片最大數	元件行為
Frame (顯示頁)	1	OL[0] 是背景圖片。 **註 1:** 在物件清單 OL 儲存超過 1 張圖片，超過的部分是無效的。 **註 2:** 指定預設顯示頁 (Default 屬性為 True) 的參考物件屬性，會自動地指定相同的物件清單給專案中所有的顯示頁。
TextPushButton (文字彈回式按鈕)	理論上無限	OL[0] 是背景圖片。 當文字彈回式按鈕在放開的狀態，它是顯示 OL[0] 的。 而當它在按壓的狀態，第一次按壓時顯示 OL[1]，放開第一次的按壓後再按第二次時是顯示 OL[2]，以此類推。 最後，當文字彈回式按鈕顯示了最後的圖片 OL[n-1] 時，接著在下一次的按壓時，它會重頭重新開始顯示，即 OL[1]，重新整個循環。
Slider (滑桿)	理論上無限	OL[0] 是背景圖片。 滑桿會分成 n-1 段，依滑桿的值 (value) 落在那一段來依序顯示出不同的圖片。舉例如下一頁的詳細說明表。
CheckBox (核取方塊)	理論上無限	每按一次核取方塊則顯示不同的圖片，從 OL[0] ~ OL[n-1]，一個接著一個。一旦顯示到最後一個圖片時 (即 OL[n-1])，則下一次就會從第一個圖片重新重頭開始顯示 (即 OL[0])。
Label (文字顯示框)	1	OL[0] 是背景圖片。 **註:** 在物件清單 OL 儲存超過 1 張圖片，超過的部分是無效的。
RadioButton (單選按鈕)	2	OL[0] 是背景圖片。 OL[1] 是選取的 (Selected) 的圖片。 **註:** 在物件清單 OL 儲存超過 2 張圖片，超過的部分是無效的。

➢ **RefObject** (參考物件) 屬性的 **Slider** (滑桿) 範例。

範例	說明
	在物件清單 OL 中有 6 個圖片。 從左到右分別是 OL[0]、OL[1]、…、OL[5]。
	OL[0] 做為背景圖片。 滑桿被分為 5 部分，每部分 20%，依滑桿的值 (value) 來選擇圖片： 0% ~ 20%: OL[1] 20% ~ 40%: OL[2] 40% ~ 60%: OL[3] 60% ~ 80%: OL[4] 80% ~ 100%: OL[5] 如左欄圖形所示。

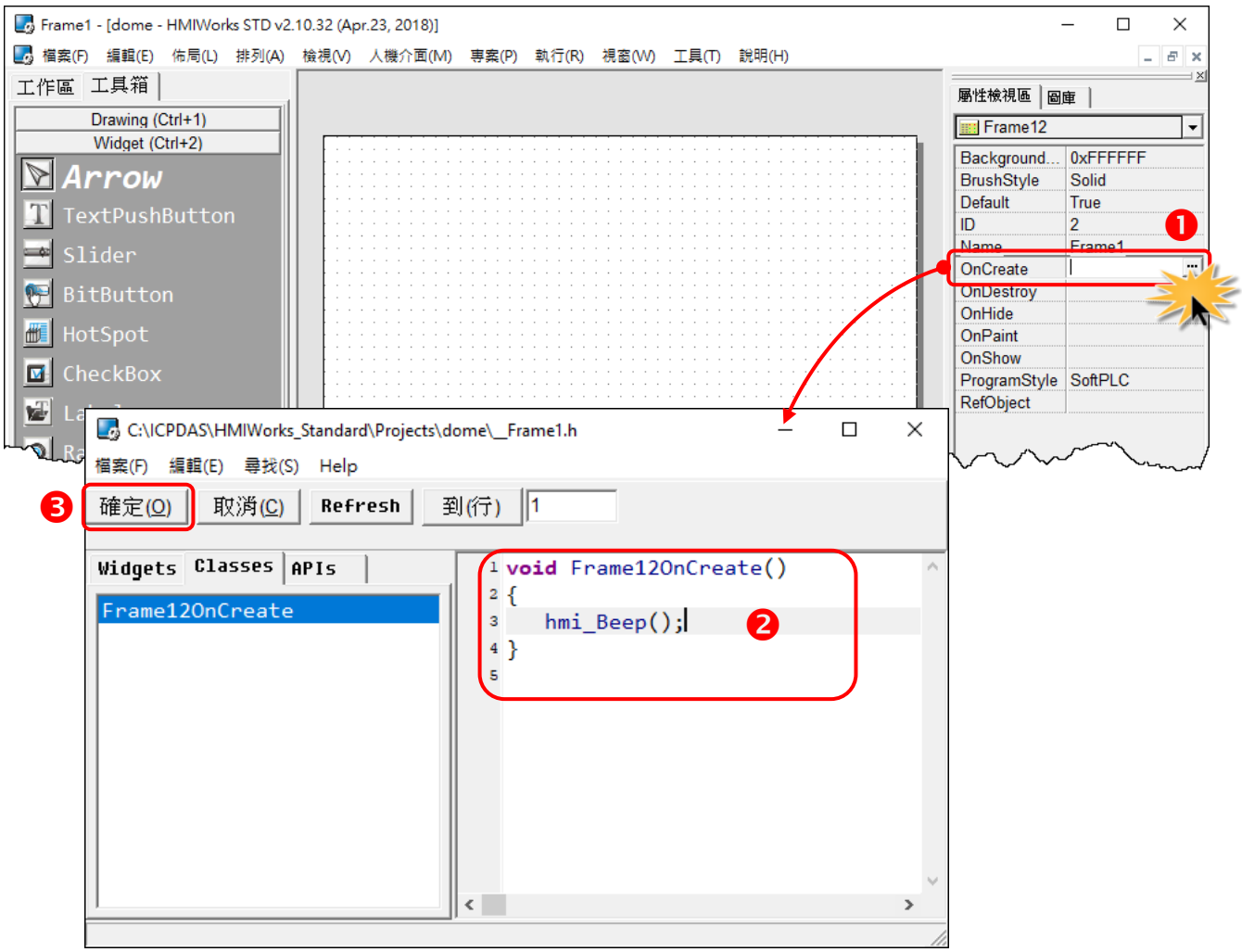
實現事件處理程序

事件處理程序 (Event Handler) 只在該顯示頁的 **“程式類別是標準 C 語言”** 時才有支援，階梯圖 (Ladder) 則沒有。若該元件有兩個以上的事件處理程序，則以滑鼠雙擊會開啟 OnClick 事件處理程序的程式編輯視窗。

元件/頁	支援的事件處理程序
Frame (顯示頁)	OnCreate、OnDestroy OnHide、OnShow OnPaint
TextPushButton (文字彈回式按鈕) **BitButton (位元圖按鈕)** **HotSpot (觸發連結框)**	OnClick、OnRelease
Slider (滑桿)	OnSliderChange
CheckBox (核取方塊)	OnChange
Timer (計時器)	OnExecute
PaintBox (繪圖框)	OnPaint
RadioButton (單選按鈕)	OnRadioChange

➢ 以顯示頁的 **OnCreate 事件**處理程序為例：

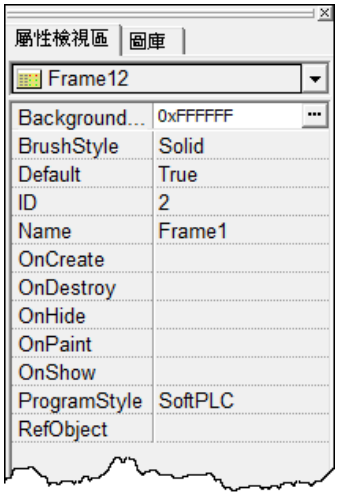
1. 在 **Frame (顯示頁)** 的 **“屬性檢視區”** 中，點選 **OnCreate 屬性**，再按下**「…」**按鈕以開啟程式編輯視窗。
2. 這裡以呼叫 **hmi_Beep()** 這一函式來發出一嗶聲為例。
3. 按下 **“確定(O)”** 來儲存檔案然後離開。



3.4.2 Frame

Frame (顯示頁) 的特有屬性

在顯示頁上用滑鼠點一下，可以在**“屬性檢視區”**檢視顯示頁的屬性。



屬性	說明
BackgroundColor (背景顏色)	背景的顏色。 顏色是用三個位元組 (byte) 以十六進位表示。從最高的到最低的排列，位元組依序代表：藍色、綠色、紅色。
BrushStyle (筆觸)	填滿 (Solid) 或無填滿 (Clear)。 如果**筆觸設為填滿**的話，背景顏色屬性就會生效。但是這在某些情況可能會造成 **“螢幕閃爍”**，尤其是背景顏色和載入圖片的顏色差距過大的時候。 如果**筆觸設為無填滿**的話，背景顏色屬性就會失效，螢幕就不會閃爍。
Default (預設)	是否為 TouchPAD 一開機預設的頁面。
ProgramStyle (程式類別)	標準 C 語言或階梯圖。

Frame (顯示頁) 的事件處理程序

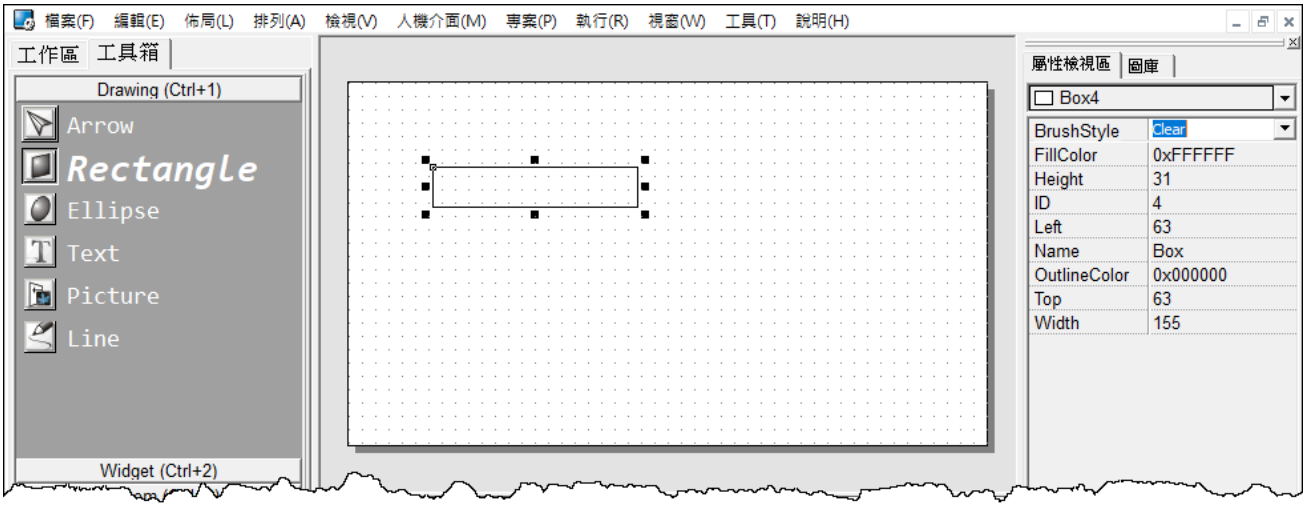
舉例說明，當切到顯示頁 Frame1 時，

- **OnCreate**: 先執行此一 OnCreate 函式。
- **OnShow**: OnCreate 後會把該頁 (Frame1) 所用到的 widget (控制項) 加入 (含 Frame1)，接著就執行 OnShow。(即 OnShow 有該 Frame1 的 widget 可以使用)
- **OnPaint**: 當該頁需要重繪 (paint) 時，如果是在一轉到此頁時，則 OnPaint 在 OnShow 之後執行。

當切頁到其他頁時，即離開 Frame1 時，

- **OnHide**: 先執行此一函式。
- **OnDestroy**: OnHide 後會把該頁 (Frame1) 所用到的 widget (控制項) 移除 (含 Frame1)，接著就執行 OnDestroy。

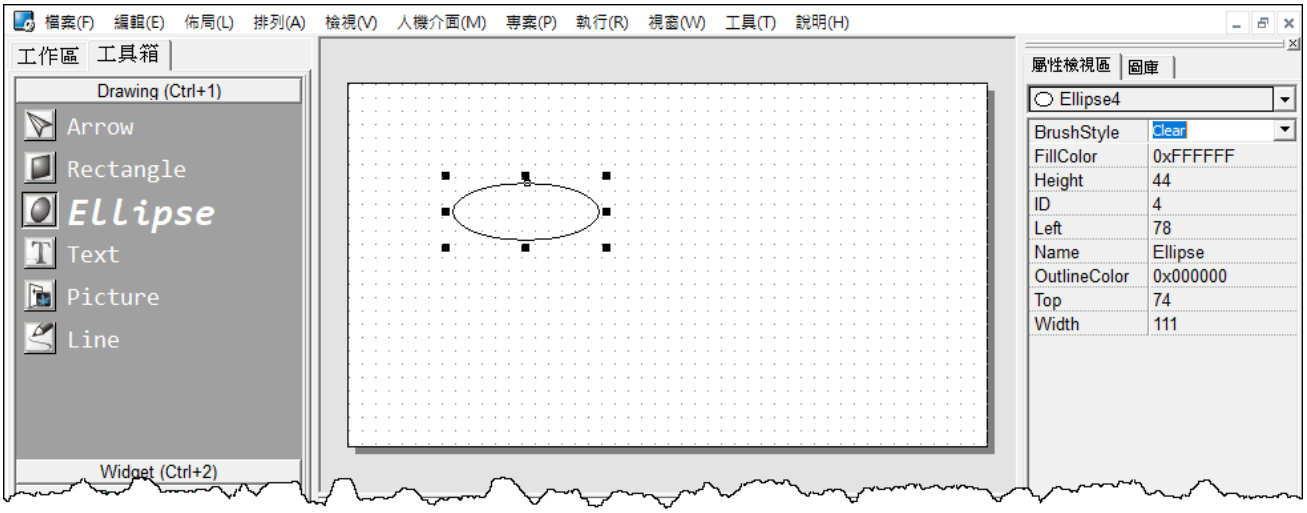
3.4.3 Rectangle



Rectangle (矩形) 的特有屬性:

屬性	說明
BrushStyle (筆觸)	矩形中間填滿 (Solid) 或無填滿 (Clear)。

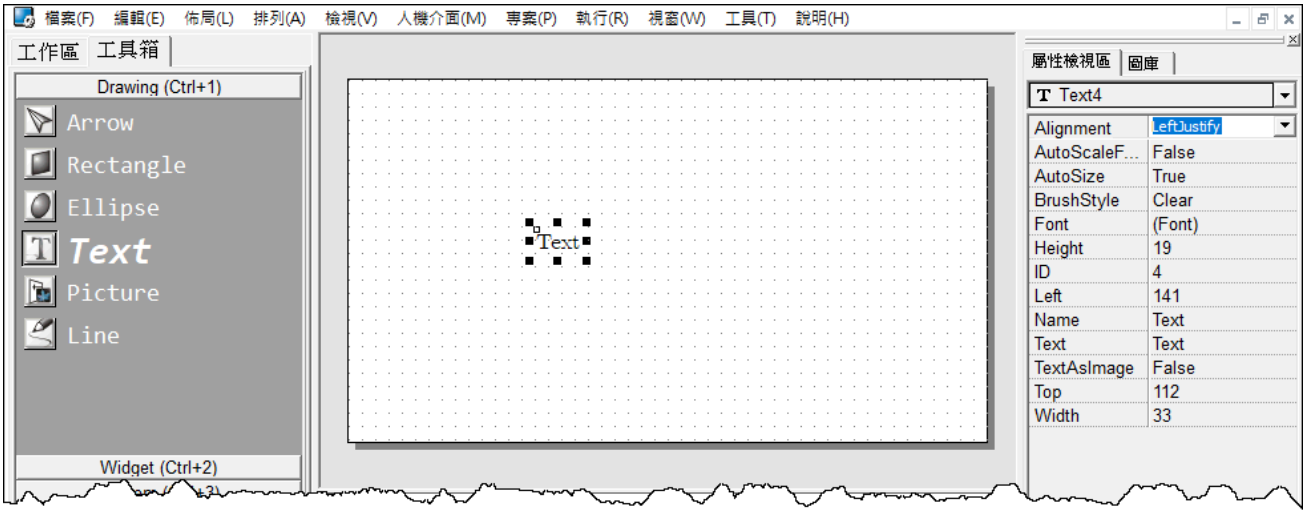
3.4.4 Ellipse



Ellipse(橢圓) 的特有屬性:

屬性	說明
BrushStyle (筆觸)	橢圓中間填滿 (Solid) 或無填滿 (Clear)。

3.4.5 Text



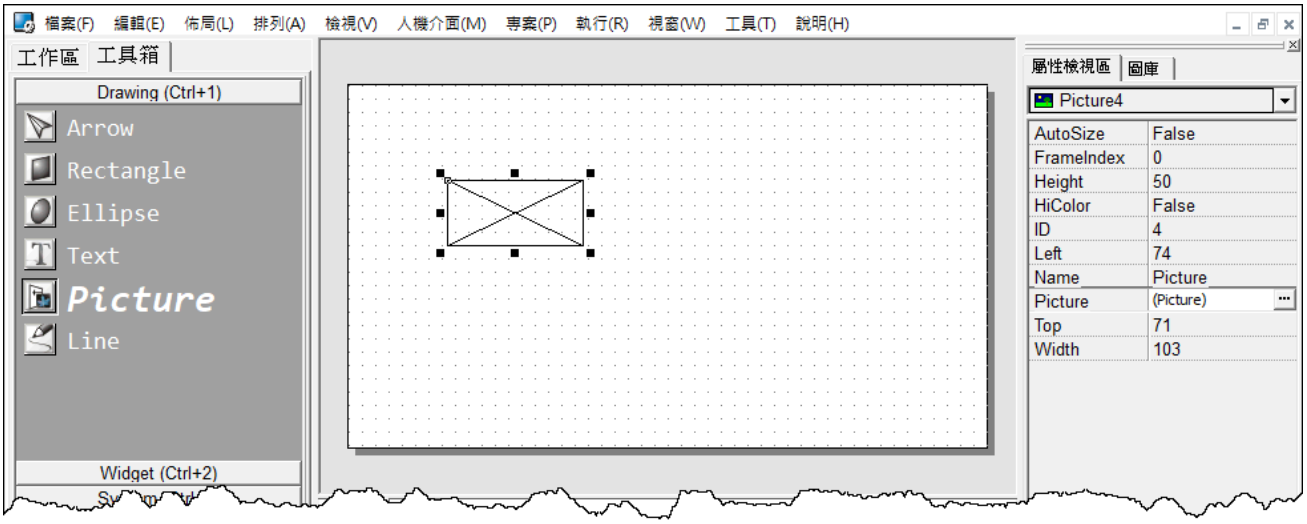
在顯示頁上放置文字的另一種方法

更簡單地，把複製到剪貼簿 (clipboard) 的文字，在 HMIWorks 裡直接貼上，HMIWorks 就會直接產生一個已經載入剪貼簿中之文字的恆定文字框。

Text (恆定文字框) 的特有屬性:

屬性	說明
Alignment (對齊)	這項屬性決定文字要置於包圍恆定文字框之矩形的那個位置。共有三個選項：靠左(LeftJustify)、靠右(RightJustify)、置中(Center)。 **註:** 這項屬性只有在 **“AutoSize”** 設為 **True (真)** 時才會有作用。
AutoScaleFontSize (自動調整字型大小)	依據包圍恆定文字框之矩形的垂直高度，自動調整字型的大小，以便字體的高度佔滿該矩形的垂直高度。 **註:** 這項屬性只有在 **“AutoSize”** 設為 **True (真)** 時才會有作用。
AutoSize (自動調整框的大小)	True (真)或 False (假)。 這項屬性用來決定是否自動調整恆定文字框之矩形的大小，以使該框能夠包含整個文字。
BrushStyle (筆觸)	中間填滿 (Solid) 或無填滿 (Clear)。
TextAsImage (文字視同圖片)	True (真)或 False (假)。 恆定文字框中的文字在儲存時，是否要當成圖片來儲存。當然以圖片形式儲存會佔較多的記憶體空間且下載時間也較久。

3.4.6 Picture



Picture (圖片框) 的特有屬性:

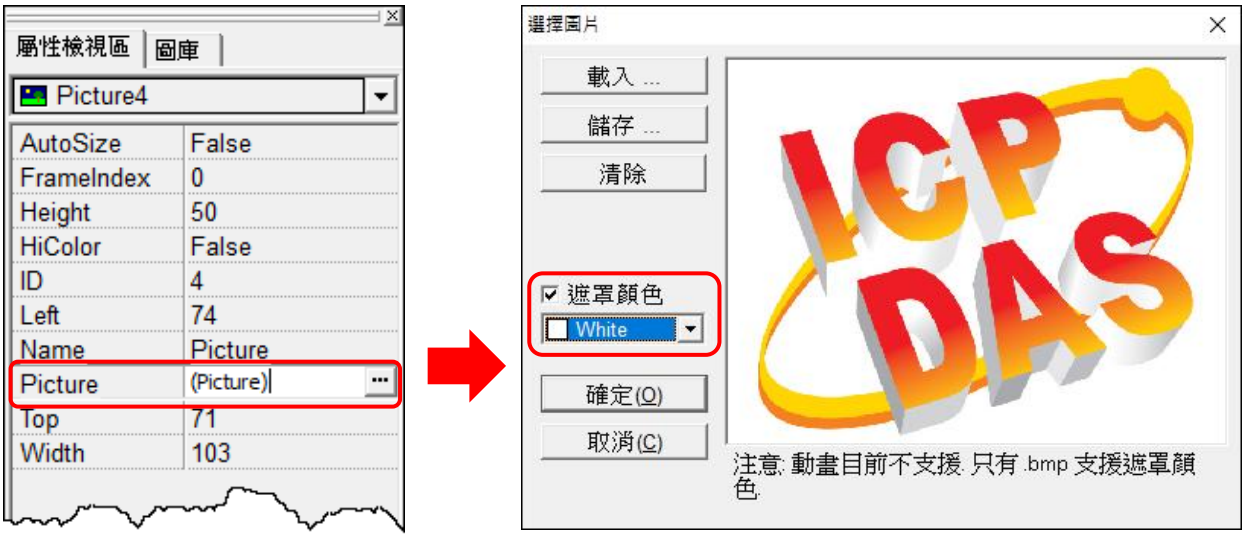
屬性	說明
AutoSize **(自動調整圖片大小)**	True (真) 或 False (假)。 這項屬性用來決定是否自動調整圖片的大小至原來的大小。
FrameIndex **(顯示頁序號)**	忽略 (保留未來使用)。
HiColor **(高解析色彩)**	True (真)或 False (假)。 **True (真):** 載入的圖片是用 16 位元的色彩儲存，圖片的解析度較佳，但需要較大的儲存空間。 **False (假):** 載入的圖片是用 8 位元的色彩儲存，圖片的解析度較差，但儲存空間較小。(預設設定 8 位元儲存)
Picture (圖片)	載入的圖片。

載入一張圖片到顯示頁

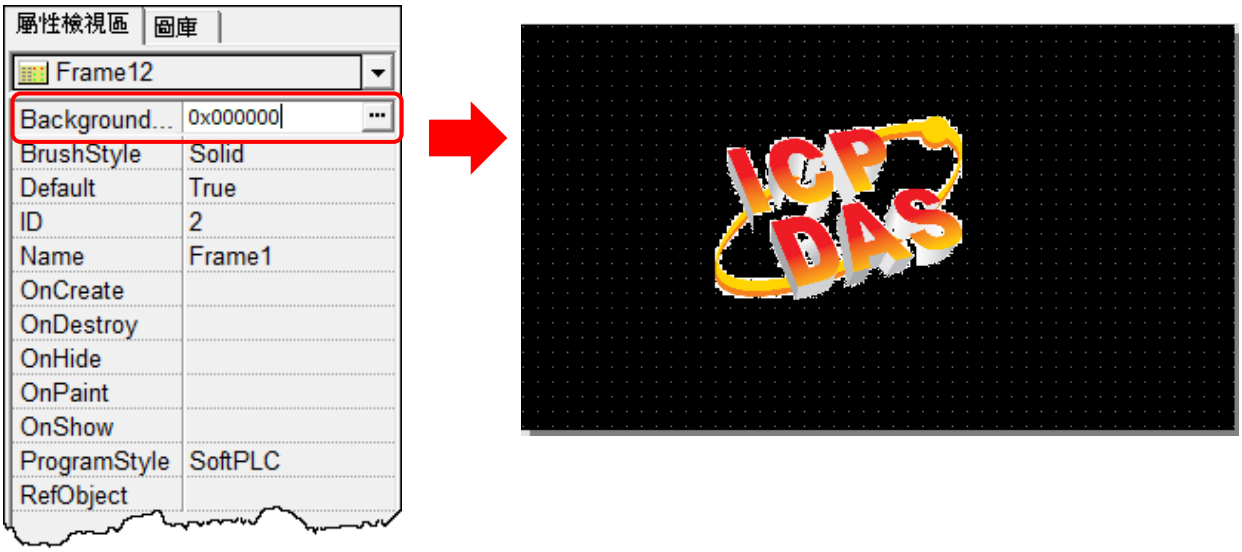
載入一張圖片至 Frame (顯示頁)，有下面兩種方式：

1. 簡單地，把複製到剪貼簿 (clipboard) 的圖片，在 HMIWorks 裡直接貼上，HMIWorks 就會直接產生一個已經載入剪貼簿中之圖片的圖片框。

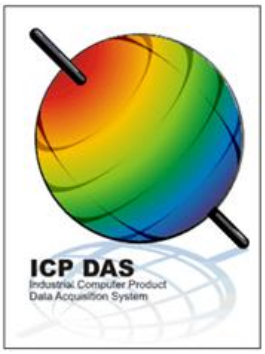
2. 在 “**屬性檢視區**” 中，點選 **Picture 屬性**欄中「**...**」按鈕來開啟 “**選擇圖片**” 視窗，點選 “載入”按鈕來載入圖片，然後按下 **“確定(O)”**。在“**選擇圖片**” 視窗中有個 **“遮罩顏色”** 選項，可以用來**實現顏色的透明**。**注意:** 目前只有 **.bmp** 檔案支援 **“遮罩顏色”** 選項。



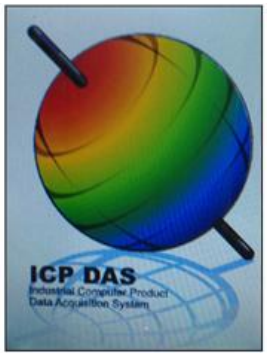
如上所示，勾選 **“遮罩顏色”** 選項後，選擇白色把白色擋掉，亦即圖片中白色的部分變透明了。把顯示頁 (Frame) 的 **BackgroundColor (背景顏色) 屬性**設為黑色，可以更明顯地看出效果。



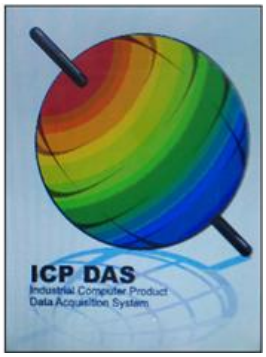
程式大小與解析度的取捨



在 **HMIWorks** 中



HiColor = True (16 位元色彩) TouchPAD 的螢幕上 (大小: 189KB)



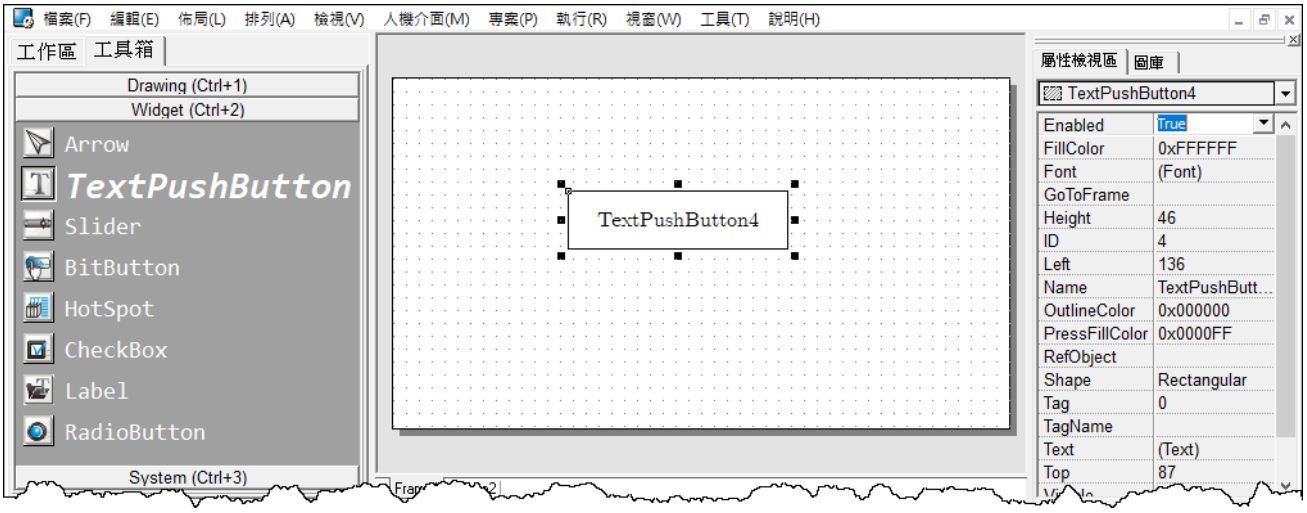
HiColor = False (8 位元色彩) TouchPAD 的螢幕上 (大小: 69KB)

上圖是 **HiColor = True** 和 **HiColor = False** 的比較圖。
最左邊的圖是原來在 HMIWorks 中的情形。右邊兩張圖是實際從 TouchPAD 上拍出來的圖。一張是 HiColor 設為 True 的圖，另一張是設為 False 的圖。

如上圖所示，將 HiColor 設為 False 會讓圖呈現漸層效果，但是若將 HiColor 設為 True 則不會。這是因為 8 位元的圖沒有足夠的顏色來顯示所有色彩 (總共 256 色而已)，所以相近的顏色被迫用相同的顏色來代表，這就會使圖有漸層的效果。

雖然，為了避免漸層效果，可以設 HiColor 為 True，但是這需要較大的儲存空間。以上圖為例，設 HiColor 為 True 需要 189 KB 的記憶體，但若是設 HiColor 為 False 則只需要 69 KB。

3.4.7 TextPushButton



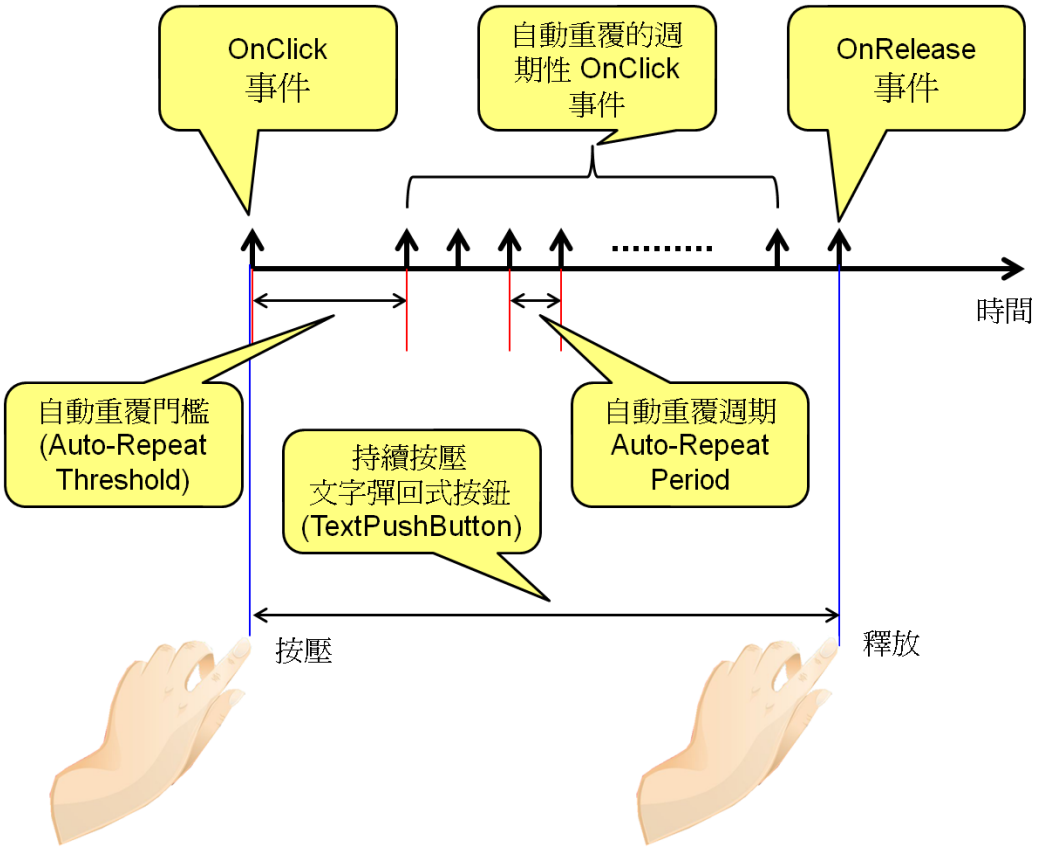
什麼是 TextPushButton (文字彈回式按鈕)?

文字彈回式按鈕是一種上面有文字，且按下鬆開後，狀態會彈回原狀的按鈕。

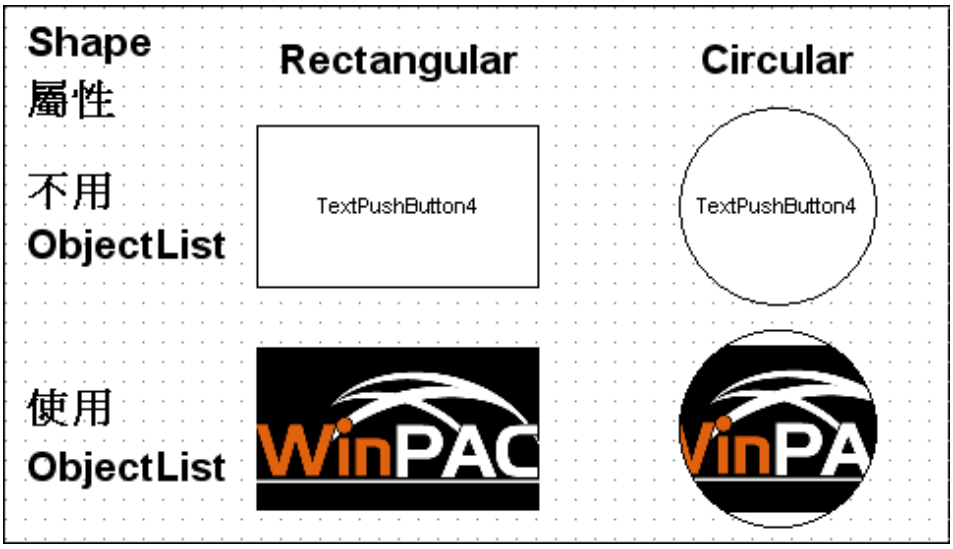
TextPushButton (文字彈回式按鈕)的特有屬性:

屬性	說明
AutoRepeatPeriod (自動重復週期)	持續按壓文字彈回式按鈕的情形下，重覆地觸發 **OnClick 事件**的週期。這個屬性只有在**程式類別為 "標準 C 語言"** 時才有。(單位: ms)
AutoRepeatThreshold (自動重覆門檻)	按壓文字彈回式按鈕而觸發一個 **OnClick 事件**後，持續按壓而觸發第一個週期性自動重**覆 OnClick 事件** (不是第一個觸發的 **OnClick 事件**) 所需要時間。這個屬性只有在程式類別為 **"標準 C 語言"** 時才有。(單位: ms)
PressFillColor (按住時的填滿色彩)	按鈕被按住時，用來填滿文字彈回式按鈕外框包含的矩形的顏色。顏色是用三個位元組 (byte) 以十六進位表示。從最高的到最低的排列，位元組依序代表：藍色、綠色、紅色。
Shape (外形)	文字彈回式按鈕的外形，有兩個選擇：圓形 (Circular) 或是矩形 (Rectangular)

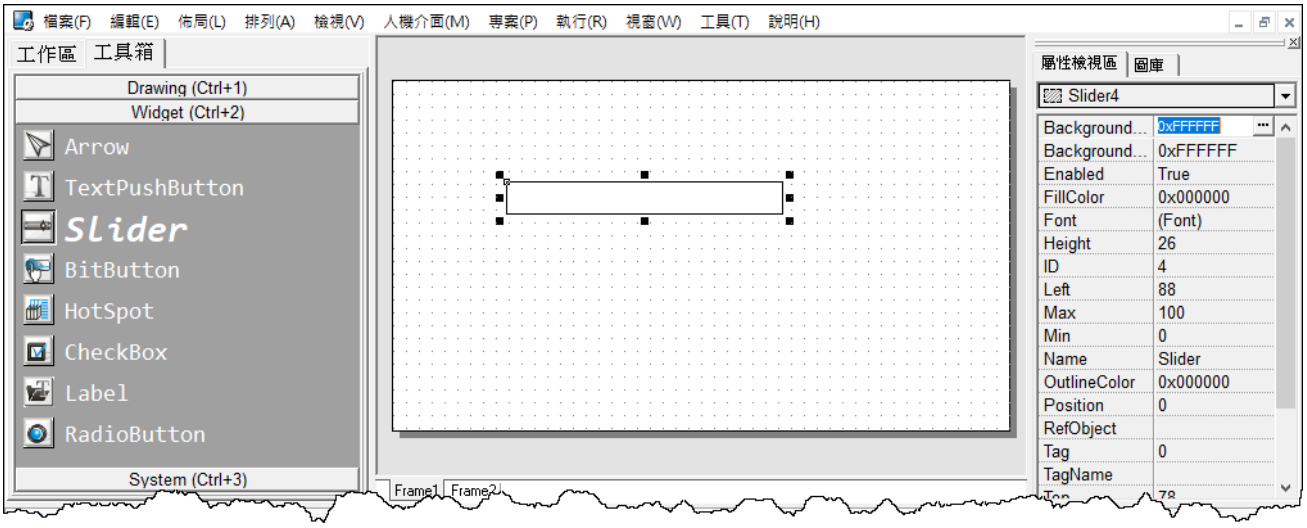
觸發事件



Shape (外形) 屬性舉例



3.4.8 Slider



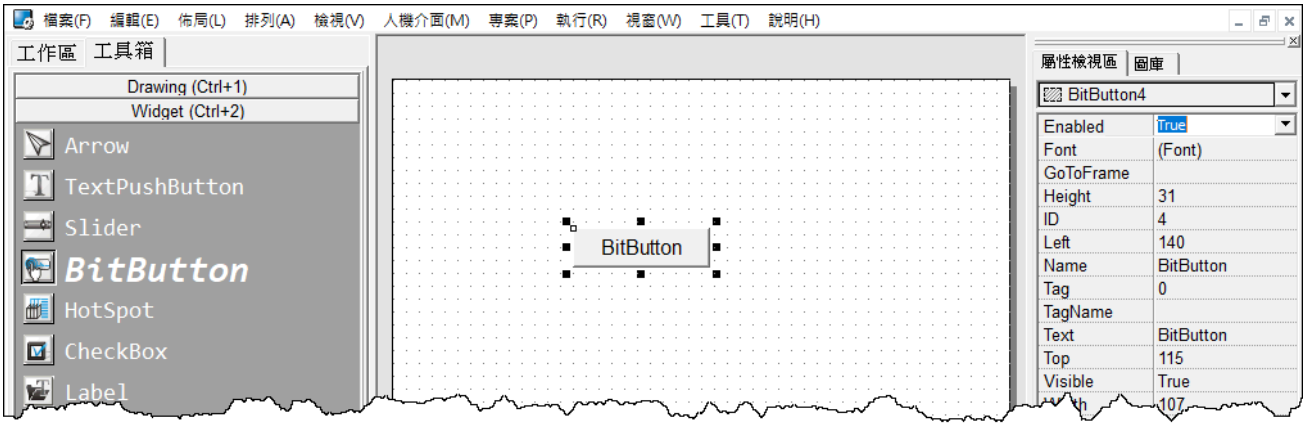
什麼是 Slider (滑桿)?

