以 AppWizard 設計新能源閘道器 GUI 專案

NuMicro® 32位系列微控制器範例代碼介紹

文件資訊

應用簡述	本程式使用 SEGGER emWin AppWizard 工具，結合 NUC980 系列開發平台與開發資源，完成新能源閘道器(New Energy Gateway)之圖形化使用介面。
BSP 版本	emWin Software Package for NUC980
開發平台	NuMaker-IIoT-NUC980G2D V1.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. 概述

本範例程式將示範如何使用 SEGGER emWin AppWizard 工具，搭配 NuMaker-IIoT-NUC980G2D 開發平台，逐步完成新能源閘道器(Gateway)的 GUI(Graphical User Interface)介面設計與畫面互動。

1.1 原理

1.1.1 系統方塊圖

此範例程式使用新唐 GUI 開發平台 NuMaker-IIoT-NUC980G2D 開發板，其使用解析度 240x320 且帶有尺寸為 2.4 吋的 TFT-LCD 電阻式觸控面板、主控板上 NUC980G2D 微處理器與容量大小為 1 Gbit 的 SPI NAND Flash。其系統方塊圖如圖 1-2 所示。

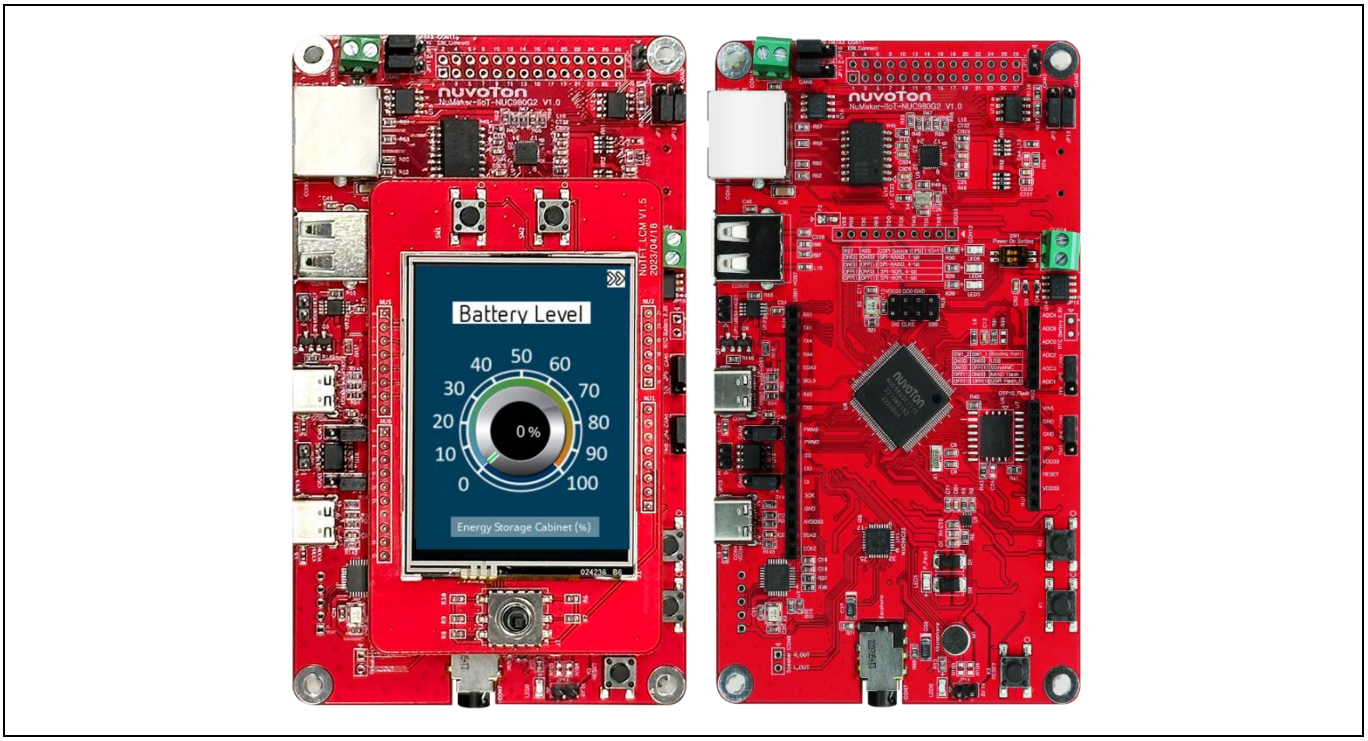


圖 1-1 NuMaker-IIoT-NUC980G2D 開發板

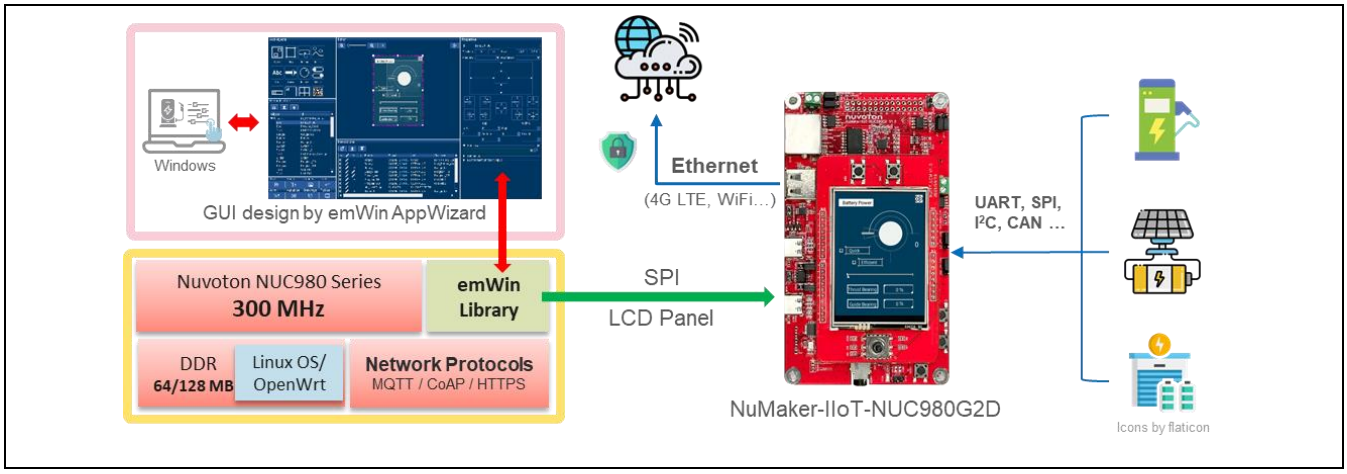


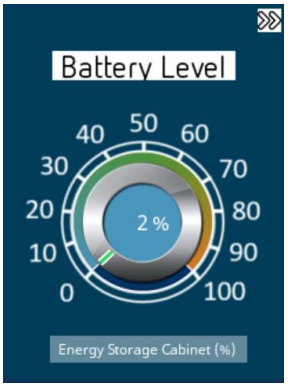
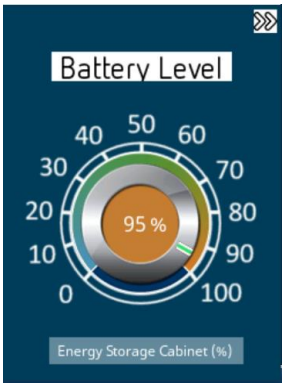
圖 1-2 系統方塊圖

1.1.2 相關資源

- 新唐官方論壇：Nuvoton emWin Forum
- SEGGER emWin 官方論壇：Segger emWin Forum
- SEGGER emWin AppWizard 工具使用手冊：AppWizard User Manual

1.2 新能源閘道器使用者介面(UI)設計

此範例設計畫面如表 1-1 所示：

	初始畫面	互動效果
主畫面		

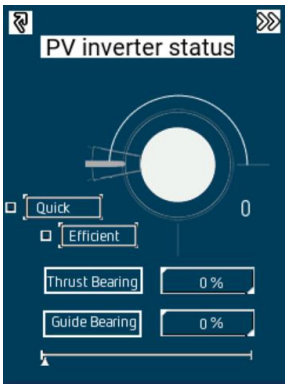
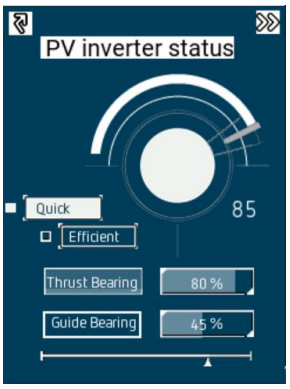
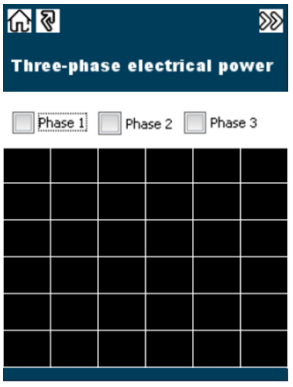
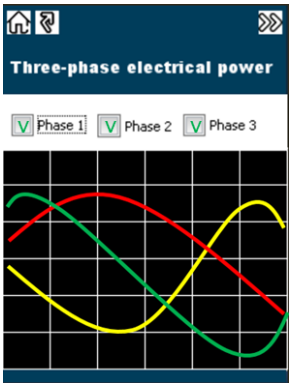
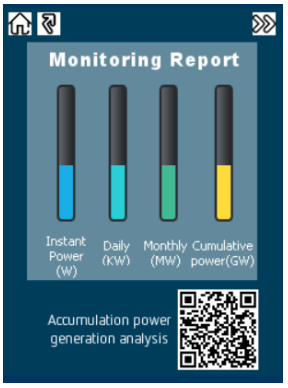
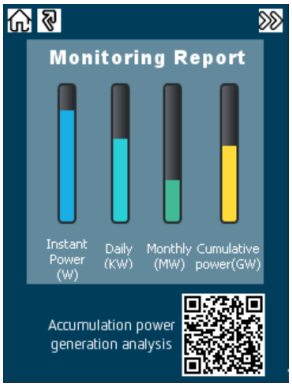
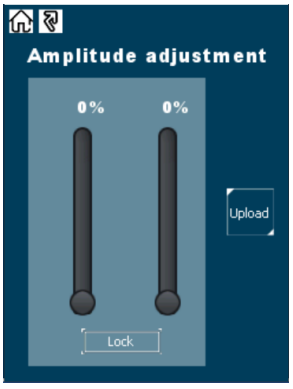
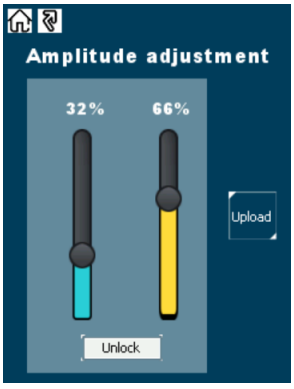
頁面二		
頁面三		
頁面四		
頁面五		

表 1-1 新能源閘道器畫面

2. 操作步驟

使用 AppWizard 完成一項 GUI 專案的執行步驟可分為四大階段，第一階段需準備硬體與軟體下載，第二階段則於 AppWizard 中完成對應的畫面設計、物件、互動設置、動畫設置與繪圖設定，第三階段使用 AppWizard 匯出程式碼並同時加入客製化程式，以完成更多應用功能，第四階段則是將應用程式實際燒錄到開發板操作。以下章節就各項階段詳細說明執行步驟。

2.1 準備相關硬體與軟體資源

1. 硬體使用 NuMaker-IIoT-NUC980G2D，相關資源: 開發板介紹、參考線路、使用手冊。

2. 軟體資源需下載 GUI_Builder 及 AppWizard 軟體工具

   - emWin IDE
     - GUIBuilder.exe
   - emWin AppWizard IDE 與 emWin Package
     - 請於新唐官網下載 emWin Package 檔案 (需註冊並登入官網使用者帳號)
     - 將 emWin Package 解壓縮後，點選 *AppWizard_Install.exe* 即可安裝 emWin AppWizard IDE

GUIBuilder.exe 資源已包含在範例程式包內，檔案位置如圖 2-1 所示。

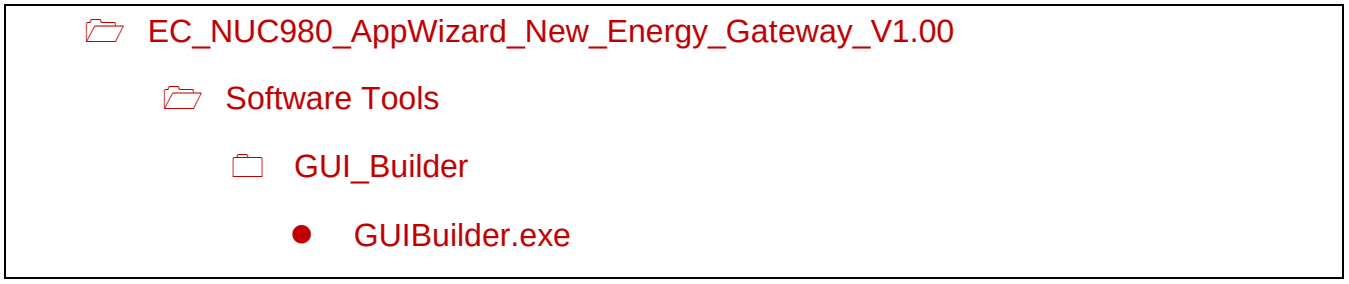


圖 2-1 必要工具路徑

3. NuMaker-IIoT-NUC980G2D 使用 Linux 環境開發，請確認電腦已安裝 VMWare 虛擬機，或透過新唐 Github 網站下載 NUC980 系列專用虛擬機。

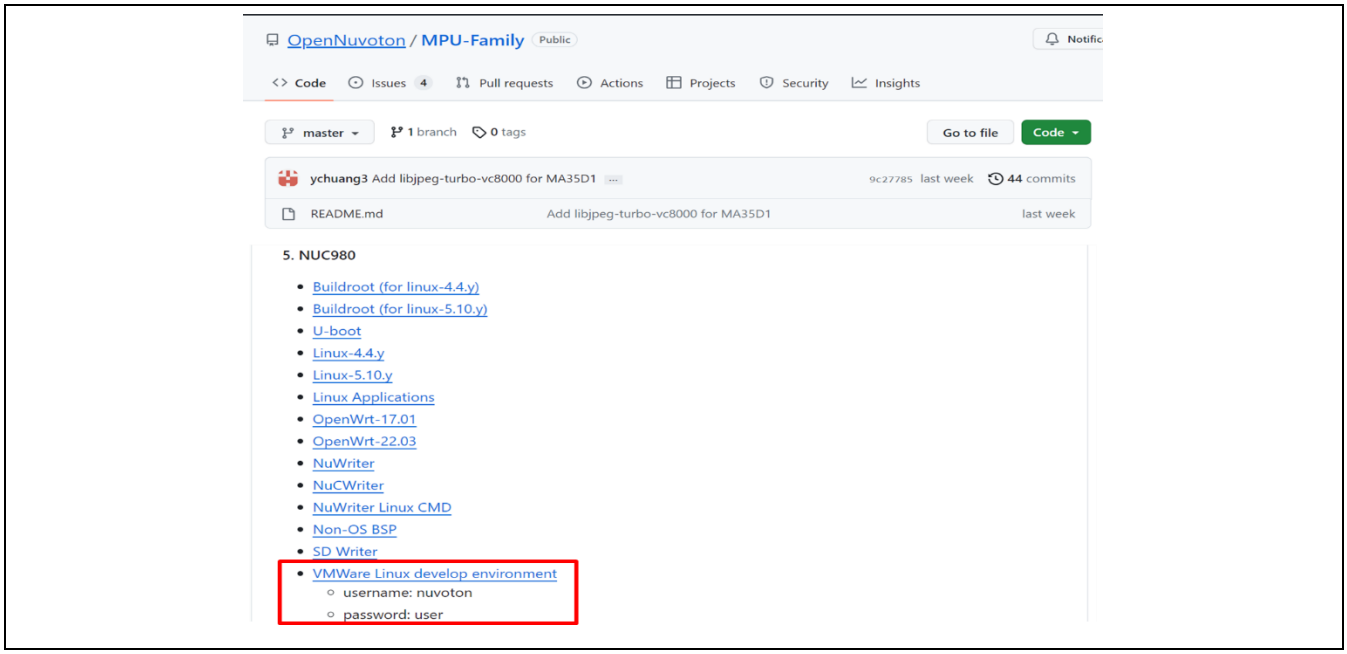


圖 2-2 新唐提供之 VMWare 虛擬機載點

4. 執行此範例前，使用者需先完成 NUC980 Linux Kernel 與基礎建設(包含 Buildroot, u-boot, tool chain…等設置)，目前 NuMaker-IIoT-NUC980G2D 有提供 Linux kernel 5.10 版本的資源，詳細檔案編譯流程如下所述：

- 開啟 NUC980 VMWare，由 Terminal 進入 MA35D1_Buildroot 資料夾後，透過指令 **make** 編譯 NuMaker-IIoT-NUC980G2D 設置檔 "nuvoton_nuc980_iot_g2d_defconfig"

```
users@ubuntu: ~/MA35D1_Buildroot$ make nuvoton_nuc980_iot_g2d_defconfig
users@ubuntu: ~/MA35D1_Buildroot$ make
```

- 編譯完成後，將有五個映像檔可下載，用途分別為：
  - Image: 於 DDR 上運作的 Linux 核心映像檔
  - uImage: 於 uboot 引導的 Linux 核心映像檔
  - dtb: Device Tree 二進位文件，詳細描述外設的設定
  - u-boot.bin：Uboot 映像檔
  - u-boot-spl.bin：Uboot 輔助程式載入器映像檔，用於從 SPI NAND 啟動
- 編譯完成後，可分別於下述路徑存取檔案：

- *"/MA35D1_Buildroot/output/images"* : **Image**、**uImage** 與 **nuc980-iot-g2d-v1.0.dtb**
- *"/MA35D1_Buildroot/output/build/uboot-customer"*: **U-boot.bin**
- *"/MA35D1_Buildroot /output/build/uboot-master/spl"* : **u-boot-spl.bin**

5. 由於有數個需要燒錄的檔案，因此使用者需要再建立一份 env.txt 配置映像檔燒錄位置與設定，env.txt 內容如下：

```
baudrate=115200
bootdelay=1
stderr=serial
stdin=serial
stdout=serial
loadkernel=nand read 0x7fc0 0x200000 0x800000
loaddtb=nand read 0x1400000 0x180000 0x20000
bootcmd=run loadkernel;run loaddtb;bootm 0x07fc0 - 0x1400000
```

6. 下載 NUC980 專用的 NuWriter 燒錄器，將編譯好的映像檔與 env.txt 透過 NuWriter 燒錄到 NuMaker-IIoT-NUC980G2D 開發板，燒錄流程如下所述：

- 將開發板 SW4 切換至 USB boot mode
- 燒錄模式(Choose Type)選擇：SPI NAND
- 清除所有區塊內容
- 燒錄 u-boot-spl.bin
  - Image Name: 選擇指定檔案路徑
  - Image Type: Loader
  - Image execute address: 0x200
  - Image start offset: N/A
- 燒錄 uBoot
  - Image Name: 選擇指定檔案路徑
  - Image Type: Data
  - Image execute address: N/A
  - Image start offset: 0x100000
- 燒錄 nuc980-iot-g2d-v1.0.dtb

  - Image Name: 選擇指定檔案路徑
  - Image Type: Data
  - Image execute address: N/A
  - Image start offset: 0x180000
- 燒錄 uImage
  - Image Name: 選擇指定檔案路徑
  - Image Type: Data
  - Image execute address: N/A
  - Image start offset: 0x200000
- 燒錄環境 (env.txt)
  - Image Name: 選擇指定檔案路徑
  - Image Type: Environment
  - Image execute address: N/A
  - Image start offset: 0x80000
- 燒錄完成後，將 SW4 切回 Nand Boot mode 並重啟開發板即可使用。

備註: 更詳細 Kernel 編譯與燒錄流程說明，請參閱手冊「NUC980 Linux Environment on VMware User Manual」與「NuMaker-IIoT-NUC980G2D User Manual」(章節 5.5 NuWriter Tool)。

2.2 開始 AppWizard 設計專案

1. 於桌面捷徑開啟 AppWizard 工具。

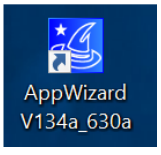


圖 2-3 AppWizard 桌面捷徑圖示

2. 點選”Create new project”建立新專案。

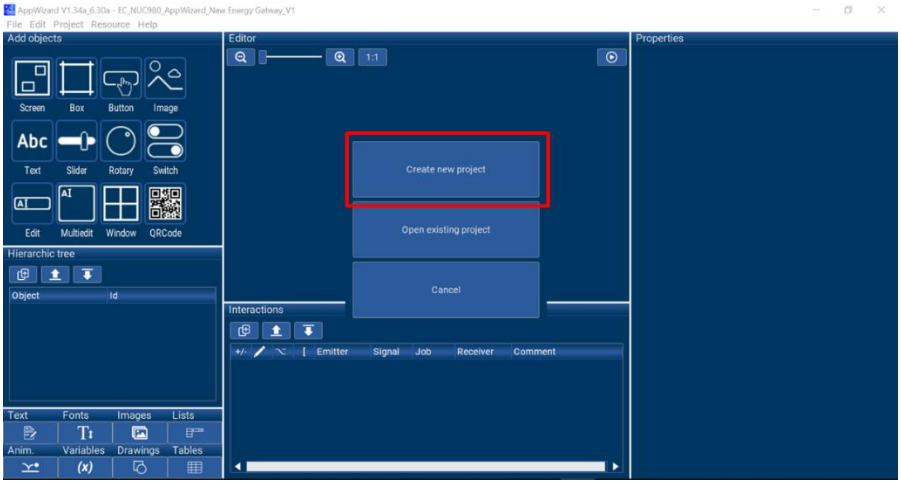


圖 2-4 新專案建立頁面

3. 依圖 2-5 設定專案路徑、名稱，符合使用屏幕的大小(240x320)及顏色格式：16 Bit, GUICC_M565，勾選 “Enable Multibuffering”後點選 “ok”開啟專案設計畫面 。

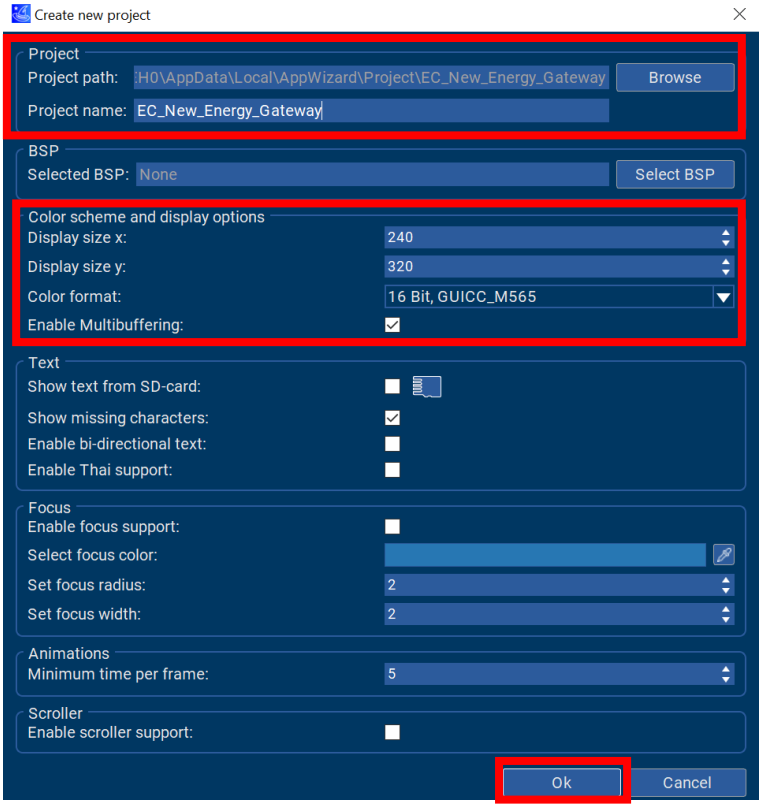


圖 2-5 新專案設定項目

4. 於 Add objects 中點選五次 Screen 創建五個畫面，並在 Hierarchic tree 內將 Screen 分別命名為 ID_SCREEN_Home、ID_SCREEN_02~ID_SCREEN_05，以利辨認。

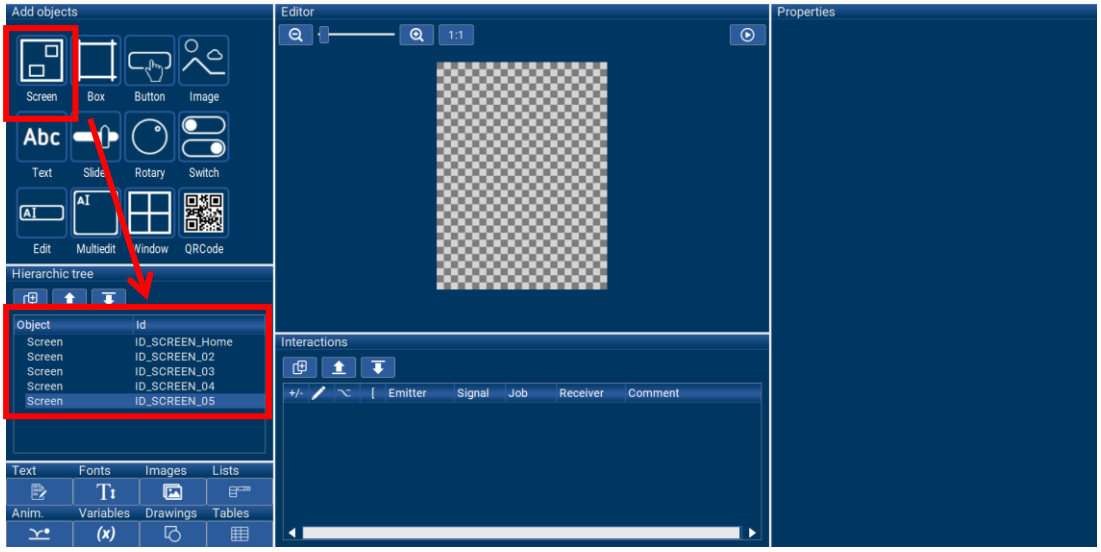


圖 2-6 新增 Screen

5. 點選畫面左下角”Image”，將此專案使用的 PNG 圖片導入 AppWizard。

(檔案位於：EC_NUC980_AppWizard_New_Energy_Gateway_V1.00\EC_New Energy Gateway\Resource\Image)

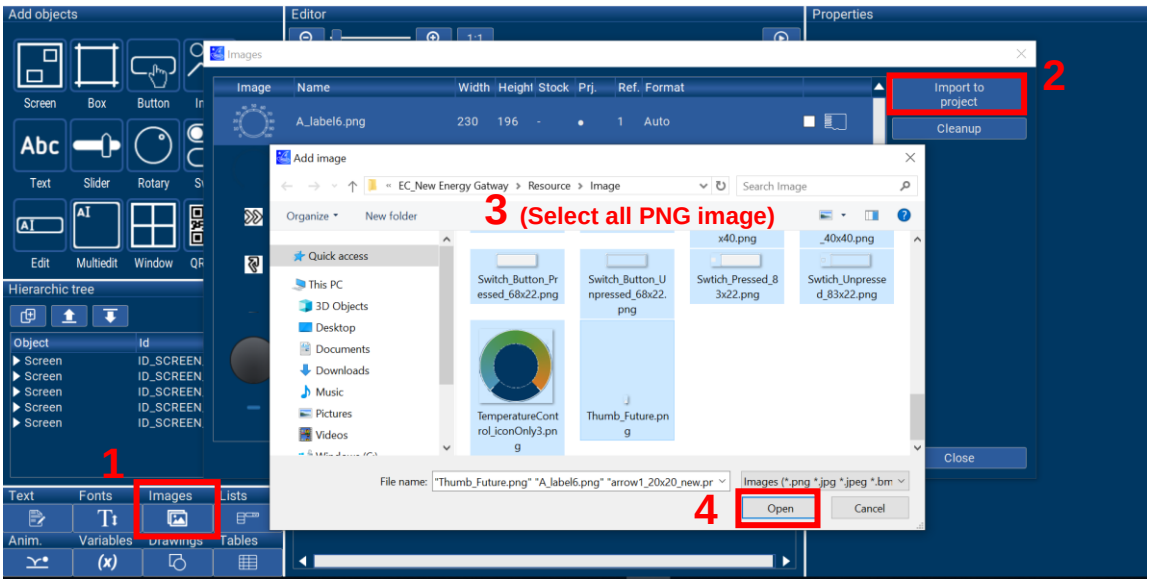


圖 2-7 導入 PNG 影像至 AppWizard

6. 分別點選 ID_SCREEN_Home、ID_SCREEN_02~ ID_SCREEN_05，透過 Add objects 中的 Box 新增背景，並透過 box 屬性欄位的”Set color”調整顏色。

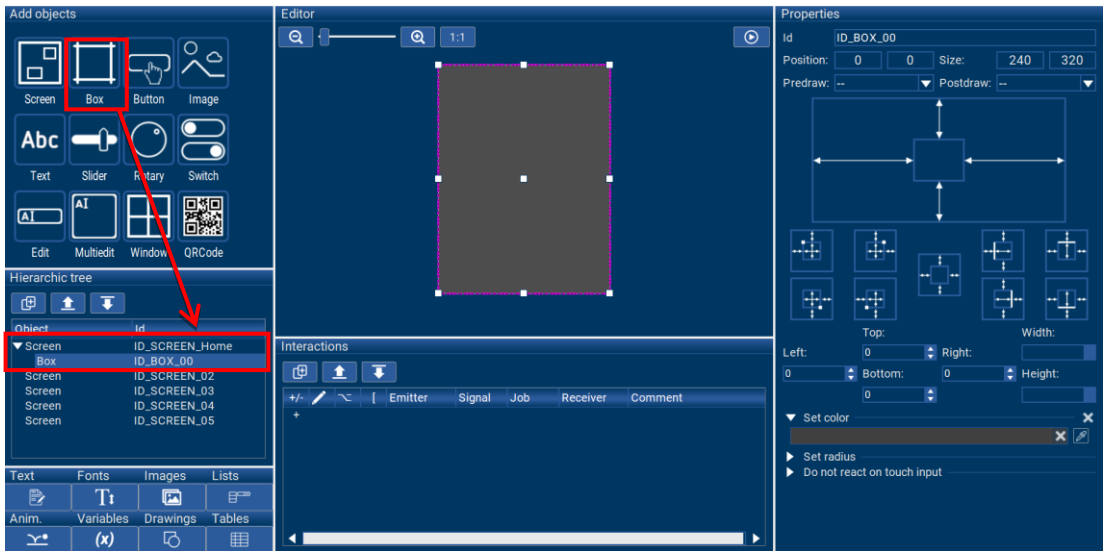


圖 2-8 新增 box 物件作為背景

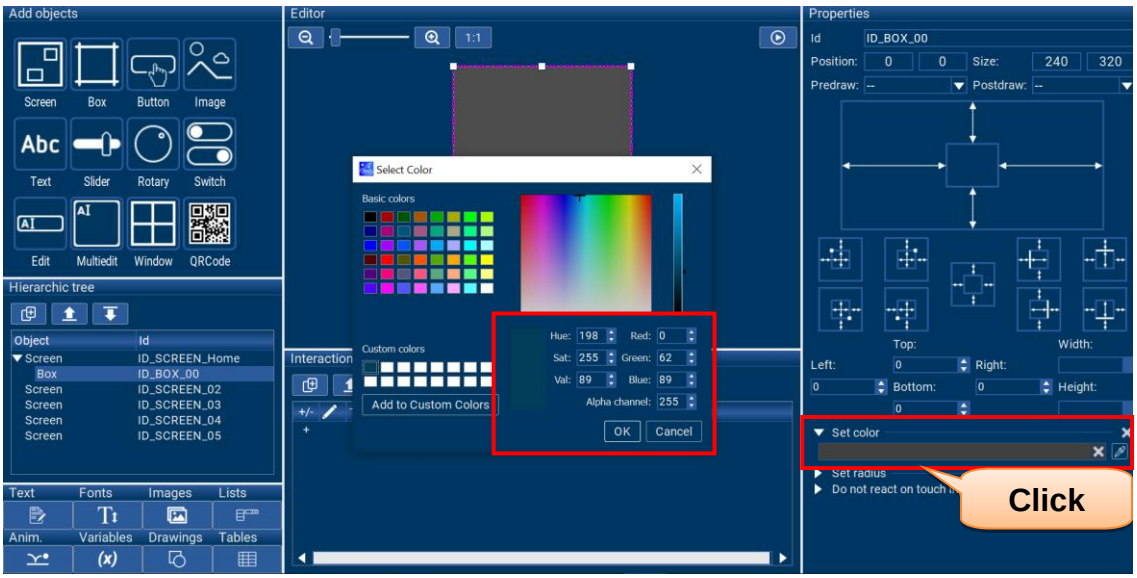


圖 2-9 設置 box 顏色

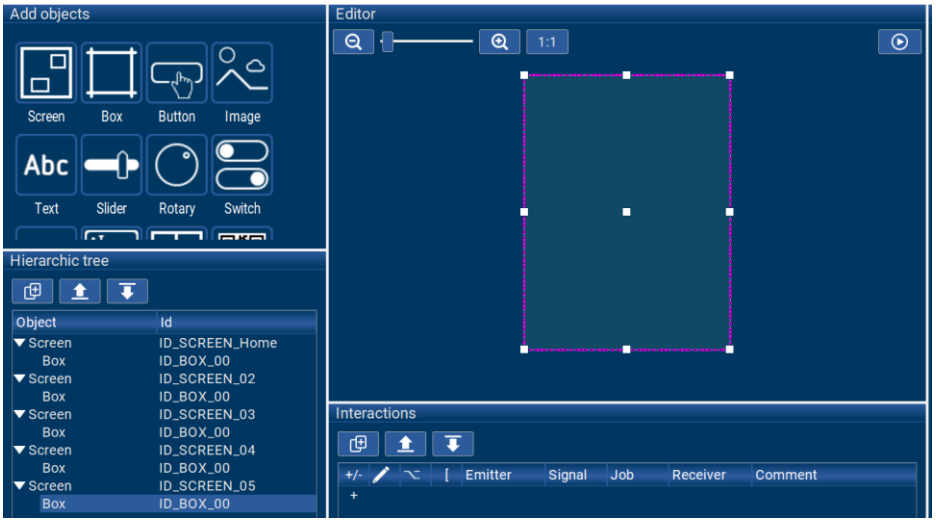


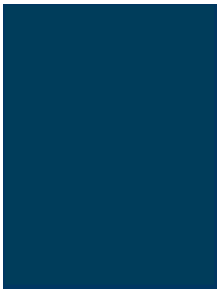
圖 2-10 新增所有畫面的純色背景

2.3 AppWizard 頁面設計實作

新能源閘道器(New Energy Gateway)應用廣泛，常見應用如儲能、充電樁、光伏逆變器與電流量表等多種工業應用，本範例將針對幾種常見應用進行 GUI 設計，並詳述各畫面所使用的物件、互動設置及動畫設計，使用者可依需求再自行調整。

2.3.1 電量表設計(主畫面)

電池量表的設計可應用在充電樁等需要顯示用電量的產品，此設計使用的物件如表 2-1 所示。

New Energy Gateway Description	Application GUI	Widget Type
背景圖 240x320		Box
電量刻度背景 (置於 Rotary 下層)		Image

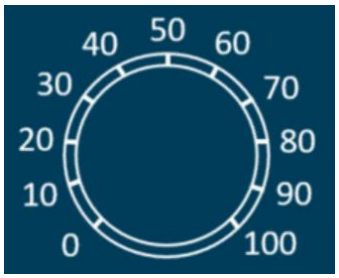
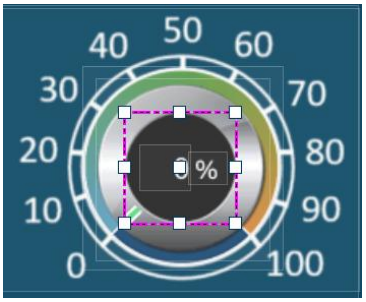
Rotary1 背景圖片		Rotary
Rotary1 Maker		
電量刻度表		Rotary
電量變色區塊		Box
頁面標題		Text
電量數值		
頁尾註記		
下一頁按鈕		Button

表 2-1 電量表設計物件

2.3.1.1 GUI 物件配置

1. 點選 ID_SCREEN_Home 開始編輯主頁畫面，透過 Add objects 新增一個大小為 123x123 的 image 物件(ID_IMAGE_00)，並選擇圖片 ColorVariety.png 作為 bitmap 設置。

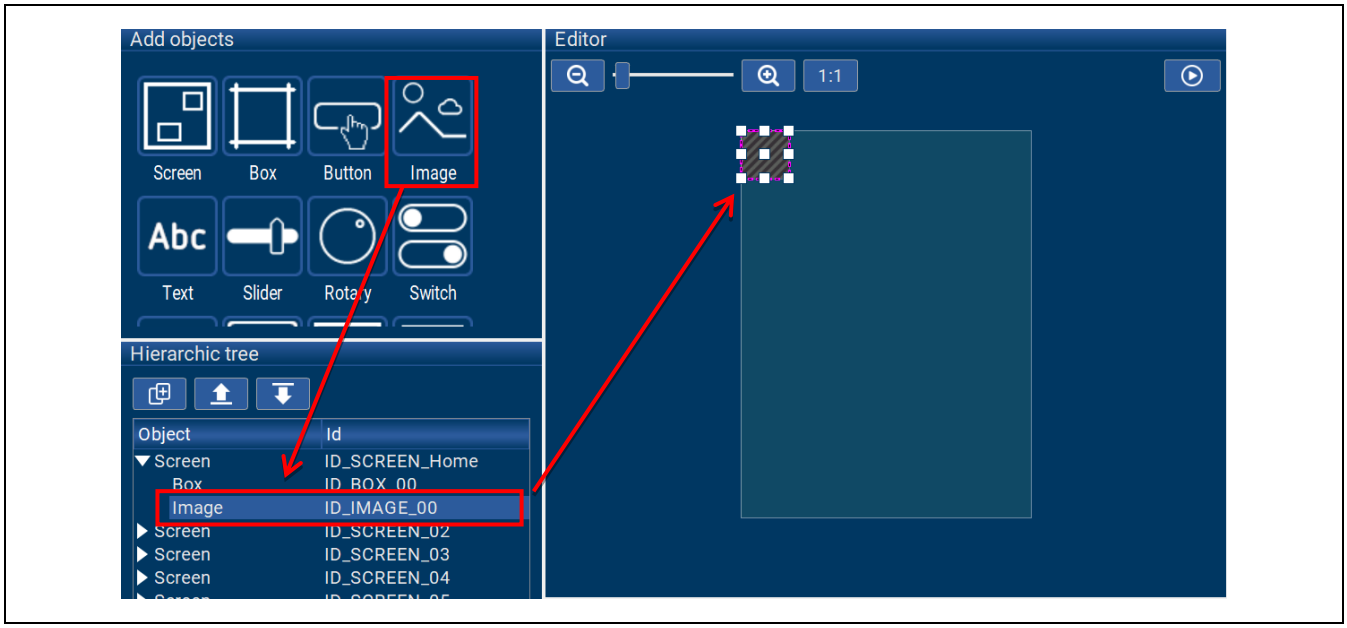


圖 2-11 新增 image 物件

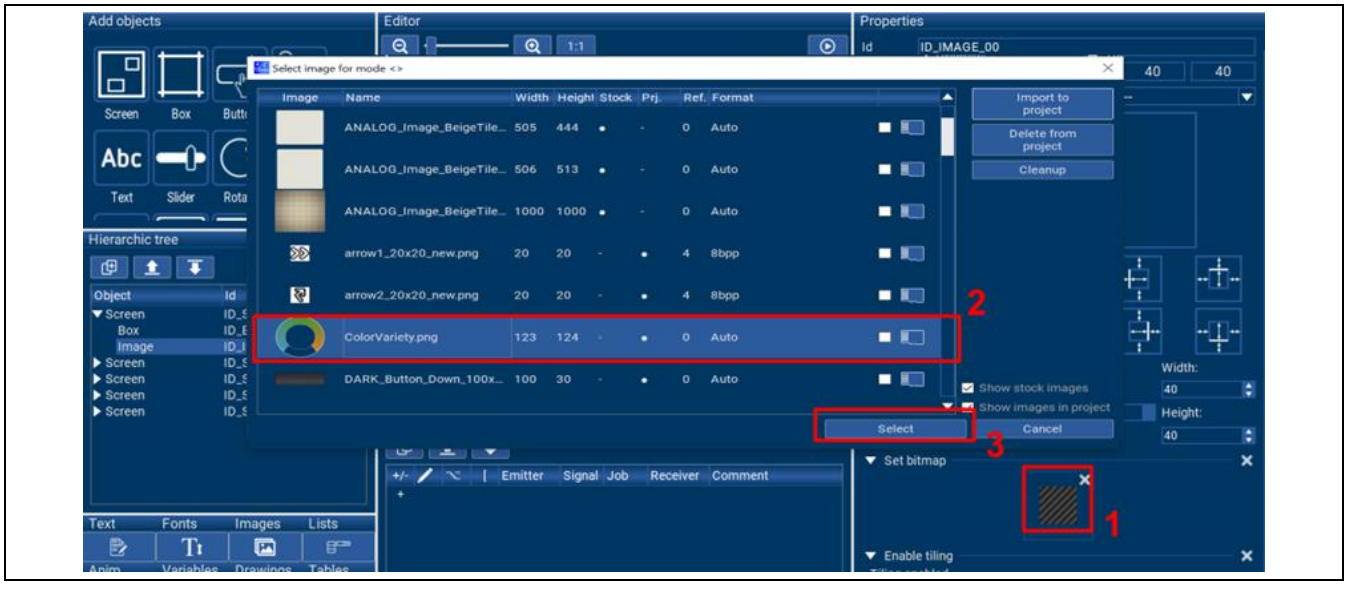


圖 2-12 物件 bitmap 設置方式

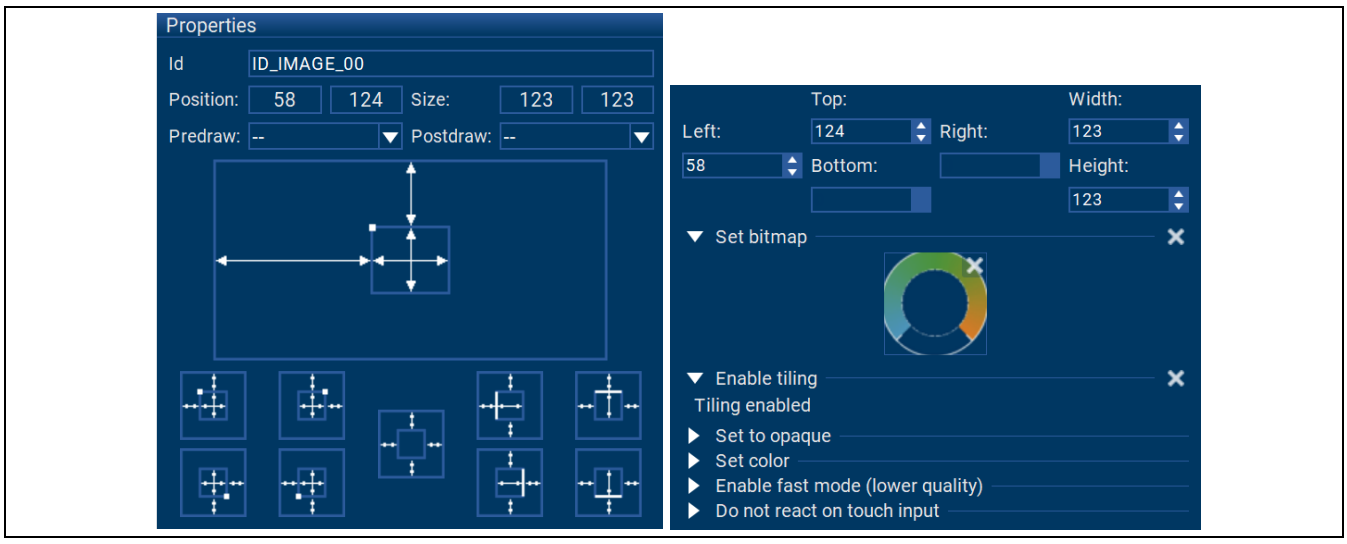


圖 2-13 ID_IMAGE_00 屬性設置

2. 新增大小為 219x181 的 Rotary 物件(ID_ROTARY_idx)疊在 image 物件上，選擇圖片 scaleValue.png 做為 bitmap，並取消 Maker 使用。

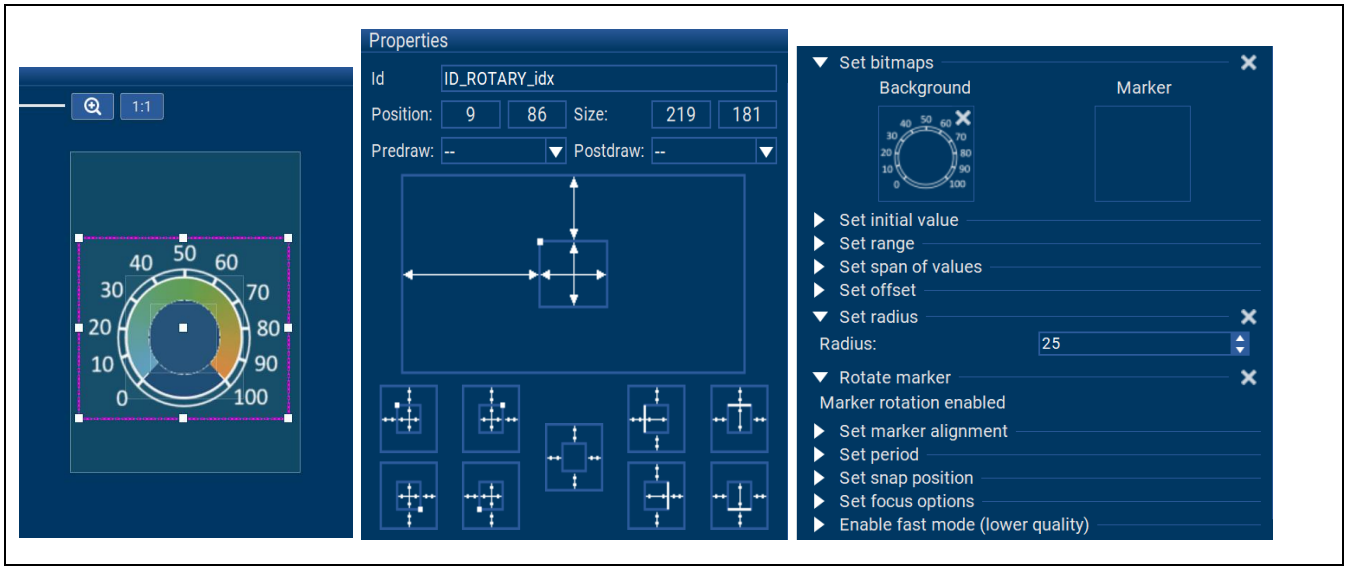


圖 2-14 ID_ROTARY_idx 屬性設置

3. 新增一個大小為 69x69 的 box 物件(ID_BOX_color)作為變色區塊(取消預設色彩)，並疊放在 ID_ROTARY_idx 上方。

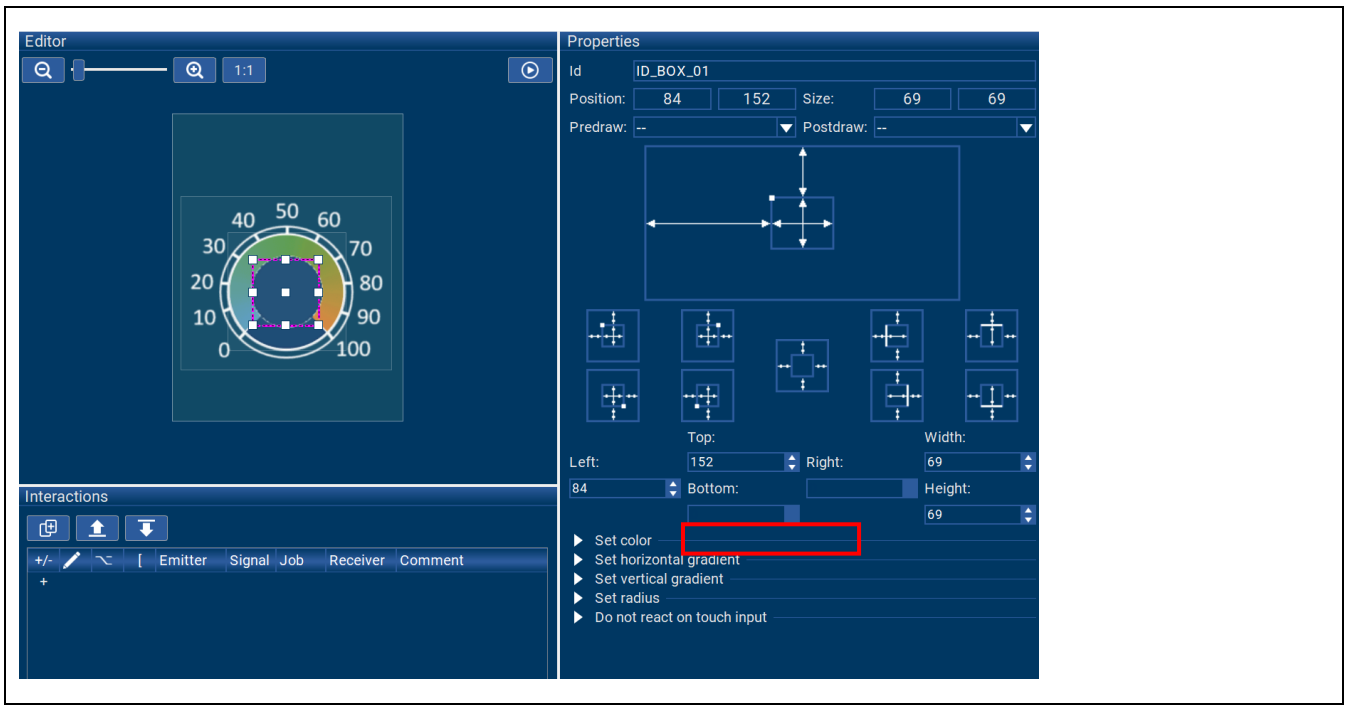


圖 2-15 ID_BOX_color 屬性設置

4. 再新增一個大小為 107x112 的 Rotary(ID_ROTARY_val)疊在 ID_BOX_color 物件之上，作為數值轉輪，並依表 2-2 設置 Rotary 屬性。

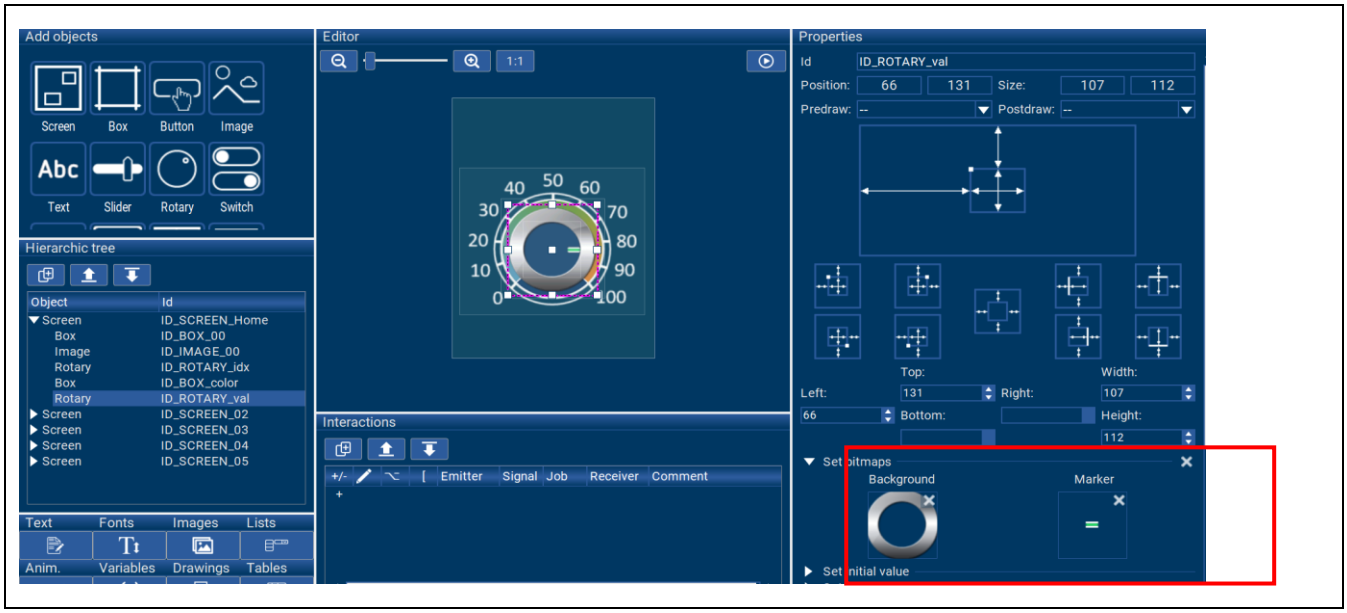


圖 2-16 Rotary Bitmaps 設置

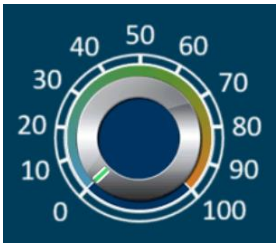


圖 2-17 Rotary Properties 設置結果

Properties	Descriptions	Setting values
Initial Value	Starting speed value	0
Span of values	Range of speed value	Min: 0 Max: 999
Range	Range of rotation angle and direction	Positive: 1325 Negative: 1350 (measured in 10th of degrees)
Offset	Starting angle	900 (measured in 10th of degrees)
Radius	Radius of the Rotary.	43
period	Period how long the marker moves when released.	300

