以 AppWizard 设计新能源网关 GUI 项目

NuMicro® 32位系列微控制器范例代码介绍

文件信息

应用简述	本程序使用 SEGGER emWin AppWizard 工具，结合 NUC980 系列开发平台与开发资源，完成新能源网关(New Energy Gateway)之图形化使用接口。
BSP 版本	emWin Software Package for NUC980
开发平台	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

本范例程序将示范如何使用 SEGGER emWin AppWizard 工具，搭配 NuMaker-IIoT-NUC980G2D 开发平台，逐步完成新能源网关(Gateway)的 GUI(Graphical User Interface)接口设计与画面互动。

1.1 原理

1.1.1 系统方块图

此范例程序使用新唐 GUI 开发平台 NuMaker-IIoT-NUC980G2D 开发板，其使用分辨率 240x320 且带有尺寸为 2.4 吋的 TFT-LCD 电阻式触控面板、主控板上 NUC980G2D 微处理器与容量大小为 1 Gbit 的 SPI NAND Flash。其系统方块图如图 1-2 所示。

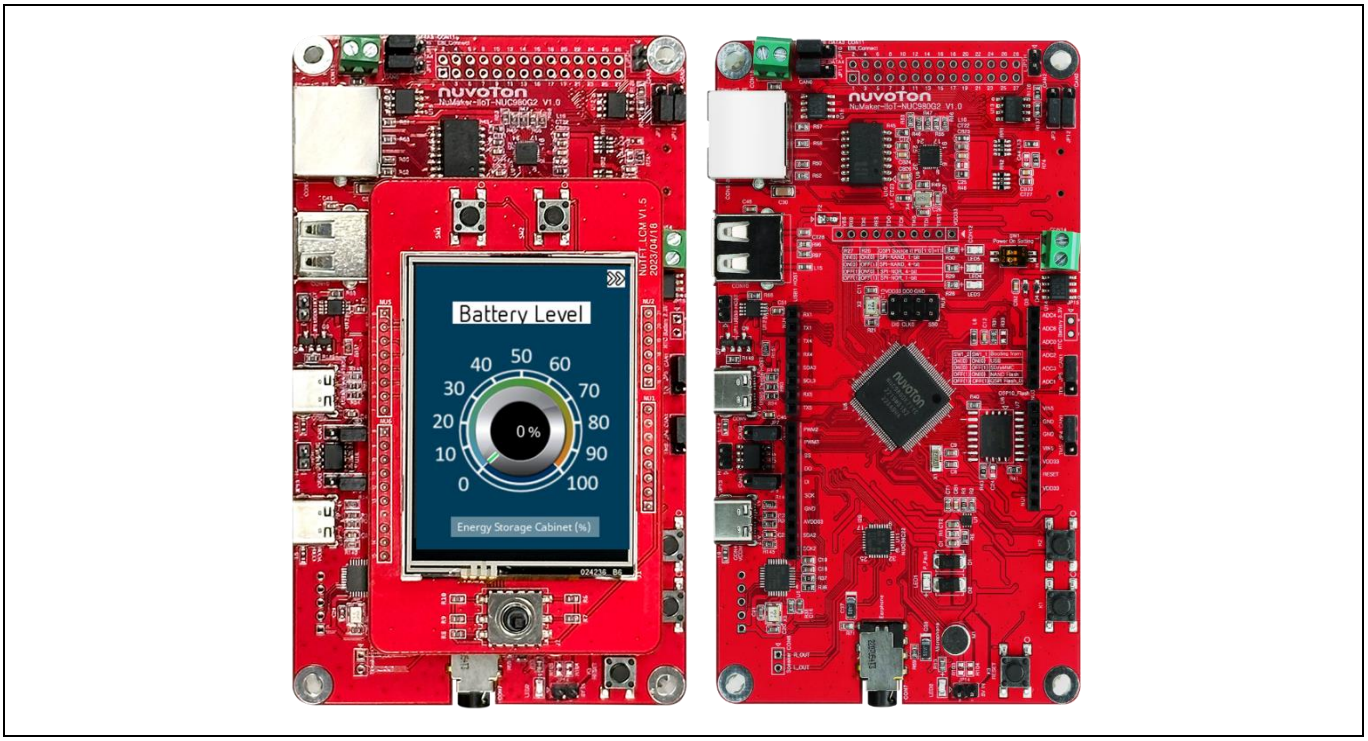


图 1-1 NuMaker-IIoT-NUC980G2D 开发板

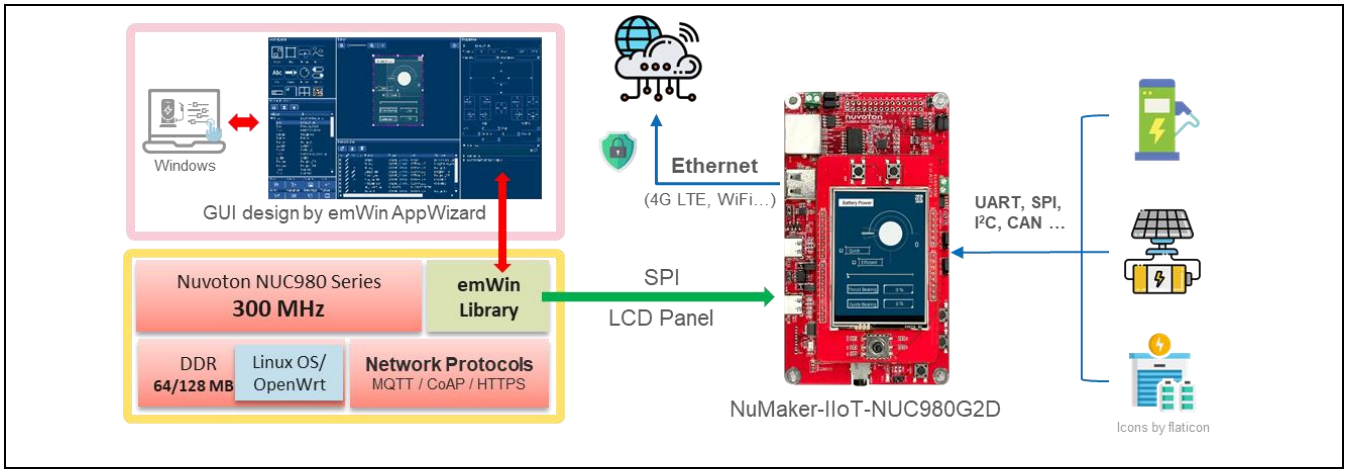


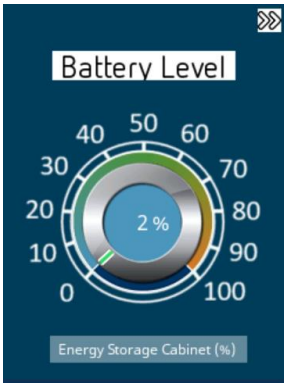
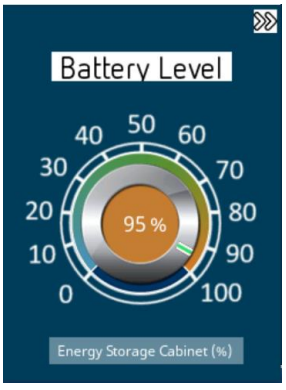
图 1-2 系统方块图

1.1.2 相关资源

- 新唐官方论坛：Nuvoton emWin Forum
- SEGGER emWin 官方论坛：Segger emWin Forum
- SEGGER emWin AppWizard 工具使用手册：AppWizard User Manual

1.2 新能源网关用户接口(UI)设计

此范例设计画面如表 1-1 所示：

	初始画面	互动效果
主画面		

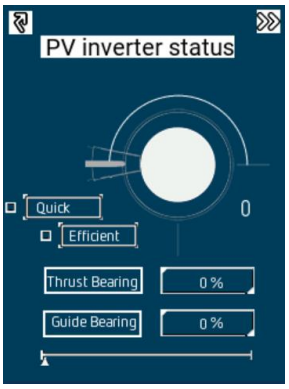
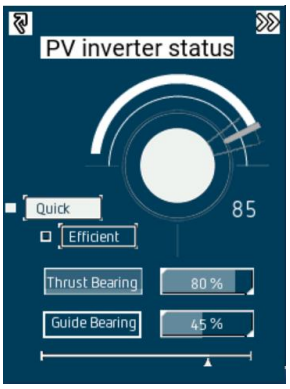
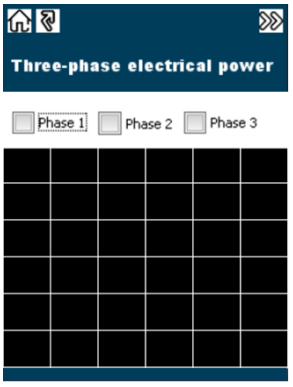
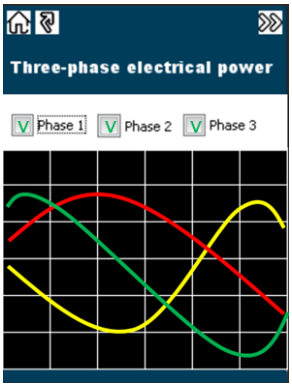
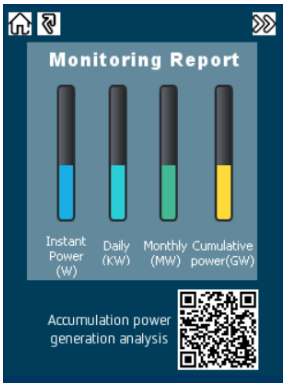
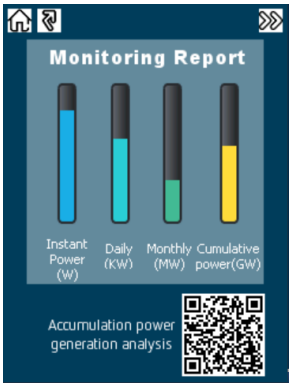
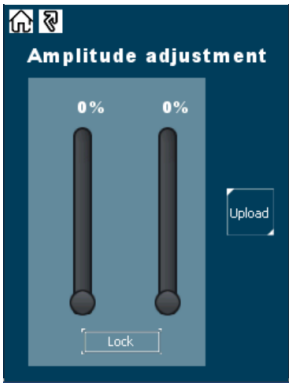
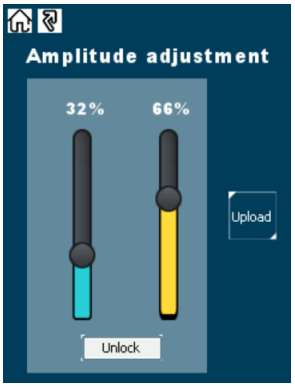
页面二		
页面三		
页面四		
页面五		

表 1-1 新能源网关画面

2. 操作步骤

使用 AppWizard 完成一项 GUI 项目的执行步骤可分为四大阶段，第一阶段需准备硬件与软件下载，第二阶段则于 AppWizard 中完成对应的画面设计、对象、互动设置、动画设置与绘图设定，第三阶段使用 AppWizard 导出程序代码并同时加入客制化程序，以完成更多应用功能，第四阶段则是将应用程序实际烧录到开发板操作。以下章节就各项阶段详细说明执行步骤。

2.1 准备相关硬件与软件资源

1. 硬件使用 NuMaker-IIoT-NUC980G2D，相关资源: 开发板介绍、参考线路、使用手册。
2. 软件资源需下载 GUI_Builder 及 AppWizard 软件工具
   - emWin
     - GUIBuilder.exe
   - emWin AppWizard IDE 与 emWin Package
     - 请于新唐官网下载 emWin Package 档案 (需注册并登入官网使用者账号)
     - 将 emWin Package 解压缩后，点选 *AppWizard_Install.exe* 即可安装 emWin AppWizard IDE

软件资源皆已包含在范例程序包内，档案位置如图 2-1 所示。

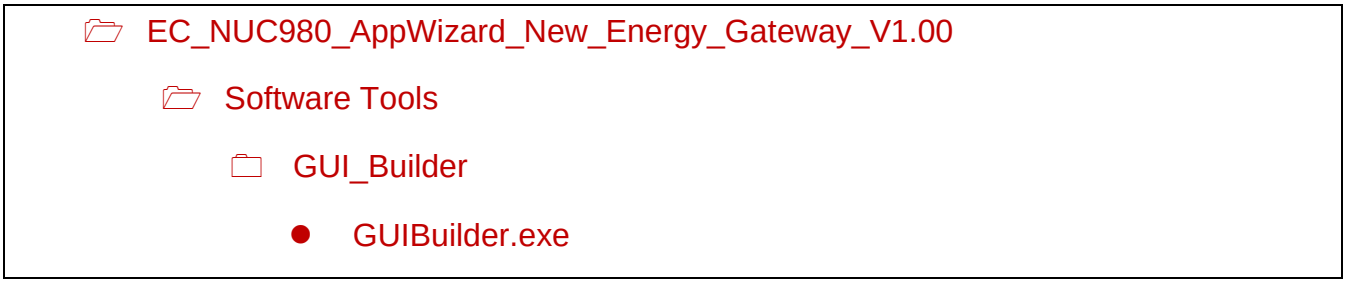


图 2-1 必要工具路径

3. NuMaker-IIoT-NUC980G2D 使用 Linux 环境开发，请确认计算机已安装 VMWare 虚拟机，或透过新唐 Github 网站下载 NUC980 系列专用虚拟机。

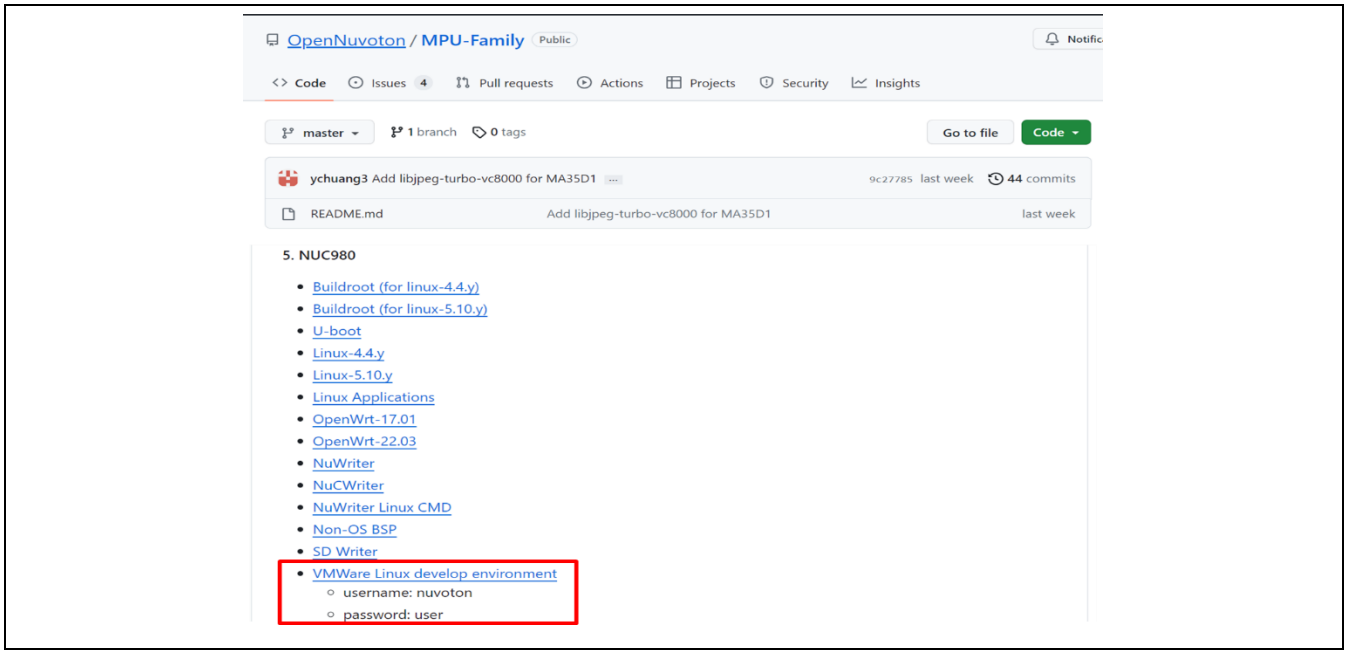


图 2-2 新唐提供之 VMWare 虚拟机载点

4. 执行此范例前，用户需先完成 NUC980 Linux Kernel 与基础建设(包含 Buildroot, u-boot, tool chain…等设置)，目前 NuMaker-IIoT-NUC980G2D 有提供 Linux kernel 5.10 版本的资源，详细档案编译流程如下所述：

- 开启 NUC980 VMWare，由 Terminal 进入 MA35D1_Buildroot 文件夹后，透过指令 **make** 编译 NuMaker-IIoT-NUC980G2D 设置档 "nuvoton_nuc980_iot_g2d_defconfig"

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

- 编译完成后，将有五个映像档可下载，用途分别为：
  - Image: 于 DDR 上运作的 Linux 核心映像档
  - uImage: 于 uboot 引导的 Linux 核心映像档
  - dtb: Device Tree 二进制文件，详细描述外设的设定
  - u-boot.bin：Uboot 映像档
  - u-boot-spl.bin：Uboot 辅助程序加载器映像文件，用于从 SPI NAND 启动
- 编译完成后，可分别于下述路径存取档案：