一個滑動拉桿可以決定大小或者是百分比的工具。通常用於調整音量大小等。

Slider (滑桿) 的特有屬性:

屬性	說明
BackgroundFillColor (背景填滿色彩)	在背景區，用來填滿滑桿外框所包含之矩形的顏色。 顏色是用三個位元組 (byte) 以十六進位表示。從最高的到最低的排列，位元組依序代表: 藍色、綠色、紅色。
BackgroundTextColor (背景文字色彩)	文字在背景區的顏色。 顏色是用三個位元組 (byte) 以十六進位表示。從最高的到最低的排列，位元組依序代表: 藍色、綠色、紅色。
Max (最大值)	Position (位置) 屬性的最大值。
Min (最小值)	Position (位置) 屬性的最小值。
Position (位置)	滑桿的位置 (在最大值和最小值之間)
Vertical (方向)	滑桿的方向，垂直或是水平。 **True (真):** 垂直。 **False (假):** 水平。

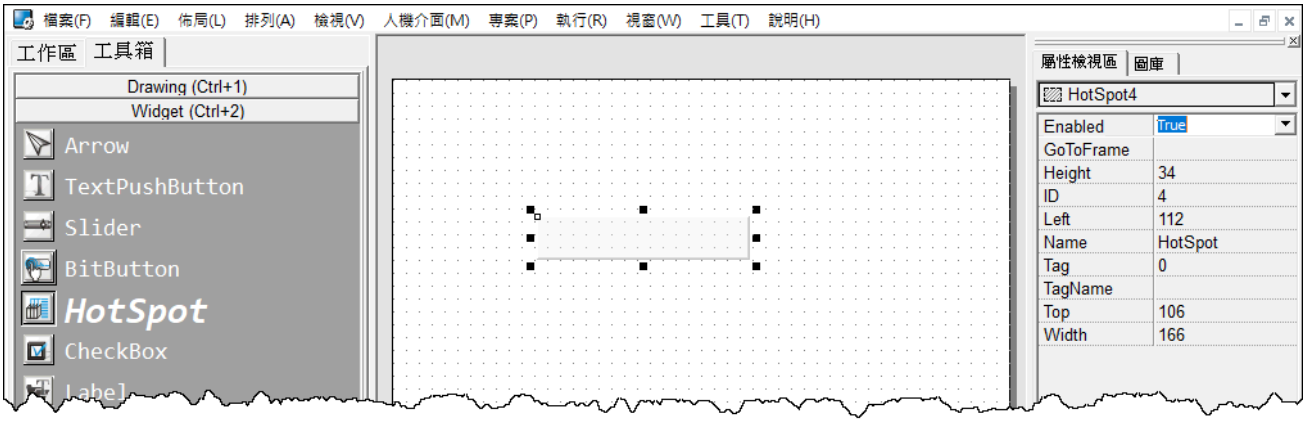
3.4.10 BitButton



什麼是 BitButton (位元圖按鈕)?

按下觸發後會自動彈回原狀的按鈕。另外，與 **TextPushButton (文字彈回式按鈕)** 相比，位元圖按鈕有 3D 的外觀。不按時用一張圖來表示，按下時又用另一張圖來表示，所以可以做出 3D 的效果，但也因為是用圖片表示，所以需要較大的儲存空間，也耗費較久時間來下載。

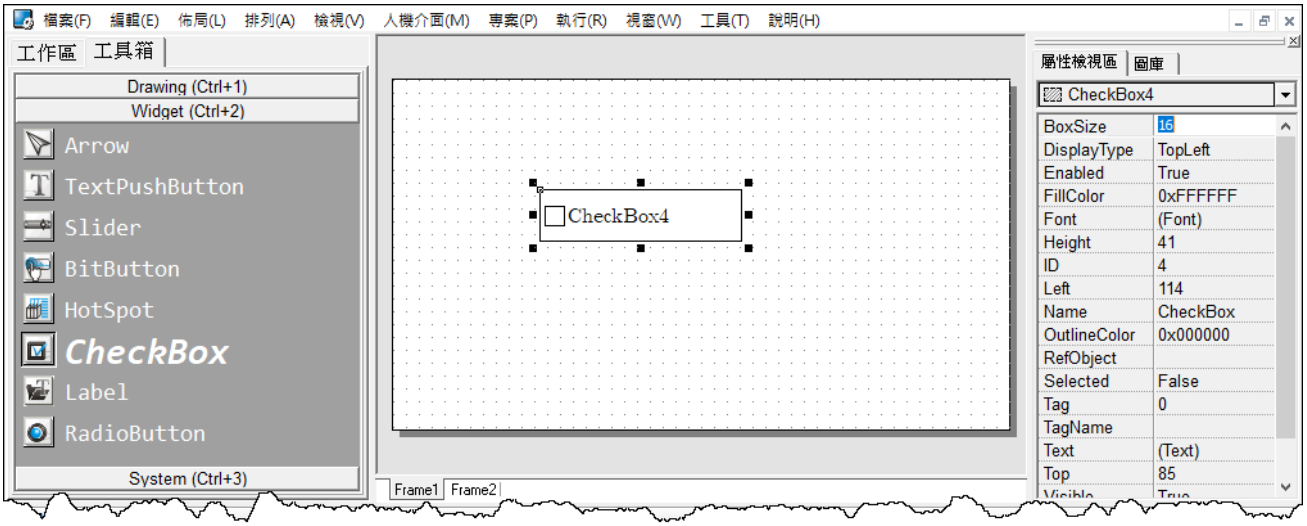
3.4.11 HotSpot



什麼是 HotSpot (觸發連結框)?

一個區域，在該區域觸碰可以產生一個 **“點擊事件 (On-Click Event)”**。通常將觸發連結框放置於圖片上，因為下載到 TouchPAD 後，觸發連結框是隱形的，所以看起來就好像是碰觸該圖片，觸發了一個點擊事件一樣。

3.4.12 CheckBox



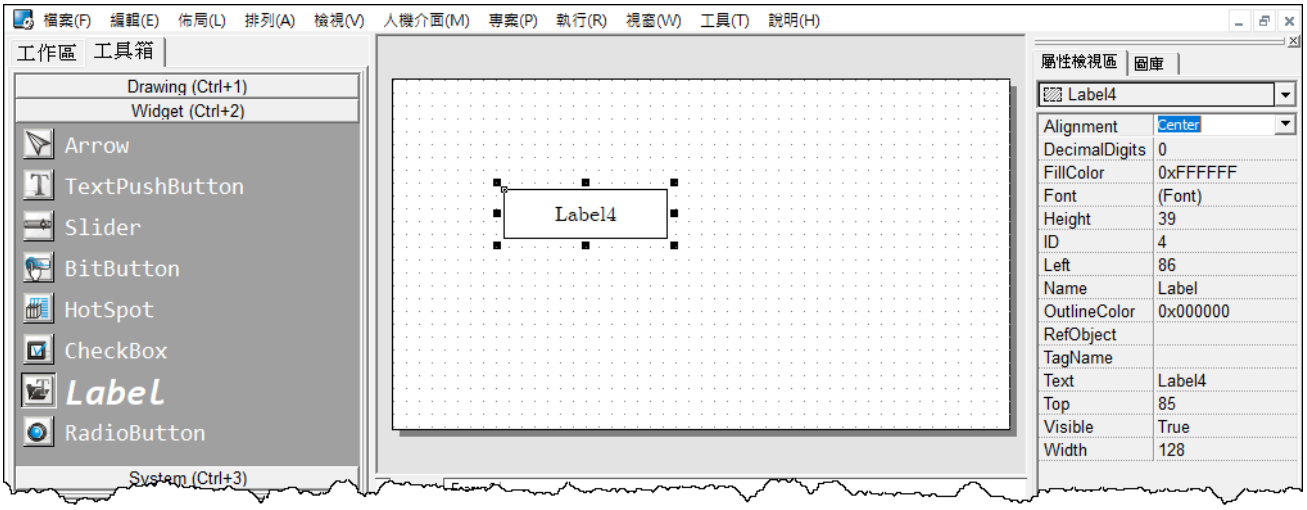
什麼是 CheckBox (核取方塊)?

一個提供是非選擇的工具。

CheckBox (核取方塊) 的特有屬性:

屬性	說明
BoxSize (方塊尺寸)	核取方塊中，可以打勾或是其他表示方式的小正方形，該正方形的大小。
DisplayType (顯示類別)	如何顯示 **RefObject (參考物件)** 屬性所指定的 **ObjectList** (物件清單)中的圖片。
Selected (選取)	True (真)或 False (假)。核取方塊有沒有被選取。

3.4.13 Label



什麼是 Label (文字顯示框)?

一個執行時可以更新文字的顯示文字區域。

Label (文字顯示框) 的特有屬性:

屬性	說明
Alignment (對齊)	這項屬性決定文字要置於 **Lable 文字顯示框**的那個位置。 共有三個選項：LeftJustify (靠左)、RightJustify (靠右)、Center (置中)。
DecimalDigits (小數位數)	由本屬性來決定 **Label (文字顯示框)** 所關連標籤的值帶有 (要顯示) 幾位小數。 例: 標籤數值為 12345，而 **DecimalDigits 屬性**設為 2 (帶 2 位小數)，則 Label 顯示 123.45。 **註:** 這項屬性只有在程式類別選 **"階梯圖(Ladder)"** 時才會生效。

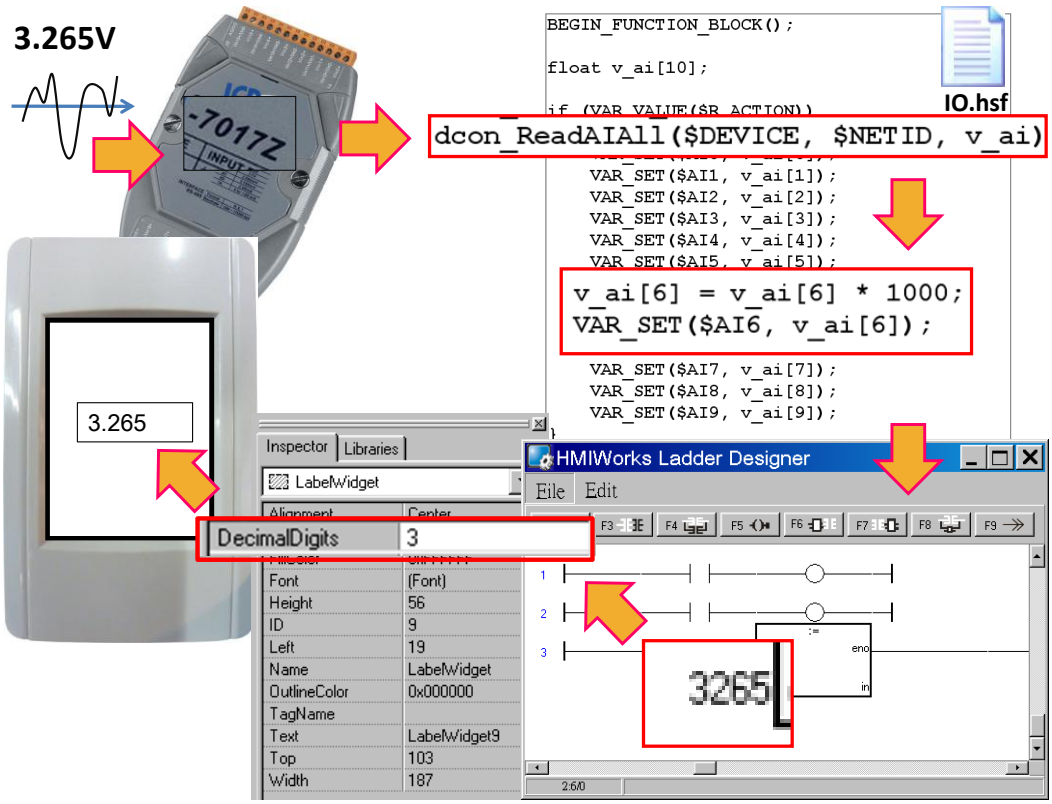
使用階梯圖設計家時，如何顯示小數

所有在**階梯圖設計家 (Ladder Desinger)** 中的數字都是整數，不接受小數。但是在一些情況中，使用者可能需要計算或是顯示小數，所以在下一頁將提供一個方法克服這個問題。

這裡以 I-7017Z 為例。假設我們用 I-7017Z 來讀回遠端設備的電壓值 3.265 V，並且使用 HMIWorks 的**階梯圖設計家**來處理。因為**階梯圖設計家**無法處理小數，所以如果想要在**階梯圖設計家**直接處理 I-7017Z 的電壓值的話，就必需要執行下列兩個步驟：

1. 將 **DecimalDigits (小數位數) 屬性**設為所要顯示於文字顯示框上的小數位數。例如，我們將 **DecimalDigits** 設定為 **3**。
2. 修改 I/O 模組的 **IO.hsf** 檔。讓讀回的 AI (類比輸入) 值乘上 10 的 n 次方，其中 n 是 **DecimalDigits (小數位數) 屬性**的值。使用者可以在下列路徑找到 I/O 模組的 **IO.hsf** 檔: **“[HMIWorks 安裝路徑]\ bin\Modules\”**。
   例如，I-7017Z 的 **IO.hsf** 位於：
   “C:\ICPDAS\HMIWorks_Standard\bin\Modules\I-7000\I-7017Z”，其中 **“C:\ICPDAS\HMIWorks_Standard\”** 是 HMIWorks 的安裝路徑。
   然後我們修改 **IO.hsf**，加入 **v_ai[6] = v_ai[6] * 1000**
   這是假設我們從第 6 個通道讀回 AI 值。

如圖所示，在**階梯圖設計家**的標籤 **“$AI6”** 是實際值的 1000 倍，然後將 **DecimalDigits (小數位數) 屬性**設為 3，就可以在 TouchPAD 上顯示正確的值 3.265 了。



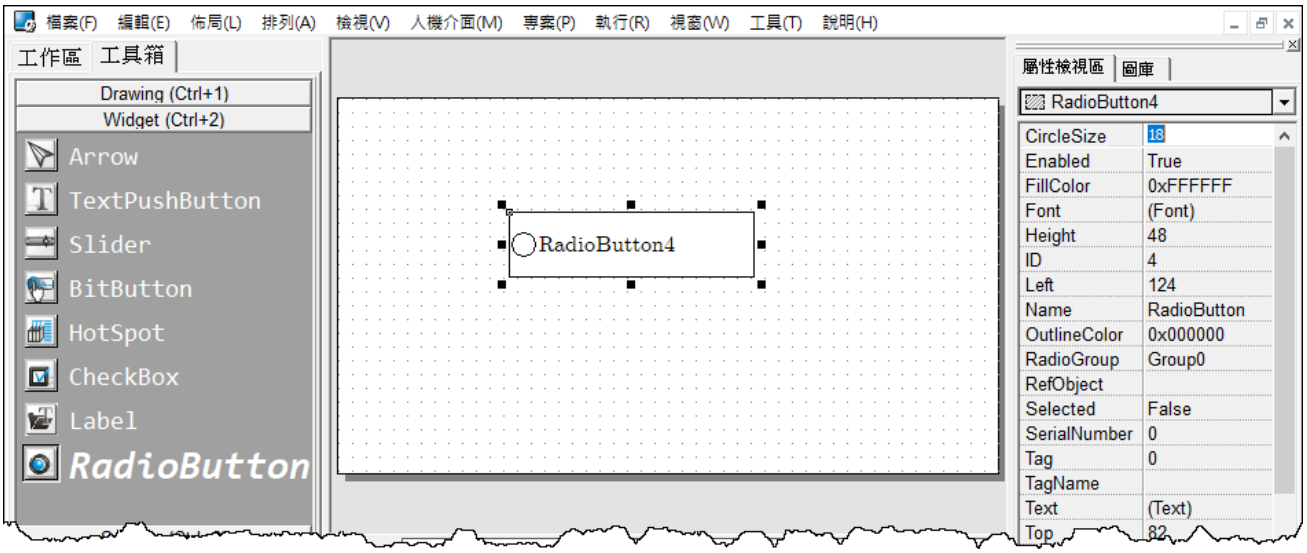
用 C 語言顯示小數

在 **“標準 C 語言”** 的顯示頁中，要顯示小數可能會一點困難，因為 HMIWorks 並**不支援** “**sprint”**這一個函式。

因此，我們用 **“usprintf** (或 **usnprintf)”** 來替代 **“sprintf”**，但是 **“usprintf”** 卻無法如 **“sprintf”**支援參數 **“%f”**。故為了顯示浮點數，我們提供了一個新的函式 **“FloatToStr”** 來把浮點數轉成字串，如下例所示。

```
void TextPushButton4OnClick(tWidget *pWidget)
{
    float ret_sin;
    float angle = 1.57;
    static char str_sin[20];

    // sin
    ret_sin = sin(angle);

    // int FloatToStr(char *buf, float fVal, int precision);
    // the precision determine the number of the digits after the decimal point
    FloatToStr(str_sin, ret_sin, 3);
    LabelTextSet(&Label5, str_sin);    // The result is 1.000
}
```

3.4.14 RadioButton



什麼是 RadioButton (單選按鈕)?

RadioButton (單選按鈕) 是用來做「**多選一**」的選擇時使用的。也就是說，在一個單選按鈕的群組 (稱為 “單選按鈕群組”) 中，只有一個單選按鈕處於選取的狀態，其餘的不被選取。

RadioButton (單選按鈕) 的特有屬性:

屬性	說明
CircleSize (圓圈尺寸)	單鈕按鈕中，用來選擇的圓圈的大小。
RadioGroup (單選按鈕群組)	一個包含多個單選按鈕的群組，其中只有一個單選按鈕可以被選取。 在每個顯示頁 (Frame) 最多可以有八個單選按鈕群組 (Group0 ~ Group7)。
Selected (選取)	True (真)或 False (假)。單選按鈕有沒有被選取。
SerialNumber (序號)	序號是從 0 開始的唯一數字，用來辨認在同一群組中的不同單選按鈕。 序號屬性只提供使用者知道存取那一個單選按鈕，例如: 使用 **“RadioButtonGroupValueSet”** 函式時。 **註 1**: 序號是自動指定的，是唯讀不可更改的。 **註 2**: 當一個單選按鈕的 **TagName (標籤名稱) 屬性**被分配了一個標籤時，此時與該單選按鈕在同一群組中的其他單選按鈕同時會被指定到相同的標籤。若標籤的值等於該群組中某單選按鈕的序號時，則此單選按鈕就是被選取。(通常一個標籤是用來代表遠方的一個 I/O 點。)

TagName (標籤名稱) 屬性有不同的行為

和其他控制項不同的是，在同一單選按鈕群組中所有單選按鈕有相同的標籤名稱屬性。這是因為單選按鈕是用來提供「**多選一**」的選擇，因此標籤名稱屬性就必須相同。

例如: 有三個單選按鈕 0、1、2，其中 0、1、2 是這三個單選按鈕的序號。且這三個單選按鈕都被指定到相同的單選按鈕群組 (Group0)。現在，如果我們指定一個 AI 標籤給這三個單選按鈕，稱為 Dev_AI0，則我們有下列的行為：

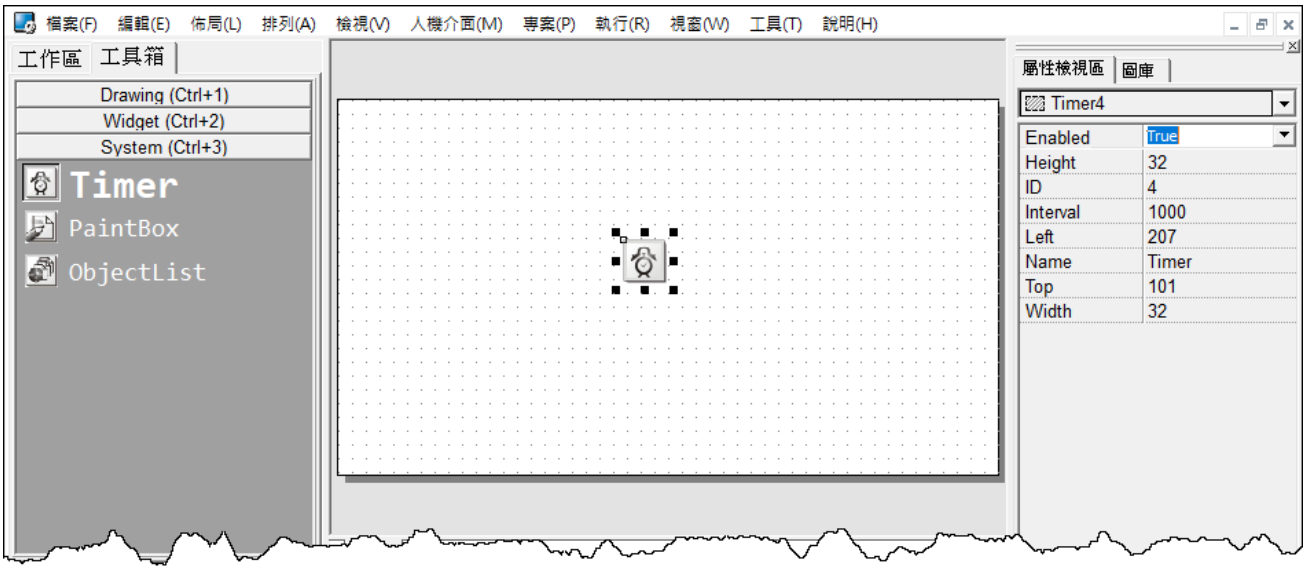
1. 當 **Dev_AI0 = 0**，只有序號 0 的單選按鈕被選擇 (Selected = True)。
   (同時其他兩個單選按鈕沒有被選擇。)
2. 當 **Dev_AI0 = 1**，只有序號 1 的單選按鈕被選擇。
3. 當 **Dev_AI0 = 2**，只有序號 2 的單選按鈕被選擇。

OnRadioChange 事件函式屬性

和 **TagName (標籤名稱) 屬性**不同的是，每個單選按鈕都有自己的 **OnRadioChange** 事件函式屬性，如下所示：

```
void RadioButton6OnRadioChange(tWidget *pWidget, unsigned char ucValue)
{
    // ucValue 是包含本 RadioButton 之單選按鈕群組中，
    // 被選擇的 RadioButton 的序號。
    // 而您觸發的 RadioButton 和實際動作的 RadioButton 不一定是同一個。
    // 因此不一定是觸發這個 OnRadioChange 事件函式的 RadioButton。
}
```

3.4.15 Timer



注意:

這項元件只有在 **“程式類別為標準 C 語言”** 時才會生效。

什麼是 Timer (計時器)?

用來週期性執行一段程式碼的工具。

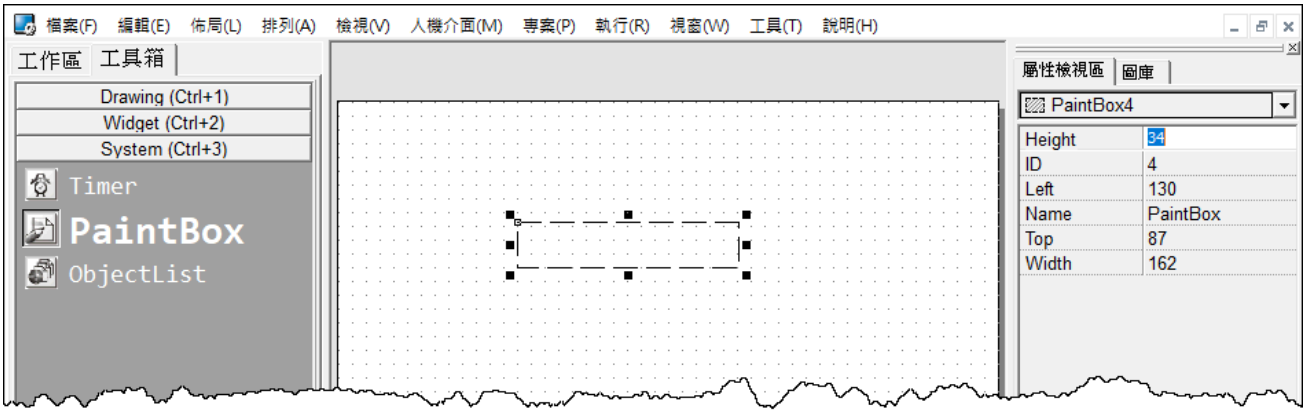
使用 Timer (計時器)

不必考慮計時器的大小和位置，因為下載至 TouchPAD 後，使用者是看不見計時器的，所以不必考慮點擊的位置，甚至計時器也不必一定要擺放在顯示頁上。

Timer (計時器) 的特有屬性:

屬性	說明
Enabled (啟動)	True (真)或 False (假)，計時器是否要開始計時。
Interval (週期)	相鄰兩次 **OnExecute (執行事件)** 的時間間隔。

3.4.16 PaintBox



注意:

這項元件只有在 **“程式類別為標準 C 語言”** 時才會生效。

什麼是 PaintBox (繪圖框)?

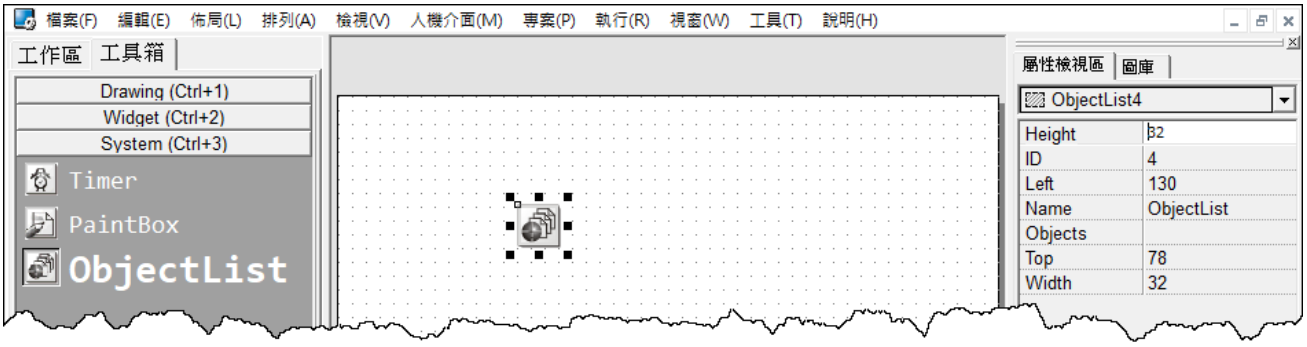
在執行時繪圖 (如，矩形、橢圓...等) 於設定區域的元件。

清除 PaintBox (繪圖框)

用函式 **“hmi_SetForeground”** 來畫一個白色的矩形以清除繪圖框。如下範例。詳請參閱 HMIWorks API Reference 1.12 hmi_SetForeground()。

```
//white; R-G-B; used to clear the PaintBox
hmi_SetForeground(pContext, 0xFFFFFF);

hmi_FillRect(pContext,
             widgetLeft(pWidget),
             widgetTop(pWidget),
             widgetRight(pWidget),
             widgetBottom(pWidget));
```

3.4.17 ObjectList



什麼是 ObjectList (物件清單)?

物件清單是用來紀錄從 **“圖庫”** 中選取出來之物件的清單。和 **TextPushButton** (文字彈回式按鈕)、**Slider** (滑桿)、**CheckBox** (核取方塊) 和 **RadioButton** (單選按鈕) 的 **“RefObject (參考物件)” 屬性**共用，使用者可以輕鬆地切換兩張或多張圖片，如，在開關應用時，特別好用。

ObjectList (物件清單) 的特有屬性:

屬性	說明
Objects (清單)	紀錄圖片的清單。

雙擊“物件清單”圖示來開啟 **“物件清單(ObjectList)”** 視窗，關於圖片選項如下:

屬性	說明	預設
與控項同大小	調整控制項參考的物件清單中圖片的大小，使圖片覆蓋整個控制項區域。	勾選
高色彩	編譯時使物件清單中的圖片做為 16-bit 色彩(高色彩) 或 8-bit 色彩的。	勾選
強制編譯	強制 HMIWorks 去編譯物件清單的圖，即使沒有任何控制項參考到這個物件清單。	無勾選

注意:

若要正確地顯示**透明 (遮罩) 顏色**，下列的條件必須滿足。

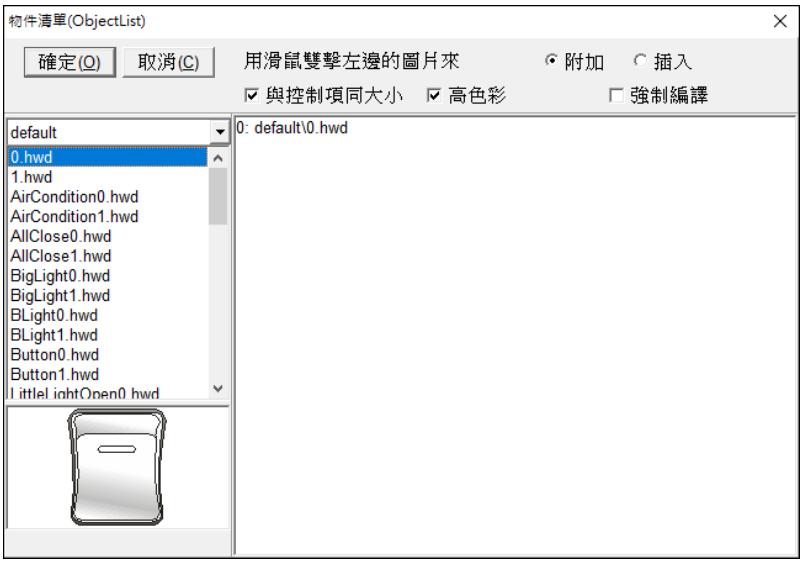
1. 在物件清單視窗中的 **“與控制項同大小”** 選項必須勾選。
2. 在物件清單中每個物件只能含有一個 **Picture (圖片框) 元件**。當 **“新增到圖庫 (Add to library)”** 時，該圖片框會自動置於群組中。
3. **TextPushButton (文字彈回式按鈕)** 當 **“外形 (Shape)” 屬性**設為 **“Circular(圓形)”** 時，即便物件清單指定給 **RefObject (參考物件) 屬性**，也不支援遮罩 (透明) 顏色。

使用 ObjectList (物件清單)

1. 不必考慮物件清單的大小和位置，因為下載至 TouchPAD 後，使用者是看不見物件清單的圖示的，所以不必考慮點擊的位置。

2. 程式下載至 TouchPAD 後，物件清單會依據 **TextPushButton** (文字彈回式按鈕) 及 **CheckBox** (核取方塊) 的狀態，來決定顯示在 TouchPAD 上的圖片。**TextPushButton** (文字彈回式按鈕) 及 **CheckBox** (核取方塊) 原來的外觀則被這些圖片取代，不再看得見了。

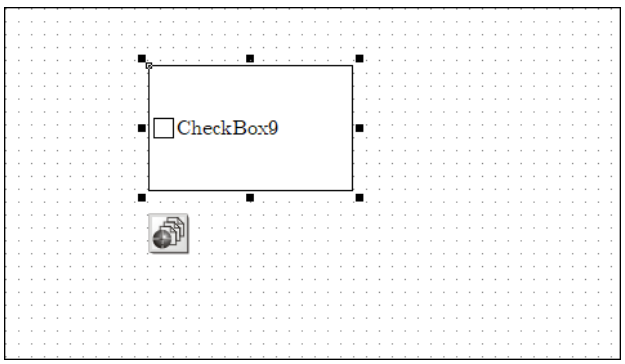
3. 如果要從 **“圖庫”** 新增兩張圖片到物件清單裡，雙擊 **“物件清單”** 圖示來開啟 **“物件清單 (ObjectList)”** 視窗。再雙擊點選清單中的圖片物件，點選後可以看見右方的顯示區多了剛點選的物件。



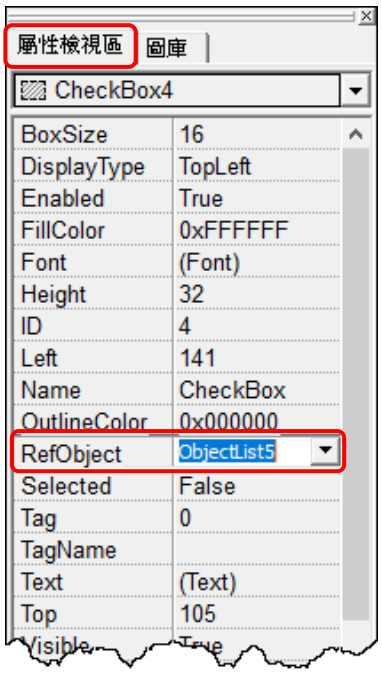
注意:

如果要**刪除**在物件清單視窗右側區域中的物件，請雙擊點選該物件便能刪除。

4. 例如，在顯示頁中放置一個 **CheckBox (核取方塊)**。



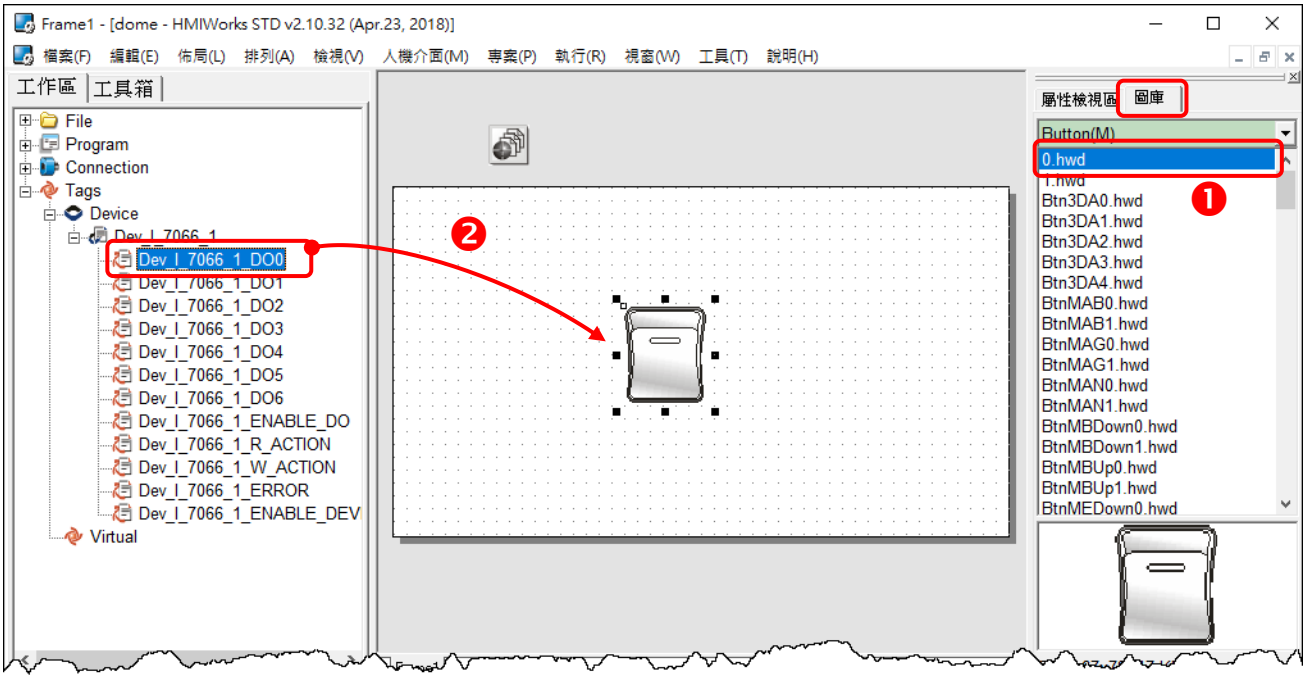
5. 在 **“屬性檢視區”** 中的 **“RefObject(參考物件)”** 屬性欄中，選擇要連結的物件清單。



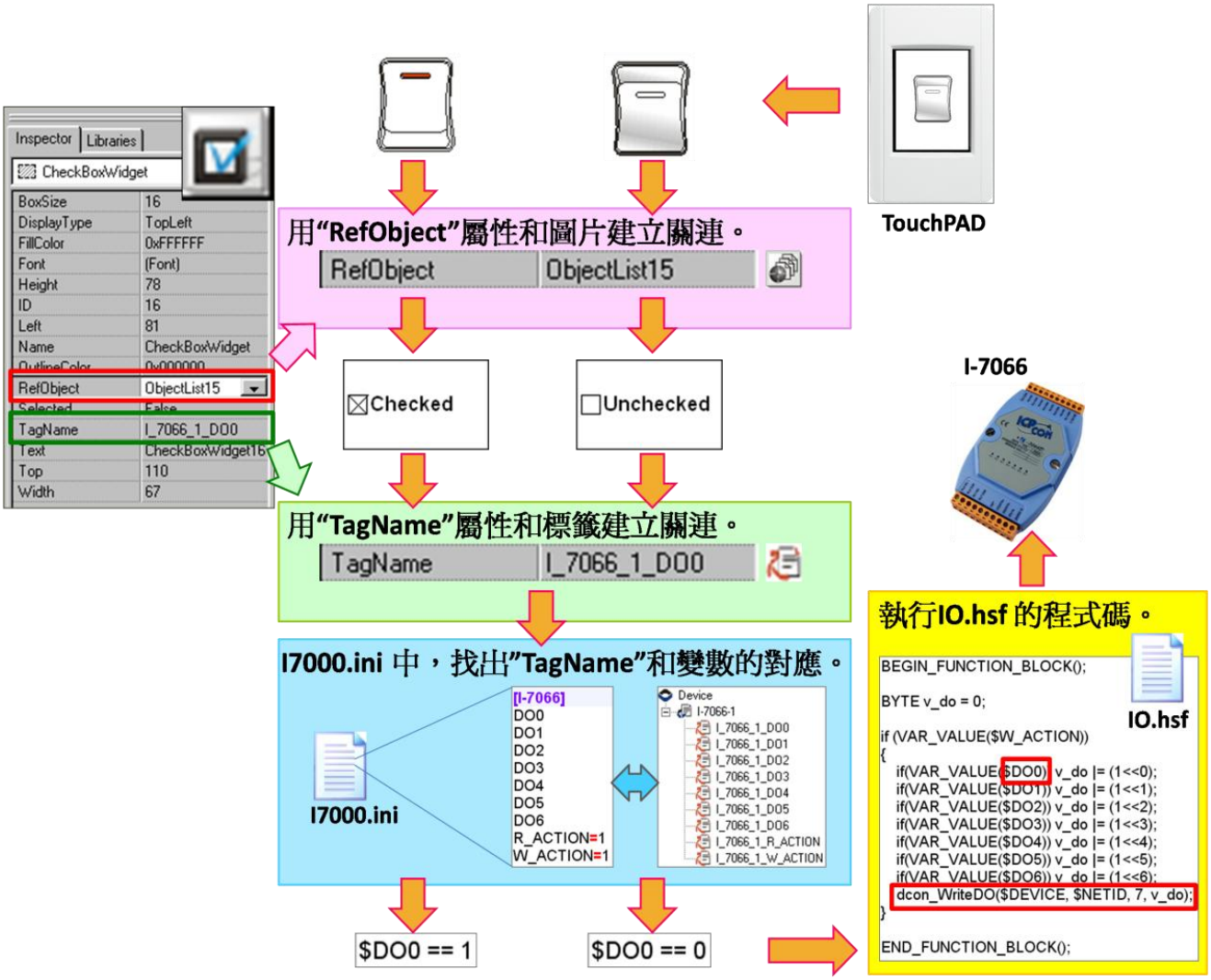
6. 編譯下載後，只能看見圖片，**TextPushButton** (文字彈回式按鈕) 及 **CheckBox** (核取方塊) 原來的外觀不再看得見了，觸碰該圖片可以切換到另一張圖片。