表 2-2 Rotary 屬性設定說明

5. 新增文字以顯示標題與說明文字，在 Add objects 中點選兩次 Text 物件。

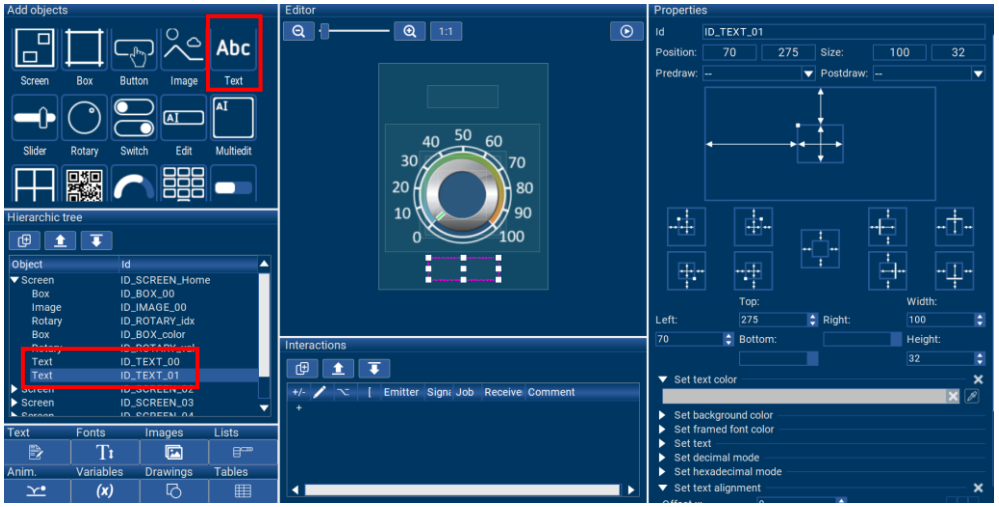


圖 2-18 新增 Text 作為標題與說明文字

1) 依圖 2-19~圖 2-20 步驟，建立文字顯示的語言標籤並新增 ID_TEXT_00 顯示文字。

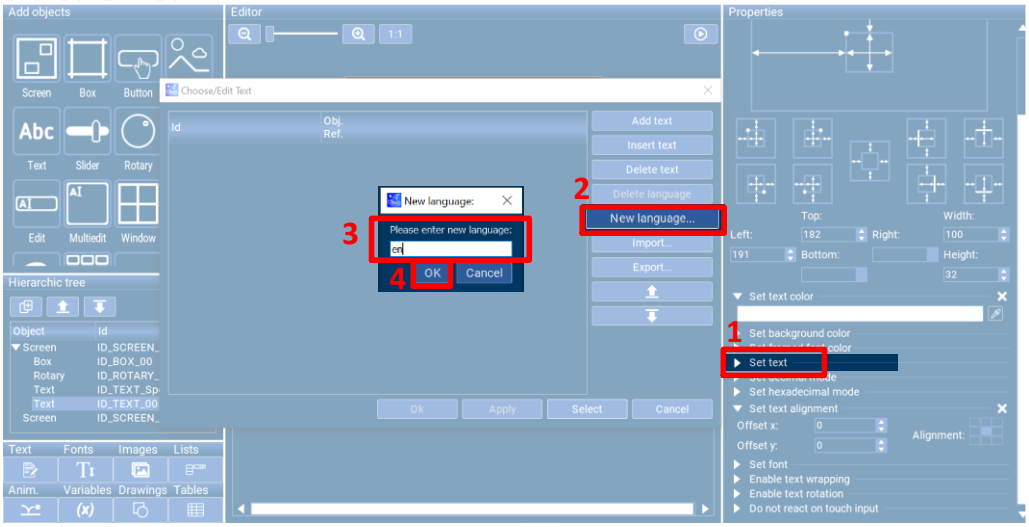


圖 2-19 Text 物件 - 創建語言(英文)

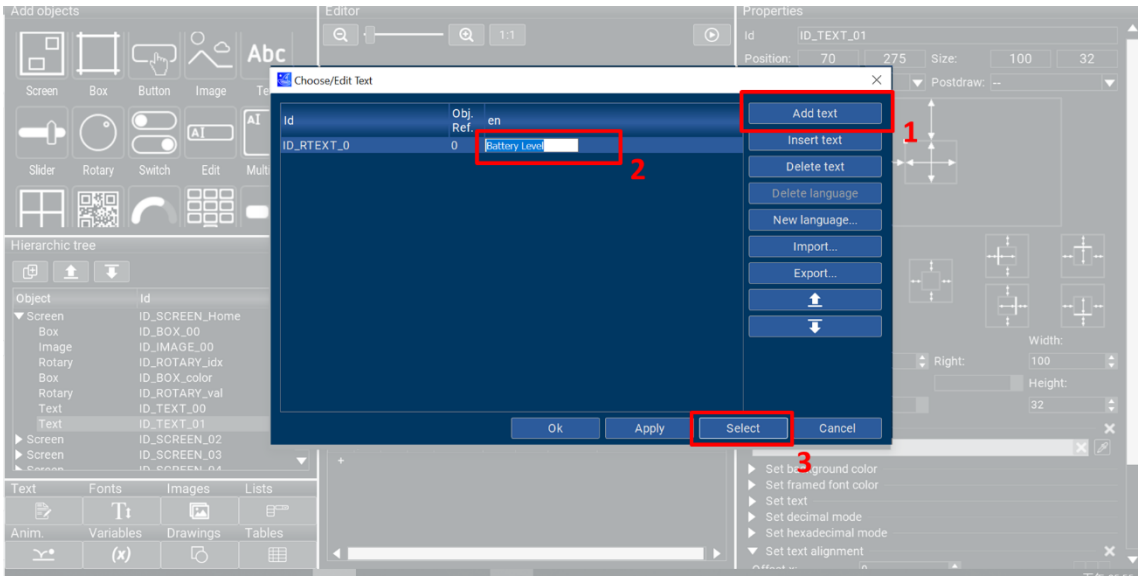


圖 2-20 Text 物件- 編輯標題文字內容

2) 依圖 2-21~圖 2-23 步驟，設定標題文字的顏色、背景與 text alignment。

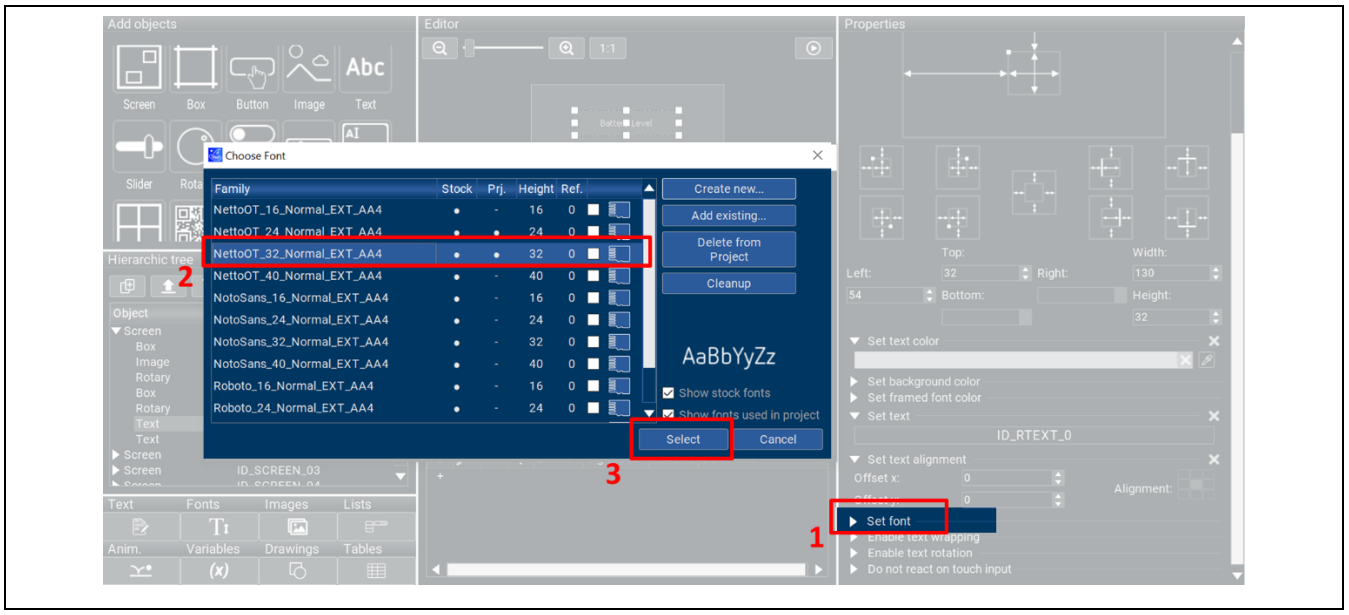


圖 2-21 Text 物件- 標題格式設定

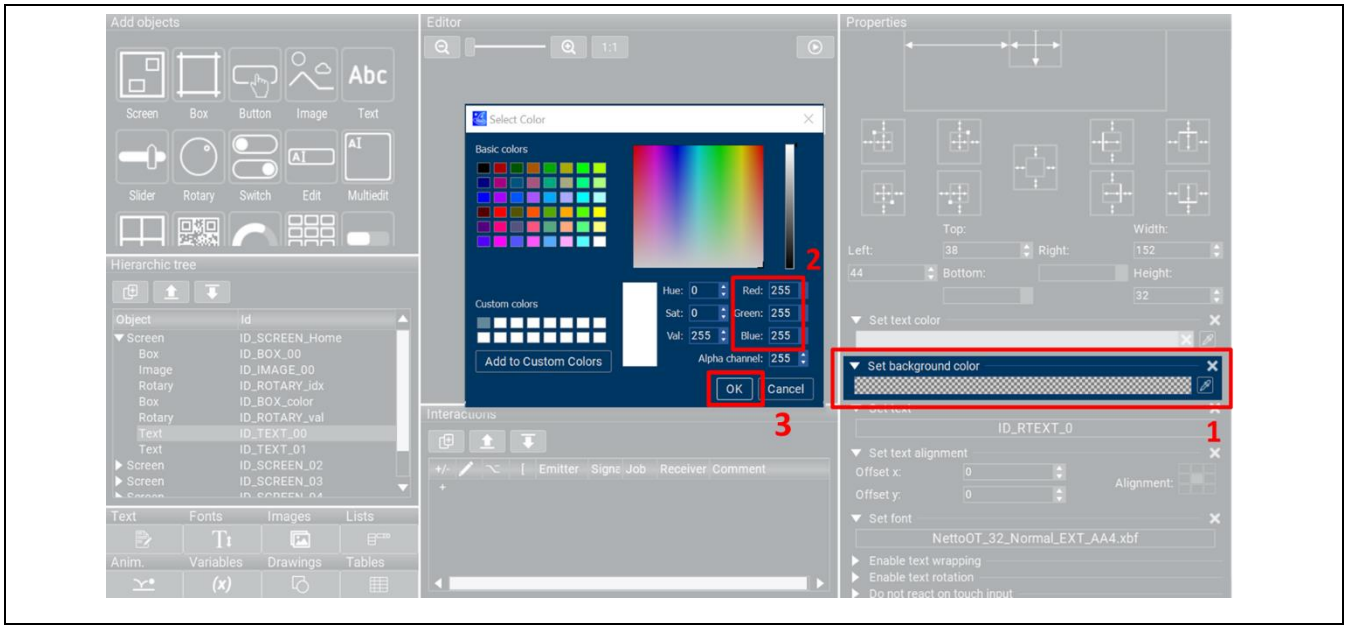


圖 2-22 Text 物件-標題背景顏色設置(白色)

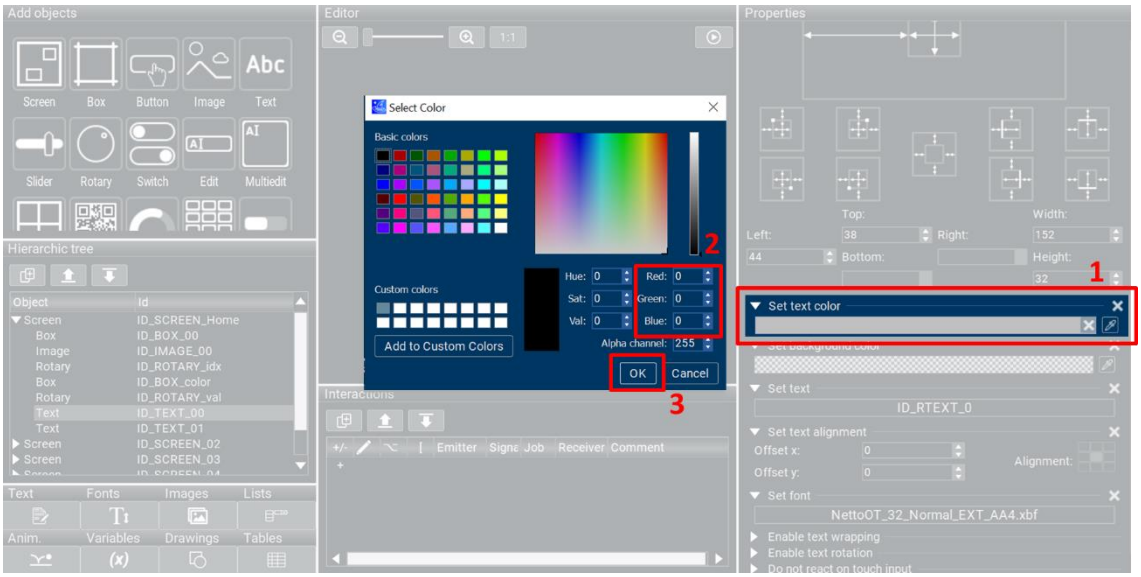


圖 2-23 Text 物件-標題顏色(黑色)

3) 用相同方式設定說明文字內容，設定值如圖 2-24~圖 2-25 所示。

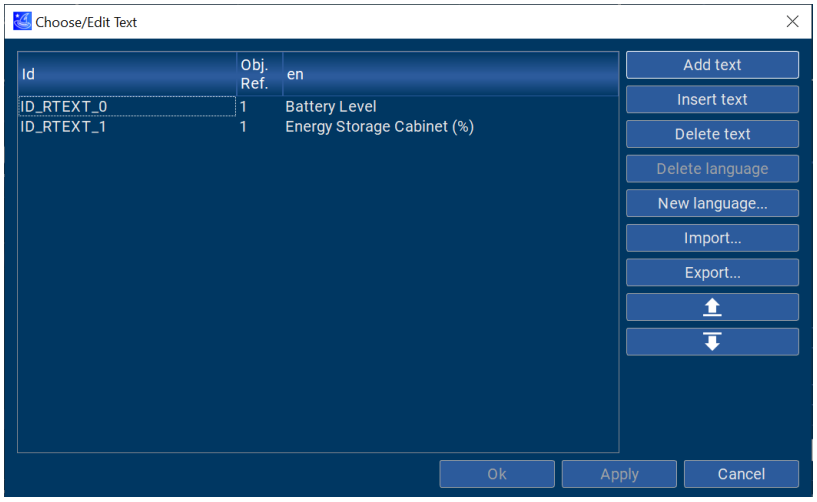


圖 2-24 Text 物件-下標文字內容設定

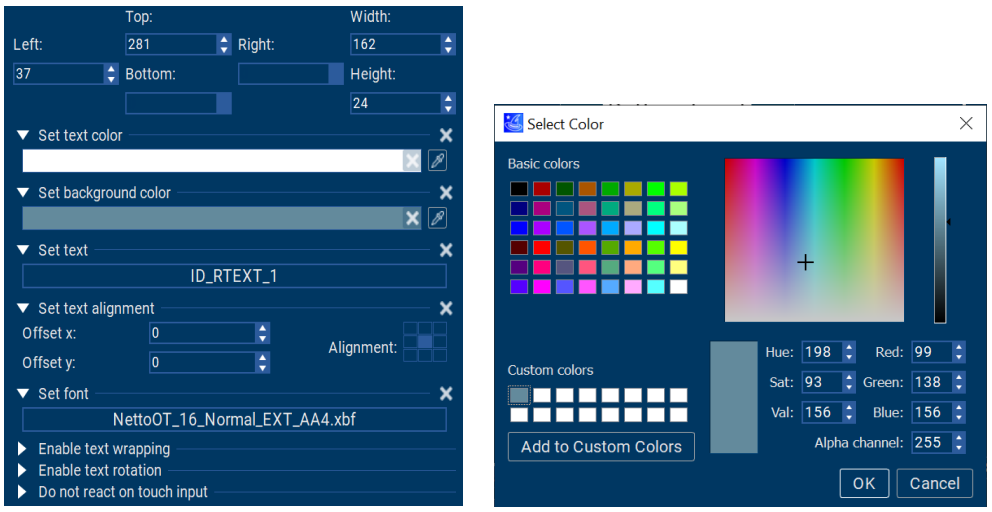


圖 2-25 Text 物件-說明文字格式設定

6. 新增換頁按鈕，在 Add objects 中，點選新增一個 Button 物件並命名為 *ID_BUTTON_next*，依圖 2-26~圖 2-27 設置其 bitmap 與屬性。

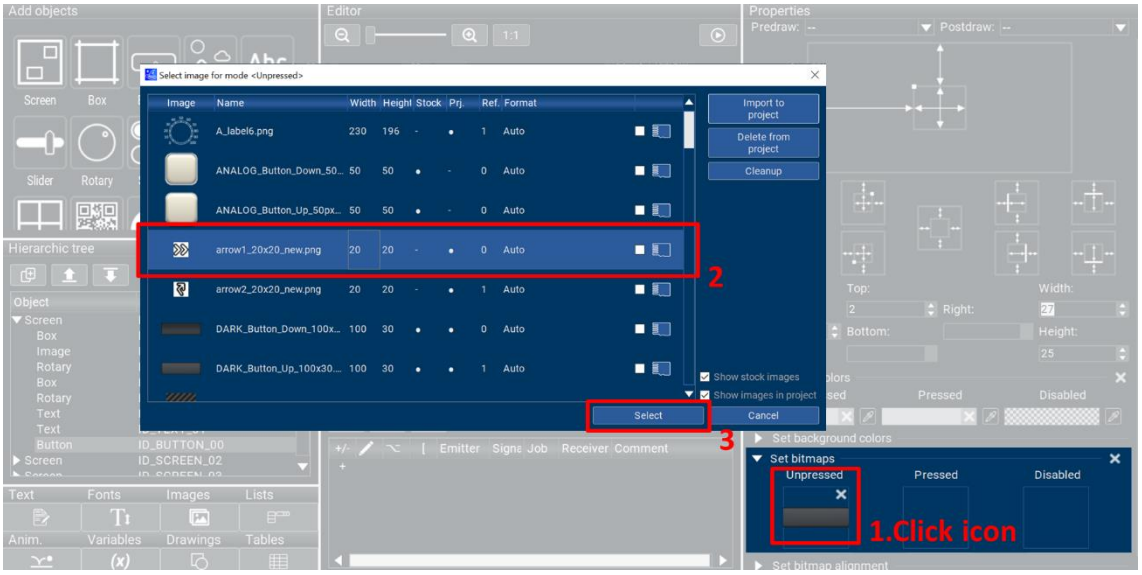


圖 2-26 換頁按鈕外觀設置

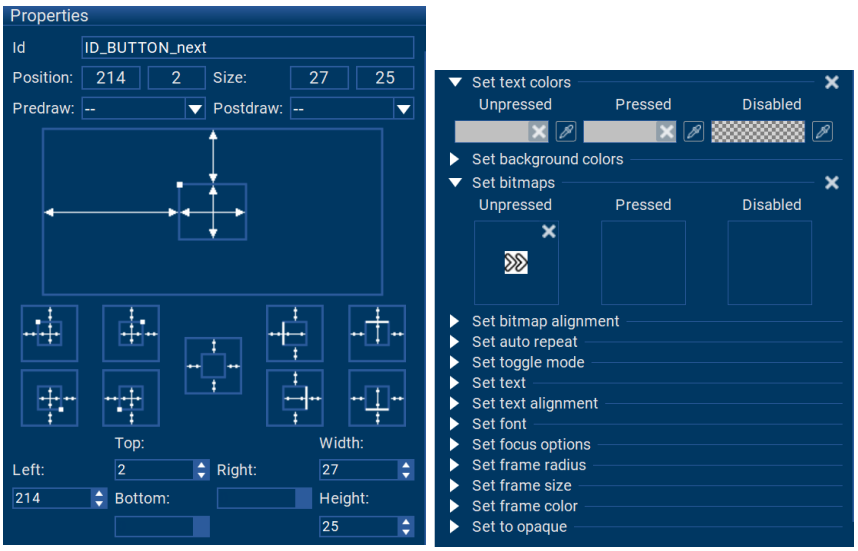


圖 2-27 ID_BUTTON_next 屬性設置

7. 在 Add objects 中，新增兩個 TEXT 物件分別顯示 Rotary 數值(ID_TEXT_TEMP)及單位符號(ID_TEXT_SIGN)，並依圖 2-28~圖 2-30 步驟設置其屬性。

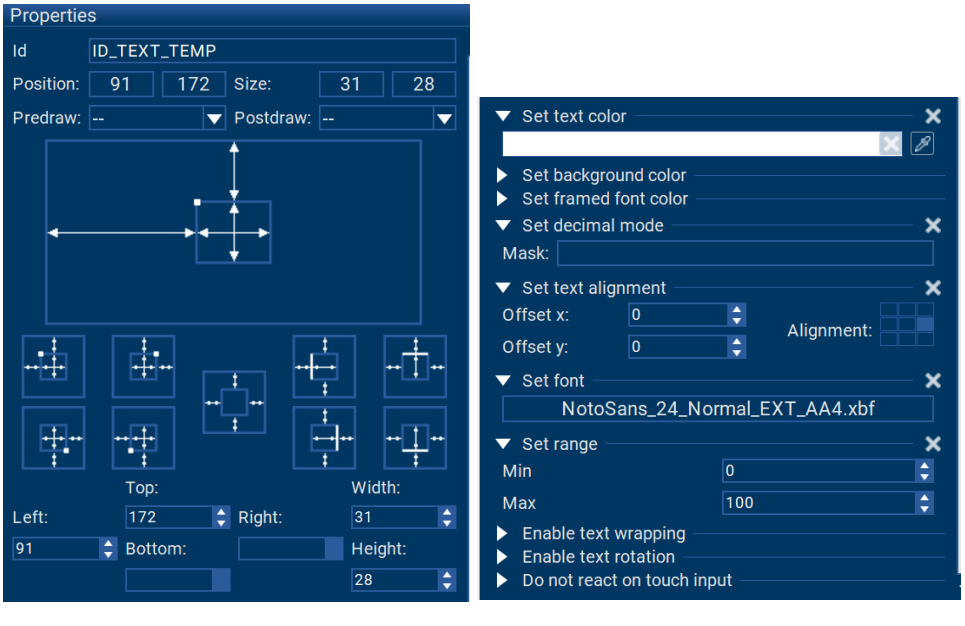


圖 2-28 ID_TEXT_TEMP 屬性設置

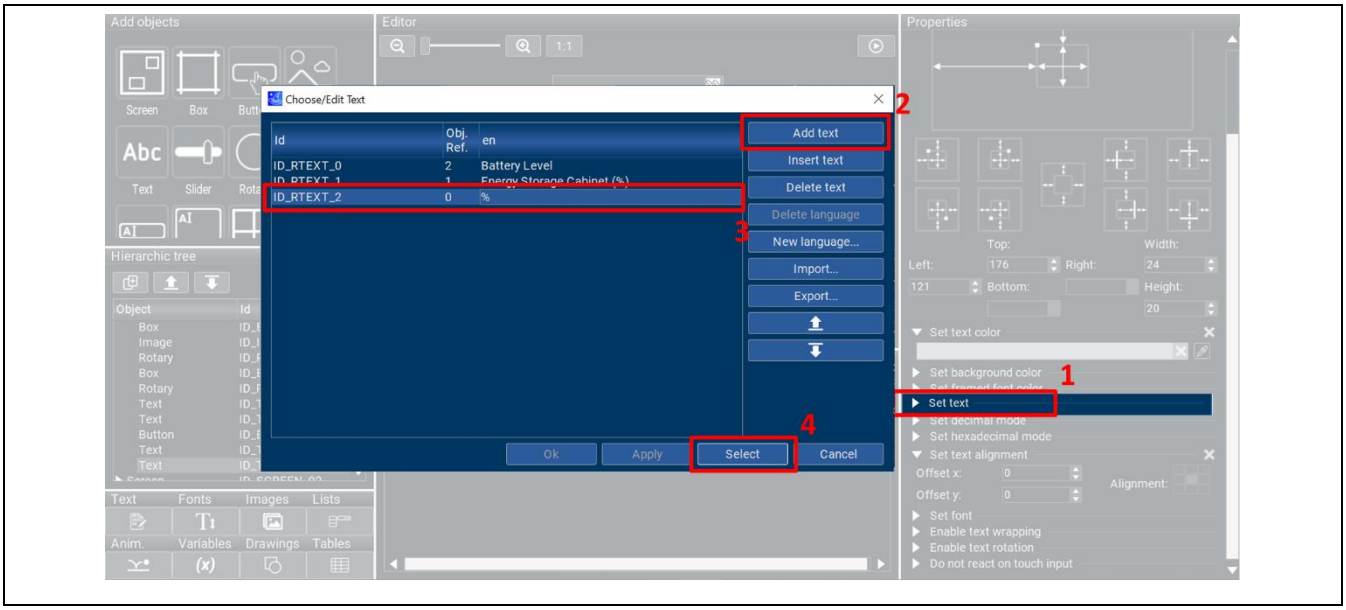


圖 2-29 ID_TEXT_SIGN 文字設置

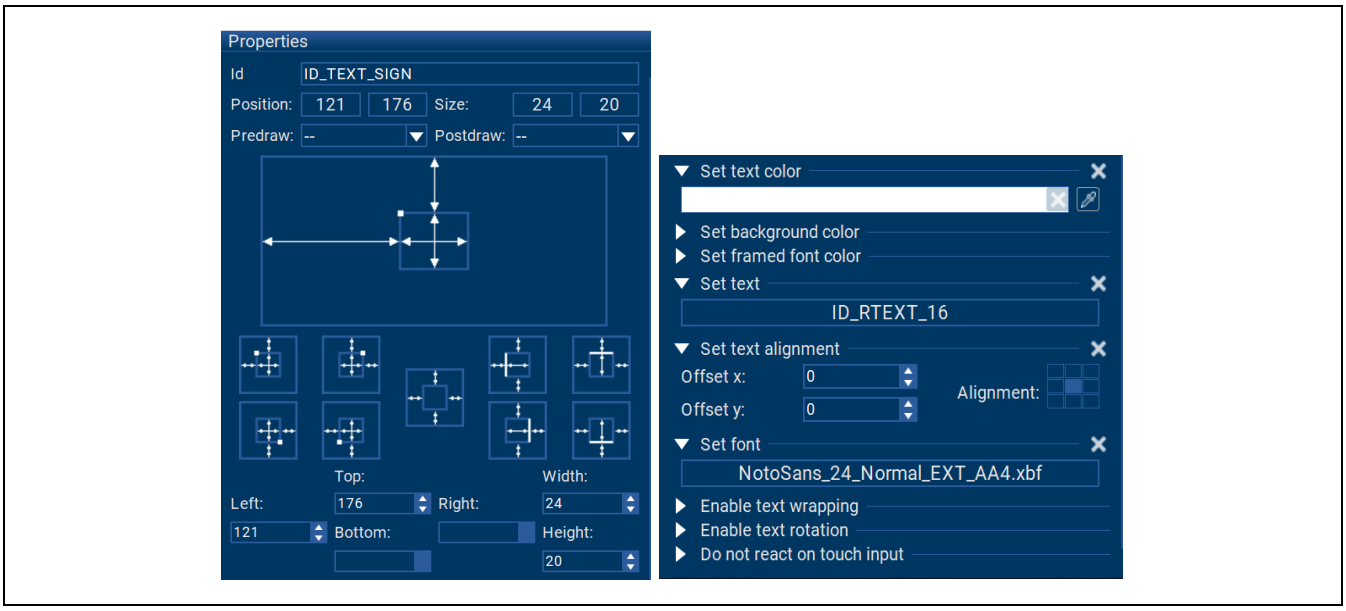


圖 2-30 ID_TEXT_SIGN 屬性設置

2.3.1.2 物件互動設置

物件設置完成畫面如圖 2-31 所示，接著要建立物件間的交互作用(interactions)，使操作物件時有相應行為顯示。

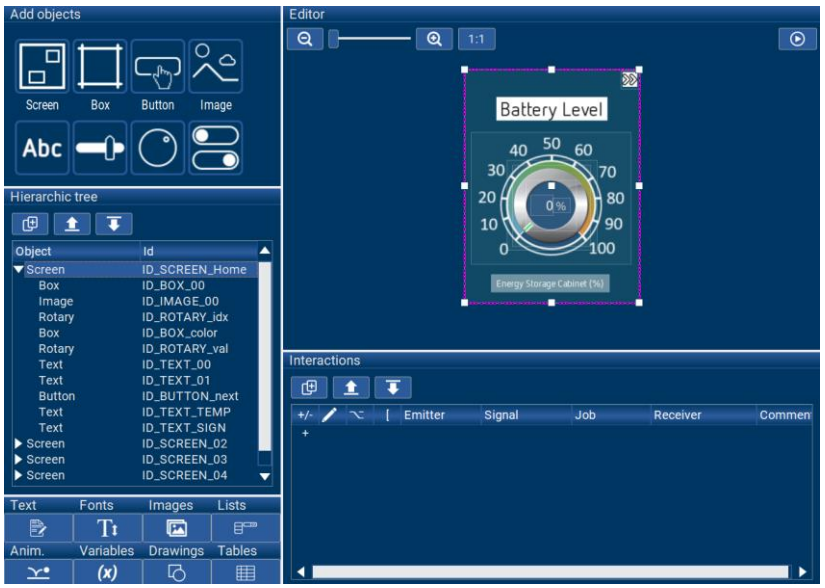


圖 2-31 主畫面設計完成圖

1. 設置 Rotary(ID_ROTARY_val)與 TEXT(ID_TEXT_TEMP)物件的交互作用。

   1) 依圖 2-32~圖 2-35 步驟，透過 Variables 新增變數使用。

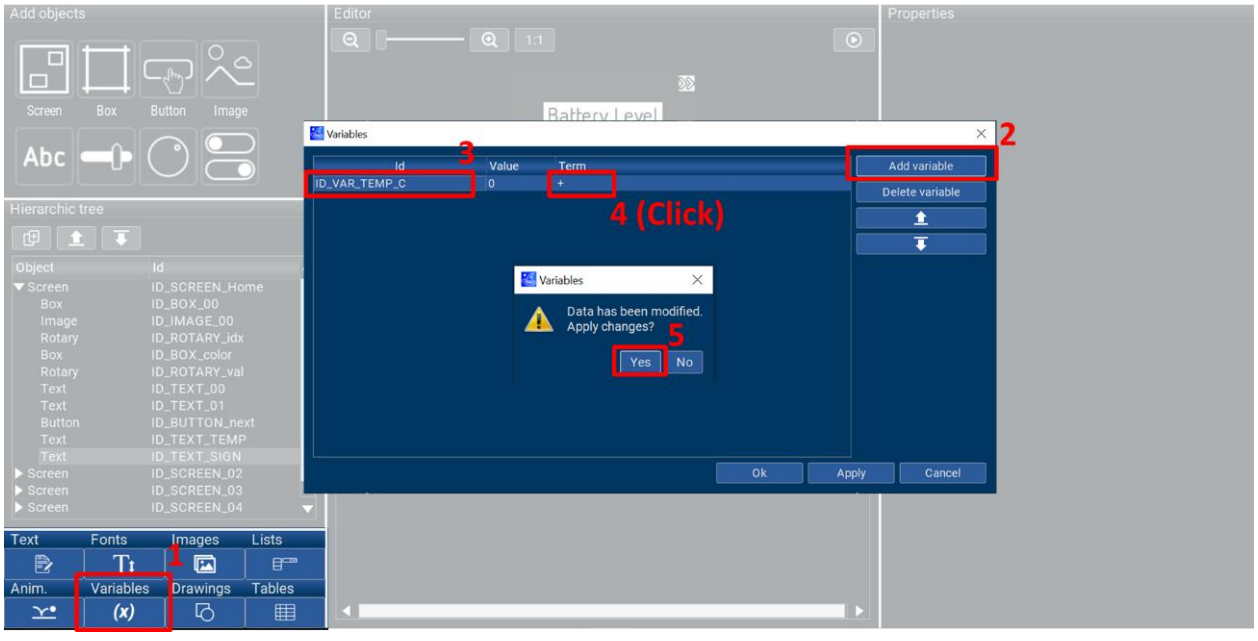


圖 2-32 新增變數 ID_VAR_TEMP_C

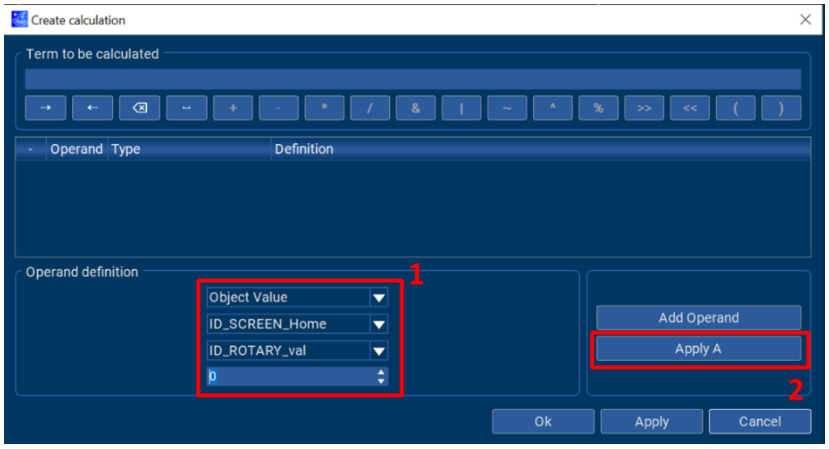


圖 2-33 設置 ID_VAR_TEMP_C 計算值 1

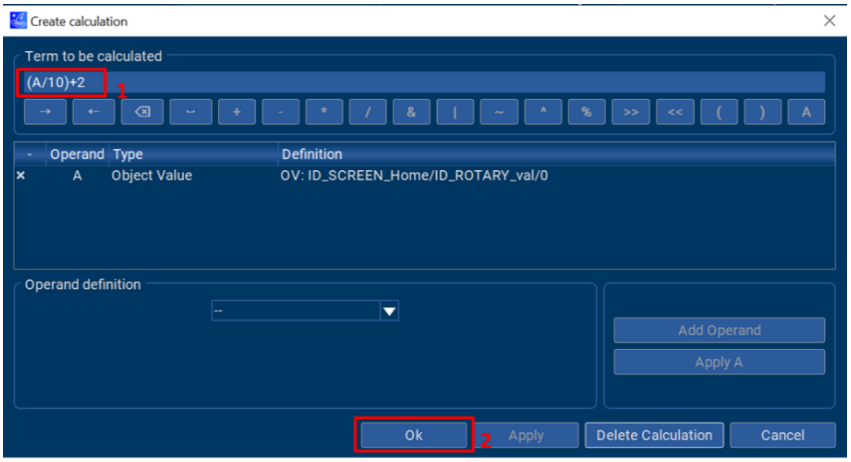


圖 2-34 設置 ID_VAR_TEMP_C 計算值 2

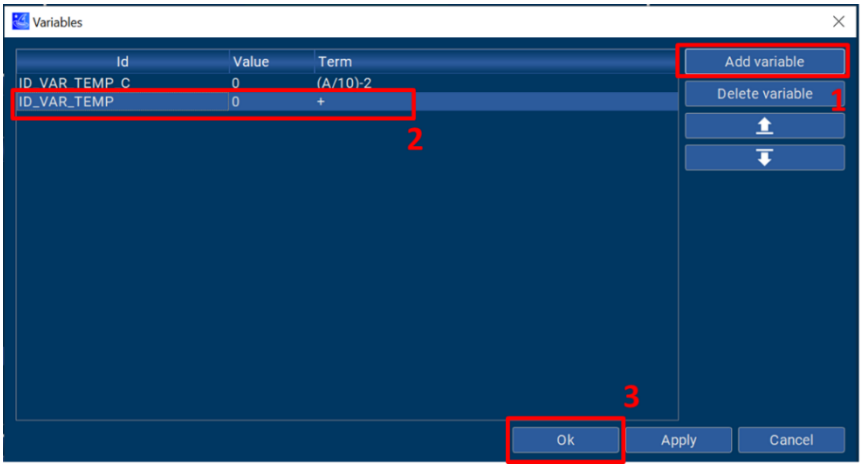


圖 2-35 新增變數(ID_VAR_TEMP)

2) 在 interactions 欄位設置 Rotary 與 TEXT 交互作用，令 Rotary 轉動同時改變 TEXT 顯示值。

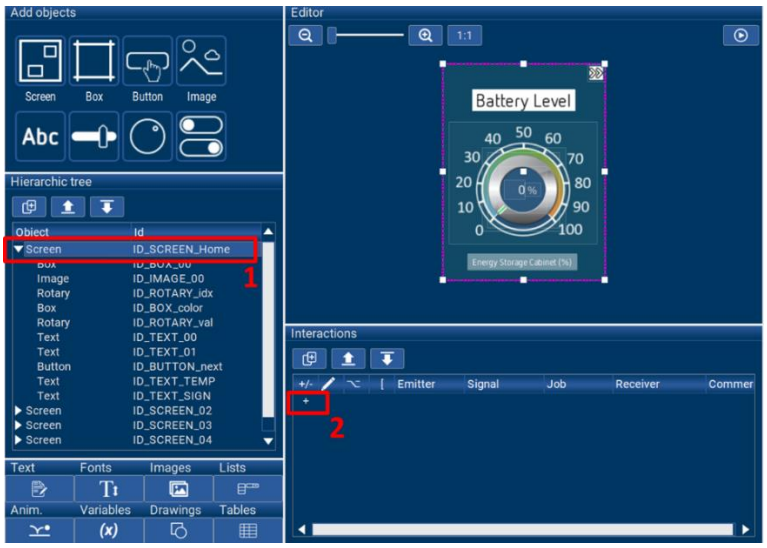


圖 2-36 新增 interactions 項目

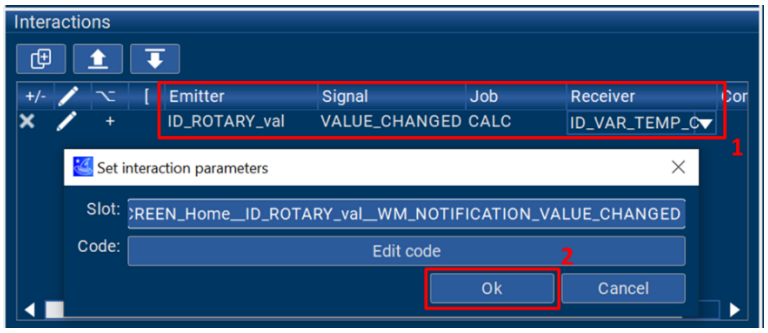


圖 2-37 將 Rotary 數值映射到變數 ID_VAR_TEMP_C

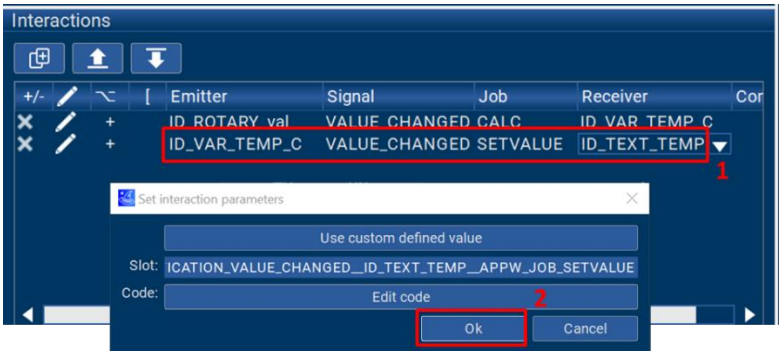


圖 2-38 賦予 ID_VAR_TEMP_C 數值給物件 ID_TEXT_TEMP

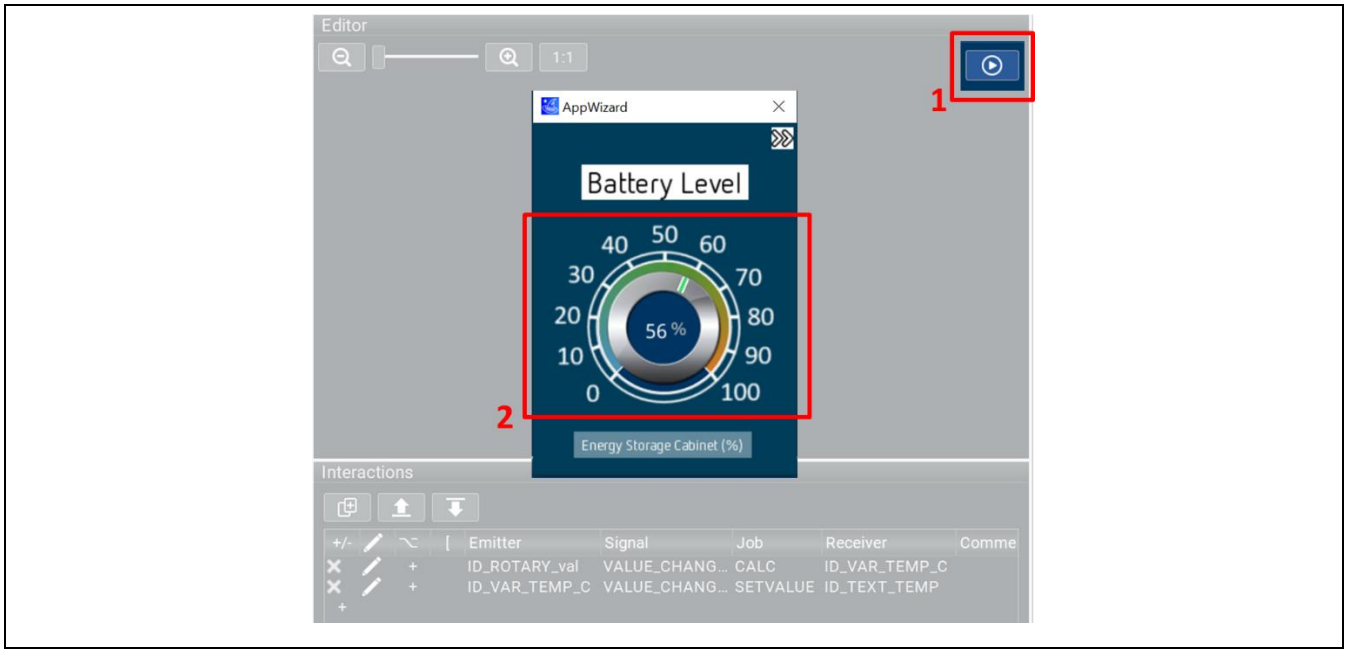


圖 2-39 透過 Start Play Mode 確認 interaction 行為是否正確

2. 設置 Rotary(ID_ROTARY_val)數值對應顏色，使數值變化時影響 Box (ID_BOX_color)色彩。

1) 依圖 2-40~圖 2-43 步驟，新增顏色(ID_VAR_COLOR_BLUE、ID_VAR_COLOR_GREEN、ID_VAR_COLOR_ORANGE)、變色開關(ID_VAR_COLOR_0、ID_VAR_COLOR_1)與色彩變化指針(ID_VAR_COLOR_POS)的初始值。

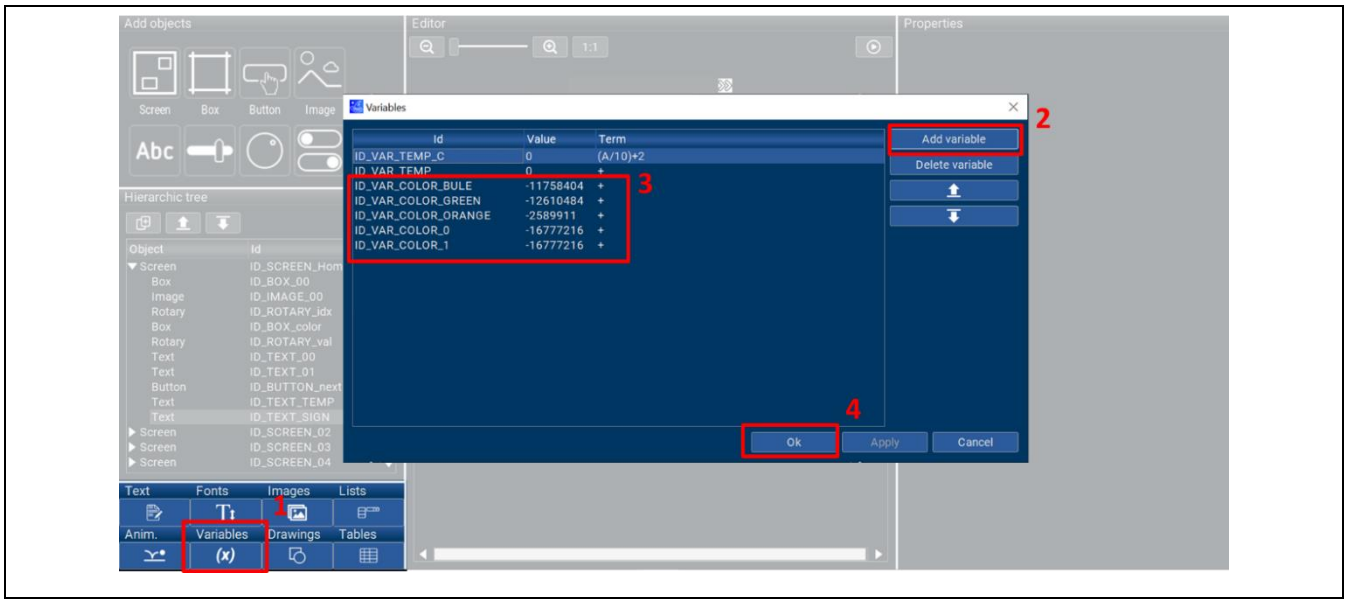


圖 2-40 新增顏色與變色開關變數初始值

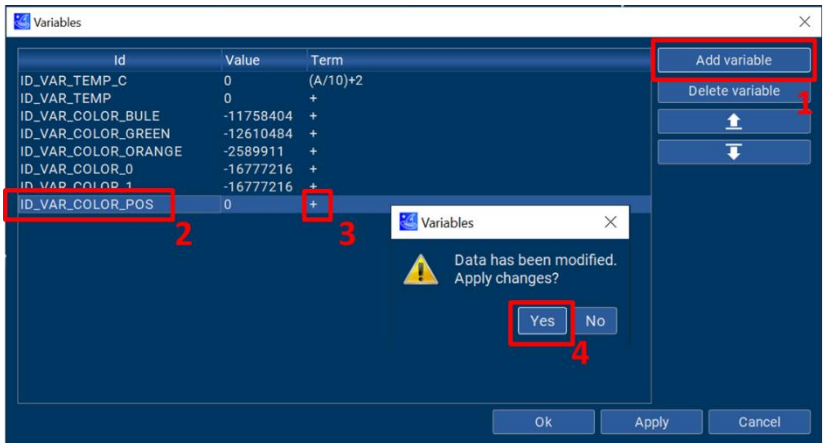


圖 2-41 新增色彩變化指針變數

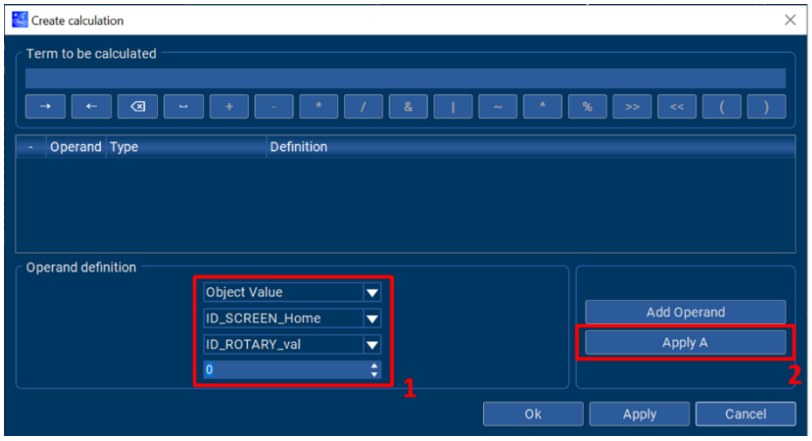


圖 2-42 色彩指針變數設定 1

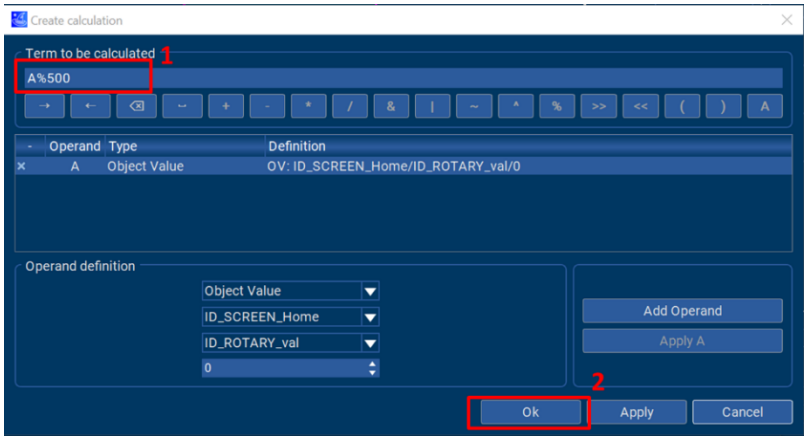


圖 2-43 色彩指針變數設定 2

2) 在 interactions 欄位設置 Rotary 與顏色變化的交互作用，讓 Rotary 轉動同時改變顯示顏色。

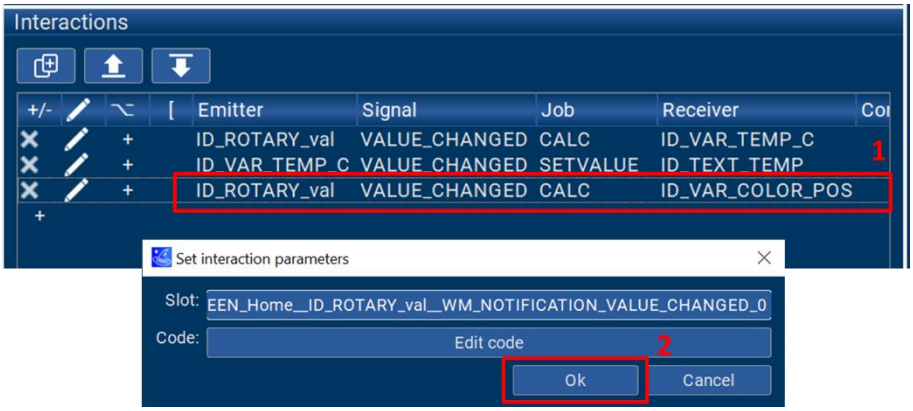


圖 2-44 將 Rotary (ID_Rotary_val)數值賦予給變數(ID_VAR_COLOR_POS)