- *"/MA35D1_Buildroot/output/images"* : **Image**、**uImage** 与 **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. 由于有数个需要烧录的档案，因此使用者需要再建立一份 env.txt 配置映像文件烧录位置与设定，env.txt 内容如下：

```
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. 下载 NUC980 专用的烧录器 NuWriter，将编译好的映像档与 env.txt 透过 NuWriter 烧录到 NuMaker-IIoT-NUC980G2D 开发板，烧录流程如下所述：

- 将开发板 SW4 切换至 USB boot mode
- 烧录模式(Choose Type)选择：SPI NAND
- 清除所有区块内容
- 烧录 u-boot-spl.bin
  - Image Name: 选择指定档案路径
  - Image Type: Loader
  - Image execute address: 0x200
  - Image start offset: N/A
- 烧录 uBoot
  - Image Name: 选择指定档案路径
  - Image Type: Data
  - Image execute address: N/A
  - Image start offset: 0x100000
- 烧录 nuc980-iot-g2d-v1.0.dtb

  - Image Name: 选择指定档案路径
  - Image Type: Data
  - Image execute address: N/A
  - Image start offset: 0x180000
- 烧录 uImage
  - Image Name: 选择指定档案路径
  - Image Type: Data
  - Image execute address: N/A
  - Image start offset: 0x200000
- 烧录环境 (env.txt)
  - Image Name: 选择指定档案路径
  - Image Type: Environment
  - Image execute address: N/A
  - Image start offset: 0x80000
- 烧录完成后，将 SW4 切回 Nand Boot mode 并重启开发板即可使用。

※备注: 更详细 Kernel 编译与烧录流程说明，请参阅手册「NUC980 Linux Environment on VMware User Manual」与「NuMaker-IIoT-NUC980G2D User Manual」(章节 5.5 NuWriter Tool)。