從 TouchPAD 到 I/O 模組的關連

以 I-7066 為例，從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(P)”** 項目 (或按 **<F3>鍵**) 來產生 I-7066 用的標籤，然後再把它拖放到顯示頁上。



如下圖所示，HMIWorks 做下列步驟建立 TouchPAD 和 I/O 模組的關連。



注意:

TagName (標籤名稱) 屬性只有在“**程式類別為階梯圖**”時才有作用。

若“**程式類別為標準 C 語言**”時就簡單多了，只要在 **CheckBox (核取方塊)** 的 **事件函數 (Event Function)** 中實作控制 I/O 的函數 **dcon_WriteDO** 就可以了。

3.5選單

本節將介紹 **“主選單欄”** 及**“彈出式選單”**，詳細說明如下。

3.5.1 主選單欄

“主選單欄” 設定項目詳細說明如下:

檔案(F)

檔案功能選單中提供您新增、開啟、儲存 HMIWorks 專案。詳細關於 **“匯入圖片”** 請參考 第 3.5.2.5 節 匯入圖片(I)。

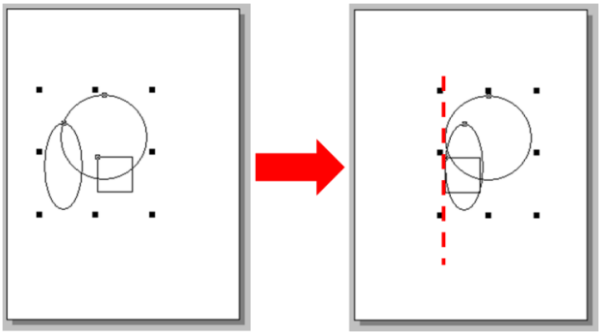
編輯(E)

編輯功能選單中提供您修改專案中的物件，您可對元件進行複製、貼上、刪除、90 度旋轉、水平及垂直翻轉…等。

佈局(L)

佈局功能選單中提供可以讓物件沿軸對齊，如: 靠左/靠右對齊、靠上/靠下對齊、水平/垂直置中…等。 **注意：**所有對齊的功能都是以最後畫的圖形為對齊參考標準。如下例所示，參考標準為最後畫的正方形。

範例: 畫三個圖形，再點選 **“靠左對齊”** 結果最後畫的是正方形，所以所有圖形的最左邊都對齊正方形之最左邊。



排列(A)

排列功能選單中提供可以讓選擇的物件移到某一層，以及可將物件框在一起，形成一個集合為一個群組。

檢視(V)

檢視功能選單中提供可以設定 HMIWorks 介面，如: 變更介面語言 (參考 第 3.2.1 節 語言選項) 及是否顯示屬性檢視區、圖庫、結果視窗…等。

人機介面(M)

人機介面功能選單中提供您可以管理顯示頁 (新增、刪除和重新命名)、建立標籤 (裝置及虛擬標籤) 及開啟階梯圖設計家 (參考 第 3.3 節 階梯圖設計家) …等。

專案(P)

專案功能選單中提供您可以配置設定專案 (參考 第 3.2.2 節 專案組態設定)、開啟專案目錄及查看專案檔…等。

執行(R)

執行功能選單中提供您可以設置裝置、程式編譯及產生原始碼，以及將應用程式載入至 TouchPAD 中…等。(參考 第 4 章 完成一個簡單專案)

視窗(W)

視窗功能選單中提供您可以設定多個視窗顯示方式，如: 層疊、平鋪…等。

工具(T)

工具功能選單中提供您可以更新 TouchPAD 的 MiniOS8。**注意:** 此功能只適用於 TPD-280U/238U/430/433/432F/433F 及 VPD-130(N)/132(N)/133(N)/142(N)/143(N)。

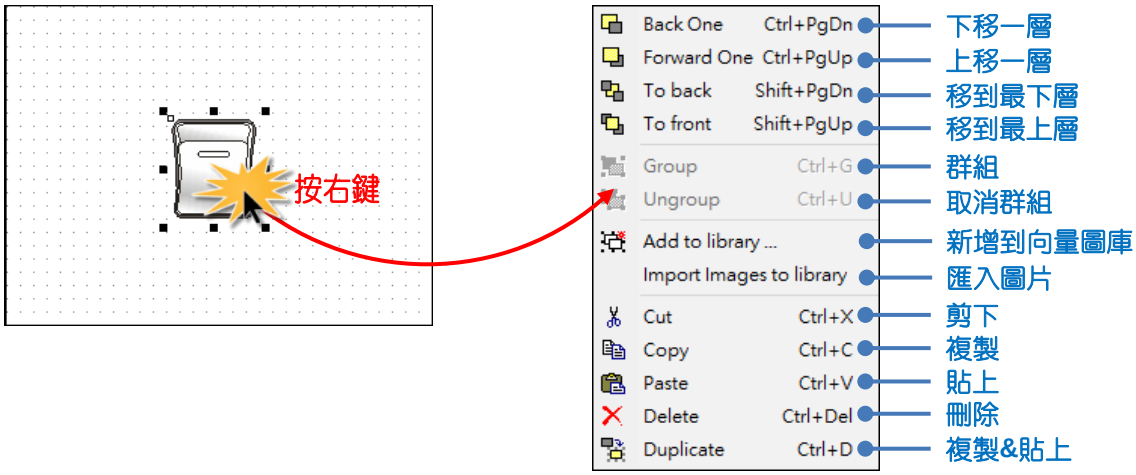
說明(H)

說明功能選單中提供您可以設定顯示歡迎畫面及查看關於 HMIWorks 版本及電腦的記憶體…等資訊。

.

3.5.2 彈出式選單及圖庫管理

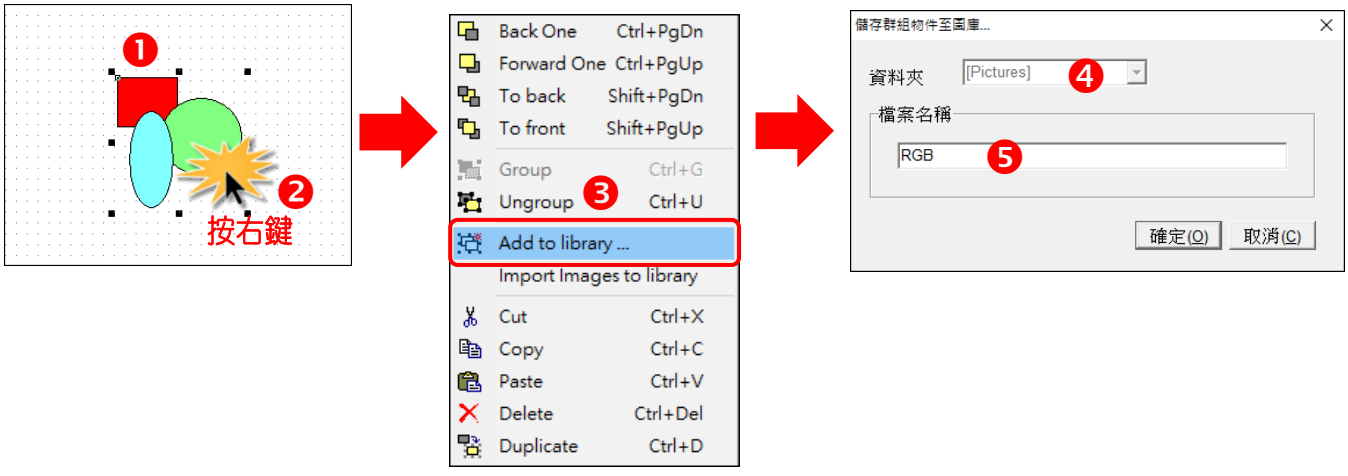
在顯示頁設計區中，我們在物件上用滑鼠右鍵點擊一下將會開啟 **“彈出式選單”**。

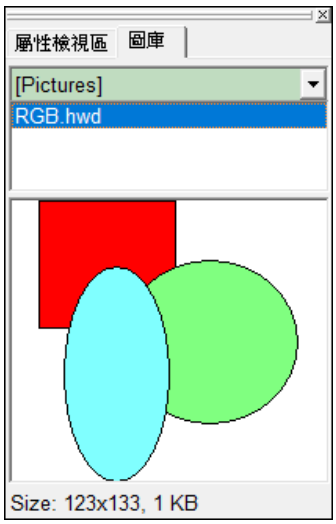


3.5.2.1 如何新增物件到圖庫

所有新增物件的檔案，它們的副檔名為 **“hwd”**。下面我們將新增一個物件圖到圖庫中：

1. 畫三個圖形，可將它們設成同一群組。
2. 滑鼠右鍵點選群組的物件開啟彈出式選單。
3. 點選 **“Add to library …”** 項目。
4. 在下拉式選單中選擇資料夾 (預設為 **[Pictures]**)，新增的物件將被置於此資料夾中。
5. 為要新增到 **“圖庫”** 的物件取名，按下 **“確定”** 按鈕。





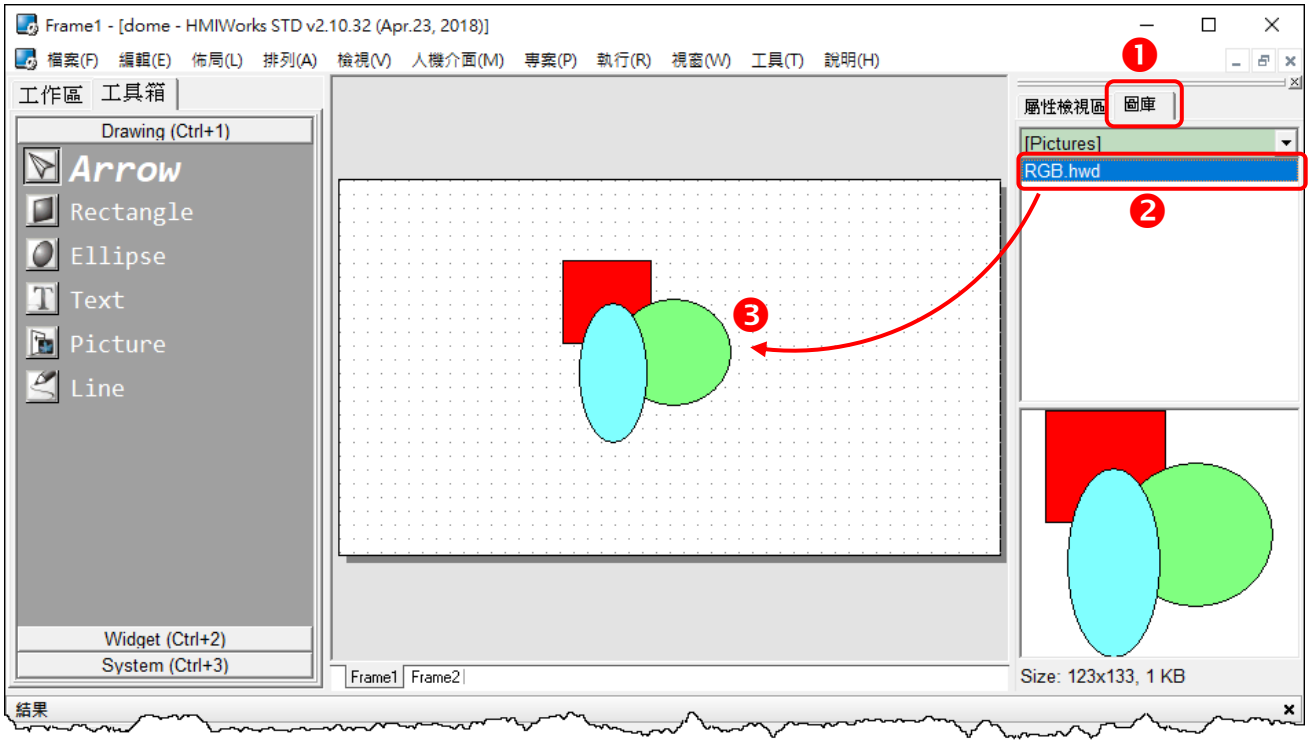
注意:

可以在“**圖庫 (Libraries)**”視窗中預覽剛新增的物件。該物件的大小顯示在其下方。

3.5.2.2 如何使用圖庫中的物件

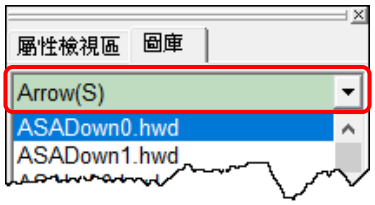
依據下列步驟來使用圖庫中物件:

1. 點選“**圖庫**”。
2. 點選所要的物件，可以在下面的窗格中預覽。
3. 點選 (先不要釋放滑鼠) 預覽窗格中 (或清單中) 的物件，並把它拖到顯示頁設計區上，再釋放滑鼠將該物件丟到顯示頁設計區。



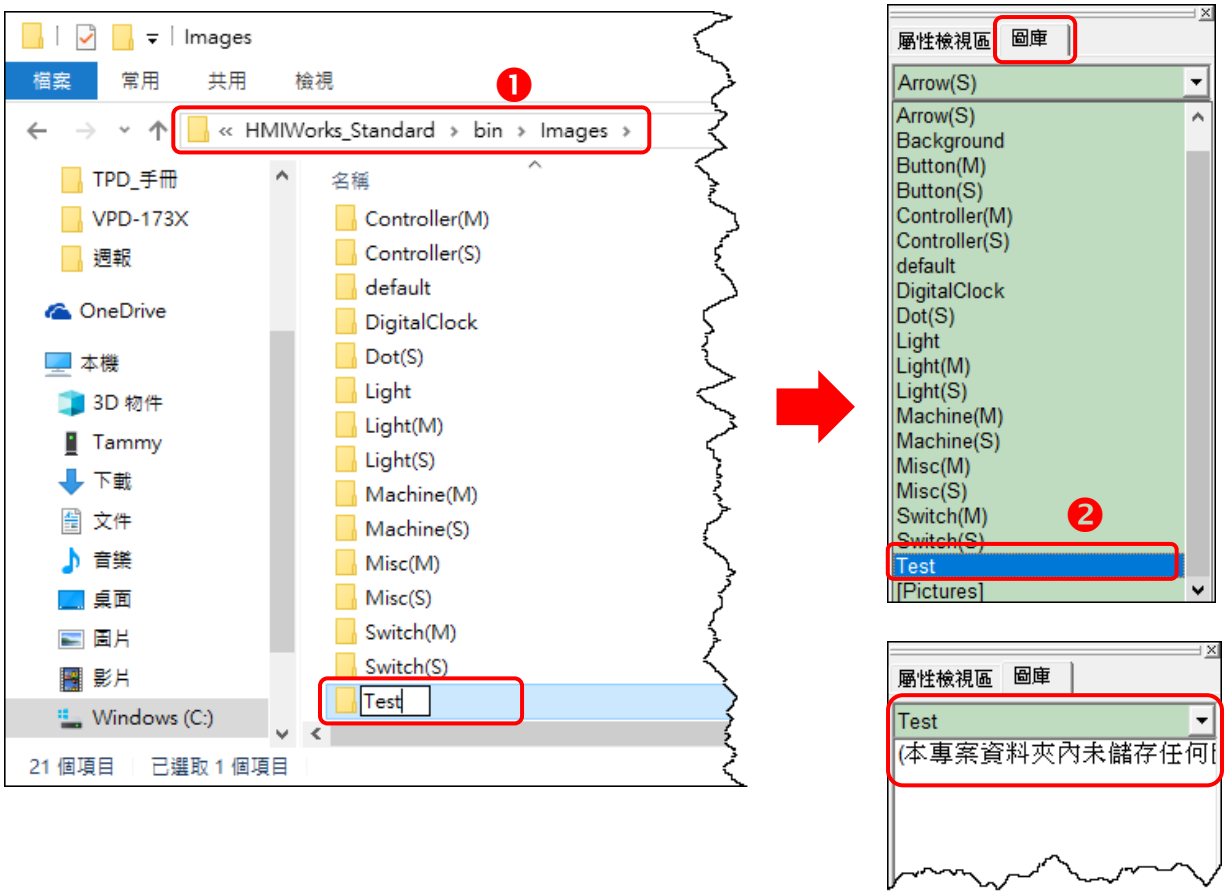
3.5.2.3 如何在圖庫新一個分類

在 “**圖庫**” 視窗中，可以看到分類，以 **“Arrow(S)”** 為例 (如右圖)。您可以在路徑 「**HMIWorks_install_path\bin\Images**」裡新增一個資料夾就可以在 “**圖庫**” 中新增一個分類。



依據下列步驟，我們將在圖庫中新增一個分類:

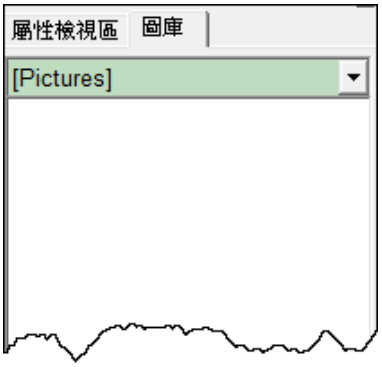
1. 到 HMIWorks 的安裝路徑下，找到「**C:\ICPDAS\HMIWorks_Standard\bin\Images**」的子目錄，在該子目錄下新增一個叫做 **“Test”** 的分類。
2. 再重新回到 HMIWorks 的 “**圖庫**” 視窗，點選分類的下拉式選單，便可以看到剛新增的 **“Test”** 名稱。



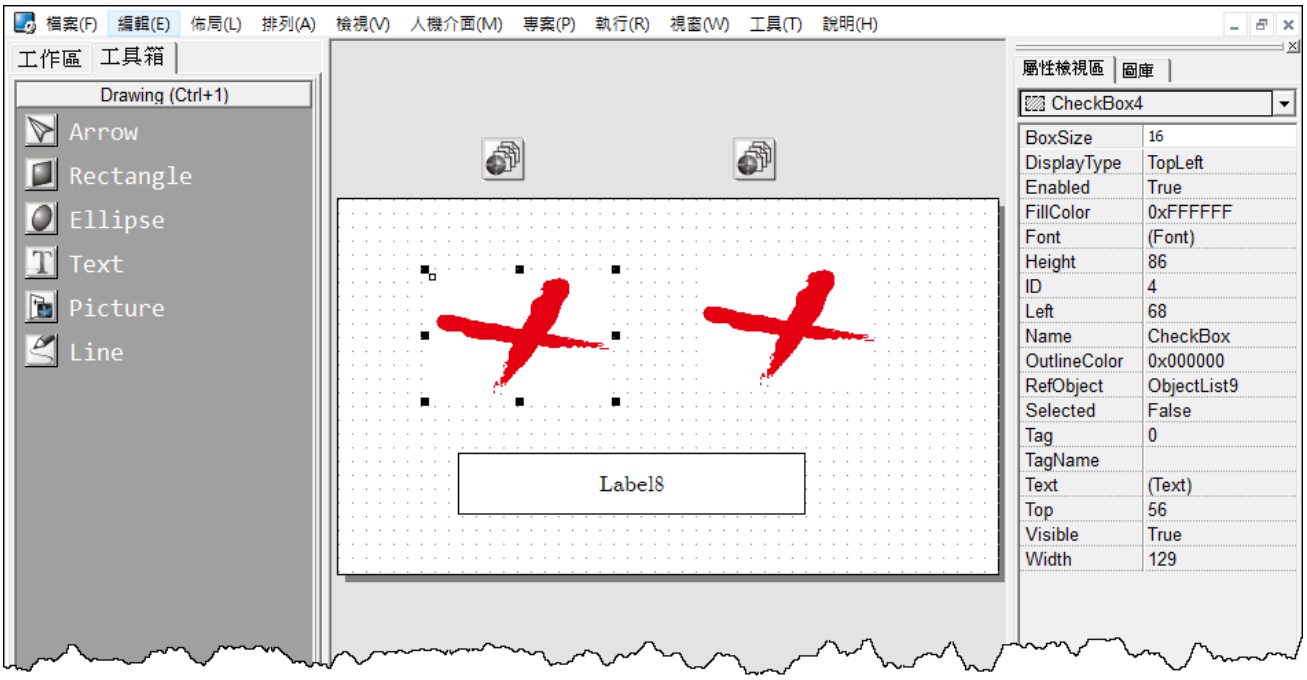
3.5.2.4 在專案目錄中特殊的 [Picture] 目錄

點選 **“圖庫”**，從分類下拉式選單中，選擇 **[Picture]** 項目，如下圖所示。

與下拉式選單中其他的選項不同的是，**[Picture] 資料夾是放在專案目錄內**。任何新增到 [Picture] 資料夾的物件，將會永遠跟著專案在一起，這有效解決了專案若移到其他的電腦中，圖片可能會無法存取的情形。

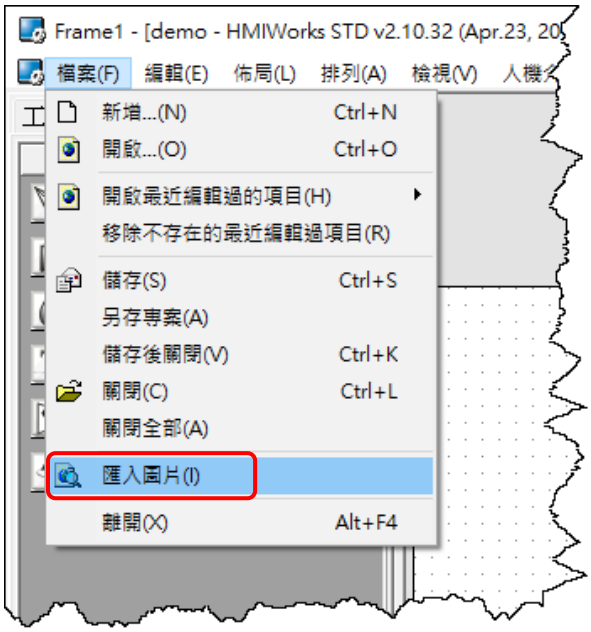


當開啟一個舊的專案的時候，若 HMIWorks 無法載入控制項所連結到的圖片時，就會在其位置上顯示一個紅色的叉叉，如下圖所示。此時，可參考 FAQ: 如何修復圖片損毀或遺失 (顯示紅 X) 問題? 來解決此問題。



3.5.2.5 匯入圖片(I)

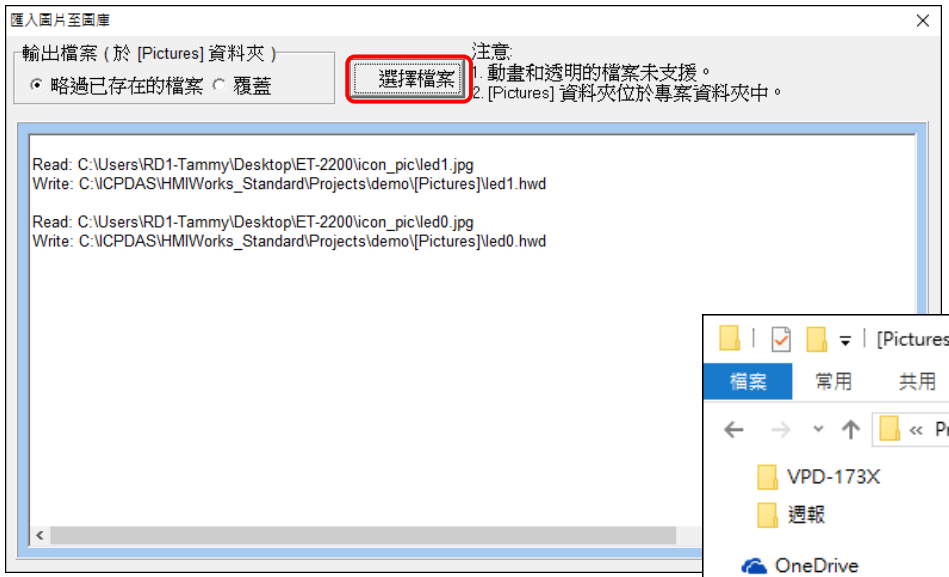
從 **“檔案(F)”** 功能選單中，點選 **“匯入圖片(I)”** 項目來開啟 “匯入圖片至圖庫”視窗。

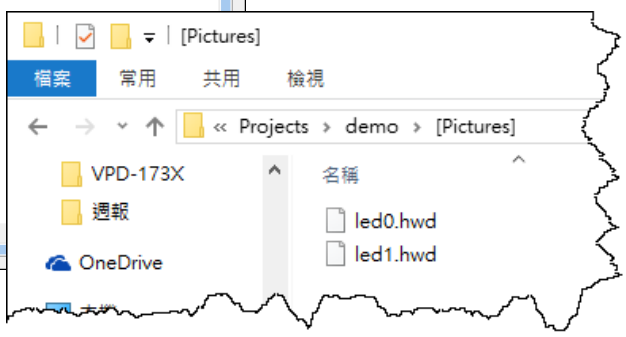


注意:

因為轉好的檔案 **“.hwd”** 會放在目前專案目錄下的 [Pictures] 資料夾，所以在執行這個項目前，必須先新增或是開啟一個專案。

按下 **“選擇檔案”** 按鈕來選取超過一個以上的圖形檔，接著會把這些檔案轉成 HMIWorks 可以辨認的 **“.hwd”** 檔，最後會將轉好的檔案會放在目前專案目錄下的 **[Pictures] 資料夾**。





注意:

目前支援的圖形檔格式 JPG/BMP/WMF/EMF。

4. 完成一個簡單專案

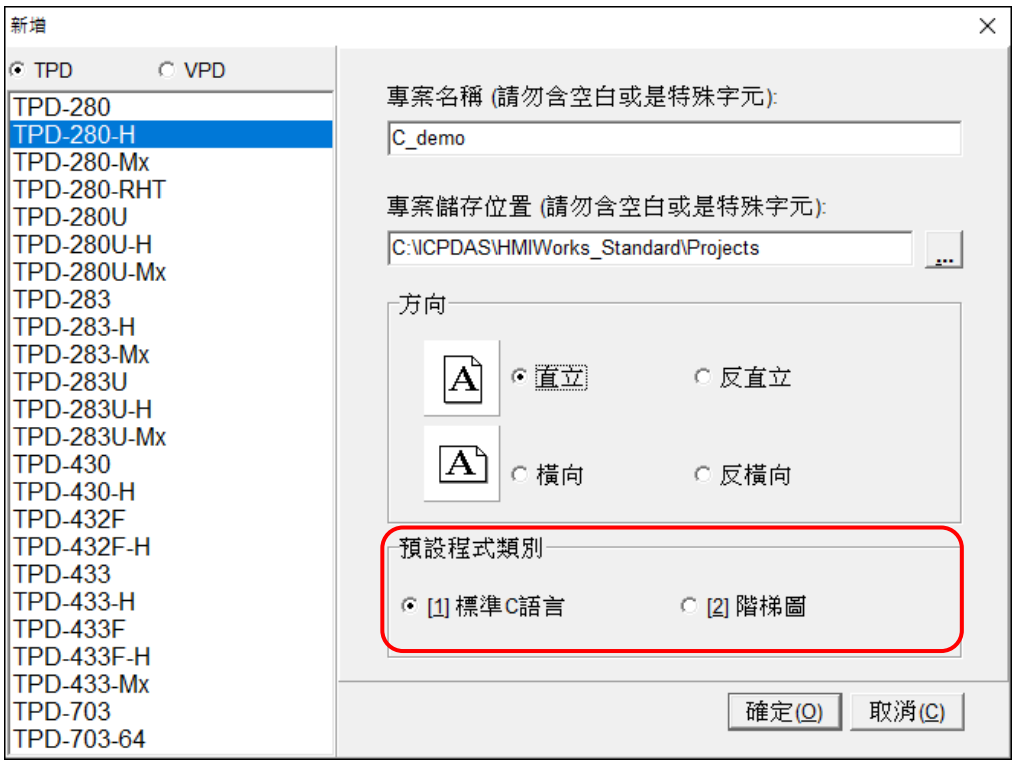
HMIWorks 有兩種程式類別: **“標準C語言”** 及 **“階梯圖”**，本章將分別介紹如何用不同的程式方法來開發一個專案，以及如何整合 TouchPAD 與 I/O 模組。

4.1 使用標準 C 語言開發專案

下面我們將以 TPD-280-H 為範例，詳細說明如何來建立一個標準 C 語言的專案。

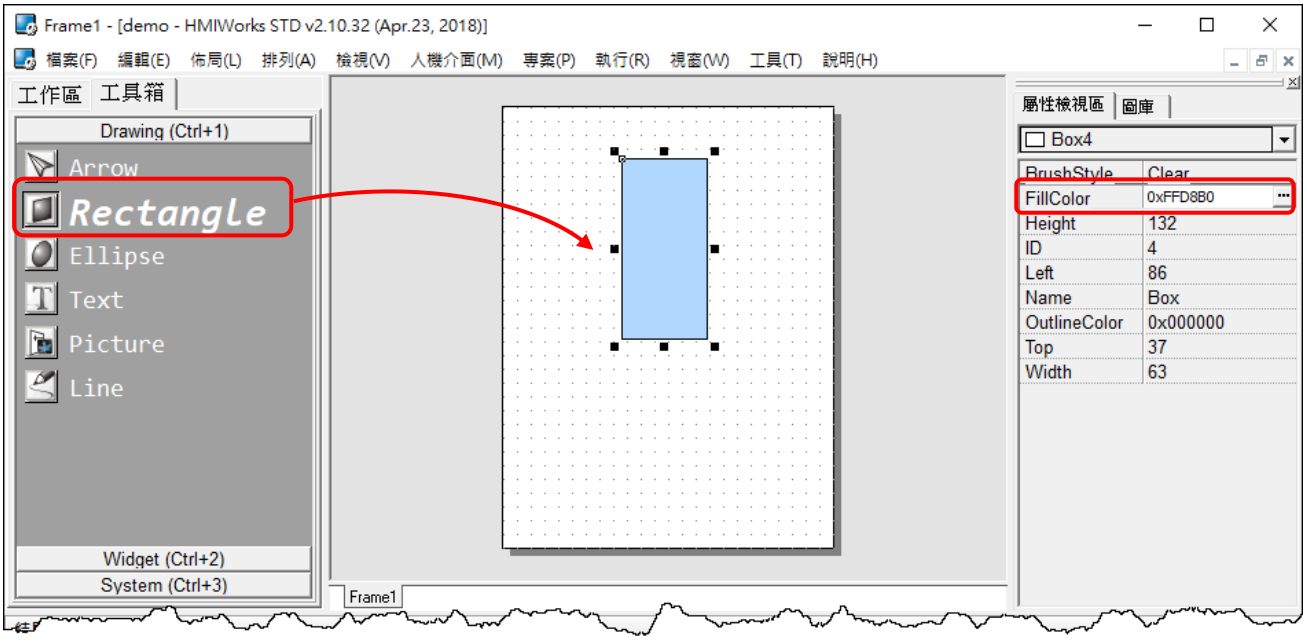
步驟 1 建立一個新的專案

從 **“檔案(F)”** 功能選單中，點選 **“新增...(N)”** 項目來開啟 “新增”視窗。接著在開新專案的視窗中選擇型號、決定專案名稱、專案儲存位置、方向及**預設程式類別 (請選擇 “[1] 標準 C 語言”)**。

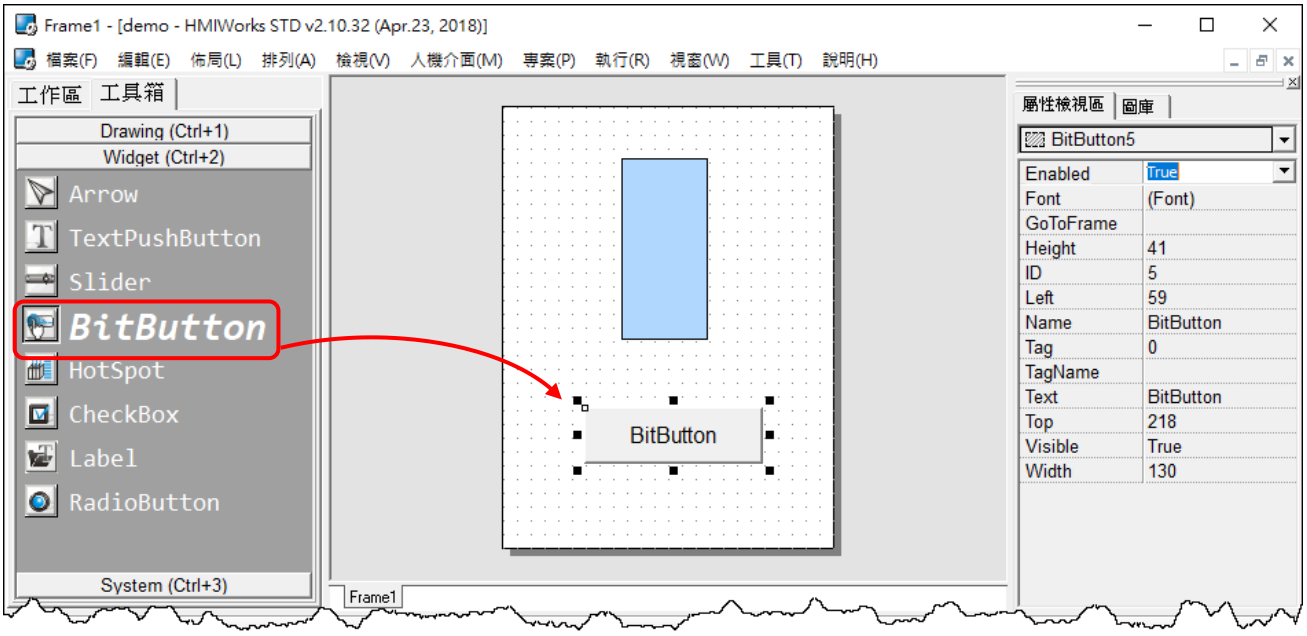


步驟 2 設計使用者圖形介面 (GUI)

這裡我們將簡單畫一個矩形 (Rectangle) 並填上顏色。當然使用者可以設計更漂亮的介面。

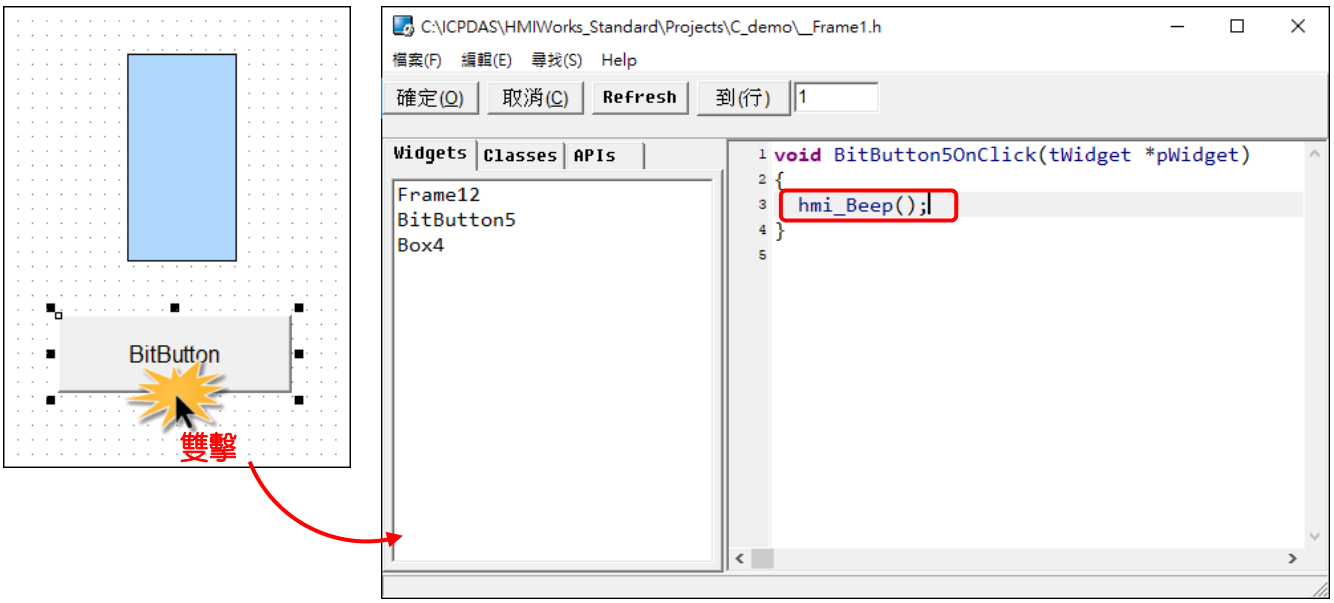


然後在“**工具箱**”中點選一個**控制項 (Widget)**，如: **BitButton (位元圖按鈕)**。



步驟 3 編寫程式碼

雙擊 **“BitButton (位元圖按鈕)”** 開啟程式碼編輯視窗。
用 **“hmi_Beep();”** 函式來發出一嗶聲，接著儲存檔案，按下 **“確定(O)”**。

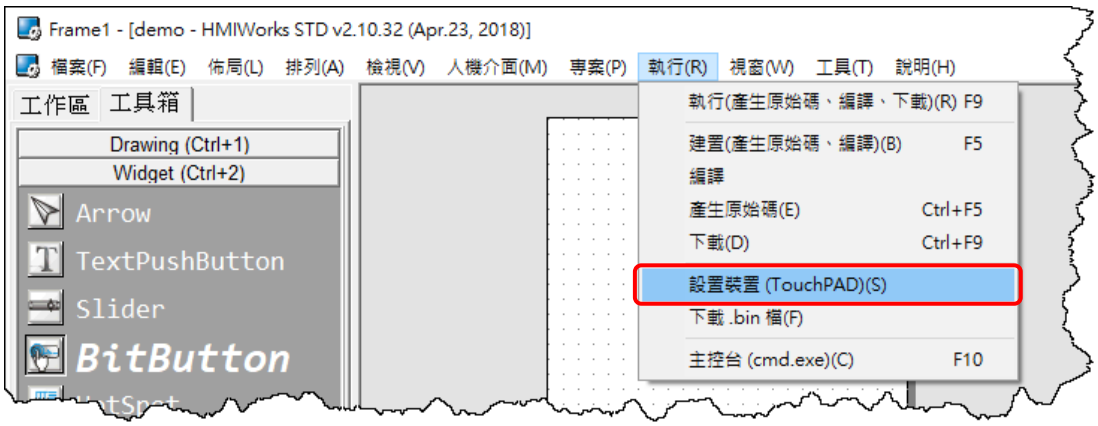


步驟 4 設置裝置(TouchPAD)