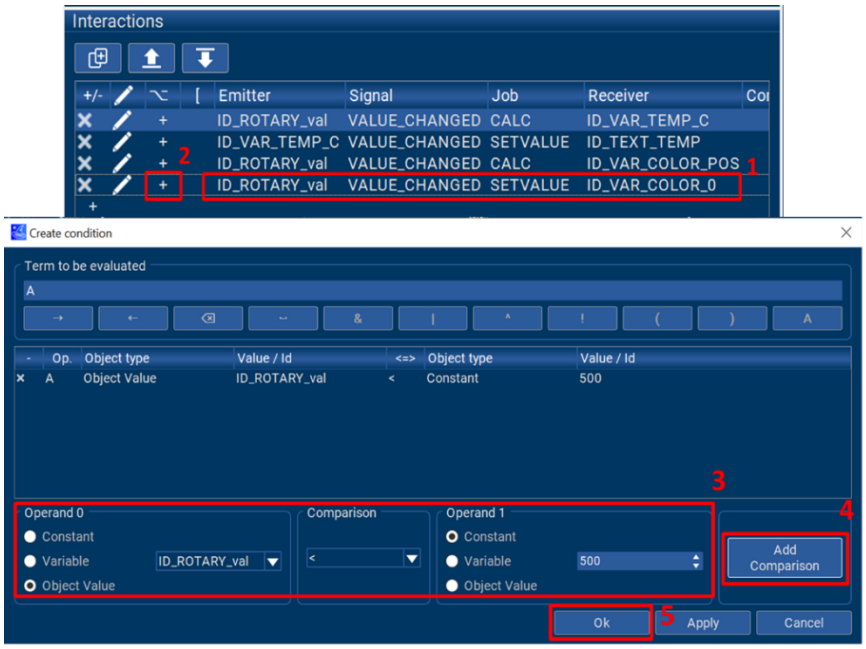


圖 2-45 色彩切換設置 1

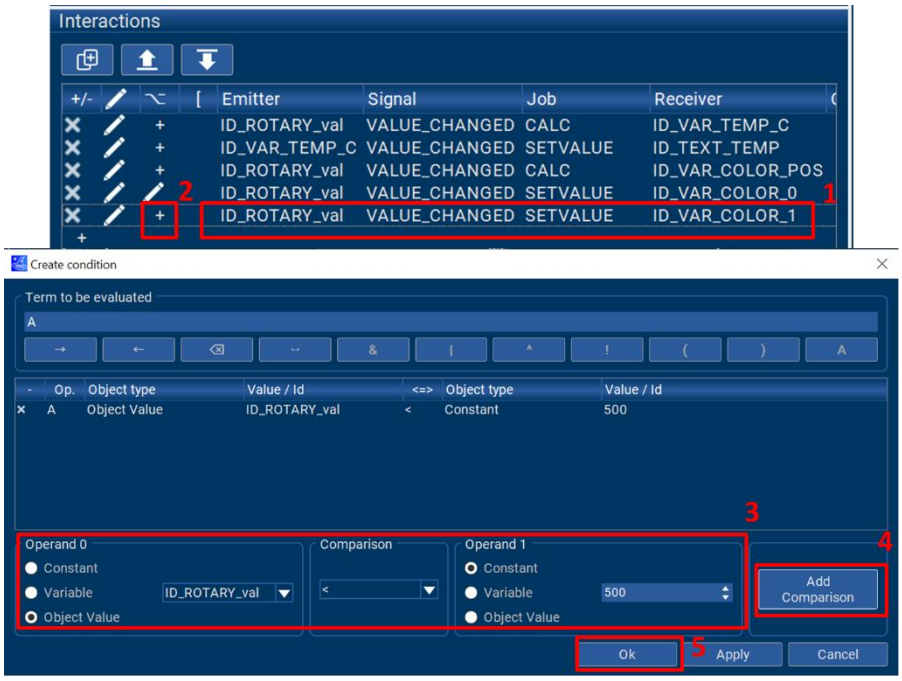


圖 2-46 色彩切換設置 2

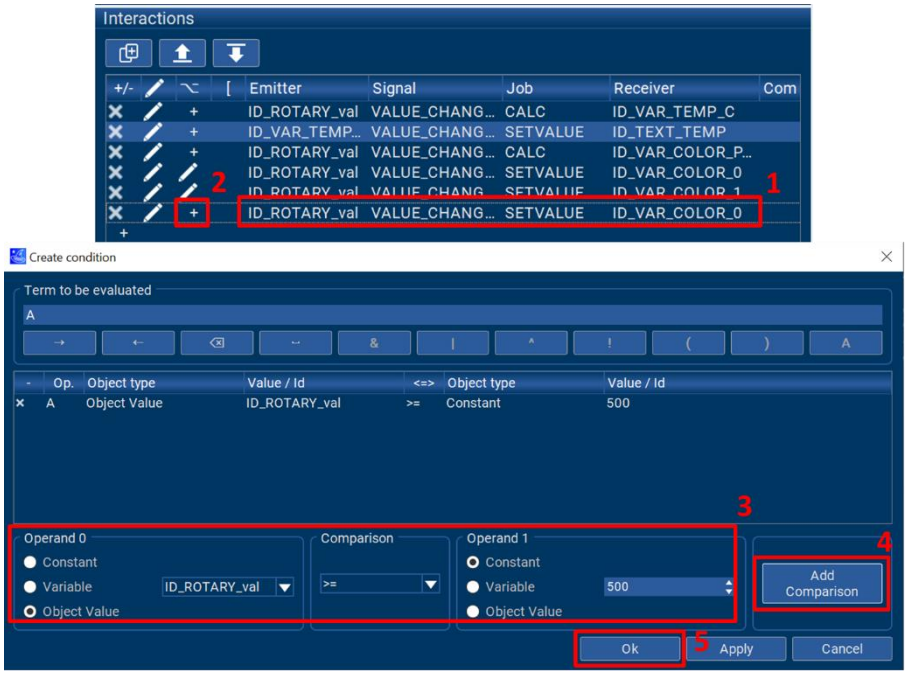


圖 2-47 色彩切換設置 3

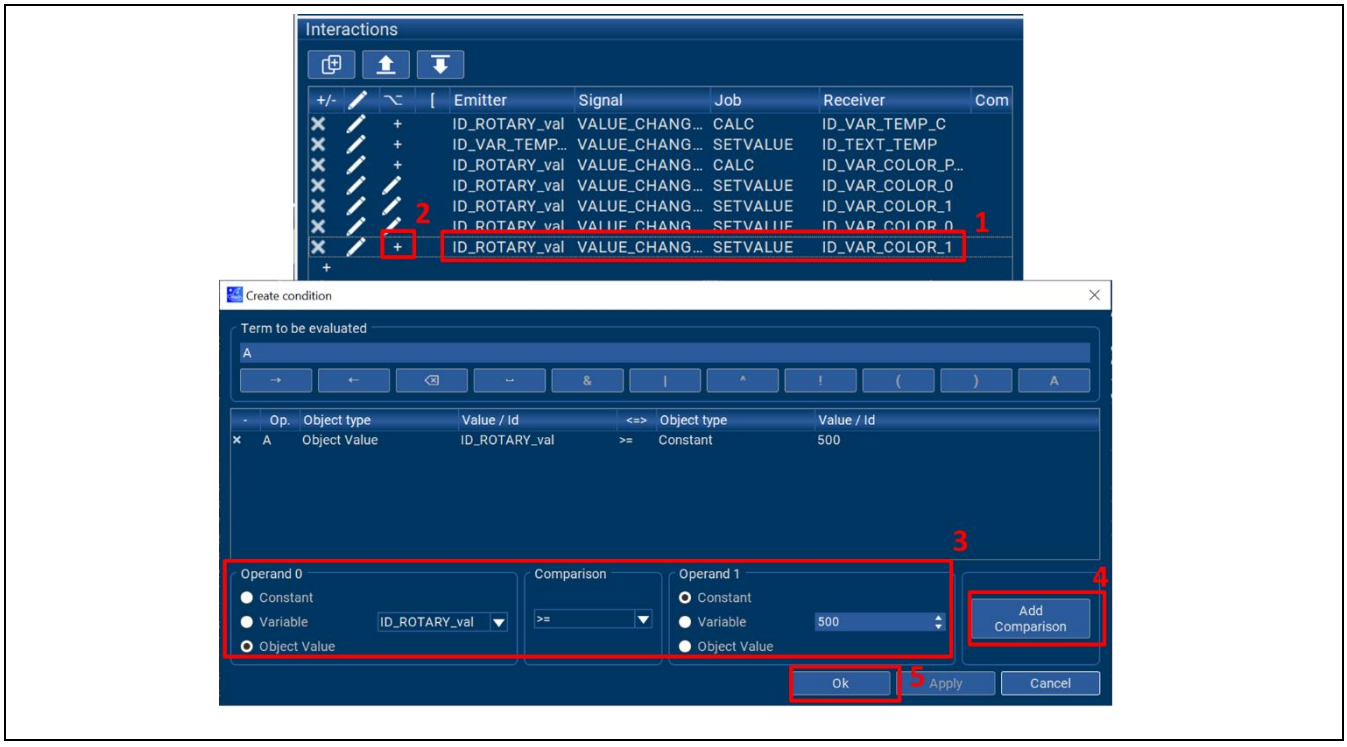


圖 2-48 色彩切換設置 4

3) 依圖 2-49~圖 2-52 的步驟設置每個切換動作的起始變數值。

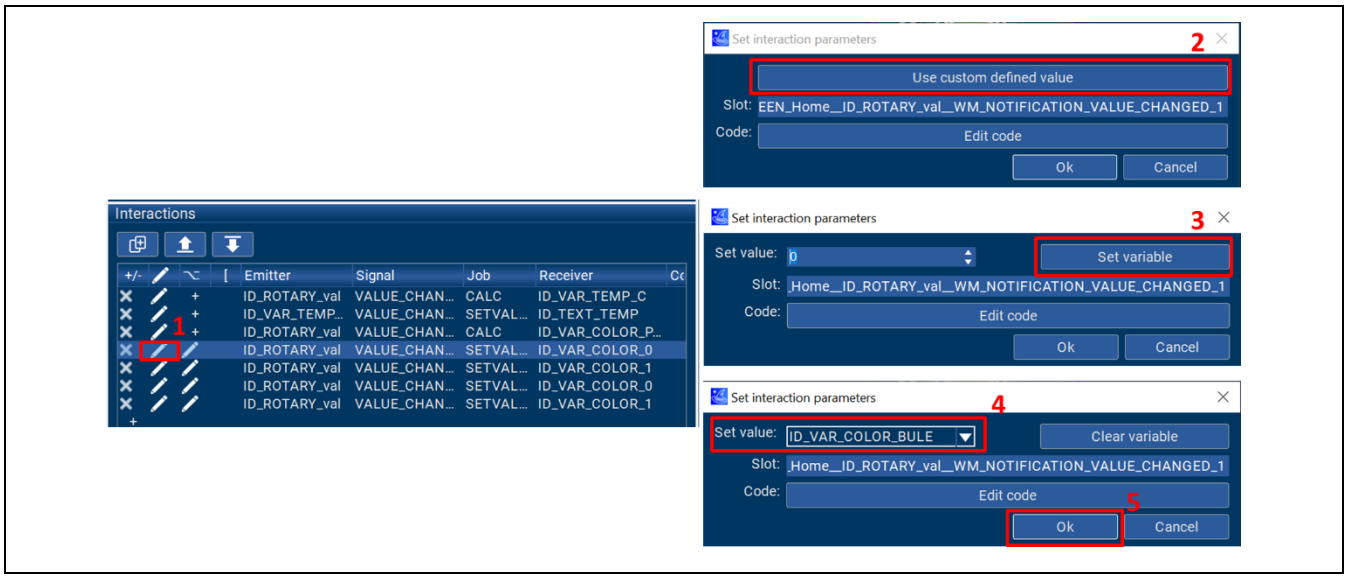


圖 2-49 色彩變換初始值 1

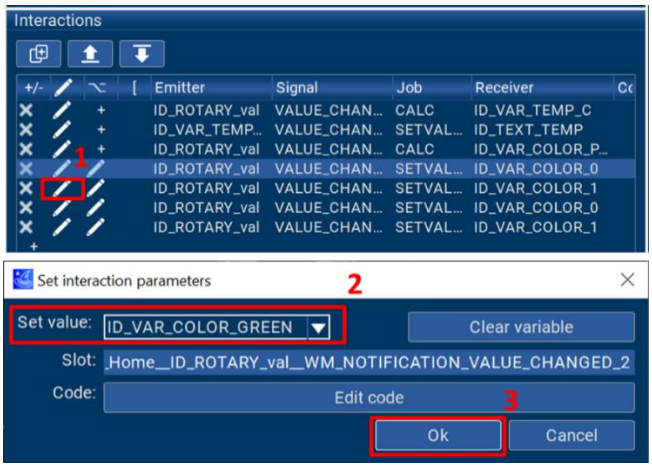


圖 2-50 色彩變換初始值 2

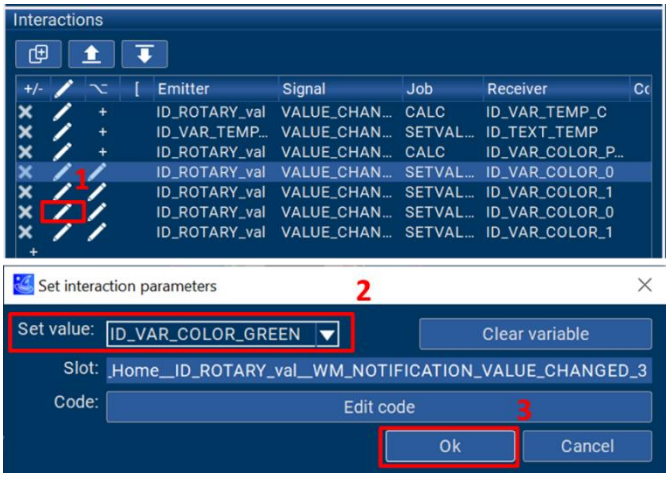


圖 2-51 色彩變換初始值 3

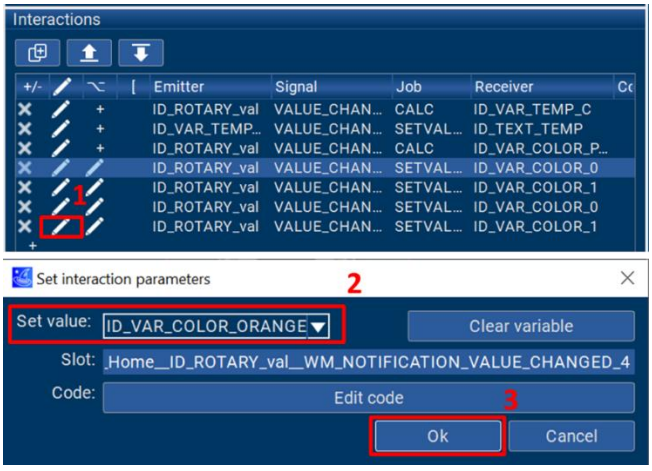


圖 2-52 色彩變換初始值 4

4) 依圖 2-53~圖 2-58 的步驟新增 Drawings 物件，設置將色彩隨數值切換顯示的功能。

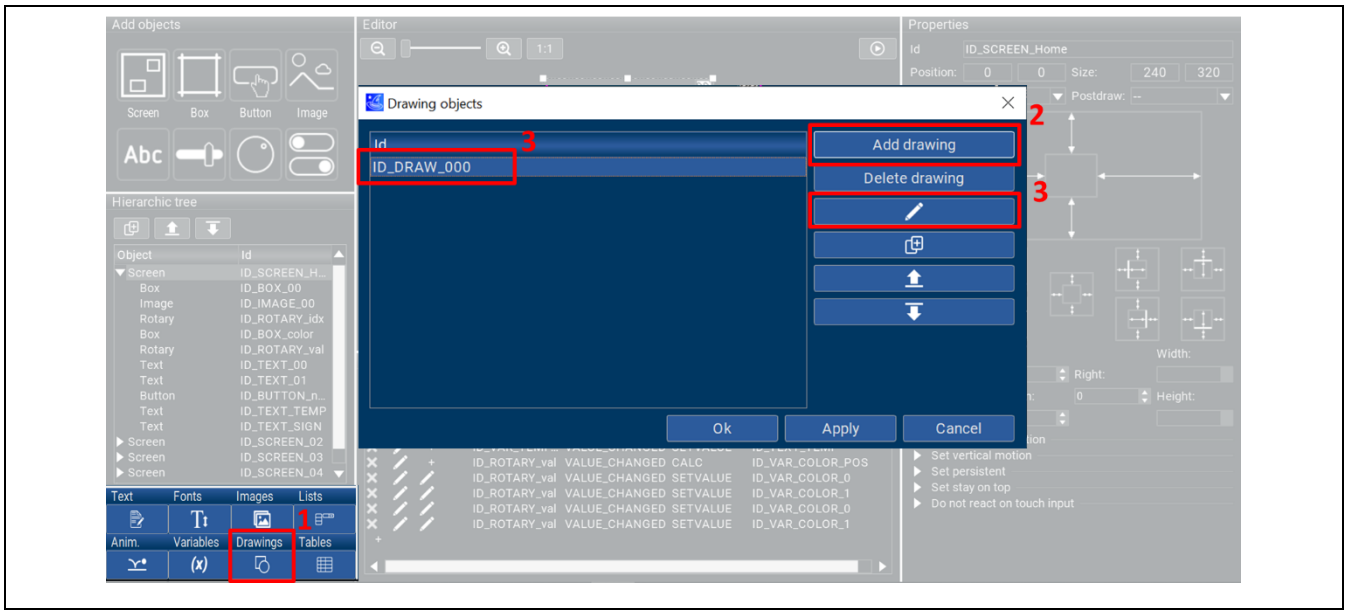


圖 2-53 新增 Drawings 物件(ID_DRAW_000)

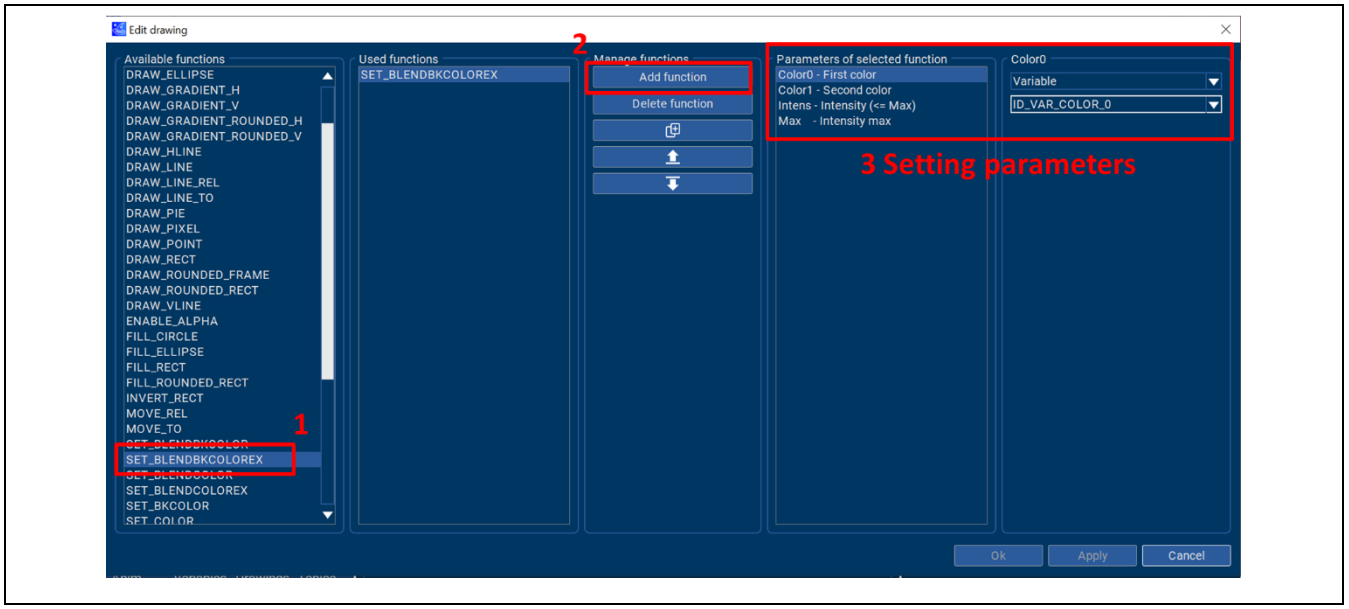


圖 2-54 在 ID_DRAW_000 內新增著色 function

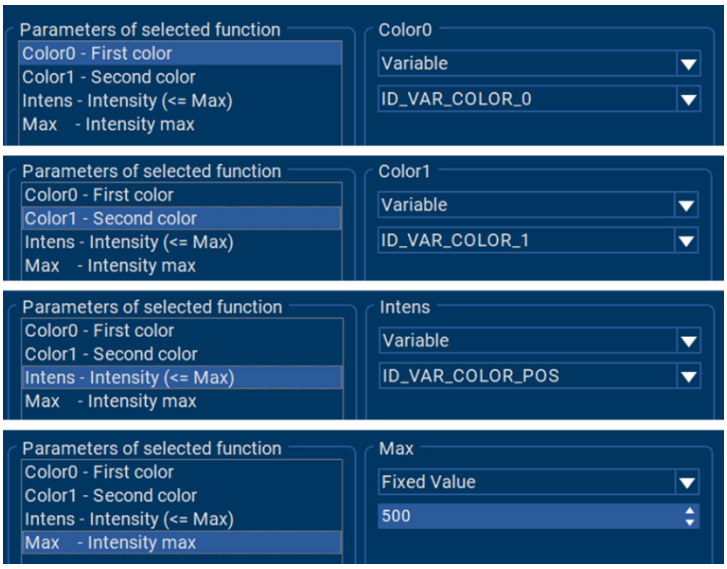


圖 2-55 著色 function 各項參數設定

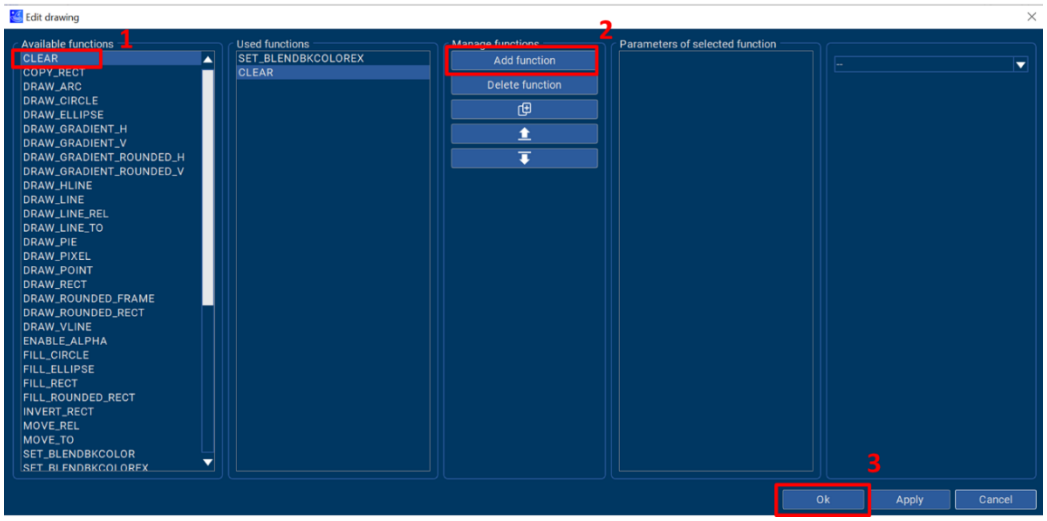


圖 2-56 在 ID_DRAW_000 內新增 clear function

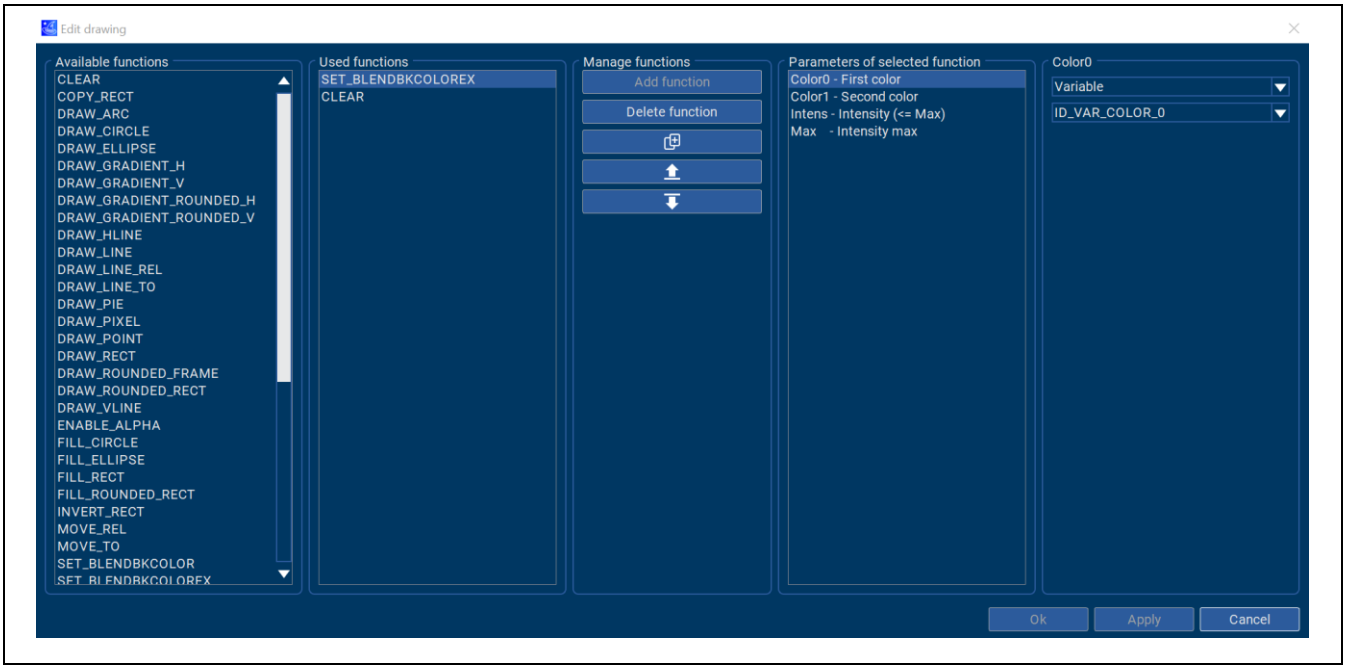


圖 2-57 ID_DRAW_000 設置總覽

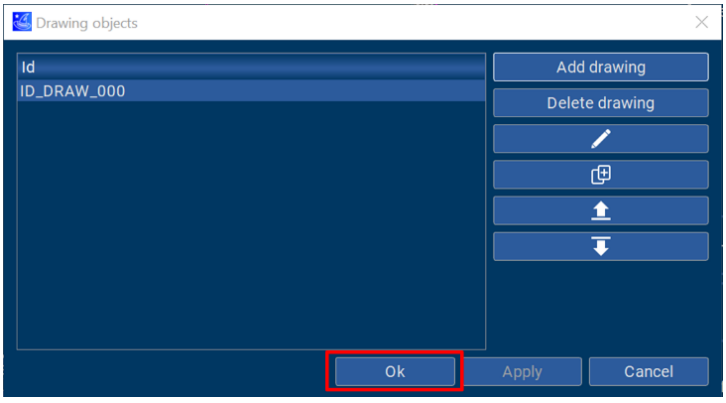


圖 2-58 儲存 ID_DRAW_000 設置

5) 點選物件 ID_BOX_color，在屬性欄位將 Postdraw 設置為 ID_DRAW_000。

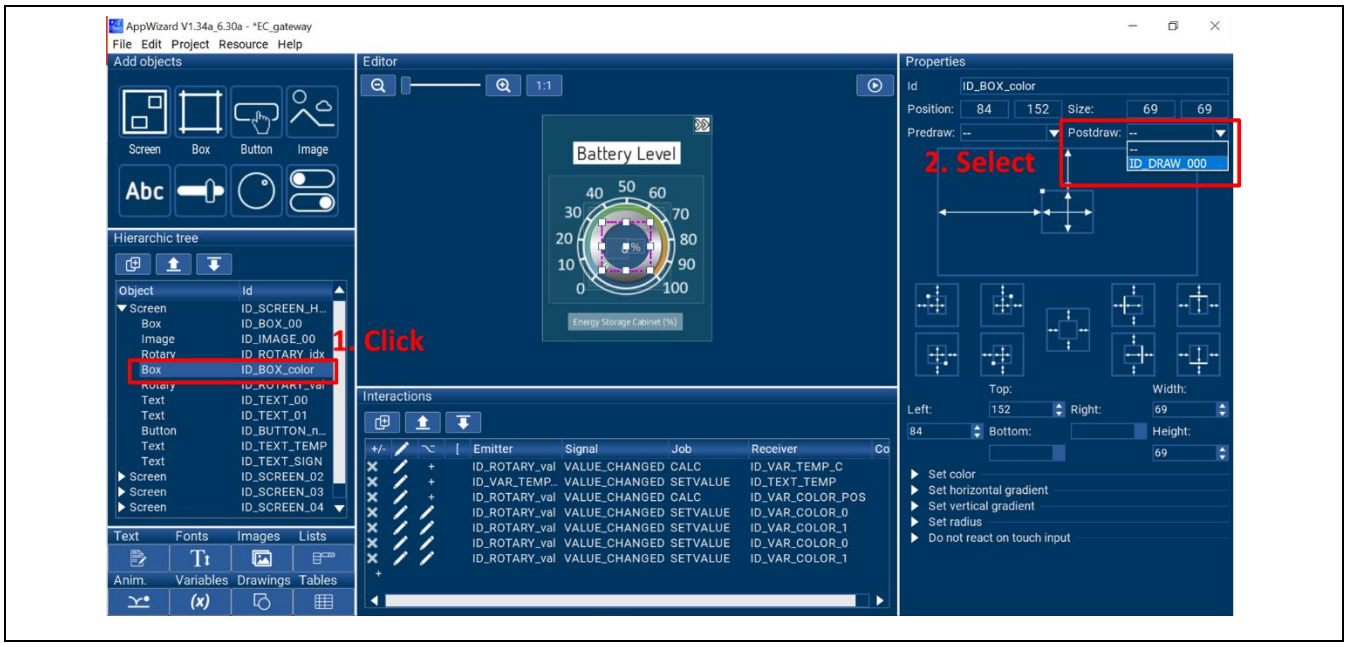


圖 2-59 設置 ID_BOX_color 的 Postdraw 屬性

6) 完成上述動作後， ID_BOX_color 色彩便會隨 ID_Rotary_val 數值變化。

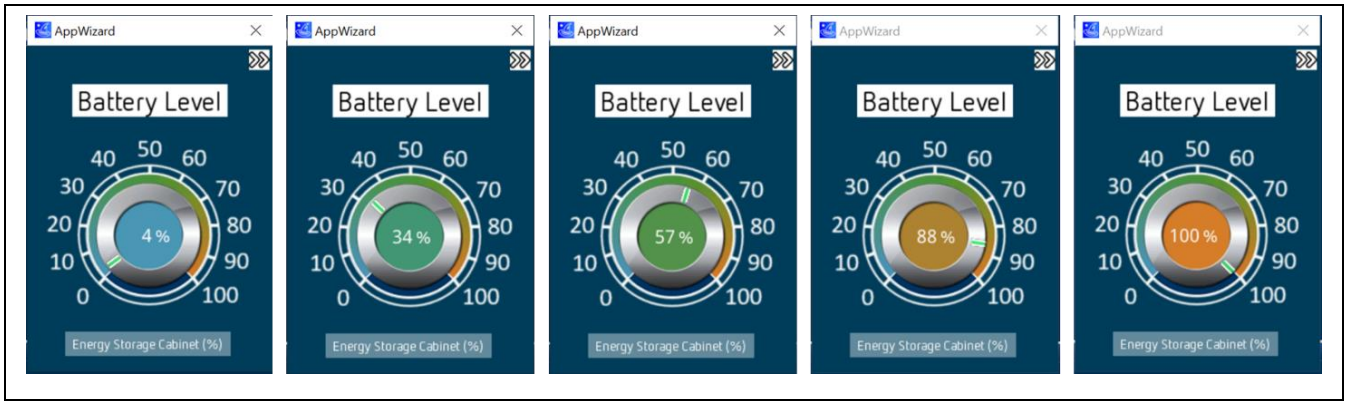


圖 2-60 透過 Play Mode 預覽 ID_BOX_color 變化

3. 在 interactions 欄位設置按鈕的交互作用，使點選按鈕可以切換到下一頁。

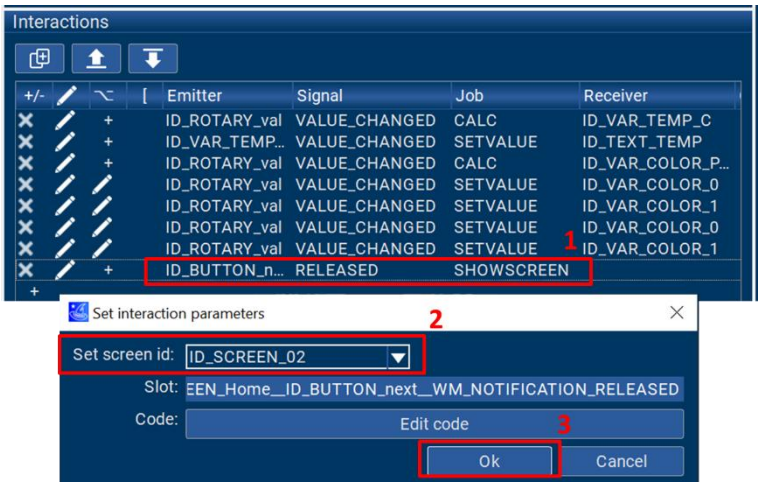
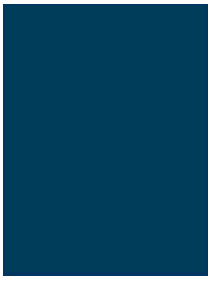


圖 2-61 換頁按鈕(ID_Button_next)設置

2.3.2 光伏逆變器設計(頁面二)

光伏逆變器是種可以將太陽能光電(PV)太陽能板產生的可變直流電壓轉換為市電頻率交流電(AC)的變頻器，可以回饋回商用輸電系統或供離網的電網使用等，會有轉換率控制及顯示的需求，也可就此設計應用為其他類型逆變器的使用者介面，此設計所用物件如表 2-3 所示。

New Energy Gateway Description	Application GUI	Widget Type
背景圖 240x320		Box
數值轉動旋鈕與 Maker Rotary		Rotary
旋鈕數值條背景 Gauge_bar		Gauge

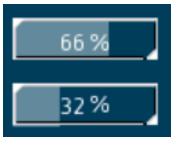
旋鈕數值條		Gauge
軸承切換開關		Switch
軸承轉換率顯示		Progbar
軸承數值條		Slider
標題文字		Text
旋鈕數值顯示		
上一頁按鈕		Button
下一頁按鈕		

表 2-3 太陽能變頻器設計物件

2.3.2.1 GUI 物件配置

1. 點選 ID_SCREEN_02 開始編輯光伏逆變器畫面，透過 Add objects 新增 TEXT 物件 (ID_TEXT_title)設置頁面標題。

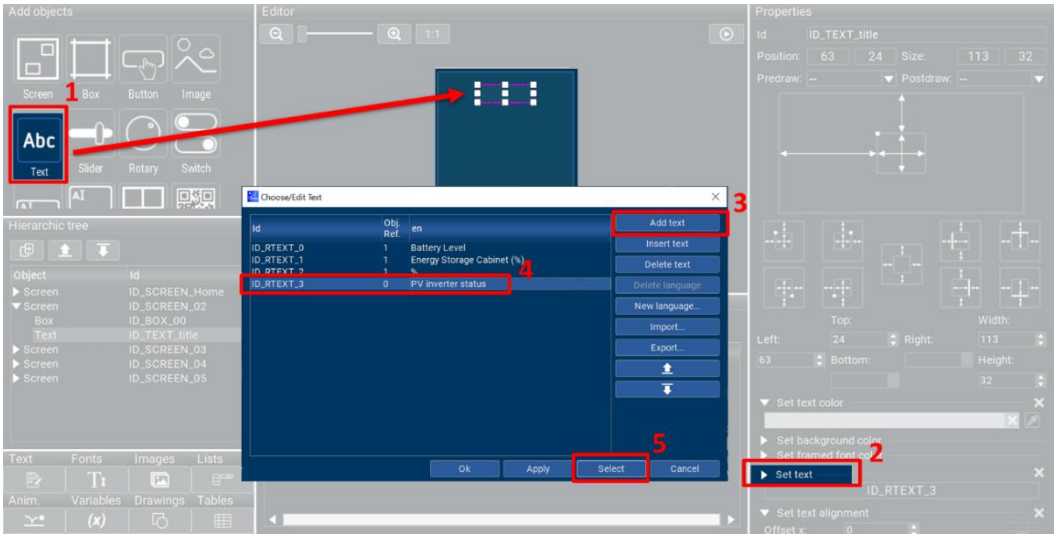


圖 2-62 設置標題文字

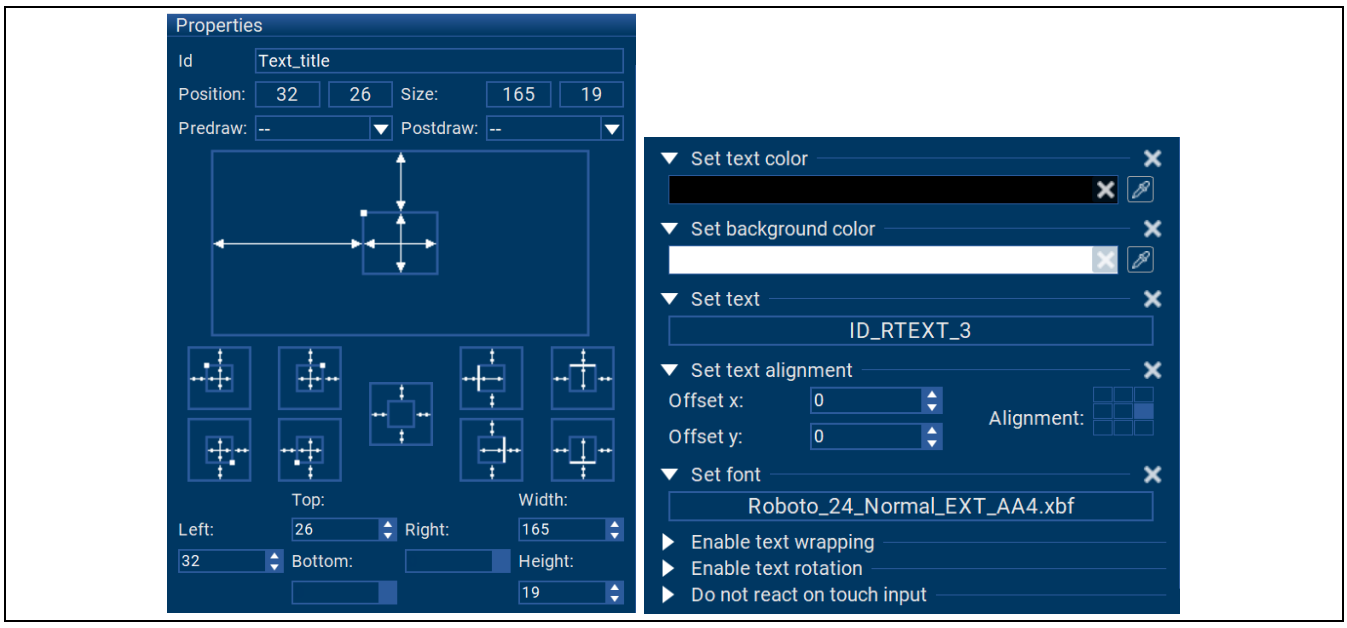


圖 2-63 ID_TEXT_title 屬性設置

2. 透過 Add objects 新增 Gauge 物件(ID_Gauge_BG)作為變化數值的背景。

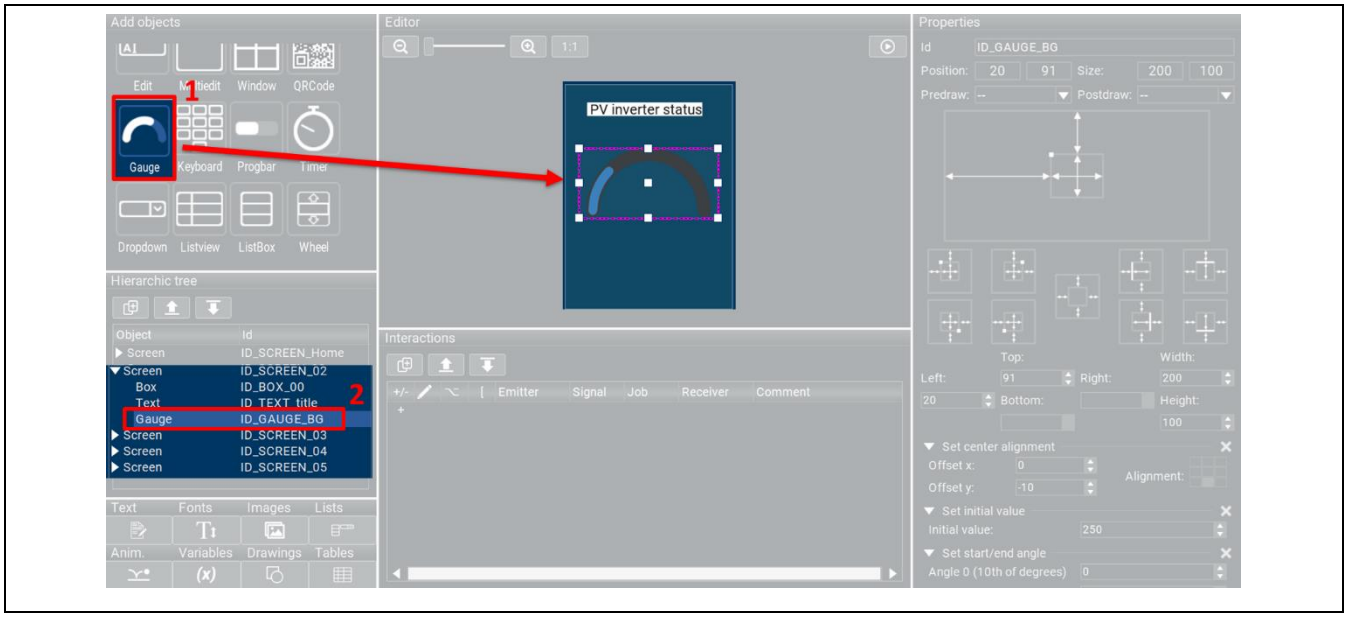


圖 2-64 新增物件 ID_Gauge_BG

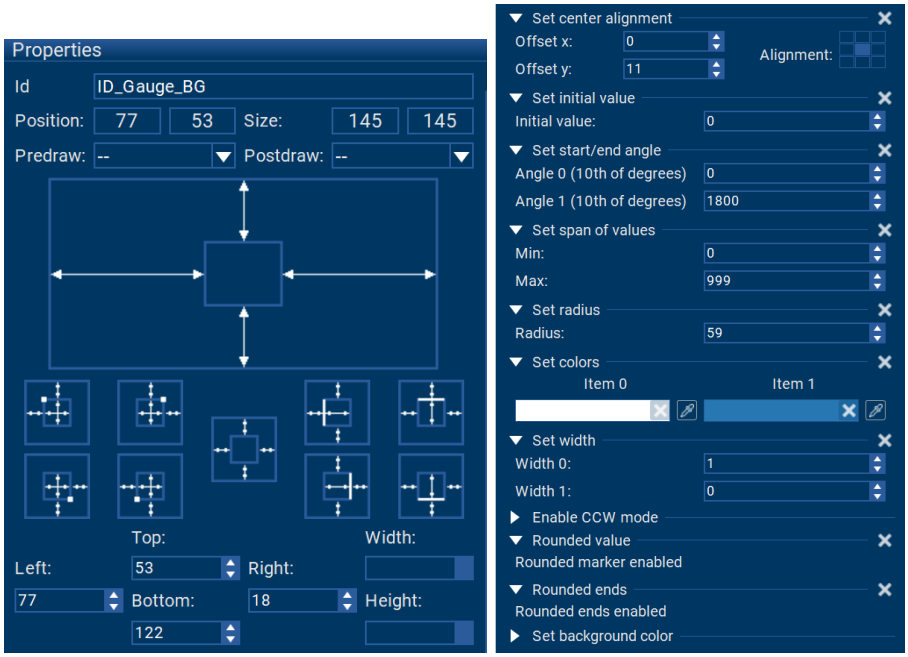


圖 2-65 ID_Gauge_BG 屬性設置

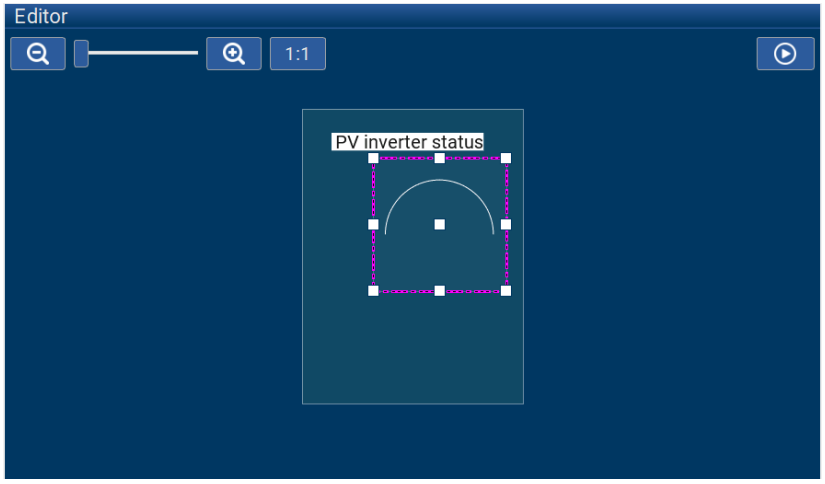


圖 2-66 ID_Gauge_BG 設置畫面

3. 新增第二個 Gauge(ID_Gauge_BAR)物件作為數值顯示。

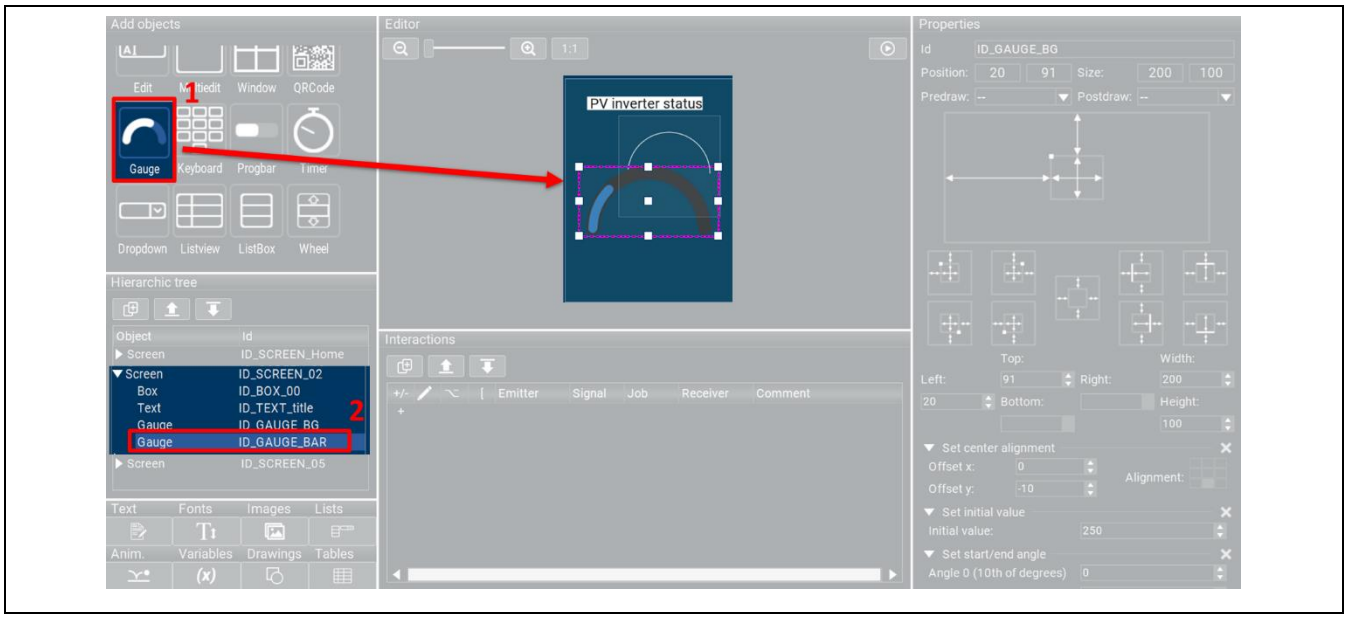


圖 2-67 新增物件 ID_Gauge_BAR

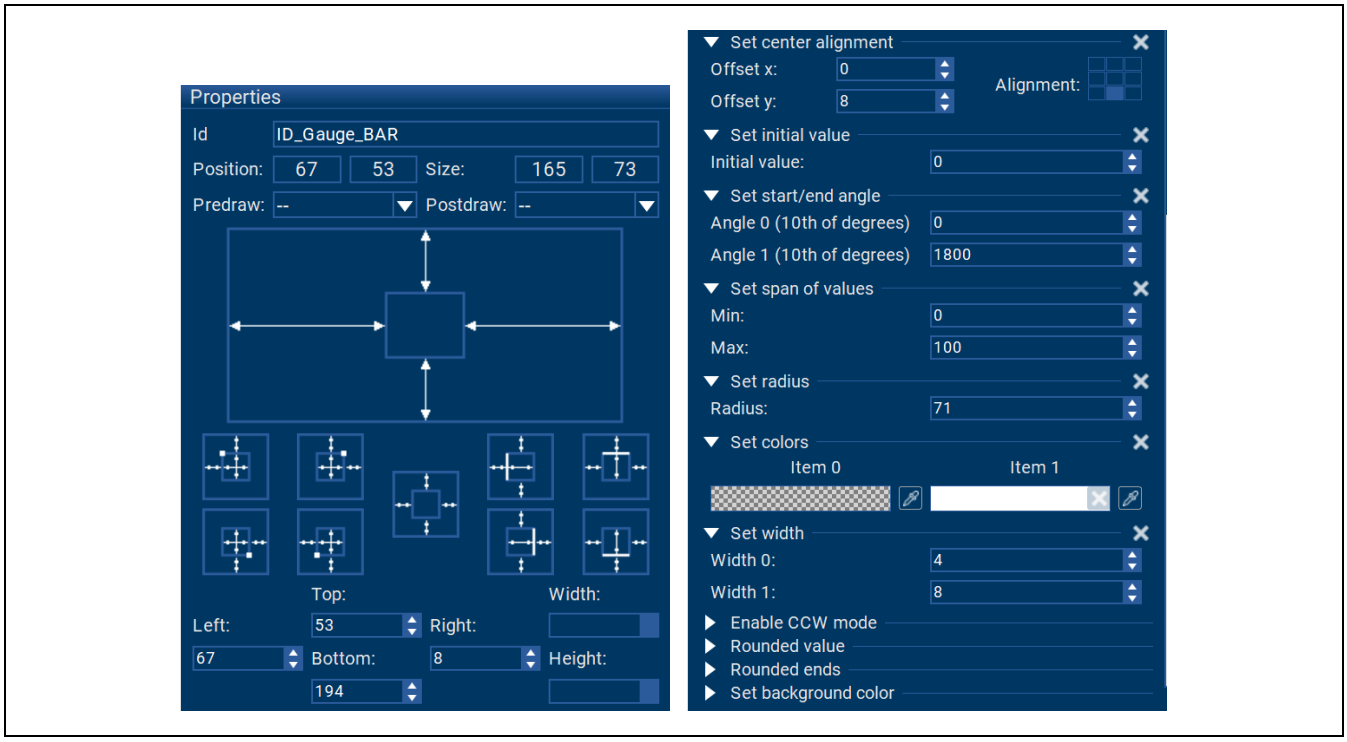


圖 2-68 ID_Gauge_BAR 屬性設置

4. 新增 Rotary 物件(ID_Rotary)做數值轉動，並選擇圖片 Rotary_Background.png、Rotary_Marker.png 設置為背景及指標。