2.2 开始 AppWizard 设计项目

1. 于桌面快捷方式开启 AppWizard 工具。

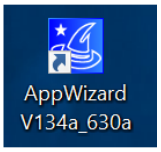


图 2-3 AppWizard 桌面快捷方式图示

2. 点选”Create new project”建立新项目。

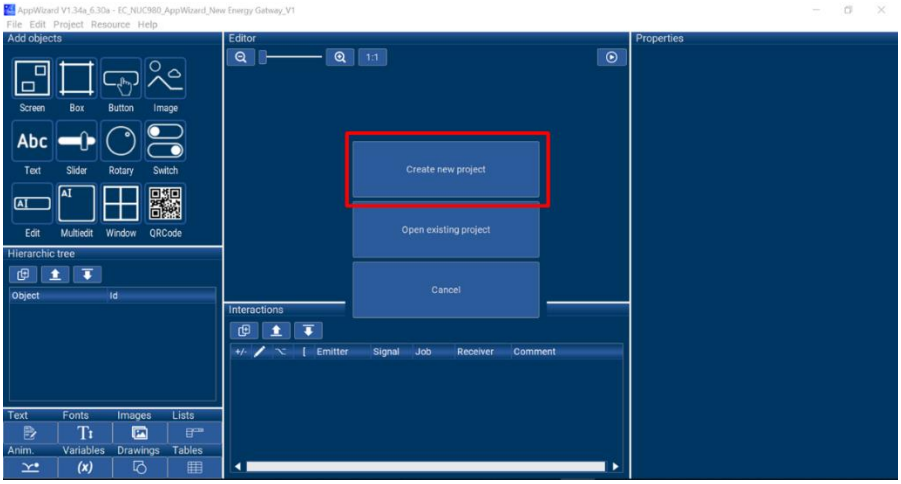


图 2-4 新项目建立页面

3. 依图 2-5 设定项目路径、名称，符合使用屏幕的大小(240x320)及颜色格式：16 Bit, GUICC_M565，勾选 “Enable Multibuffering”后点选 “ok”开启项目设计画面 。

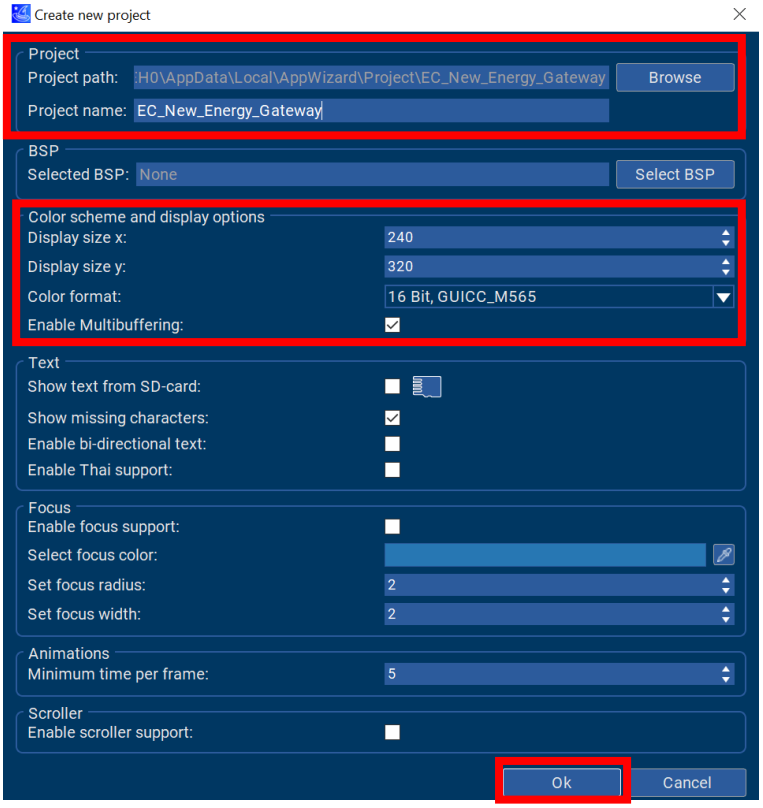


图 2-5 新项目设定项目

4. 于 Add objects 中点选五次 Screen 创建五个画面，并在 Hierarchic tree 内将 Screen 分别命名为 ID_SCREEN_Home、ID_SCREEN_02~ID_SCREEN_05，以利辨认。

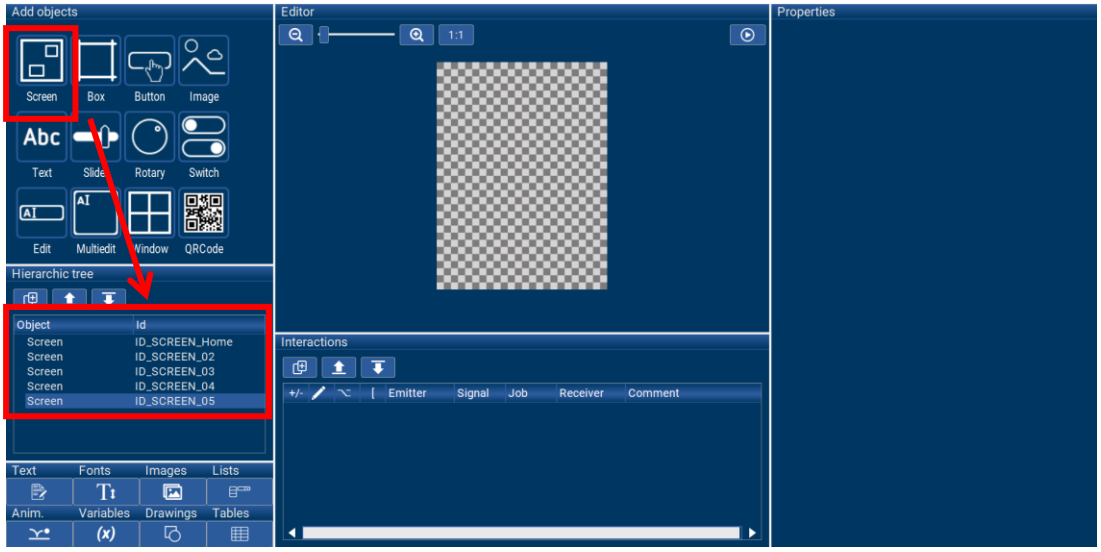


图 2-6 新增 Screen

5. 点选画面左下角”Image”，将此项目使用的 PNG 图片导入 AppWizard。

(档案位于：EC_NUC980_AppWizard_New_Energy_Gateway_V1.00\EC_New Energy Gateway\Resource\Image)

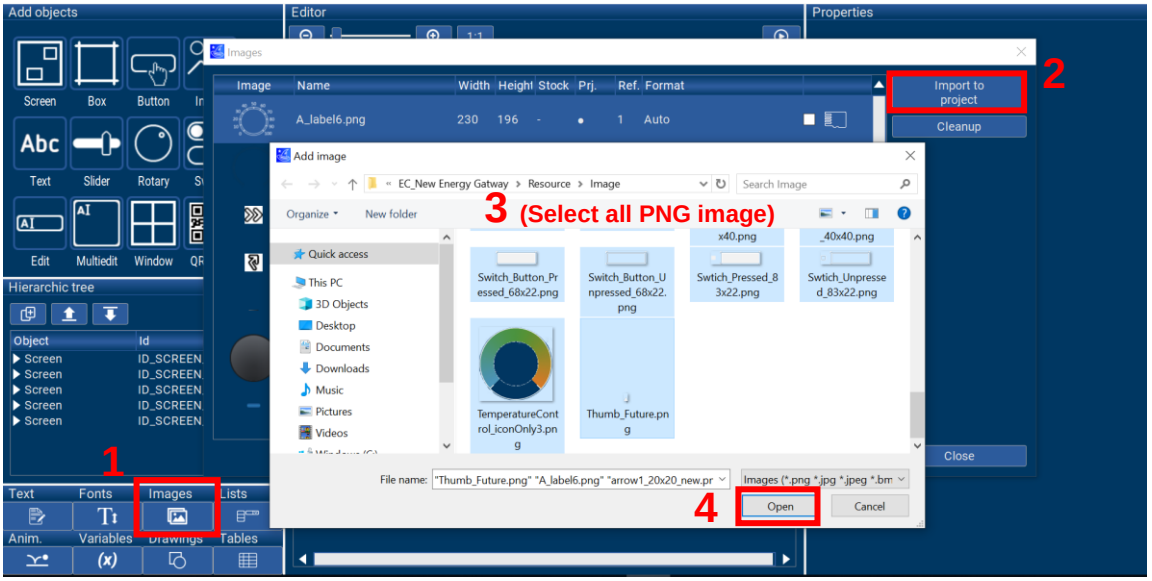


图 2-7 导入 PNG 影像至 AppWizard

6. 分别点选 ID_SCREEN_Home、ID_SCREEN_02~ ID_SCREEN_05，透过 Add objects 中的 Box 新增背景，并透过 box 属性字段的”Set color”调整颜色。

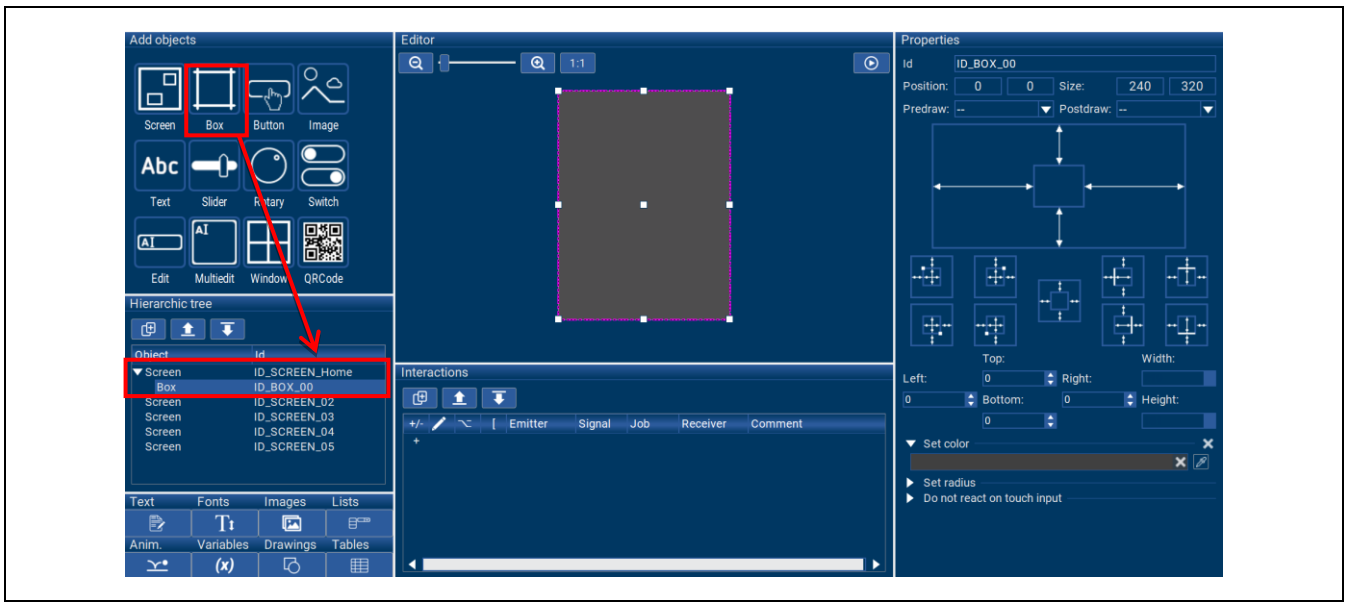


图 2-8 新增 box 对象作为背景

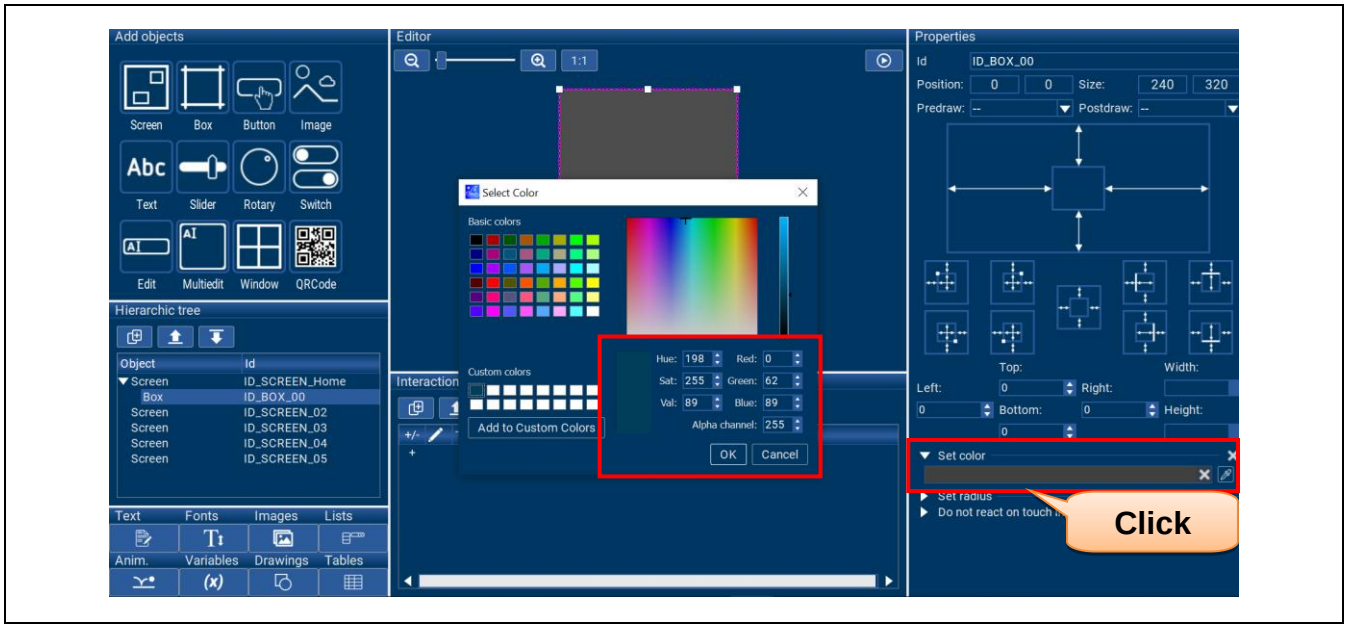


图 2-9 设置 box 颜色

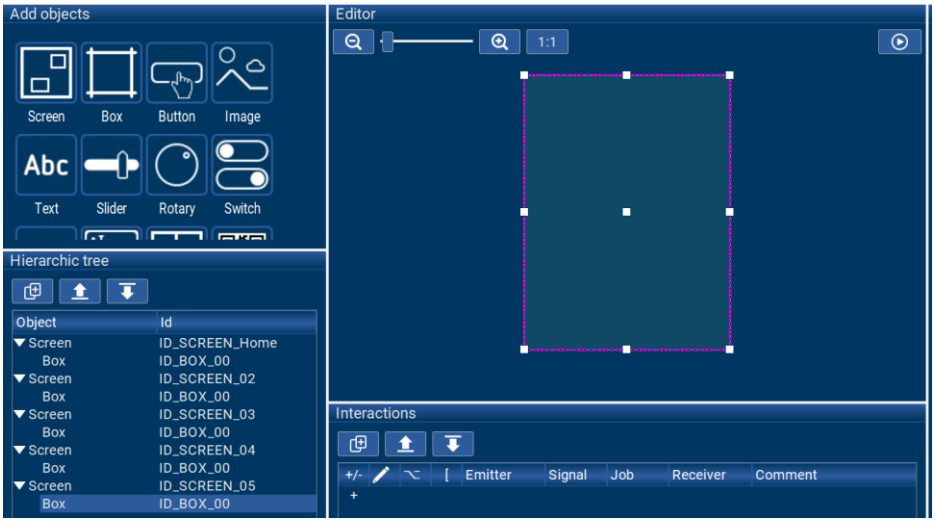


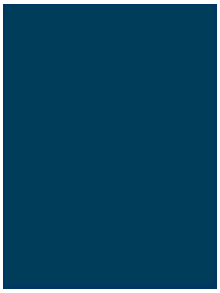
图 2-10 新增所有画面的纯色背景

2.3 AppWizard 页面设计实作

新能源网关(New Energy Gateway)应用广泛，常见应用如储能、充电桩、光伏逆变器与电流量表等多种工业应用，本范例将针对几种常见应用进行 GUI 设计，并详述各画面所使用的对象、互动设置及动画设计，使用者可依需求再自行调整。

2.3.1 电量表设计(主画面)