您必需依據 TouchPAD 類型來選擇程式載入方式。更多更詳細程式載入說明，可參考 TouchPAD 硬體使用手冊中 第 3.4 節 程式載入至 TouchPAD。

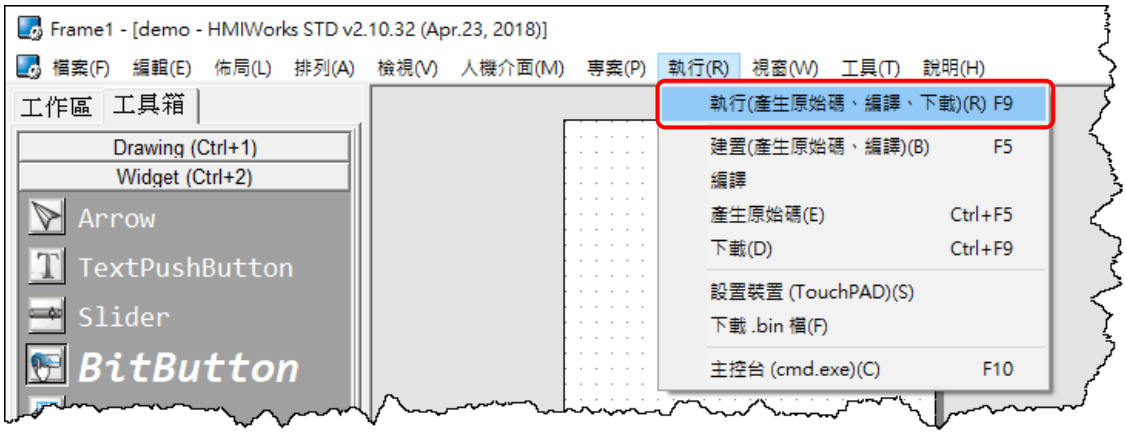
此範例中，我們使用 TPD-280-H 透過 RS-485 與電腦主機連接在一起，並將 TouchPAD 上**旋轉開關(Rotary Switch)** 切換至 **“1” (更新模式)** 的位置，再將 TouchPAD 斷電重新啟動。

點選 **“執行(R)” → “設置裝置(TouchPAD)(S)”** 來選擇正確的 COM Port 配置。

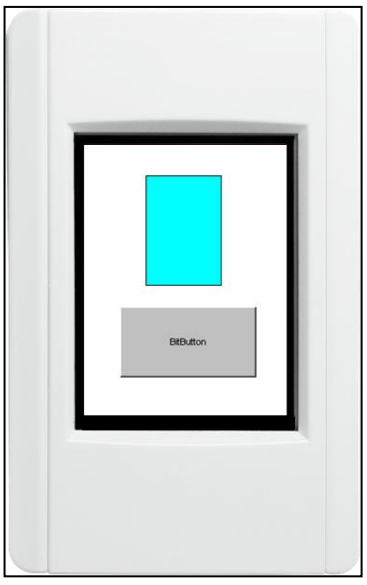


步驟 5 建置、下載、執行

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。
一旦載入完成，請將 TouchPAD 上的**旋轉開關(Rotary Switch)** 調回 **“0” (執行模式)** 的位置，再將 TouchPAD 斷電關機。



再將 TouchPAD 供電開機後的執行畫面如下，按下按鈕後就可以發出嗶聲。

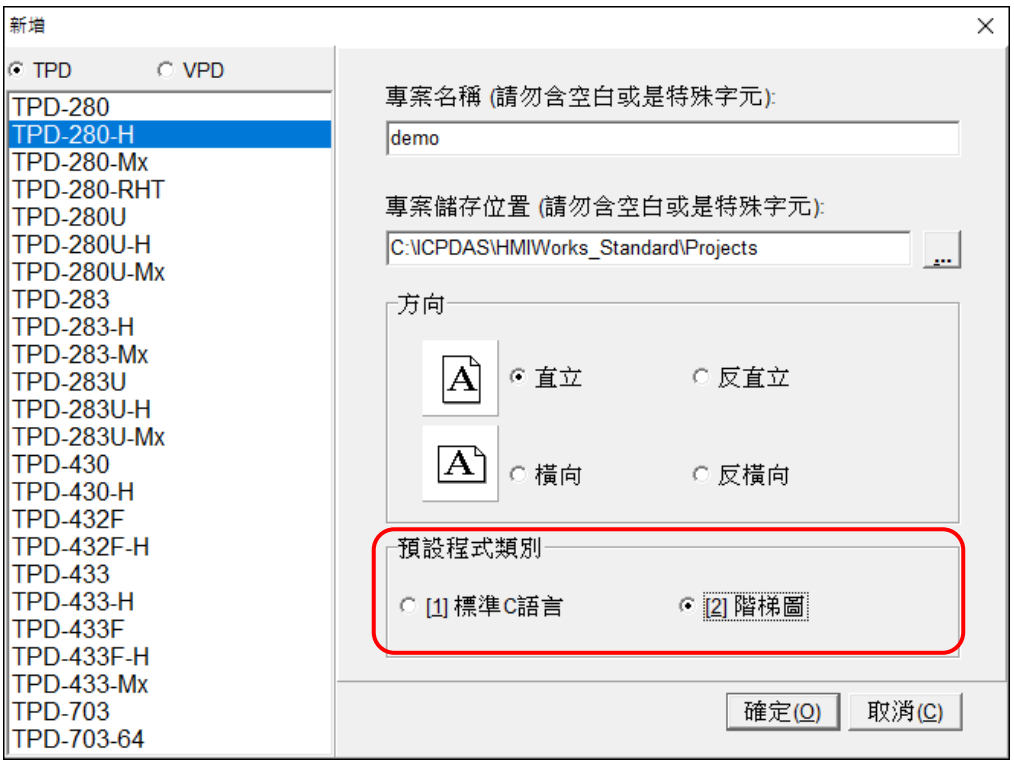


4.2 使用階梯圖開發專案

下面我們將以 TPD-280-H 為範例，詳細說明如何來建立一個階梯圖的專案。

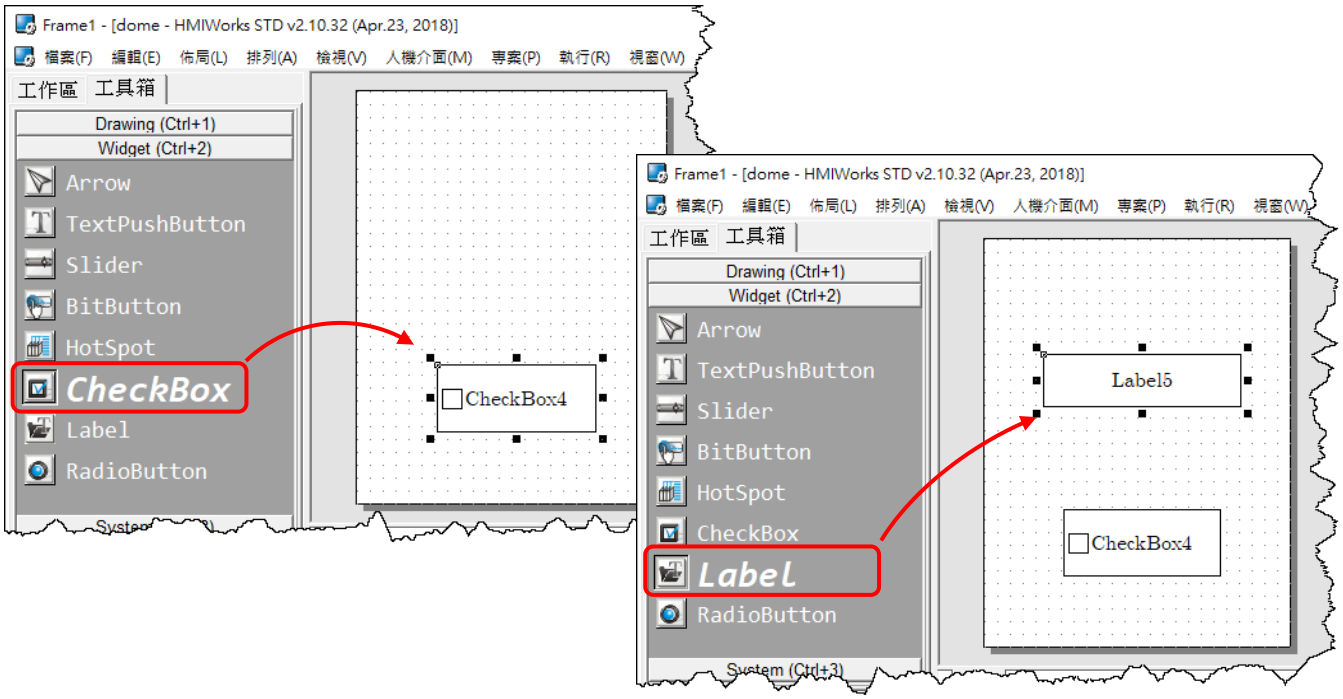
步驟 1 建立一個新的專案

從 **“檔案(F)”** 功能選單中，點選 **“新增...(N)”** 項目來開啟 “新增”視窗。接著在開新專案的視窗中選擇型號、決定專案名稱、專案儲存位置、方向及**預設程式類別 (請選擇 “[2] 階梯圖”)**。



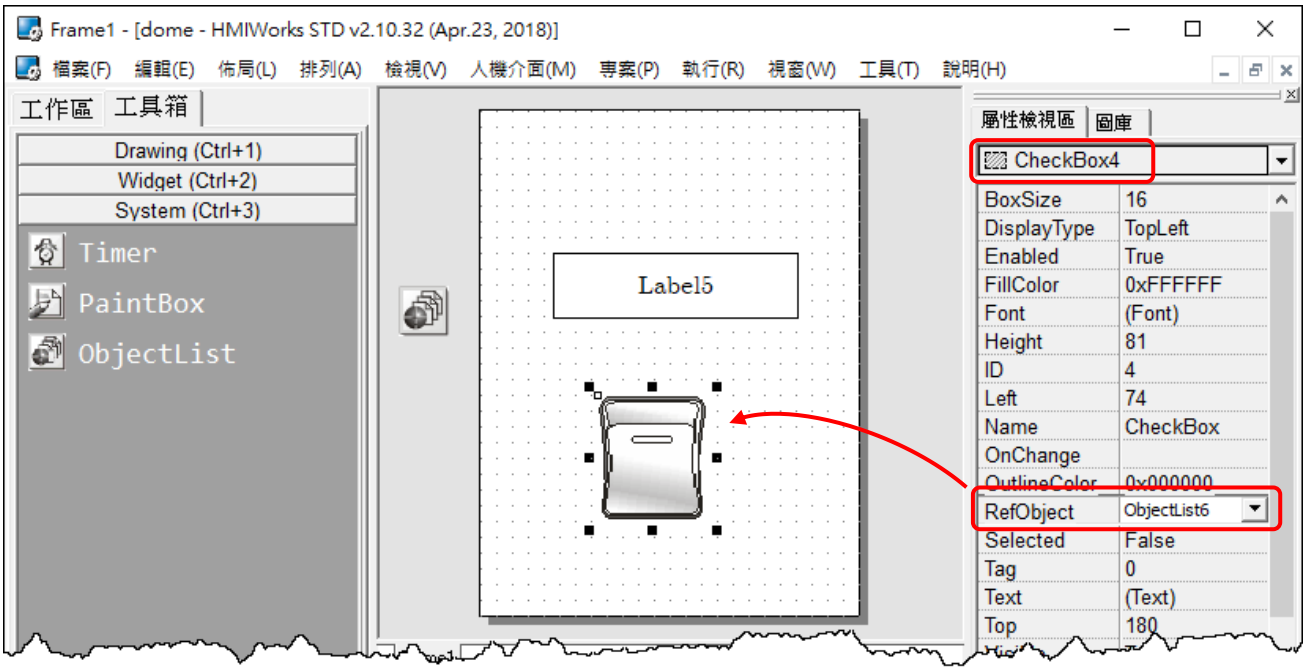
步驟 2 設計使用者圖形介面（GUI）

1. 這裡我們在顯示頁上放一個 **CheckBox (核取方塊)** 及一個 **Label (文字顯示框)**，將把 CheckBox 當作輸入，Label 當作輸出。 詳細如下一頁操作圖示。



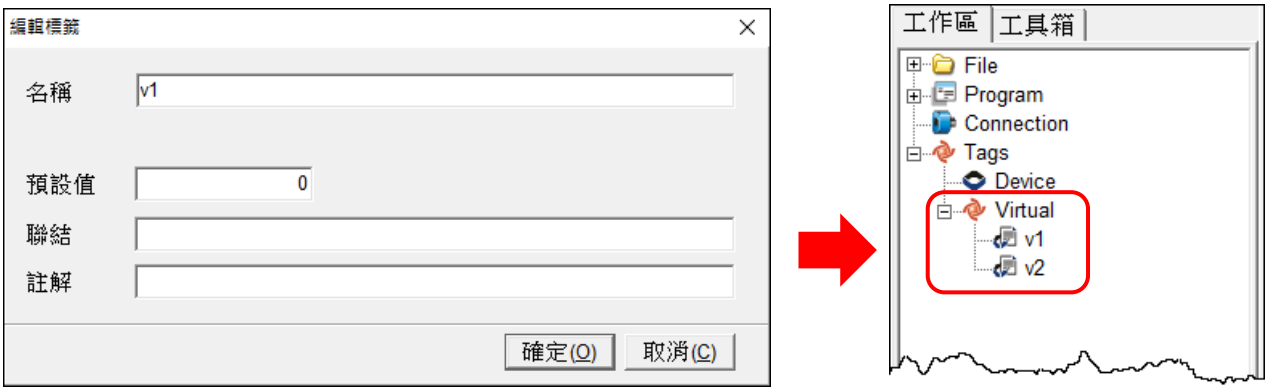
2. 拖一個 **ObjectList (物件清單)** 並把它放在顯示頁設計區。雙擊 **ObjectList** 的圖示叫出 **“物件清單”** 視窗。在 **“物件清單”** 視窗中點選想要的圖形，共需要兩個圖形，分別會對應到 **CheckBox (核取方塊)** 的兩個狀態。按下 **“確定”** 結束這個步驟。

3. 點選 **CheckBox (核取方塊)** 後，在 **“屬性檢視區”** 中的 **“RefObject”** 設定欄位選擇連結的 **ObjectList (物件清單)**。

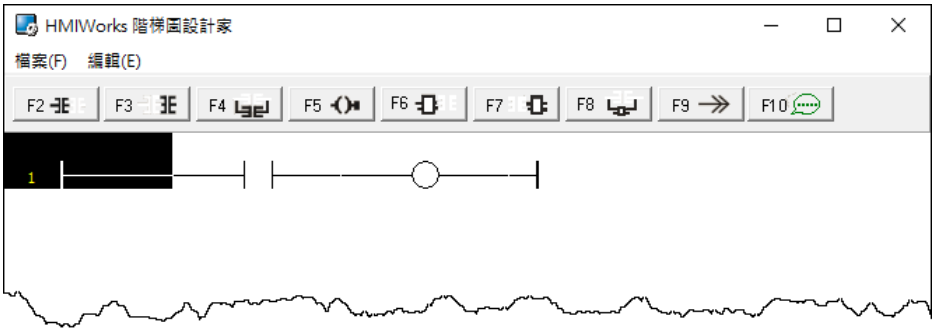


步驟 3 設計階梯圖

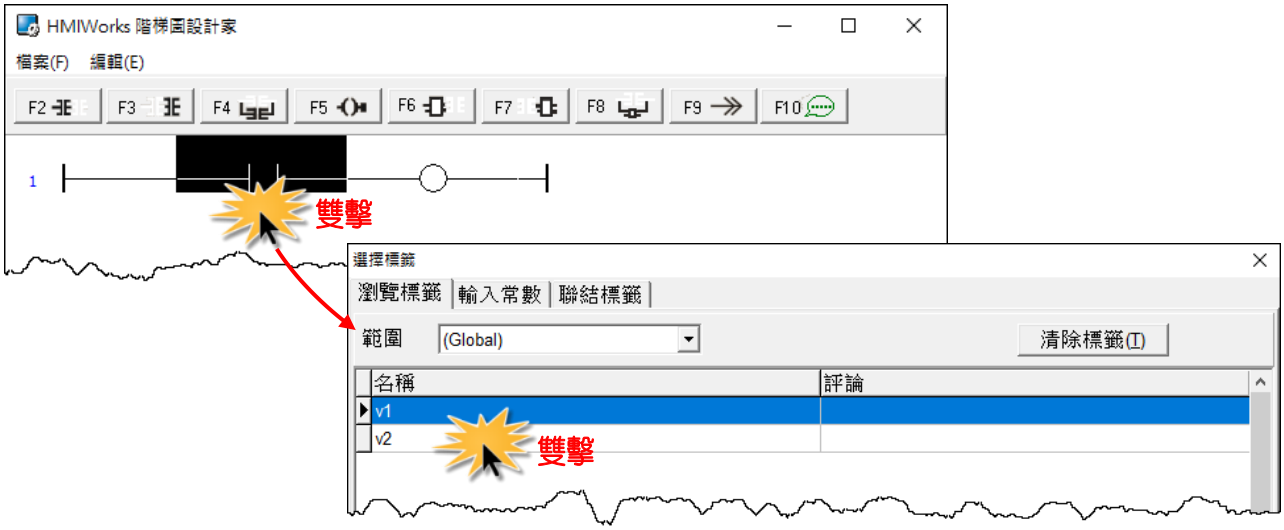
1. 從 **“人機介面(M)”** 功能選單中，點選 **“新增虛擬標籤(N)”** 項目來開啟“編輯標籤”視窗 (或按 **<F2>** 鍵)。這裡我們新增 v1 及 v2 二個標籤要給階梯圖使用，且可以在 **“工作區”** 中 **“Virtual”** 下查看到。



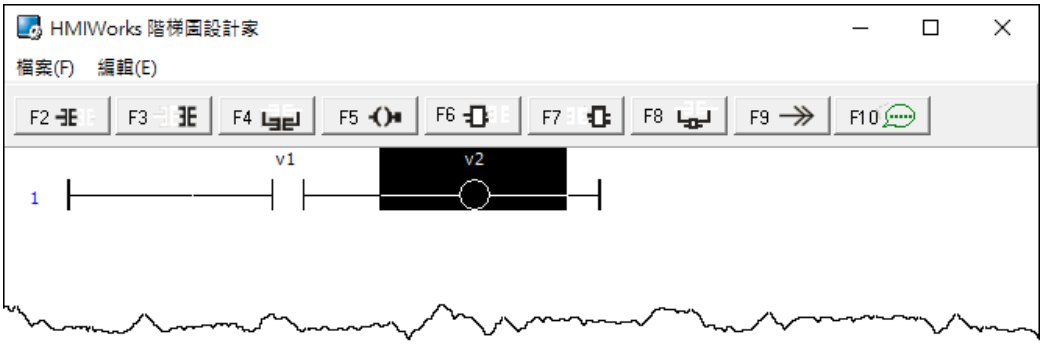
2. 從 **“人機介面(M)”** 功能選單中，點選 **“階梯圖設計家(L)”** 項目(或按 **<F4>** 鍵)。接著在“階梯圖設計家”視窗中，按 **<F2>** 鍵來產生新的一階。



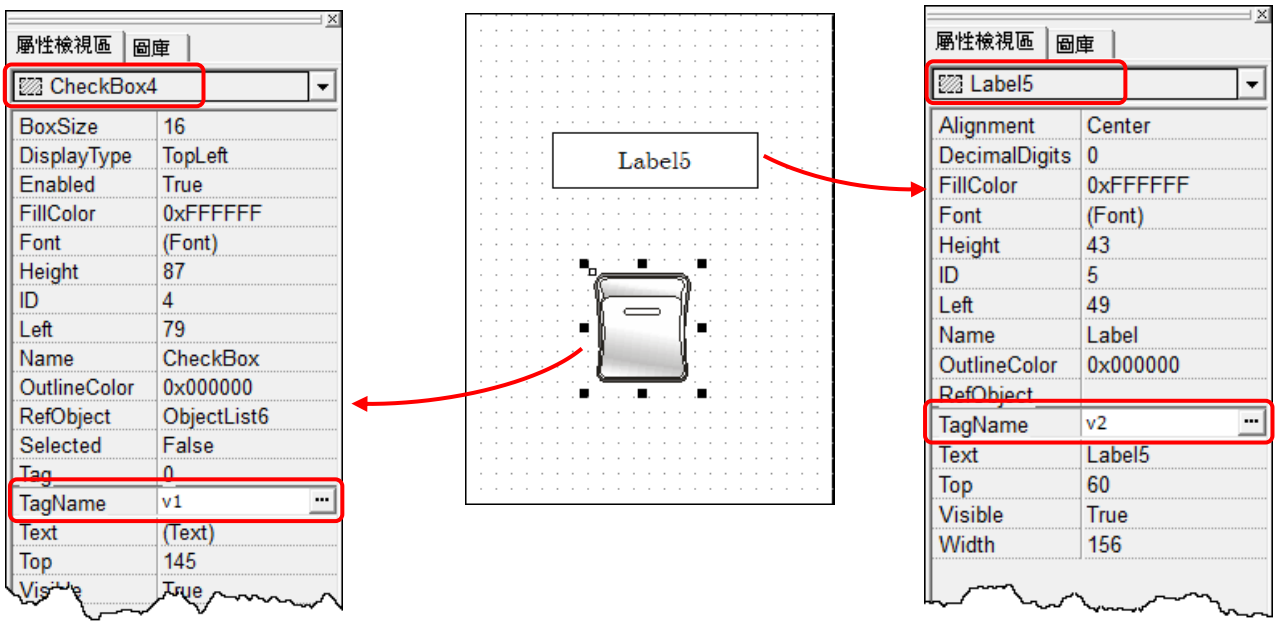
3. 在剛產生的一階中雙擊開關輸入 (Contact Input)，然後出現 **“選擇標籤”** 視窗。選擇要和開關輸入產生關連的標籤。



4. 這裡我們使標籤 v1 和開關輸入 (Contact Input) 產生關連，標籤 v2 和線圈輸出 (Coil Output)產生關連。



5. 點選核取方塊 (CheckBox)，在 **“屬性檢視區”** 中的 **“TagName” 屬性**欄選擇連結 v1 標籤。在點選文字顯示框 (Lable)，在 **“屬性檢視區”** 中的 **“TagName” 屬性**欄選擇連結 v2 標籤。

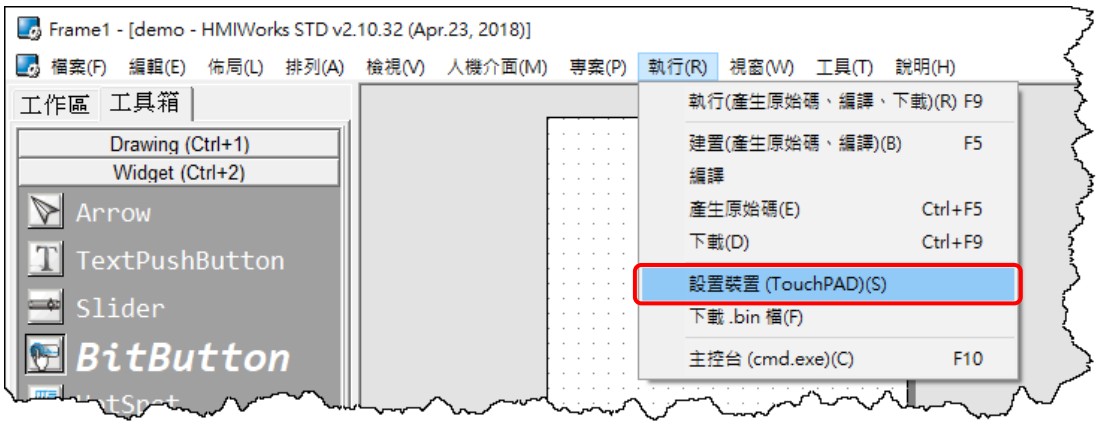


步驟 4 設置裝置(TouchPAD)

您必需依據 TouchPAD 類型來選擇程式載入方式。更多更詳細程式載入說明，可參考 TouchPAD 硬體使用手冊中 第 3.4 節 程式載入至 TouchPAD。

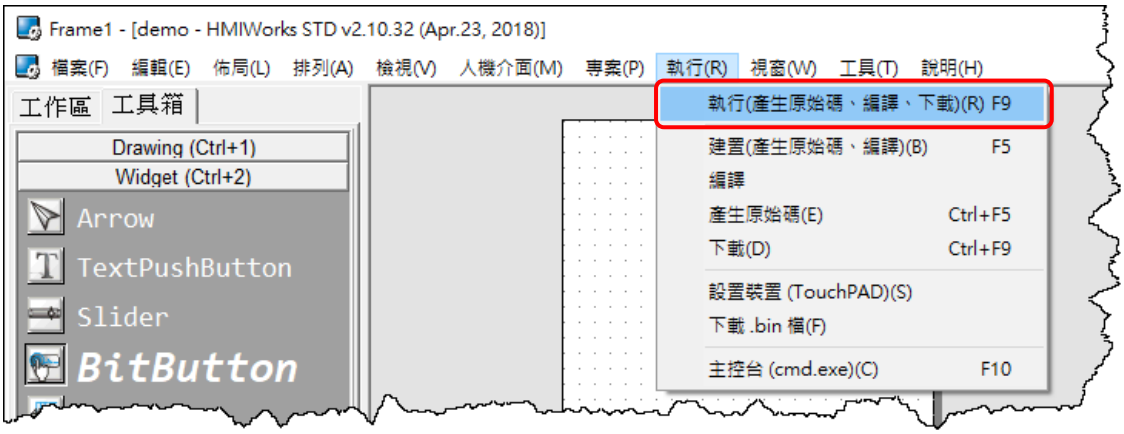
此範例中，我們使用 TPD-280-H 透過 RS-485 與電腦主機連接在一起，並將 TouchPAD 上**旋轉開關(Rotary Switch)** 切換至 **“1” (更新模式)** 的位置，再將 TouchPAD 斷電重新啟動。

點選 **“執行(R)” → “設置裝置(TouchPAD)(S)”** 來選擇正確的 COM Port 配置。

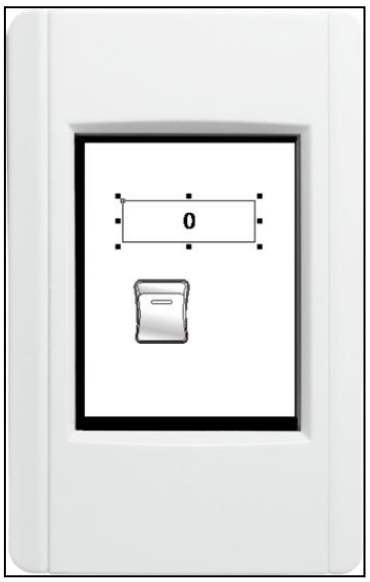


步驟 5 建置、下載、執行

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。
一旦載入完成，請將 TouchPAD 上的**旋轉開關(Rotary Switch)** 調回 **“0” (執行模式)** 的位置，再將 TouchPAD 斷電關機。



再將 TouchPAD 供電開機後的執行畫面如下，碰觸按鈕圖示會改變文字顯示框的值，從 0 → 1 或 1 → 0。



4.3 TouchPAD 與 I/O 模組整合

本節將詳細介紹如何來存取我司泓格的 M-7000 (Modbus RTU)、I-7000 (DCON) 及 PET-7000 (Modbus TCP) 遠端 I/O 模組。

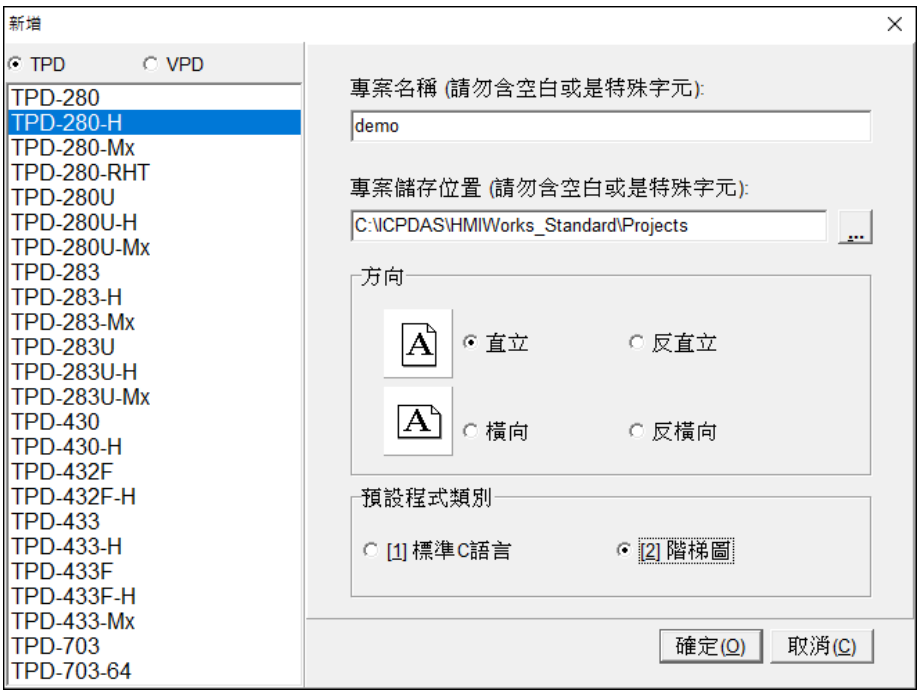
如果您的從站設備是非本公司的 Modbus RTU/TCP 裝置，請參考下面 FAQ 連結來配置。
FAQ: 如何使用 TouchPAD 來存取非本公司 (泓格) 的 Modbus RTU 從站設備?
FAQ: 如何使用 TouchPAD 來存取非本公司 (泓格) 的 Modbus TCP 從站設備?

4.3.1 使用 TouchPAD 來存取 M-7000

本例中，將以 TPD-280-H 來控制泓格的 M-7060 模組 (Modbus RTU I/O 設備)，它具有 4 通道數位輸入及 4 通道 Relay 輸出。首先，設定好 M-7060 的站號 (Net ID)、Baud Rate、Data Bit、Parity 及 Stop Bit…等資訊，再將 M-7060 使用 RS-485 接線連接至 TPD-280-H。

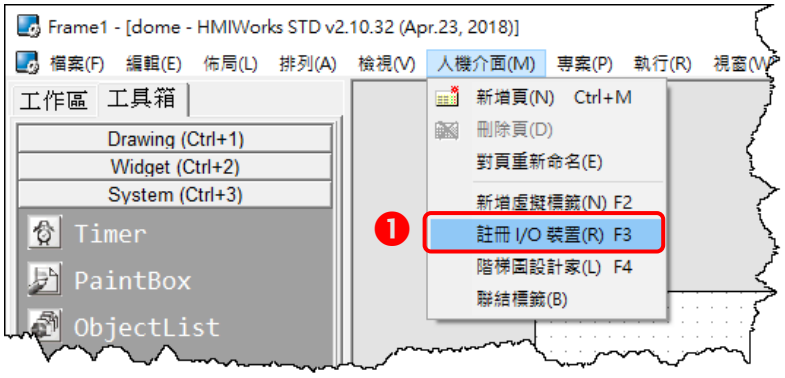
步驟 1 建立一個新的專案

從 **“檔案(F)”** 功能選單中，點選 **“新增…(N)”** 項目來開啟 “新增”視窗。接著在開新專案的視窗中選擇型號、決定專案名稱、專案儲存位置、方向及預設程式類別。



步驟 2 配置裝置（I/O）標籤

1. 從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(R)”** 項目 (或按 **<F3>** 鍵) 來開啟 “裝置” 視窗註冊 M-7060 模組。



2. 從 **“TouchPAD 作為”** 下拉式選單中，選擇 **“Modbus RTU Master”** 項目。

3. 從 **“裝置系列”** 下拉式選單中，選擇 **“M-7000”** 項目。

4. 從 **“連線”** 下拉式選單中，選擇 **“新增...”** 項目來開啟 “新增/編輯連線...” 視窗，接著設定連線至您的 Modbus RTU 設備資訊，如下：

① 在 **“連線名稱”** 欄位，輸入**連線名稱** (預設 SER_1)。

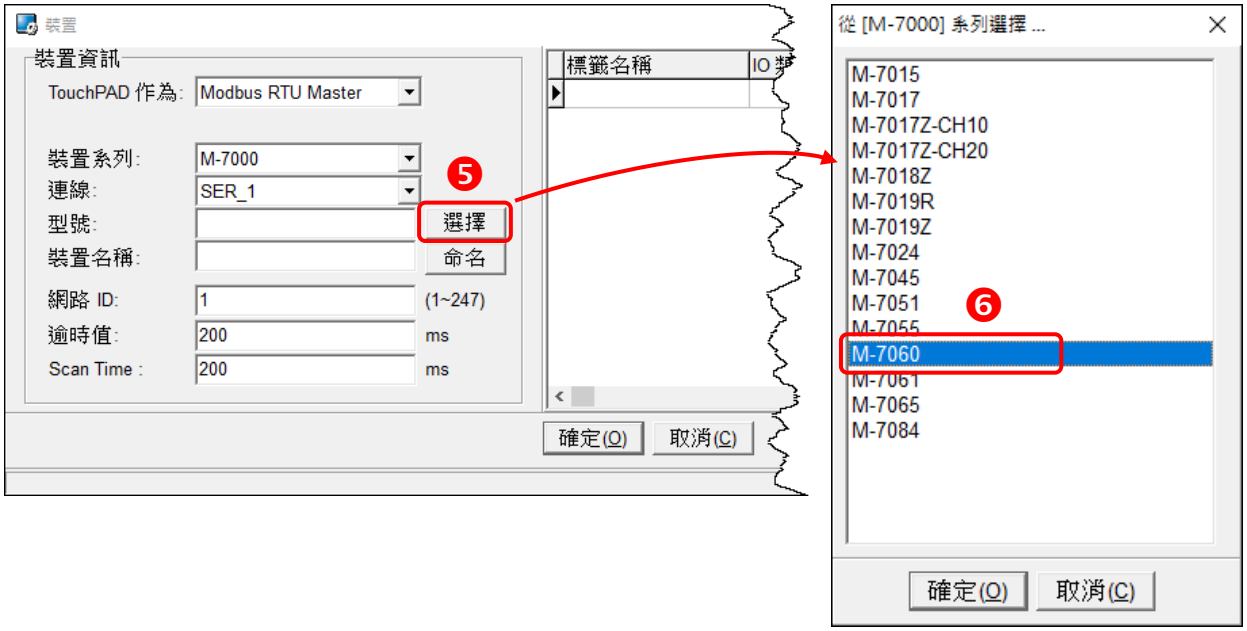
② 在 **“連線介面”** 下拉式選單中，選擇 **“COM1”** 項目。

③ 在 **“鮑率 (Baud Rate)”**、**“資料位元 (Data Bit)”**、**“校驗位元 (Parity)”**、 **“停止位元 (Stop Bit)”** 欄位輸入 **M-7060 的 Baud Rate 及 Data Format**。

④ 單擊 **“確定(O)”** 按鈕來完成建立連線。

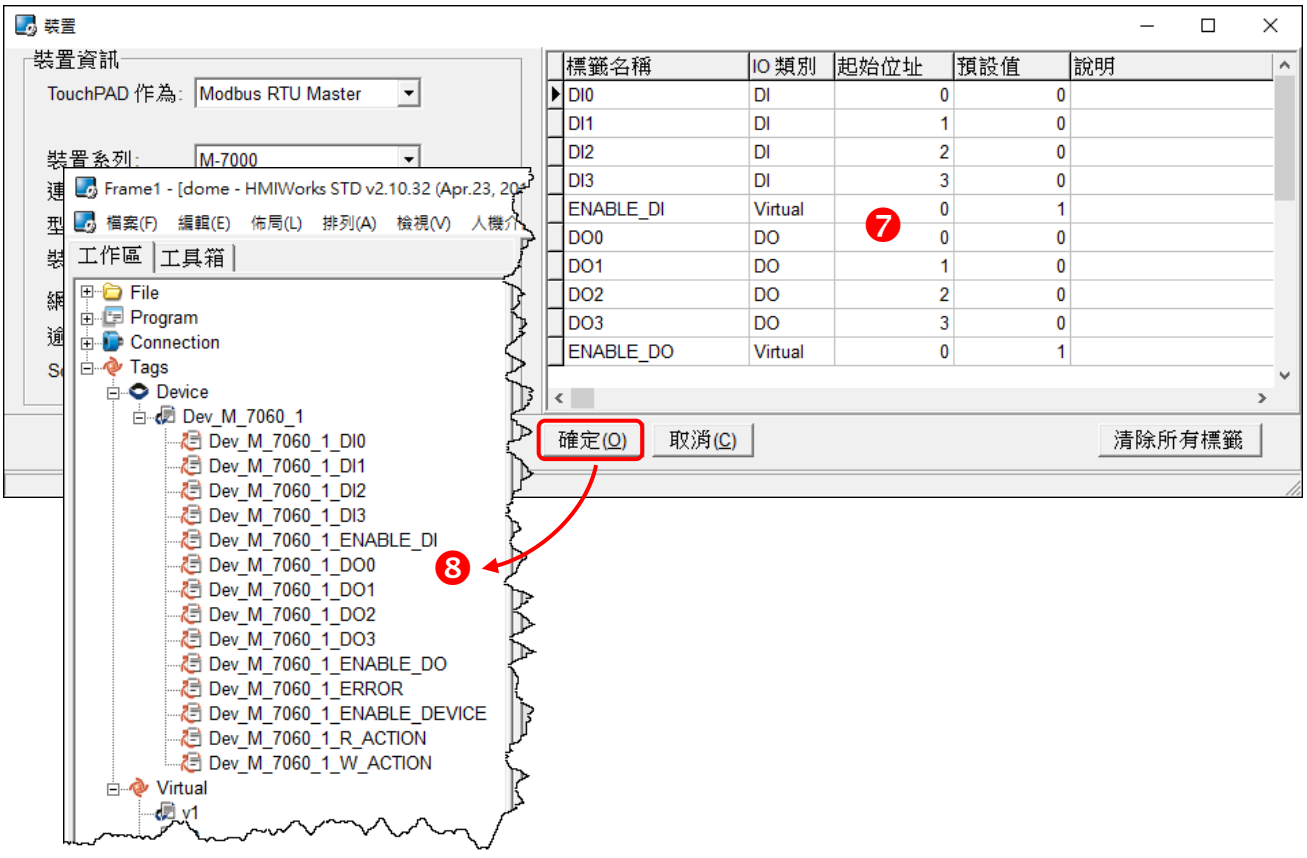
5. 按下 **“選擇”** 按鈕來開啟選型視窗。

6. 在選型視窗中，選擇 M-7060 模組，再按 **“確定(O)”** 按鈕。



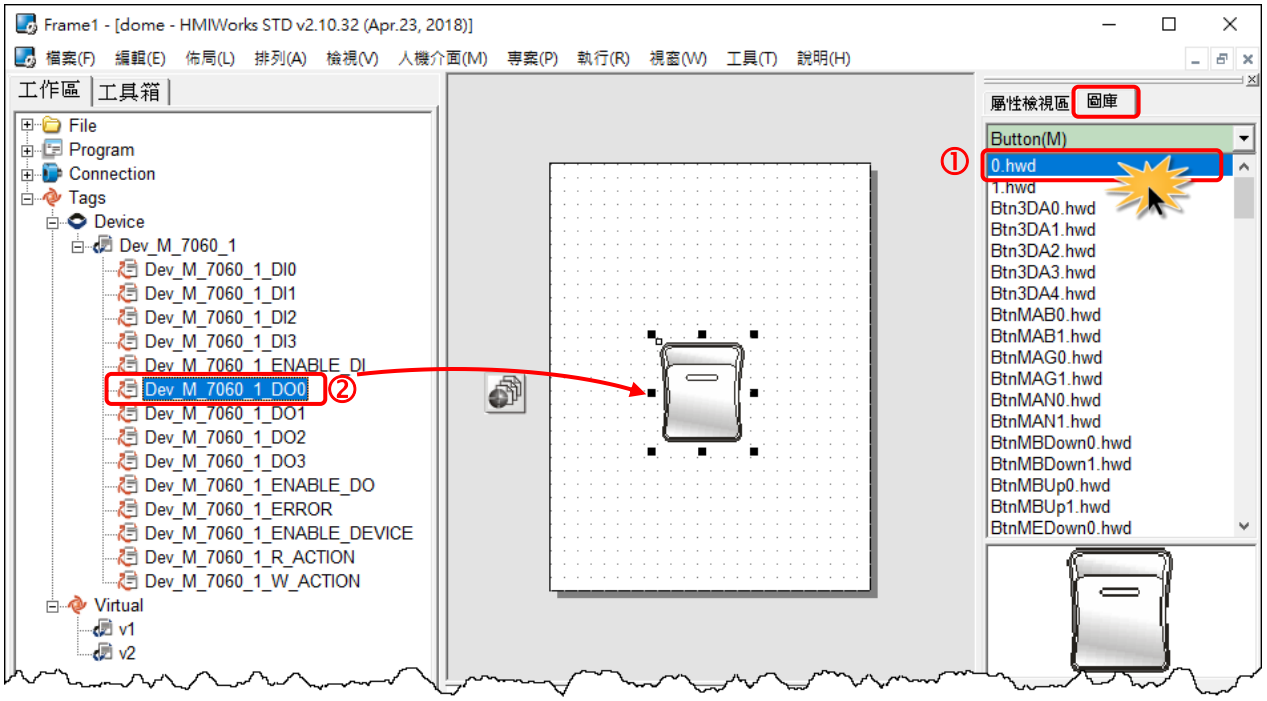
7. 將顯示 M-7060 的詳細資訊 (如: 裝置名稱、網路 ID、標籤名稱、IO 類別、起始位址及預設值...等)，再按下 **“確定(O)”**。