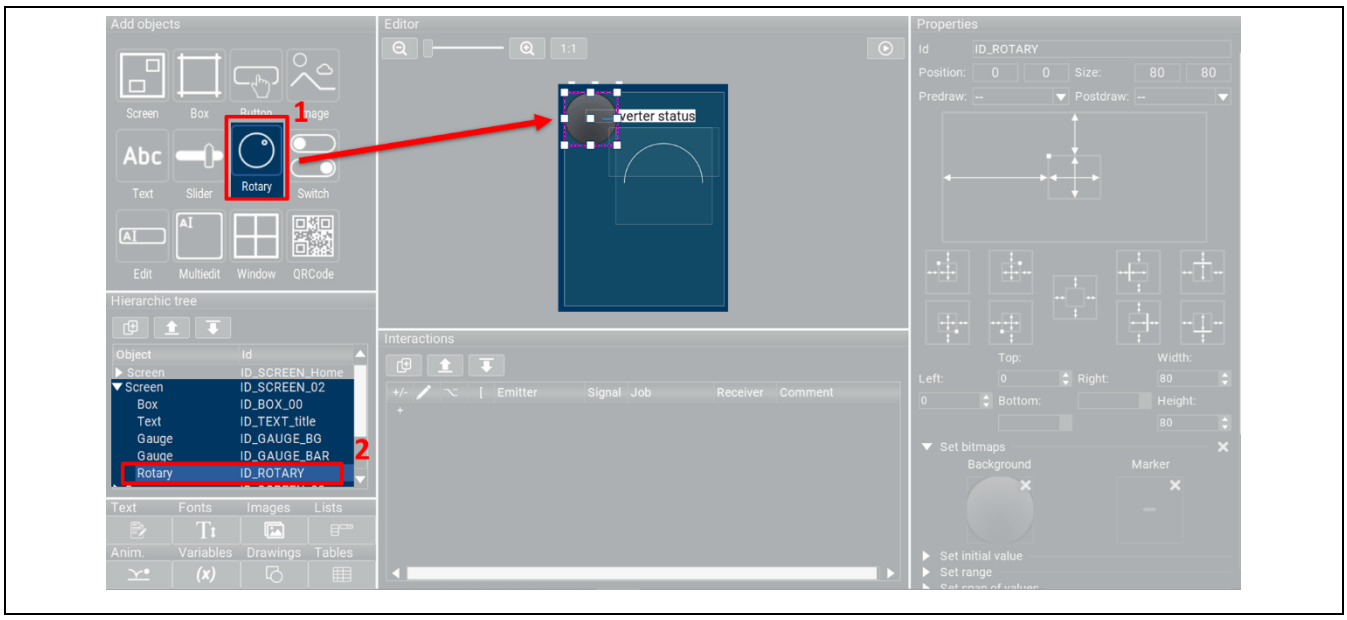


圖 2-69 新增 Rotary 物件

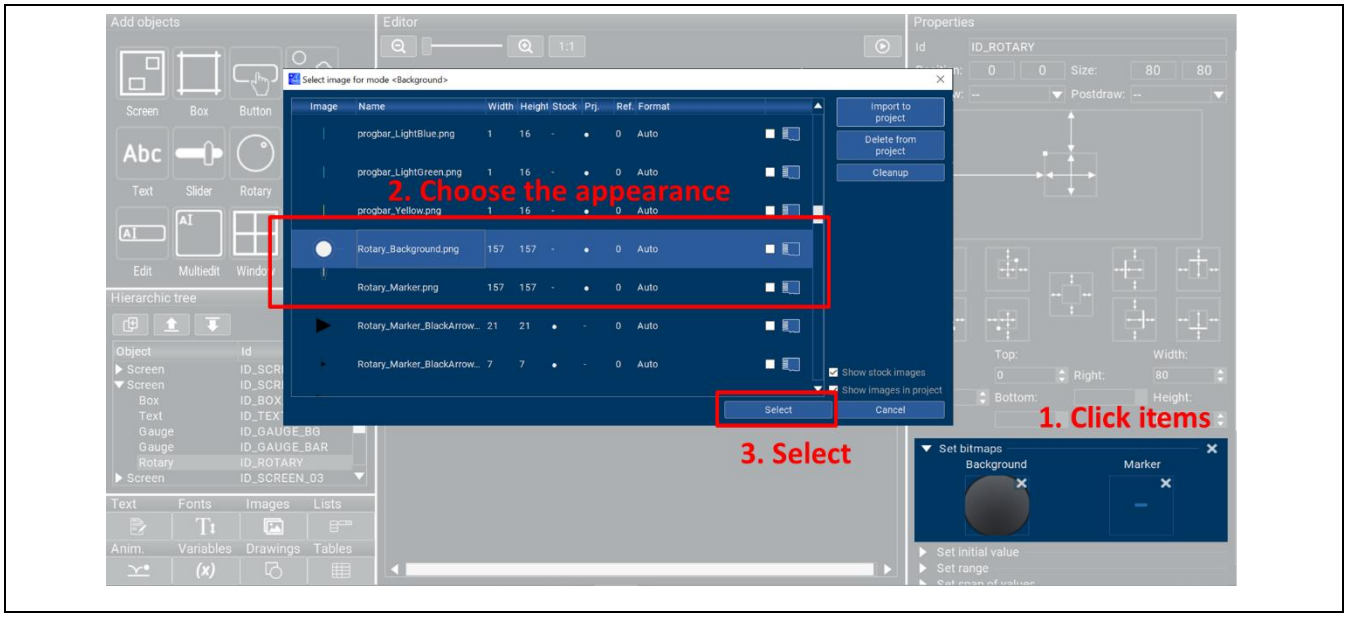


圖 2-70 ID_Rotary 外觀設置

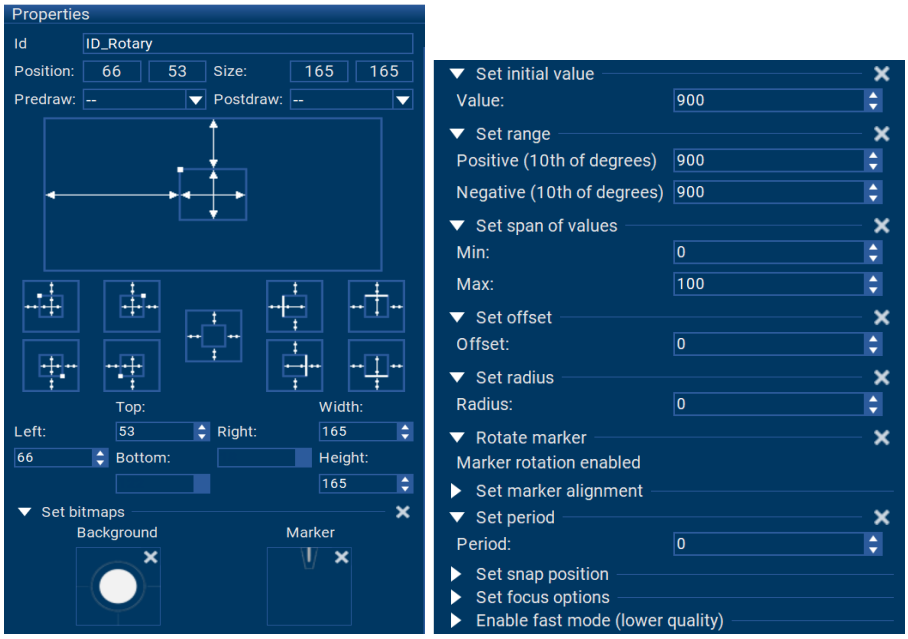


圖 2-71 ID_Rotary 屬性設置與外觀

5. 新增一個 TEXT 物件(TEXT_value)顯示 Rotary(ID_Rotary)數值，並依圖 2-72 設置屬性。

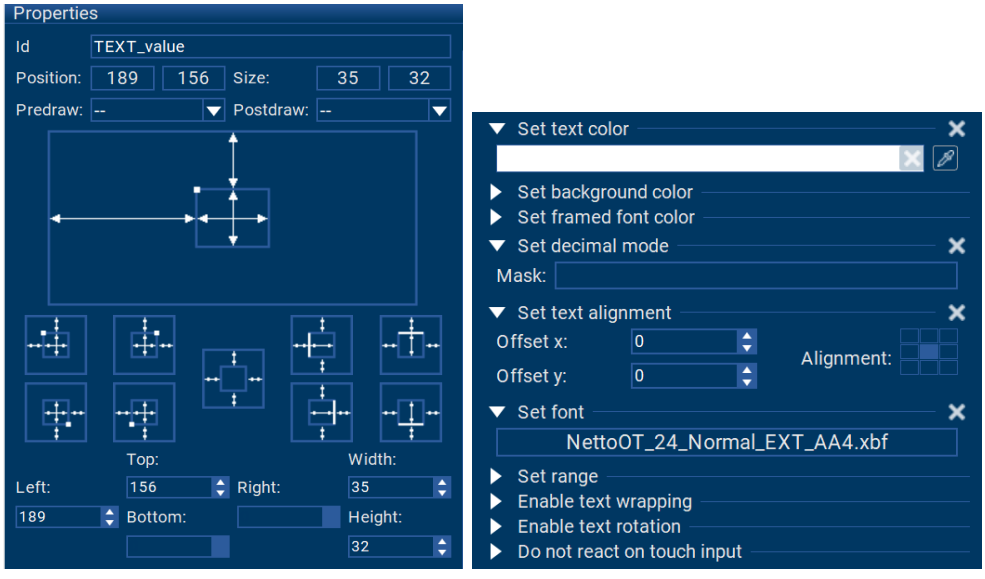


圖 2-72 TEXT_value 屬性設置

6. 新增兩個 Switch 物件(SWITCH_01、SWITCH_02)作為功能開關，並將圖片 Swtich_Pressed_83x22.png、Swtich_Unpressed_83x22.png 設置為 BG-Left 與 BG-Right 顯示畫面，並依圖 2-74~圖 2-77 設置屬性。

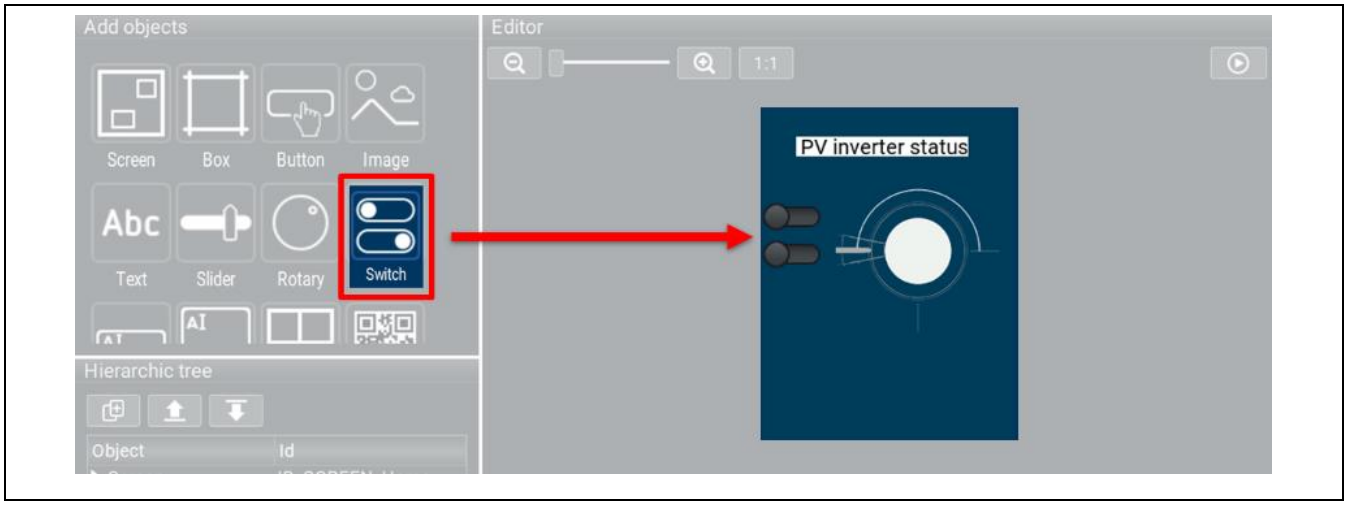


圖 2-73 新增 Switch 物件

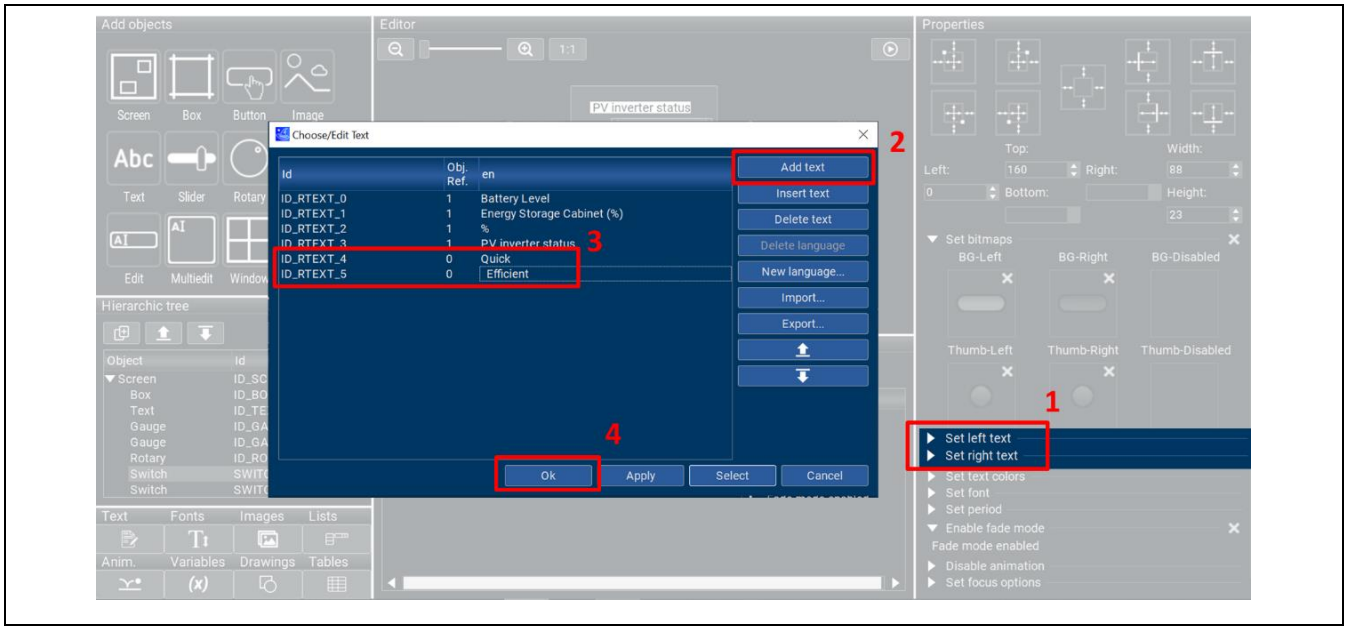


圖 2-74 新增 Switch 顯示文字

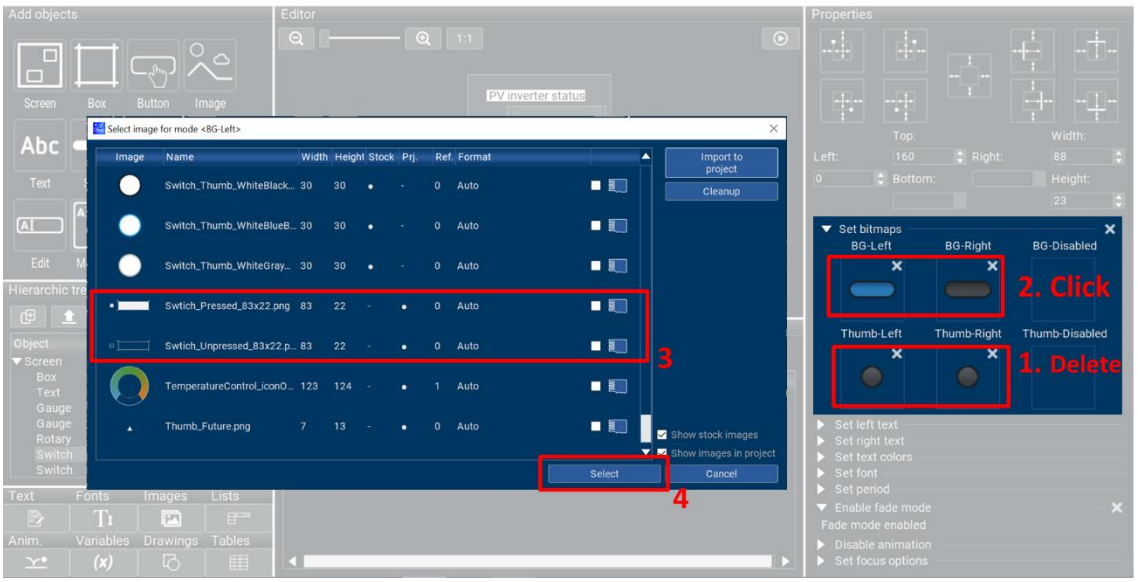


圖 2-75 設定 Switch 外觀

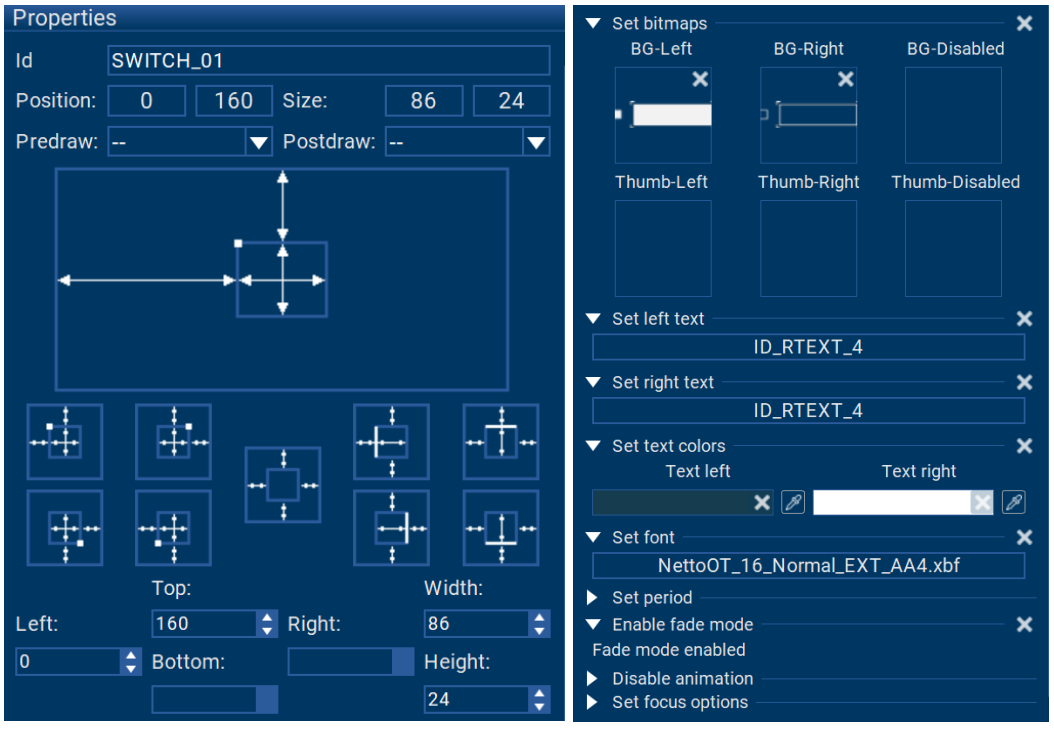


圖 2-76 SWITCH_01 屬性設置

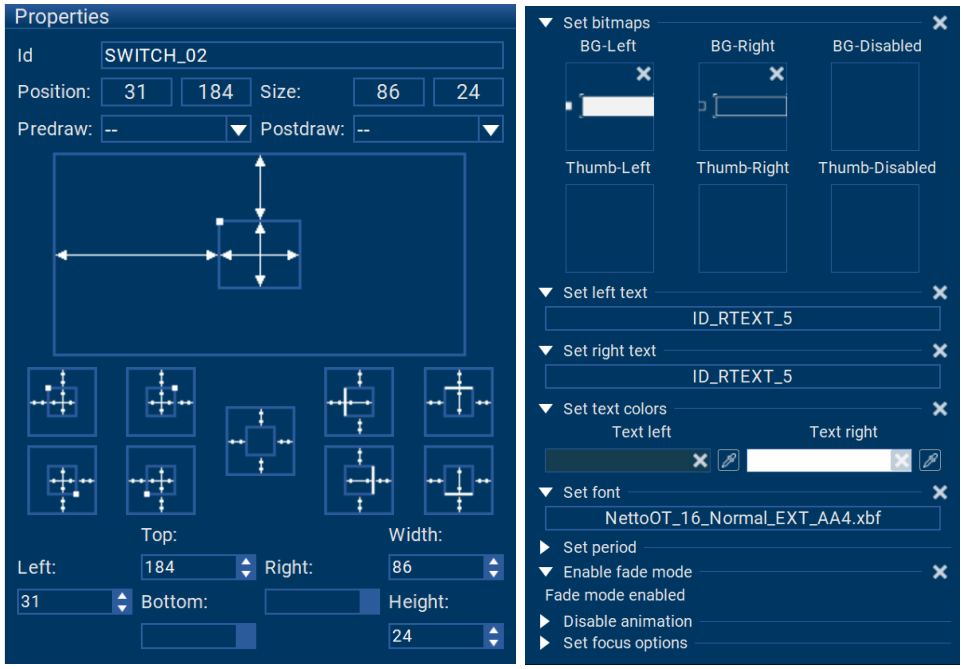


圖 2-77 SWITCH_02 屬性設置

7. 依圖 2-78~圖 2-89 步驟新增 Thrust bearing 與 Guide bearing 的控制開關。

1) 新增兩組 Switch 及 Progbar 物件(SWITCH_TB、PROGBAR_TB、SWITCH_GB、PROGBAR_GB)，選擇圖片 Future_Switch_Button_Pressed_20x84.png、Future_Switch_Button_Unpressed_84x23.png 作為 Switch 圖示。

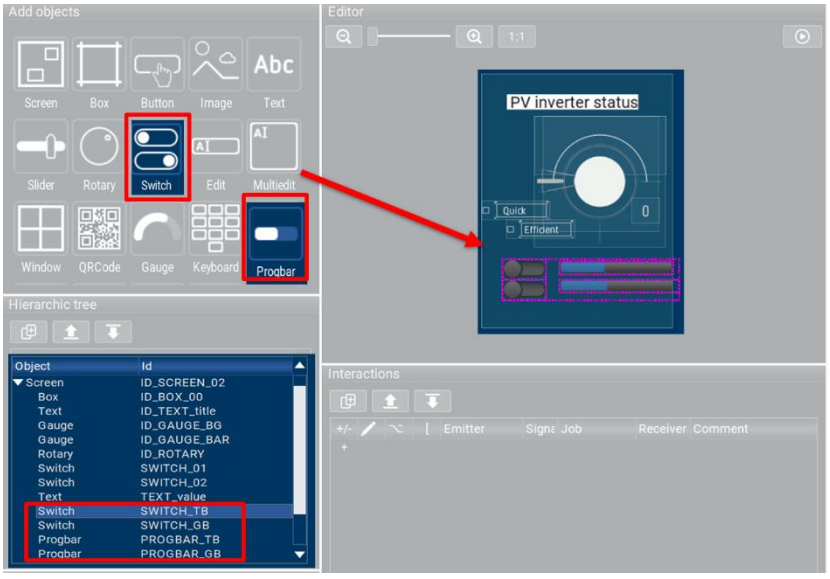


圖 2-78 新增兩組 Switch 與 Progbar 物件

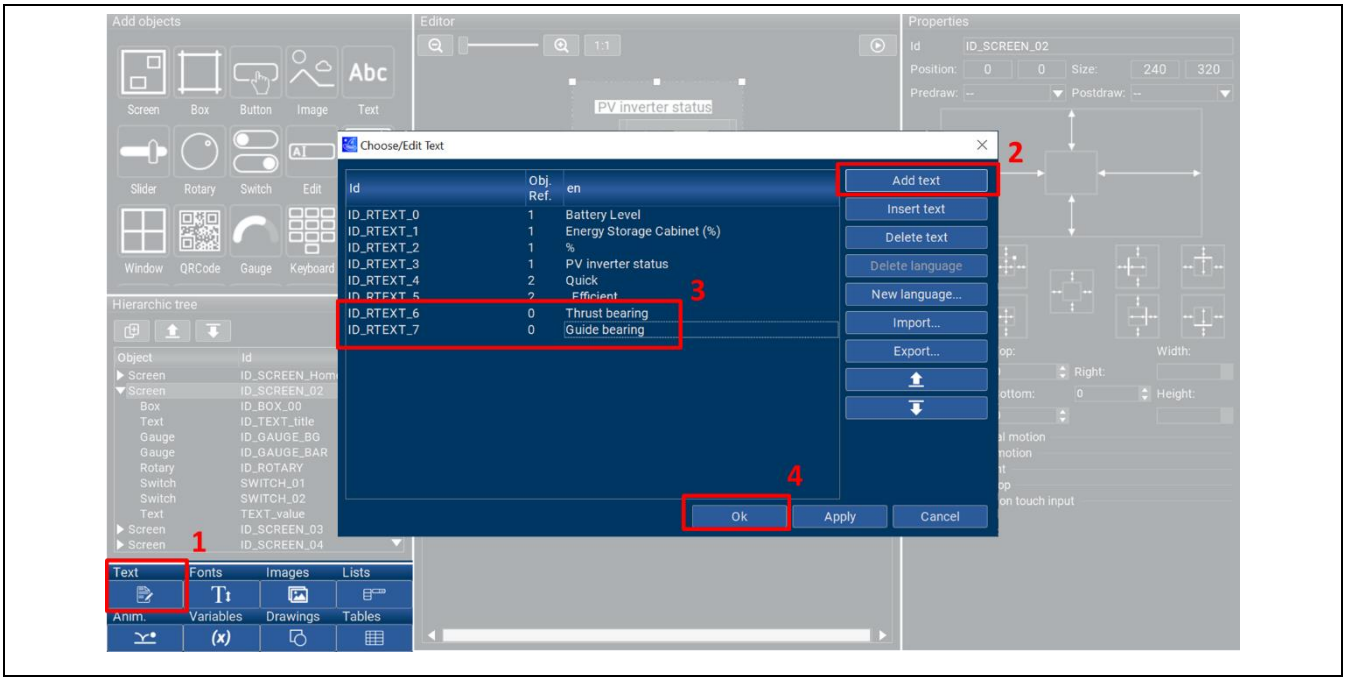


圖 2-79 新增 switch 文字內容

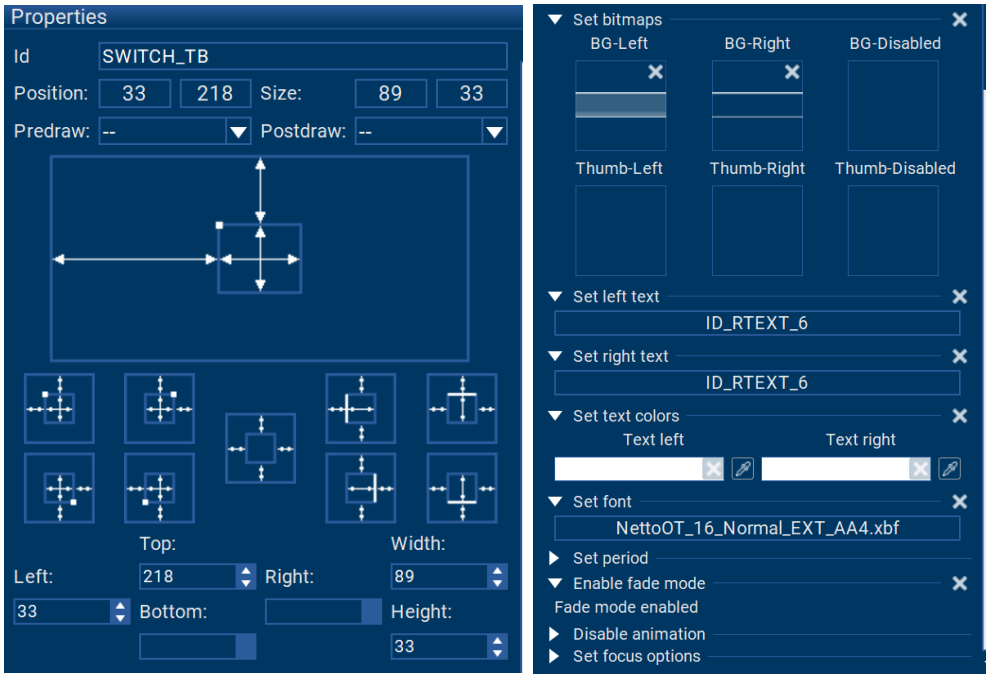


圖 2-80 SWITCH_TB 屬性設置

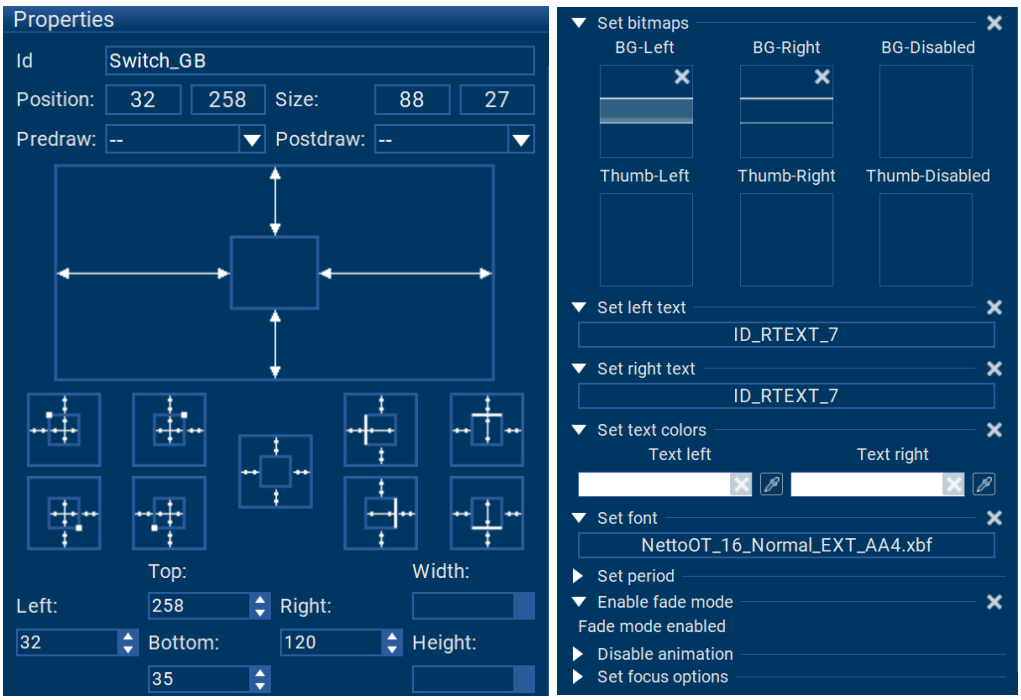


圖 2-81 SWITCH_GB 屬性設置

2) 選擇圖片 Progbar_Future_START_80x23.png 作為 Progbar(PROGBAR_TB、PROGBAR_GB)的 bitmap。

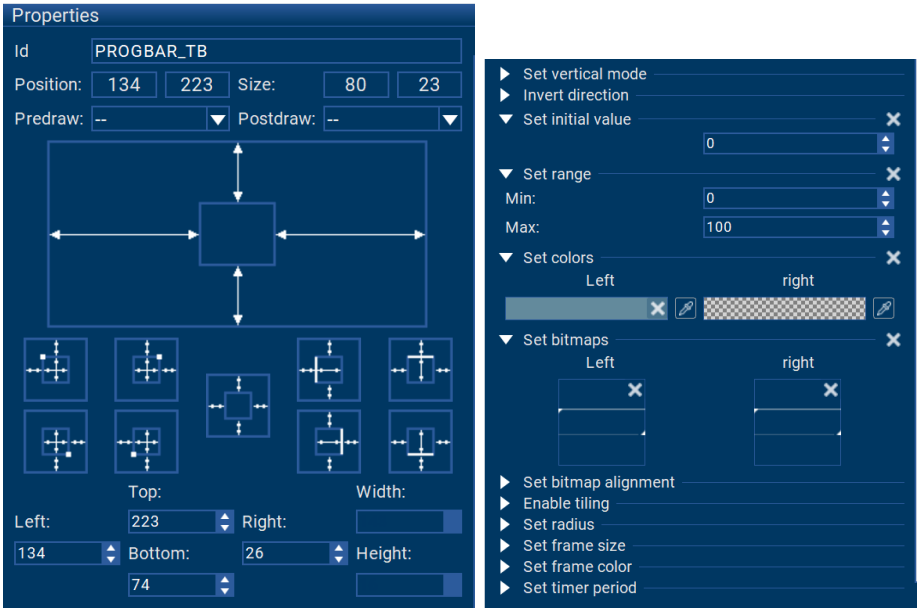


圖 2-82 PROGBAR_TB 屬性設置

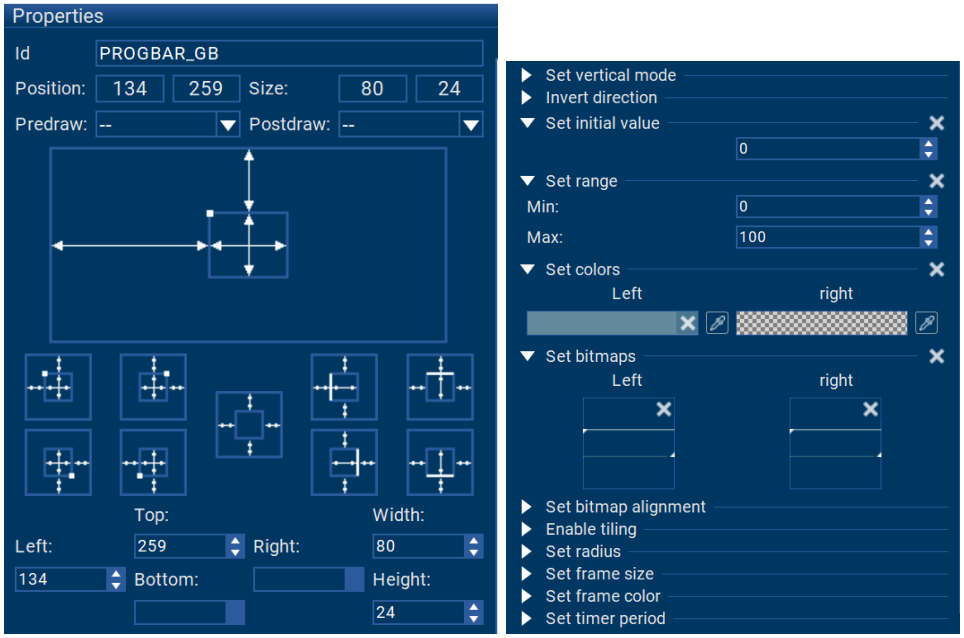


圖 2-83 PROGBAR_GB 屬性設置

3) 新增四個 TEXT 物件來顯示數值與單位。

4) 設置第一組 Progbar(PROGBAR_TB)的 TEXT 物件，分別以 TEXT_num1、TEXT_unit1 表示數值與單位，並疊放在 Progbar 物件上。

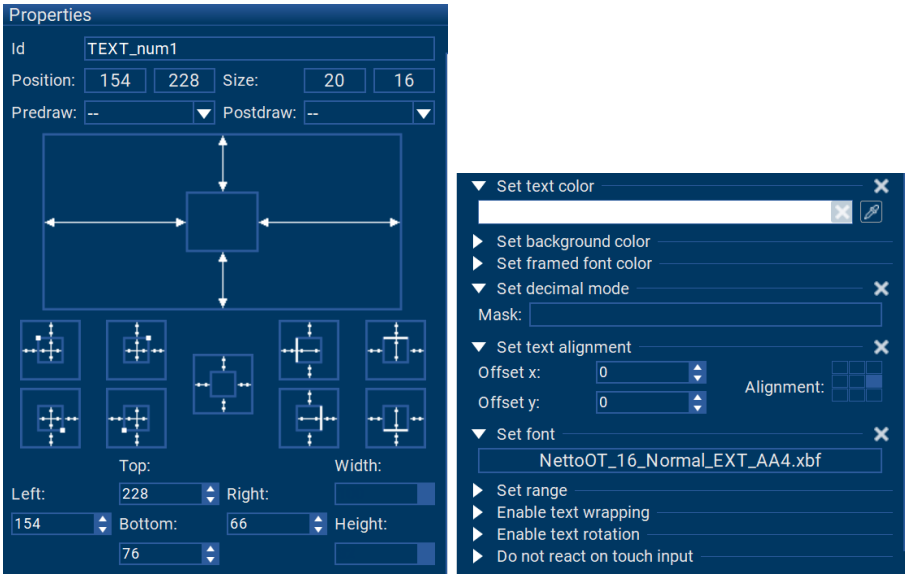


圖 2-84 TEXT_num1 屬性設置

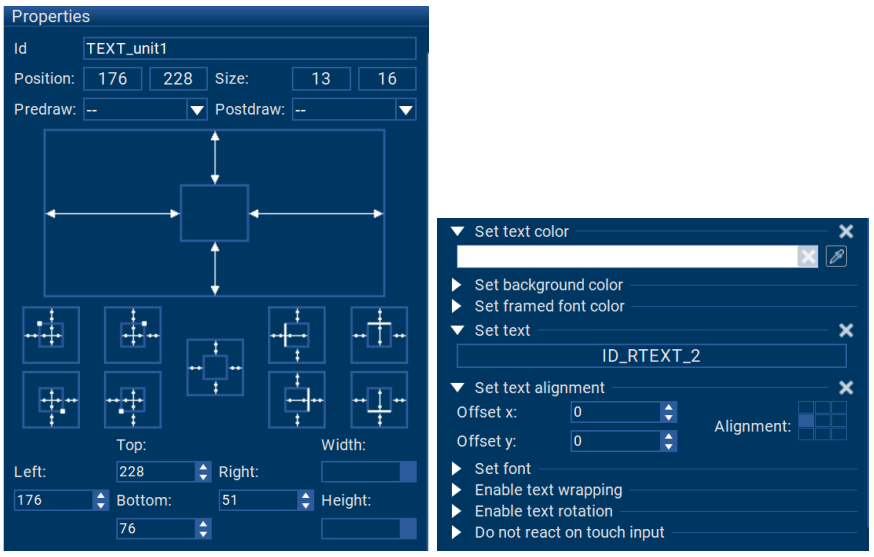


圖 2-85 TEXT_unit1 屬性設置

5) 設置第二組 Progbar(PROGBAR_GB)的 TEXT 物件，分別以 TEXT_num2、TEXT_unit2 表示數值與單位，並疊放在 Progbar 物件上。

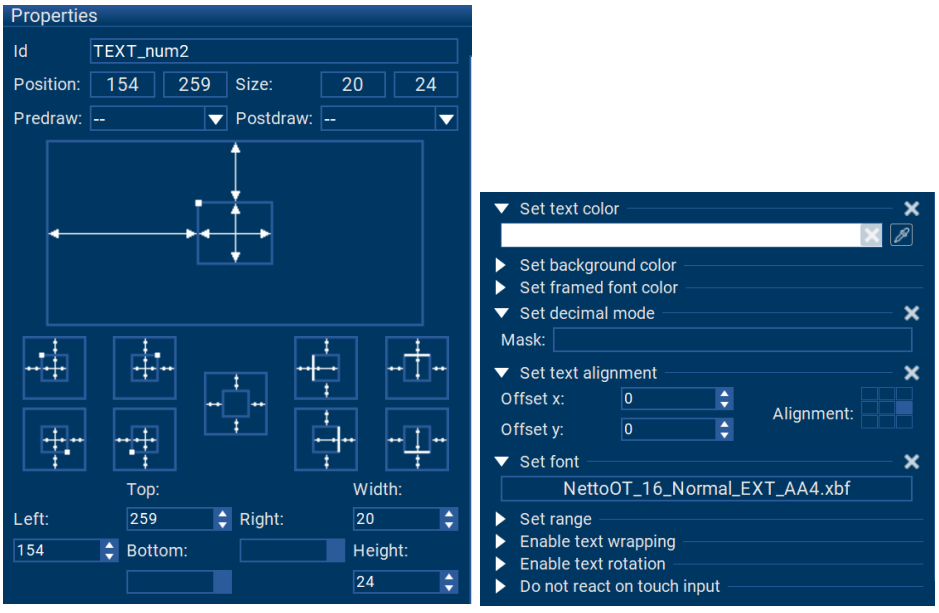


圖 2-86 TEXT_num2 屬性設置

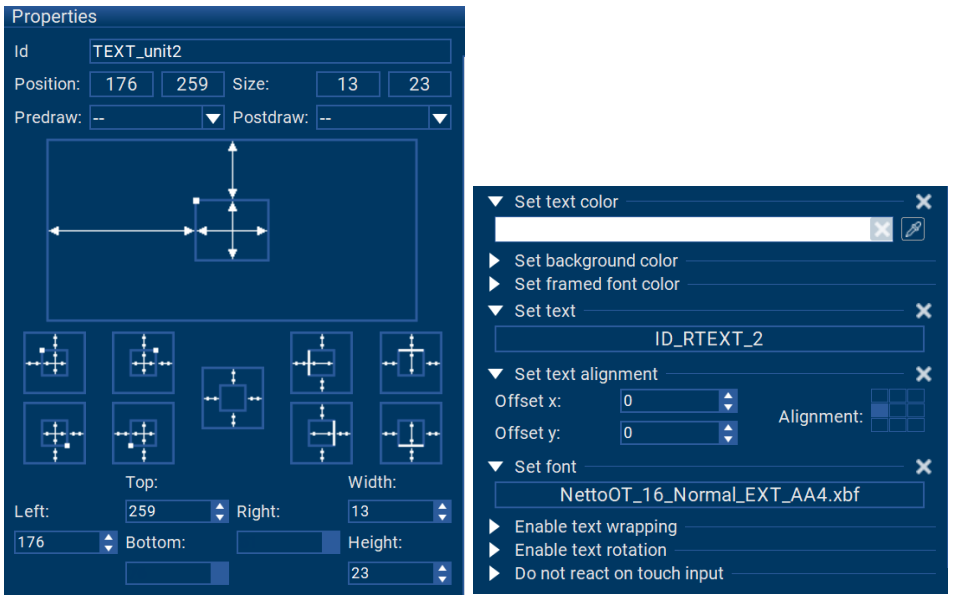


圖 2-87 TEXT_unit2 屬性設置

6) 新增 Slider 物件 (ID_SLIDER) 作為數值調整開關，並將圖片 Slider_Start_Future.png、Slider_End_Future.png、Thumb_Future.png 設置為 Slider 的 Shaft left、Shaft right 與 Thumb up。

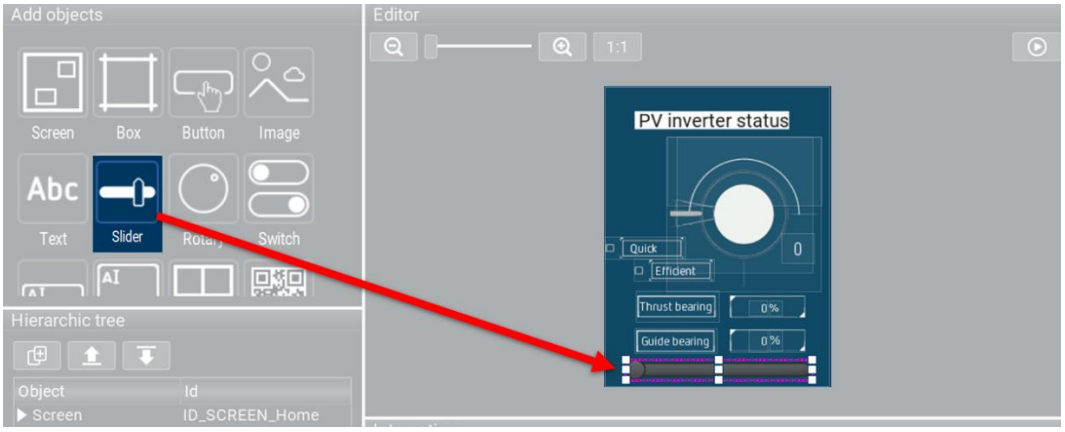


圖 2-88 新增 Slider 物件

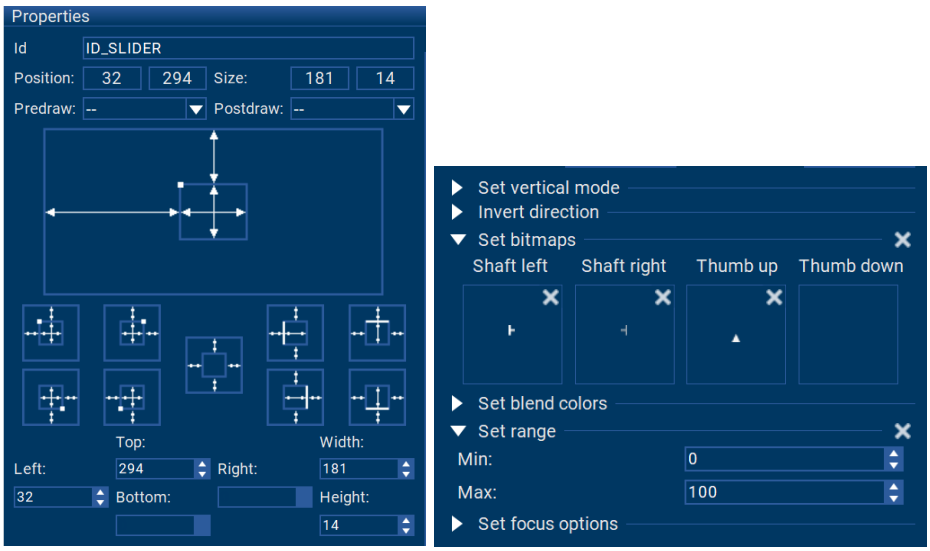


圖 2-89 ID_SLIDER 屬性設置

8. 新增兩個 Button 物件作為前頁(btn_p2_back)與後頁(btn_p2_next)的換頁按鈕，並分別設圖片 arrow2_20x20_new.png、arrow1_20x20_new.png 作為按鈕圖示。

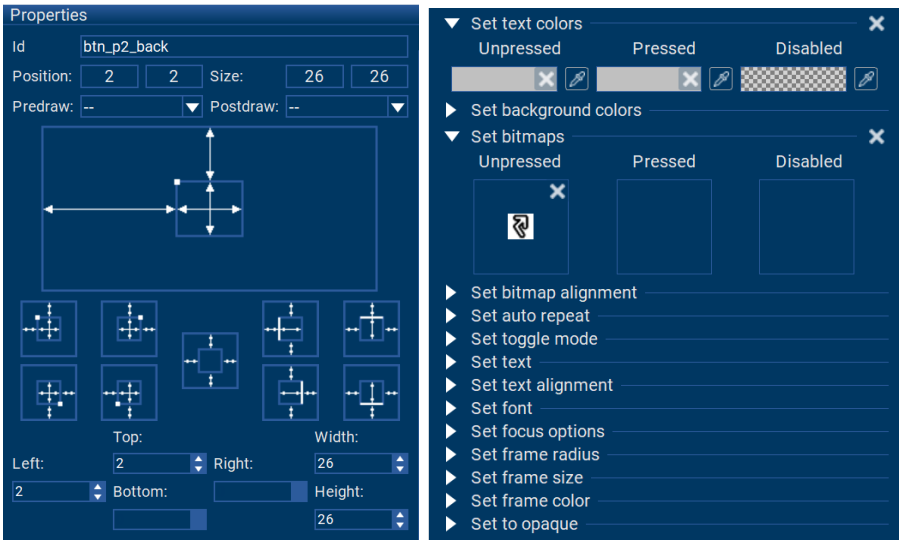


圖 2-90 btn_p2_back 屬性設置

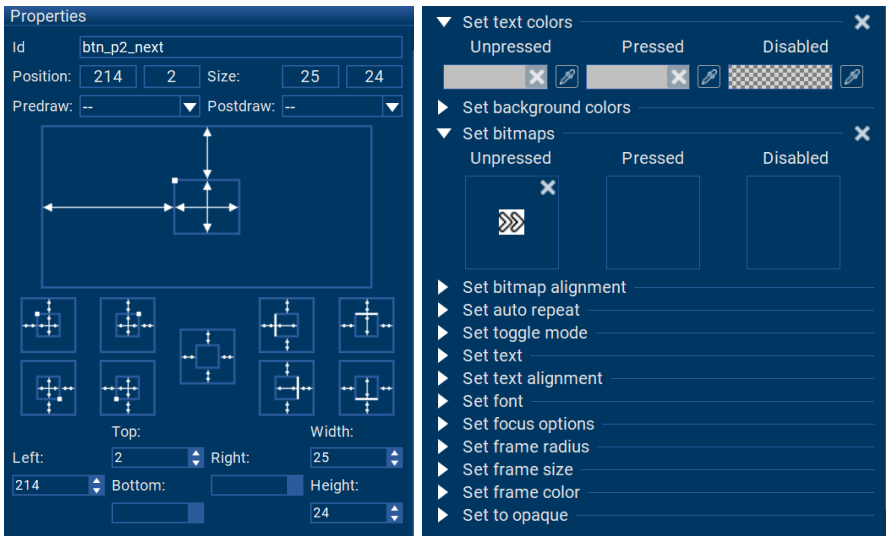


圖 2-91 btn_p2_next 屬性設置

2.3.2.2 物件互動設置

物件設置完成畫面如圖 2-92 所示，接著要建立物件間的交互作用(interactions)，使操作物件時有相應行為顯示。

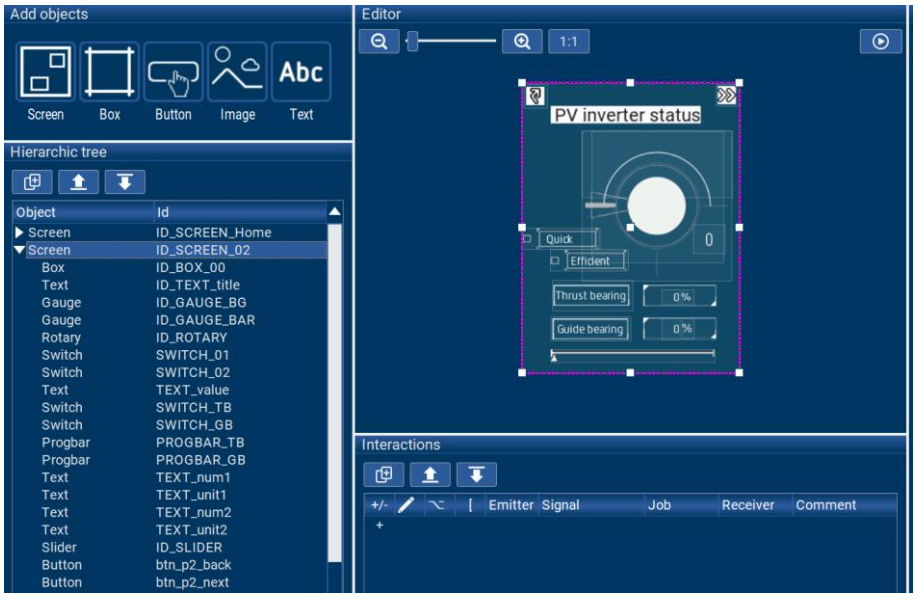


圖 2-92 頁面二設計完成圖

1. 新增 Rotary(ID_ROTARY)、Gauge(ID_GAUGE_BAR)與 TEXT(TEXT_value)物件間的交互關係，使得 Gauge 數值會隨 Rotary 轉動而變化並顯示數值。

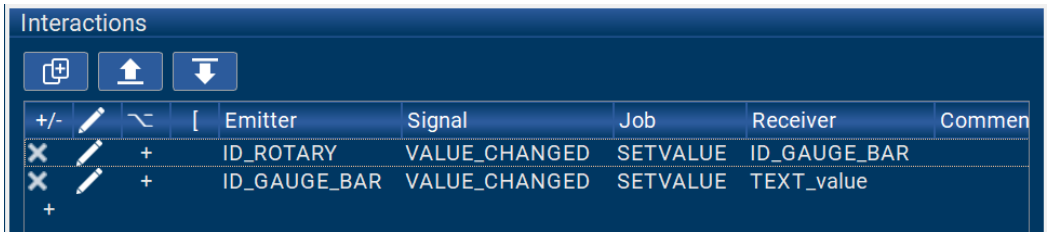


圖 2-93 Rotary、Gauge 與 TEXT 交互作用設置

2. 新增兩組 Progbar(PROGBAR_TB、PROGBAR_GB)與 TEXT(TEXT_num1、TEXT_num2)物件的交互關係，使得 TEXT 顯示數值會隨著 Progbar 變化。

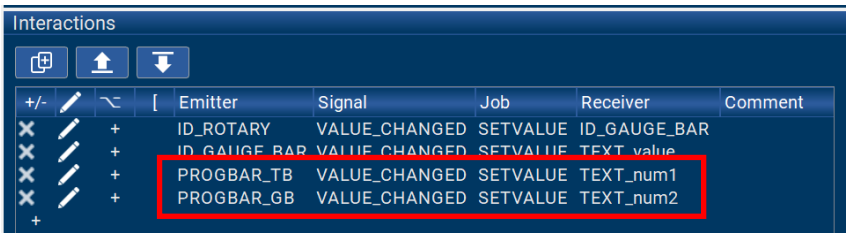


圖 2-94 Progbar 與 TEXT 交互作用設置

3. 設置 Slider(ID_SLIDER)、Progbar(PROGBAR_TB、PROGBAR_GB)與 Switch(SWITCH_TB、SWITCH_GB)物件的交互關係，讓使用者可透過 Slider 調整 Progbar 數值，並以 Switch 切換數值要映射數值的 Progbar。