电池量表的设计可应用在充电桩等需要显示用电量的产品，此设计使用的对象如表 2-1 所示。

New Energy Gateway Description	Application GUI	Widget Type
背景图 240x320		Box
电量刻度背景 (置于 Rotary 下层)		Image

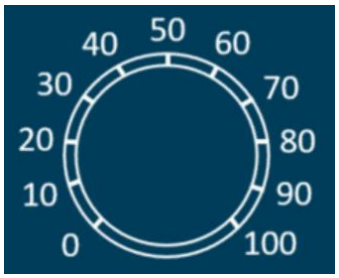
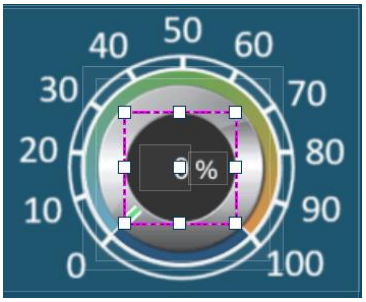
Rotary1 背景图片		Rotary
Rotary1 Maker		
电量刻度表		Rotary
电量变色区块		Box
页面标题	Battery Level	Text
电量数值	0 %	
页尾注记	Energy Storage Cabinet (%)	
下一页按钮		Button

表 2-1 电量表设计对象

2.3.1.1 GUI 对象配置

1. 点选 ID_SCREEN_Home 开始编辑主页画面，透过 Add objects 新增一个大小为 123x123 的 image 对象(ID_IMAGE_00)，并选择图片 ColorVariety.png 作为 bitmap 设置。

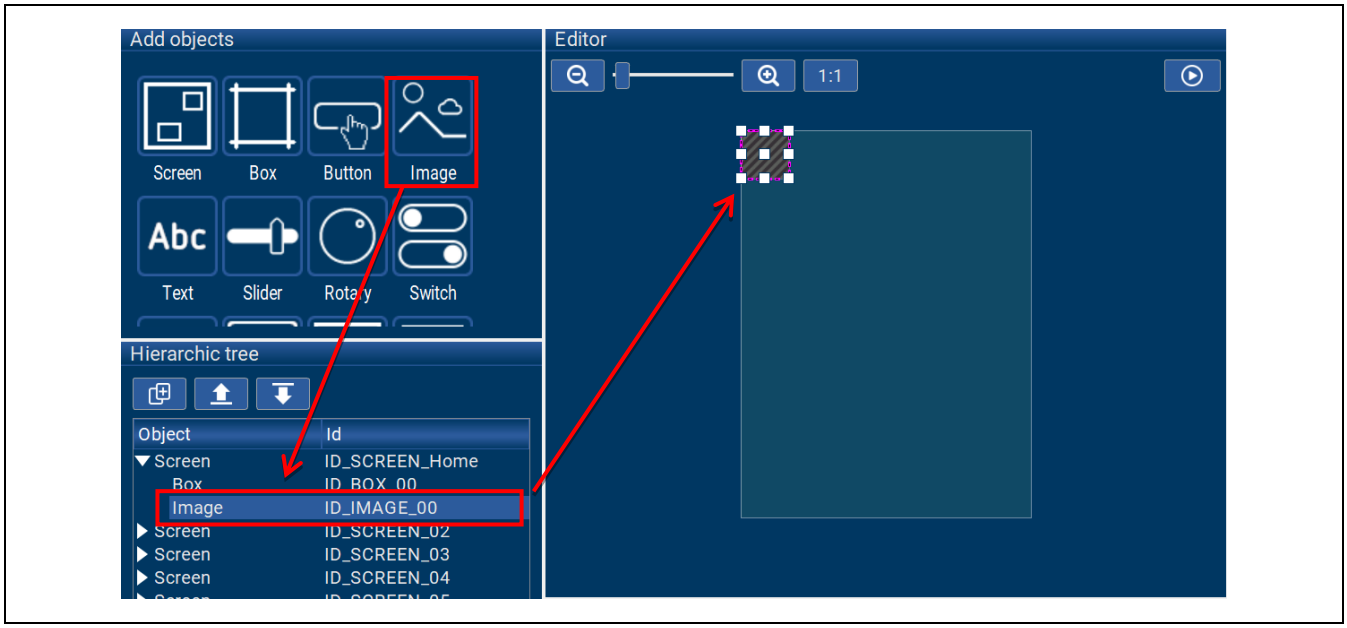


图 2-11 新增 image 物件

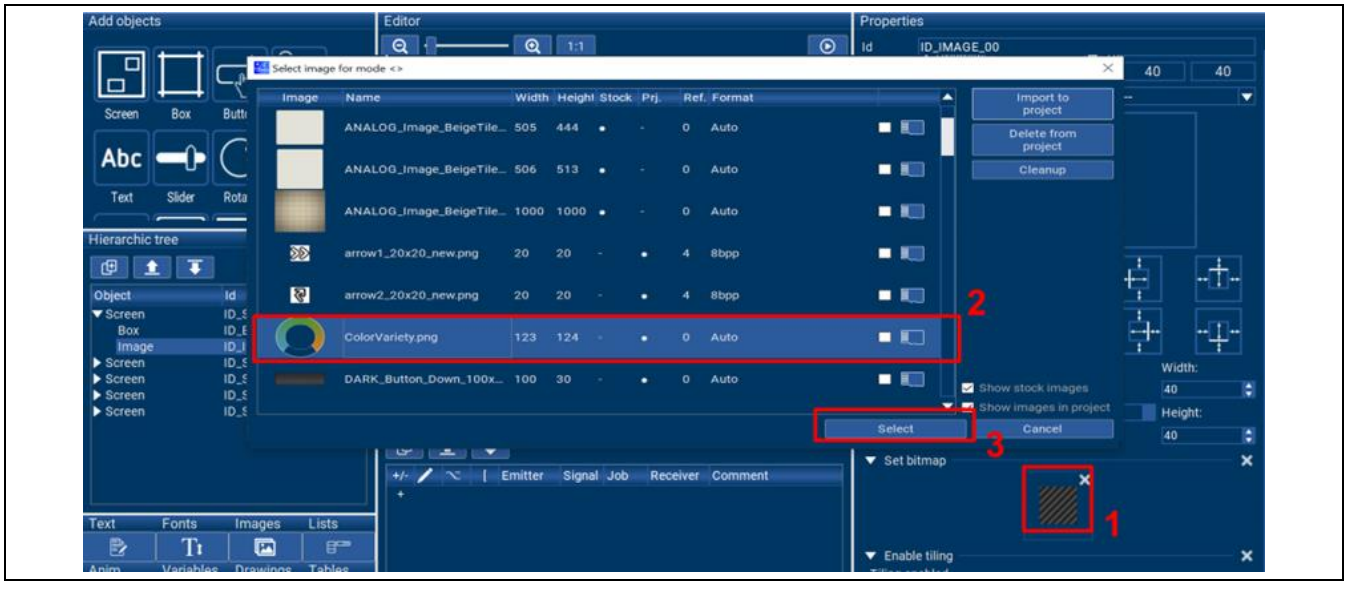


图 2-12 对象 bitmap 设置方式

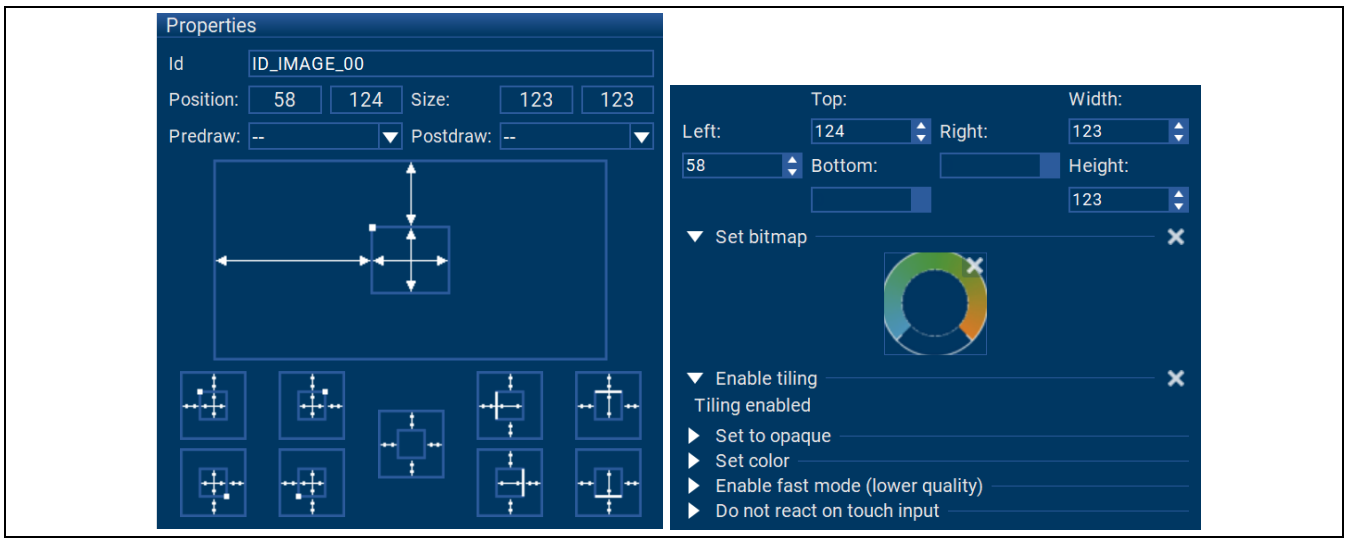


图 2-13 ID_IMAGE_00 属性设置

2. 新增大小为 219x181 的 Rotary 对象(ID_ROTARY_idx)迭在 image 对象上，选择图片 scaleValue.png 做为 bitmap，并取消 Maker 使用。

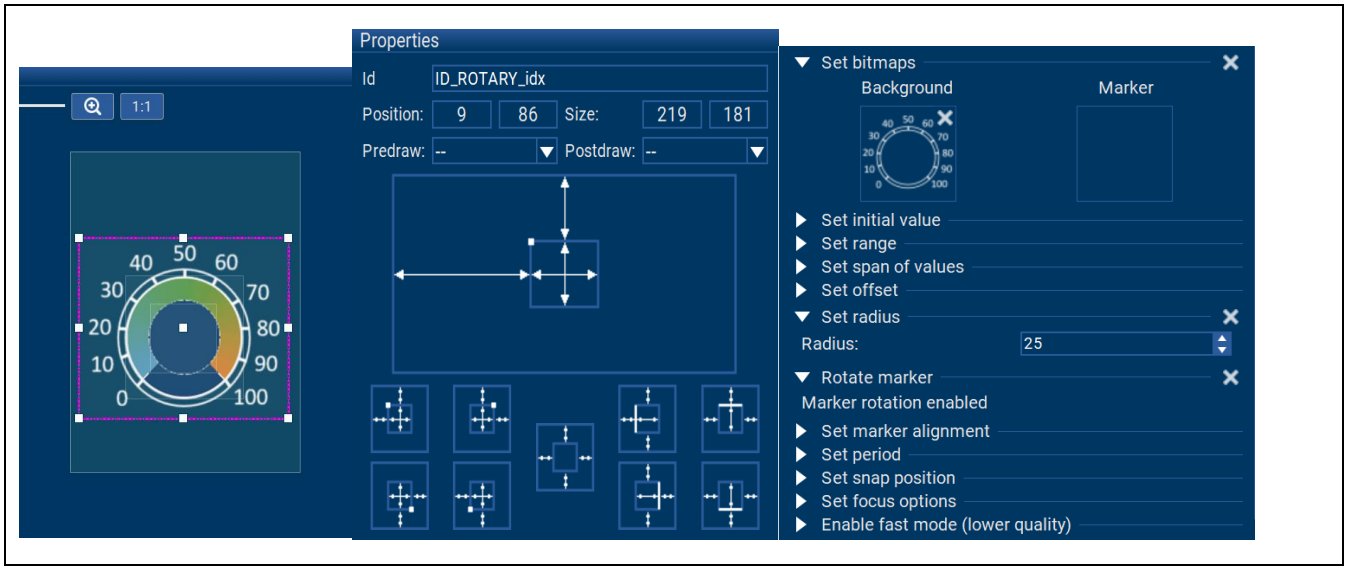


图 2-14 ID_ROTARY_idx 属性设置

3. 新增一个大小为 69x69 的 box 对象(ID_BOX_color)作为变色区块(取消预设色彩)，并迭放在 ID_ROTARY_idx 上方。

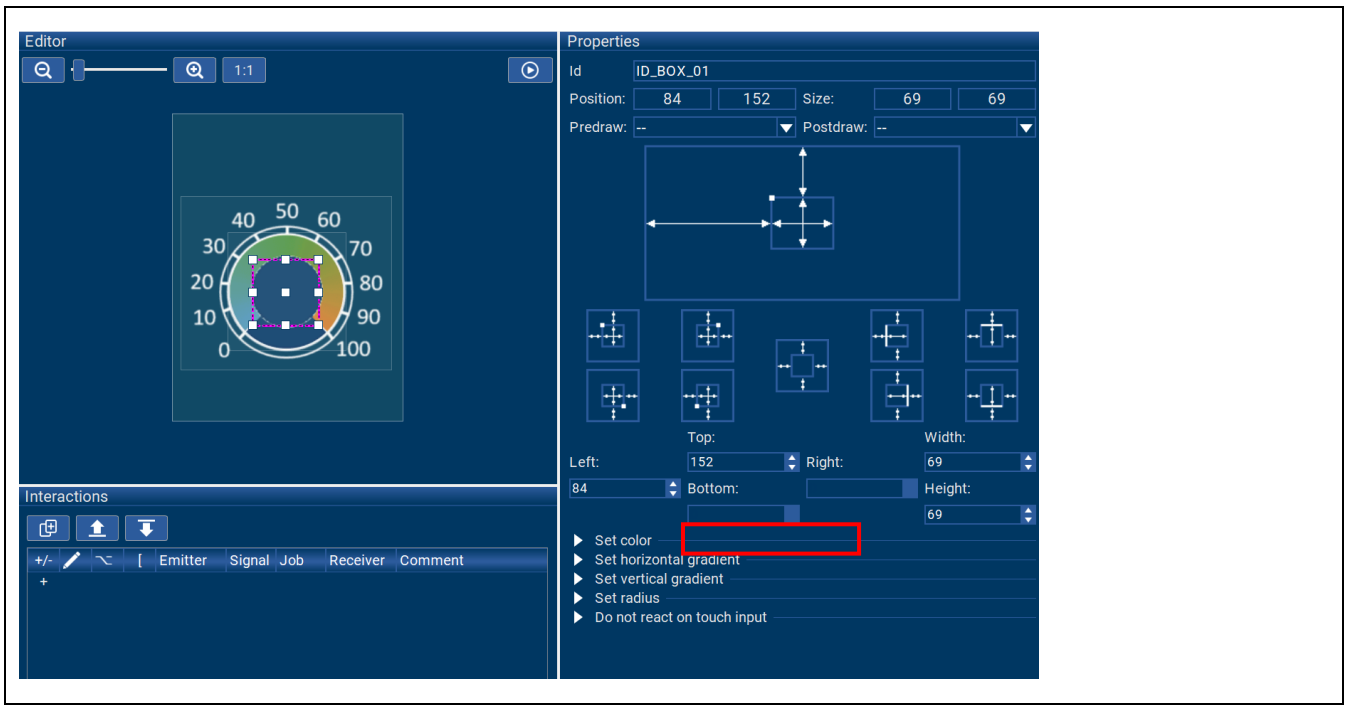


图 2-15 ID_BOX_color 属性设置

4. 再新增一个大小为 107x112 的 Rotary(ID_ROTARY_val)迭在 ID_BOX_color 对象之上，作为数值转轮，并依表 2-2 设置 Rotary 属性。

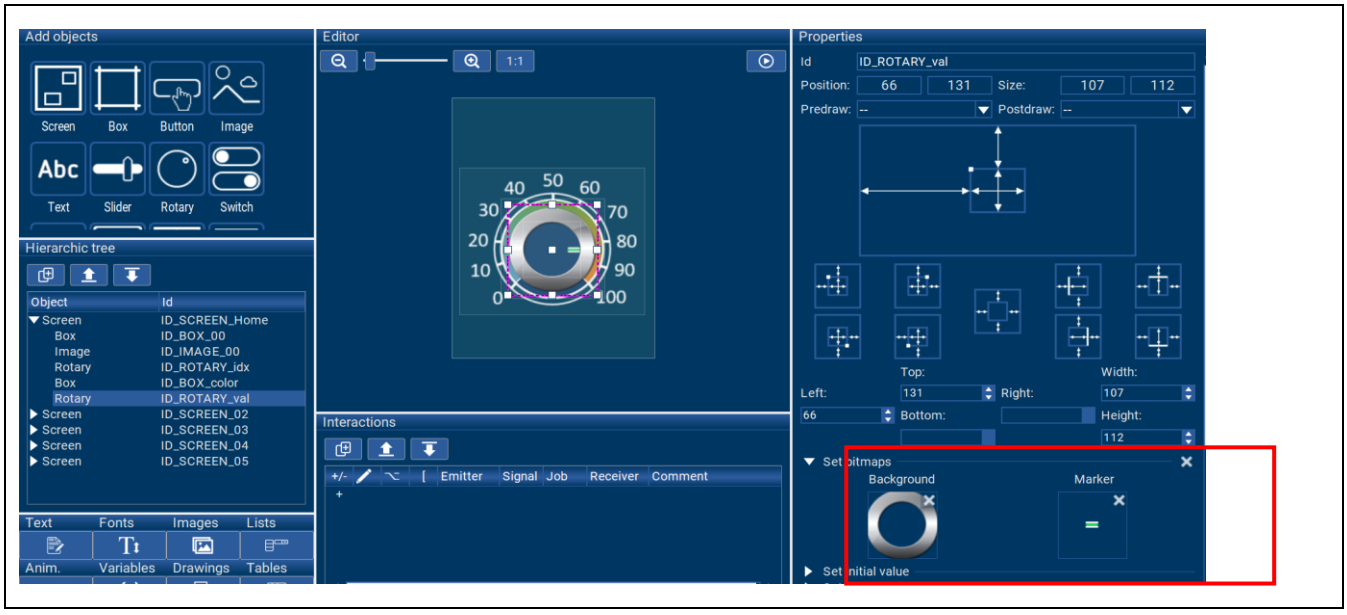


图 2-16 Rotary Bitmaps 设置

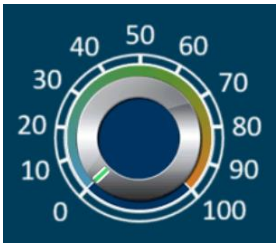