8. 在 **“工作區”** → **“Device”** 下確認 “Dev_M-7060_1” 裝置標籤建立完成。



步驟 3 設計階梯圖

在 **“圖庫”** 區選擇要代表 DO0 輸出的物件圖。在 **“工作區”** 點選 **“Dev_M_7060_1_DO0”** 項目，並拖移放至顯示頁設計區，此時將顯示 DO0 按鈕圖示。

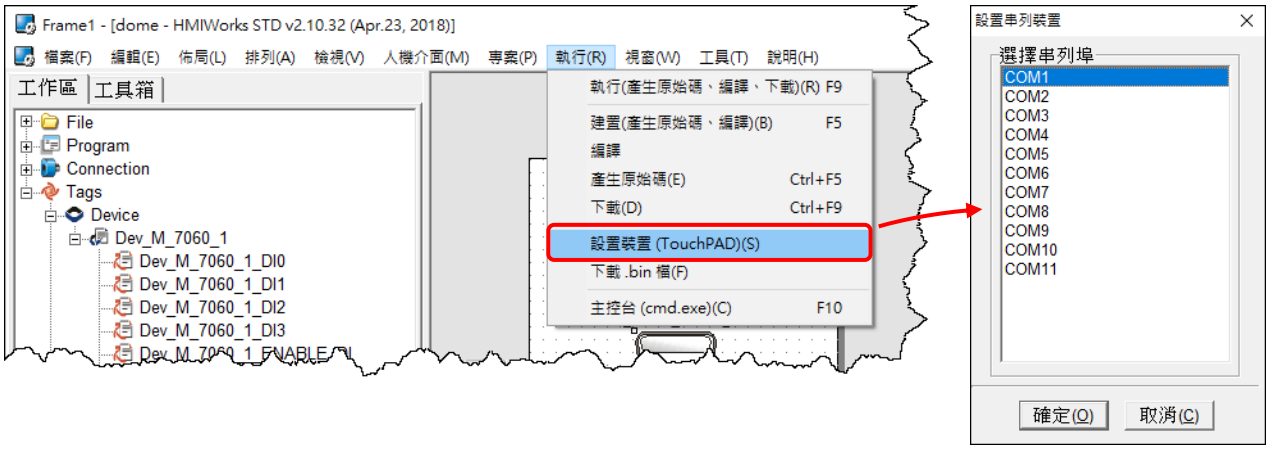


步驟 4 設置裝置（TouchPAD）

設置裝置的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

本例中，TPD-280-H 通過 RS-485 接線連接至電腦主機，並將 TouchPAD 模組上旋轉開關切換到 **"位置 1 (更新模式)”**，然後斷電重新啟動 TouchPAD。

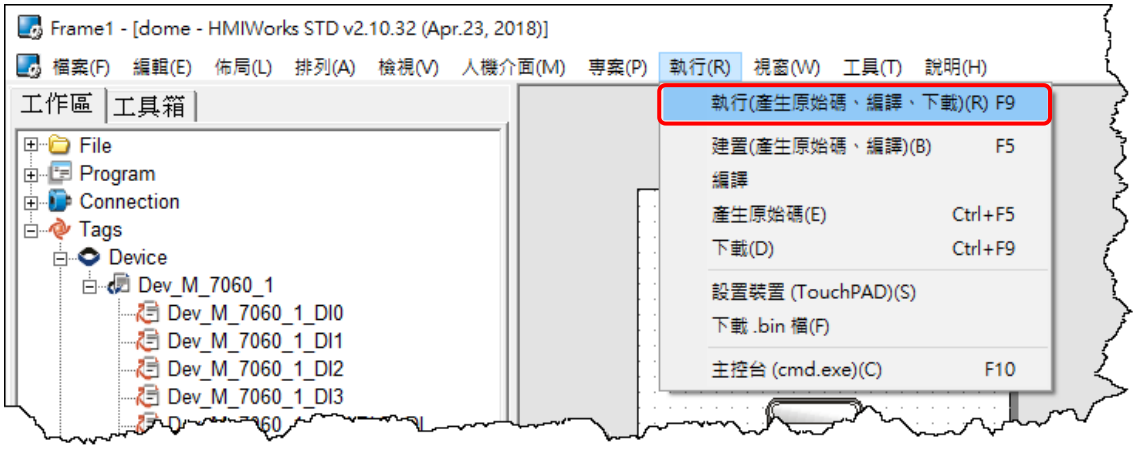
從 **“執行(R)”** 功能選單中，點選 **“設置裝置(TouchPAD)(S)”** 項目來選擇正確的 COM Port。



步驟 5 建置、下載、執行

載入程式的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。
一旦載入完成，請將 TouchPAD 上的**旋轉開關(Rotary Switch)** 調回 **“0” (執行模式)** 的位置，再將 TouchPAD 斷電關機。



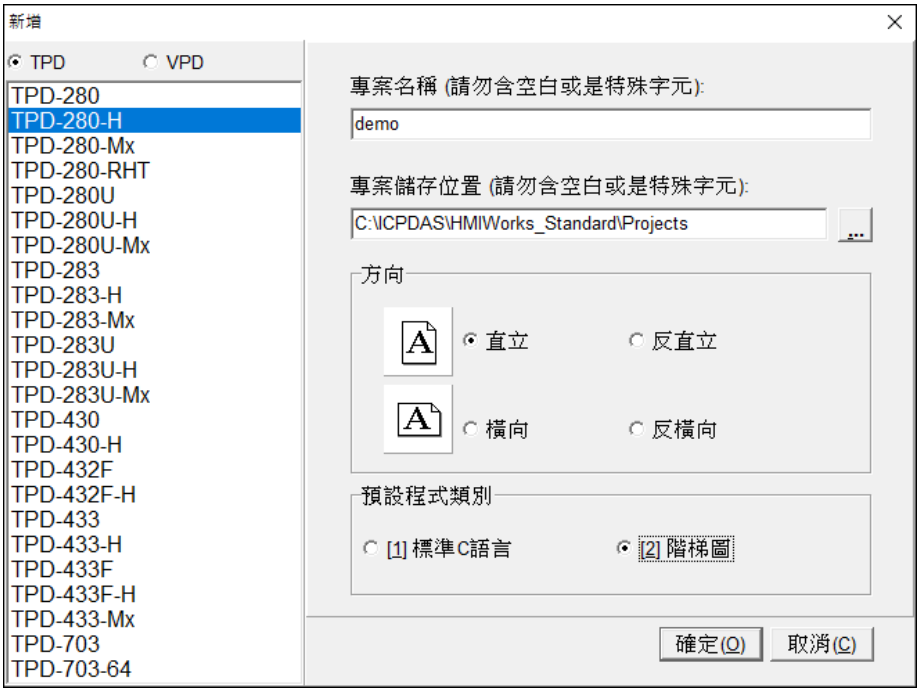
再將 TouchPAD 供電開機後的執行畫面如下，碰觸按鈕圖示就可以改變 M-7060 該通道的輸出。

4.3.2 使用 TouchPAD 來存取 I-7000

本例中，我們將以 TPD-280-H 來控制泓格的 I-7066 (DCON I/O 設備)，它具有 7 通道光耦合電晶體型繼電器輸出。首先，設定好 I-7066 的站號 (Net ID)、Baud Rate、Data Bit、Parity 及 Stop Bit…等資訊，再將 I-7066 使用 RS-485 接線連接至 TPD-280-H。

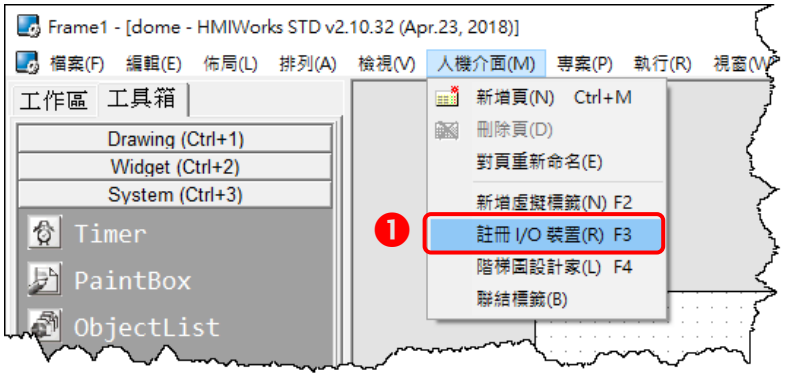
步驟 1 建立一個新的專案

從 **“檔案(F)”** 功能選單中，點選 **“新增…(N)”** 項目來開啟 “新增”視窗。接著在開新專案的視窗中選擇型號、決定專案名稱、專案儲存位置、方向及預設程式類別。



步驟 2 配置裝置（I/O）標籤

1. 從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(R)”** 項目 (或按 **<F3>** 鍵) 來開啟 “裝置” 視窗註冊 I-7066 模組。



2. 從 **“TouchPAD 作為”** 下拉式選單中，選擇 **“DCON Master”** 項目。

3. 從 **“裝置系列”** 下拉式選單中，選擇 **“I-7000”** 項目。

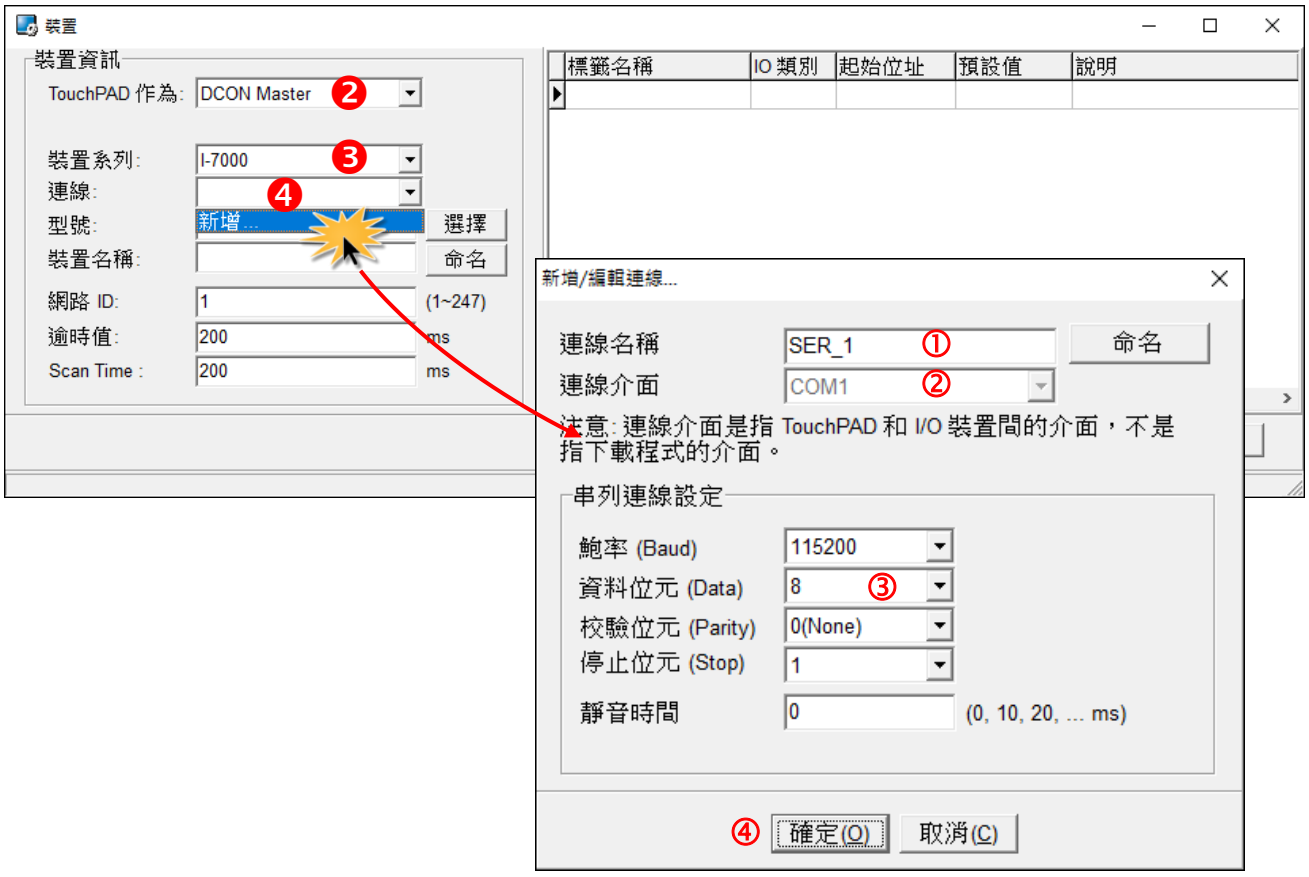
4. 從 **“連線”** 下拉式選單中，選擇 **“新增...”** 項目來開啟 “新增/編輯連線...” 視窗，接著設定連線至您的 DCON I/O 設備資訊，如下：

① 在 **“連線名稱”** 欄位，輸入**連線名稱** (預設 SER_1)。

② 在 **“連線介面”** 下拉式選單中，選擇 **“COM1”** 項目。

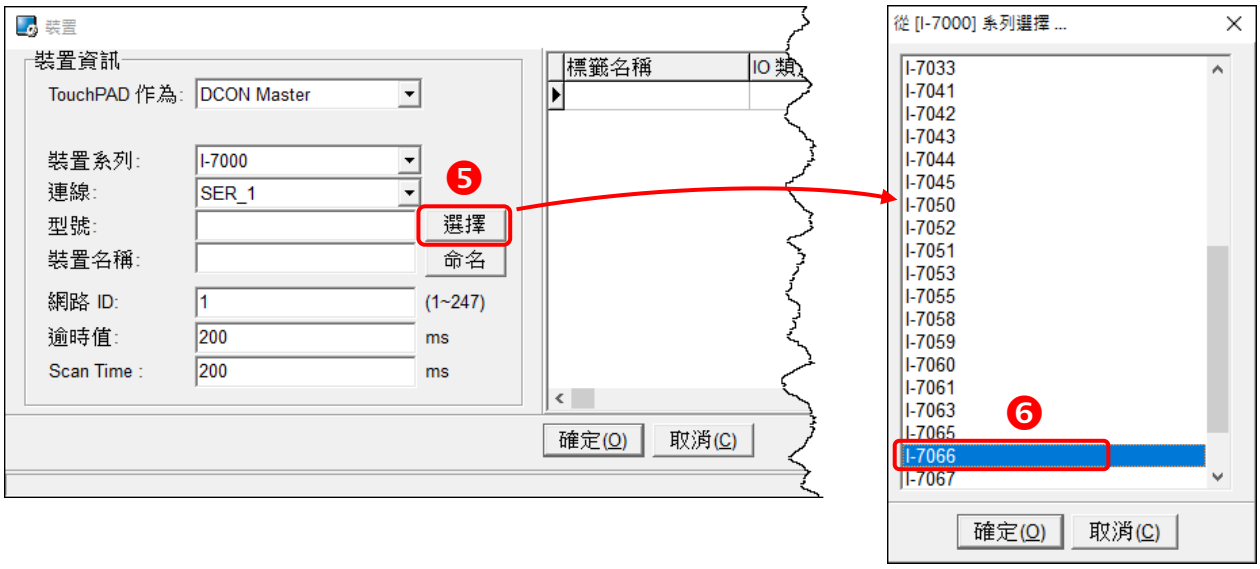
③ 在 **“鮑率 (Baud Rate)”**、**“資料位元 (Data Bit)”**、**“校驗位元 (Parity)”**、 **“停止位元 (Stop Bit)”** 欄位輸入 **I-7066 的 Baud Rate 及 Data Format**。

④ 單擊 **“確定(O)”** 按鈕來完成建立連線。



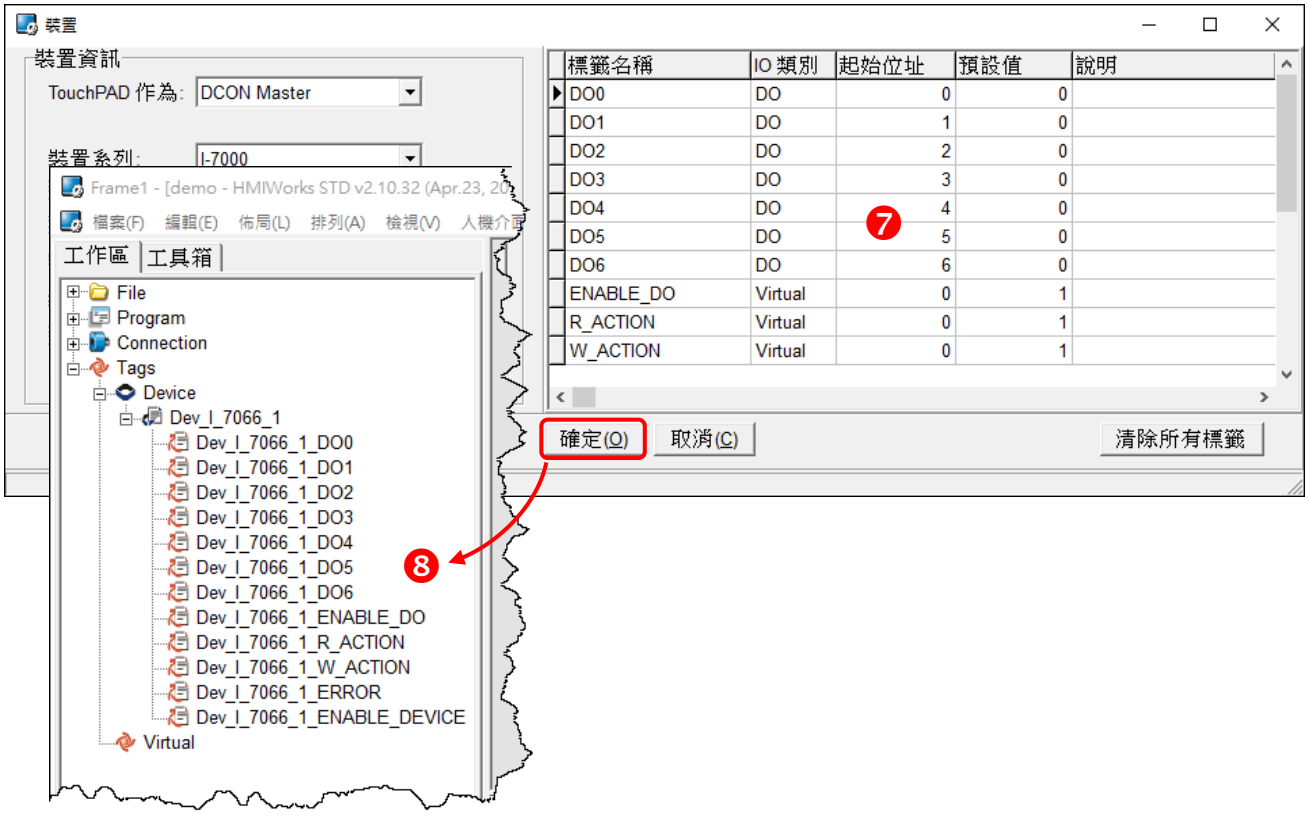
5. 按下 **“選擇”** 按鈕來開啟選型視窗。

6. 在選型視窗中，選擇 I-7066 模組，再按 **“確定(O)”** 按鈕。



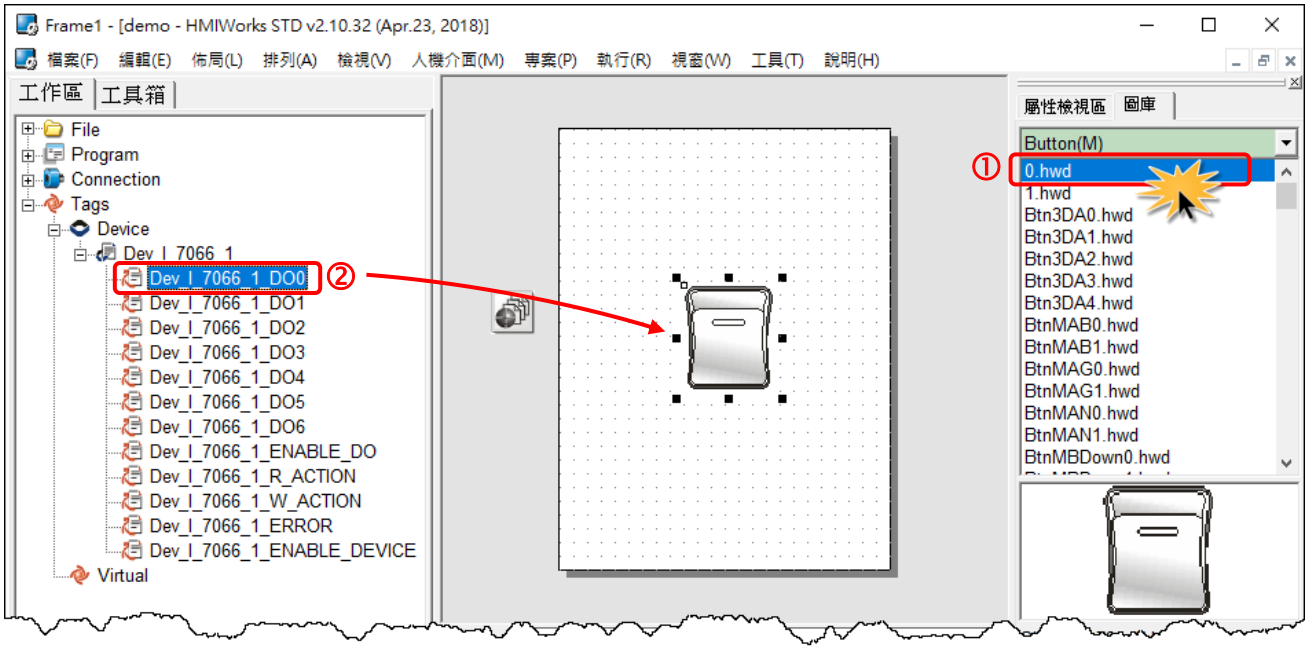
7. 將顯示 I-7066 的詳細資訊 (如: 裝置名稱、網路 ID、標籤名稱、IO 類別、起始位址及預 設值...等)，再按下 **“確定(O)”**。

8. 在 **“工作區”** → **“Device”** 下確認 “Dev_I-7066_1” 裝置標籤建立完成。



步驟 3 設計階梯圖

在 **“圖庫”** 區選擇要代表 DO0 輸出的物件圖。在 **“工作區”** 點選 **“Dev_I_7066_1_DO0”** 項目，並拖移放至顯示頁設計區，此時將顯示 DO0 按鈕圖示。

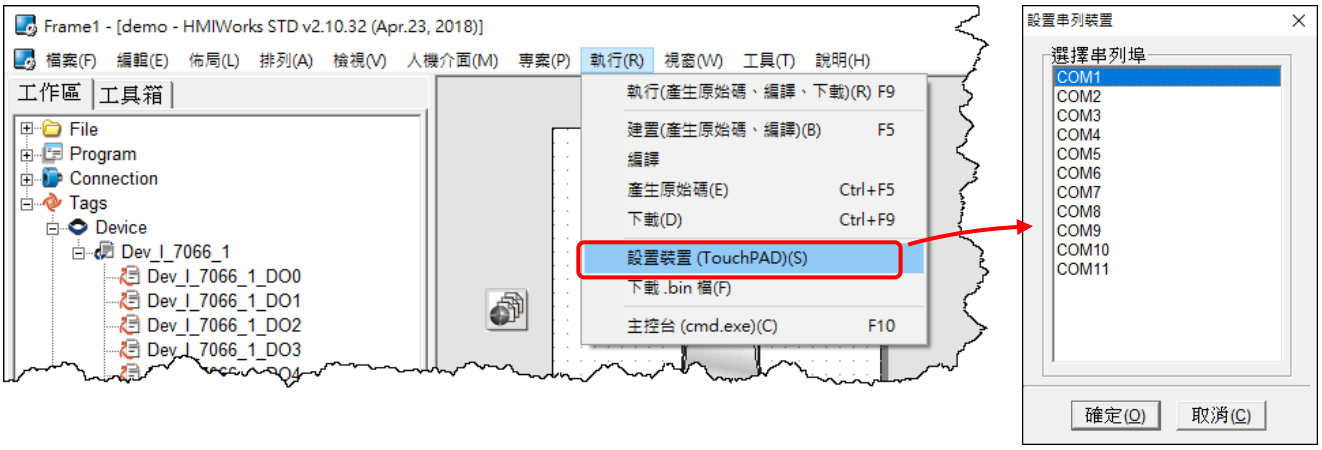


步驟 4 設置裝置（TouchPAD）

設置裝置的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

本例中，TPD-280-H 通過 RS-485 接線連接至電腦主機，並將 TouchPAD 模組上旋轉開關切換到 **"位置 1 (更新模式)”**，然後斷電重新啟動 TouchPAD。

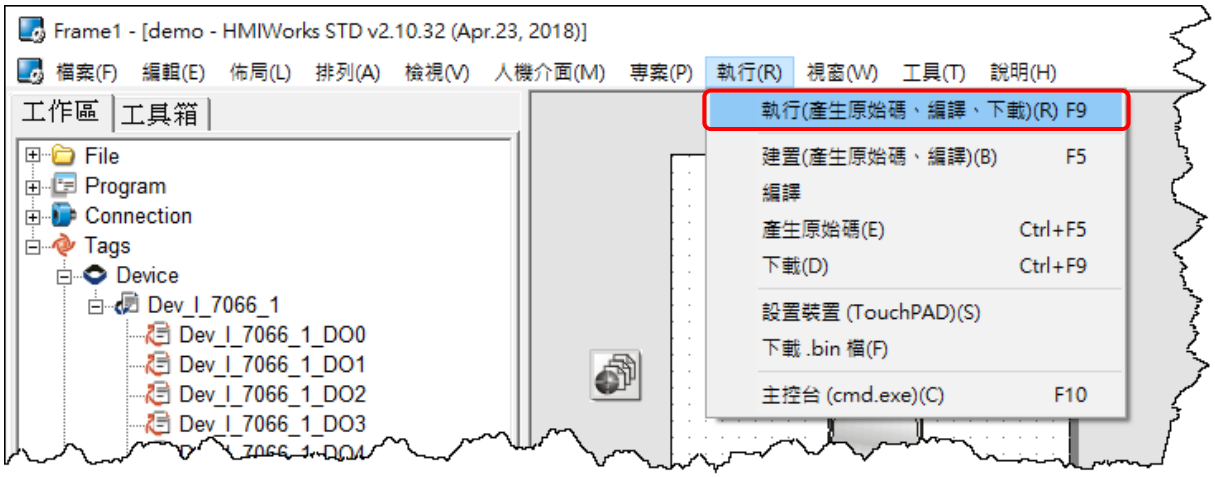
從 **“執行(R)”** 功能選單中，點選 **“設置裝置(TouchPAD)(S)”** 項目來選擇正確的 COM Port。



步驟 5 建置、下載、執行

載入程式的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。
一旦載入完成，請將 TouchPAD 上的**旋轉開關(Rotary Switch)** 調回 **“0” (執行模式)** 的位置，再將 TouchPAD 斷電關機。



再將 TouchPAD 供電開機後的執行畫面如下，碰觸按鈕圖示就可以改變 I-7066 該通道的輸出。

4.3.3 使用 TouchPAD 來存取 PET-7000

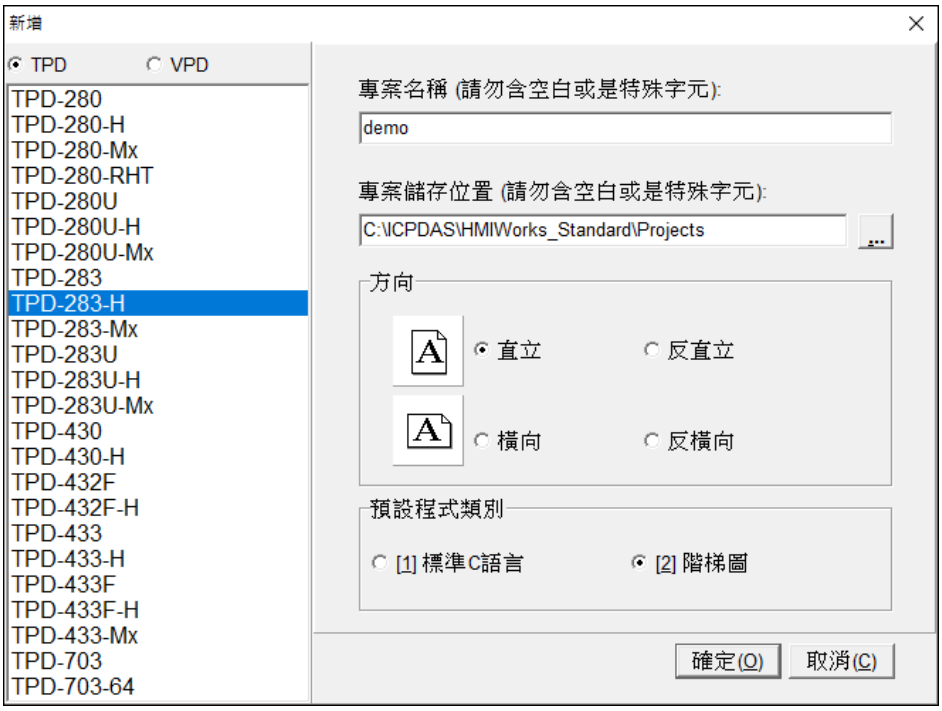
本例中，我們將以 TPD-283-H 來控制泓格的 PET-7060 (Modbus TCP I/O 設備)，它具有 6 通道數位輸入及 6 通道 Relay 輸出。首先，將 PET-7060 及 TPD-283-H 與您的電腦連接至同一個集線線器 (Hub) 或同一個子網域。

步驟 1 配置 PET-7060

確認您電腦的網路設定正確且可運作，然後將 PET-7060 模組供電開機，並配置正確有效的網路設定。詳細接線、網路配置資訊，請參考 PET-7060 快速入門指南。

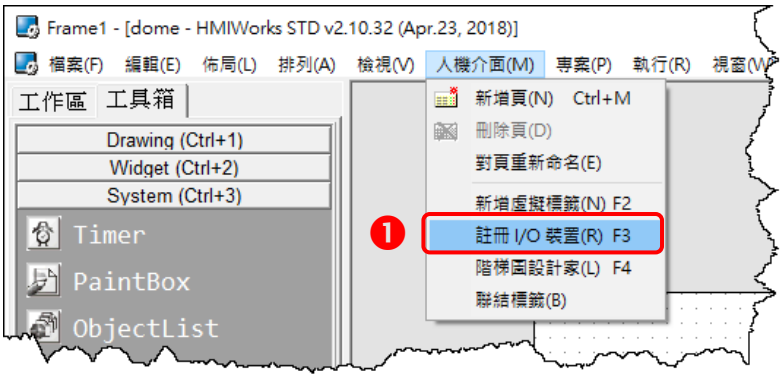
步驟 2 建立一個新的專案

從 **“檔案(F)”** 功能選單中，點選 **“新增...(N)”** 項目來開啟 “新增”視窗。接著在開新專案的視窗中選擇型號、決定專案名稱、專案儲存位置、方向及預設程式類別。



步驟 3 配置裝置（I/O）標籤

1. 從 **“人機介面(M)”** 功能選單中，點選 **“註冊 I/O 裝置(R)”** 項目 (或按 **<F3>** 鍵) 來開啟 “裝置” 視窗註冊 PET-7060 模組。

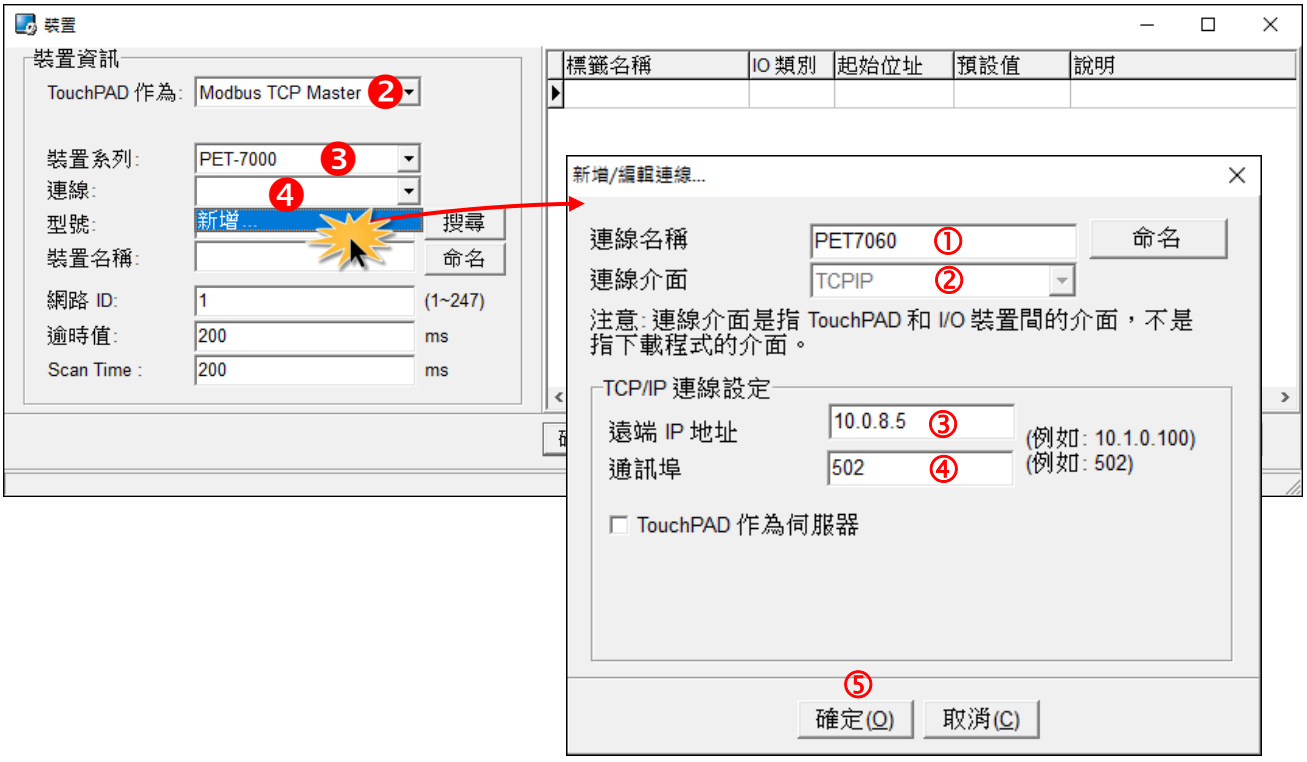


2. 從 **“TouchPAD 作為”** 下拉式選單中，選擇 **“Modbus TCP Master”** 項目。

3. 從 **“裝置系列”** 下拉式選單中，選擇 **“PET-7000”** 項目。

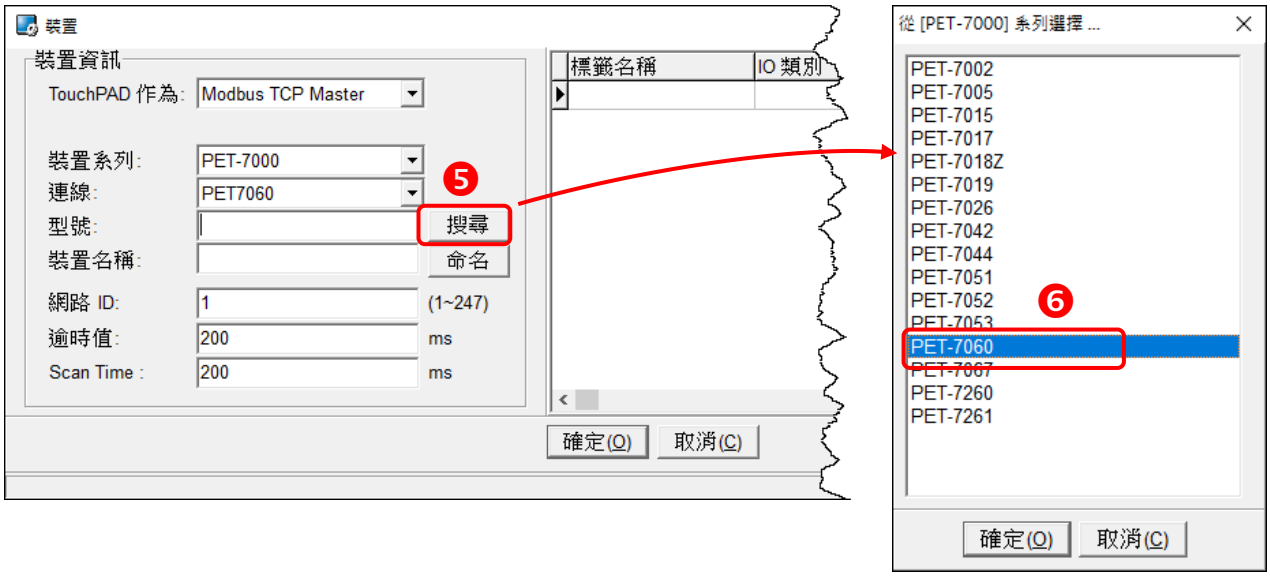
4. 從 **“連線”** 下拉式選單中，選擇 **“新增...”** 項目來開啟 “新增/編輯連線...” 視窗，接著設定連線至您的 Modbus TCP Master 設備資訊，如下：