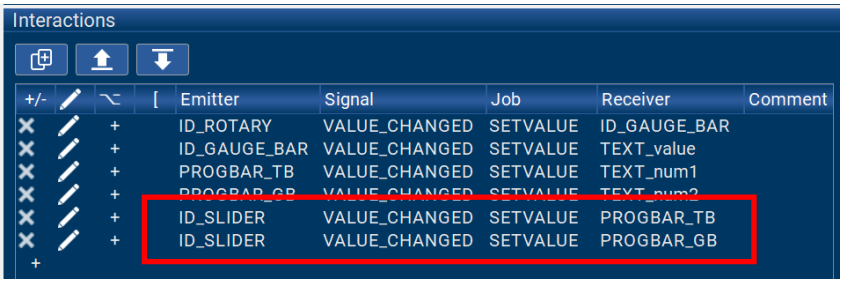


圖 2-95 Slider 與 Progbar 交互作用設置

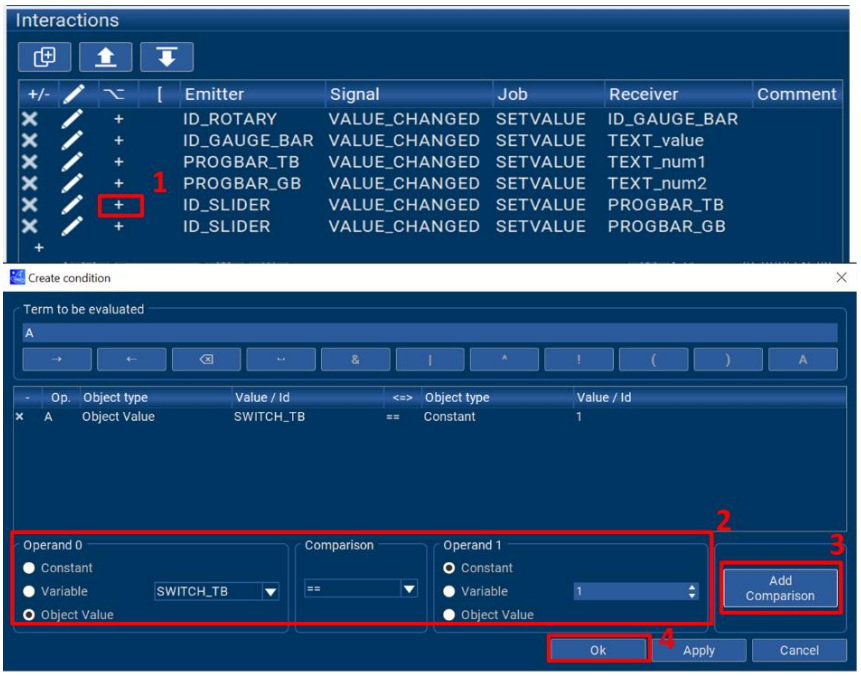


圖 2-96 Switch 與 Slider 開關設置 1

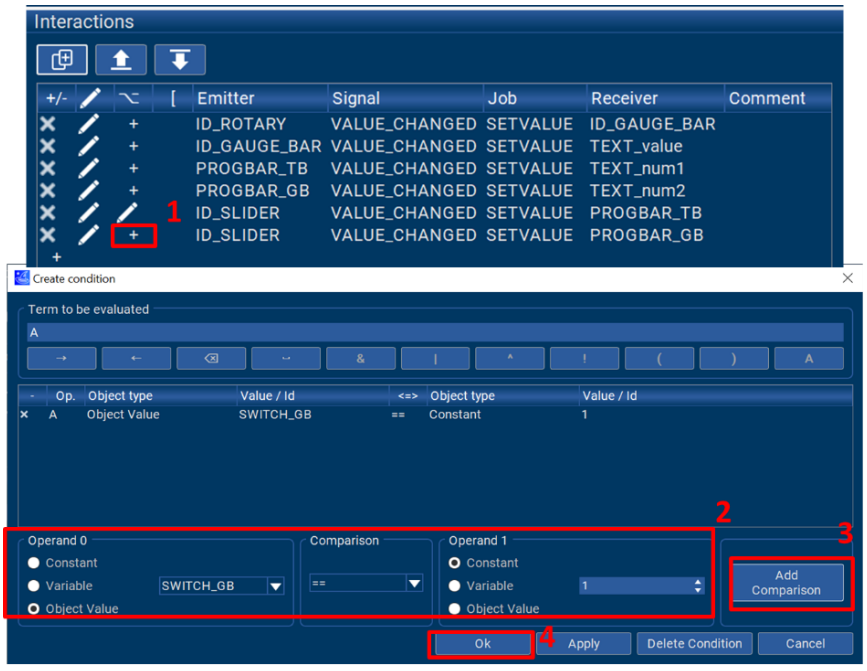


圖 2-97 Switch 與 Slider 開關設置 2

4. 設置 Button(btn_p2_back、btn_p2_next)物件的換頁功能，使點選按鈕可以切回前一頁與下一頁。

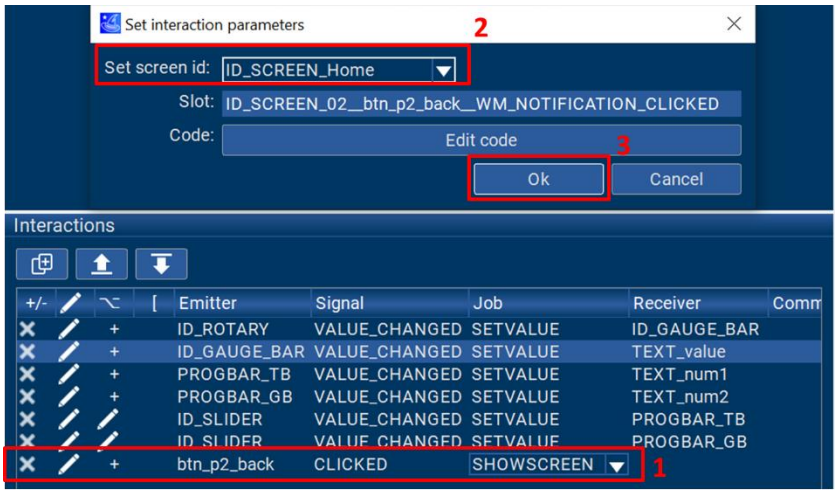


圖 2-98 btn_p2_back 換頁設置

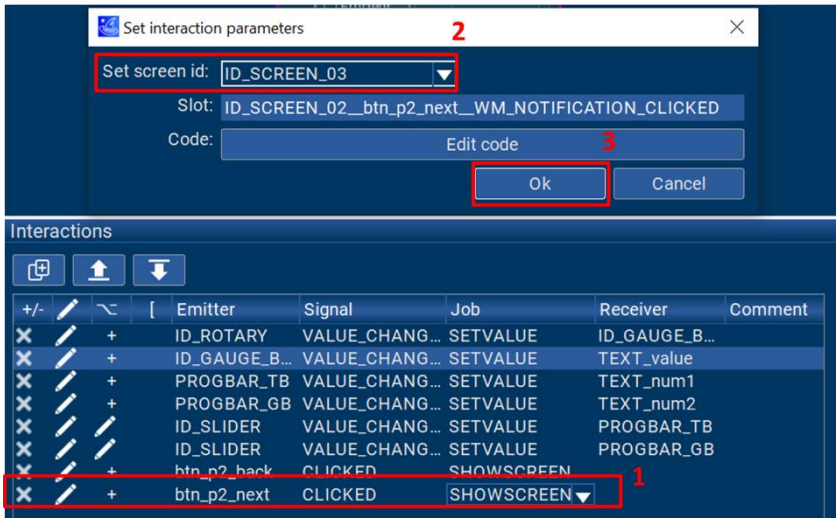


圖 2-99 btn_p2_next 換頁設置

2.3.3 三相電流顯示設計(頁面三)

三相電流應用大多需要顯示電能波形，由於目前 AppWizard 不支援 Graphic 物件，故波形顯示需透過 emWin library 以 GUI_Builder 設計，再將 emWin dialog 整合到 appWizard 專案中，此章節將詳述 emWin dialog 設置波形以及將其嵌入 appWizard 的方法，此設計所用物件如表 2-4~表 2-5 所示。

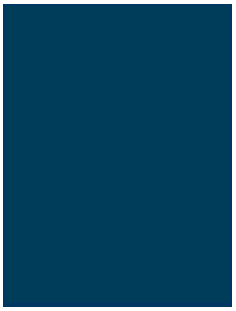
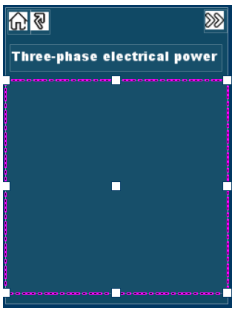
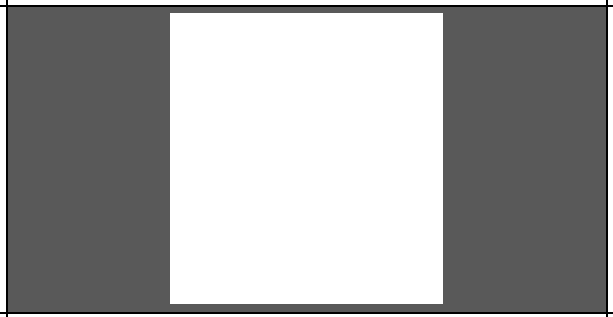
New Energy Gateway Description	Application GUI	Widget Type
背景圖 240x320		Box
標題文字		Text
emWin dialog 放置區		window
回主頁按鈕		Button
上一頁按鈕		
下一頁按鈕		

表 2-4 三相電流顯示設計物件 (AppWizard)

New Energy Gateway Description	Application GUI	Object Type
240x230 物件框		window
顯示波形選擇		Check Box

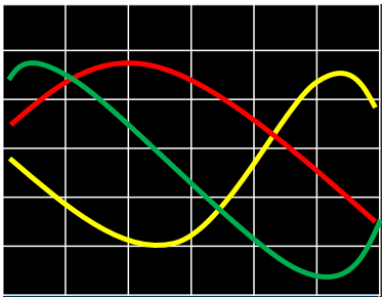
波形顯示		Graphic

表 2-5 三相電流顯示設計物件 (emWin)

2.3.3.1 EmWin Dialog 設置

1. 開啟 Gui_Builder.exe，新增大小為 240*220 的 Window 物件作為 emWin dialog 基底。(位置: \EC_NUC980_AppWizard_New_Energy_Gateway_V1.00\Software Tools\GUI_builder)

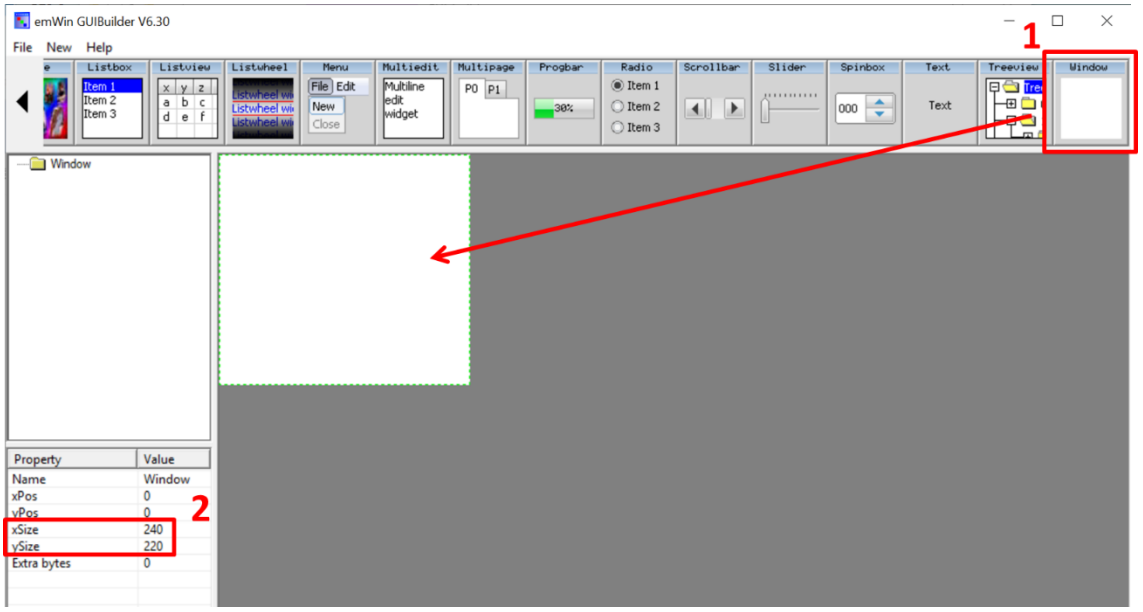


圖 2-100 新增 window 物件並調整大小

2. 新增大小為 240x174 的 Graphic 物件作波形繪製，並將其位置貼齊 Window 物件下方。

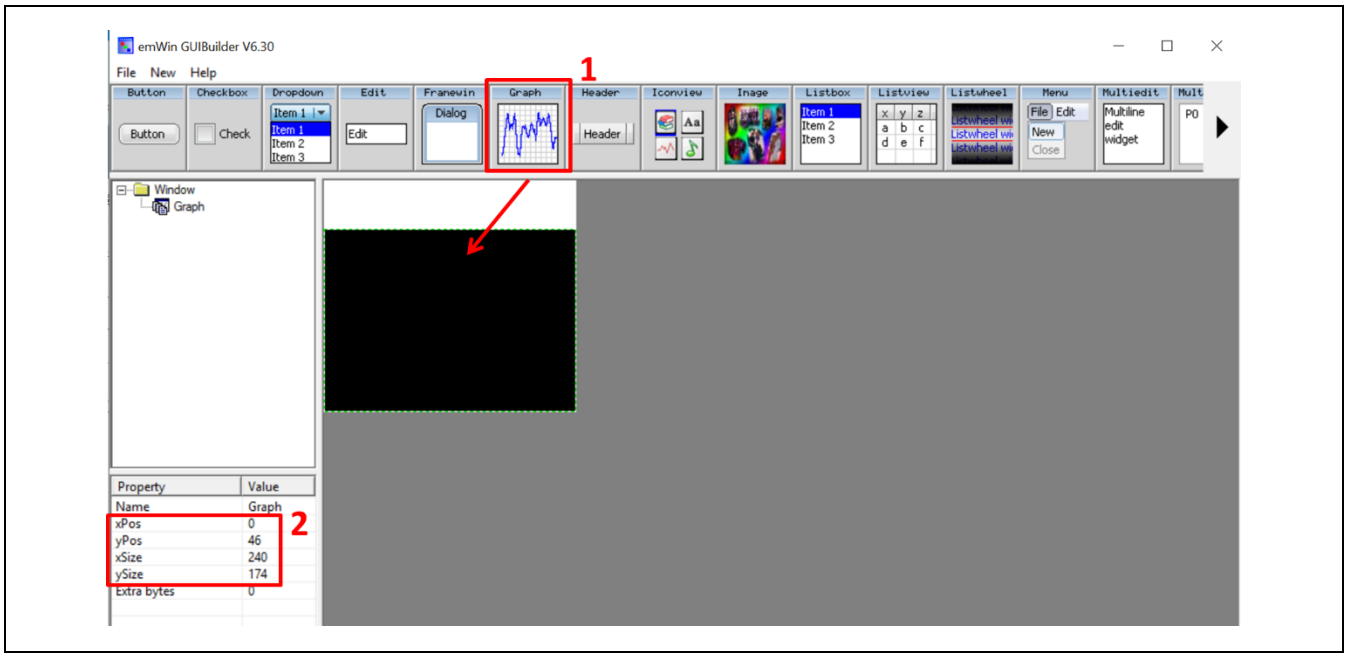


圖 2-101 新增 Graphic 物件並調整位置

3. 新增三個 checkbox 物件作為各波形顯示的功能開關。

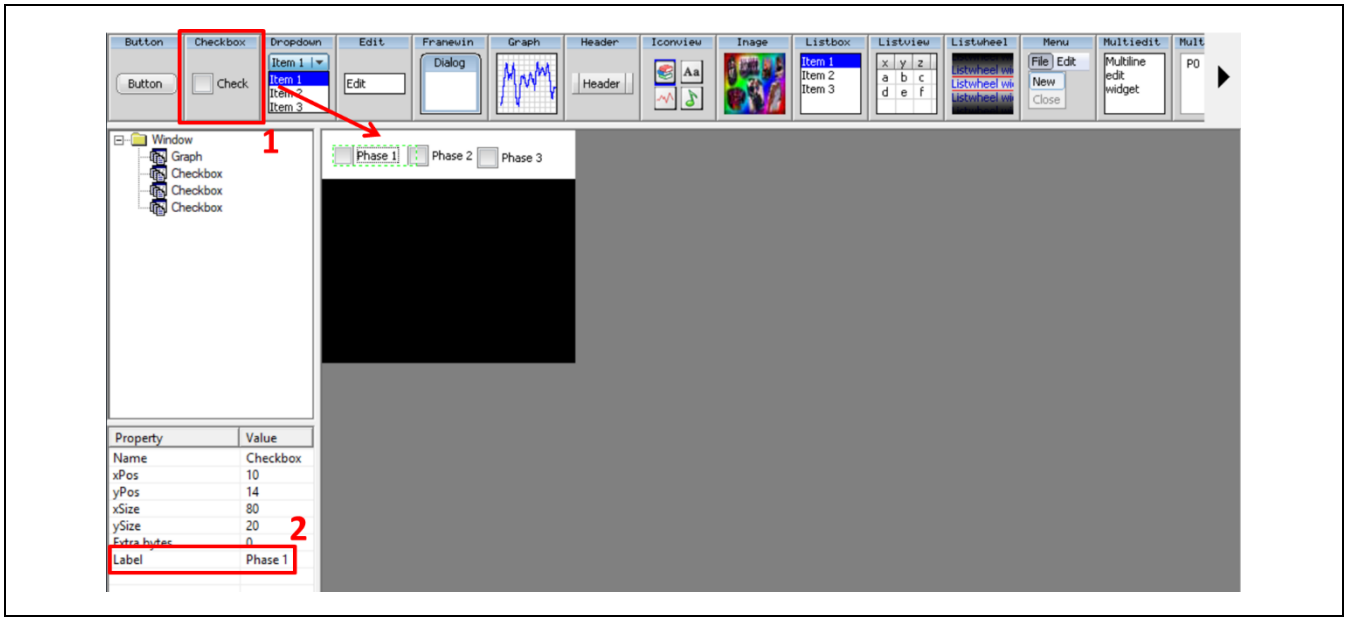


圖 2-102 新增 Checkbox 物件並調整顯示名稱

4. 介面設置完後，點選左上角工具列進行存檔，檔案將自動儲存於與 GUI_Builder.exe 相同目錄底下。

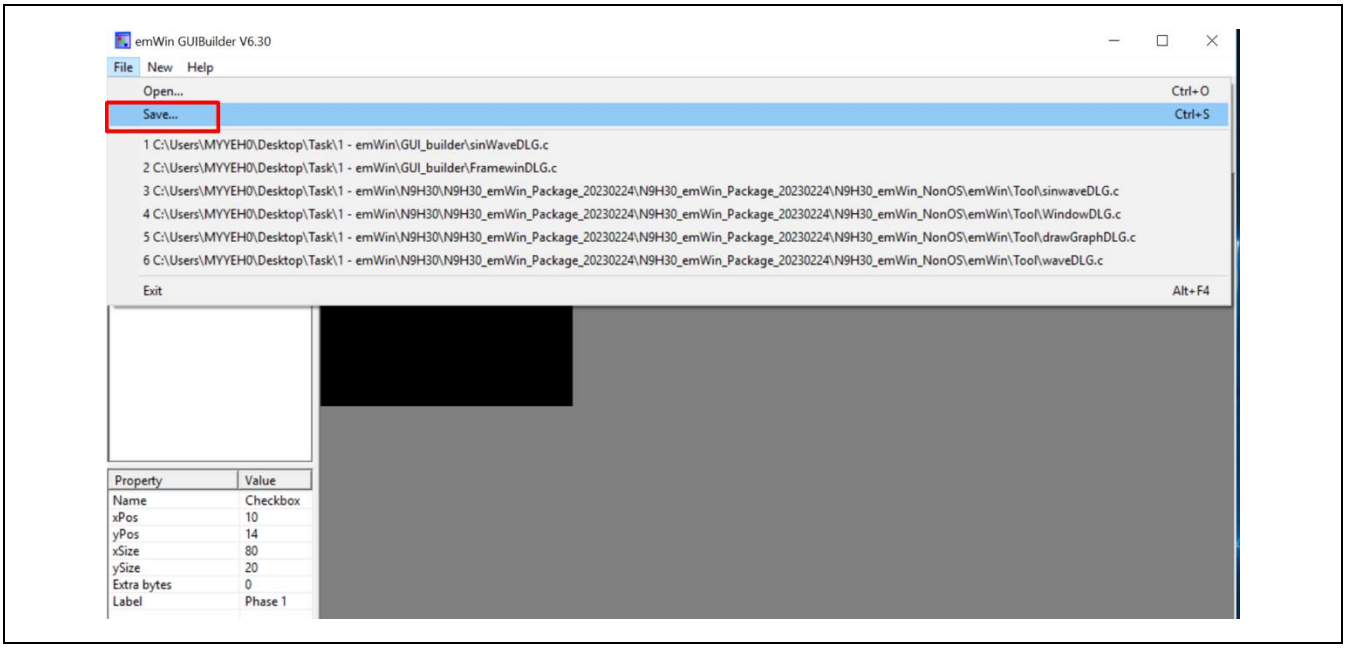


圖 2-103 儲存 emWin dialog

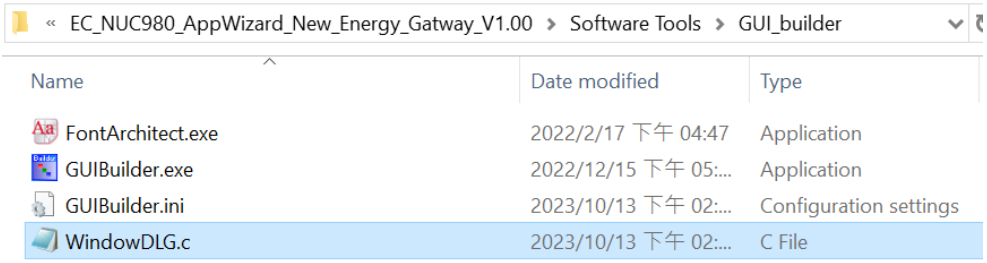


圖 2-104 emWin 檔案預設儲存位置

5. 開啟完成的 emWin dialog 檔案(DLG.c)，在*_cbDialog* 函式內新增程式碼控制波形的繪製，並設置透過切換 check box 選擇是否顯示指定線條的功能。

   1) 宣告陣列 ahData[3]來控制波形。(若要顯示三項波形數值，可另外宣告一組陣列儲存數值，並於 AppWizard 內新增相關變數接收數值。詳細方法請參考 AppWizard 官方範例 *HowTo_AddDialog*)

```
static void _cbDialog(WM_MESSAGE * pMsg) {
  WM_HWIN hItem;
  int     NCode;
  int     Id;
  // USER START (Optionally insert additional variables)
```

```
  static GRAPH_DATA_Handle ahData[3];
  static int               a;
  int                      v;
  //U16                    aIds[3] = { ID_VAR_V1, ID_VAR_V2, ID_VAR_V3 };  //Optional
  int                      i;
  // USER END
```

2) 在”case WM_INIT_DIALOG”內設定變數初始值。

```
switch (pMsg->MsgId) {
  case WM_INIT_DIALOG:
    …

    hItem = WM_GetDialogItem(pMsg->hWin, ID_CHECKBOX_2);
    CHECKBOX_SetText(hItem, "Phase 3");

    // USER START (Optionally insert additional code for further widget initialization)
    hItem     = WM_GetDialogItem(pMsg->hWin, ID_GRAPH_0);
    v         = WM_GetWindowSizeX(pMsg->hWin);
    ahData[0] = GRAPH_DATA_YT_Create(GUI_RED,    v, NULL, 0);
    ahData[1] = GRAPH_DATA_YT_Create(GUI_GREEN,  v, NULL, 0);
    ahData[2] = GRAPH_DATA_YT_Create(GUI_YELLOW, v, NULL, 0);
    GRAPH_SetGridVis(hItem, 1);
    WM_CreateTimer(pMsg->hWin, 0, 25, 0);
    // USER END
    break;
```

3) 在 switch (pMsg->MsgId)內新增 case WM_TIMER，使波形數值隨時間更新。

```
static void _cbDialog(WM_MESSAGE * pMsg) {
 …

switch (pMsg->MsgId) {
    …
    // USER START (Optionally insert additional message handling)
    case WM_TIMER:
       a++;
       if (a == 360) { a = 0; }
       for (i = 0; i < (int)GUI_COUNTOF(ahData); i++) {
           v = (50 * GUI__SinHQ((a + (i * 90)) * 1000)) >> 16;
           v += 115 / 2;
           GRAPH_DATA_YT_AddValue(ahData[i], v);
           //APPW_SetVarData(aIds[i], a + (i * 90));  //Optional
       }
       WM_RestartTimer((WM_HTIMER)pMsg->Data.v, 0);
       break;
    // USER END
…
```

4) 於 case WM_NOTIFY_PARENT 內新增一組 Switch-Case，使程式依 checkbox 的狀態更新 GRAPHIC 物件的數值。

```
switch (pMsg->MsgId) {
    …
  case WM_NOTIFY_PARENT:
    …
    switch(Id) {
```

```
        case ID_CHECKBOX_0: // Notifications sent by 'Checkbox'
          switch(NCode) {
                …
                case WM_NOTIFICATION_VALUE_CHANGED:
                  // USER START
                  hItem = WM_GetDialogItem(pMsg->hWin, ID_GRAPH_0);
                  if (CHECKBOX_GetState(pMsg->hWinSrc) == 1) {
                    GRAPH_AttachData(hItem, ahData[0]);
                  } else {
                    GRAPH_DetachData(hItem, ahData[0]);
                  }
                  // USER END
                  break;
        case ID_CHECKBOX_1: // Notifications sent by 'Checkbox'
          switch(NCode) {
                …
                case WM_NOTIFICATION_VALUE_CHANGED:
                  // USER START
                  hItem = WM_GetDialogItem(pMsg->hWin, ID_GRAPH_0);
                    GRAPH_AttachData(hItem, ahData[1]);
                  } else {
                  GRAPH_DetachData(hItem, ahData[1]);
                  }
                  // USER END
                  break;

        case ID_CHECKBOX_2: // Notifications sent by 'Checkbox'
          switch(NCode) {
                …
                case WM_NOTIFICATION_VALUE_CHANGED:
                    // USER START
                   hItem = WM_GetDialogItem(pMsg->hWin, ID_GRAPH_0);
                    if (CHECKBOX_GetState(pMsg->hWinSrc) == 1) {
                        GRAPH_AttachData(hItem, ahData[2]);
                    } else {
                       GRAPH_DetachData(hItem, ahData[2]);
                    }
                    // USER END
                break;
                    // USER START
                    // USER END
```

5) 將更新後的 emWin dialog 檔案置於 AppWizard project 指定路徑 (/Source/CustomCode) 下，位置如圖 2-105 所示。

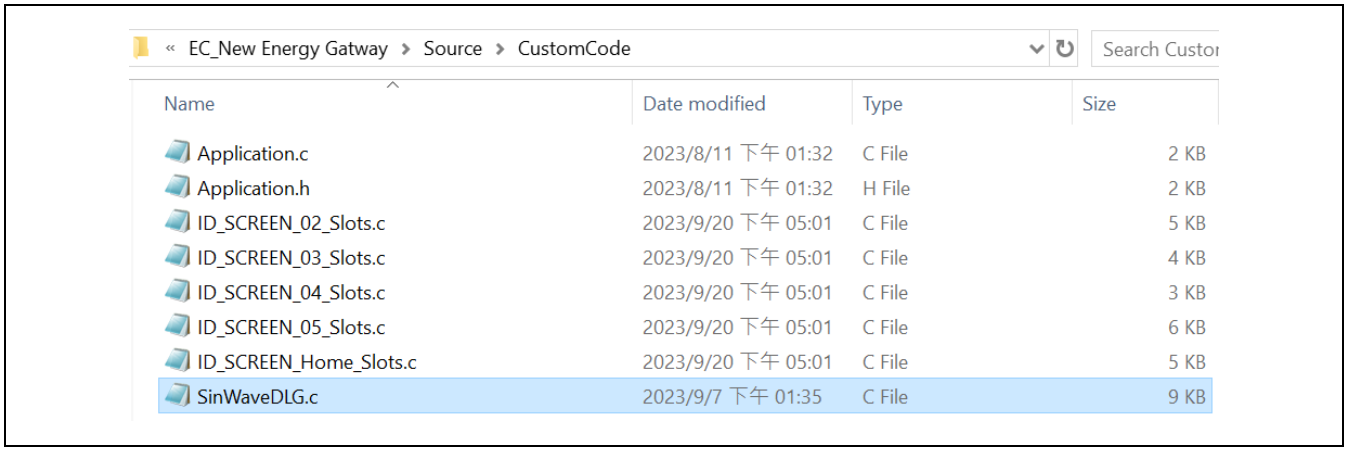


圖 2-105 EmWin Dialog 放置位置

2.3.3.2 GUI 物件配置

1. 點選 ID_SCREEN_03 開始編輯三相電流顯示頁面，透過 Add objects 新增三個 Button 物件，以圖片 home_20x20_new.png、arrow2_20x20_new.png、arrow1_20x20_new.png 作為回首頁(btn_p3_home)、上一頁(btn_p3_back)與下一頁(btn_p3_next)的按鈕圖示，並依圖 2-106 ~ 圖 2-108 設置屬性。

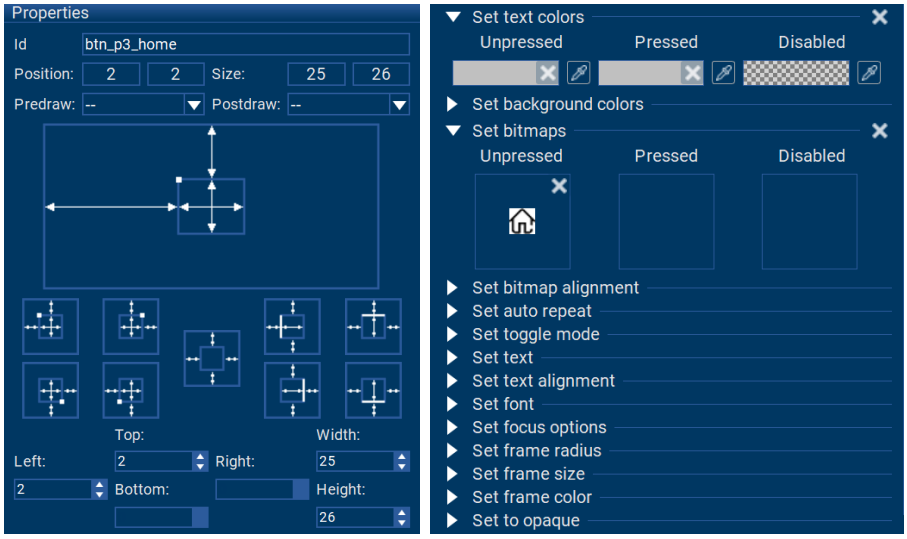


圖 2-106 btn_p3_home 屬性設置

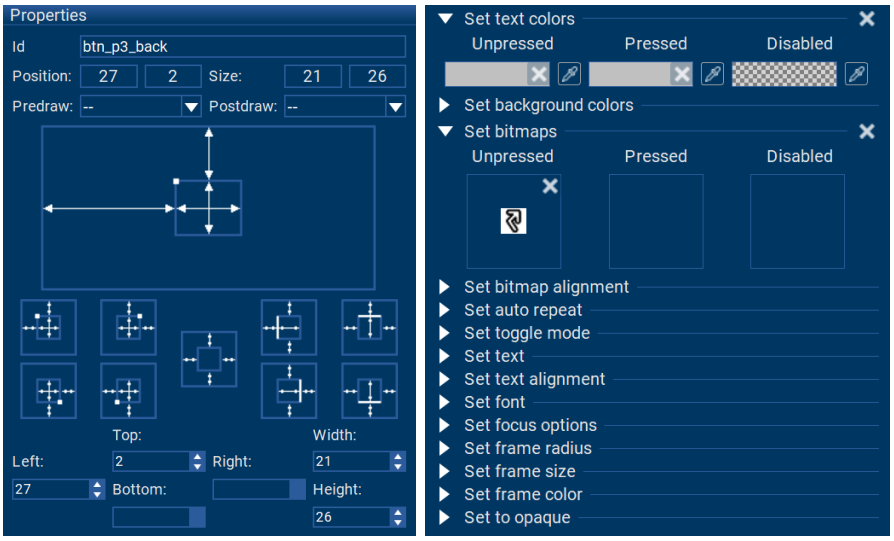


圖 2-107 btn_p3_back 屬性設置

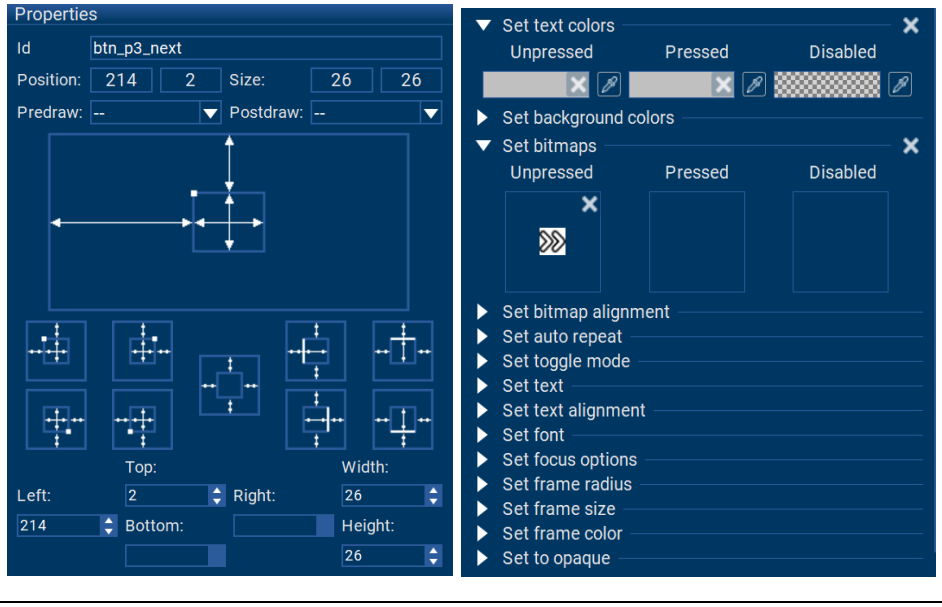


圖 2-108 btn_p3_next 屬性設置

2. 為了在 appWizard 內嵌入 emwin dialog，因此新增一個大小與 emWin dialog 相同的 window 物件(emwin_window)，並依圖 2-110 設置物件屬性。

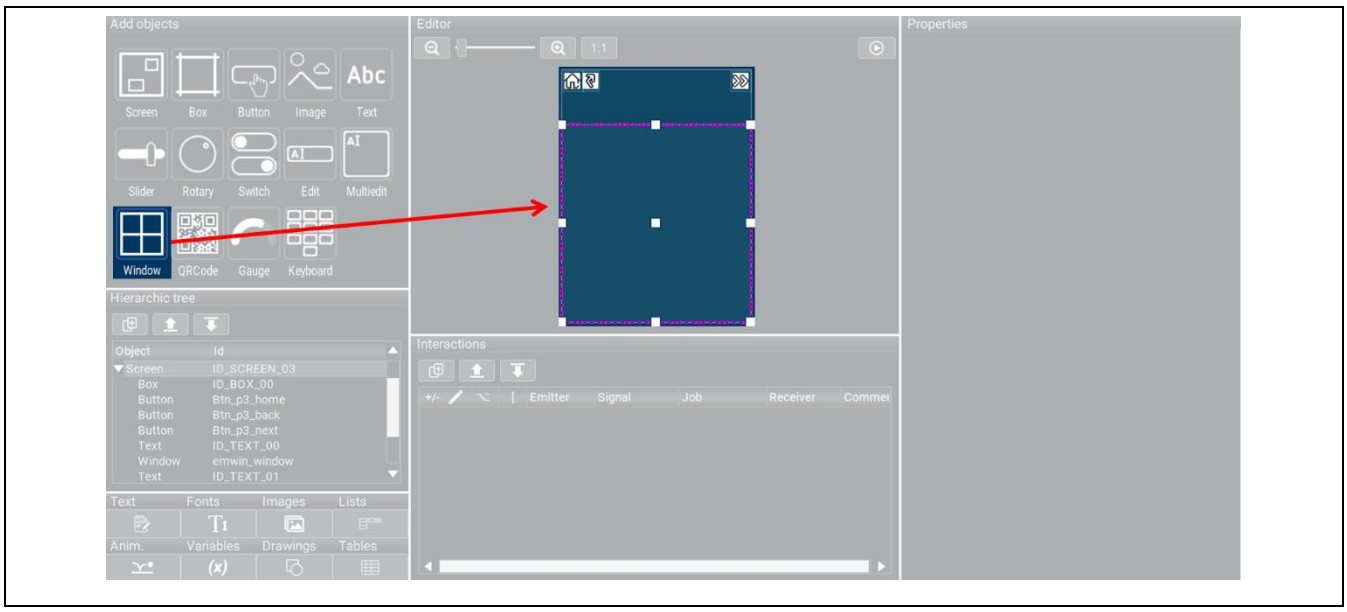


圖 2-109 新增 window 物件

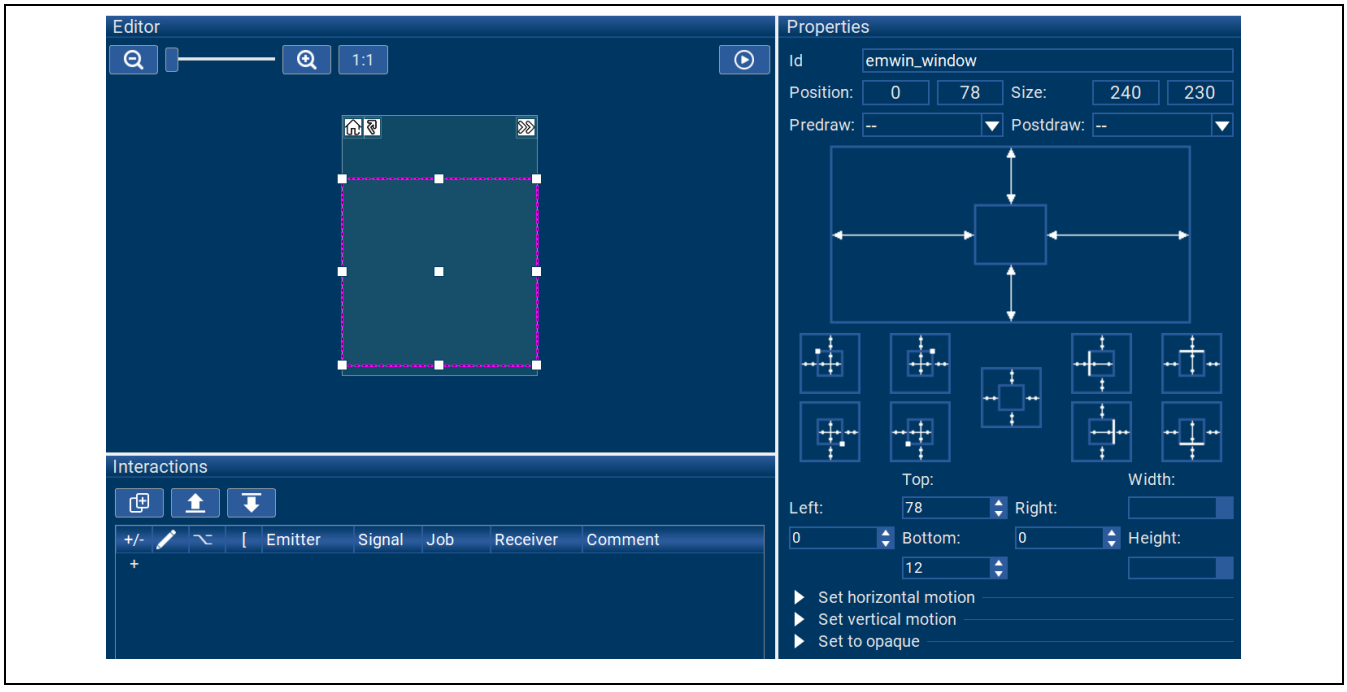


圖 2-110 emwin_window 屬性設置

3. 新增一個 TEXT 物件(ID_TEXT_00)顯示標題文字，依圖 2-111~圖 2-113 設置顯示文字”Three-phase electrical power”與屬性。

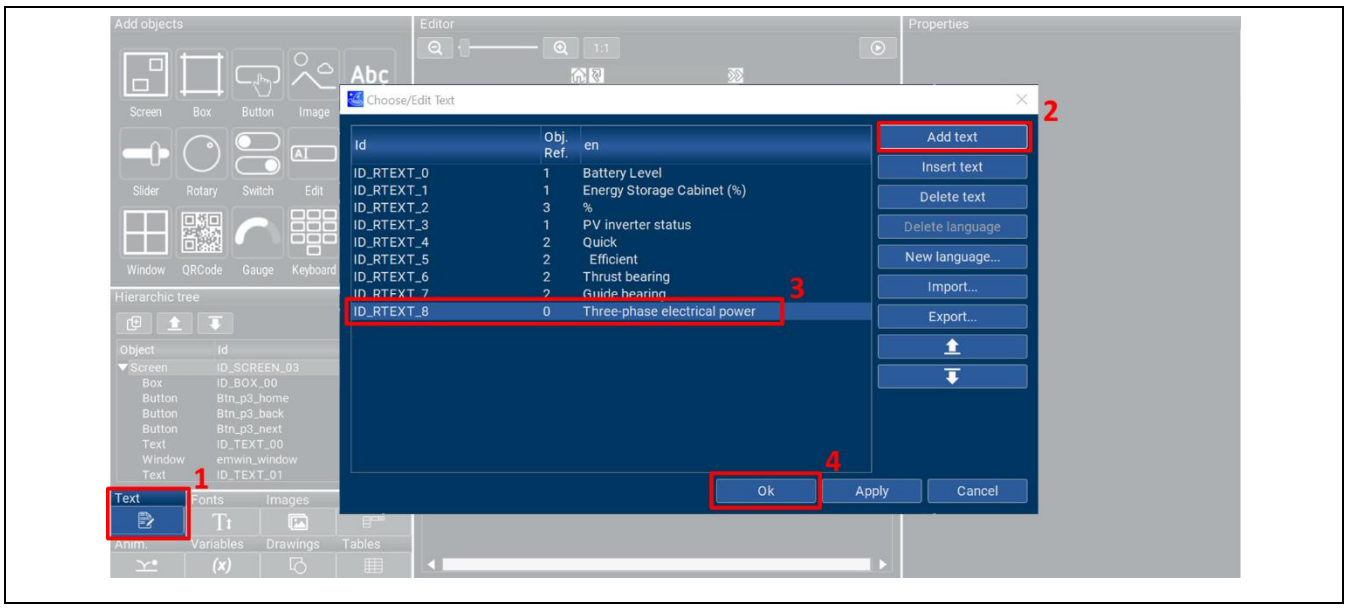


圖 2-111 新增顯示文字(ID_RTEXT_8)

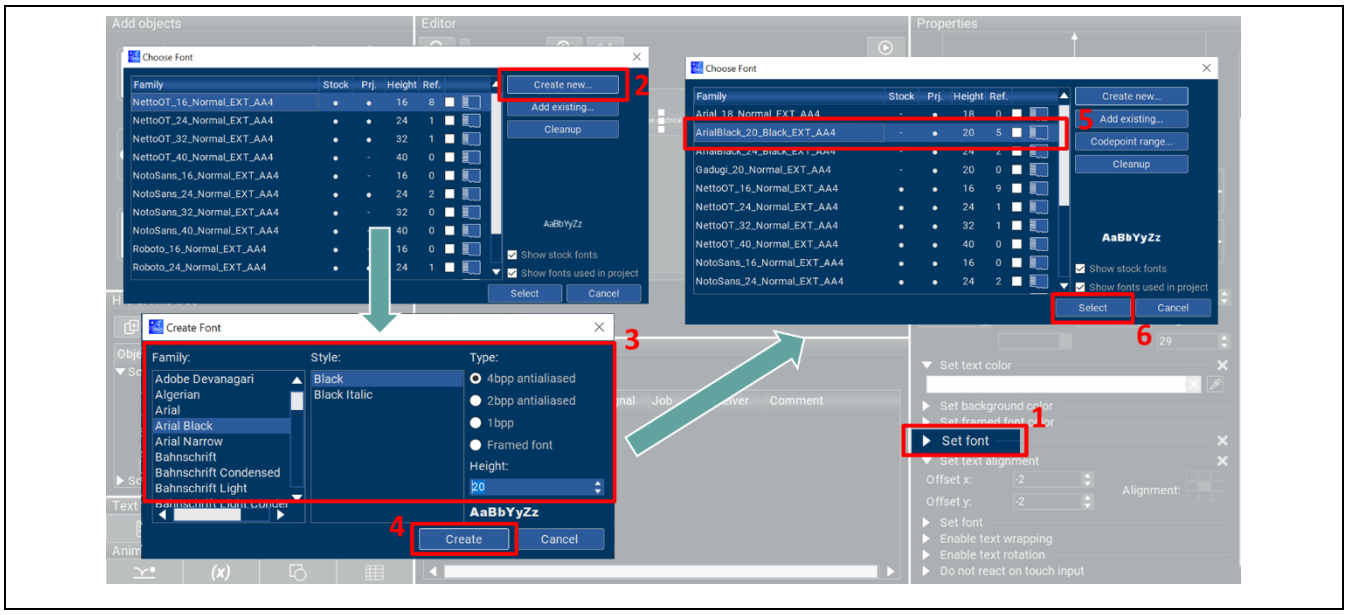


圖 2-112 新增自訂文字格式

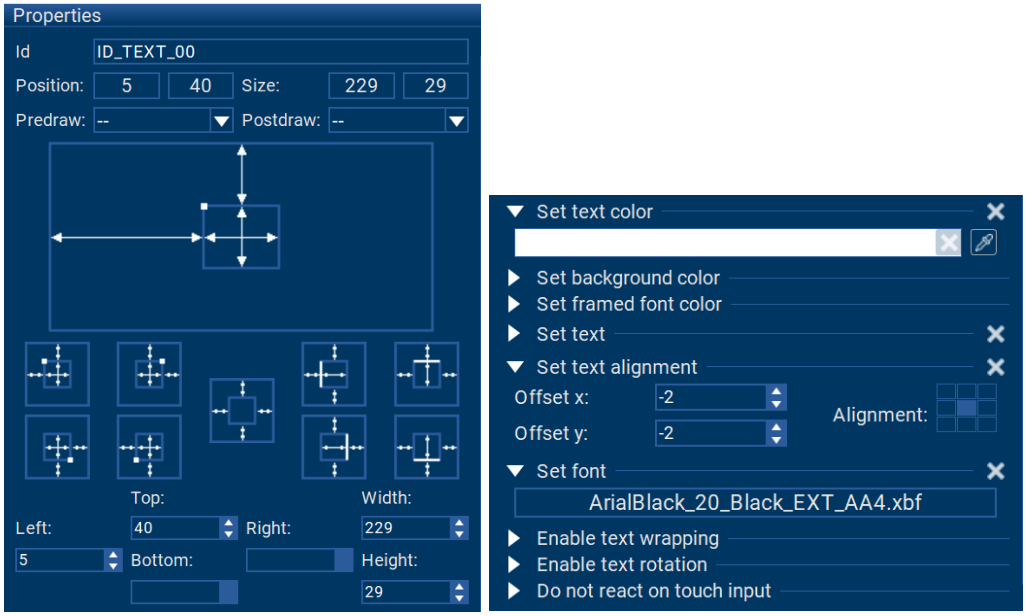


圖 2-113 ID_TEXT_00 屬性設置

2.3.3.3 將 EmWin Dialog 放入 AppWizard 專案

要將 emWin dialog 嵌入 AppWizard 專案的話，需透過 AppWizard 的 interactions 操作，詳細流程如下所述。

1. 在 interactions 內新增一組 **Emitter** 為 *ID_SCREEN_03*(要被嵌入 emwin dialog 的 screen ID)、**Signal** 為 *INITDIALOG* 且 **Job** 為 *NULL* 的交互設置，並設定 Slot 名稱為 *ID_SCREEN_03_Init*。(Slot 代表此動作在 *ID_SCREEN_03_slot* 內的函式名稱)

圖 2-114 EmWin Dialog 交互作用設置

2. 先將目前專案導出，以便更新程式使 emWin dialog 結合到 AppWizard 專案中。

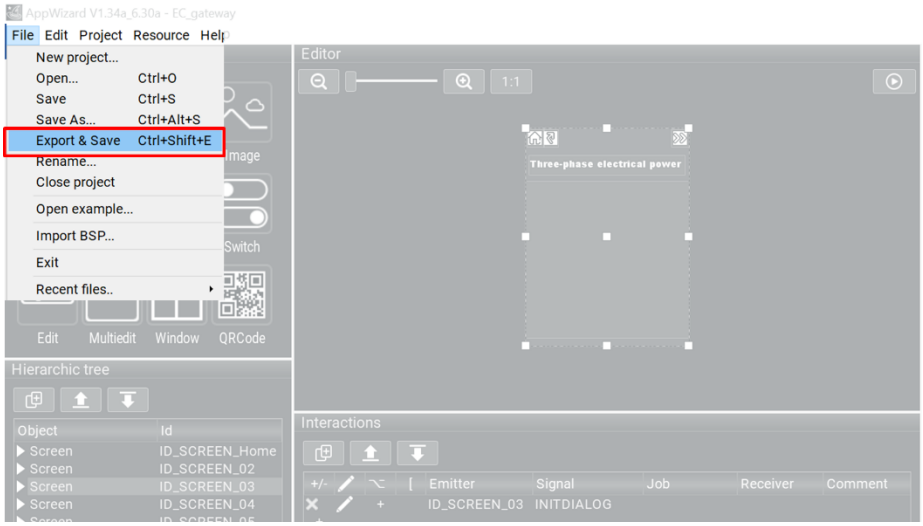


圖 2-115 導出目前專案

3. 在專案資料夾位置*Source**CustomCode* 開啟要嵌入 emWin dialog 的介面程式 (ID_SCREEN_03_Slots.c)，找到步驟 1 設置的 slot 函式名稱(ID_SCREEN_03_Init)後，對內容進行下述修改：

```
void ID_SCREEN_01_Init(APPW_ACTION_ITEM * pAction, WM_HWIN hScreen, WM_MESSAGE * pMsg, int
* pResult) {
  WM_HWIN hWindow;

  GUI_USE_PARA(pAction);
  GUI_USE_PARA(hScreen);
  GUI_USE_PARA(pMsg);
  GUI_USE_PARA(pResult);

  //assign window ID
  hWindow = WM_GetDialogItem(hScreen, emwin_window);
  CreateDialog(hWindow);
}
```

4. 在相同路徑開啟 emWin dialog 程式(SinWaveDLG.c)，依圖 2-116~圖 2-117 修改後存檔。

圖 2-116 新增 include 宣告

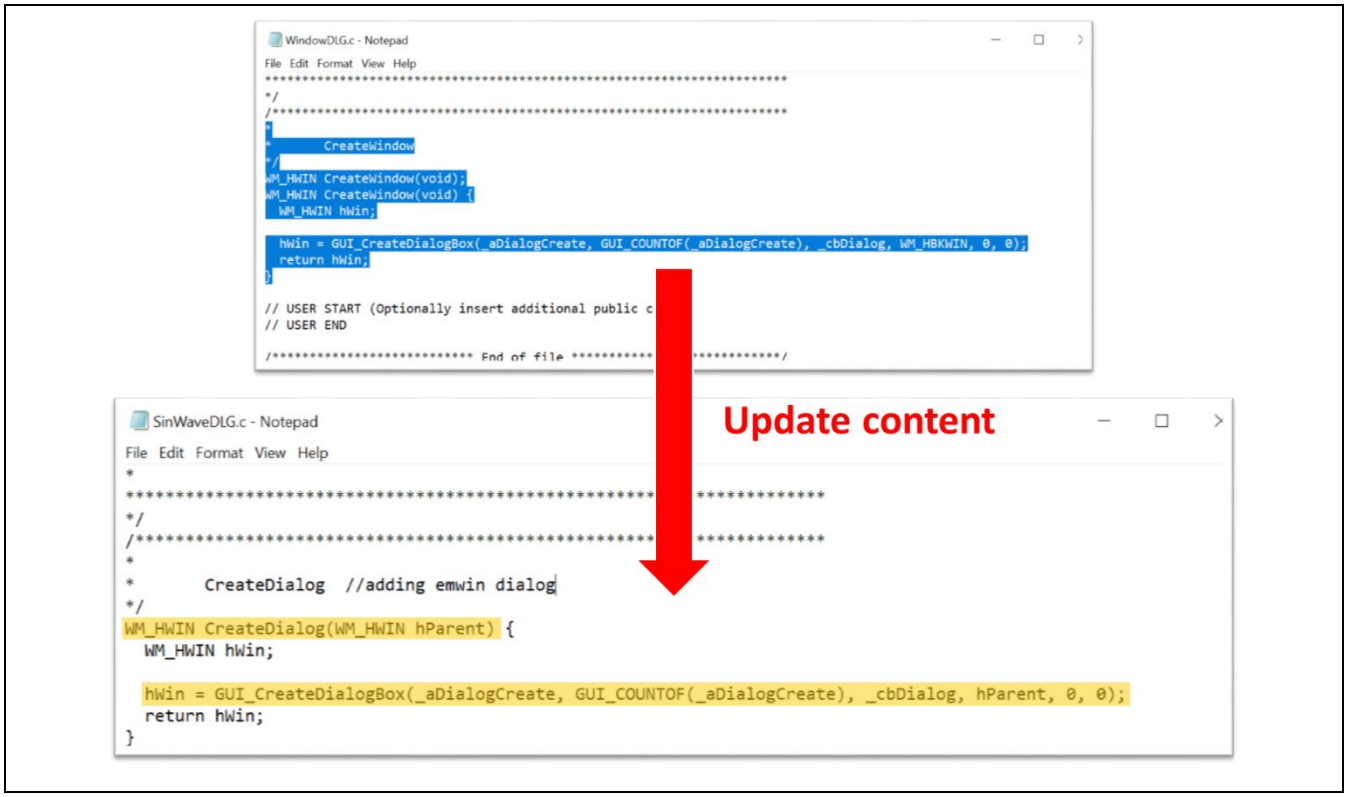


圖 2-117 更新 dialog 宣告函式

emWin dialog 內容：

```
…

#include "../Generated/Resource.h"
#include "../Generated/ID_SCREEN_03.h"

…
```

```
/*********************************************************************
*
*       CreateDialog   // emwin dialog
*/

WM_HWIN CreateDialog(WM_HWIN hParent) {
  WM_HWIN hWin;

  hWin = GUI_CreateDialogBox(_aDialogCreate, GUI_COUNTOF(_aDialogCreate), _cbDialog,
hParent, 0, 0);
  return hWin;
}
…
```

2.3.3.4 物件互動設置

物件設置完成畫面如圖 2-118，接著要建立物件間的交互作用(interactions)，使操作物件時能有相應行為顯示。 (Note: emWin dialog 效果無法在 AppWizard 預覽，需部屬到開發版確認)