图 2-17 Rotary Properties 设置结果

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 属性设定说明

5. 新增文字以显示标题与说明文字，在 Add objects 中点选两次 Text 对象。

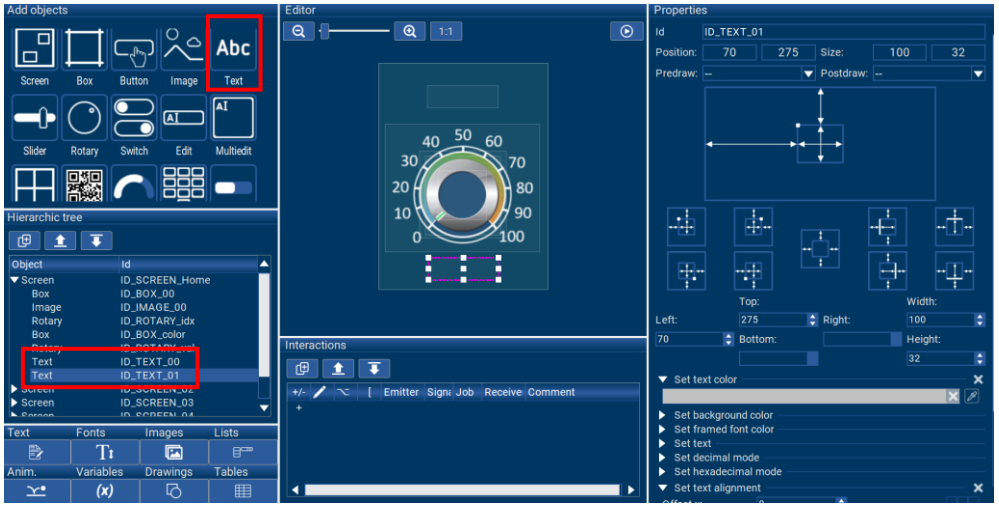


图 2-18 新增 Text 作为标题与说明文字

1) 依图 2-19~图 2-20 步骤，建立文字显示的语言标签并新增 ID_TEXT_00 显示文字。

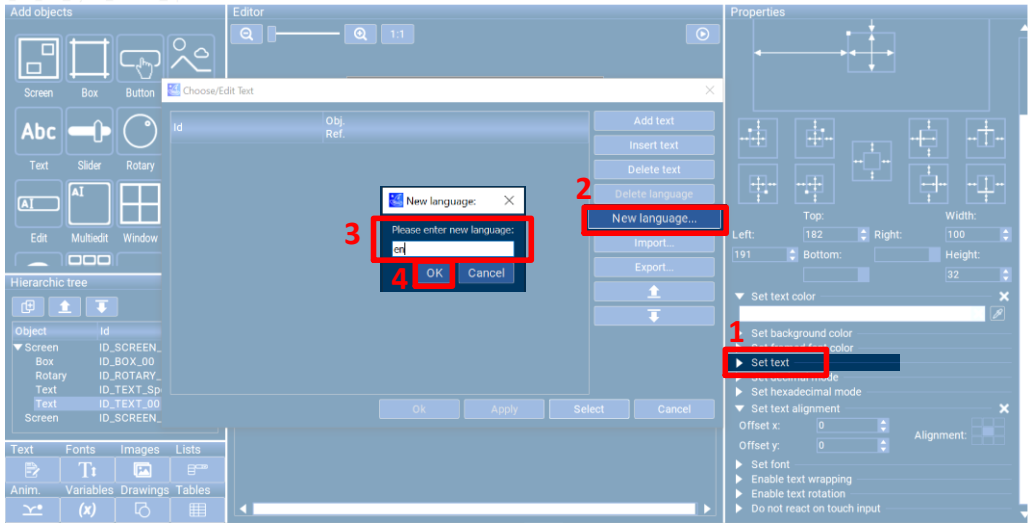


图 2-19 Text 对象 - 创建语言(英文)

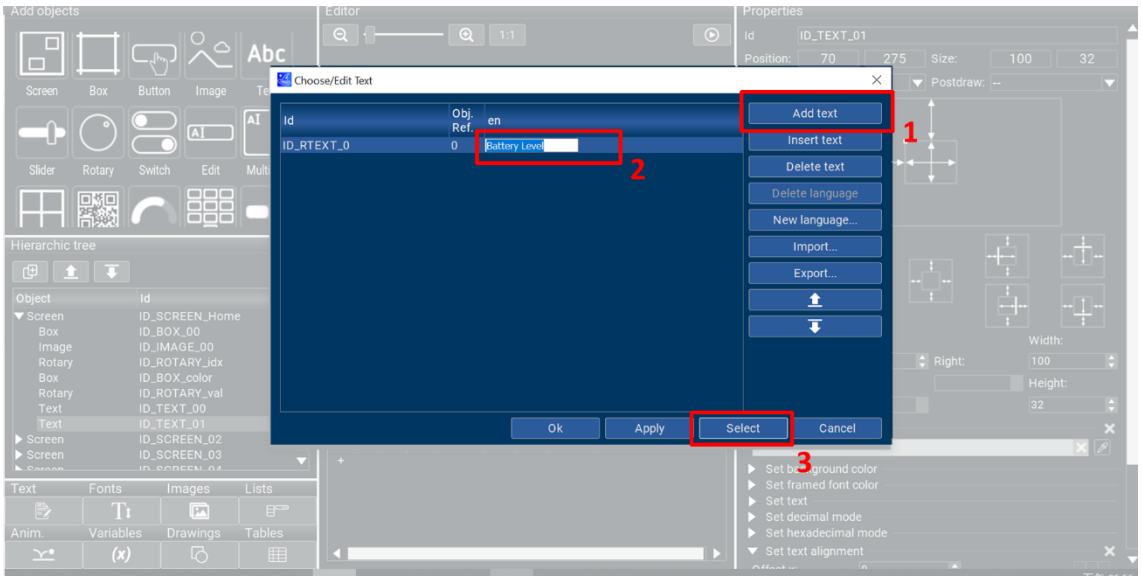


图 2-20 Text 对象- 编辑标题文字内容

2) 依图 2-21~图 2-23 步骤，设定标题文字的颜色、背景与 text alignment。

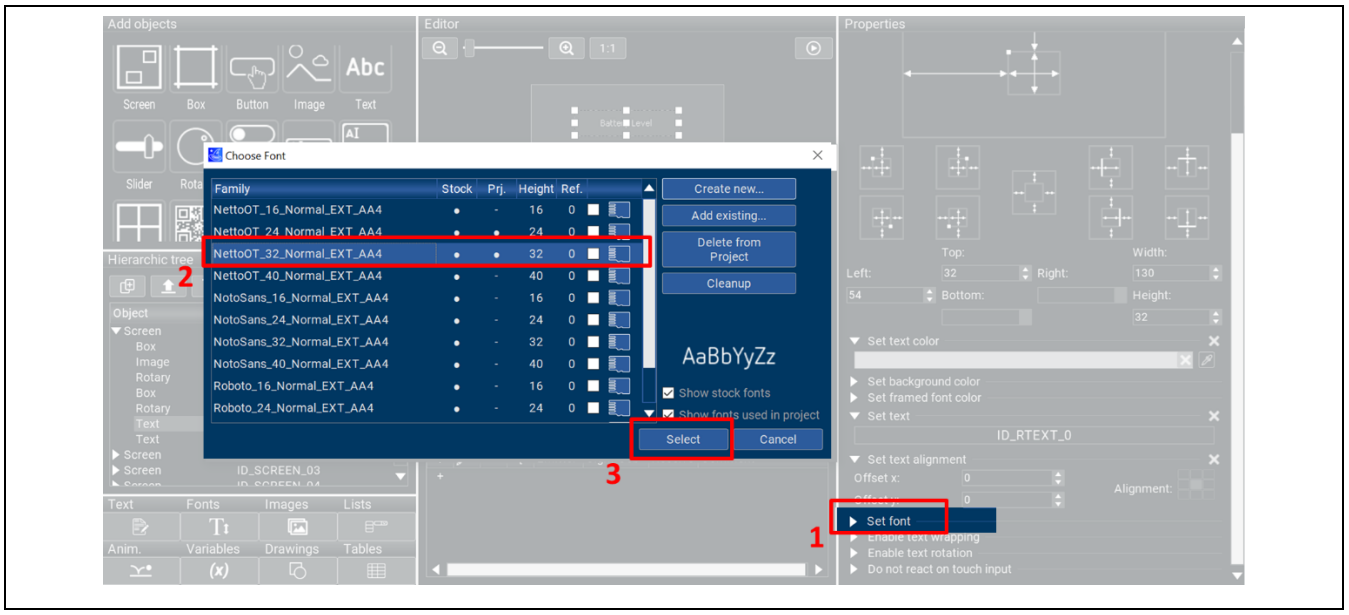


图 2-21 Text 对象- 标题格式设定

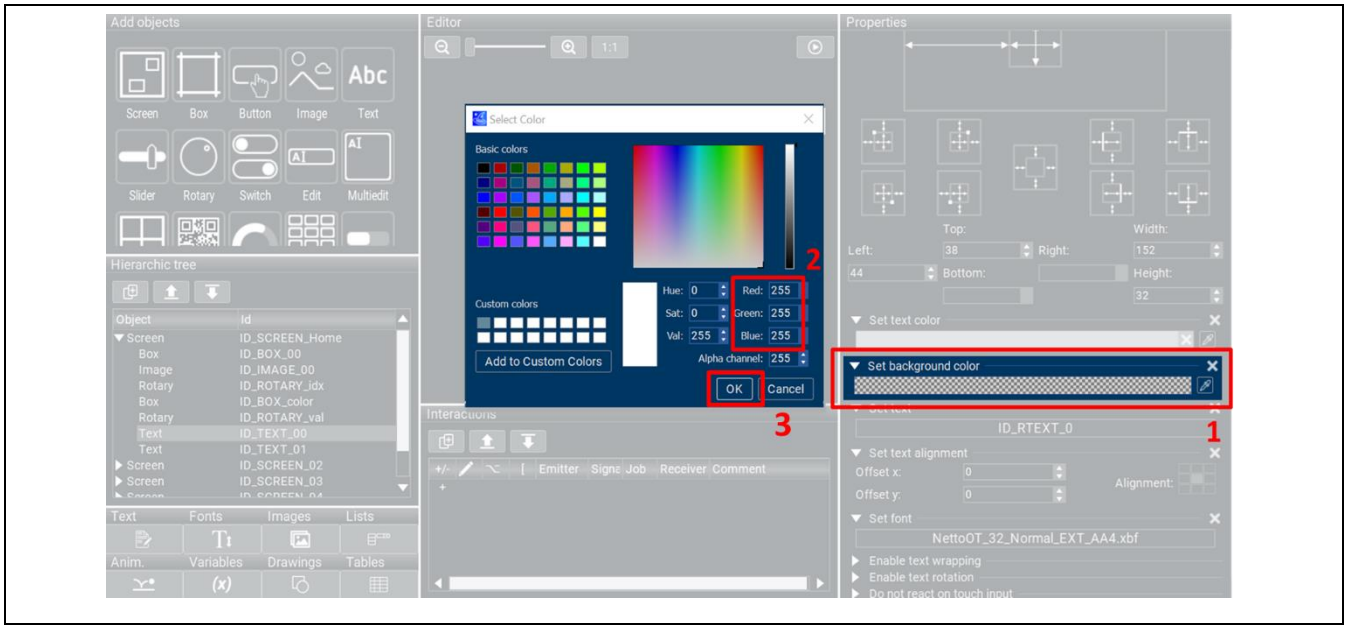


图 2-22 Text 对象-标题背景颜色设置(白色)

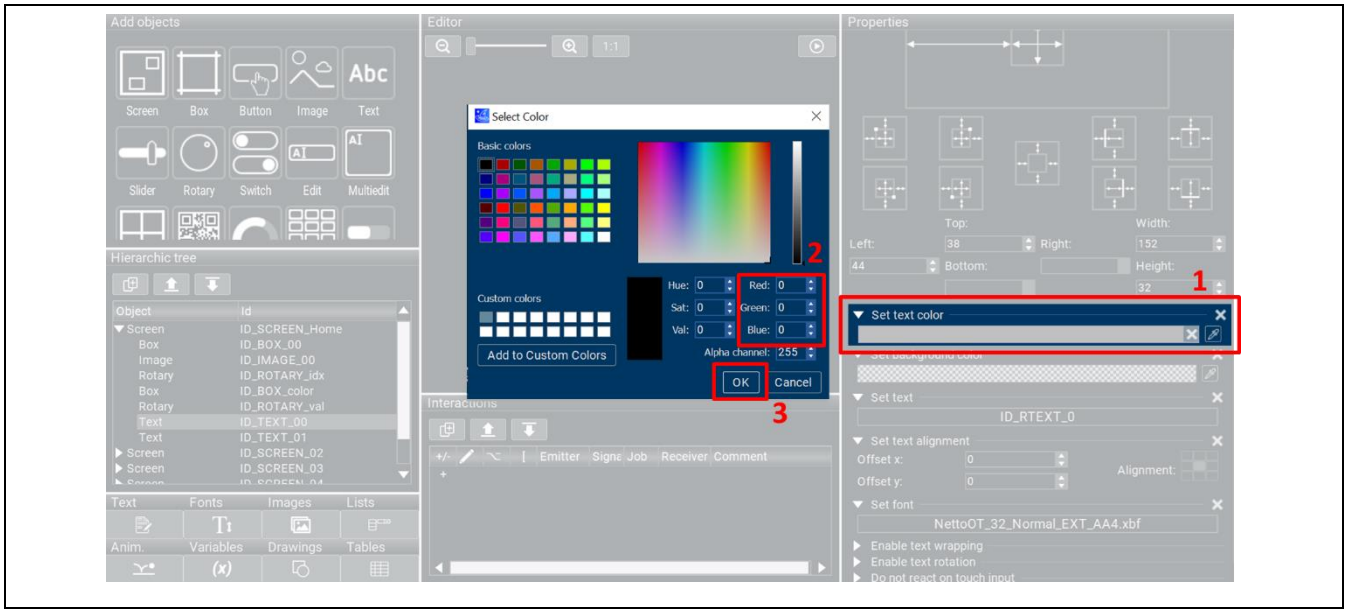


图 2-23 Text 对象-标题颜色(黑色)

3) 用相同方式设定说明文字内容，设定值如图 2-24~图 2-25 所示。

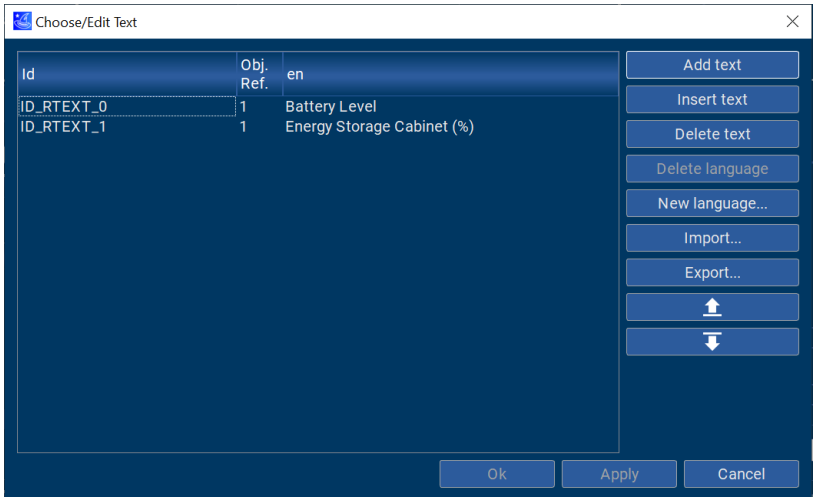


图 2-24 Text 对象-下标文字内容设定

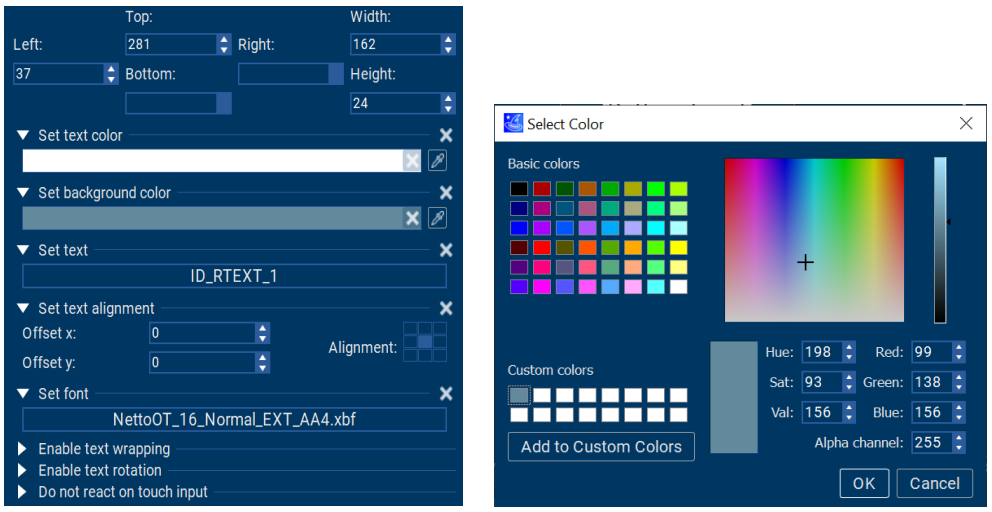


图 2-25 Text 对象-说明文字格式设定

6. 新增换页按钮，在 Add objects 中，点选新增一个 Button 对象并命名为 *ID_BUTTON_next*，依图 2-26~图 2-27 设置其 bitmap 与属性。

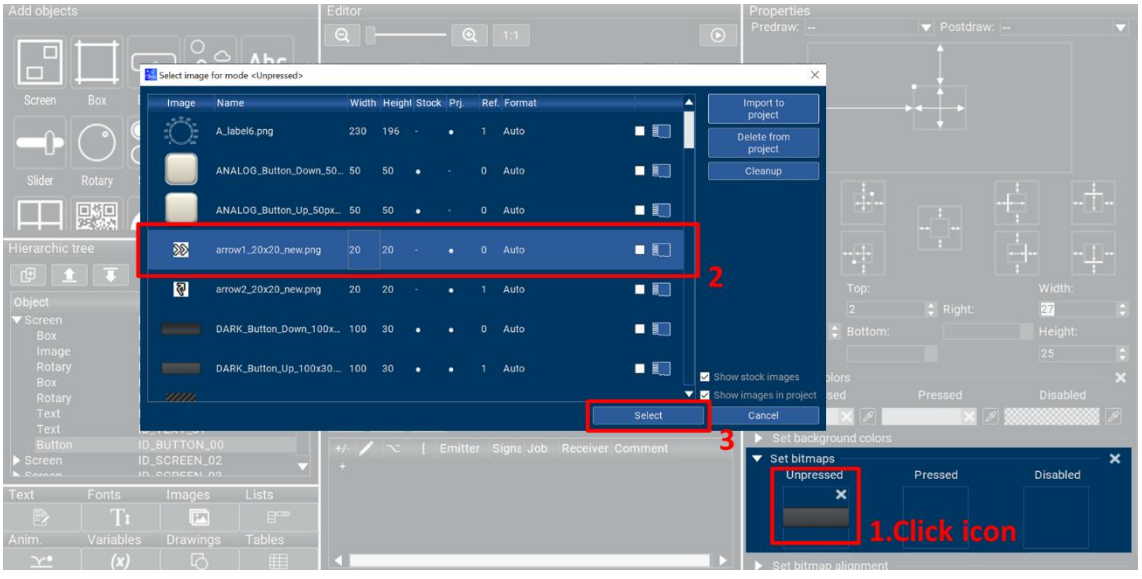


图 2-26 换页按钮外观设置

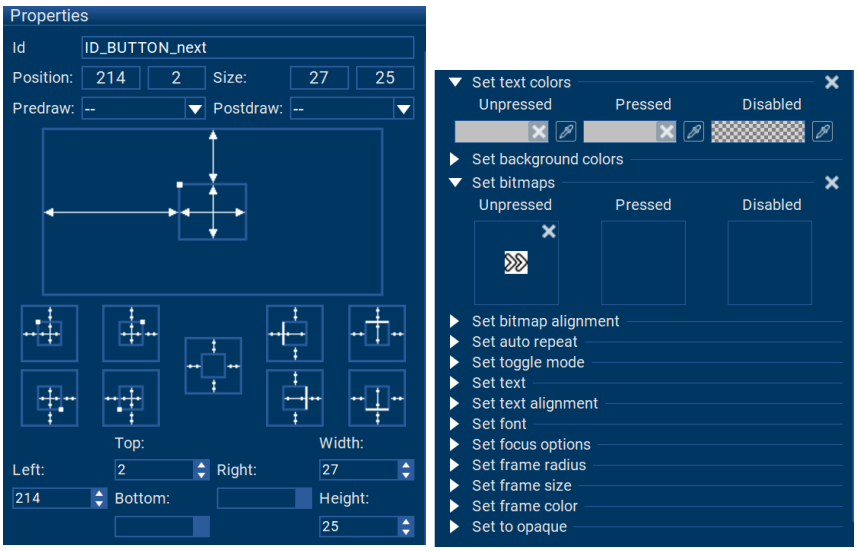


图 2-27 ID_BUTTON_next 属性设置

7. 在 Add objects 中，新增两个 TEXT 对象分别显示 Rotary 数值(ID_TEXT_TEMP)及单位符号(ID_TEXT_SIGN)，并依图 2-28~图 2-30 步骤设置其属性。

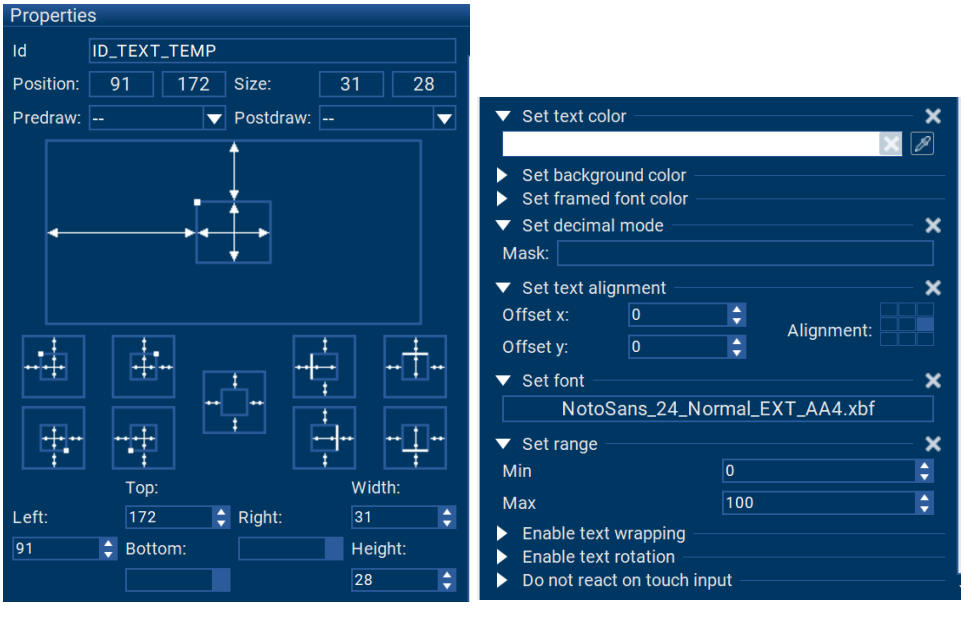


图 2-28 ID_TEXT_TEMP 属性设置