① 在 **“連線名稱”** 欄位，輸入**連線名稱** (如: PET7060)。
② 在 **“連線介面”** 下拉式選單中，選擇 **“TCPIP”** 項目。
③ 在 “IP Address” 欄位，輸入 **PET-7060 的 IP Address**。
④ 在 “Port” 欄位，輸入 **PET-7060 的 TCP Port** 。
⑤ 單擊 **“確定(O)”** 按鈕來完成建立 PET-7060 連線。



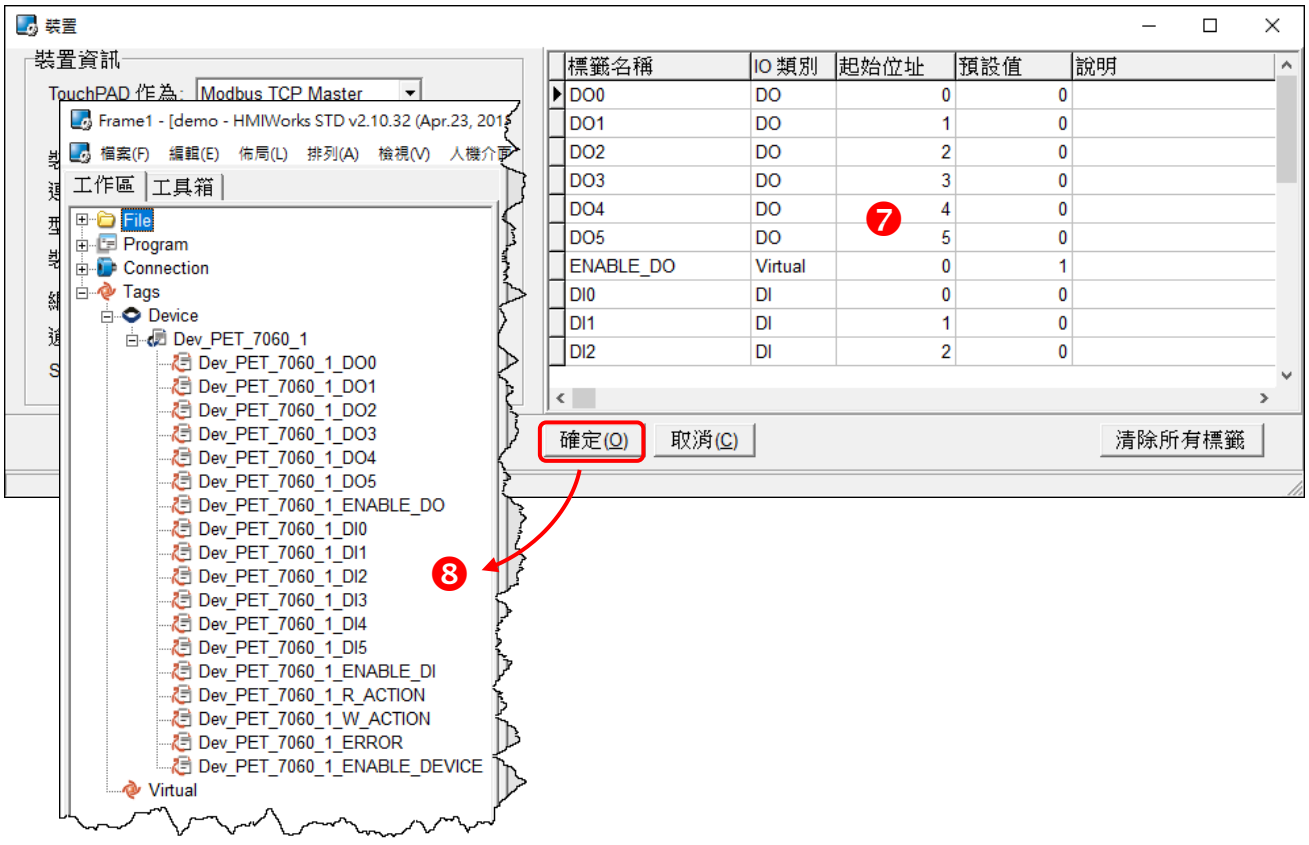
5. 按下 **“選擇”** 按鈕來開啟選型視窗。

6. 在選型視窗中，選擇 M-7060 模組，再按 **“確定(O)”** 按鈕。



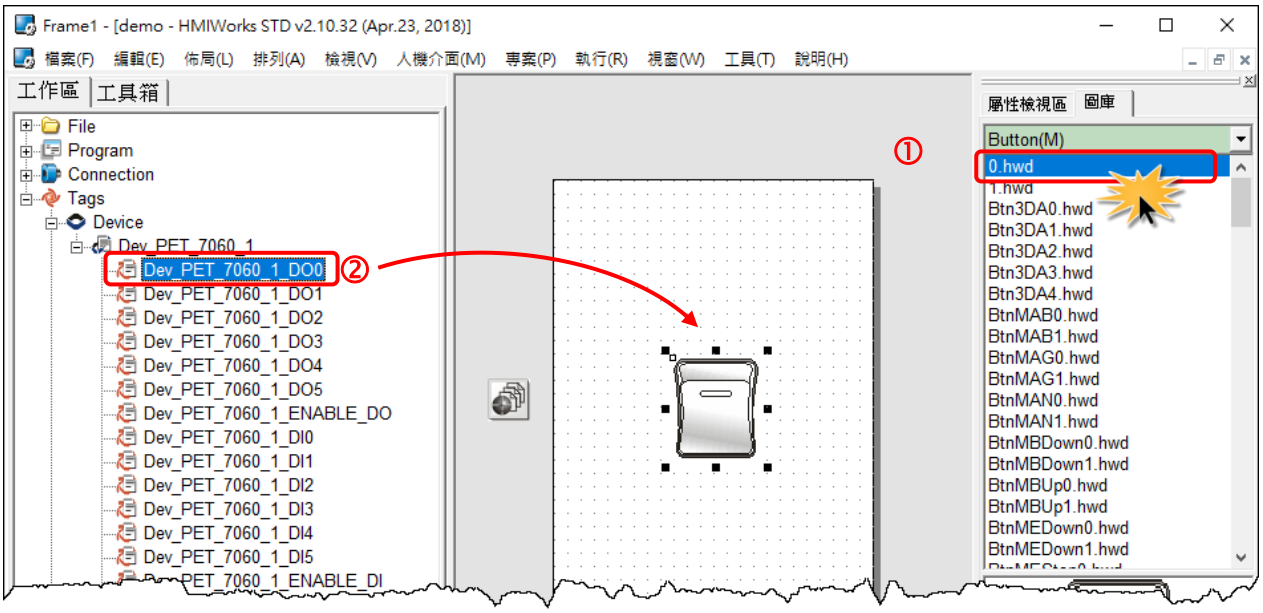
7. 將顯示 PET-7060 的詳細資訊 (如: 裝置名稱、網路 ID、標籤名稱、IO 類別、起始位址及預 設值…等)，再按下 **“確定(O)”**。

8. 在 **“工作區”** → **“Device”** 下確認 “Dev_PET-7060_1” 裝置標籤建立完成。



步驟 4 設計階梯圖

在 **“圖庫”** 區選擇要代表 DO0 輸出的物件圖。在 **“工作區”** 點選 **“Dev_PET_7060_1_DO0”** 項目，並拖移放至顯示頁設計區，此時將顯示 DO0 按鈕圖示。

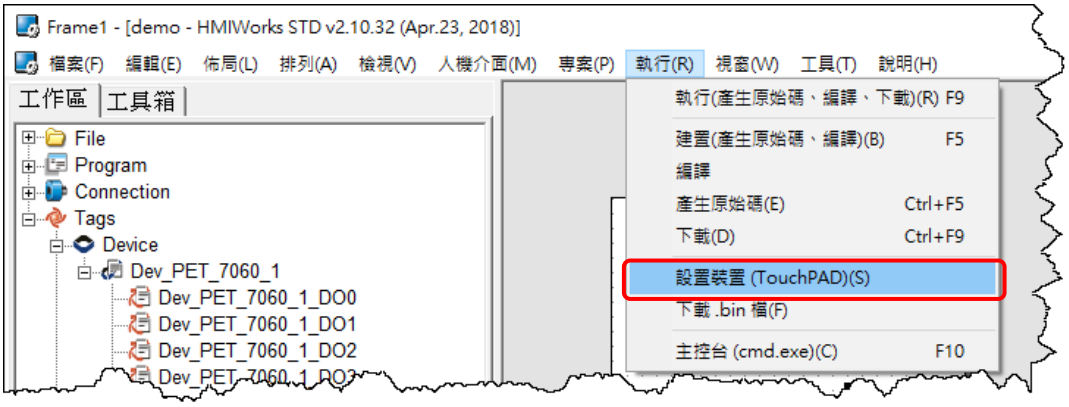


步驟 5 設置裝置（TouchPAD）

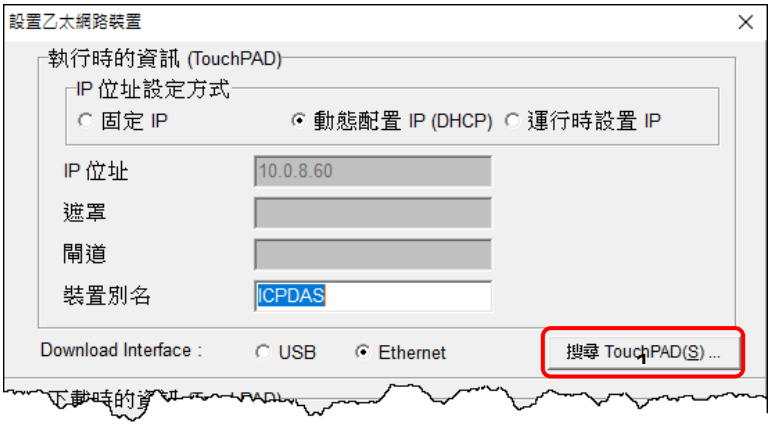
設置裝置的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

本例中，TPD-283-H 與電腦連接至同一個集線線器 (Hub) 或同一個子網域，然後將 TouchPAD 模組上旋轉開關切換到 **"位置 0 (執行/更新模式)”**，再斷電重新啟動 TouchPAD。

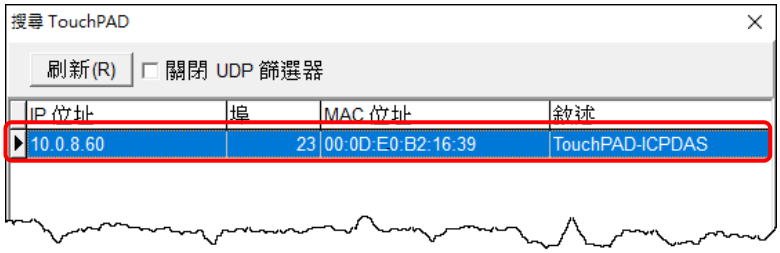
1. 從 **“執行(R)”** 功能選單中，點選 **“設置裝置(TouchPAD)(S)”** 項目來配置正確的運作 IP 位址及下載資訊。



2. 在“設置乙太網路裝置”視窗，單擊 **“搜尋 TouchPAD(S)…”** 按鈕來開啟“搜尋 TouchPAD”視窗。



3. 搜尋完成後，在尋找到的 TouchPAD 列表單中，**依據您 TouchPAD 的 MAC 位址來點選想要的 IP 位址**，再單擊 **“確定(O)”** 按鈕。



⚠**注意:** 您可以在 TouchPAD 的背面找到其 MAC 位址。.

4. 在“IP 位址設定方式”區，**選擇 “固定 IP”、“動態配置 IP (DHCP)” 或 “運行時設置 IP”**。此範例: 動態配置 IP (DHCP)。(此設定只適用於 TouchPAD 運作時期。)

項目	說明
固定 IP	由 HMIWorks 中的設定來配置 TouchPAD 的 IP 位址，並將其存儲在程式的映像檔內。
動態配置 IP (DHCP)	由 DHCP server 動態分配 TouchPAD 的 IP 位址。請確認環境中有 DHCP server。
運行時設置 IP	程式檔案本身不帶有 IP 設定值，在執行時，TouchPAD 才會從快閃記憶體(flash)中載入 IP 設定值。務必記得在使用該選項來設定 IP 之前，一定要先用相關的函式把 IP 設定值寫入快閃記憶體中，目前亦提供相關的範例程式來做這件事。

⚠**注意:** 當更換「固定 IP」、「動態配置 IP」和「運行時設置 IP」等不同的操作模式，或者是更換不同的「固定 IP」設定時，都需要重新下載程式後才能使用新的設定值。

5. 確認 **“下載時的資訊 (TouchPAD)”** 區域中的 **IP** 位址與 **“主機資訊 (電腦)”** 區域中的 **主機 IP 位址**是設定在同一個子網域內。(此設定只適用於下載程式時。)

6. 確認 **“下載時的資訊 (TouchPAD)”** 區域中的 **“TouchPAD 硬體位址 (MAC)”** 與您的 TouchPAD 的 MAC 位址相符合，再單擊 **“確定(O)”** 按鈕。

注意:

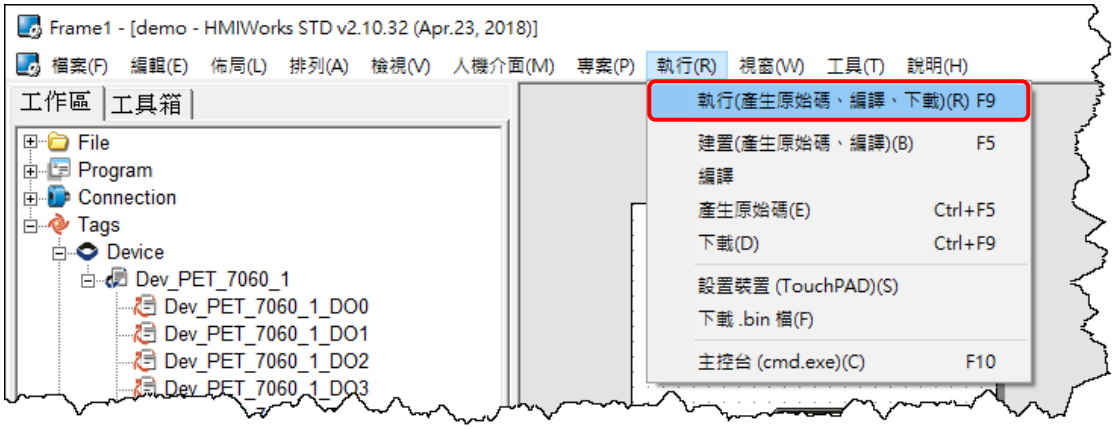
1. 您可以在 TouchPAD 的背面找到其 MAC 位址。

2. 當 TouchPAD 為 Force Update 時，搜尋顯示的 IP 位址為 0.0.0.0 是正常的。您只需另指定一個合法的 IP 位址供其下載時使用即可。

步驟 6 建置、下載、執行

載入程式的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。



一旦載入完成，執行畫面如下，碰觸按鈕圖示就可以改變 PET-7060 該通道的輸出。

4.4 TCP/IP 通訊

TouchPAD 使用者可開發適用於 TCP/IP 通訊的客制化應用。本章節將使用 TouchPAD 建立 TCP Client 及 TCP Server，詳細說明如下。

4.4.1 如何使 TouchPAD 作為 TCP Client?

本例中，使用 PC 作為 TCP Server 接收從 TouchPAD (TCP Client) 傳送出來的訊息，詳細步驟如下:

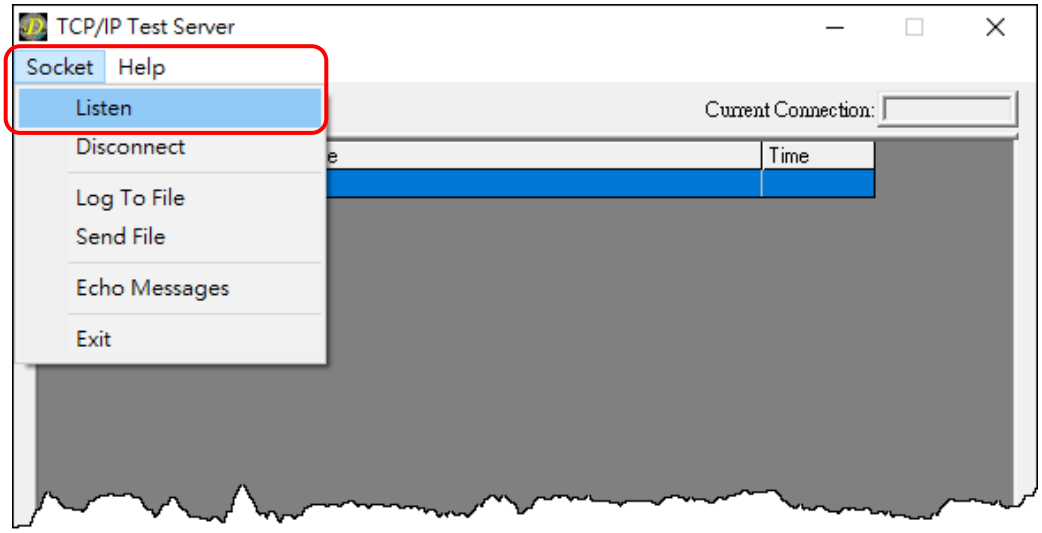
步驟 1 配置您的電腦為 TCP Server

⚠**注意:** 確認您 PC 的 Windows 防火牆以及 Anti-Virus 防火牆都已關閉，或已正確的設定，否則 TcpipEcho.exe 可能無法正確執行。

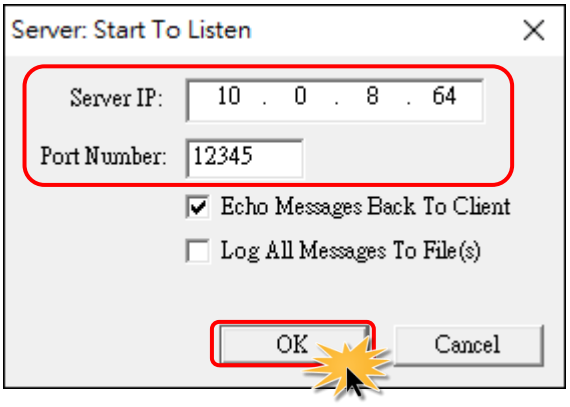
1. 安裝 **TcpIpEcho.exe (TCP/IP Test Server 程式)** 至您的電腦。TcpIpEcho.exe 下載位置如下:

   http://www.brothersoft.com/tcp-ip-test-server-27898.html

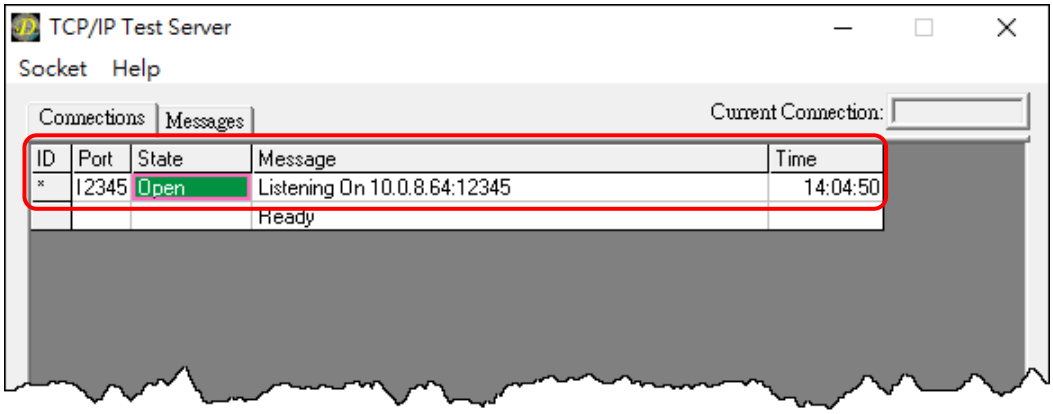
2. 安裝完成後，執行 TCPIPEcho.exe 程式。在功能表上單擊 **“Socket” → “Listen”** 來開啟 “Server: Start To Listen” 配置視窗。



3. 在“**Server IP**”及“**Port Number**”欄位輸入 **TCP Server 的 IP 位址**及 **TCP Port** (範例: PC 的 IP “10.0.8.64” 及 Port “12345”)。
4. 勾選 **“Echo Messages Back To Client”** 項目。
5. 單擊 **“OK”** 按鈕後，將開始監聽 TCP Server 的 IP/Port 組合。



6. 這將顯示在 TCP/IP Test Server 配置視窗中的 **“Open”** 行列。



步驟 2 配置 TouchPAD 為 TCP Client

1. 下載並解壓縮 TCP/IP 範例程式，可從泓格的網站中下載，下載位置如下:

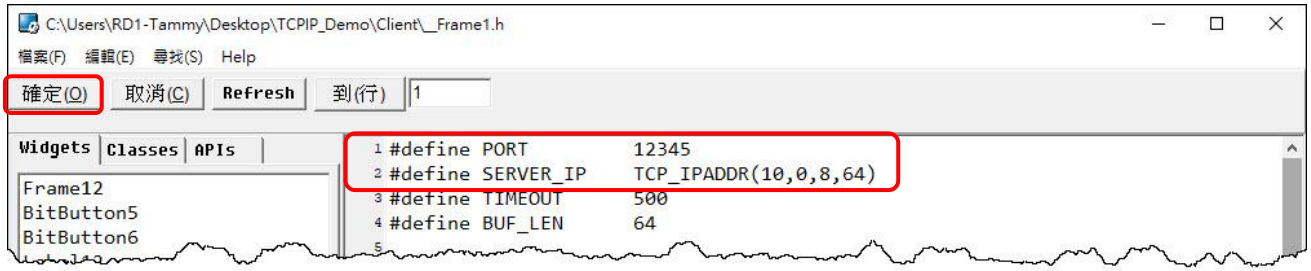
https://www.icpdas.com/en/download/show.php?num=1000

2. 執行 HMIWorks Standard 軟體並開啟現有的 **“client.hxp”** 專案。

⚠ **注意:** 檢查您的 HMIWorks 版本是否為 v2.10.40 或更新版本。如果您的 HMIWorks 版本是低於 v2.10.40，請將 HMIWorks 更新到最新版本。

3. 雙擊 **“Start”** BitButton 以編輯 **“OnClick (點擊事件)”** 函數。

4. 在 define PORT 及 define SERVER_IP 行列指定 **TCP Server 的 IP 位址及 TCP Port** (範例: PC 的 IP “10.0.8.64” 及 Port “12345”)，再按下 **“確定(O)”** 鈕來儲存離開。



下面程式範例為建立 TCP Client 連線，包含了打開連線，如: **hmi_TCPNew()** 及 **hmi_TCPOpen()**、 發送及接收訊息，如: **hmi_TCPSendCmdEx()** 及關閉連線，如: **hmi_TCPClose()**。詳細關於 TCP API 更多說明，請參考 [<< HMIWorks API Reference>>] 。

```
#define PORT            12345
#define SERVER_IP       TCP_IPADDR(10,0,8,64)
#define TIMEOUT         500
#define BUF_LEN         64

static tHandle h = INVALID_HANDLE;    // handle for TCP Communications
int index = 0;                  // timer execution count
int missingCount = 0;       // the missing count (receiving error count)

void BitButton5OnClick(tWidget *pWidget) // Start
{
    if ( h > INVALID_HANDLE ) return; // already have a connection

    // Allocate a session. Check if h < 0 to prevent using another tHandle
    h = hmi_TCPNew();           //Allocate a TCP session if possible

    if (h > INVALID_HANDLE)    // if allocating a new session successfully
    {
        LabelTextSet(&Label4, "Connecting");
        //used in a client to establish a TCP session for connecting to a server.
        hmi_TCPOpen(h, SERVER_IP, PORT, PORT);
    }
    else
        h = INVALID_HANDLE; // don't keep error code in h
}

void BitButton6OnClick(tWidget *pWidget)      // Stop
{
    if (h > INVALID_HANDLE)
    {
        hmi_TCPClose(h);    //closes and deallocates a TCP session.
        h = INVALID_HANDLE;
        LabelTextSet(&Label4, "OFF");
    }
}
```



```
void Timer10OnExecute(tWidget *pWidget)                    //Connection status
{
    if (hmi_TCPState(h) == STATE_TCP_CONNECTED)      //gets the state of the TCP session.
    {
        LabelTextSet(&Label9, "CONNECTED");
        LabelTextSet(&Label4, "ON");
    }
    else
    {
        LabelTextSet(&Label9, "DISCONNECTED");
    }
}

void Timer11OnExecute(tWidget *pWidget)              //Send data
{
    static unsigned char send_buf[BUF_LEN];
    static unsigned char recv_buf[BUF_LEN];

    if ( h == INVALID_HANDLE ) return; // not ready yet

    index++;

    if (hmi_TCPState(h) == STATE_TCP_CONNECTED)
    {
        usprintf((char*)send_buf, "DATA%05d", index);
        //sends data and then receives data through a TCP session.
        hmi_TCPSendCmdEx(h, send_buf, BUF_LEN, recv_buf, BUF_LEN, TIMEOUT);
        recv_buf[BUF_LEN -1] = 0;          // null-terminated
        LabelTextSet(&Label12, (char*)send_buf);
    }
}

void TextPushButton15OnClick(tWidget *pWidget)
{
    static unsigned char send_buf[BUF_LEN];
    static unsigned char recv_buf[BUF_LEN];

    if ( h == INVALID_HANDLE ) return; // not ready yet

    index++;

    if (hmi_TCPState(h) == STATE_TCP_CONNECTED)
    {
        usprintf((char*)send_buf, "DATA%05d", index);
        //sends data and then receives data through a TCP session.
        hmi_TCPSendCmdEx(h, send_buf, BUF_LEN, recv_buf, BUF_LEN, TIMEOUT);
        recv_buf[BUF_LEN -1] = 0;          // null-terminated
        LabelTextSet(&Label12, (char*)send_buf);
    }
}
```

5. 設置裝置 (TouchPAD)。

設置裝置的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

本例中， 使用 TPD-433-H 通過 USB 接線連接至電腦主機，並將 TouchPAD 模組上旋轉開關切換到 **“位置 9 (USB 更新模式)”** ，然後斷電重新啟動 TouchPAD。

從 **“執行(R)”** 功能選單中，點選 **“設置裝置(TouchPAD)(S)”** 項目來配置網路設定 (如: DHCP) 並選擇 USB 下載方式。

6. 建置、下載、執行。

載入程式的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。

一旦載入完成，請將 TouchPAD 上的**旋轉開關 (Rotary Switch)** 調回 **“0” (執行模式)** 的位置，再將 TouchPAD 斷電關機。

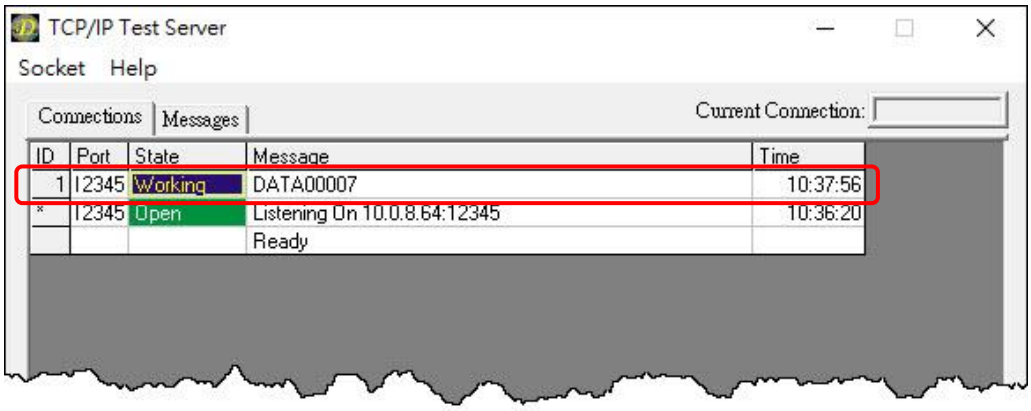
7. 查看 TouchPAD 上將顯示 **“Client”** 程式畫面。

步驟 3 TCP 應用程式測試

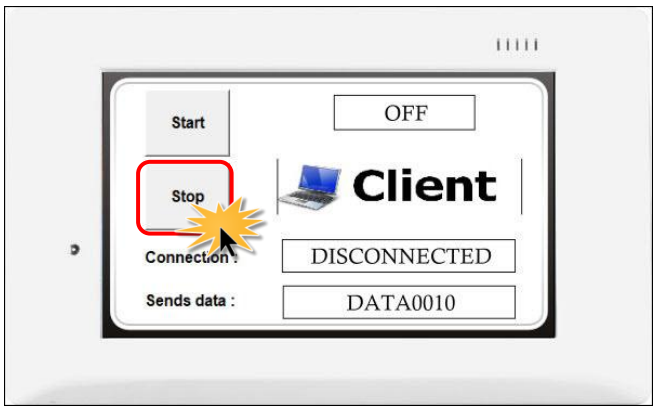
1. 碰觸 TouchPAD 上 **“Start”** 按鈕圖示來開始連線至 TCP Server (如: PC)。

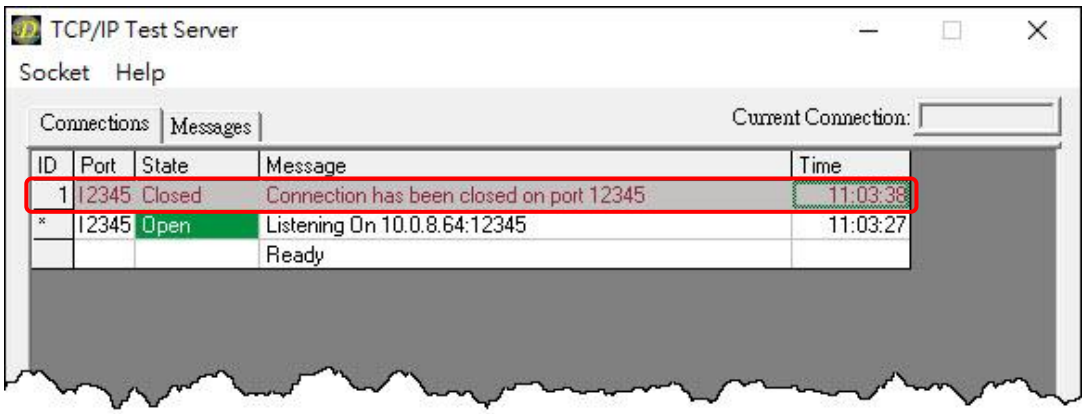
2. 確認 TouchPAD 上 **“CONNECTED”** 欄位中顯示 **“Connection”** 已連接狀態及 **“Sends data”** 欄位中有顯示發送訊息。

3. 確認 TCP/IP Test Server 配置視窗中 **“Working”** 行列將顯示接收到 TCP Clinet (如: TouchPAD) 所發送的訊息。



4. 碰觸 TouchPAD 上 **“Stop”** 按鈕圖示來關閉連線。





4.4.2 如何使 TouchPAD 作為 TCP Server?

本例中，使用 TouchPAD #1 作為 TCP Server 接收從 TouchPAD #2 (TCP Client) 傳送出來的訊息，詳細步驟如下:

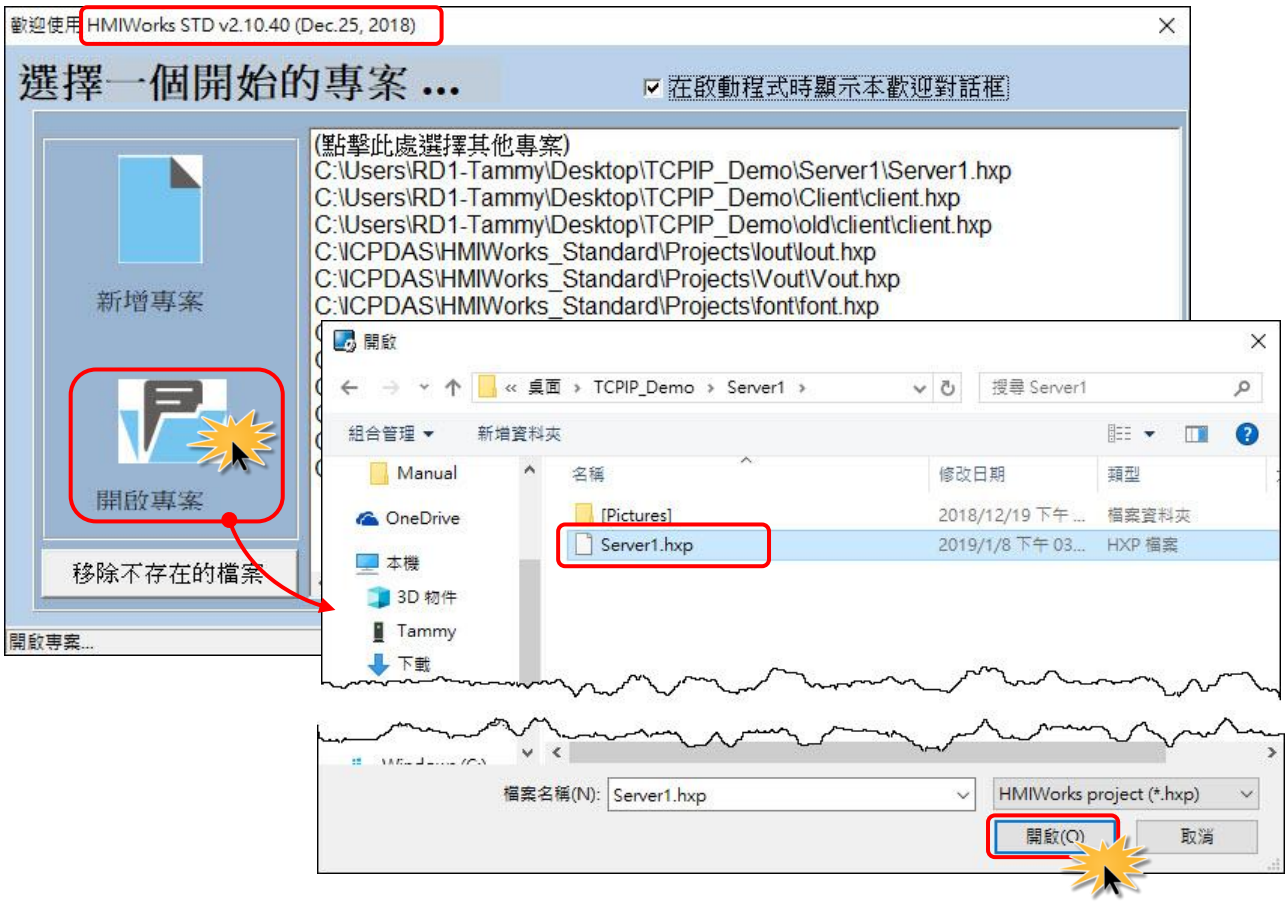
步驟 1 配置 TouchPAD #1 為 TCP Server

1. 下載並解壓縮 TCP/IP 範例程式，可從泓格的網站中下載，下載位置如下:

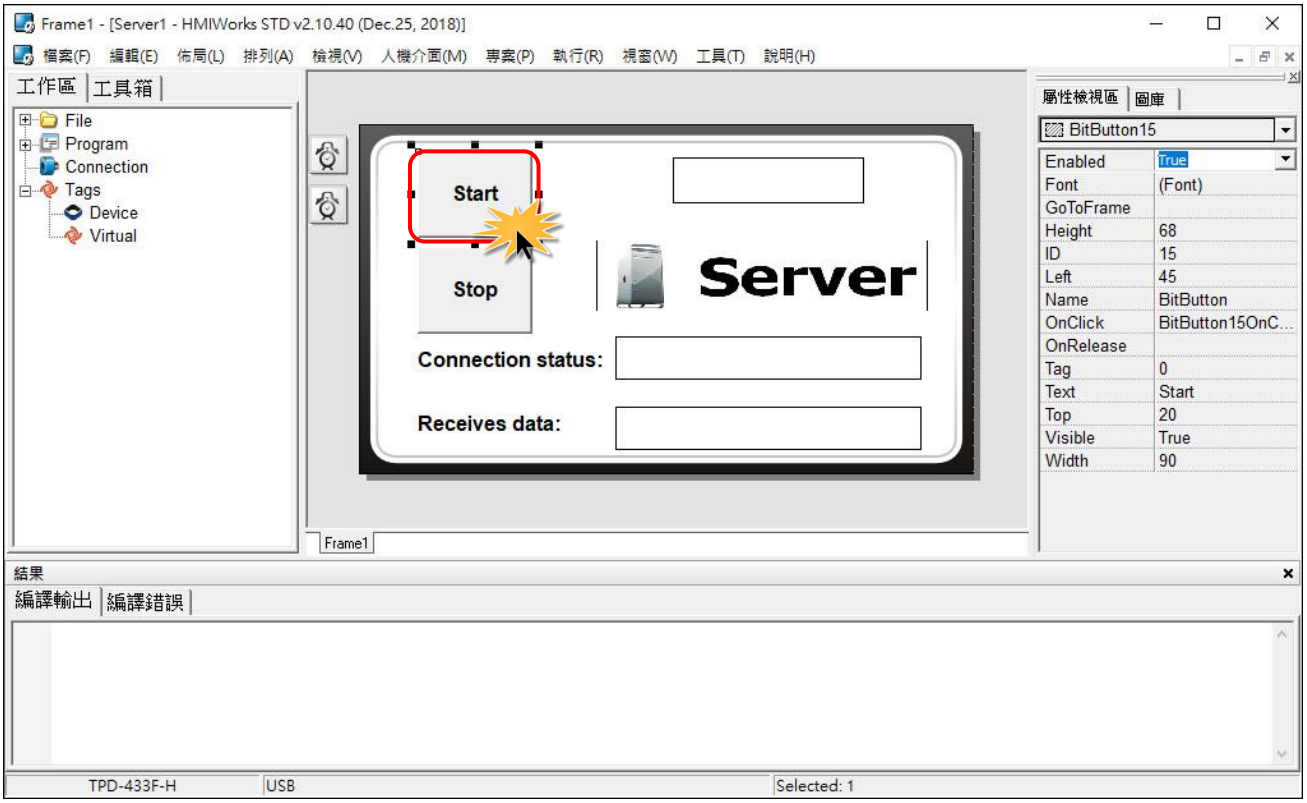
https://www.icpdas.com/en/download/show.php?num=1000