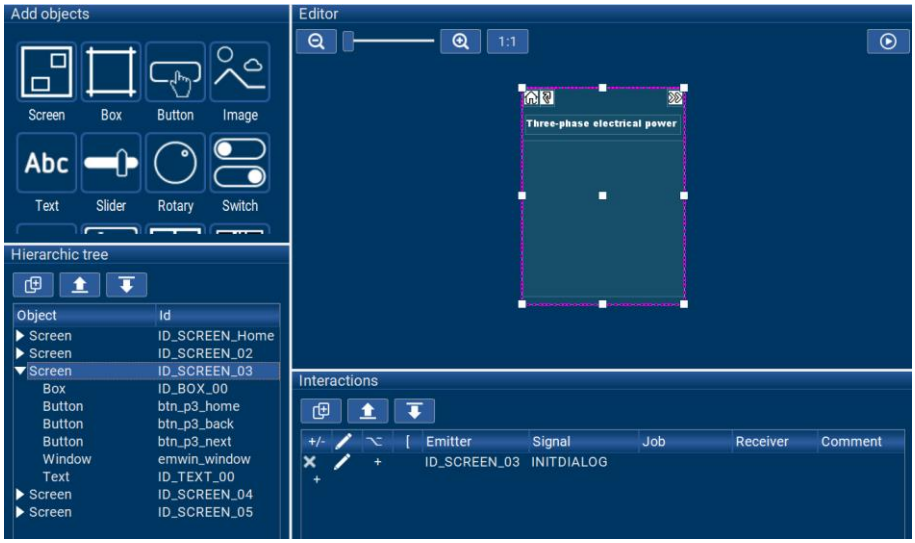


圖 2-118 頁面三設計完成圖

1. 設置 Button(btn_p3_home、btn_p3_back、btn_p3_next)物件的換頁功能，使點選按鈕可以切回首頁、前一頁及下一頁。

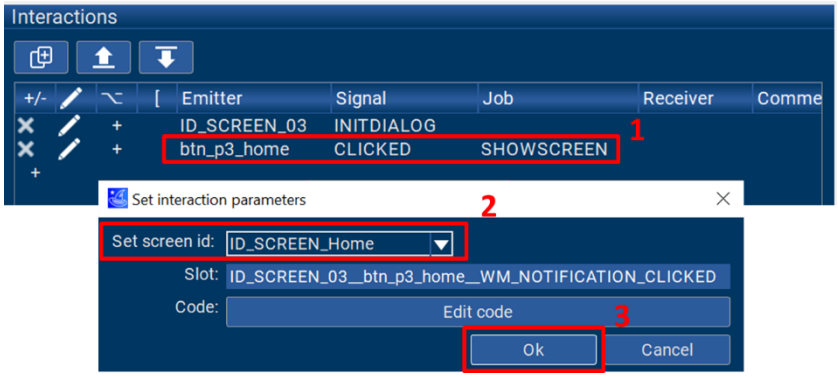


圖 2-119 btn_p3_home 換頁設置

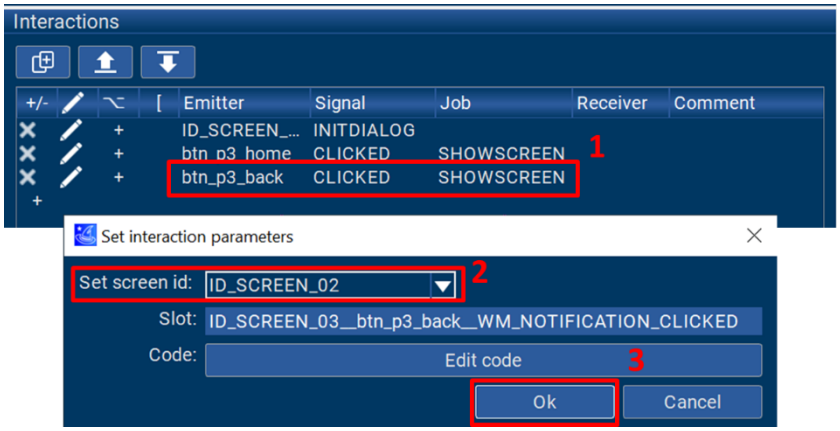


圖 2-120 btn_p3_ back 換頁設置

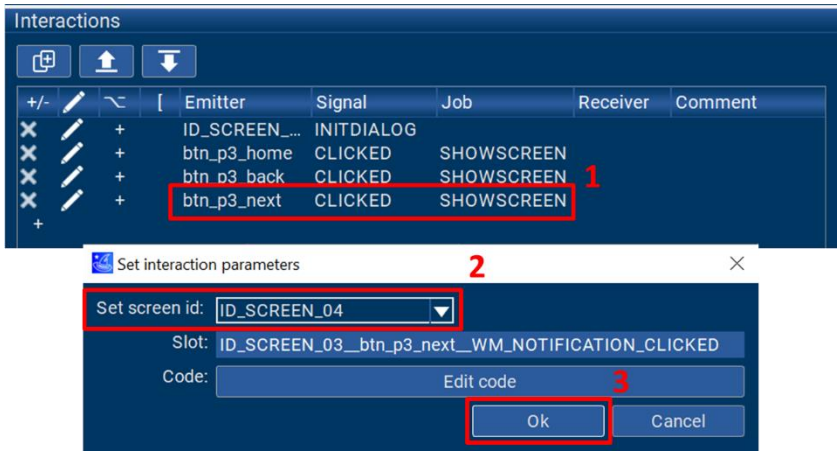


圖 2-121btn_p3_ next 換頁設置

2.3.4 用電量表設計(頁面四)

直條圖是閘道器常用來做為數值計量或呈現的方式之一，在此頁也結合 appWizard 的動畫顯示功能，使畫面內容更加豐富生動，且可利用 QR Code 導向特定網址供使用者查閱(可用於詳細用電量報表、資料分析等資料頁面的顯示)，此設計所用物件如表 2-6 所示。

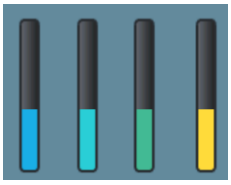
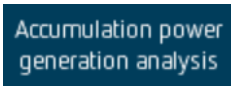
New Energy Gateway Description	Application GUI	Widget Type
背景圖 240x320		Box
動畫背景圖 200x207		Box
累積用電量長條圖		Progbar
網頁導向 QR code		QR code
標題文字		Text
長條圖說明		
QR code 說明		
回主頁按鈕		Button
上一頁按鈕		
下一頁按鈕		

表 2-6 用電量表設計物件

2.3.4.1 GUI 物件配置

1. 點選 ID_SCREEN_04 開始編輯用電量表畫面，透過 Add objects 新增三個 Button 物件，並分別以圖片 home_20x20_new.png、arrow2_20x20_new.png、arrow1_20x20_new.png 作為回首頁(btn_p4_home)、上一頁(btn_p4_back)與下一頁(btn_p4_next)的按鈕圖示。

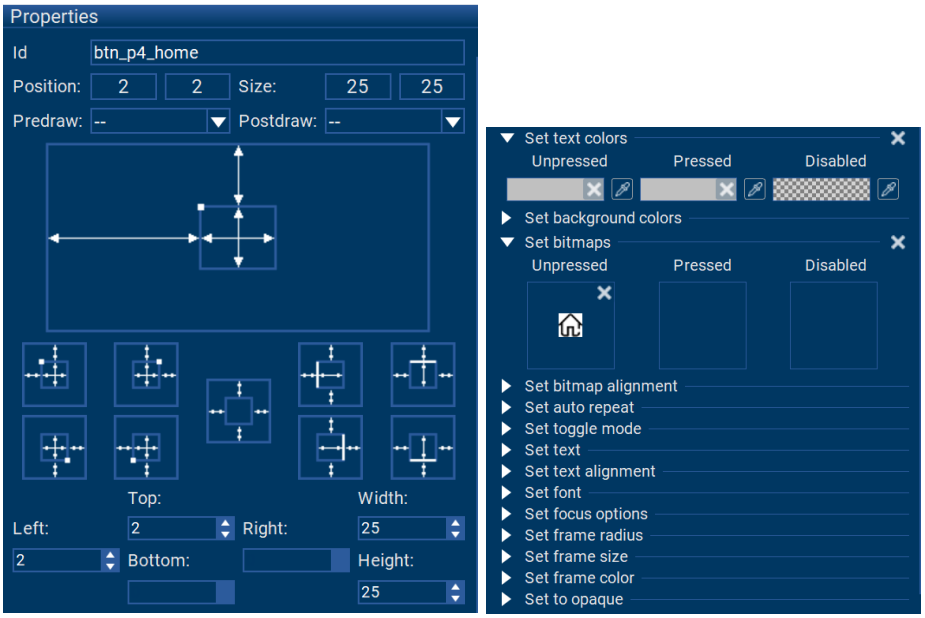


圖 2-122 btn_p4_home 屬性設置

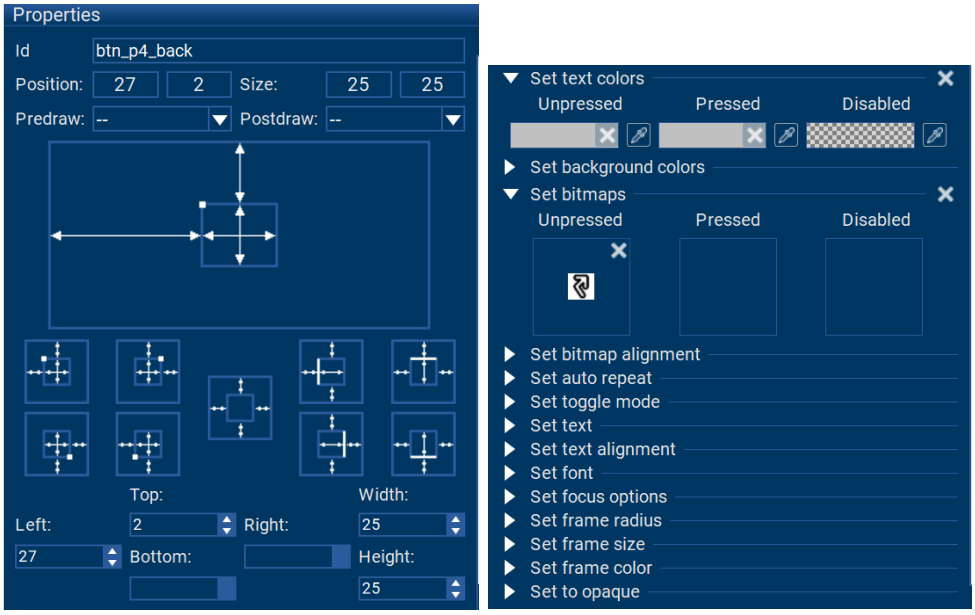


圖 2-123 btn_p4_back 屬性設置

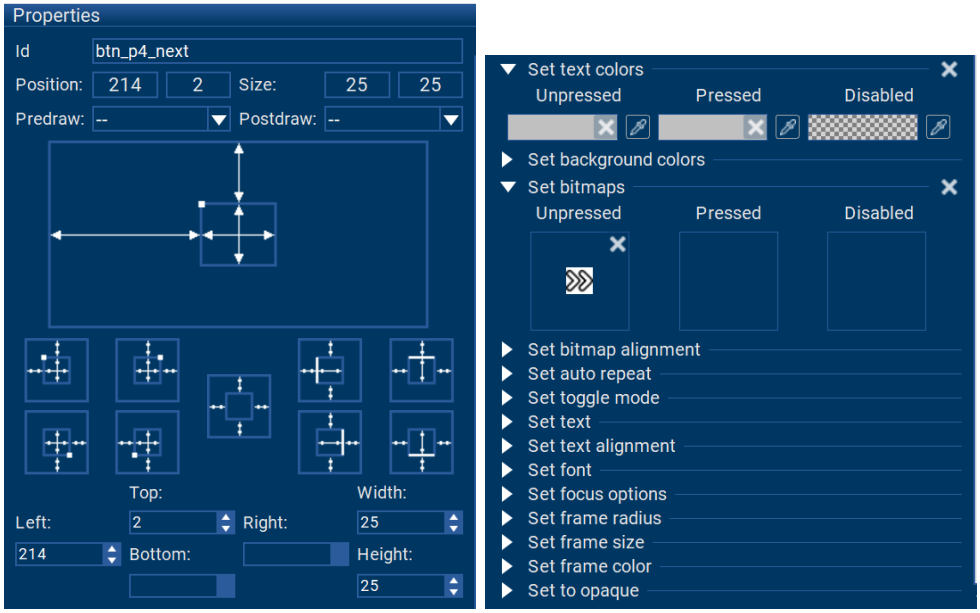


圖 2-124 btn_p4_next 屬性設置

2. 新增第二個 BOX 物件(ID_BOX_01)作為動畫區域的背景色，屬性設置如圖 2-125。

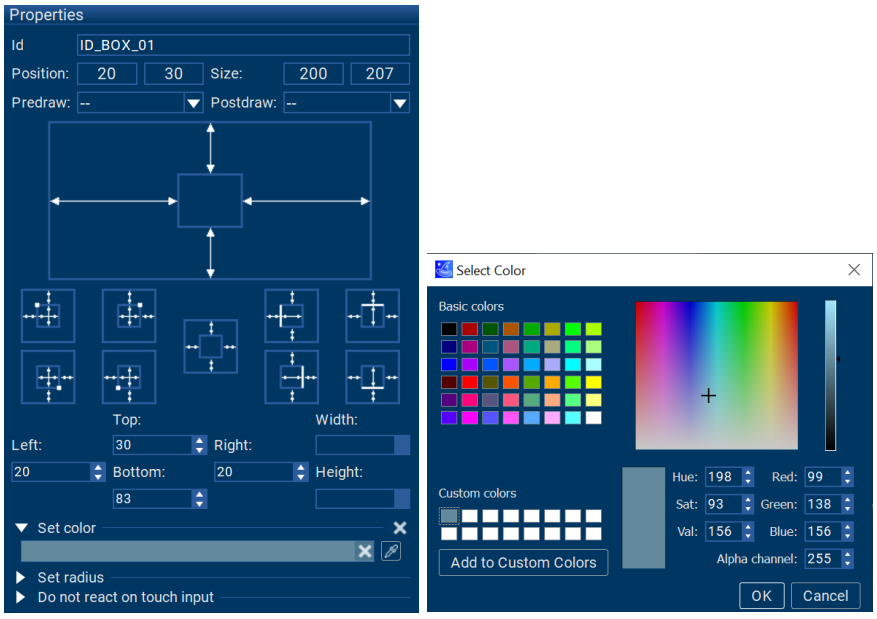


圖 2-125 ID_BOX_01 屬性設置

3. 新增四個 Progbar 物件(PROGBAR_00~PROGBAR_03)作為不同層級的電量使用顯示，依圖 2-126~圖 2-129 設置屬性。首先開啟 Progbar 的垂直模式，並分別以圖片 progbar_LightBlue.png、progbar_LightGreen.png、progbar_Green.png 與 progbar_Yellow.png 作為 Progbar 的 bitmap bottom，並選用圖片 DARK_Progbar_Tile_V_Gray_16x1.png 作為 bitmap top。

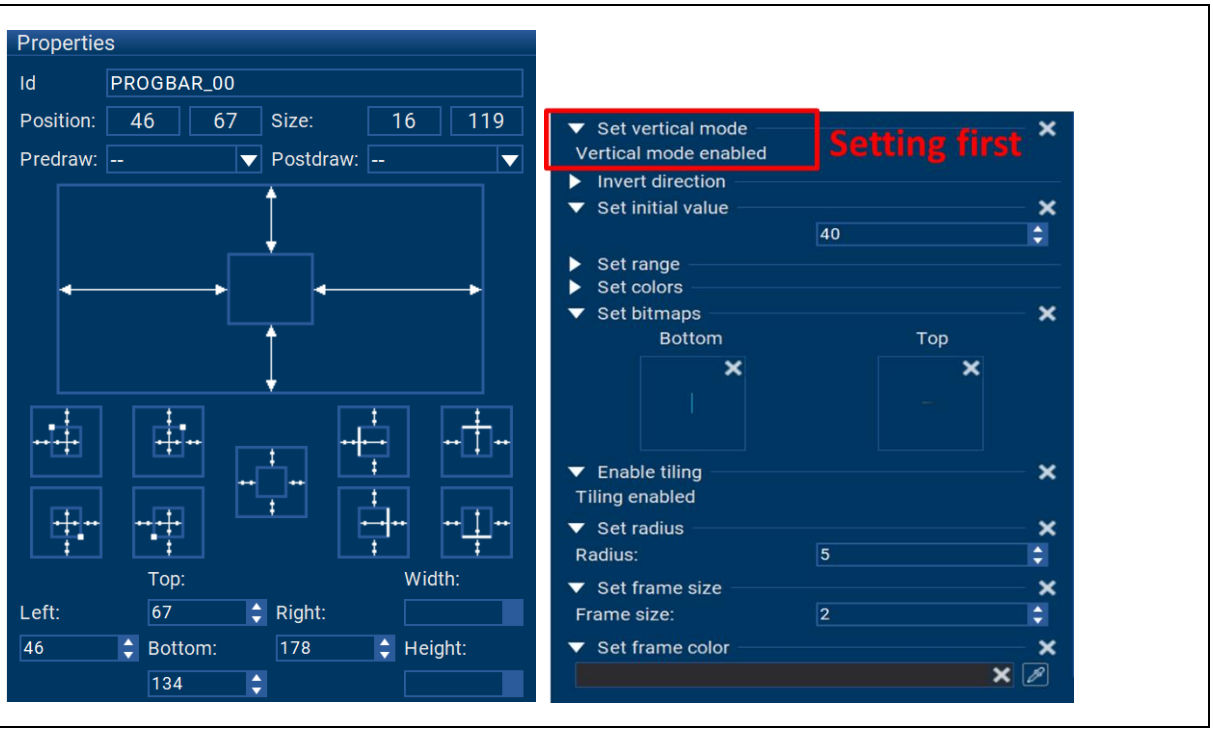


圖 2-126 PROGB AR_00 屬性設置

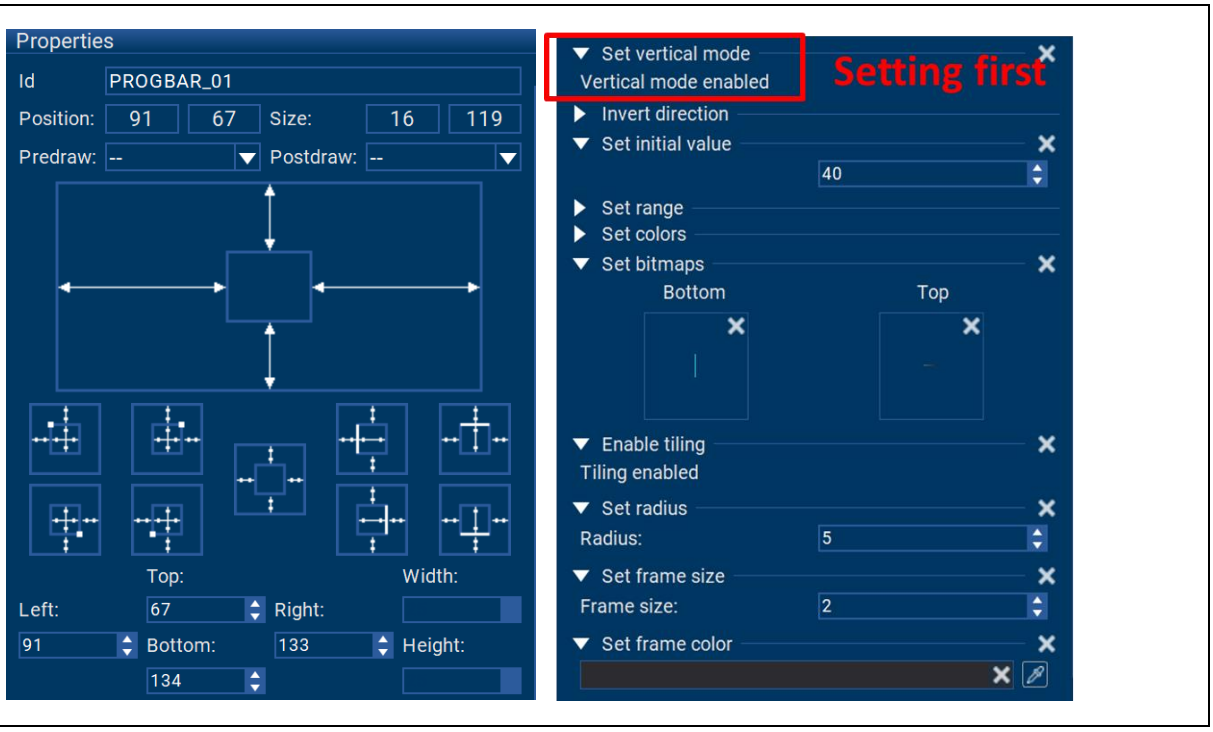


圖 2-127 PROGBAR_01 屬性設置

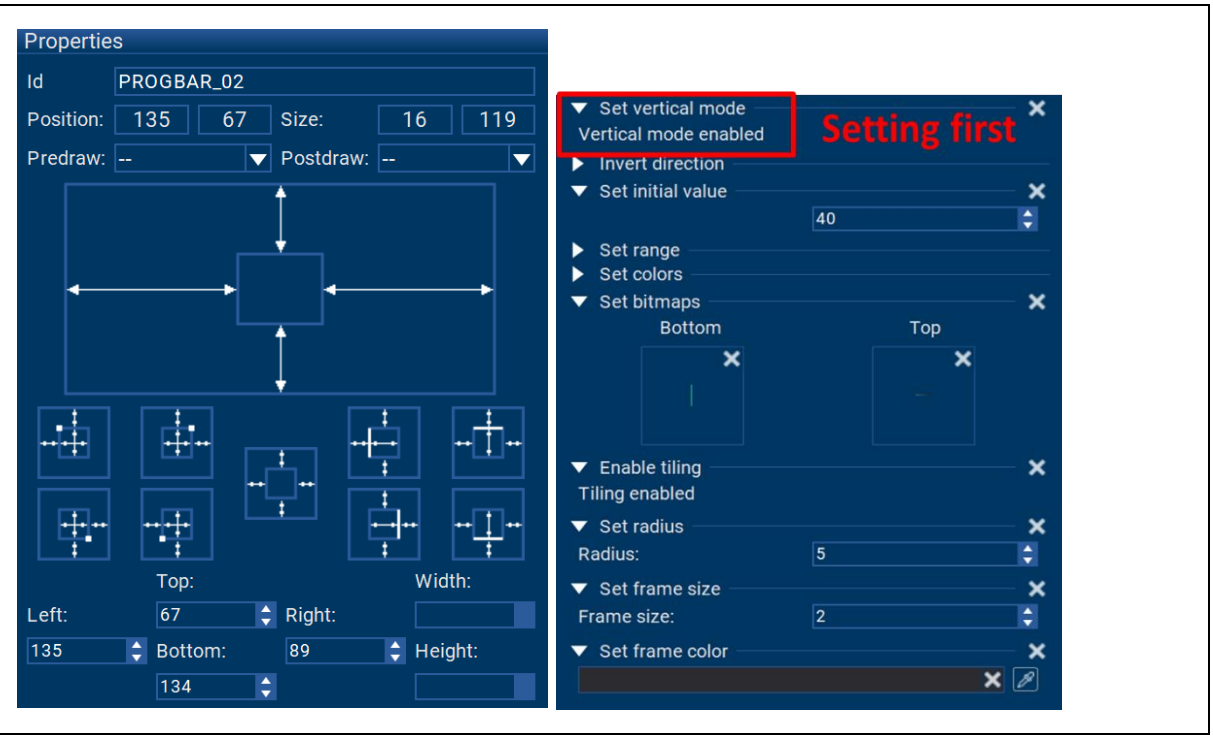


圖 2-128 PROGBAR_02 屬性設置

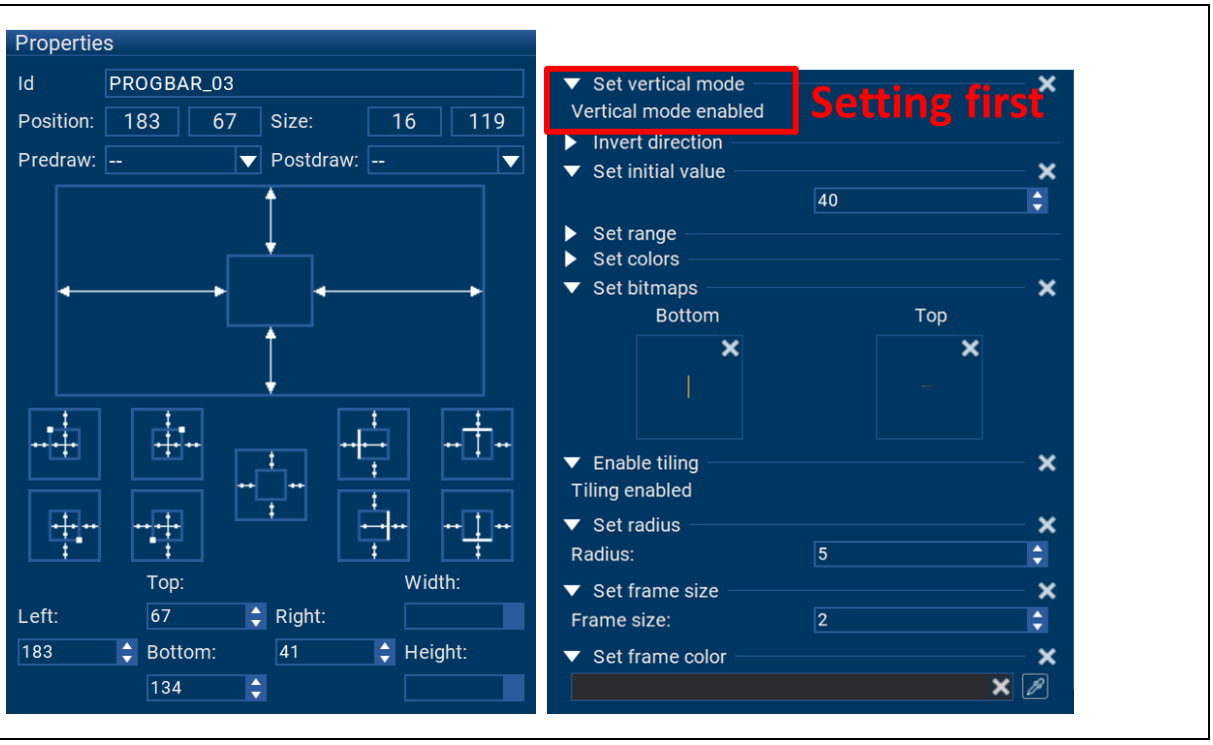


圖 2-129 PROGBAR_03 屬性設置

4. 新增兩個 TEXT 物件作為標題(ID_TEXT_00)與頁尾文字(ID_TEXT_01)，將顯示文字分別設為”Monitoring Report”與”Accumulation power generation analysis”，並依圖 2-132~圖 2-133 設置屬性。

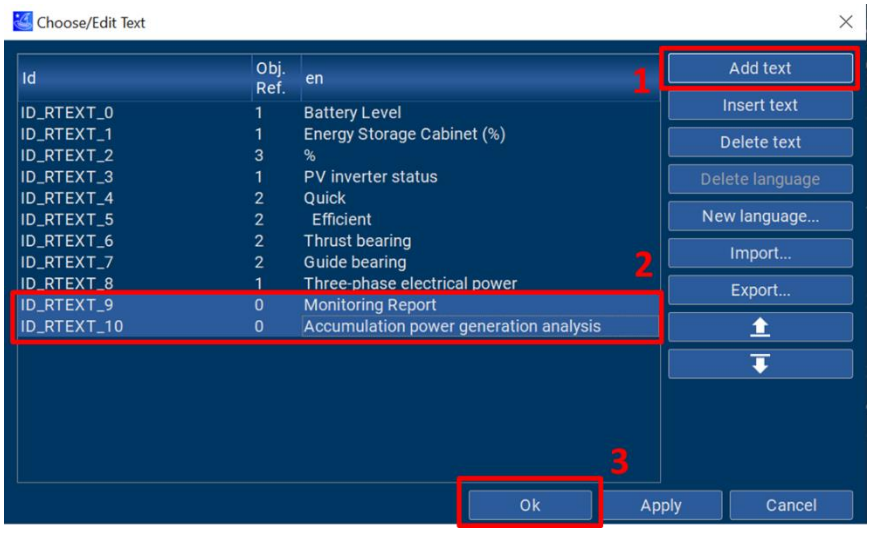


圖 2-130 新增顯示文字

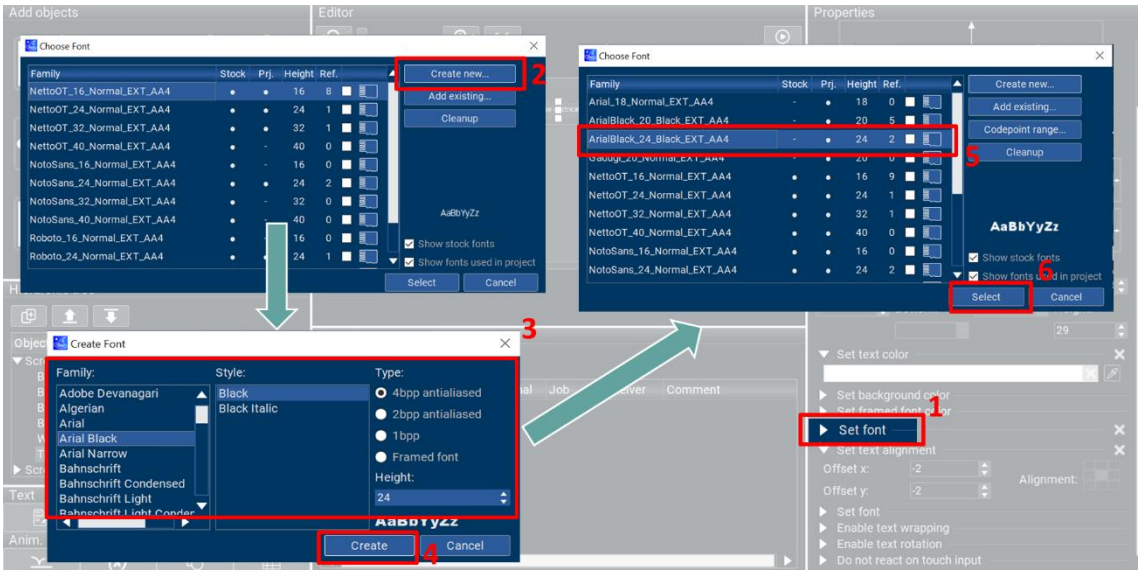


圖 2-131 設置自訂字體(ID_TEXT_00)

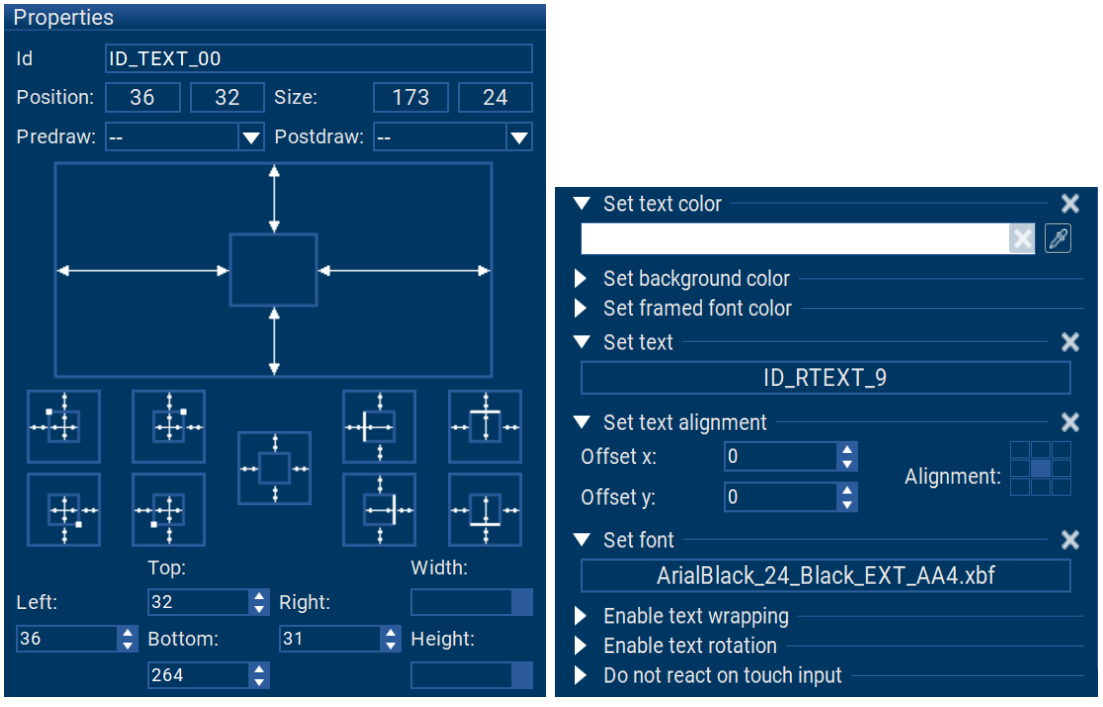


圖 2-132 ID_TEXT_00 屬性

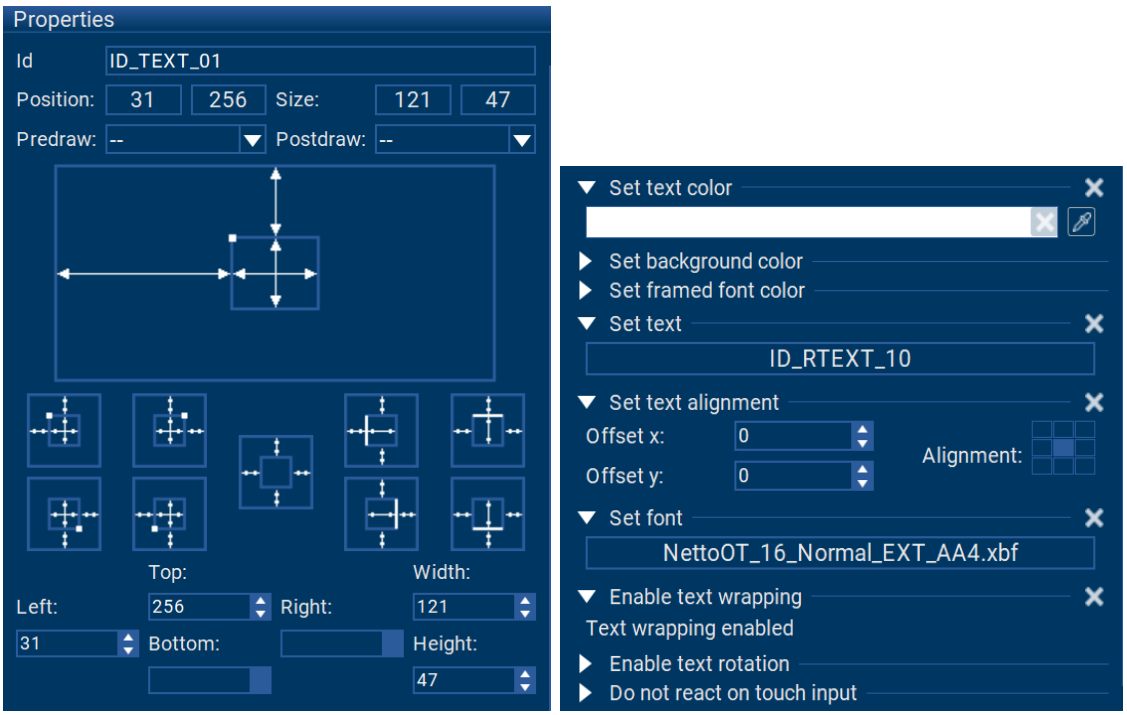


圖 2-133 ID_TEXT_01 屬性

5. 為了設置各長條圖圖說，新增四個 TEXT 物件(ID_TEXT_02~ID_TEXT_04)，分別將文字設為”Instant Power (W)”、“Daily (KW)”、“Monthly (MW)”與”Cumulative power(GW)”，依圖 2-135~圖 2-138 設置屬性。

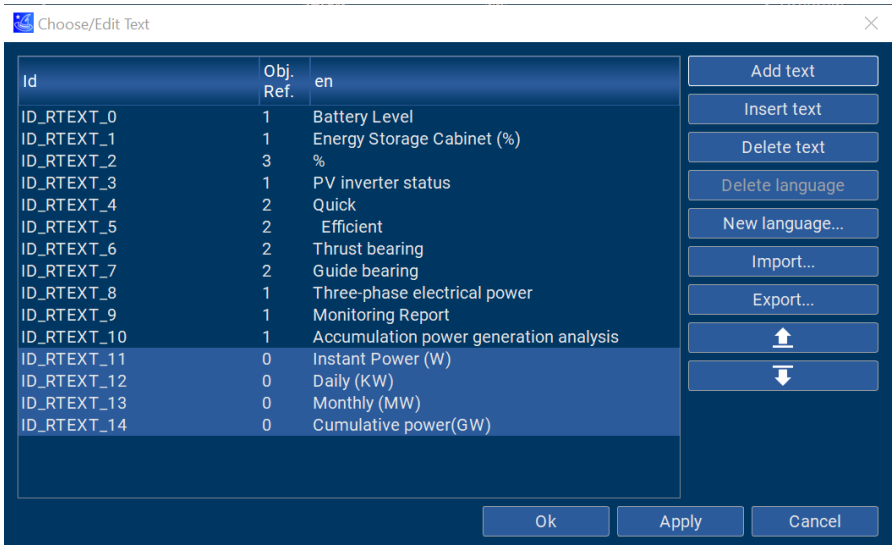


圖 2-134 新增顯示文字(Progbar 說明)

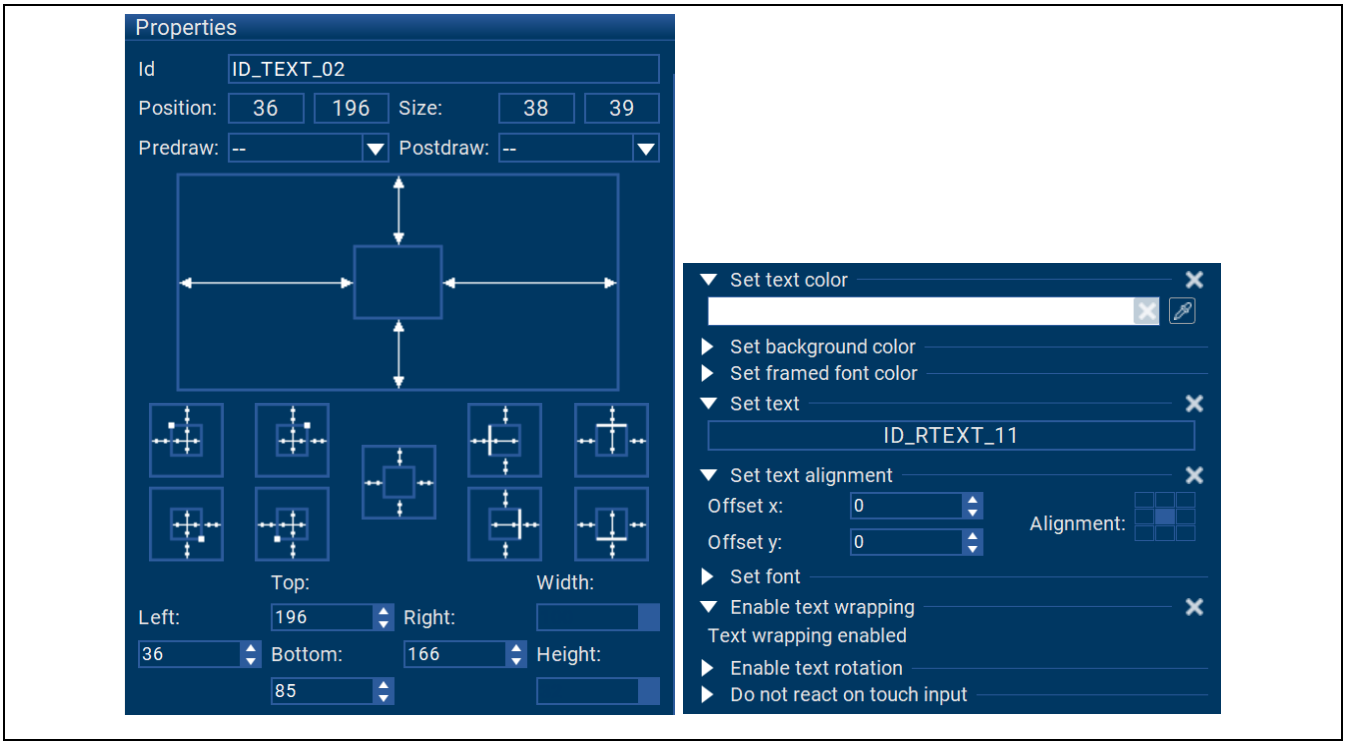


圖 2-135 ID_TEXT_02 屬性設置

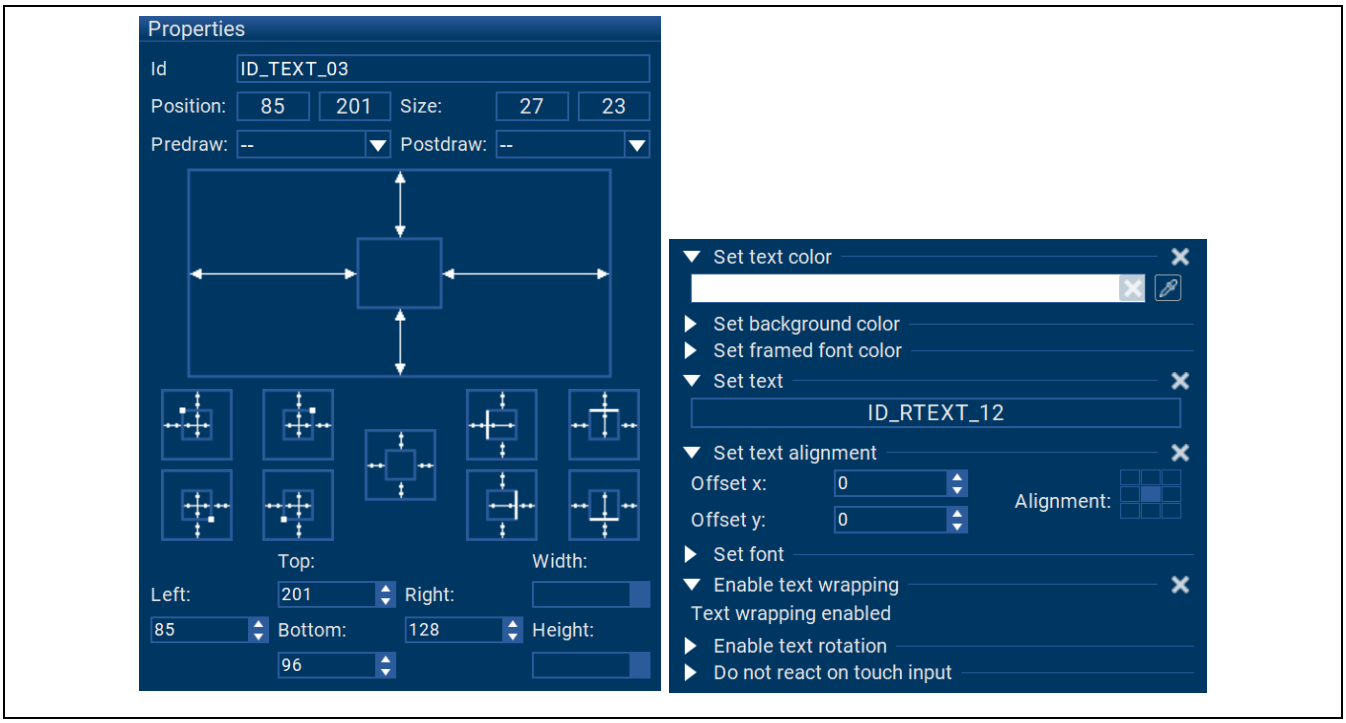


圖 2-136 ID_TEXT_03 屬性設置

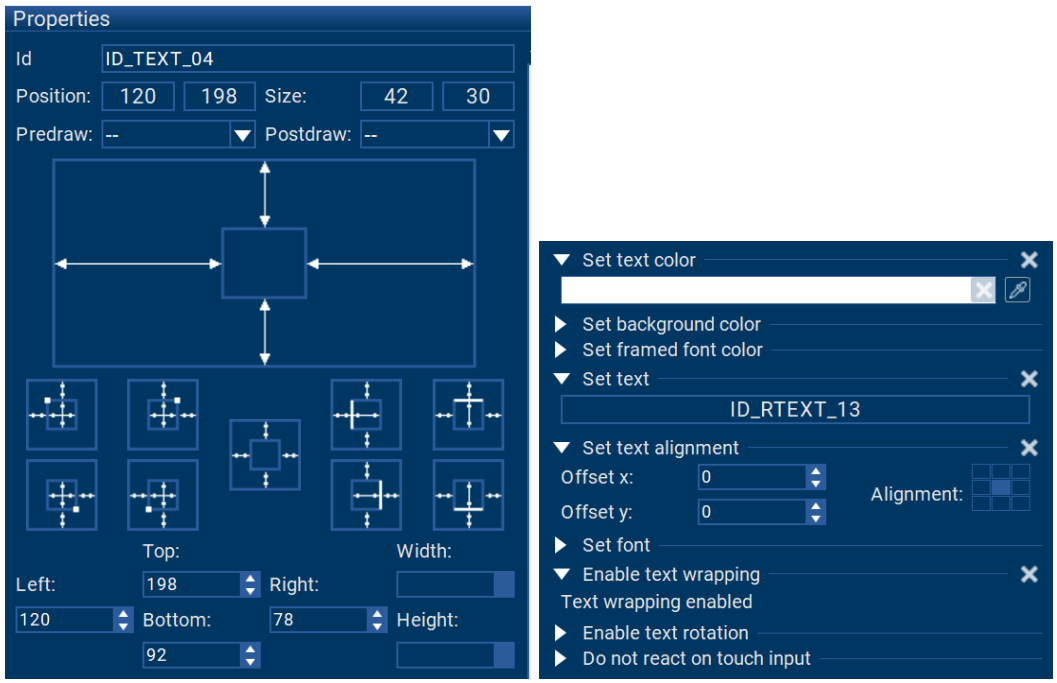


圖 2-137 ID_TEXT_04 屬性設置

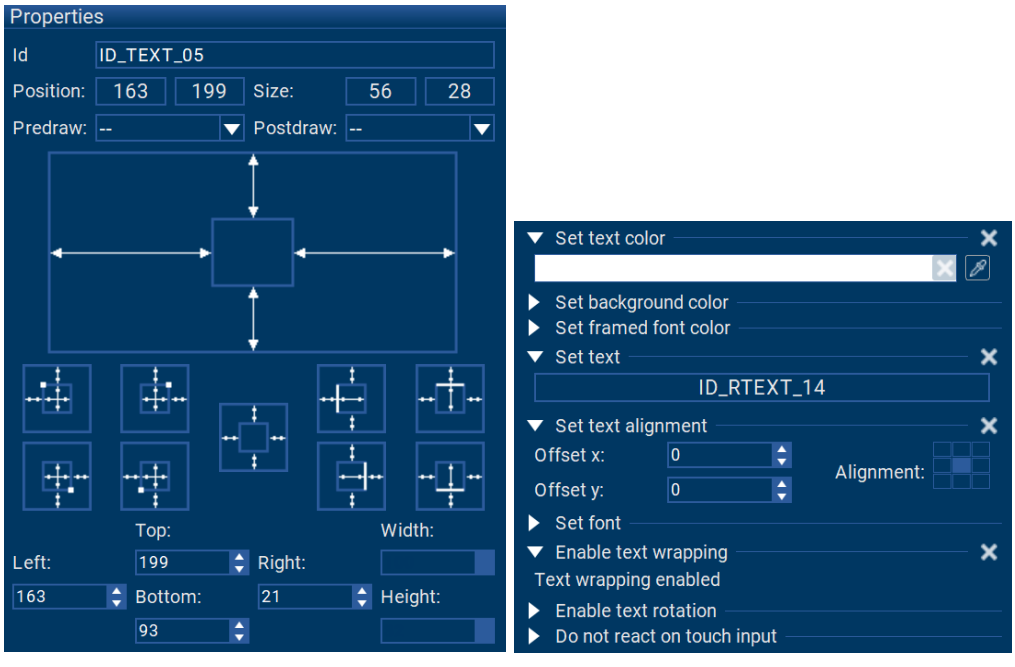


圖 2-138 ID_TEXT_05 屬性設置

6. 新增一個 QR code 物件(ID_QRCODE)，使用者可透過掃描 QR code 導向特定網頁。由 Edit Text 內新增網址(*https://www.nuvoton.com/products/microprocessors/arm9-mpus/*)後，依圖 2-141 設置屬性。

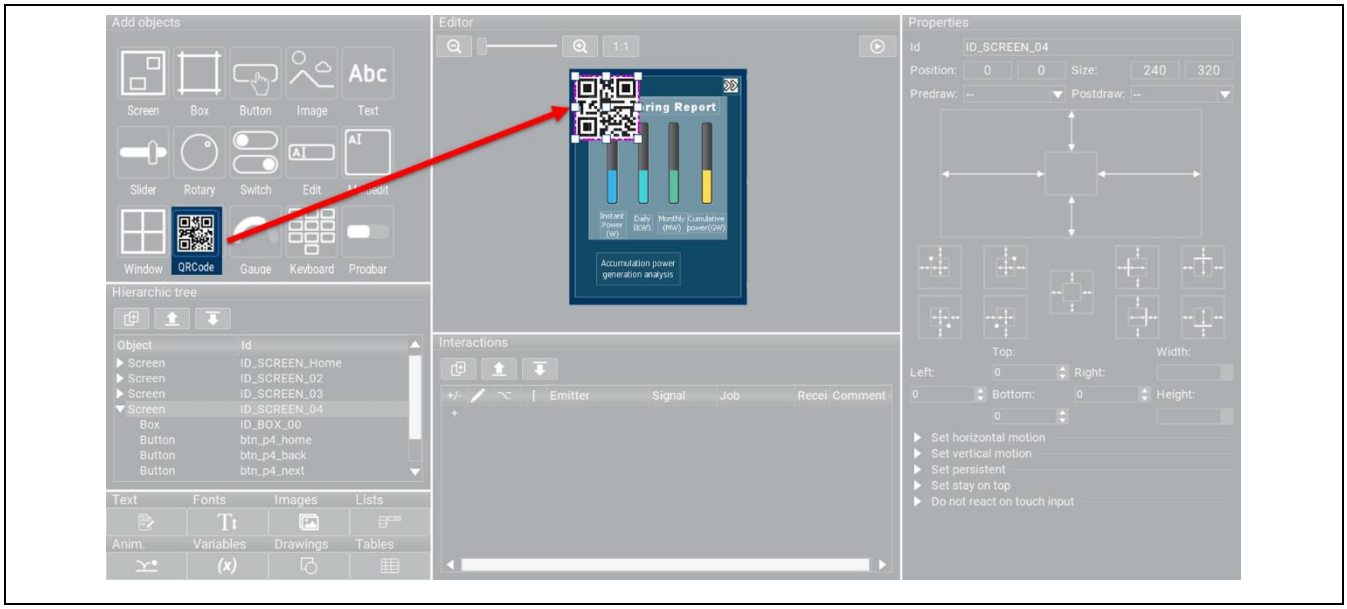


圖 2-139 新增 QR Code 物件

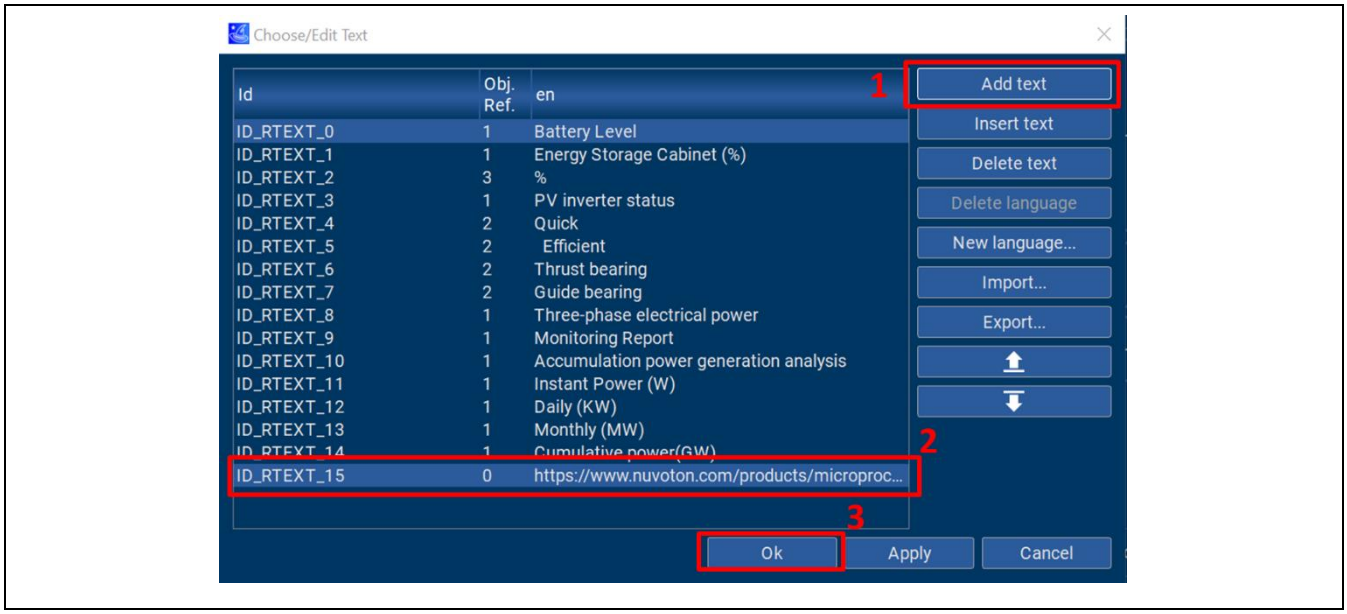


圖 2-140 在 Edit Text 新增網址做為內容

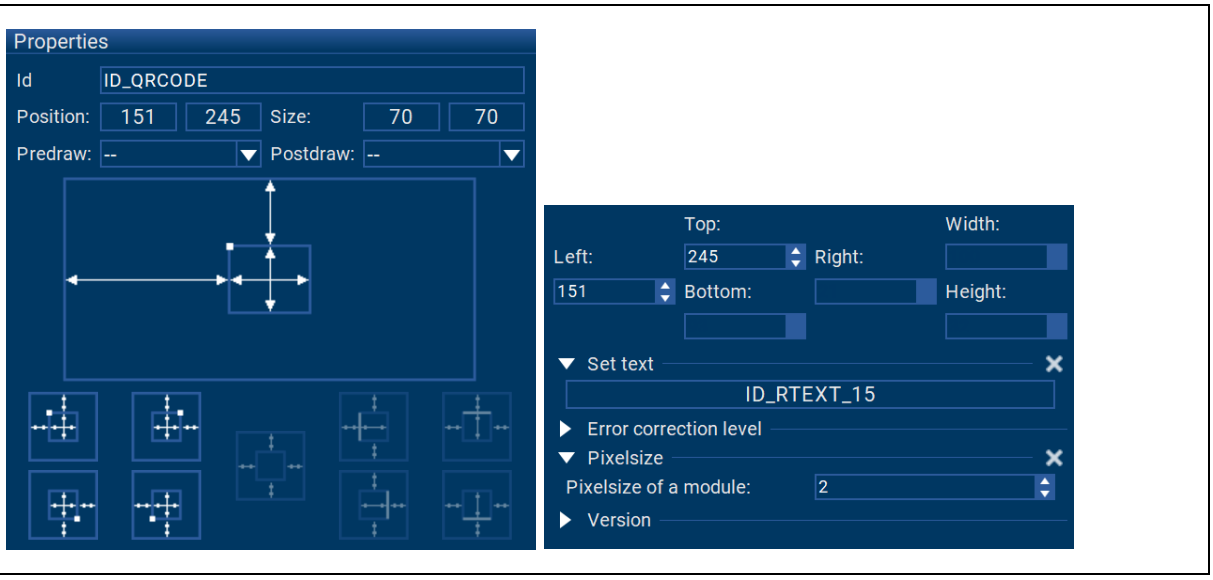


圖 2-141 ID_QRCODE 屬性設置

2.3.4.2 物件互動設置

物件設置完成畫面如圖 2-142，接著要建立物件間的交互作用(interactions)，使操作物件時有相應行為顯示。

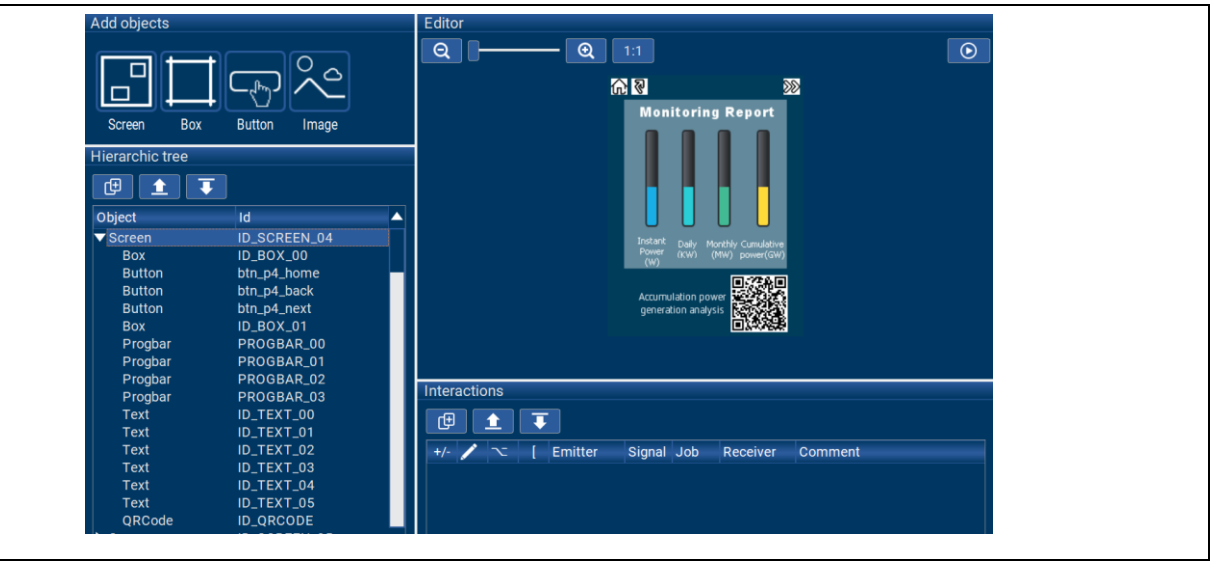


圖 2-142 頁面四設計完成圖

1. 為了使直條圖的數值可以動態呈現，點選左下角 **Anim** 按鈕新增動畫(ANIM_PW_LINE)，接著依圖 2-143~圖 2-151 步驟設置動畫細節。