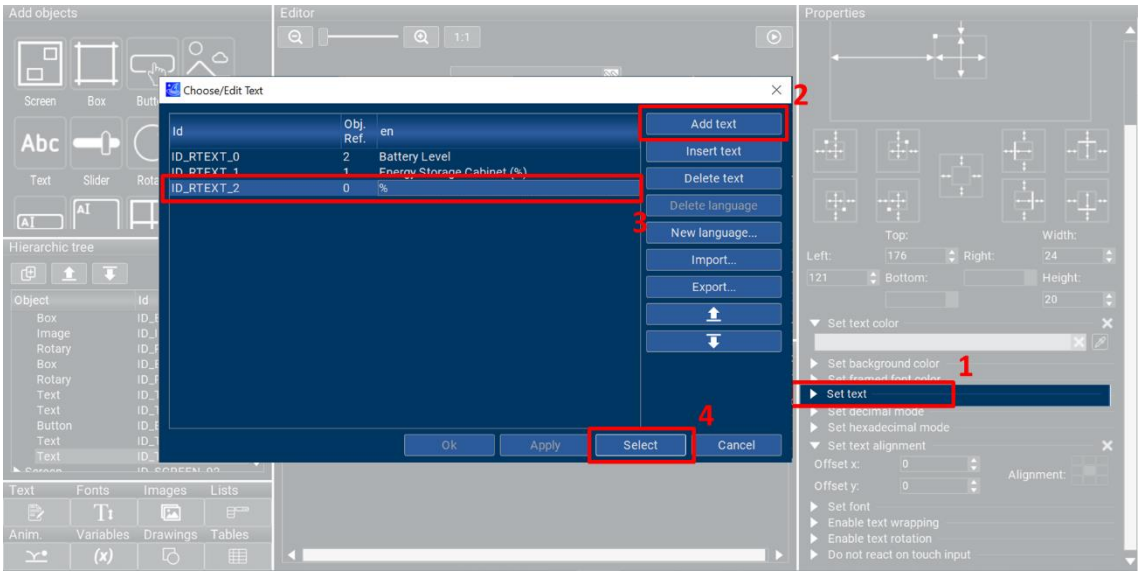


图 2-29 ID_TEXT_SIGN 文字设置

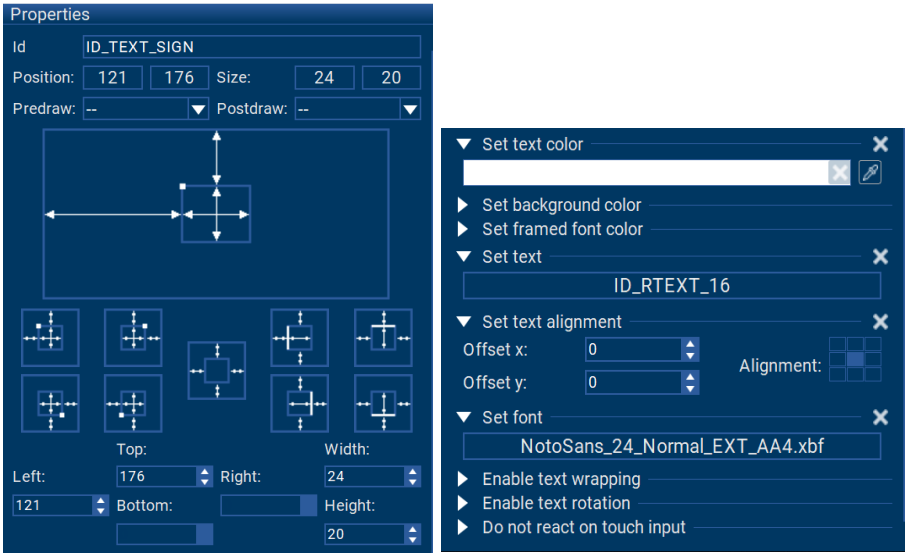


图 2-30 ID_TEXT_SIGN 属性设置

2.3.1.2 对象互动设置

对象设置完成画面如图 2-31 所示，接着要建立对象间的交互作用(interactions)，使操作对象时有相应行为显示。

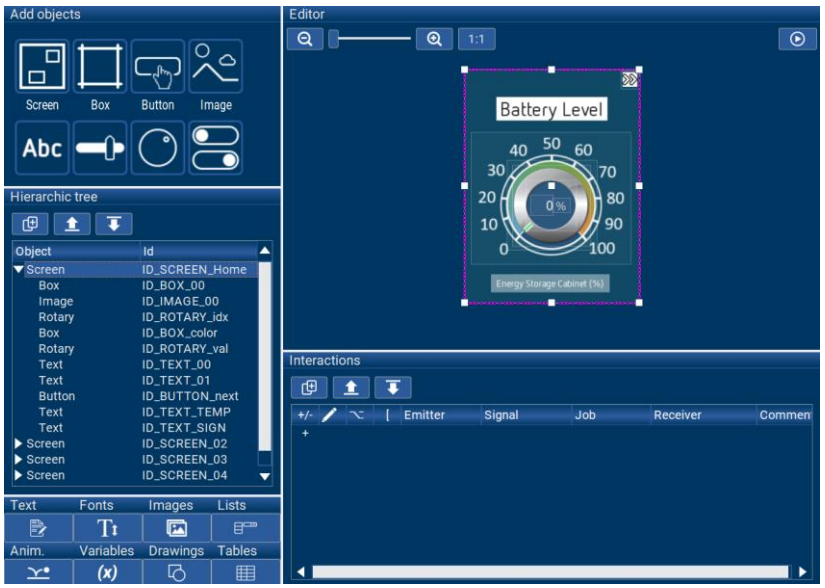


图 2-31 主画面设计完成图

1. 设置 Rotary(ID_ROTARY_val)与 TEXT(ID_TEXT_TEMP)对象的交互作用。

   1) 依图 2-32~图 2-35 步骤，透过 Variables 新增变量使用。

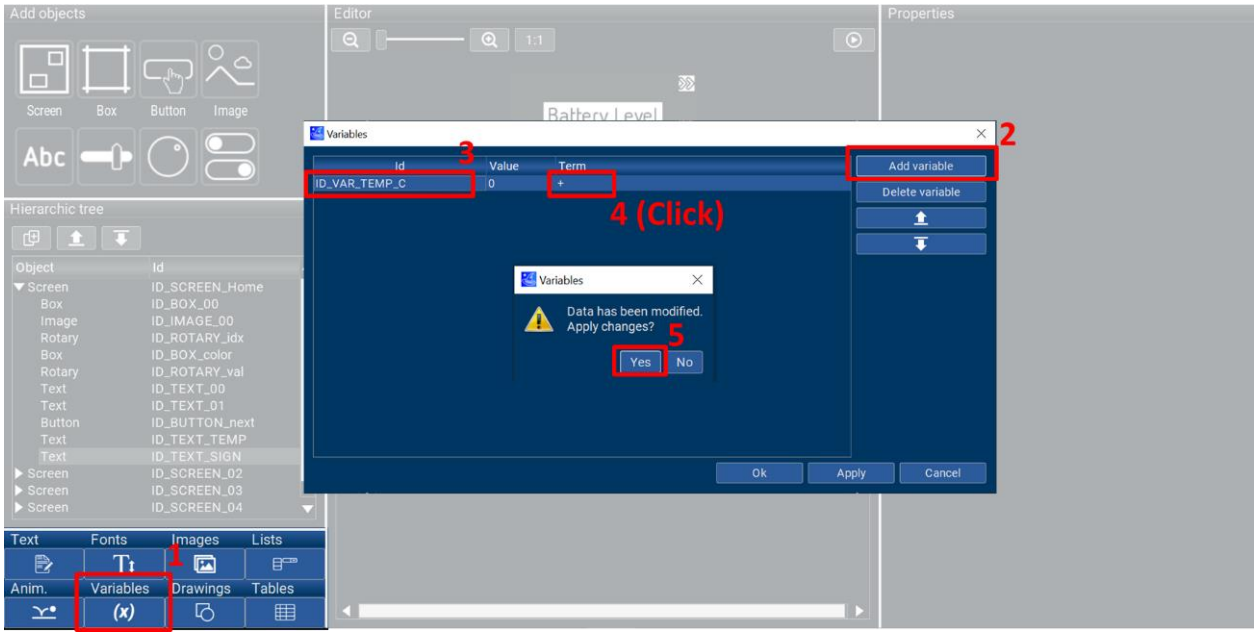


图 2-32 新增变量 ID_VAR_TEMP_C

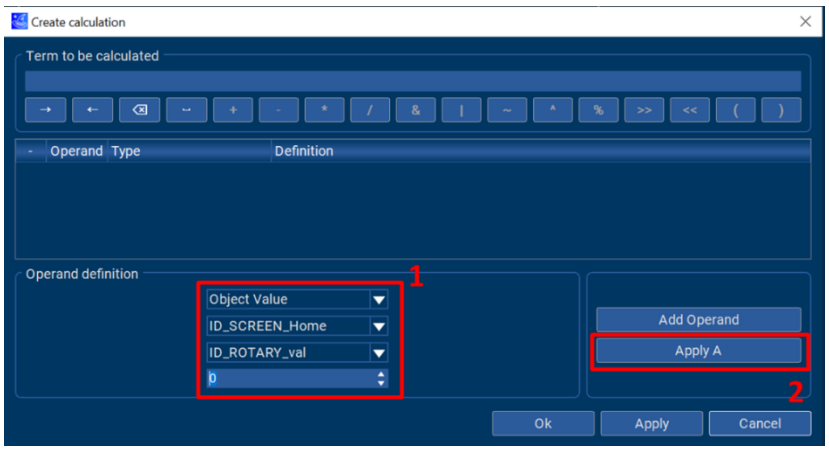


图 2-33 设置 ID_VAR_TEMP_C 计算值 1

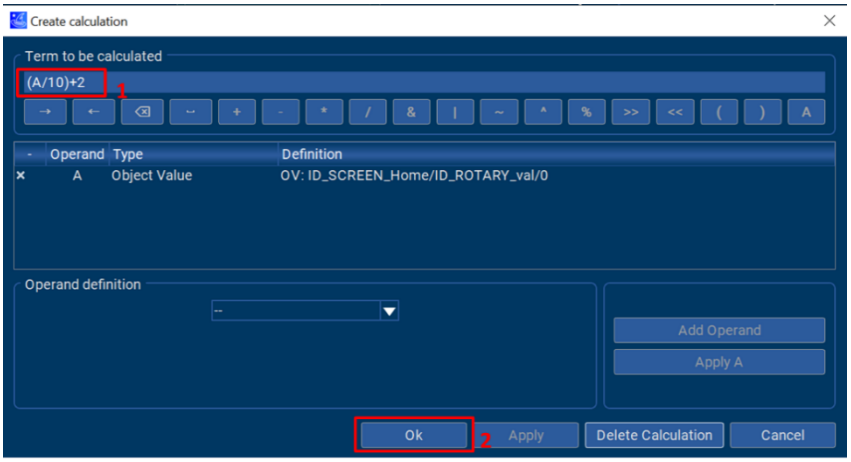


图 2-34 设置 ID_VAR_TEMP_C 计算值 2

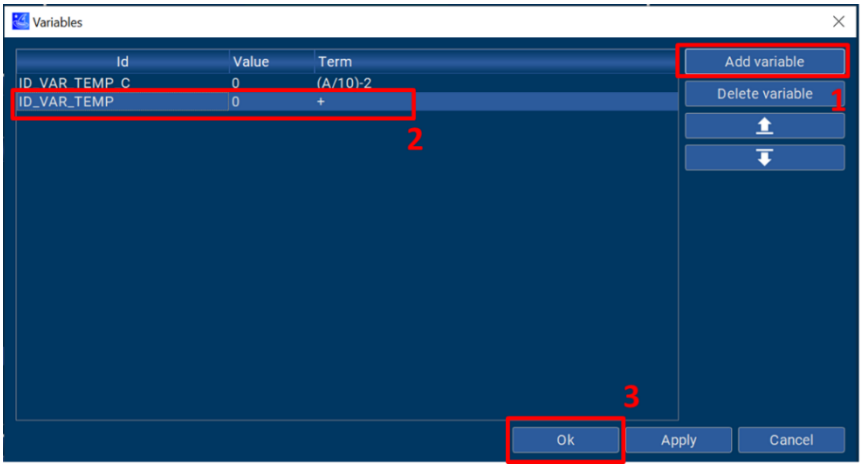


图 2-35 新增变量(ID_VAR_TEMP)

2) 在 interactions 字段设置 Rotary 与 TEXT 交互作用，令 Rotary 转动同时改变 TEXT 显示值。

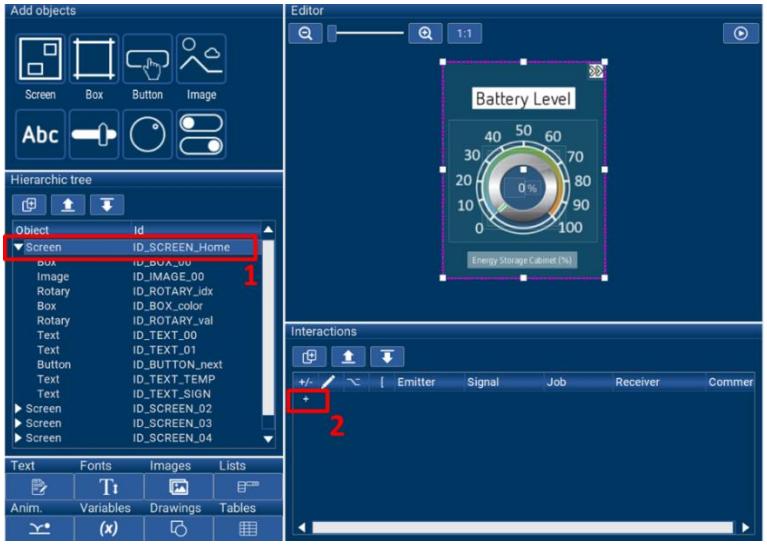


图 2-36 新增 interactions 项目

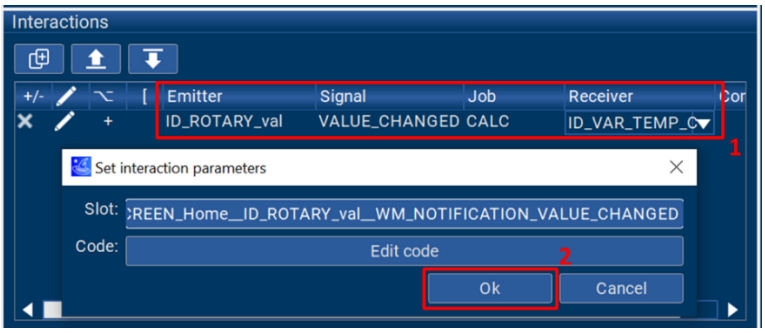


图 2-37 将 Rotary 数值映像到变量 ID_VAR_TEMP_C

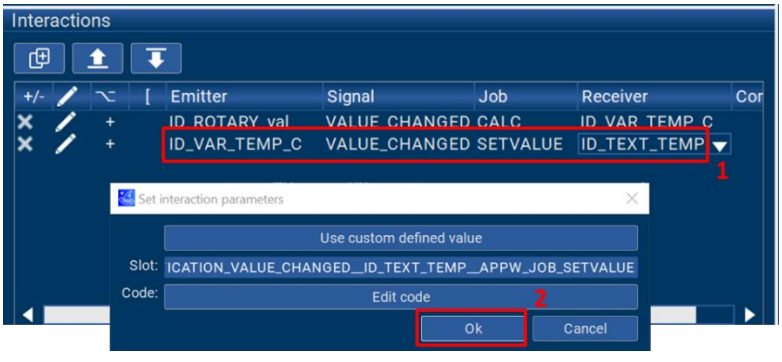


图 2-38 赋予 ID_VAR_TEMP_C 数值给对象 ID_TEXT_TEMP

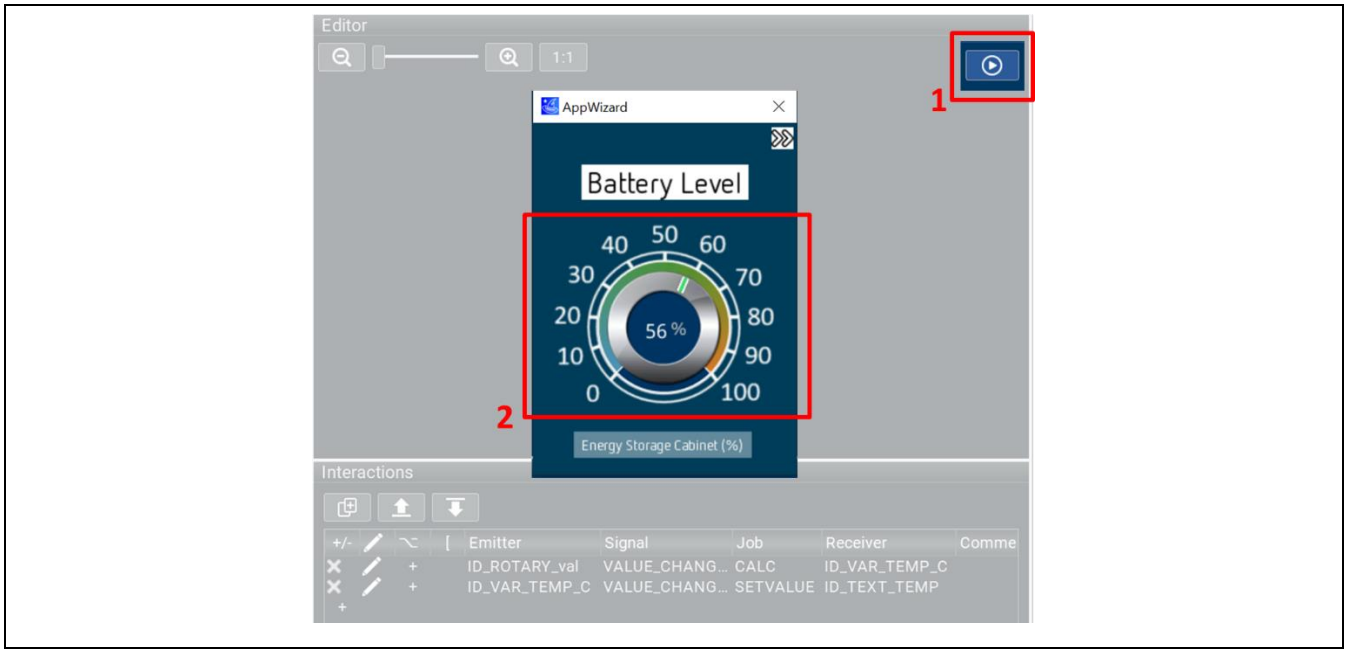


图 2-39 透过 Start Play Mode 确认 interaction 行为是否正确

2. 设置 Rotary(ID_ROTARY_val)数值对应颜色，使数值变化时影响 Box (ID_BOX_color)色彩。

   1) 依图 2-40~图 2-43 步骤，新增颜色(ID_VAR_COLOR_BLUE、ID_VAR_COLOR_GREEN、ID_VAR_COLOR_ORANGE)、变色开关(ID_VAR_COLOR_0、ID_VAR_COLOR_1)与色彩变化指针(ID_VAR_COLOR_POS)的初始值。