2. 執行 HMIWorks Standard 軟體並開啟現有的 **"Server1.hxp"** 專案。

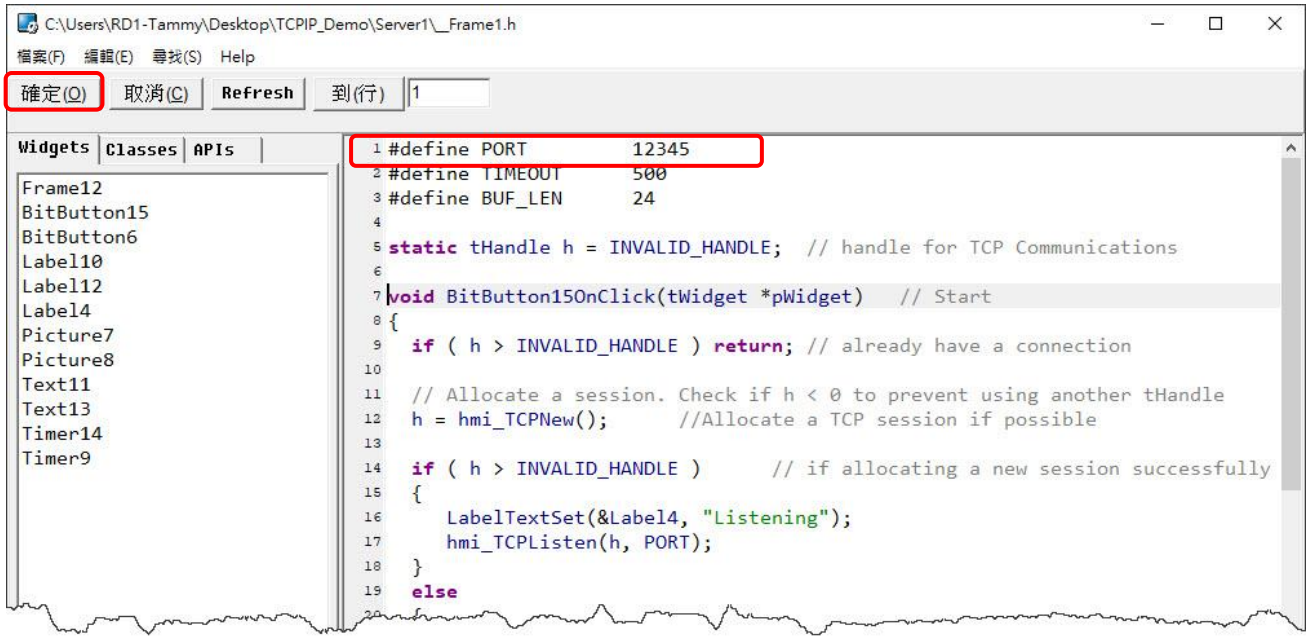
⚠ **注意:** 檢查您的 HMIWorks 版本是否為 v2.10.40 或更新版本。如果您的 HMIWorks 版本是低於 v2.10.40，那麼請將 HMIWorks 更新到最新版本。



3. 雙擊 **“Start”** BitButton 以編輯 **“OnClick (點擊事件)”** 函數。



4. 在 define PORT 行列指定 **TCP Server 的 TCP Port** (範例: Port “12345”)，再按下 **“確定(O)”** 鈕來儲存離開。



下面程式範例為建立 TCP Server 連線，包含了打開連線，如: **hmi_TCPNew()** 及 **hmi_TCPListen()**、發送及接收訊息，如: **hmi_TCPOutput()** 及 **hmi_TCPReadEx()** 及關閉連線，如: **hmi_TCPClose()**。 詳細關於 TCP API 更多說明，請參考 <<HMIWorks API Reference>> 。

```
#define PORT            12345
#define TIMEOUT         500
#define BUF_LEN         24

static tHandle h = INVALID_HANDLE;    // handle for TCP Communications

void BitButton15OnClick(tWidget *pWidget)      // Start
{
   if ( h > INVALID_HANDLE ) return; // already have a connection

   // Allocate a session. Check if h < 0 to prevent using another tHandle
   h = hmi_TCPNew();          //Allocate a TCP session if possible

   if ( h > INVALID_HANDLE )          // if allocating a new session successfully
   {
        LabelTextSet(&Label4, "Listening");
        hmi_TCPListen(h, PORT);
   }
   else
   {
        static char szMsg[20];
        usprintf(szMsg, "Err= %d", h);
        LabelTextSet(&Label4, szMsg);
        h = INVALID_HANDLE; // don't keep the error code in h
   }
}

void BitButton6OnClick(tWidget *pWidget)          // Stop
{
   if (h >= 0)
   {
      hmi_TCPClose(h);    //closes and deallocates a TCP session.
      h = -1;
      LabelTextSet(&Label4, "OFF");
   }
}

void Timer14OnExecute(tWidget *pWidget)            //Connection status
{
    if (hmi_TCPState(h) == STATE_TCP_LISTEN)                    //gets the state of the TCP
session.
    {
        LabelTextSet(&Label10, "LISTENED");
    }
    if (hmi_TCPState(h) == STATE_TCP_CONNECTED)
    {
        LabelTextSet(&Label10, "CONNECTED");
    }
    else
```

```
        {
            LabelTextSet(&Label10, "NO CONNECTED");
        }
    }

    void Timer9OnExecute(tWidget *pWidget) //Receiver data
    {
        static unsigned char recv_buf[64];
        int ret = 0;

        if ( h == INVALID_HANDLE ) return; // server does not ready

        if (hmi_TCPState(h) == STATE_TCP_CONNECTED) // client connected
        {
            //reads data through a TCP session.
            ret = hmi_TCPReadEx(h, recv_buf, BUF_LEN, TIMEOUT);
              LabelTextSet(&Label12, (char*)recv_buf);
                if (ret > 0)
                {
              //write back to the session immediately (no waiting in the queue).
                    hmi_TCPOutput(h, recv_buf, ret);
            }
        }
    }
```

5. 設置裝置 (TouchPAD)。

設置裝置的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

本例中， 使用 TPD-433-H 通過 USB 接線連接至電腦主機，並將 TouchPAD 模組上旋轉開關切換到 **“位置 9 (USB 更新模式)”** ，然後斷電重新啟動 TouchPAD。

從 **“執行(R)”** 功能選單中，點選 **“設置裝置(TouchPAD)(S)”** 項目來開啟 “設置乙太網路裝置” 視窗。

配置網路設定並選擇 USB 下載方式，然後按下 **“確定(O)”** 鈕來儲存離開。

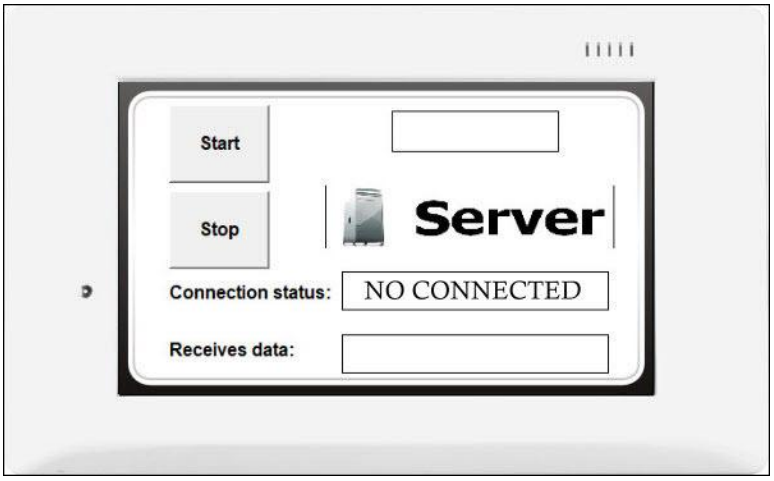
6. 建置、下載、執行。

載入程式的方式必需根據 TouchPAD 的類型將有所不同。更多更詳細的資訊請參考 TouchPAD 硬體使用手冊 第 3.4 節 程式載入至 TouchPAD。

點選 **“執行(R)” → “執行(產生原始碼、編譯、下載)(R)”** 項目 (或按 **<F9>** 鍵)。

一旦載入完成，請將 TouchPAD 上的**旋轉開關 (Rotary Switch)** 調回 **“0” (執行模式)** 的位置，再將 TouchPAD 斷電關機。

2. 查看 TouchPAD 上將顯示 **“Server1”** 程式畫面。



步驟 2 配置 TouchPAD #2 為 TCP Client

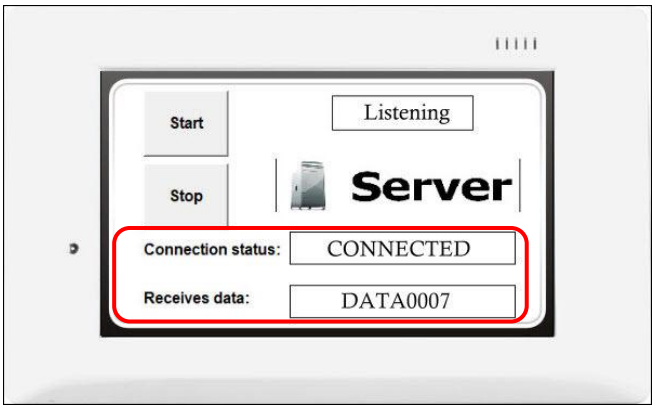
詳細執行步驟請參考 第 4.4.1 節 如何使 TouchPAD 作為 TCP Client 的**步驟 2**。

步驟 3 TCP 應用程式測試

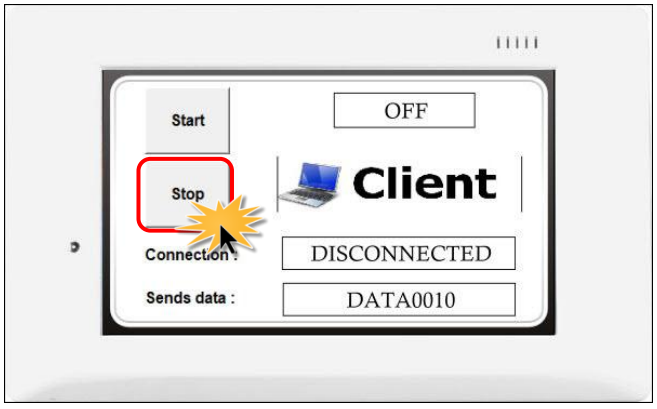
1. 碰觸 TouchPAD #1 (Server) 上 **“Start”** 按鈕圖示來開始監聽。
2. 碰觸 TouchPAD #2 (Clinet) 上 **“Start”** 按鈕圖示來開始連線至 TCP Server。

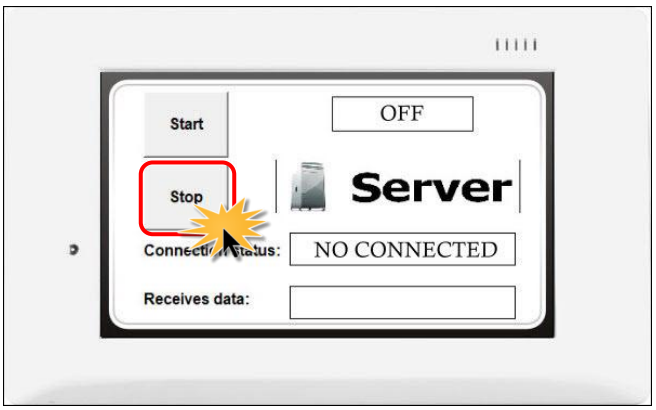
3. 確認 TouchPAD #2 (Client) 上 **“CONNECTED”** 欄位中顯示 **“Connection”** 已連接狀態及 **“Sends data”** 欄位中有顯示發送訊息。

4. 確認 TouchPAD #1 (Server) 上 **“CONNECTED”** 欄位中顯示 **“Connection”** 已連接狀態及 **“Receives data”** 欄位中顯示接收到的訊息。



5. 碰觸 TouchPAD #1 (Server) 及 TouchPAD #2 (Client) 上 **“Stop”** 按鈕圖示來關閉連線。





5. 進階 C 語言程式

我們有一份 TouchPAD 專用的 **“應用程式介面指南 (HMIWorks API Reference)”**。
https://www.icpdas.com/en/download/show.php?num=958

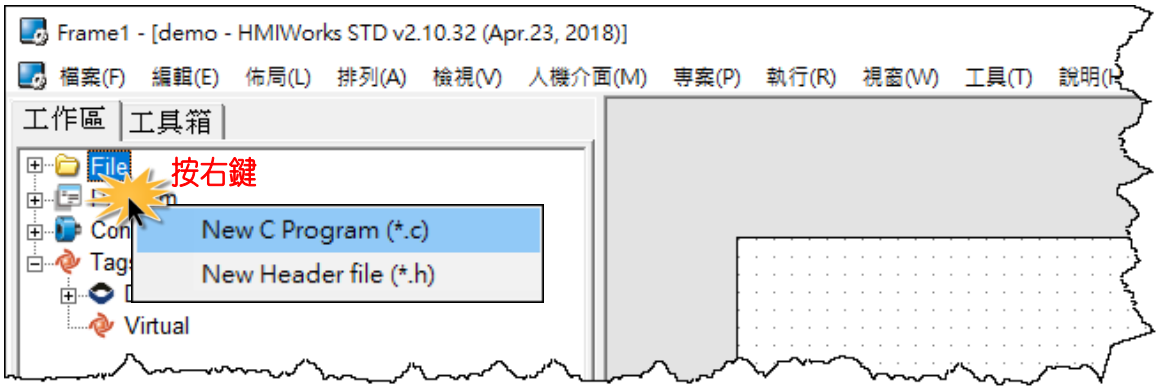
除了參考建置 (build) 程式後自動產生的程式碼來學習如何處理控制項，也可以透過標頭檔 (header file) 來了解。標頭檔定義了所有處理控制項的函數，在下面的路徑可以找到：“**C:\ICPDAS\HMIWorks_Standard\include\grlib**” 和 “**C:\ICPDAS\HMIWorks_Standard\include**”，其中“**C:\ICPDAS\HMIWorks_Standard**” 是安裝路徑。

下面我們將介紹幾個例子。

5.1 新增一個檔案到專案

在進一步介紹細節之前，首先我們先了解如何新增一個檔案 (**“.c”** 或 **“.h”** 檔) 至專案中。

1. 到 **“工作區”**。
2. 滑鼠右鍵點選 **“File”** 項目，然後一個彈出式選單會出現。
3. 在該彈出式選單中，選擇想要新增的檔案類別 (**“.c”** 或 **“.h”** 檔)。

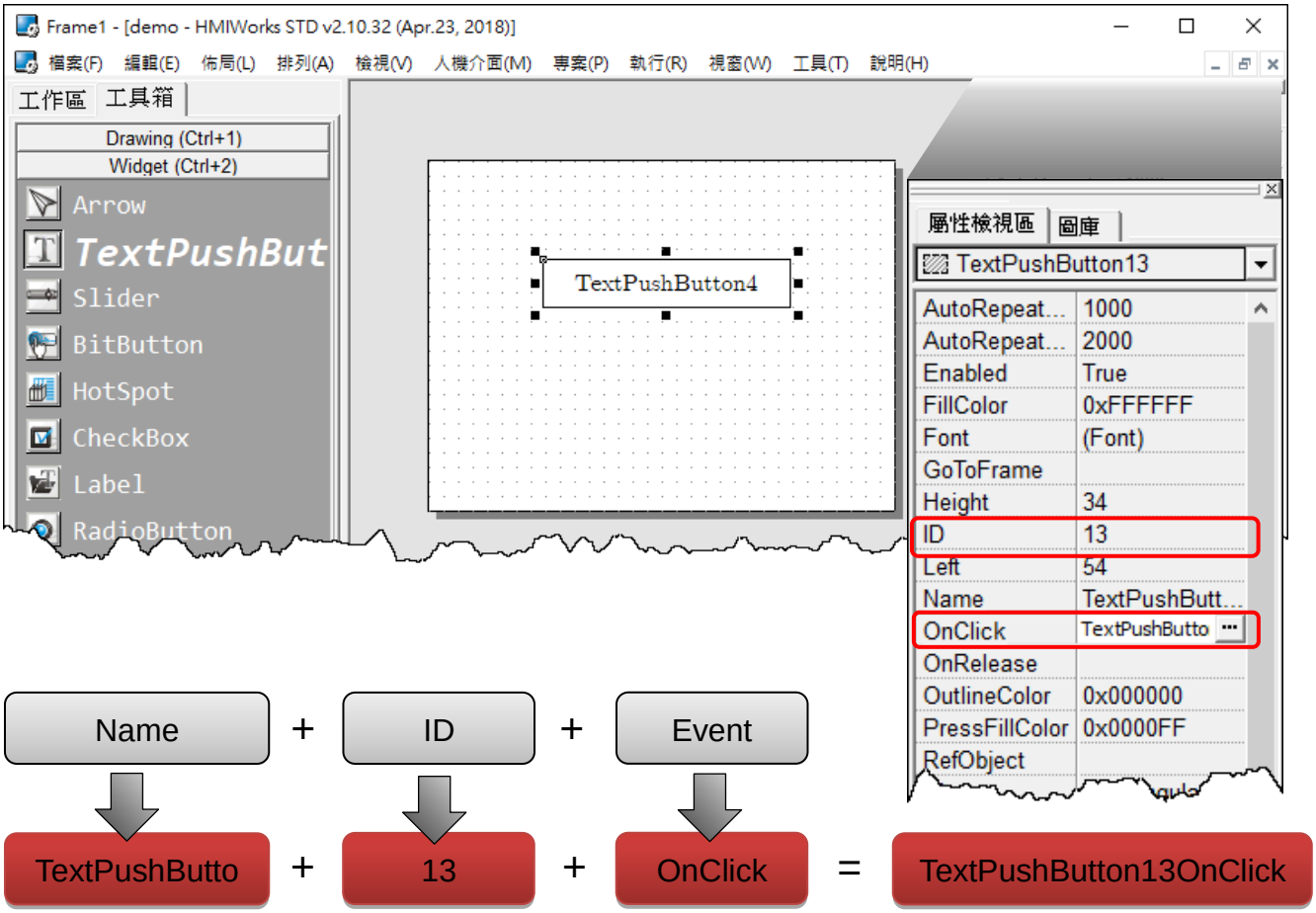


5.2 在執行時變更屬性

在執行時變更 **“Widget (控制項)” 屬性**有點複雜，這裡我們將介紹一些常見的功能。

1. **TextPushButton (文字彈回式按鈕)** 的 **FillColor (填滿色彩)** 和 **Text (文字)** 屬性，詳細說明參考 第 5.2.1 節。
2. 在 **Slider (滑桿)** 顯示百分比，詳細說明參考 第 5.2.2 節。
3. **CheckBox (核取方塊)** 的 **Selected (選取)** 屬性，詳細說明參考 第 5.2.3 節。
4. **Label (文字顯示框)** 的 **Font (字型)**、**Text (文字)** 和 **TextColor (文字色彩)** 屬性，詳細說明參考 第 5.2.4 節。

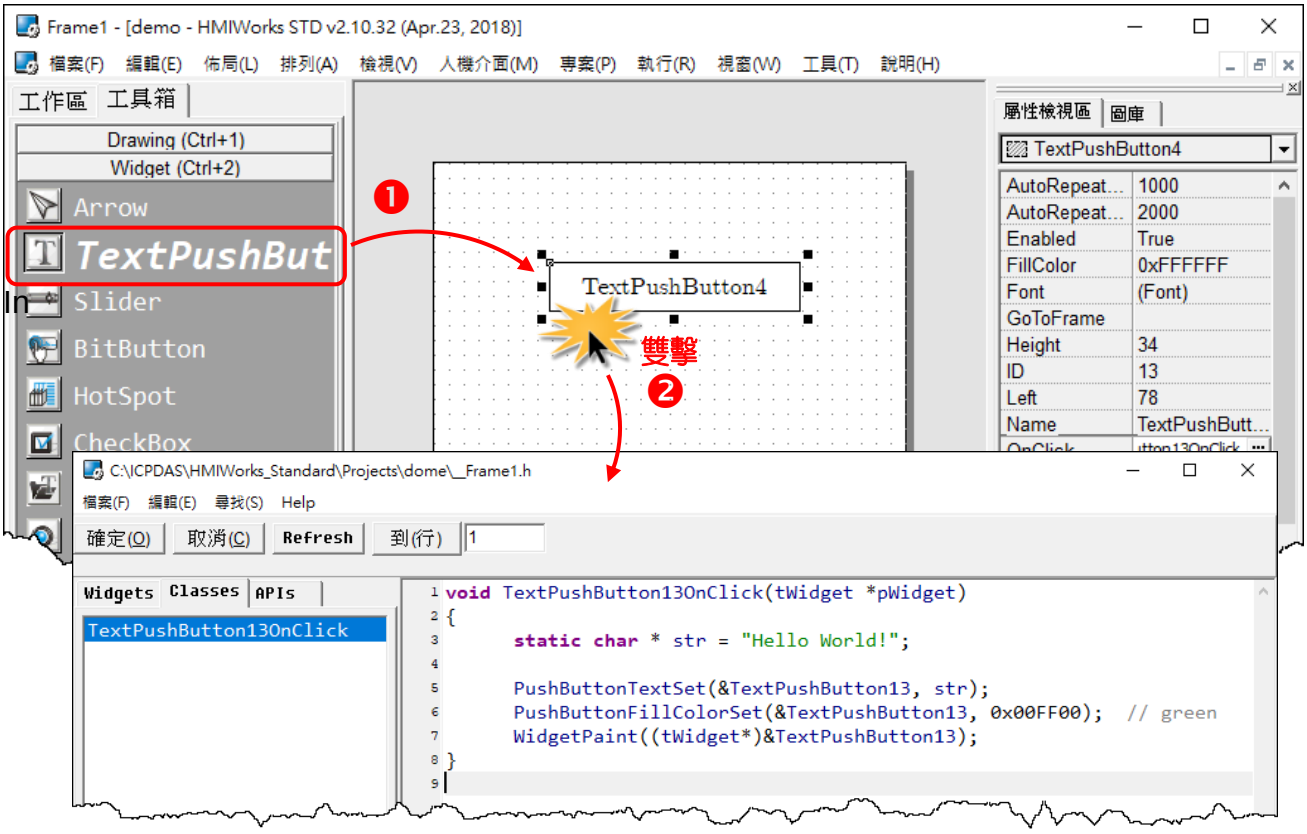
屬性的變更是在事件函數中實現的。
下面是控制項的事件函數名稱的命名規則，以文字彈回式按鈕為例。



5.2.1 文字彈回式按鈕的填滿色彩和文字屬性

本節示範如何改變 **“TextPushButton” (文字彈回式按鈕)** 的 **FillColor (填滿色彩)** 和 **Text (文字)**屬性。

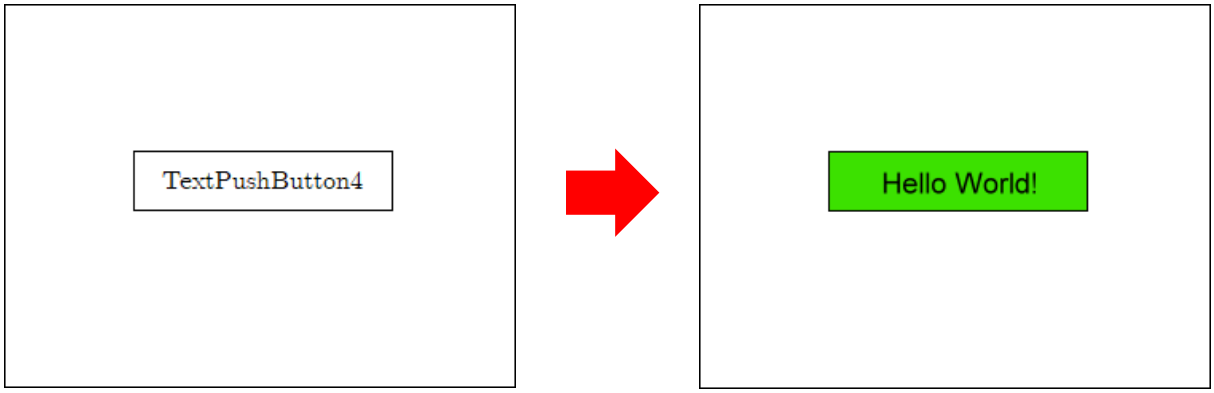
步驟 1: 拖放一個 **“TextPushButton” (文字彈回式按鈕)** 到顯示頁設計區。
步驟 2: 雙擊 **“TextPushButton” (文字彈回式按鈕)** 以編輯 **“OnClick (點擊事件)”** 函數。儲存檔案後按 **“確定(O)”** 鈕。



下面是事件函數編輯視窗中的程式碼。

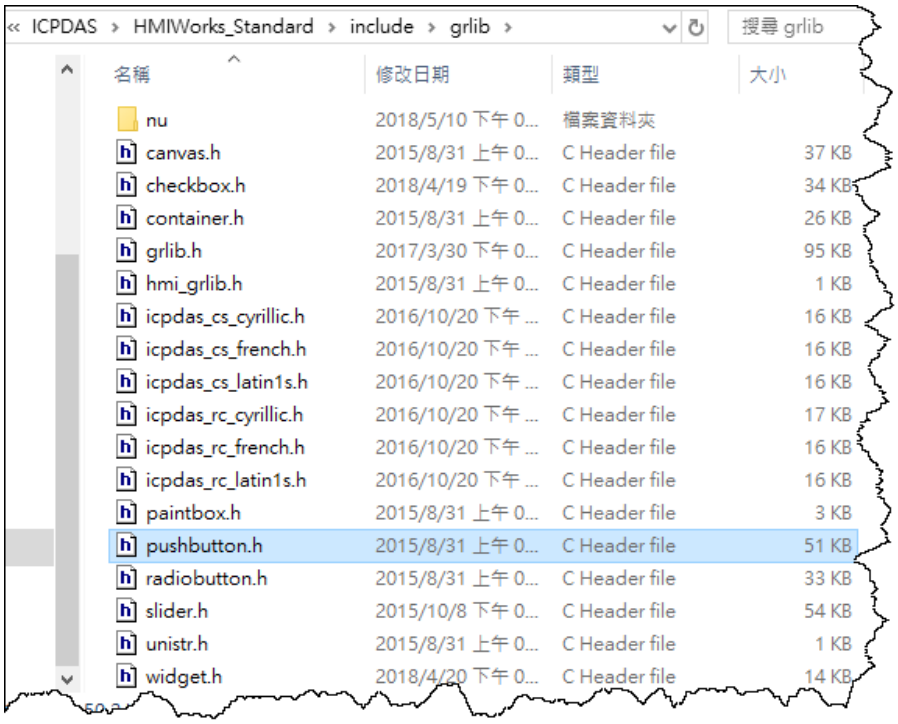
```
void TextPushButton13OnClick(tWidget *pWidget)
{
        static char * str = "Hello World!";

        PushButtonTextSet(&TextPushButton13, str);
        PushButtonFillColorSet(&TextPushButton13, 0x00FF00);    // green
        WidgetPaint((tWidget*)&TextPushButton13);
}
```

OnClick (點擊事件) 函數的效果：



為了方便使用者，要改變 **“TextPushButton (文字彈回式按鈕) ”** 的 **“Text (文字)”** 屬性，另外有一個函式可供使用 **“TextButtonTextSet”** ，詳細請參考 **“應用程式介面指南 (HMIWorks API Reference)”**，下載路徑如下:
https://www.icpdas.com/en/download/show.php?num=958

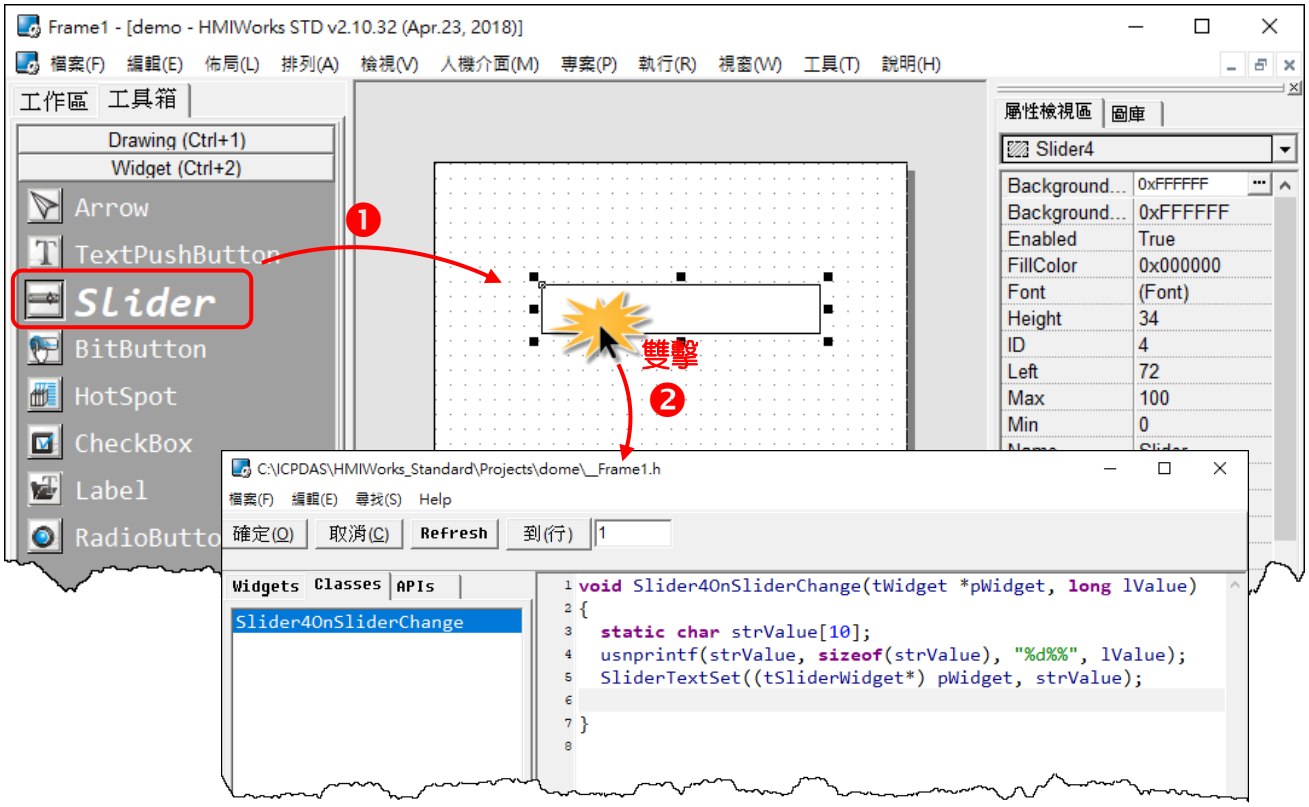
其他更多處理文 **“TextPushButton (文字彈回式按鈕) ”** 的函數，請參考位於下列路徑的 **pushbutton.h** 檔案：“**C:\ICPDAS\HMIWorks_Standard\include\grlib**”，
其中 “**C:\ICPDAS\HMIWorks_Standard**” 是安裝路徑。



5.2.2 滑桿顯示百分比

依照下列步驟可以在 **Slider (滑桿)** 顯示百分比。

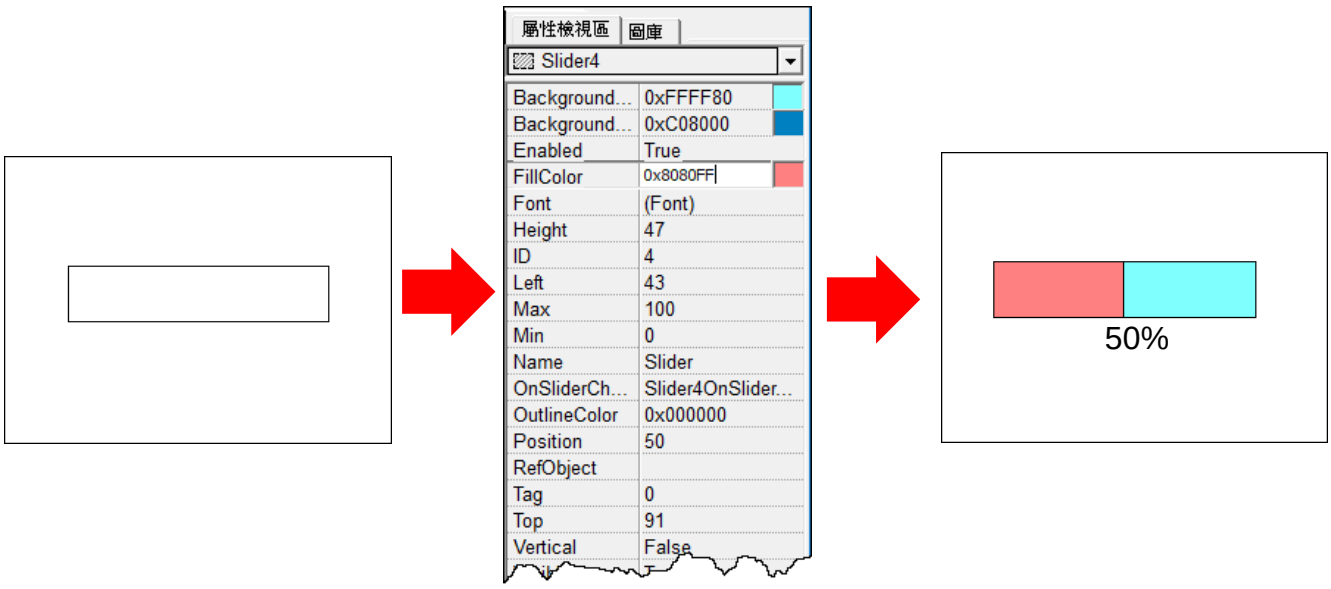
步驟 1: 拖放一個 **“Slider” (滑桿)** 到顯示頁設計區。
步驟 2: 雙擊點選滑桿以編輯 **OnSliderChange (滑桿改變事件)** 函數。儲存檔案後按 **“確認(O)”**鈕。



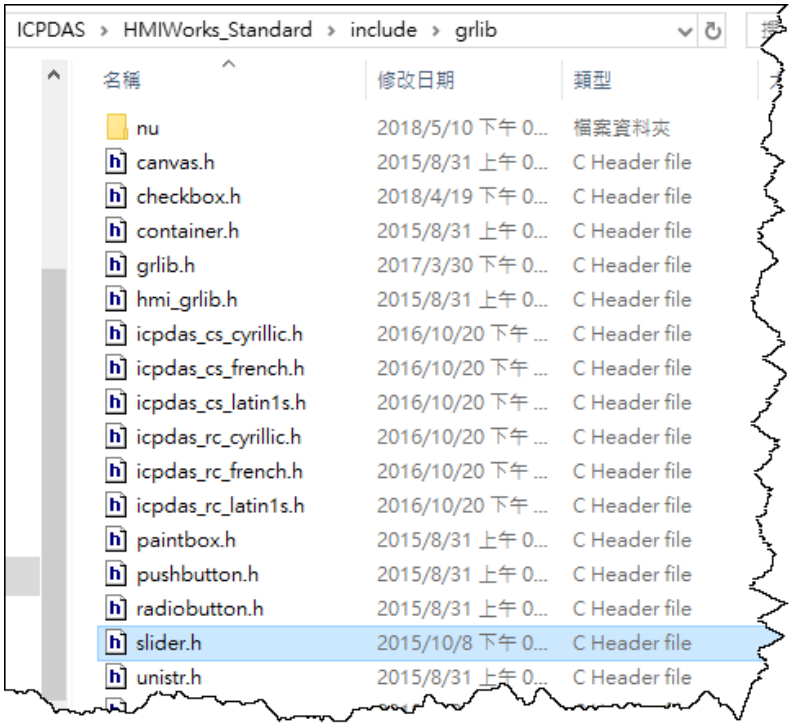
下面是事件函數編輯視窗中的程式碼。

```
void Slider4OnSliderChange(tWidget *pWidget, long lValue)
{
    static char strValue[10];
    usnprintf(strValue, sizeof(strValue), "%d%%", lValue);
    SliderTextSet((tSliderWidget*) pWidget, strValue);
}
```

OnSliderChange (滑桿改變事件) 函數的效果：
(如下圖所示，要先改變顏色才可以看見效果顯現。)



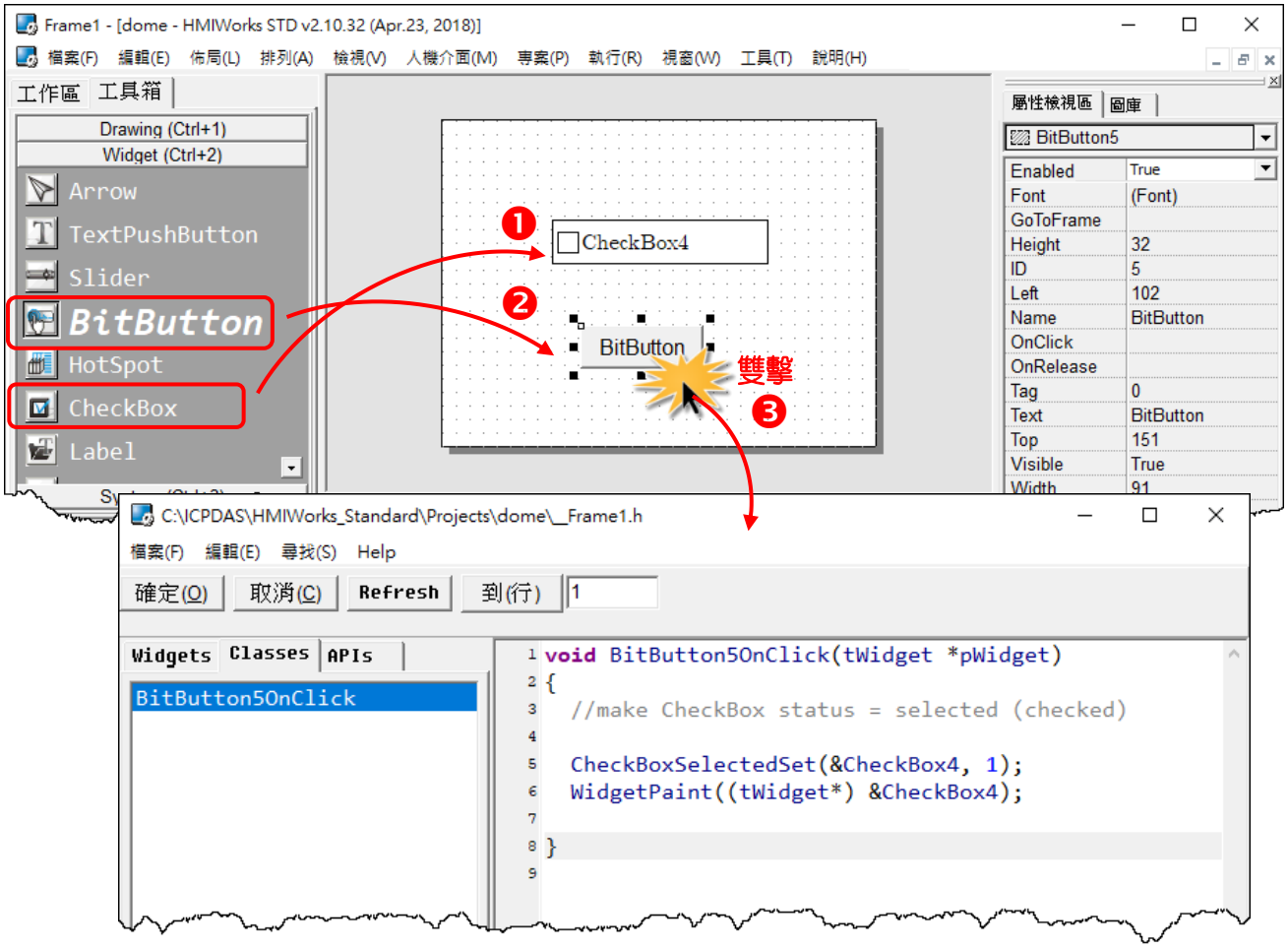
其他更多處理滑桿的函數，請參考位於下列路徑的 **slider.h** 檔案：
“**C:\ICPDAS\HMIWorks_Standard\include\grlib**”，其中 “**C:\ICPDAS\HMIWorks_Standard**” 是安裝路徑。