   1) 透過 Add item 新增動畫後，將其命名為 ANIM_PW_LINE，並設置動畫長度(Period)為 2000(ms)。

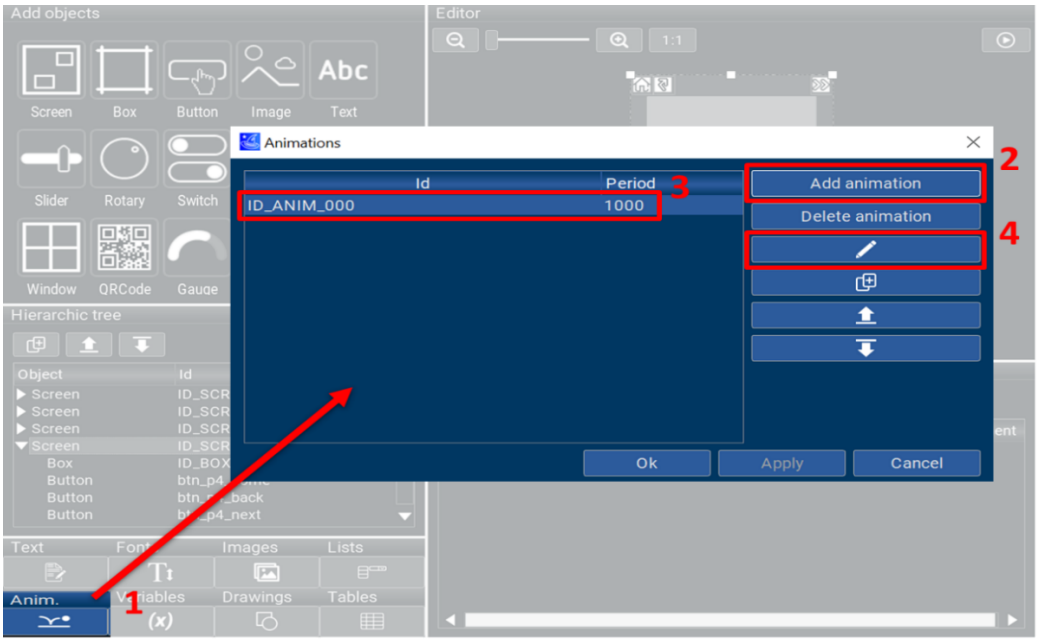


圖 2-143 新增動畫

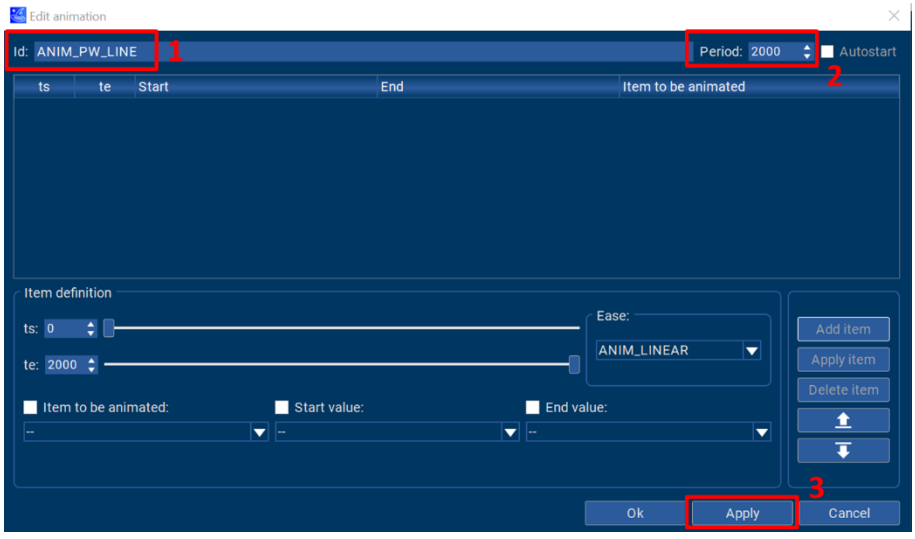


圖 2-144 設置動畫名稱與時間長度

2) 透過 ts 與 te 設置動畫的開始與結束時間。

3) 設置響應動畫的物件 ID 與變化起始值、結束數值與 Ease(ANIM_ACCELDECEL)。

4) 點選 Add item 即可將此動作新增到動畫 ANIM_PW_LINE 中。

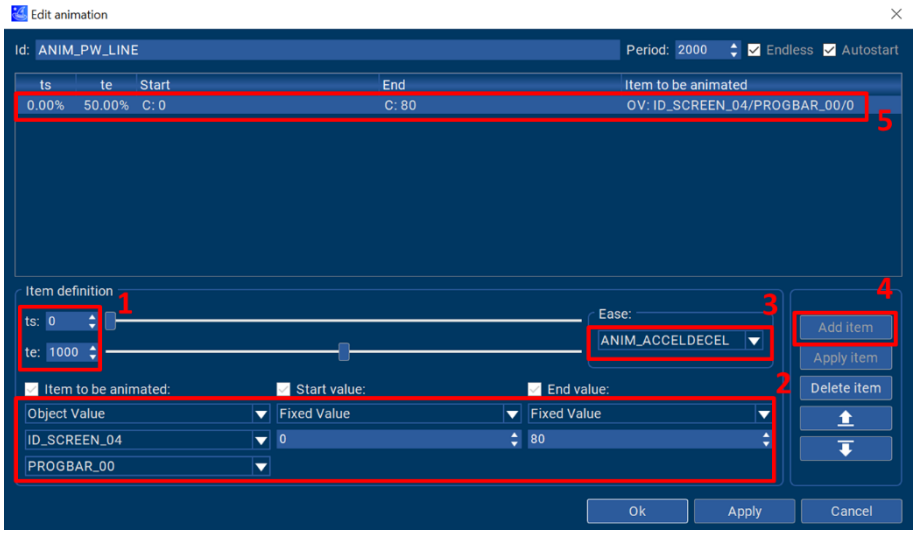


圖 2-145 新增 PROGBAR_00 動畫

5) 同步驟 1)~4)，設置其它 Progbar 物件(PROGBAR_01~ PROGBAR_03)的動畫。

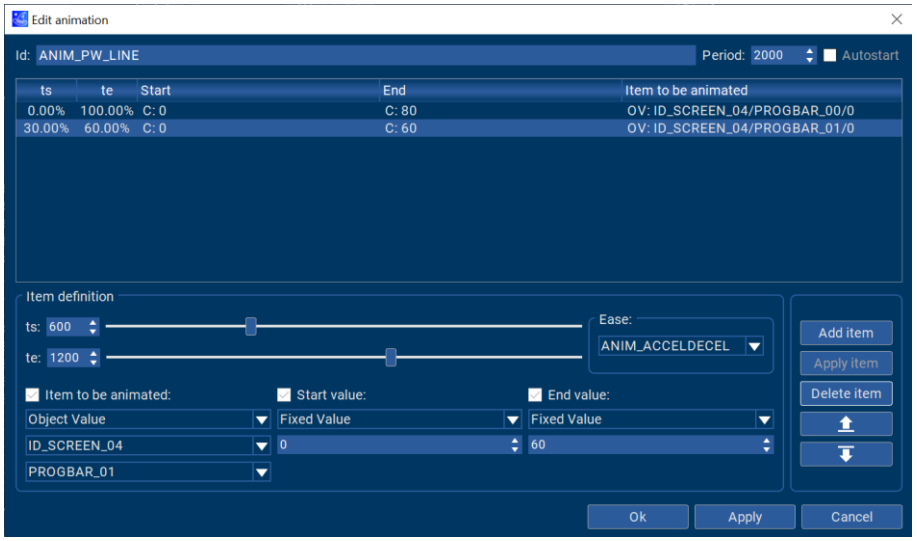


圖 2-146 新增 PROGBAR_01 動畫

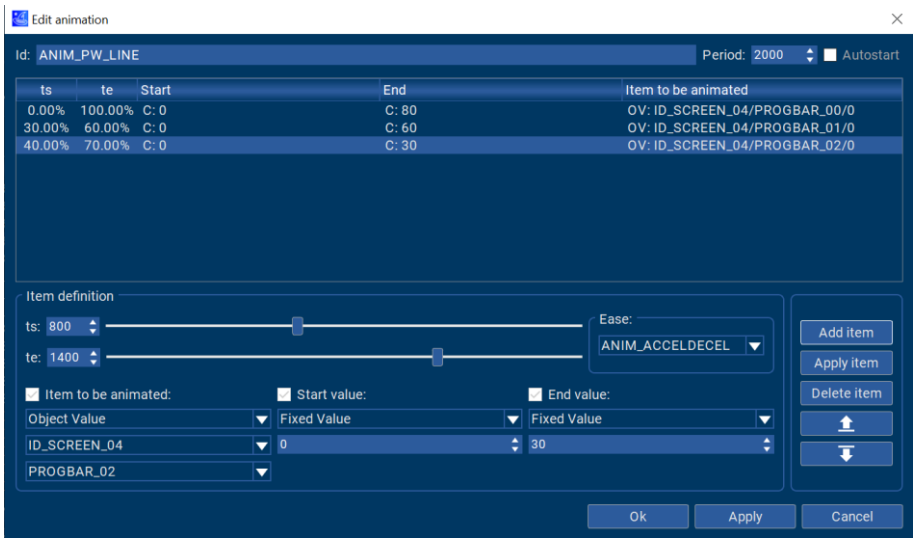


圖 2-147 新增 PROGBAR_02 動畫

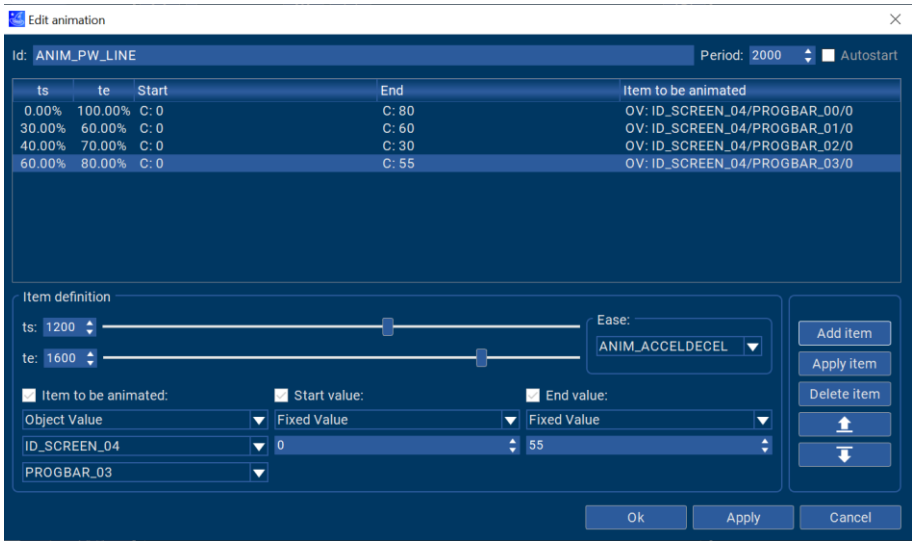


圖 2-148 新增 PROGBAR_03 動畫

6) 所有動作設置完成後，勾選「Autostart」與「Endless」使動畫自動開始並無限循環。

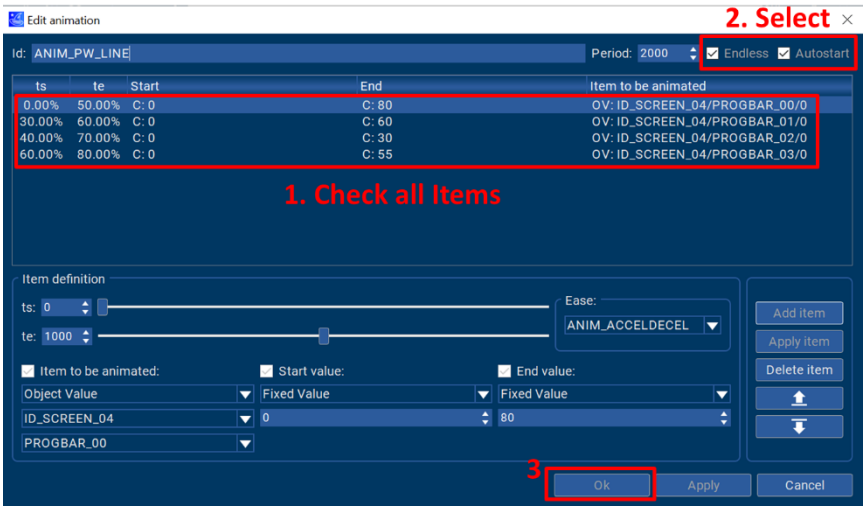


圖 2-149 設置動畫自動開始

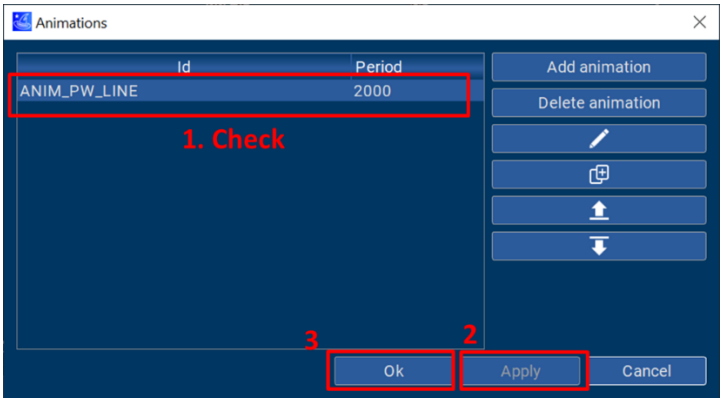


圖 2-150 儲存 ANIM_PW_LINE 動畫設置

7) 在 interactions 欄位新增:Emitter 為 ID_SCREEN_04、Signal 為 INITDIALOG、Job 為 ANIMCREATE 的項目，將 Animation id 設為 ANIM_PW_LINE。

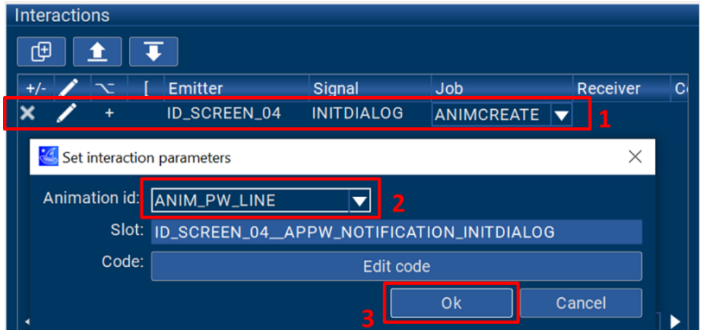


圖 2-151 設置動畫顯示

2. 設置 Button(btn_p4_home、btn_p4_back、btn_p4_next)物件的換頁功能，使點選按鈕可以切回首頁、前一頁及下一頁。

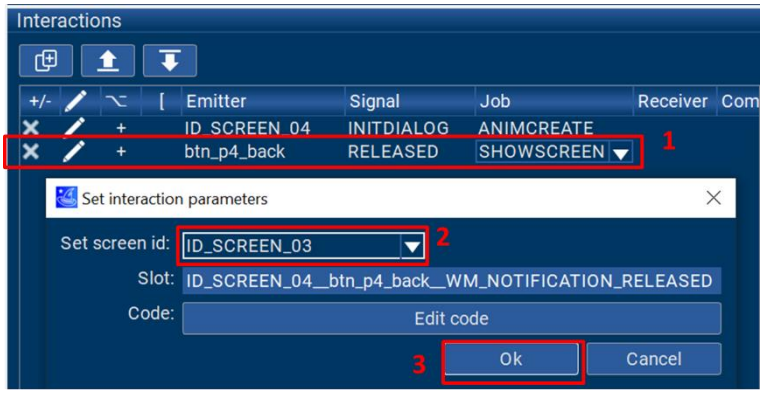


圖 2-152 btn_p4_back 換頁設置

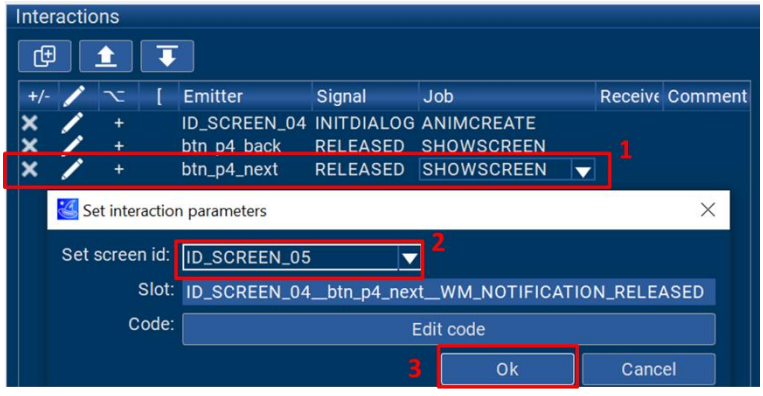


圖 2-153 btn_p4_next 換頁設置

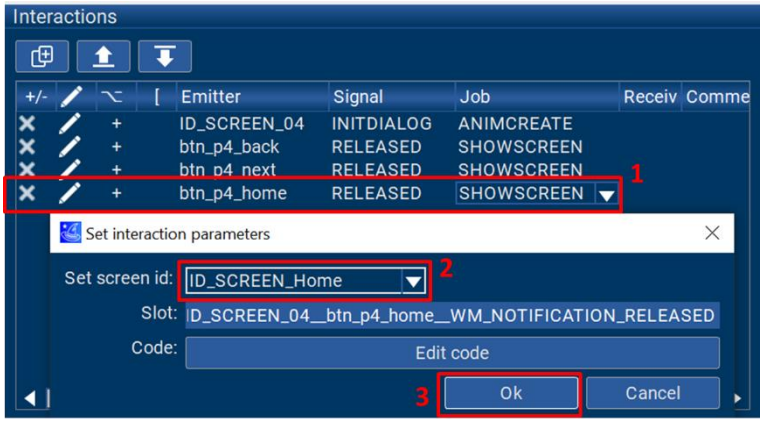
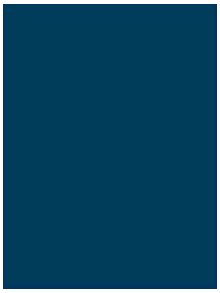
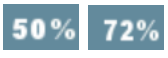


圖 2-154 btn_p4_home 換頁設置

2.3.5 轉換率調整畫面設計(頁面五)

閘道器產品中不乏能源轉換、逆變器等應用，除了可透過簡單的 Slider 設定轉換率調整外，本設計同時設置 Switch 作為安全開關(鎖定調整率)，以降低人員操作不慎、誤觸時導致的損壞，此設計所用物件如表 2-7 所示。

New Energy Gateway Description	Application GUI	Widget Type
背景圖 240x320		Box
拉霸背景圖 150x254		Box
標題文字		Text
百分比顯示		

轉換幅度調整拉霸		Slider
鎖定開關		Switch
檔案上傳按鈕		Button
回主頁按鈕		
上一頁按鈕		

表 2-7 轉換率調整設計物件

2.3.5.1 GUI 物件配置

1. 點選 ID_SCREEN_05 進行頁面五的設計，透過 Add objects 新增第二個 Box 物件 (ID_BOX_01)作為拉霸背景，並調整齊屬性。

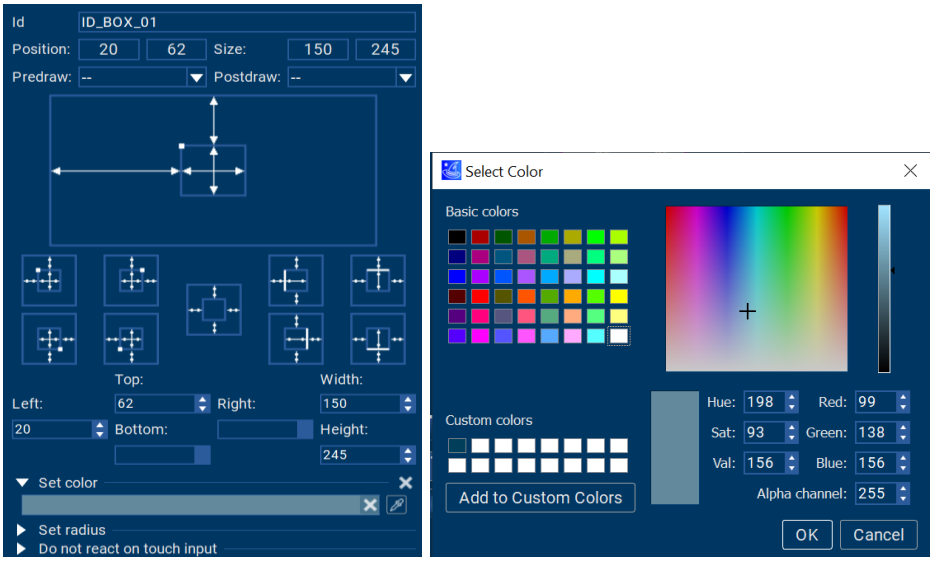


圖 2-155 ID_BOX_01 屬性設置

2. 新增五個 Text 物件，分別作為標題(ID_TEXT_00，顯示文字為 Amplitude adjustment)、兩個拉霸數值(ID_TEXT_01、ID_TEXT_03)與兩個百分比符號(ID_TEXT_02、ID_TEXT_04)，依圖 2-157~圖 2-161 設置屬性。

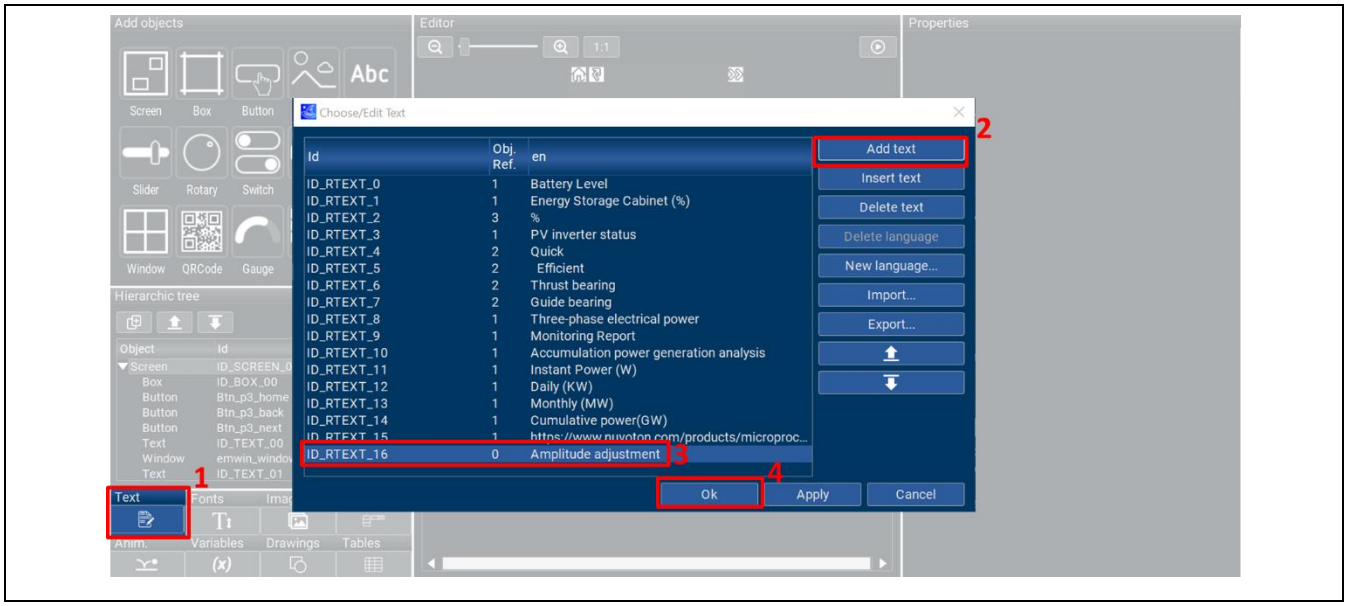


圖 2-156 新增 ID_TEXT_00 顯示文字

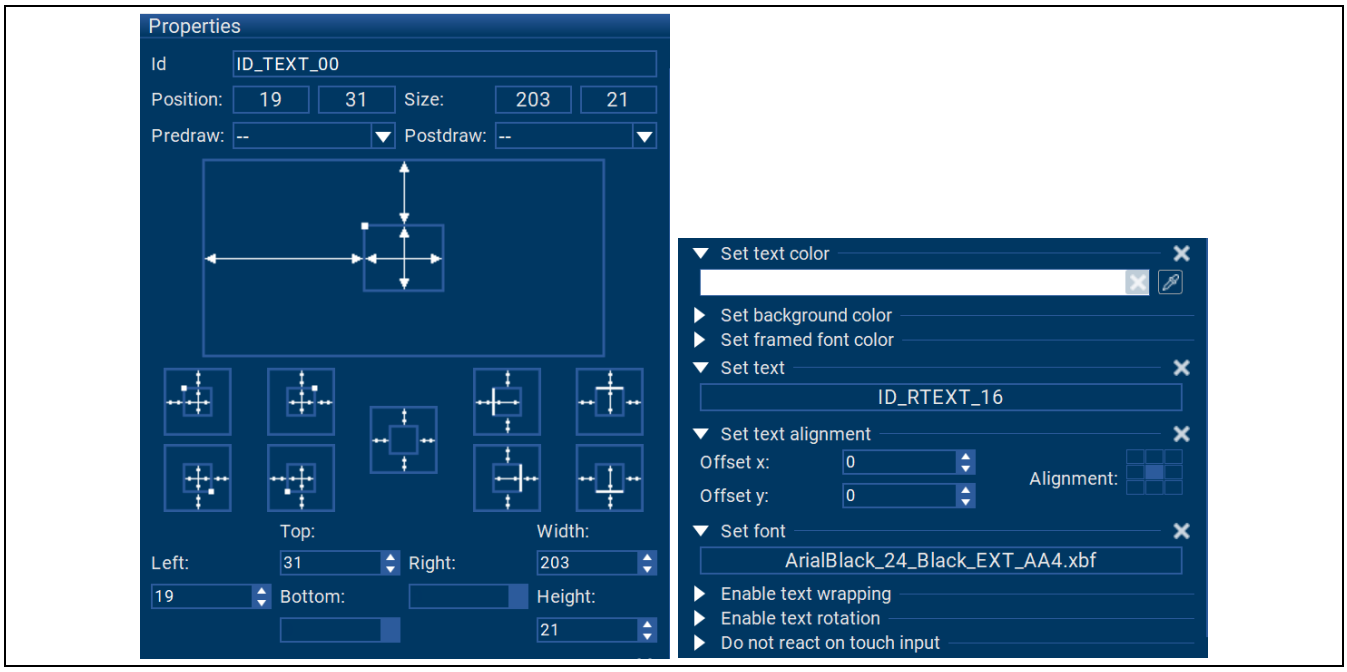


圖 2-157 ID_TEXT_00 屬性設置

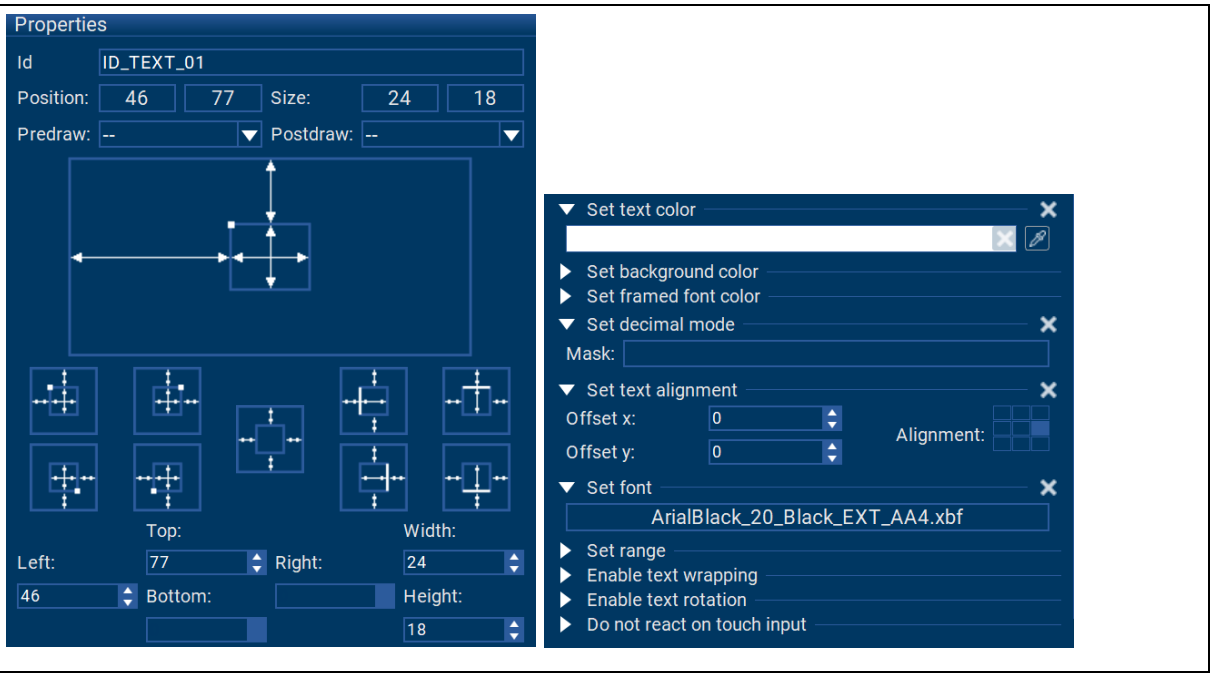


圖 2-158 ID_TEXT_01 屬性設置

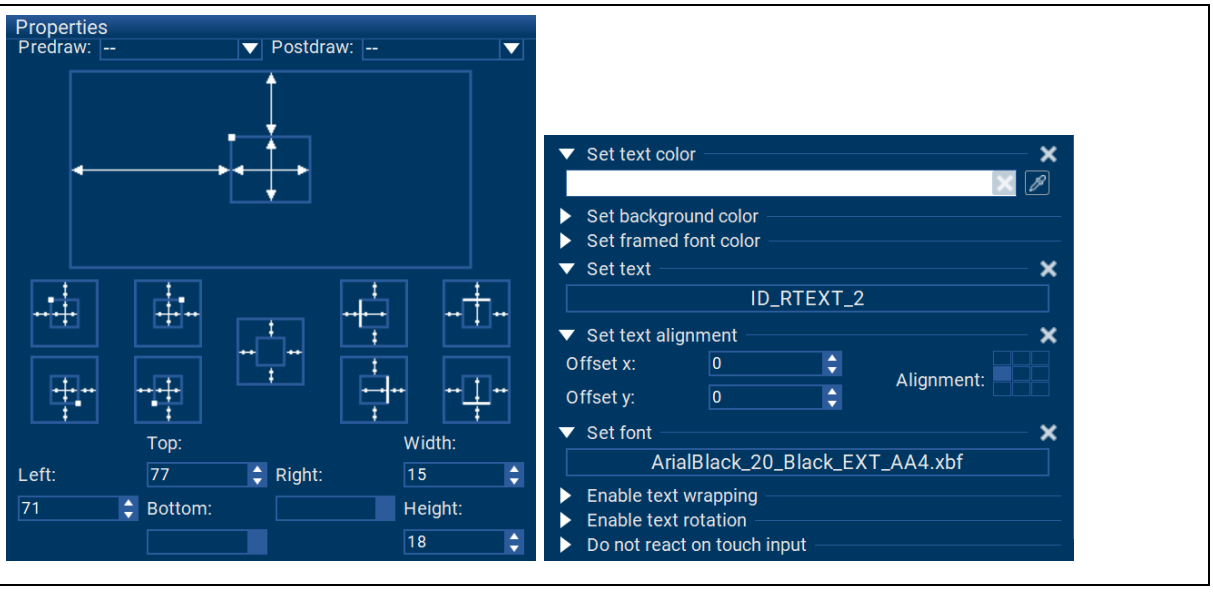


圖 2-159 ID_TEXT_02 屬性設置

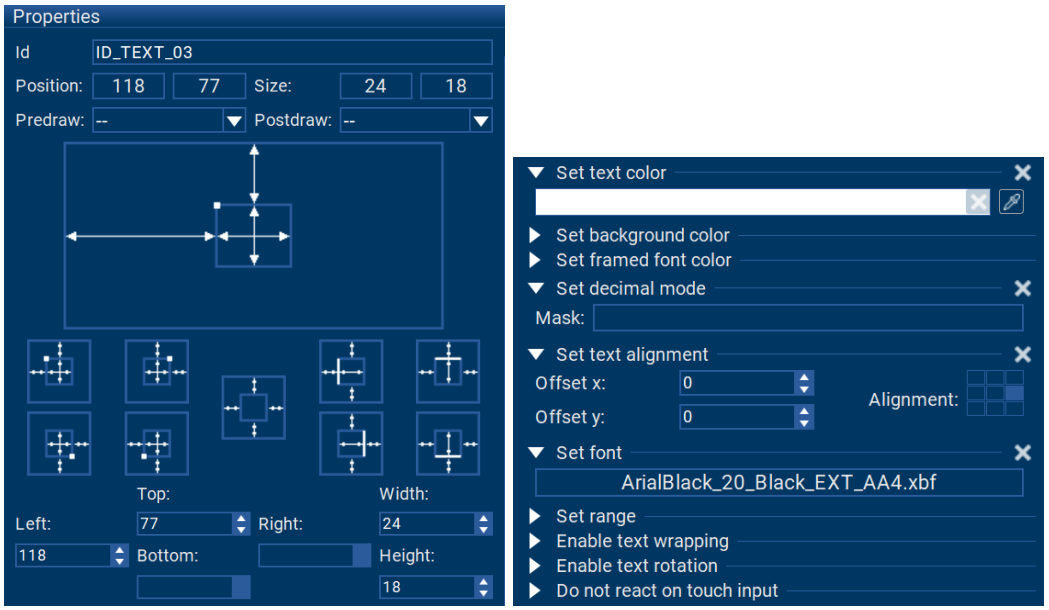


圖 2-160 ID_TEXT_03 屬性設置

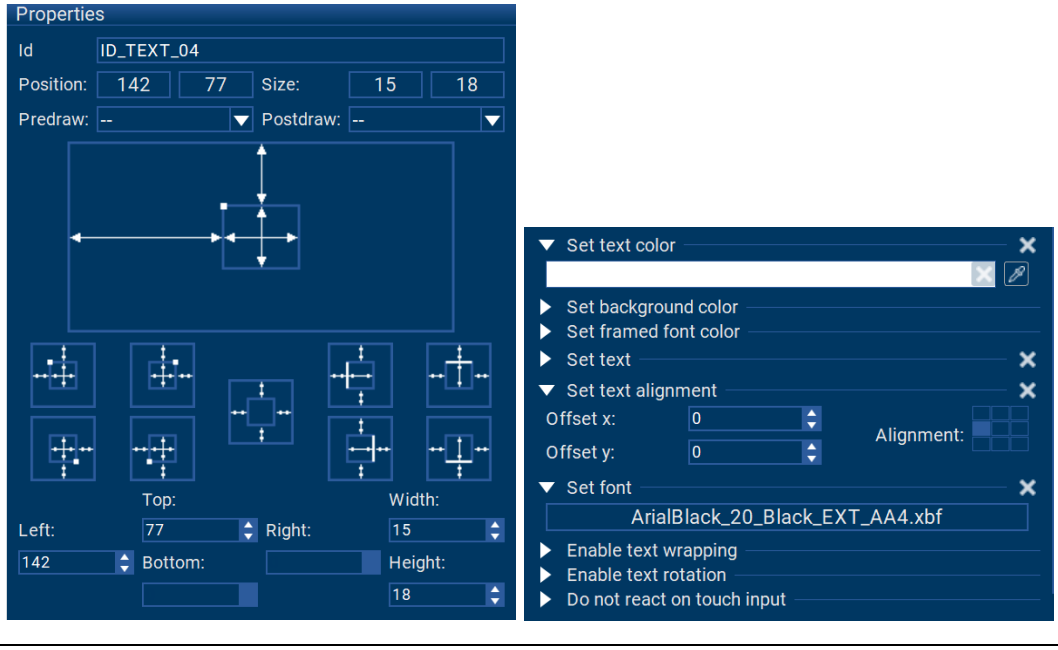


圖 2-161 ID_TEXT_04 屬性設置

3. 新增兩個 slider 物件(SLIDER_T1、SLIDER_T2)做數值調整，在屬性欄開啟 Vertical mode 與 Invert direction 功能(點擊箭頭符號開啟)，使 slider 外觀垂直、數值由下往上遞增，並以圖片 slider_blue_down_16x30.png、DARK_Slider_Shaft_V_Top_Gray_16x30.png、DARK_Slider_Thumb_Up_22x22.png、DARK_Slider_Thumb_Down_22x22.png 分別做為 bitmap 中的 Shaft bottom、Shaft top、Thumb unpressed 與 Thumb pressed。

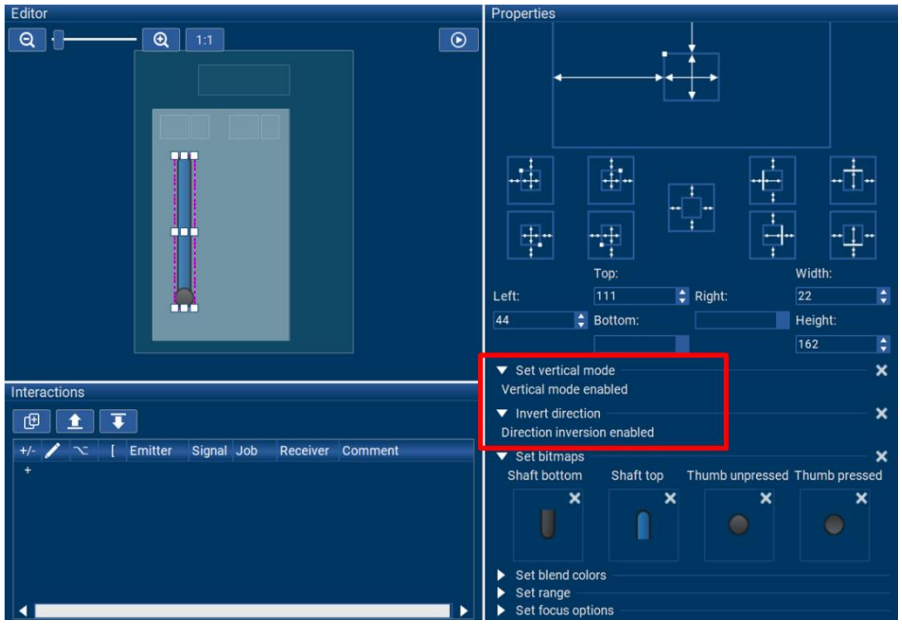


圖 2-162 設置 Vertical mode 與 Invert direction

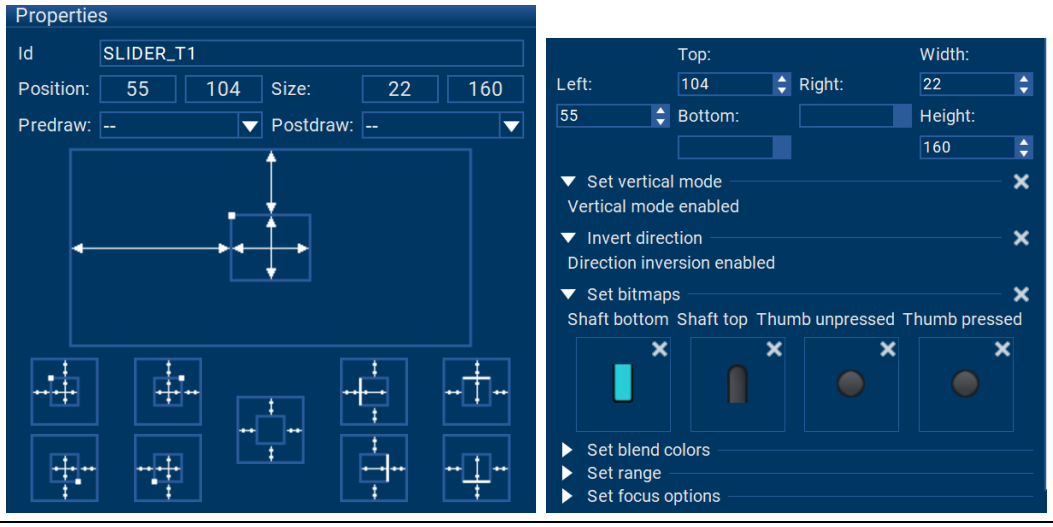


圖 2-163 SLIDER_T1 屬性設置

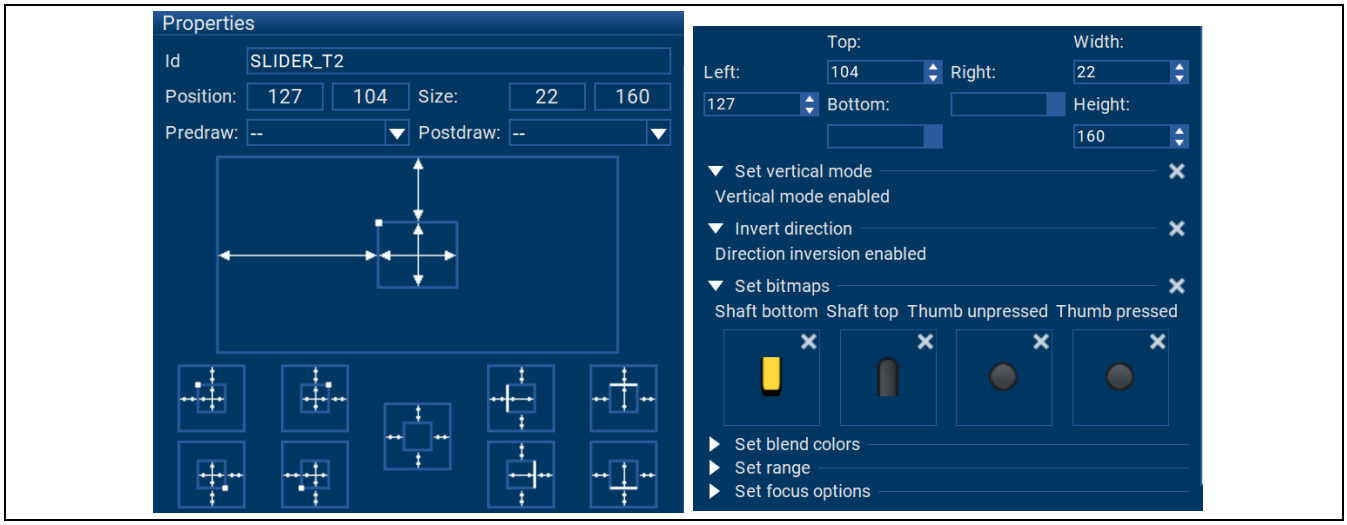


圖 2-164 SLIDER_T2 屬性設置

4. 為了設置 slider 調整數值的開關，新增一個 Switch 物件(SWITCH_locker)，以圖片 Switch_Button_Pressed_68x22.png 與 Switch_Button_Unpressed_68x22.png 作為外觀，並依圖 2-165 ~ 圖 2-166 設置屬性。

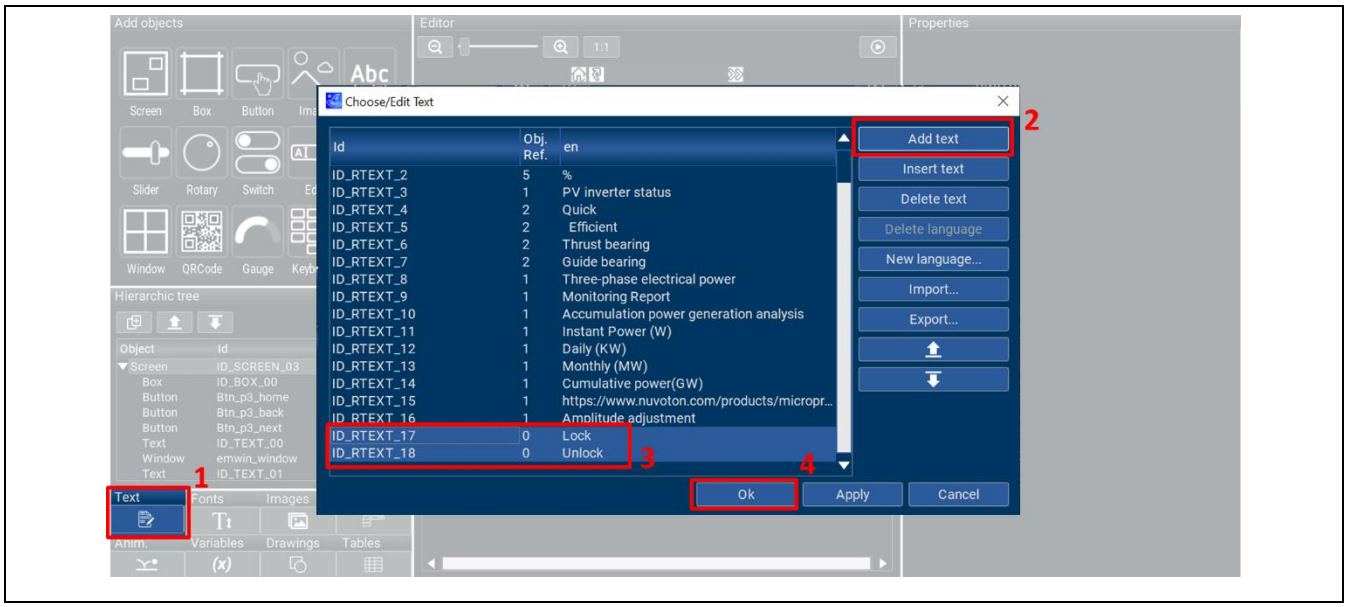


圖 2-165 新增 SWITCH_locker 顯示文字(Lock、Unlock)

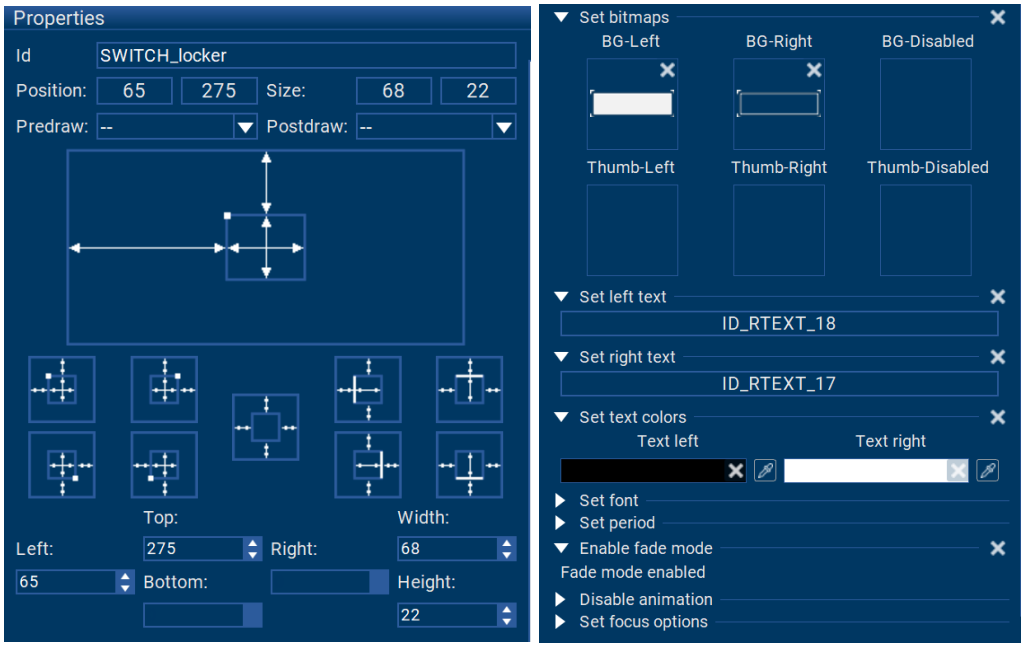


圖 2-166 SWITCH_locker 屬性設置

5. 新增三個 button 物件做換頁(btn_p5_home、btn_p5_back)與資料上傳(btn_upload)功能，屬性如圖 2-167~圖 2-170，並以圖片 home_20x20_new.png、arrow2_20x20_new.png、Squere_Button_Unpressed_40x40.png 與 Squere_Button_Pressed_40x40.png 設置外觀。