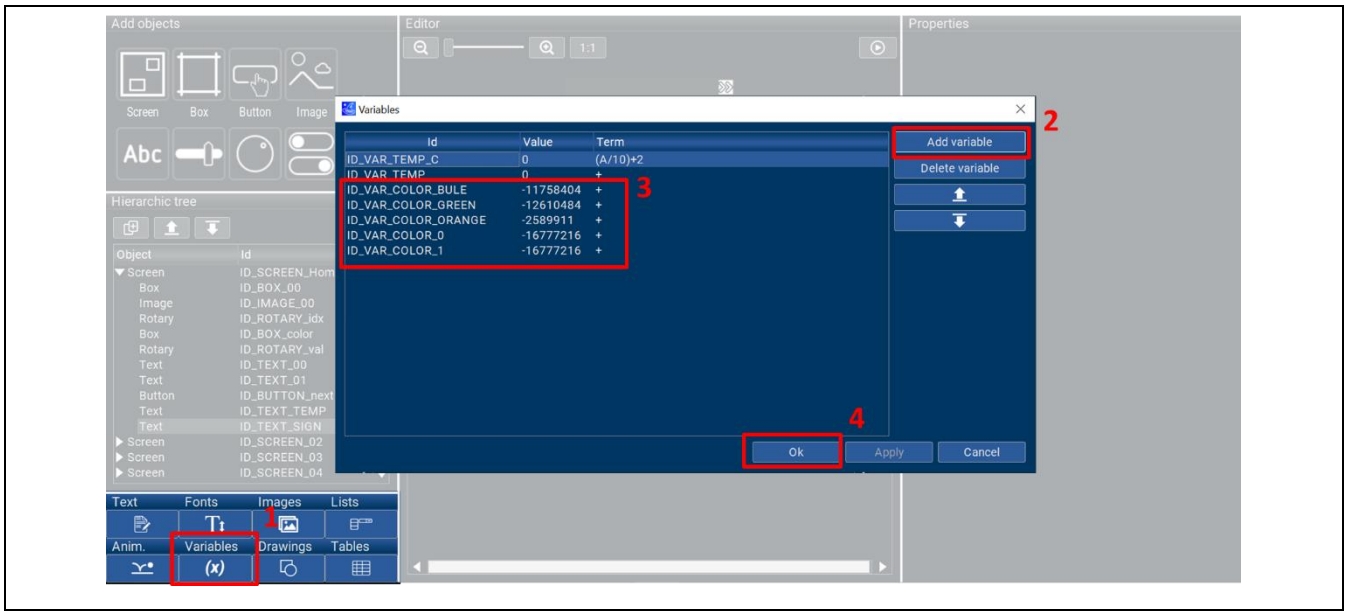


图 2-40 新增颜色与变色开关变量初始值

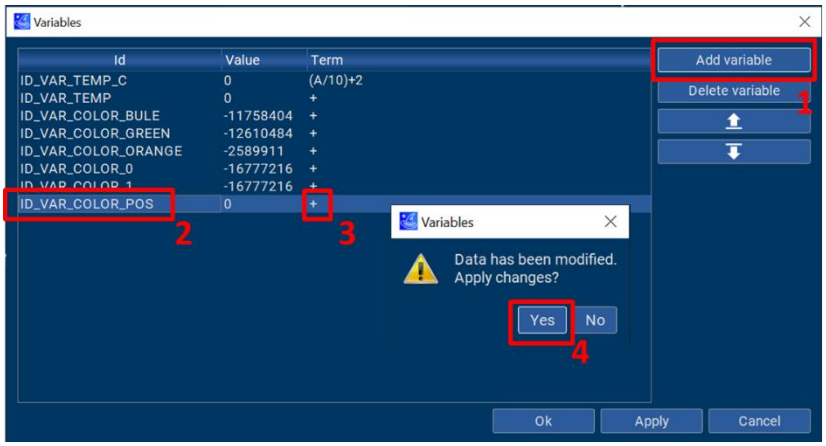


图 2-41 新增色彩变化指针变量

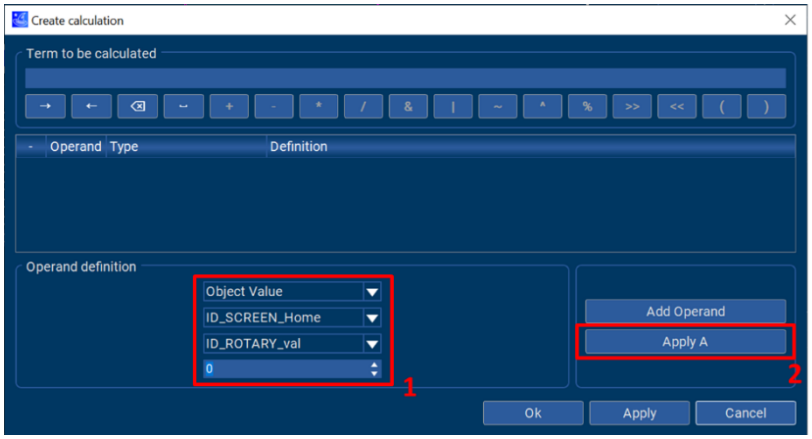


图 2-42 色彩指针变量设定 1

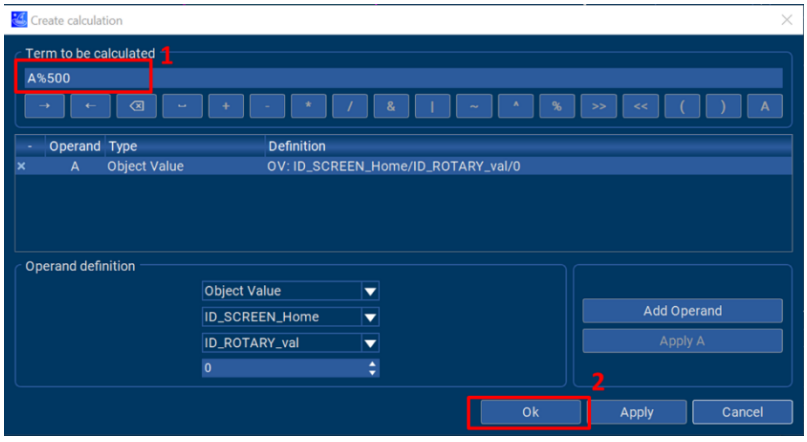


图 2-43 色彩指针变量设定 2

2) 在 interactions 字段设置 Rotary 与颜色变化的交互作用，让 Rotary 转动同时改变显示颜色。

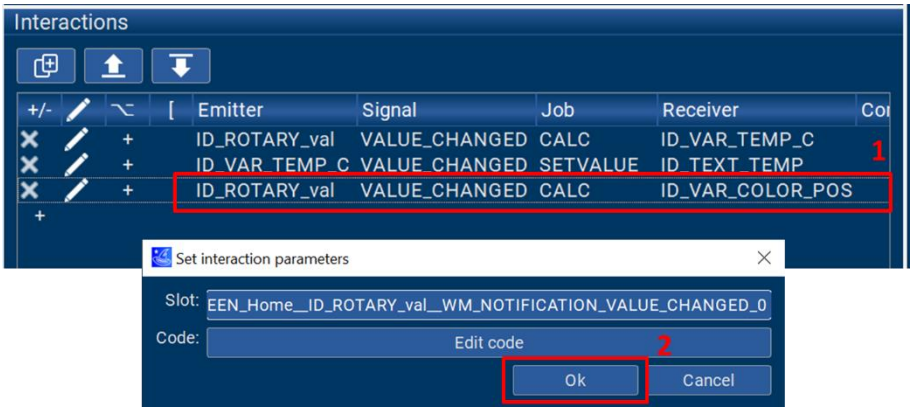


图 2-44 将 Rotary (ID_Rotary_val)数值赋予给变量(ID_VAR_COLOR_POS)

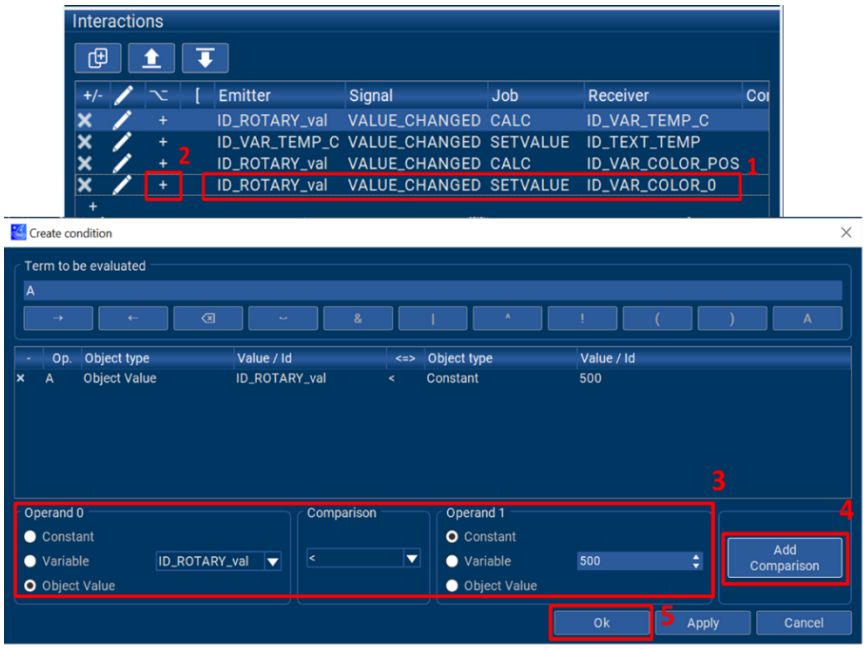


图 2-45 色彩切换设置 1

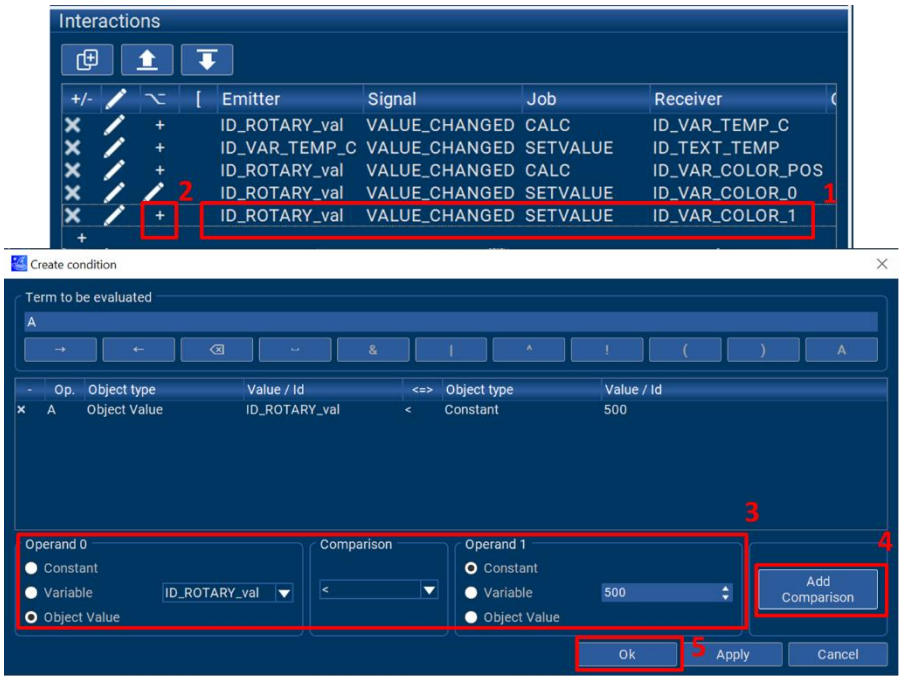


图 2-46 色彩切换设置 2

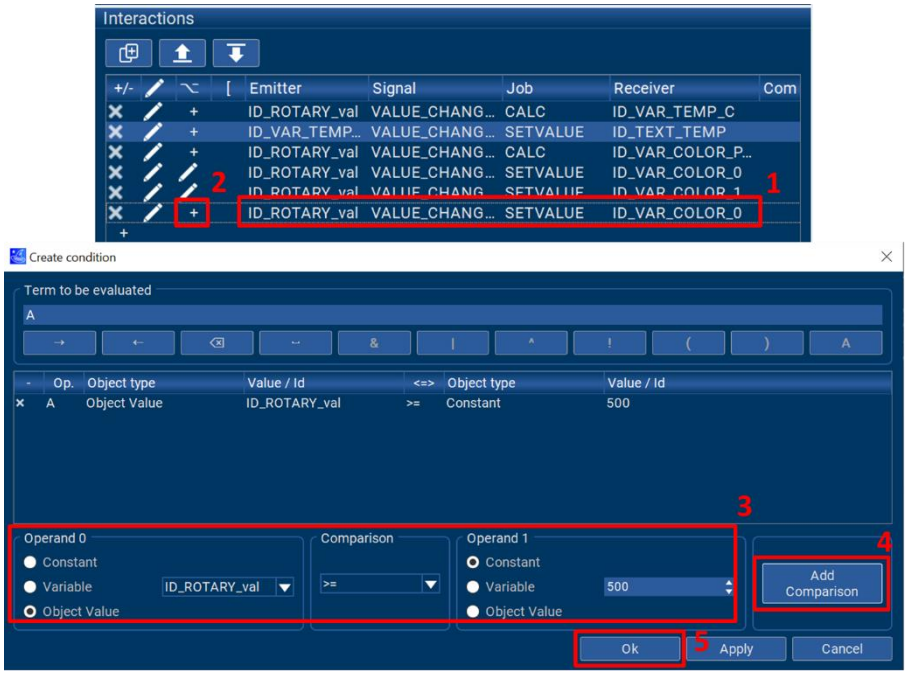


图 2-47 色彩切换设置 3

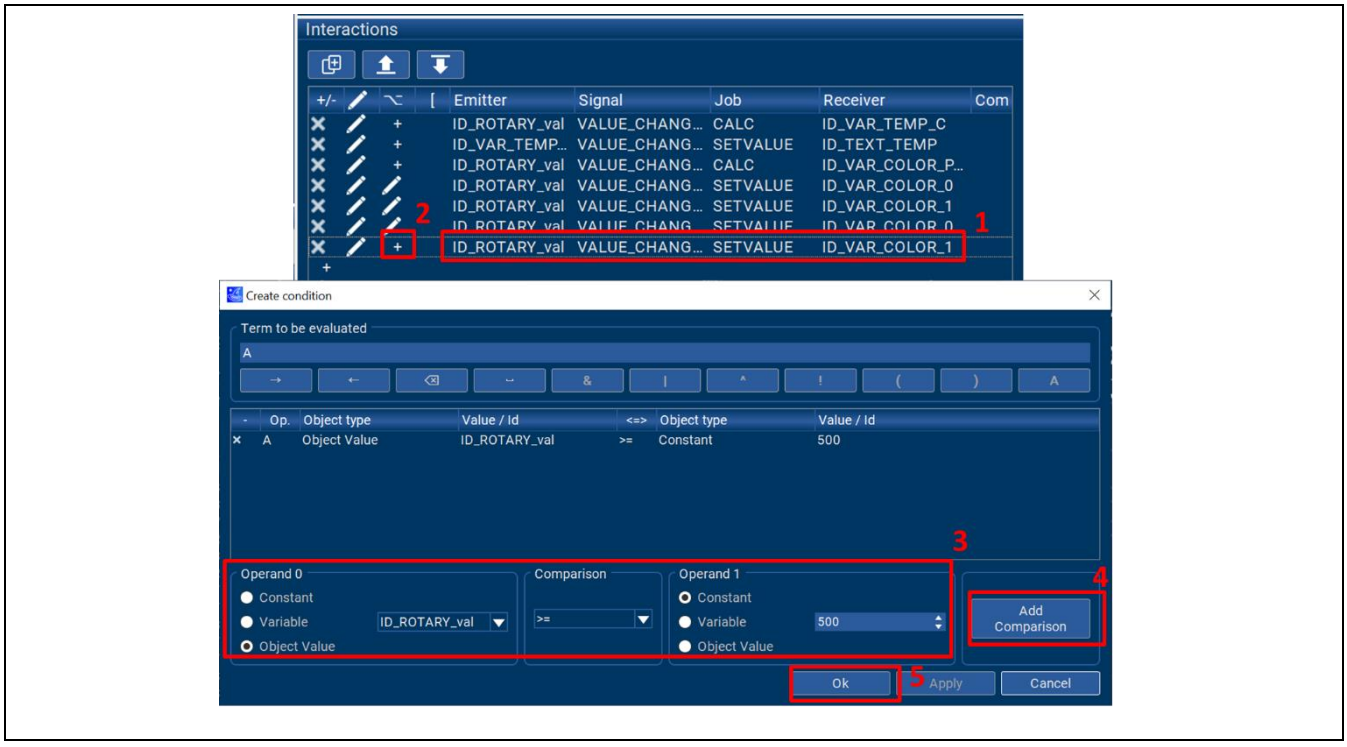


图 2-48 色彩切换设置 4

3) 依图 2-49~图 2-52 的步骤设置每个切换动作的起始变量值。

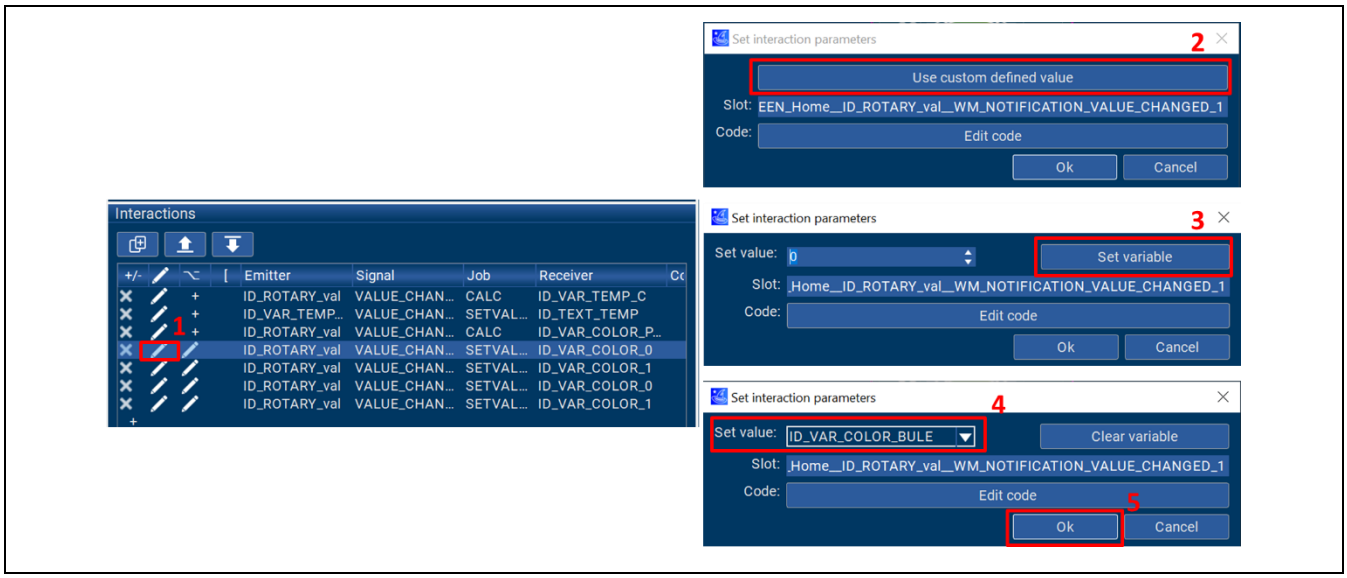


图 2-49 色彩变换初始值 1

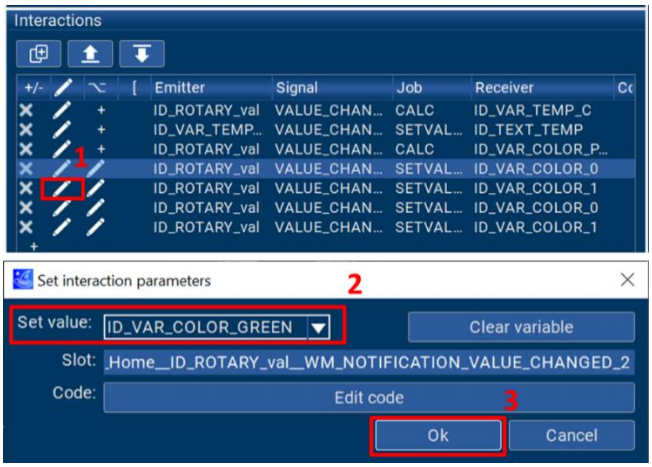


图 2-50 色彩变换初始值 2

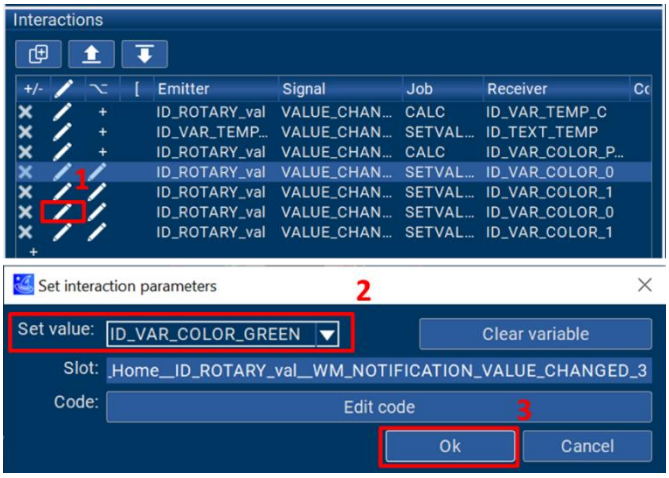


图 2-51 色彩变换初始值 3

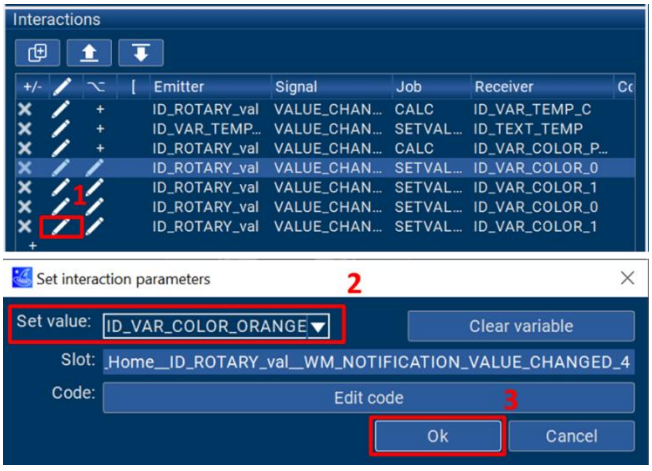


图 2-52 色彩变换初始值 4

4) 依图 2-53~图 2-58 的步骤新增 Drawings 对象，设置将色彩随数值切换显示的功能。

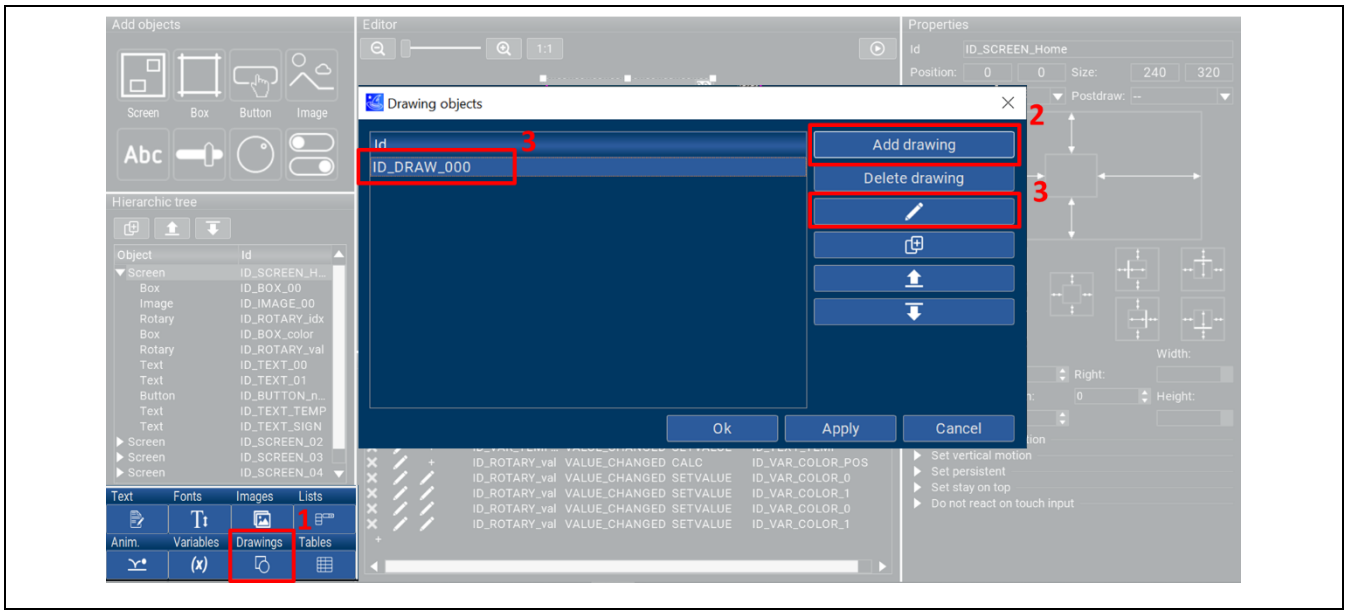


图 2-53 新增 Drawings 物件(ID_DRAW_000)

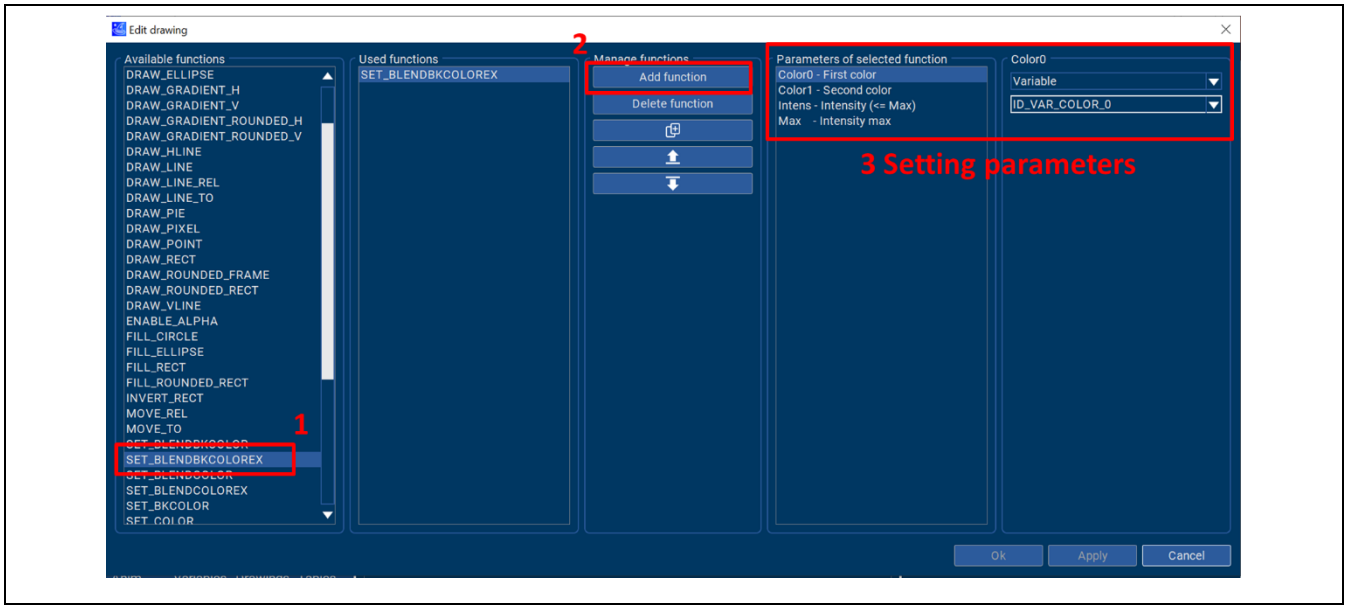


图 2-54 在 ID_DRAW_000 内新增着色 function

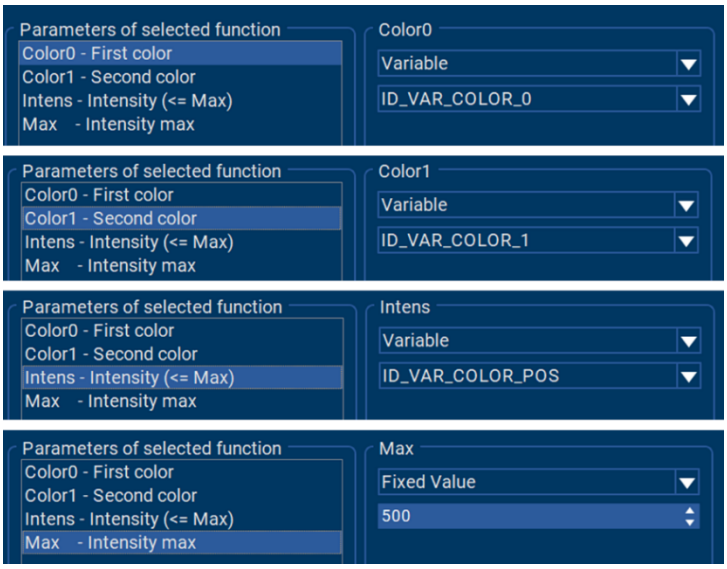


图 2-55 着色 function 各项参数设定