5.2.3 核取方塊的選取屬性

依照下列步驟可以在執行時，改變 **CheckBox (核取方塊)** 的 **Selected (選取)** 屬性。

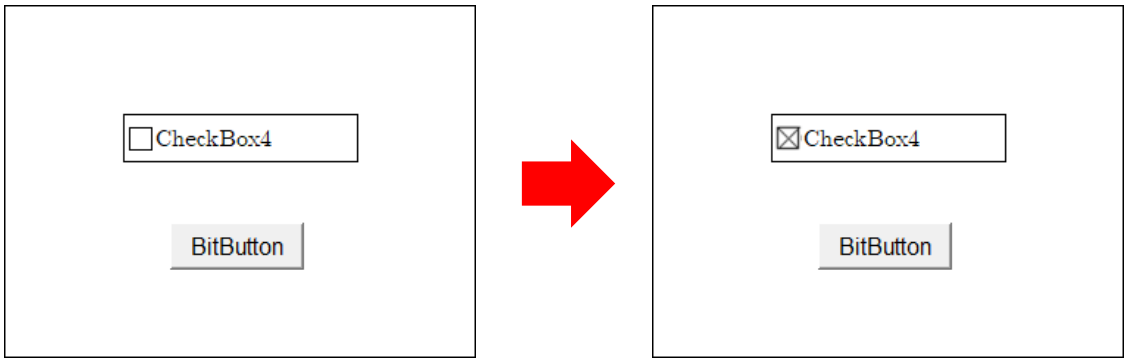
步驟 1: 拖放一個 **CheckBox (核取方塊)** 到顯示頁設計區。
步驟 2: 拖放一個 **BitButton (位元圖按鈕)** 到顯示頁設計區。
步驟 3: 雙擊 **BitButton (位元圖按鈕)** 以編輯 **OnClick (點擊事件)** 函數。儲存檔案後按 **“確認(O)”** 鈕。



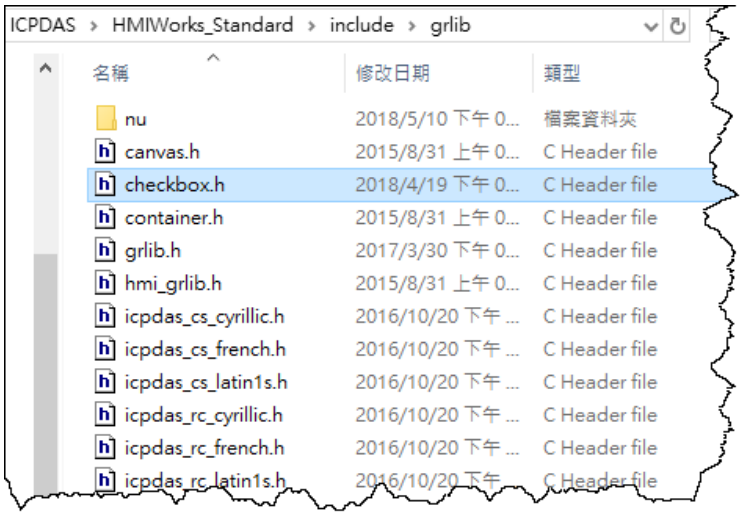
下面是事件函數編輯視窗中的程式碼。

```
void BitButton5OnClick(tWidget *pWidget)
{
    //make CheckBox status = selected (checked)

    CheckBoxSelectedSet(&CheckBox4, 1);
    WidgetPaint((tWidget*) &CheckBox4);
}
```

OnClick (點擊事件) 函數的效果：



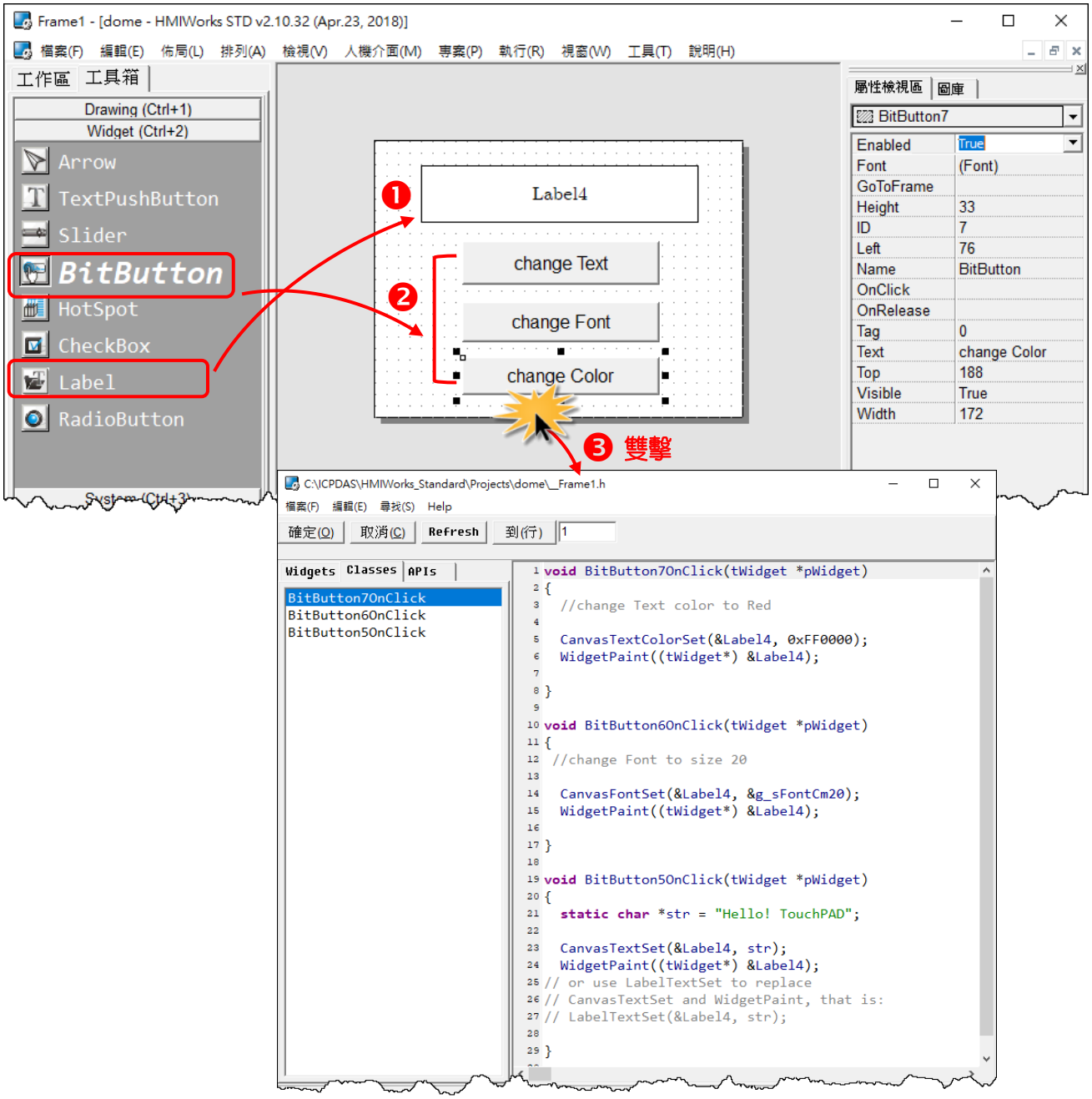
其他更多處理 **CheckBox (核取方塊)** 的函數，請參考位於下列路徑的 **checkbox.h** 檔案：“**C:\ICPDAS\HMIWorks_Standard\include\grlib**”，其中 “**C:\ICPDAS\HMIWorks_Standard**” 是安裝路徑。



5.2.4 文字顯示框的字型、文字和文字顏色

依照下列步驟可以在執行時，更新 **Label (文字顯示框)** 的 **Font (字型)**、**Text (文字)**、**TextColor (文字顏色)** 屬性。

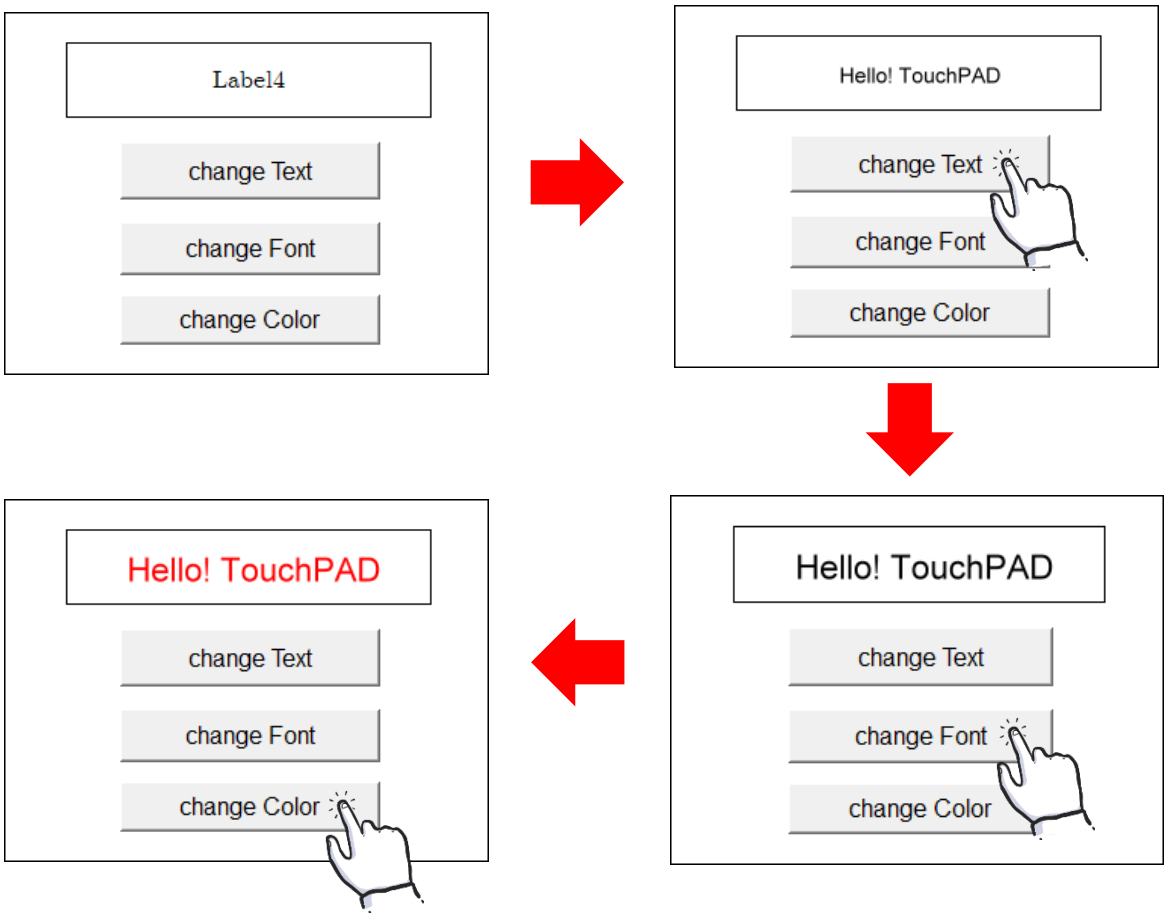
步驟 1: 拖放一個 **Label (文字顯示框)** 到顯示頁設計區。
步驟 2: 拖放三個 **BitButton (位元圖按鈕)** 到顯示頁設計區。
步驟 3: 雙擊 **BitButton (位元圖按鈕)** 以編輯 **OnClick (點擊事件)** 函數。儲存檔案後按 **“確認(O)”** 鈕。



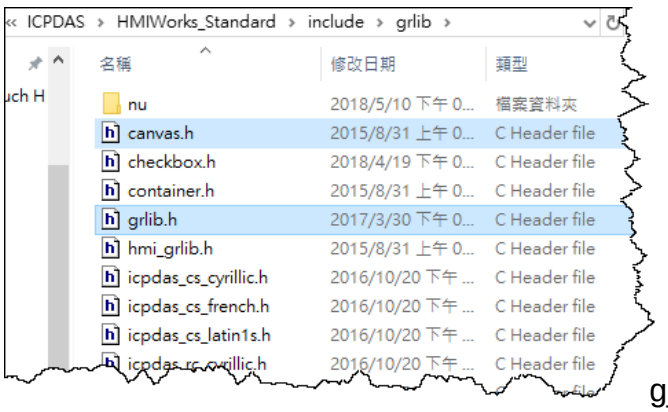
下面是事件函數編輯視窗中的程式碼。

```
//Click on BitButton7 “change Color”
void BitButton7OnClick(tWidget *pWidget)
{
    //change Text color to Red

    CanvasTextColorSet(&Label4, 0xFF0000);
    WidgetPaint((tWidget*) &Label4);
}

//Click on BitButton6 “change Font”
void BitButton6OnClick(tWidget *pWidget)
{
    //change Font to size 20

    CanvasFontSet(&Label4, &g_sFontCm20);
    WidgetPaint((tWidget*) &Label4);
}

//Click on BitButton5 “change Text”
void BitButton5OnClick(tWidget *pWidget)
{
    static char *str = "Hello! TouchPAD";

    CanvasTextSet(&Label4, str);
    WidgetPaint((tWidget*) &Label4);
// or use LabelTextSet to replace
// CanvasTextSet and WidgetPaint, that is:
// LabelTextSet(&Label4, str);
}
```

OnClick (點擊事件) 函數的效果：



為了方便使用者，要改變 **Label (文字顯示框)** 的 **“Text (文字)”** 屬性，另外有一個函式可供使用 “**LabelTextSet**”。詳請參閱 **“應用程式介面指南(HMIWorks API Reference) ”**，其下載路徑如下 https://www.icpdas.com/en/download/show.php?num=958



其他更多處理 **Label (文字顯示框)** 的函數，請參考位於下列路徑的 **canvas.h** 檔案：“**C:\ICPDAS\HMIWorks_Standard\include\grlib**”，其中 “**C:\ICPDAS\HMIWorks_Standard**” 是安裝路徑。

在相同的路徑下，另有一個 **grlib.h** 檔案。**grlib.h** 定義有預設的字型種類，如 g_sFontCm20。

5.3 存取階梯圖中的標籤值

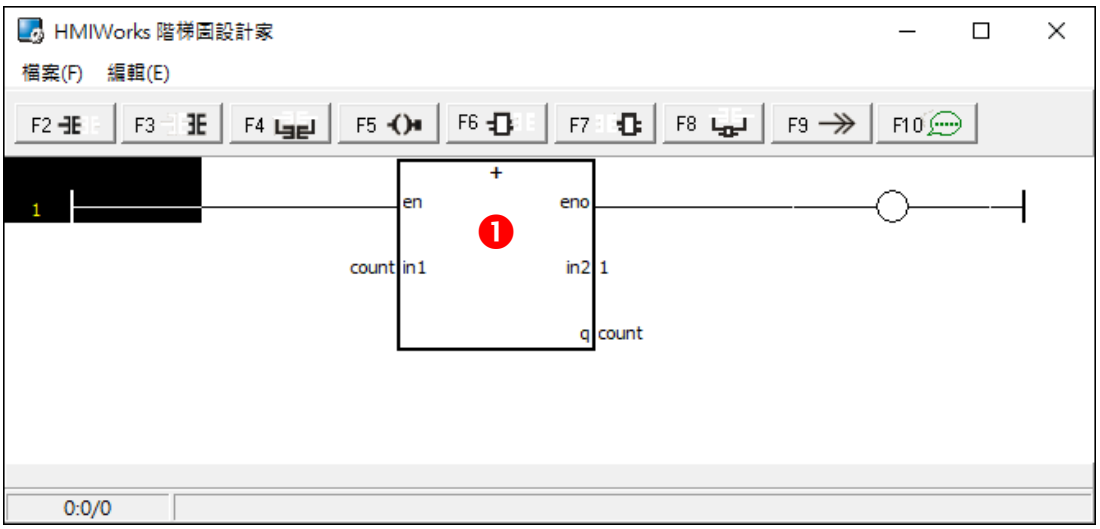
在 HMIWorks 中，一個專案可以同時有很多個 **Frame (顯示頁)**，而這些顯示頁可以是屬於 **“標準C語言”** 或是 **“階梯圖”** 這兩個類別的其中之一。而在 **“階梯圖”** 中的變數，即標籤 (Tag)，會在專案建置 (build) 時被轉為 C 語言的結構，因此 **“標準 C 語言”** 的顯示頁是可以存取階梯圖中標籤的值。

下列兩個**巨集函數**就是用來實現這個目的：

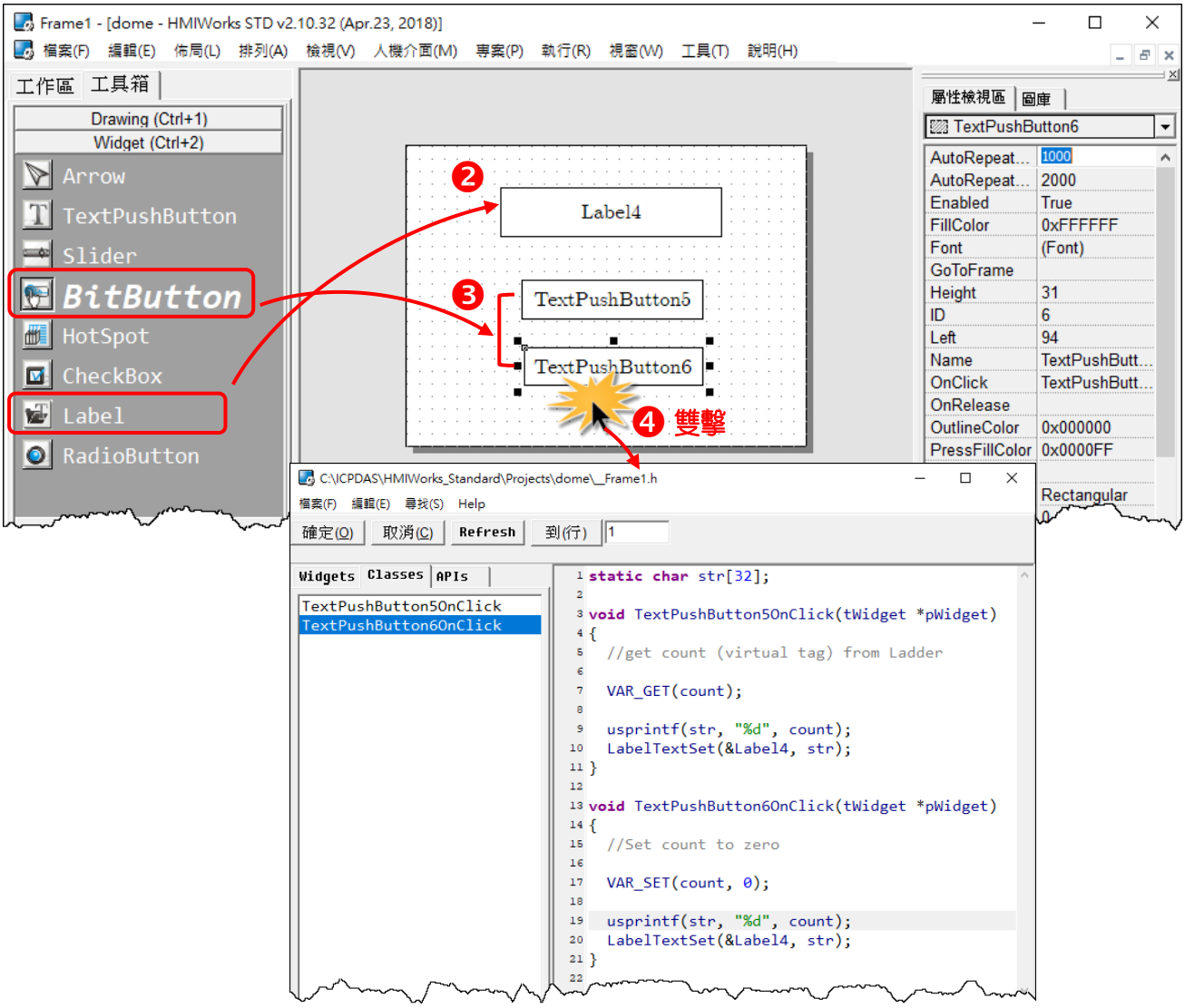
1. **VAR_GET**: 取得階梯圖中標籤的值
2. **VAR_SET**: 寫入值於階梯圖中的標籤

下面範例中，我們將新增一個**標籤 “count”**，然後在階梯圖中持續被加一，也可以將**標籤 “count”**的值設為零。

步驟 1: 按 **<F2> 鍵**來新增一個 **“count”** 標籤後，再按 **<F4> 鍵**來開啟 **“HMIWorks 階梯圖設計家”** 視窗，然後建立 **“count”** 遞增的計算式，如下。



步驟 2: 拖放一個 **Label (文字顯示框)** 到顯示頁設計區。
步驟 3: 拖放二個 **BitButton (位元圖按鈕)** 到顯示頁設計區。
步驟 4: 雙擊 **BitButton (位元圖按鈕)** 以編輯 **OnClick (點擊事件)** 函數。儲存檔案後按 **“確認(O)”**

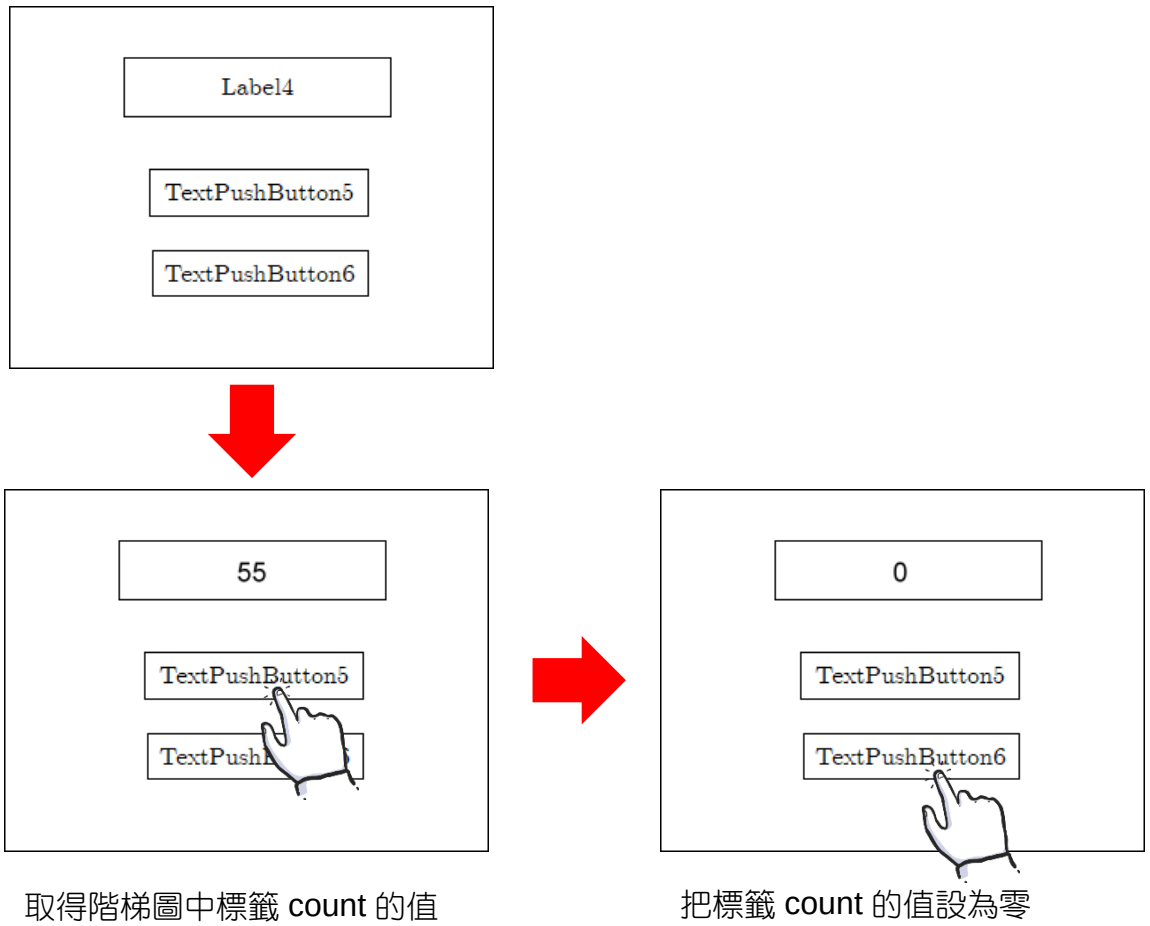


下面是事件函數編輯視窗中的程式碼。

```
static char str[32];

//Click on BitButton5 to get count (virtual tag) from Ladder
void TextPushButton5OnClick(tWidget *pWidget)
{
   VAR_GET(count);

   usprintf(str, "%d", count);
   LabelTextSet(&Label4, str);
}
```

```
// Click on BitButton6 to Set count to zero
void TextPushButton6OnClick(tWidget *pWidget)
{
    VAR_SET(count, 0);

    usprintf(str, "%d", count);
    LabelTextSet(&Label4, str);
}
```

OnClick (點擊事件) 函數的效果：



取得階梯圖中標籤 count 的值

把標籤 count 的值設為零

附錄

A. 常見問題

更多更詳細的常見問題，可參考泓格網站的 Q & A:
TouchPAD 常見問題

A.1. 當螢幕閃爍時該怎麼做?

詳細參考 第 3.4.2 節 Frame 。

A.2. 如何顯示高解析的圖片?

詳細參考 第 3.4.6 節 Picture。

A.3. TouchPAD 如何控制 I/O?

詳細參考 第 3.3.6 節 使用工具和標籤產生關聯 及 第 3.4.17 節 ObjectList。

A.4. 如何改變 Text (恆定文字框) 的字型?

詳細參考 第 3.4.5 節 Text。

A.5. 如何在階梯圖設計家使用小數?

詳細參考 第 3.4.13 節 Label。

A.6. 如何清除 Paint Box (繪圖框)?

詳細參考 第 3.4.16 節 PaintBox。

A.7. 如何取消 TouchPAD 啟動時嗶聲?

TouchPAD 裝置在啟動時會發出嗶聲，您可以透過 **“專案組態設定”** 視窗來進行修改，詳細參考 第 3.2.2 節 專案組態設定。

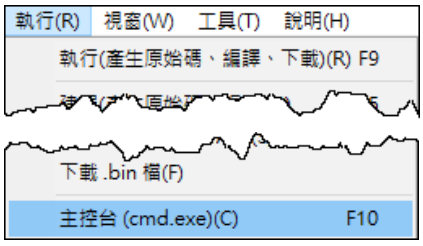
A.8. 如何修改 HMIWorks 自動產生的程式碼?

每次在**建置 (build)**一個專案時，HMIWorks 都會自動產生原始程式碼。下列步驟說明了如何客製化地修改自動產生的原始碼。

1. 在專案完成後，按 **<F5> 鍵(建置)**， 而不是按 **<F9> 鍵(執行)** 來產生原始碼。

2. 在專案所在的目錄中，開啟想要編輯的**原始碼檔案 (.c 檔)**。

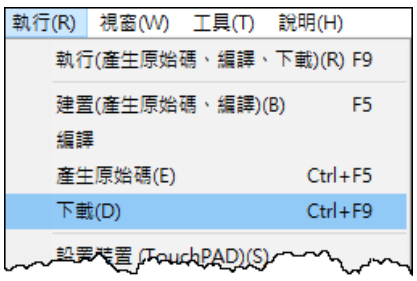
3. 編輯該**原始碼檔案 (.c 檔)**。



4. 在 HMIWorks 中，按 **<F10> 鍵**來開啟 **cmd.exe** 的視窗。在其中輸入 **“make”** 以重新建置專案。

5. 對 TPD-283U-H/TPD-283U-Mx 來說，在輸入 **“make”** 之後還有其他步驟要執行。
輸入**“make genbix”**

6. 按 **< Ctrl > + <F9> 鍵**來下載重新建置的檔案 **(.bin 或.bix 檔案)**。



A.9. 如何在 Flash 中儲存資料?

為了便利使用者，對於儲存資料於 TouchPAD 的 Flash (快閃記憶體) 中，共有兩套 API 函式。一套是給 MCU (微控制器，micro-controller unit) 內部的 Flash，另一套是給外部串列 (Serial) 的 Flash。

請更新至 HMIWorks 2.03 版 (或以上)，HMIWorks 軟體下載位置如下:
https://www.icpdas.com/en/download/show.php?num=944

項目	1	2
目標 Flash	MCU 內部的 Flash	外部串列的 Flash
可能的目標裝置	TouchPAD 全系列的裝置	TouchPAD 全系列的裝置，即除了 TPD-280 和 TPD-283 之外 (即具有外部 Flash 的裝置)
提供的 API 函式*	hmi_UserParamsGet, hmi_UserParamsSet	hmi_UserFlashReadEx, hmi_UserFlashWriteEx, hmi_UserFlashConfig, hmi_UserFlashErase
儲存空間	256 byte	4 KB ~ 7 MB
建議使用者	任何 TouchPAD 的使用者	進階的使用者。 誤用可能導致應用程式區域毀損。

*詳請參閱 **“應用程式介面指南 (API Reference)”**，下載位置如下:
https://www.icpdas.com/en/download/show.php?num=958

A.10. 如何實現軟體重開機?

有兩個方法可以實現軟體重開機，說明如下:

方法 1: 使用函式 **hmi_SoftwareReset**。

方法 2: 使用看門狗計時器。

① 開啟看門狗計時器
從 **“專案(P)”** 功能選單中，點選 **“專案組態設定(P)”** 項目來開啟“專案組態”配置視窗，然後在設定 **“看門狗計時器 (WDT)”** 區中的項目。

② 使用無窮迴圈，引發看門狗
範例: while(1){}
如使用階梯圖設計，詳細參考 第 3.3.5 節 使用者定義的功能方塊。

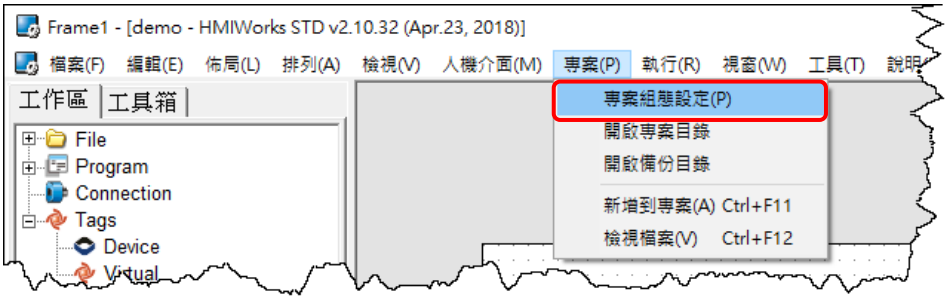
A.11. 如何使 TouchPAD 作為 Modbus RTU/TCP Slave?

詳細參考 FAQ: 如何使 TouchPAD 作為 Modbus RTU Slave? 及 FAQ: 如何使 TouchPAD 作為 Modbus TCP Slave?。

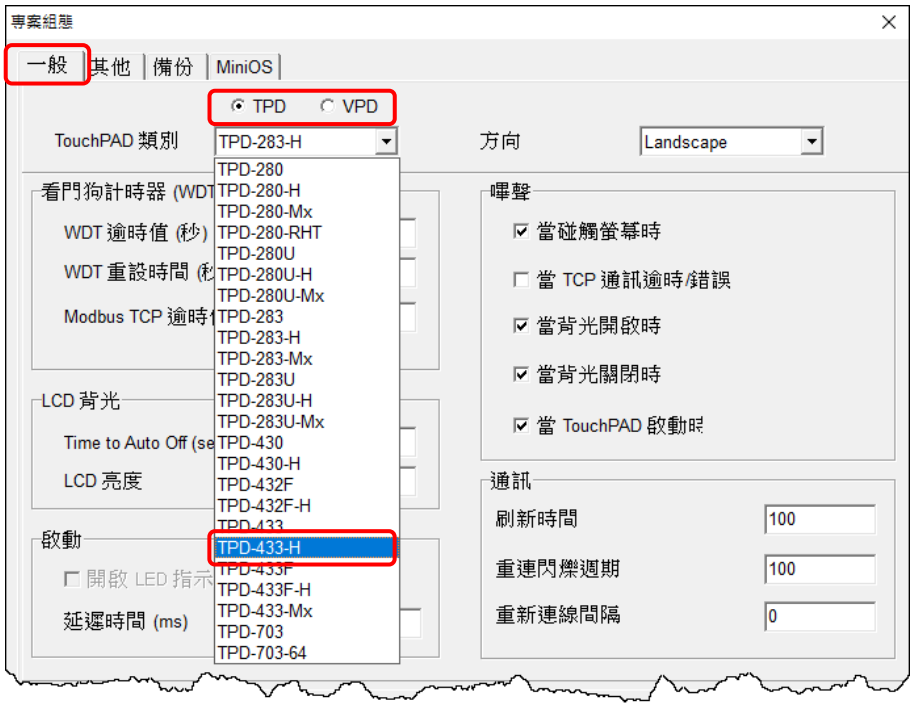
A.12. 如何將 TouchPAD 非 H 版專案轉移到 TouchPAD H 版上?

下面範例，我們將 TPD-433F 專案轉移到 TPD-433F-H 上:

1. 使用 **HMIWorks v2.10.22 或更新的版本**打開原始專案。
2. 點選 HMIWorks 功能選單內的 **"專案(P) → 專案組態設定(P)"**。



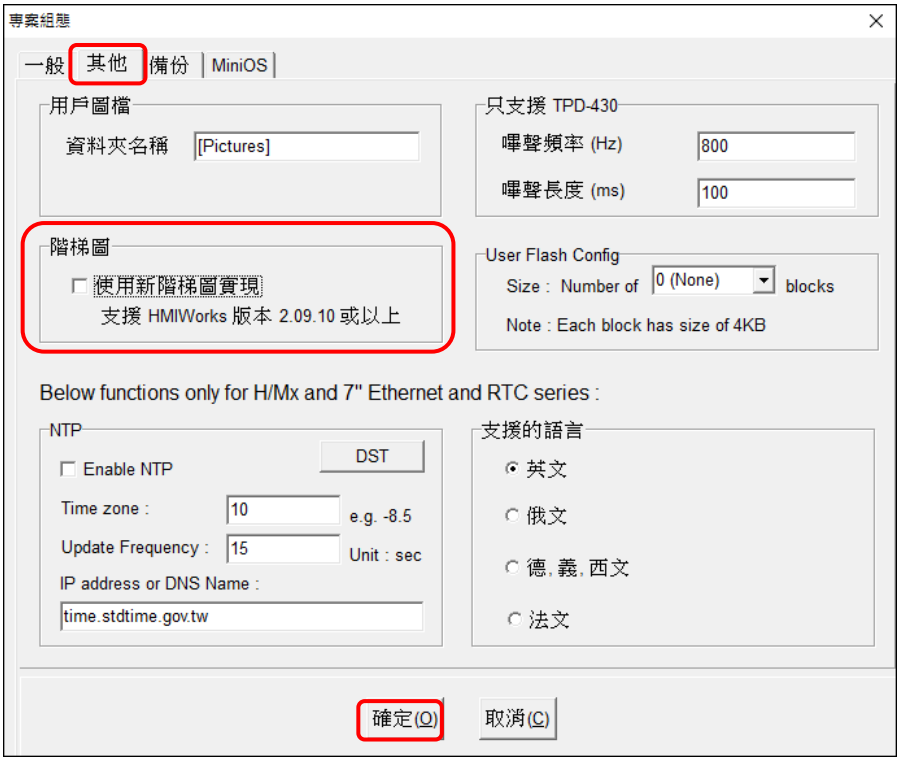
3. 在 **"一般"** 配置區，點選 **“TPD”** 項目。
4. 在 **"TouchPAD 類別"** 下拉式選單中更改為 TPD-433F-H。



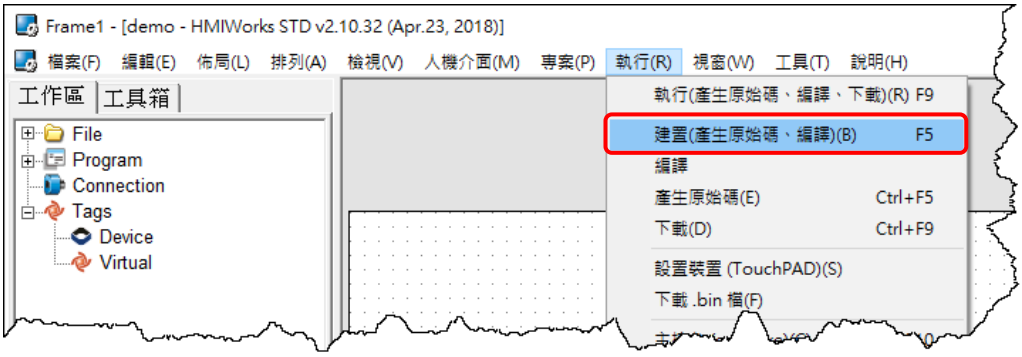
如果您原來的專案是使用 HMIWorks v2.09.09 或更早版本的階梯圖所建立的，請參考**步驟 6 到步驟 8 來關閉階梯圖模式**。

- **新的階梯圖模組:** Coil-Set 及 Coil-Reset 可改變 Coil 狀態並且 Lock 住(工業標準)直到 Coil-Reset 或 Coil-Set。
- **舊的階梯圖模組:** 無 Lock 特性。

6. 在 **"其他"** 配置區。
7. 取消 **“階梯圖”** 區域中 **"Use New Ladder Implementation”** 項目。
8. 按下 **“確定(O)”** 按鈕。



8. 點選 HMIWorks 功能選單內的 **"執行(R) → 建置(B)"** 來重新建置專案 (或按 **<F5>** 鍵)。



B. 手冊修訂記錄

本章提供此使用手冊的修訂記錄。

下表提供此文件每次修訂的日期與說明。

版本	發行日	說明
1.0.25	2015 年 5 月	首次發行。
1.1.0	2015 年 7 月	將使用手冊分為 TouchPAD 硬體手冊及 HMIWorks 軟體手冊二本。
1.2.0	2018 年 5 月	操作畫面以 HMIWorks (v2.10.32) 來更新。 更新階梯圖的功能方塊。 新增 4.3 節 TouchPAD 與 I/O 模組整合。 新增 FAQ: 如何將 TouchPAD 非 H 版專案轉移到 TouchPAD H 版上。
1.3.0	2019 年 1 月	新增 4.4 節 TCP/IP 通訊 新增 4.4.1 節 如何使 TouchPAD 作為 TCP Client? 新增 4.4.2 節 如何使 TouchPAD 作為 TCP Server?
1.4.0	2020 年 6 月	增加新增功能方塊(Function block)介紹，並獨立為 3.3.4 節。
1.5.0	2022 年 7 月	增加 DGW-521 功能方塊(Function block)介紹。