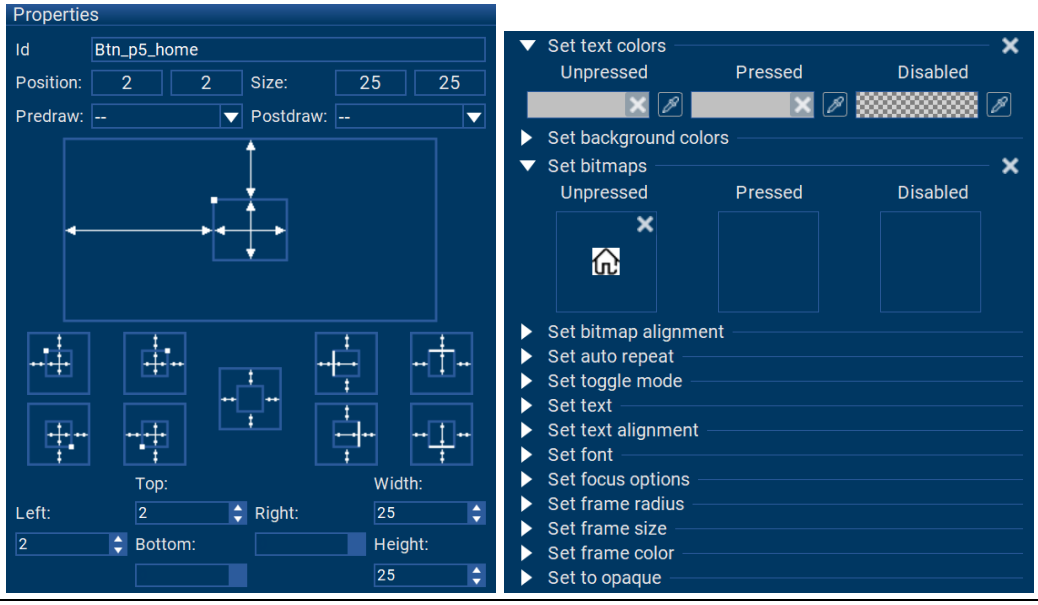


圖 2-167 btn_p5_home 屬性設置

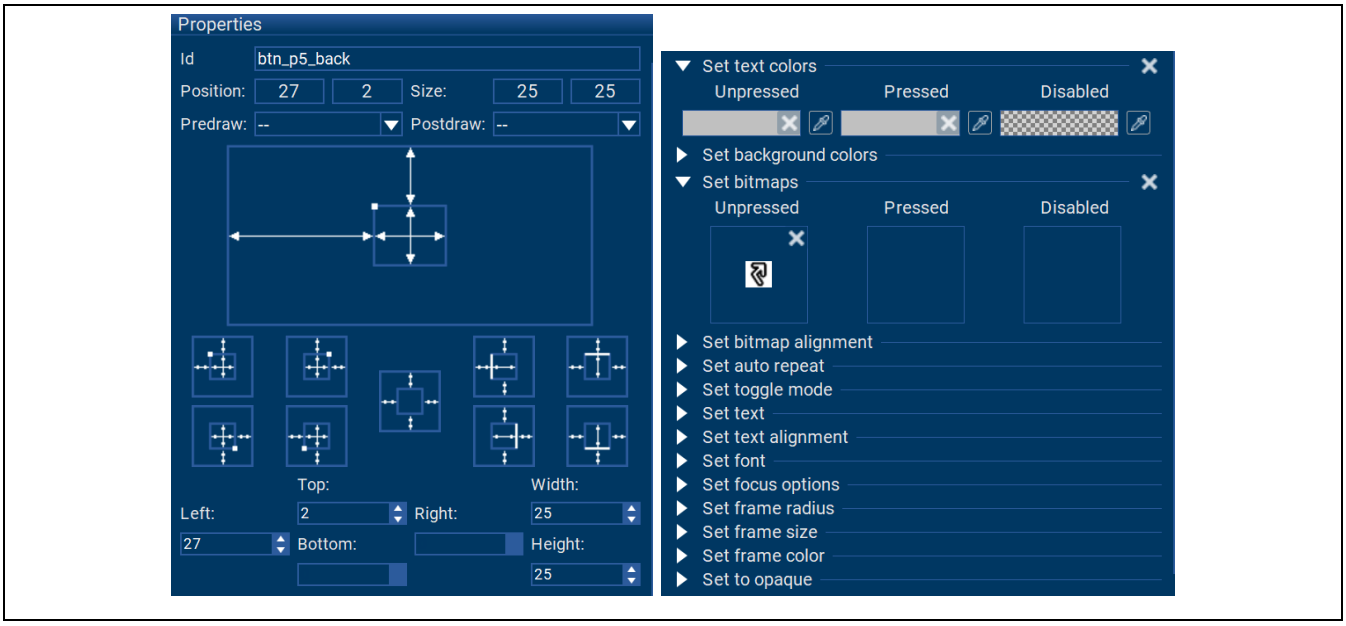


圖 2-168 btn_p5_back 屬性設置

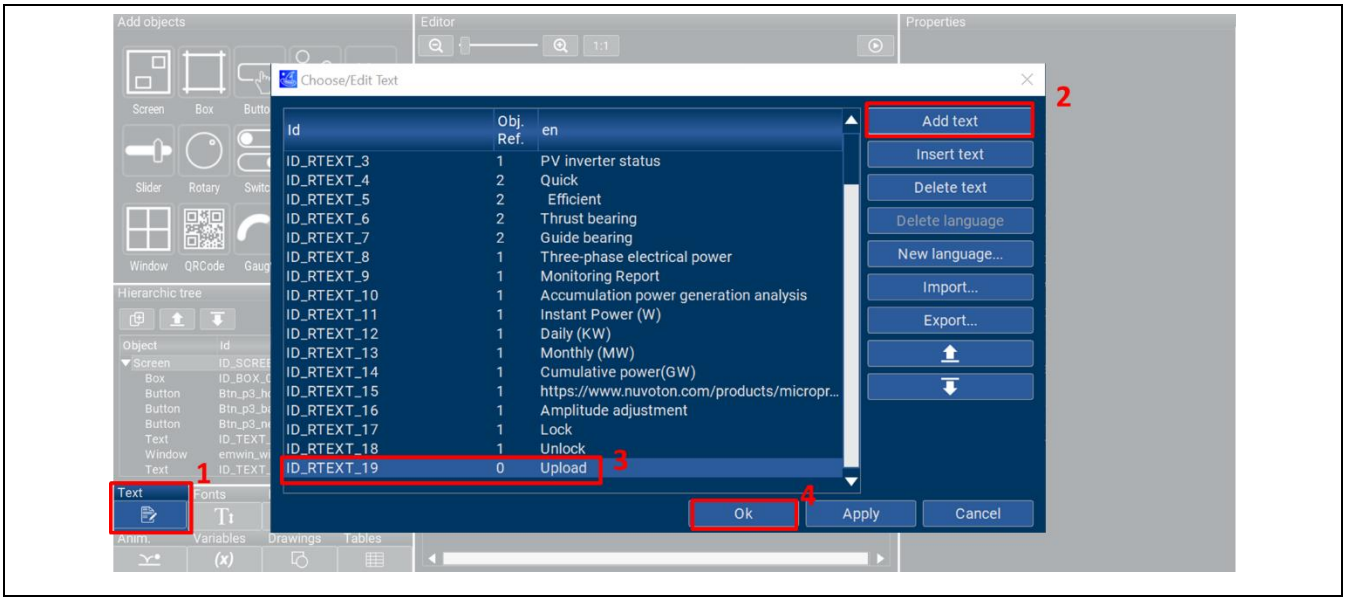


圖 2-169 新增 btn_upload 顯示文字(Upload)

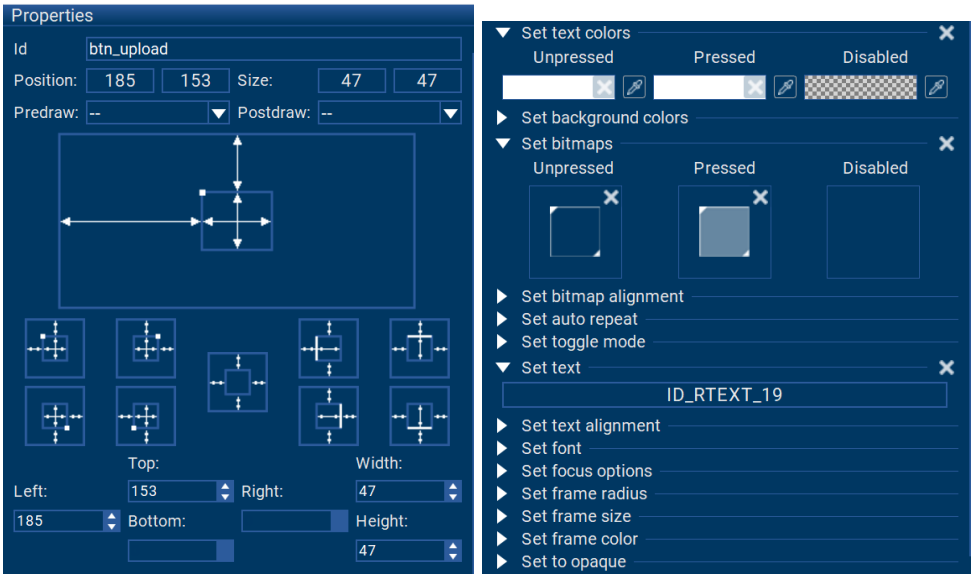


圖 2-170 btn_upload 屬性設置

2.3.5.2 物件互動設置

物件設置完成畫面如圖 2-171 所示，接著要建立物件間的交互作用(interactions)，使操作物件時有相應行為顯示。

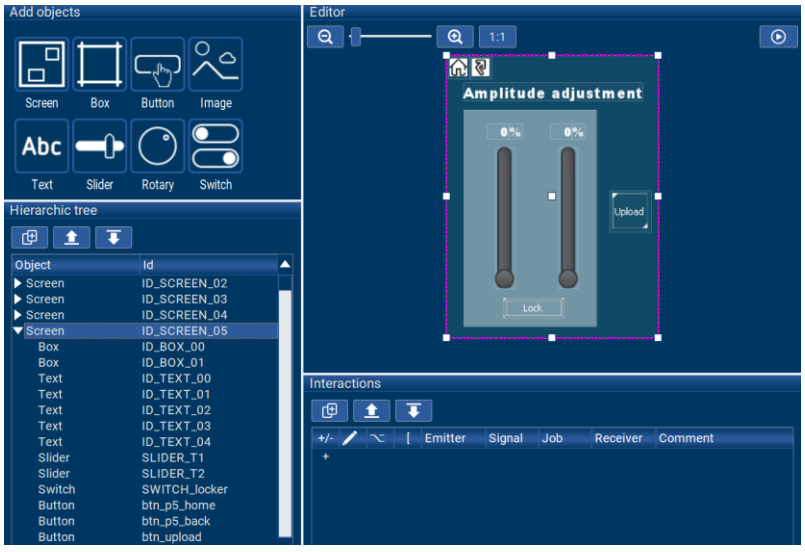


圖 2-171 頁面五設計完成圖

1. 設置 Slider 與 Text 之間的連動，使 TEXT 物件顯示 Slider 調整後的數值，功能設置圖 2-172~圖 2-173。圖 2-174、圖 2-176 設置開啟 switch 後關閉 slider 物件的功能；圖 2-175、圖 2-177 則設置關閉 switch 後重新使能 slider 物件的功能。

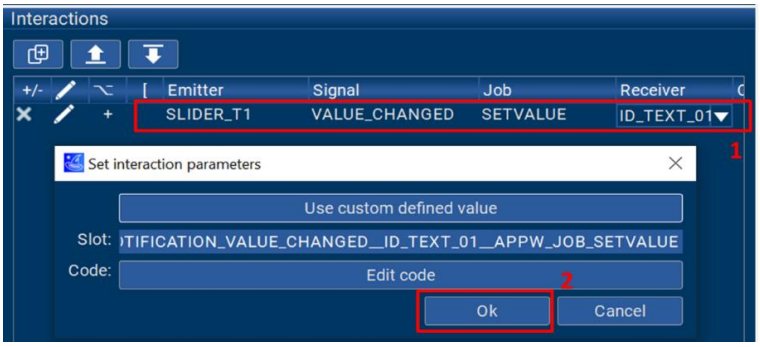


圖 2-172 設置 SLIDER_T1 與 ID_TEXT_01 互動

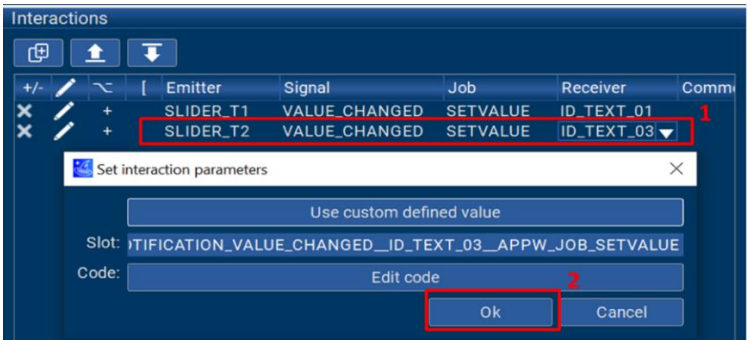


圖 2-173 設置 SLIDER_T2 與 ID_TEXT_03 互動

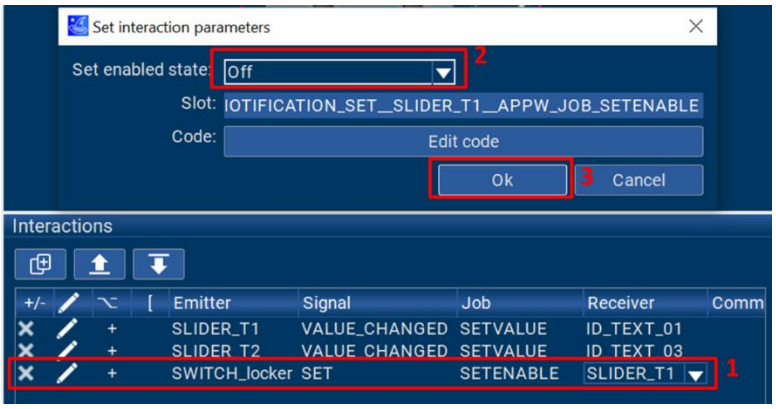


圖 2-174 設置 SLIDER_T1 與 Switch_locker 互動 1

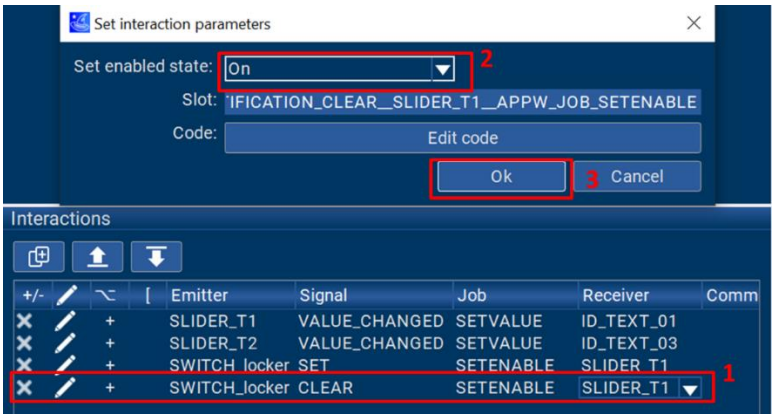


圖 2-175 設置 SLIDER_T1 與 Switch_locker 互動 2

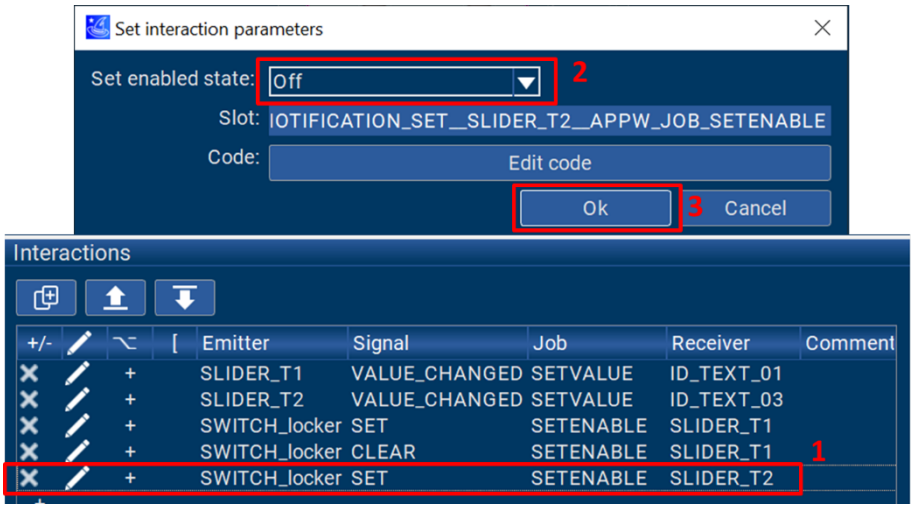


圖 2-176 設置 SLIDER_T2 與 Switch_locker 互動 1

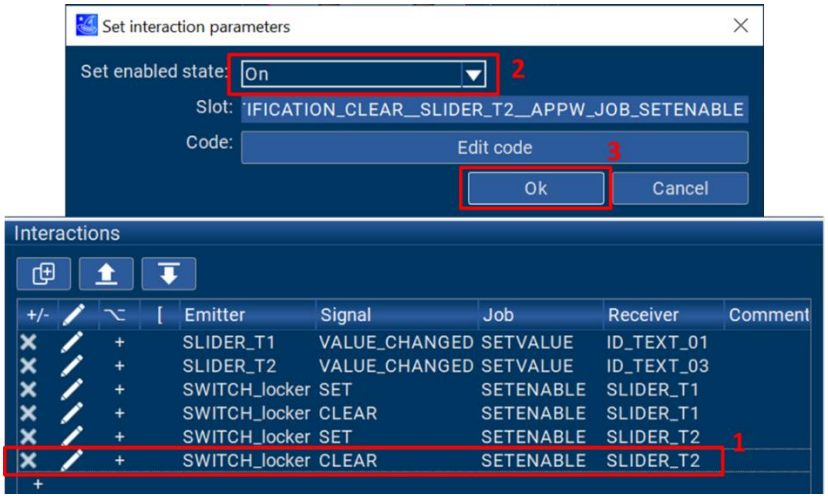


圖 2-177 設置 SLIDER_T2 與 Switch_locker 互動 2

2. 設置 Button 物件(btn_p5_home、btn_p5_back)換頁功能，透過按鈕切換顯示頁面。

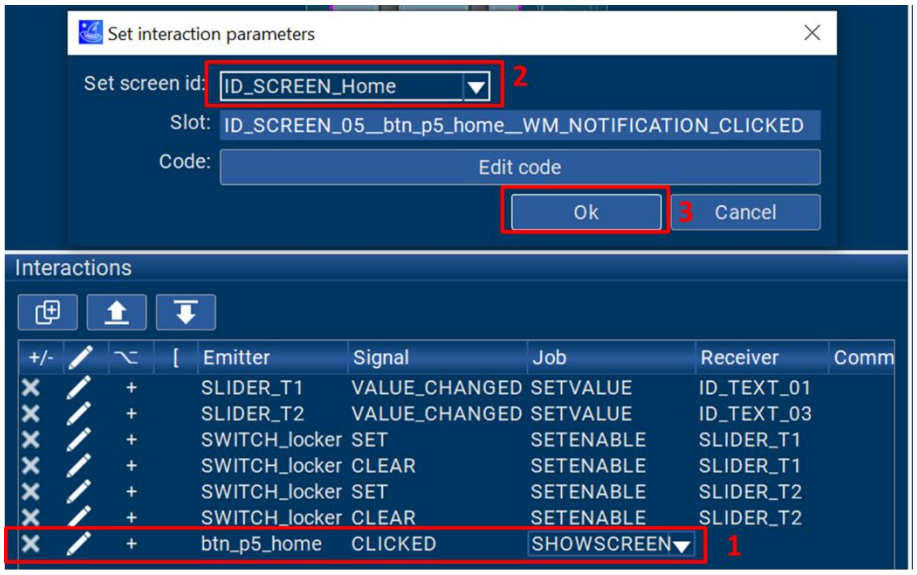


圖 2-178 btn_p5_home 換頁設置

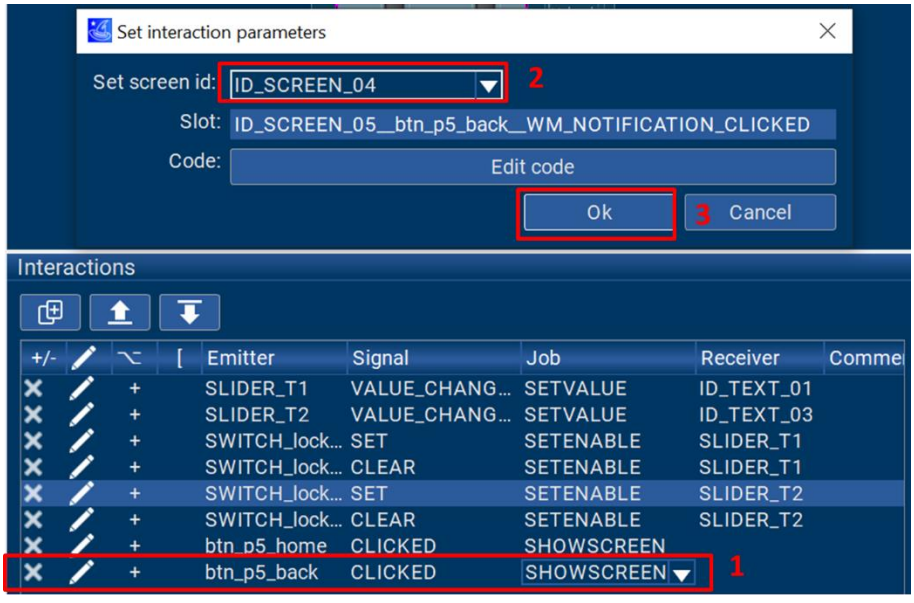


圖 2-179 btn_p5_back 換頁設置

2.3.6 導出 AppWizard 專案

透過 AppWizard 編輯器上方工具列將 GUI 設計儲存並導出專案(方法同前所述，如圖 2-115)，生成資料夾如圖 2-180 所示。

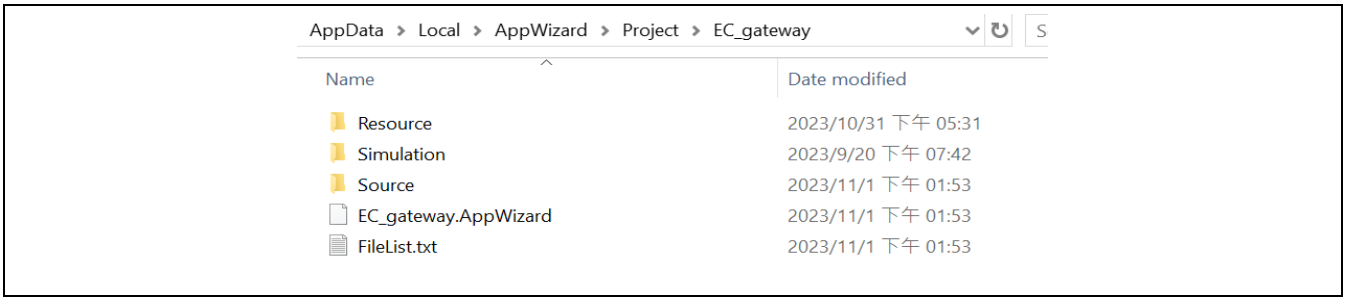


圖 2-180 AppWizard 專案生成資料夾

2.4 將應用程式佈署至開發板

2.4.1 透過 EmWin Package 生成燒錄檔

EmWin package 含所有 emWin AppWizard 需要的 library 與資源，將設計完成的 GUI 專案資料夾放入 package 內並執行 Makefile 檔案，即可生成程式 bin 檔，詳細操作流程請參考 package 內的 README.txt，而將導出的 AppWizard 專案置入 emWin Package 的方法如下所述。

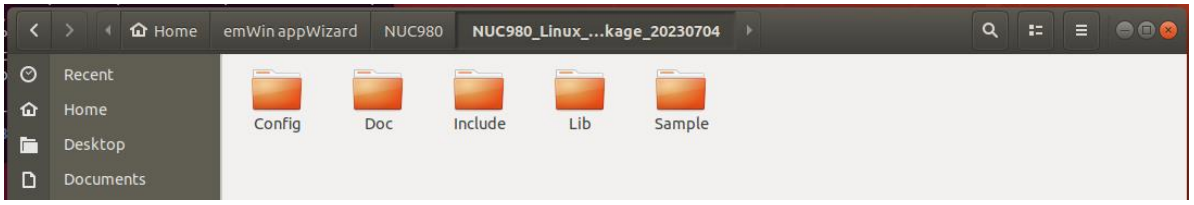


圖 2-181 emWin Package 內容(解壓縮後)

1. 將 appWizard 專案生成的資料夾”**Resource**”與”**Source**”複製到 Linux 環境中的 emWin package 路徑下: */Sample/SimpleDemoAppwizard/Application*

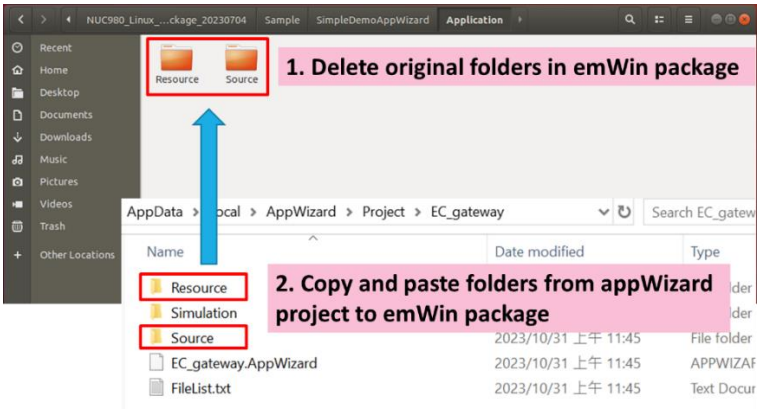


圖 2-182 將 AppWizard 生成檔案放入 Package

2. 回到上一層路徑(/Sample/SimpleDemoAppwizard)，透過滑鼠右鍵點選”Open in Terminal”開啟 console 視窗輸入指令”make”編譯檔案，如圖 2-183~圖 2-185 所示。

```
$ make
```

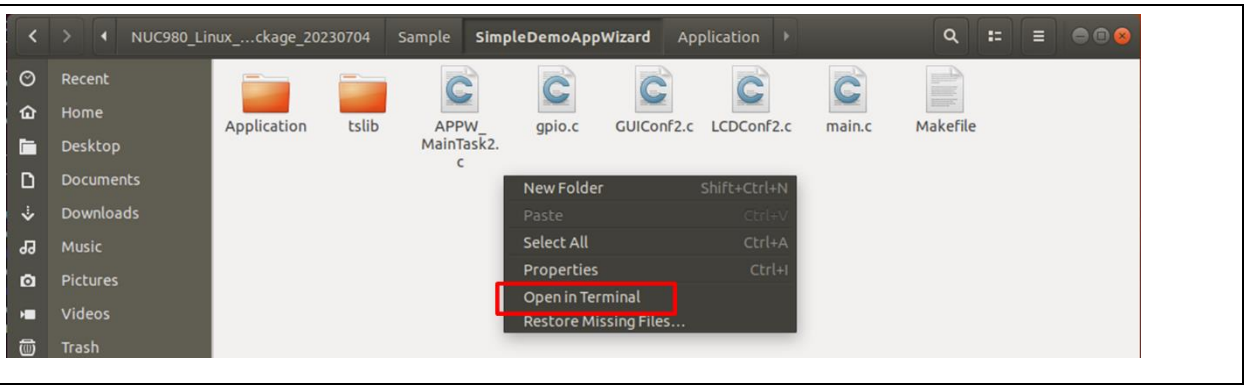


圖 2-183 在資料夾內開啟 console

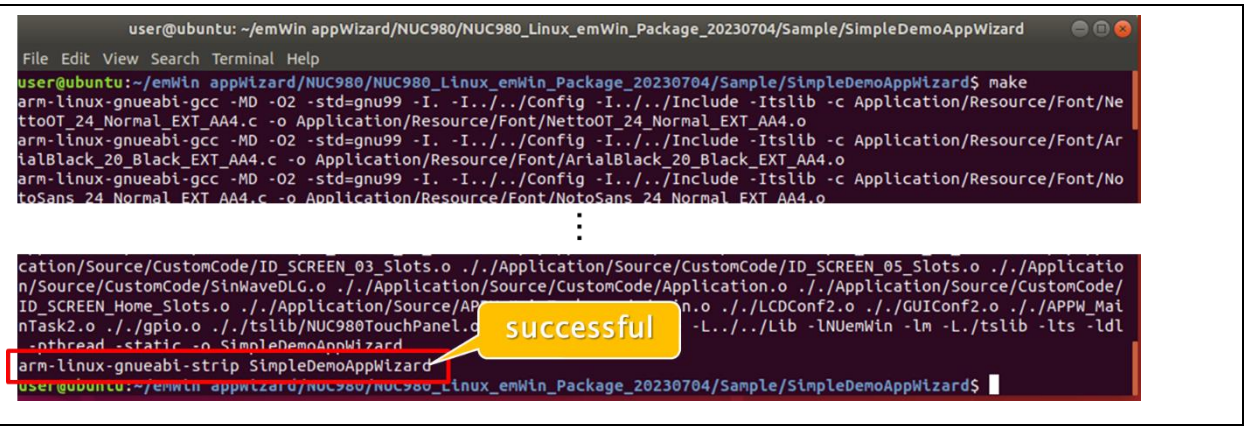


圖 2-184 檔案編譯成功畫面

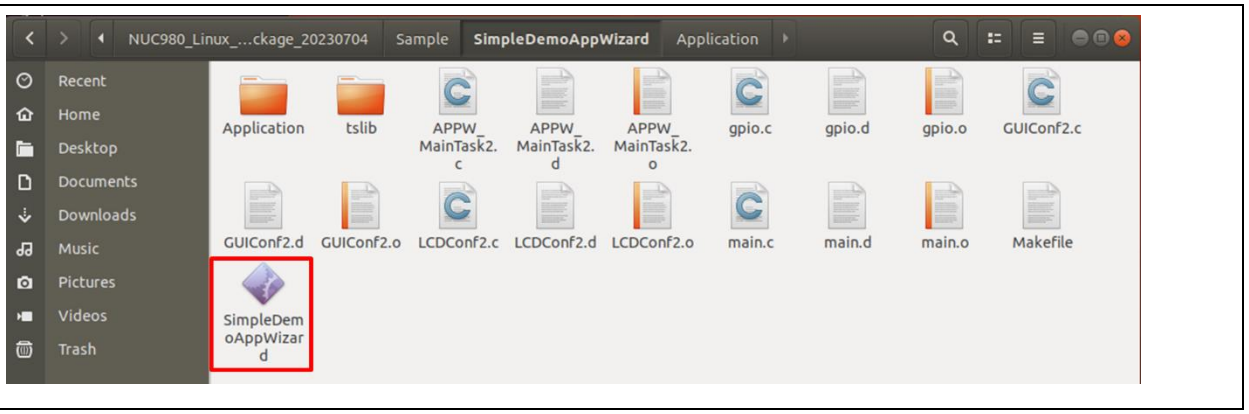


圖 2-185 編譯生成的 bin 檔

3. 將生成的 bin 檔複製到 SD 卡內備用。(備註: SD 卡需為 FAT 格式)

2.4.2 在 NuMaker-IIoT-NUC980G2D 執行 GUI 程式

透過 Type-C USB 連接 NuMaker-IIoT-NUC980G2D 的 VCOM 接口上電，在 Console 中以”root”登入後，插入 SD 卡，如圖 2-186~圖 2-188 所示。

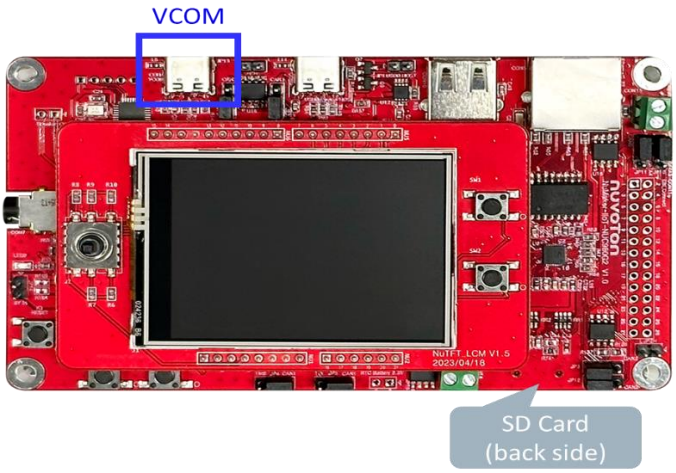


圖 2-186 NuMaker-IIoT-NUC980G2D VCOM 與 SD 卡插入位置

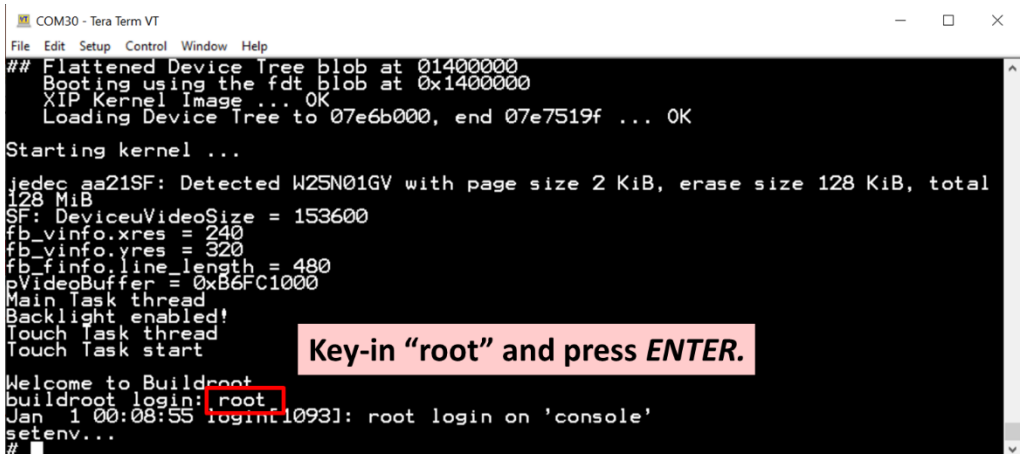


圖 2-187 以 root 登入系統

圖 2-188 SD 卡讀取成功畫面

SD 卡插入後，在 Console 內鍵入指令將 SD 卡內容掛載到 NuMaker-IIoT-NUC980G2D：

```
$ mount -t vfat -o noatime,shortname=mixed,utf8 /dev/mmcblk0p1 /mnt/
$ cd /mnt
```

接著設置 GUI 所需的環境變數：

```
$ cp /mnt/ts.conf /etc
$ export TSLIB_CONFFILE=/etc/ts.conf
$ export TSLIB_TSDEVICE=/dev/input/event0
$ /usr/bin/ts_calibrate  //ONLY ONCE
```

執行檔案：

```
$ ./SimpleDemoAppWizard //project name
```

2.4.3 執行結果

本範例程式執行結果如圖 2-189。

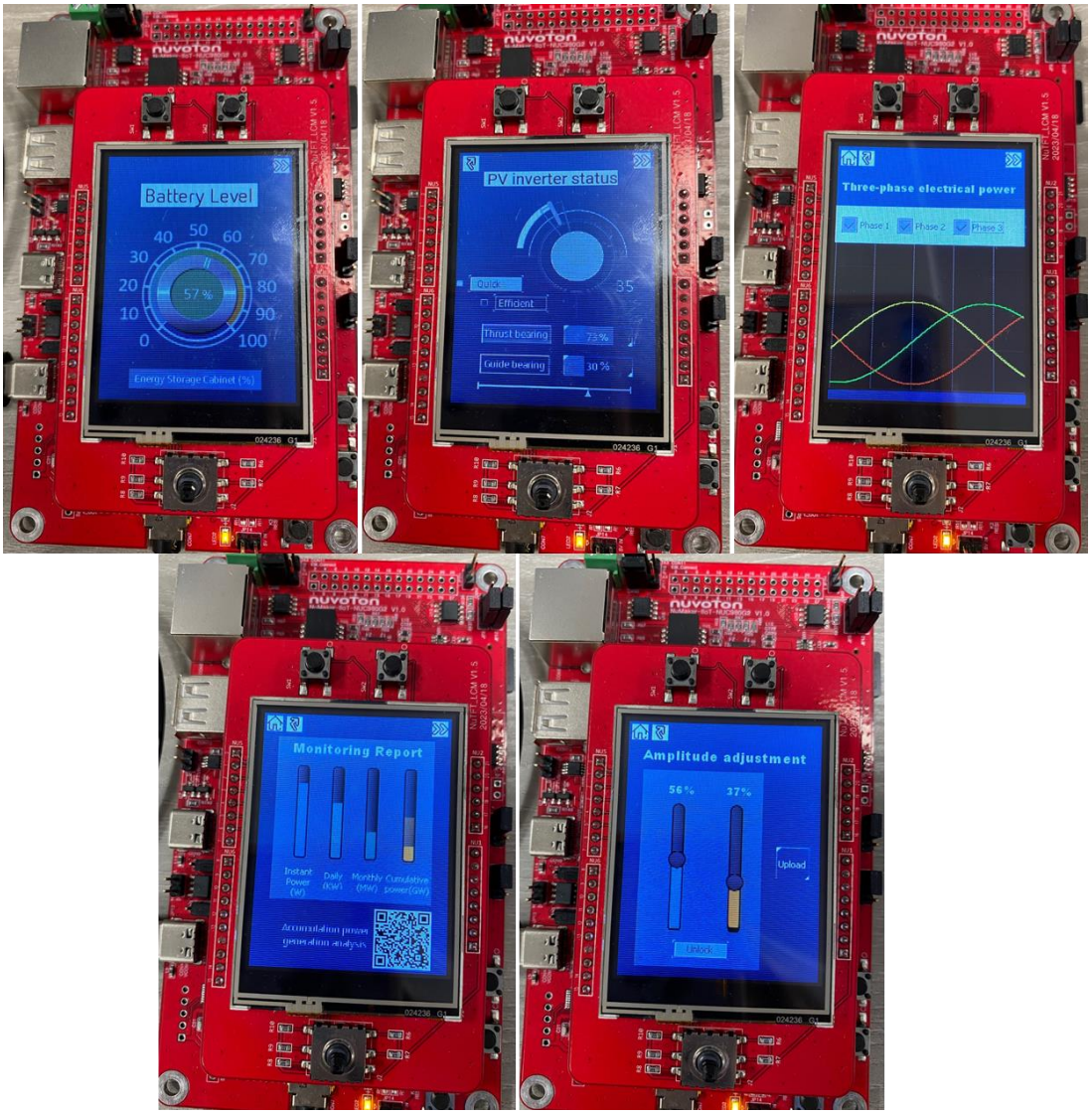


圖 2-189 執行結果

3. 程式介紹

在 emWin Linux package 中，*main.c* 會對程式進行初始化，設定觸控螢幕、frame buffer 與 Multiple Buffering 等，接著導向應用主程式 *MainTask2()*開始執行專案。下列程式位於 *main.c*。

```
……
// Declare two frame buffers
unsigned char g_au8FB[240*320*2*2];
unsigned char *g_pu8FB;

void *MainTask_ISR(void *arg)
{
    unsigned int gpio = 0xC7;

    printf("Main Task thread\n");
    //
    // Enable Backligh
    //
    gpio_export(gpio);
    gpio_set_dir(gpio, 1);
    gpio_set_value(gpio, 1);
    printf("Backlight enabled!\n");
    //
    // emWin Entry Point
    //
    GUI_Init();
    //
    // Create a dirty device.
    //
    GUI_DIRTYDEVICE_Create();
    WM_MULTIBUF_Enable(1);
    MainTask2();
}

void *TouchTask_ISR(void *arg)
{
    printf("Touch Task thread\n");
    TouchTask();
}

int main()
{
    int ret;
    int i, t = 0;
    FILE *fpVideoImg;
    int uVideoSize;
    pthread_t tid1, tid2;

    vpost_fd = open("/dev/fb0", O_RDWR);
    if (vpost_fd == -1)
    {
        printf("Cannot open fb0!\n");
        return -1;
    }
```

```
    if (ioctl(vpost_fd, FBIOGET_VSCREENINFO, &fb_vinfo) < 0)
    {
        perror("ioctl FBIOGET_VSCREENINFO");
        close(vpost_fd);
        return -1;
    }
    //
    // emWin Multiple Buffering
    //
    g_pu8FB = (unsigned char *)g_au8FB;

    fb_vinfo.yres_virtual = fb_vinfo.yres * 1;
    fb_vinfo.yoffset = 0;
    if (ioctl(vpost_fd, FBIOPUT_VSCREENINFO, &fb_vinfo))
    {
        perror("ioctl FBIOPUT_VSCREENINFO");
        close(vpost_fd);
        return -1;
    }

    if (ioctl(vpost_fd, FBIOGET_FSCREENINFO, &fb_finfo))
    {
        perror("ioctl FBIOGET_FSCREENINFO");
        close(vpost_fd);
        return -1;
    }

    g_line_length = fb_finfo.line_length;

    uVideoSize = fb_vinfo.xres * fb_vinfo.yres * fb_vinfo.bits_per_pixel / 8 * 1;

    printf("uVideoSize = %d\n", uVideoSize);
    printf("fb_vinfo.xres = %d\n", fb_vinfo.xres);
    printf("fb_vinfo.yres = %d\n", fb_vinfo.yres);
    printf("fb_finfo.line_length = %d\n", fb_finfo.line_length);

    pVideoBuffer = mmap(NULL, uVideoSize, PROT_READ|PROT_WRITE, MAP_SHARED, vpost_fd, 0);

    printf("pVideoBuffer = 0x%08X\n", (unsigned int)pVideoBuffer);
    if(pVideoBuffer == MAP_FAILED)
    {
        printf("LCD Video Map Failed!\n");
        exit(0);
    }

    g_xres = fb_vinfo.xres;
    g_yres = fb_vinfo.yres;
    g_bits_per_pixel = fb_vinfo.bits_per_pixel;

    pthread_create(&tid1, NULL, MainTask_ISR, (void *)"MainTask2");
    pthread_create(&tid2, NULL, TouchTask_ISR, (void *)"TouchTask");

    pthread_join(tid1, NULL);
    pthread_join(tid2, NULL);

    /* Close LCD */
```

```
    munmap(pVideoBuffer, uVideoSize);  //return memory
    close(vpost_fd);
    return 0;
}
```

若想更新 GUI 的可用位元組數及 Multiple Buffering 功能開關設置，可分別於 *LCDConf2.c* 及 *GUIConf2.c* 中調整相關參數。

LCDConf2.c:

```
//
// Buffers / VScreens
//
#define NUM_BUFFERS  2 // Number of multiple buffers to be used
#define NUM_VSCREENS 1 // Number of virtual screens to be used
```

GUIConf2.c:

```
#include "GUI.h"
…
//
// Define the available number of bytes available for the GUI
//
#define GUI_NUMBYTES  (4 * 1024 * 1024)
```

進入主程式 MainTask2()後，將執行 emWin appWizard 初始化、顯示 GUI 物件等動作。下列程式位於 *APPW_MainTask2.c*。

```
void MainTask2(void) {
  // Setup configuration dependent pointers
  APPW_X_Setup();

  // Initialize AppWizard
  APPW_Init(APPW_PROJECT_PATH);

  // Create all persistent screens except initial screen
  APPW_CreatePersistentScreens();

  // Create initial screen...
  APPW_CreateRoot(APPW_INITIAL_SCREEN, WM_HBKWIN);

  // ...and keep it alive

  while (1) {
#if 0
    while (GUI_Exec1()) {
      APPW_Exec();
    }
    APPW_Exec();
    GUI_X_Delay(5);
#else
    GUI_Exec1();
    APPW_Exec();
    GUI_X_Delay(1);
#endif
  }
```

```
}
```

將 AppWizard 專案導出後，所有 GUI 設計都會被程式化，分別存在對應名稱的 c 檔中，例如 Source\Generated 檔案包含所有畫面中的物件及屬性設置，Source\CustomCode 內則包含各畫面設置的所有 interactions 定義。若使用者想在觸發元件時，額外執行某些程式行為，即可在 *ID_SCREEN_Home_Slots.c*、*ID_SCREEN_02_Slots.c ~ ID_SCREEN_05_Slots.c* 的對應程式中新增。

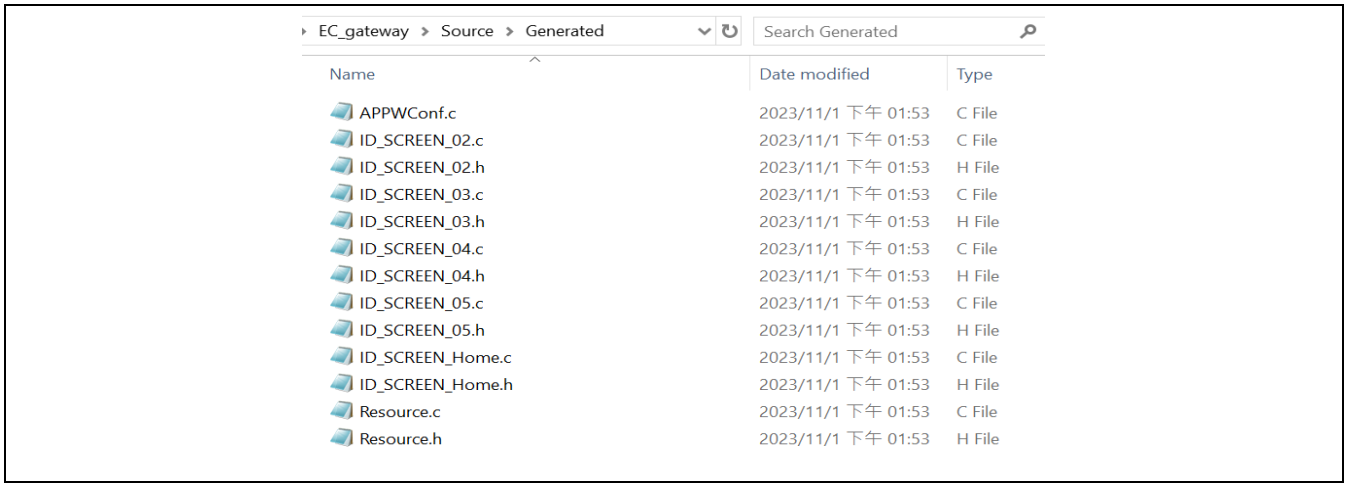


圖 3-1 物件屬性設置導出程式

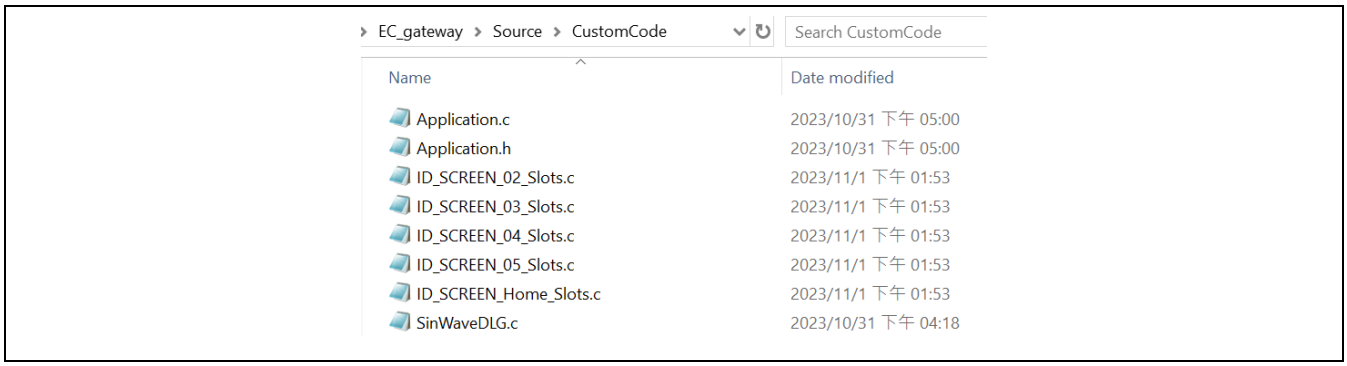


圖 3-2 物件互動設置導出程式

以在主畫面物件互動時，新增程式為例，客戶可透過註記找到對應的 interaction，並在 functions 中新增自訂程式。此段程式碼位於 *ID_SCREEN_Home.c*。

```
// interaction target : ID_SCREEN_Home__ID_ROTARY_val__WM_NOTIFICATION_VALUE_CHANGED
void ID_SCREEN_Home__ID_ROTARY_val__WM_NOTIFICATION_VALUE_CHANGED(APPW_ACTION_ITEM *
pAction, WM_HWIN hScreen, WM_MESSAGE * pMsg, int * pResult) {
  GUI_USE_PARA(pAction);
  GUI_USE_PARA(hScreen);
  GUI_USE_PARA(pMsg);
  GUI_USE_PARA(pResult);
```

```
  /*
    Add the customer's code in here
                                        */
}

……
```

4. 軟體與硬體需求

4.1 軟體需求

- IDE 版本
  - Ubuntu 18.04
- AppWizard 版本
  - AppWizard_V134a_630a

4.2 硬體需求

- 電路元件
  - NuMaker-IIoT-NUC980G2D V1.0
- 線路示意圖
  - 如圖 4-1，透過 Type-C USB 連接 NuMaker-IIoT-NUC980G2D 的 VCOM 與 PC。
  - 如圖 4-2，將 NuMaker-IIoT-NUC980G2D 開發板翻到背面，將 micro SD 插入卡槽。

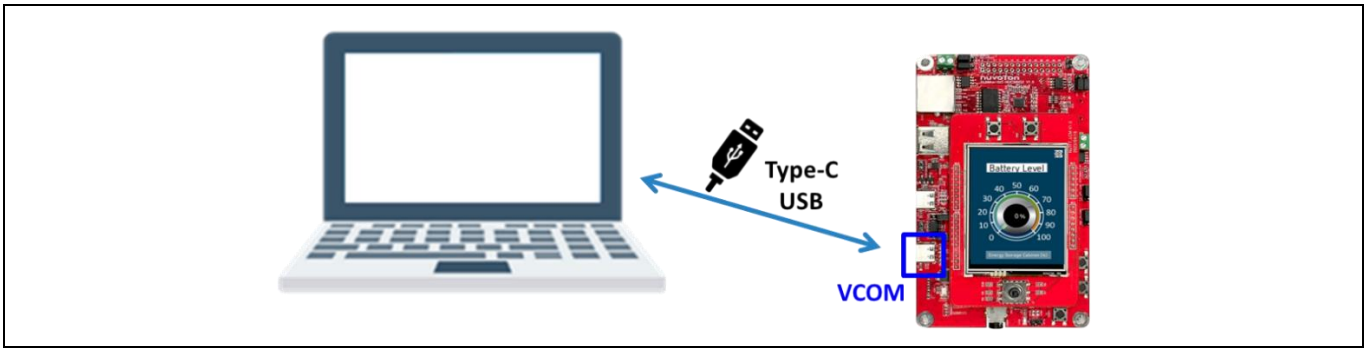


圖 4-1 以 Type-C USB 線將 NuMaker-IIoT-NUC980G2D 與 PC 相連

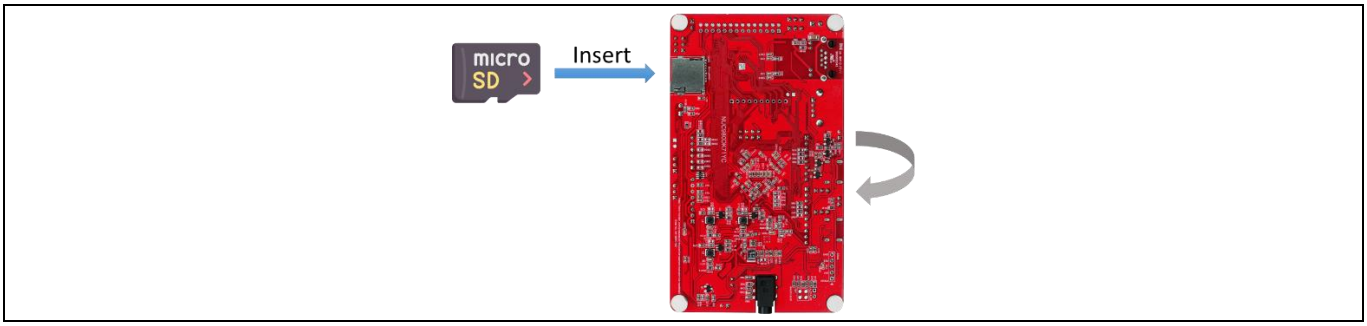


圖 4-2 將 micro SD 插入卡槽

5. 目錄資訊

EC_NUC980_AppWizard_New_Energy_Gateway_V1.00	
Software Tools	Necessary software tools for this example.
EC_gateway	New energy gateway sample project of AppWizard. Please open it by AppWizard_V134a_630a.exe.
new_energy_gateway	The execute file of this example code. Please execute in the evaluation board.
ts.conf	A configuration file used to adjust the calibration, analysis method, compensation and other data of the touch screen.

圖 5-1 目錄資訊

6. 範例程式執行

1. 於新唐官網下載 emWin package，將檔案複製到 Linux 環境中解壓縮，並依 *README.txt* 步驟操作(詳見章節 2.4.1)

2. 將 AppWizard 專案(如同目錄內資料夾 EC_gateway)生成資料夾"*Resource*"與"*Source*"複製到 Linux emWin package 路徑: */Sample/SimpleDemoAppwizard/Application*(備註:請直接刪除原始資料夾，避免透過複製貼上的方式取代)

   - 在 Linux 環境中執行 Makefile 檔案，生成程式 bin 檔
   - 將 bin 檔置於 SD 卡，並插入開發板 SD 卡插槽

3. 在 Windows 環境下，透過 console 以 root 權限登入系統

   - 掛載 SD 卡資料
   - 執行 bin 檔

備註：若發現執行程式發現有部分 Widget 功能失效，可能是 emWin Package 與 AppWizard Project 版本不匹配的緣故。若發生此情形，請依下列步驟處理：

- 安裝 emWin package 內所提供的最新版 AppWizard IDE
- 安裝完成後，以最新版 AppWizard IDE 開啟 AppWizard 專案 (*../EC_NUC980_AppWizard_New_Energy_Gateway_V1.00/EC_gateway*)
- 開啟專案後，會跳出需要升級專案的提醒視窗，點選 Yes 即可升級
- 專案升級後，透過工具列 File>Export & Save 重新匯出 AppWizard 檔案，接著即可回步驟 2 繼續範例程式執行流程

7. 修訂紀錄

Date	Revision	Description
2024.04.02	1.00	初始發布。

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.