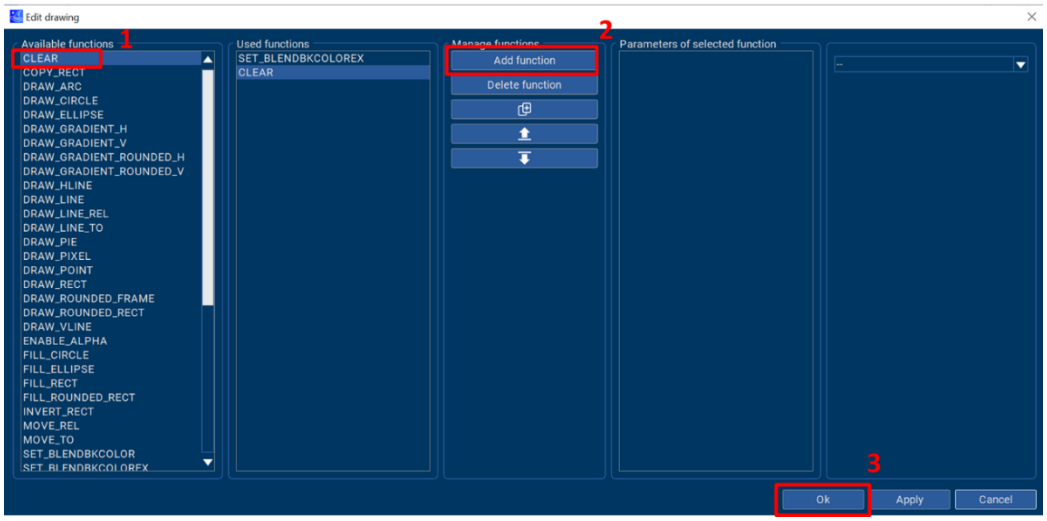


图 2-56 在 ID_DRAW_000 内新增 clear function

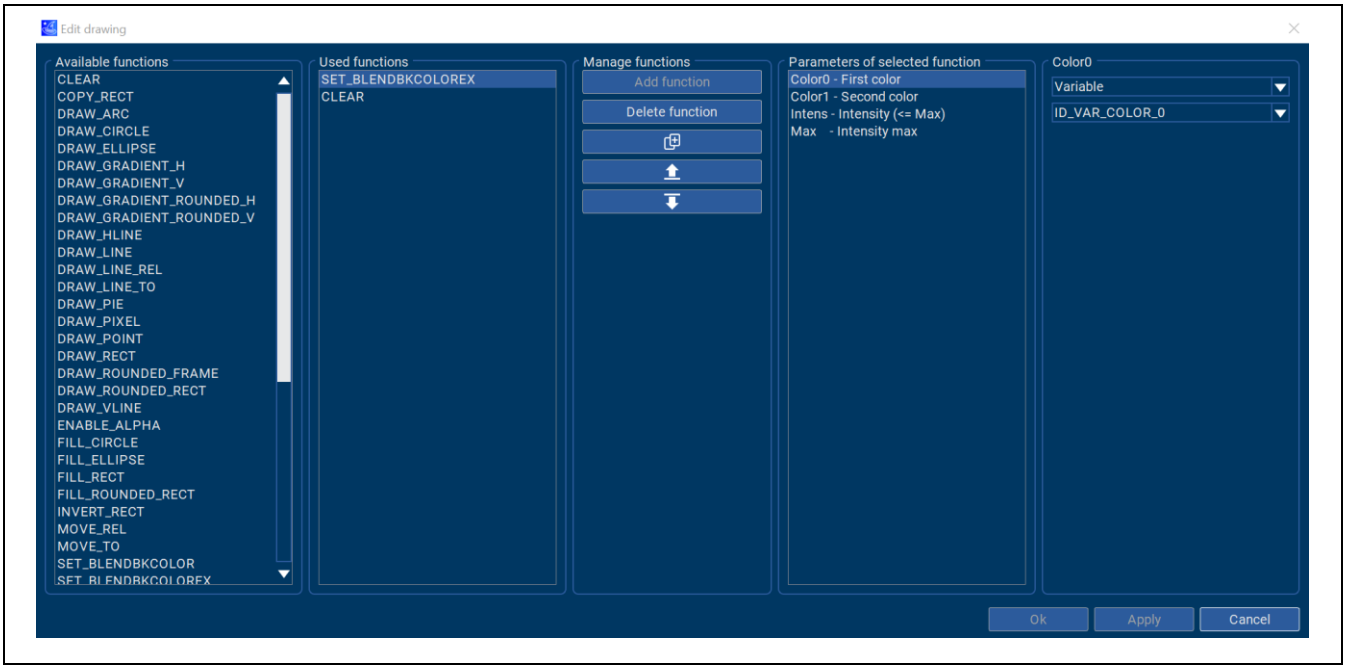


图 2-57 ID_DRAW_000 设置总览

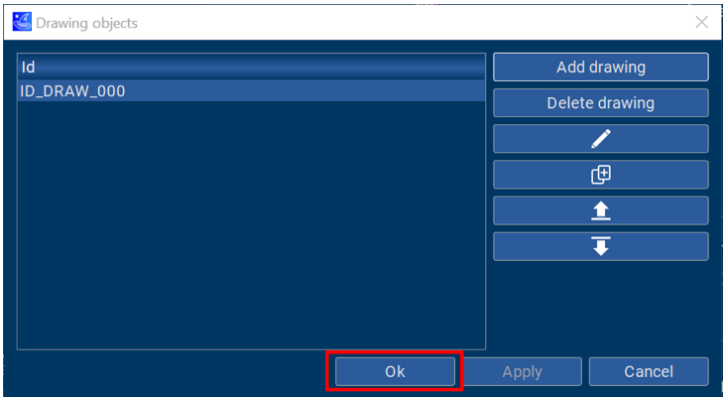


图 2-58 储存 ID_DRAW_000 设置

5) 点选对象 ID_BOX_color，在属性字段将 Postdraw 设置为 ID_DRAW_000。

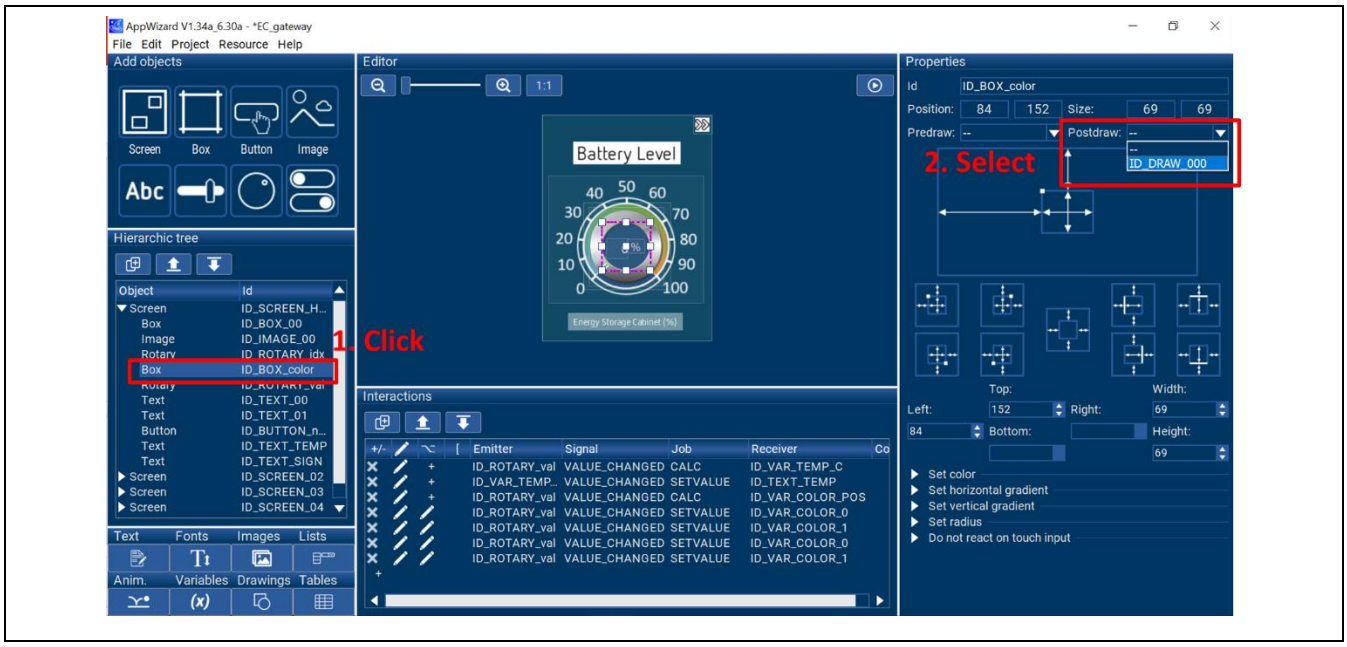


图 2-59 设置 ID_BOX_color 的 Postdraw 属性

6) 完成上述动作后， ID_BOX_color 色彩便会随 ID_Rotary_val 数值变化。

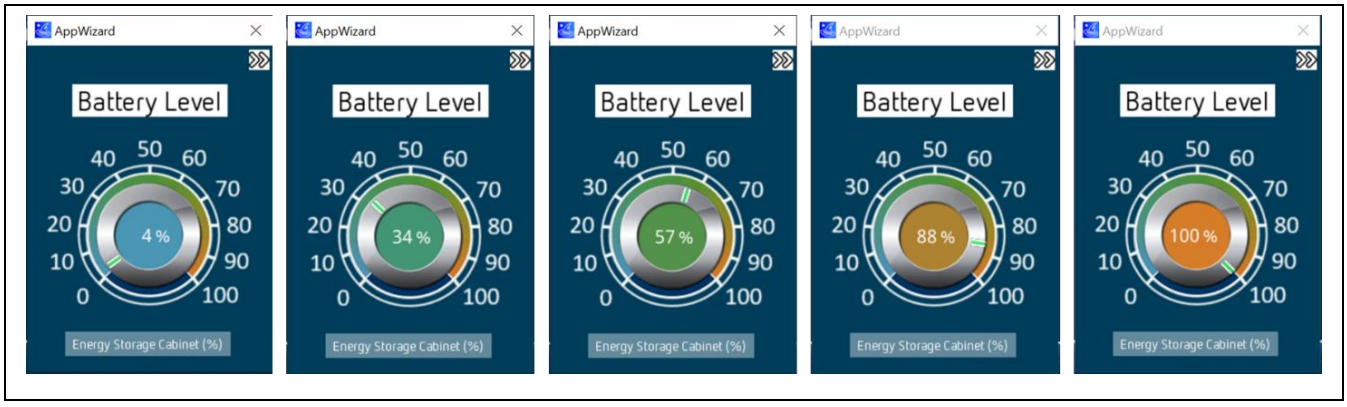


图 2-60 透过 Play Mode 预览 ID_BOX_color 变化

3. 在 interactions 字段设置按钮的交互作用，使点选按钮可以切换到下一页。

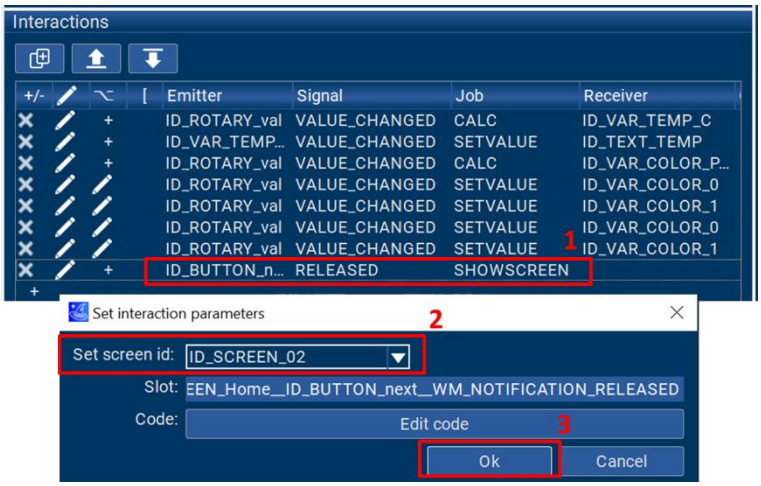
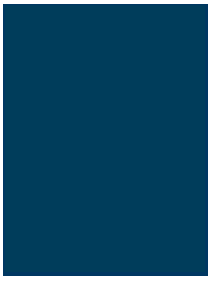


图 2-61 换页按钮(ID_Button_next)设置

2.3.2 光伏逆变器设计(页面二)

光伏逆变器是种可以将太阳能光电(PV)太阳能板产生的可变直流电压转换为市电频率交流电(AC)的变频器，可以回馈回商用输电系统或供离网的电网使用等，会有转换率控制及显示的需求，也可就此设计应用为其他类型逆变器的用户接口，此设计所用对象如表 2-3 所示。

New Energy Gateway Description	Application GUI	Widget Type
背景图 240x320		Box
数值转动旋钮与 Maker Rotary		Rotary
旋钮数值条背景 Gauge_bar		Gauge

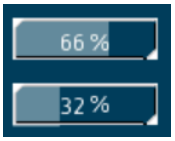
旋钮数值条		Gauge
轴承切换开关		Switch
轴承转换率显示		Progbar
轴承数值条		Slider
标题文字		Text
旋钮数值显示		
上一页按钮		Button
下一页按钮		

表 2-3 太阳能变频器设计对象

2.3.2.1 GUI 对象配置

1. 点选 ID_SCREEN_02 开始编辑光伏逆变器画面，透过 Add objects 新增 TEXT 对象 (ID_TEXT_title)设置页面标题。

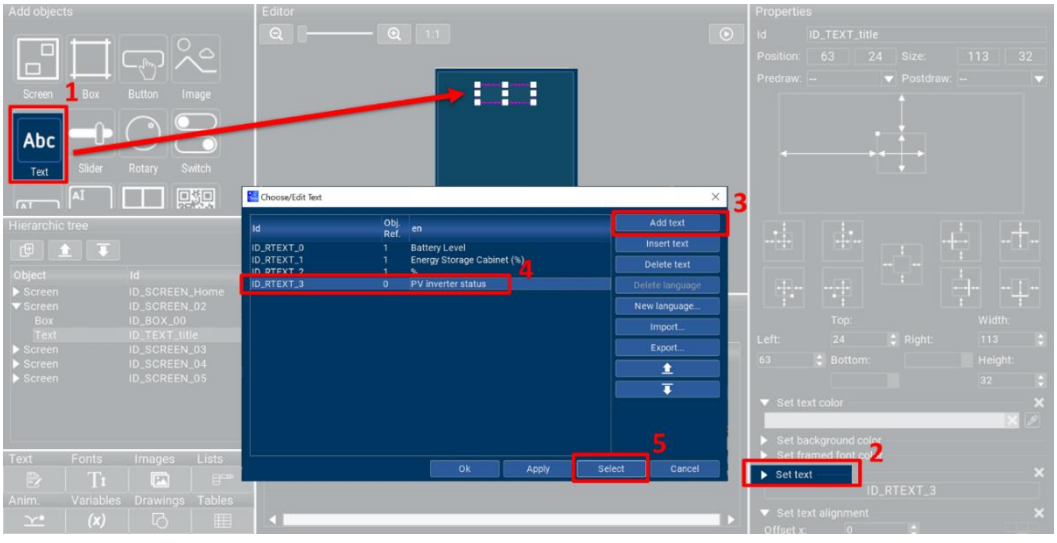


图 2-62 设置标题文字

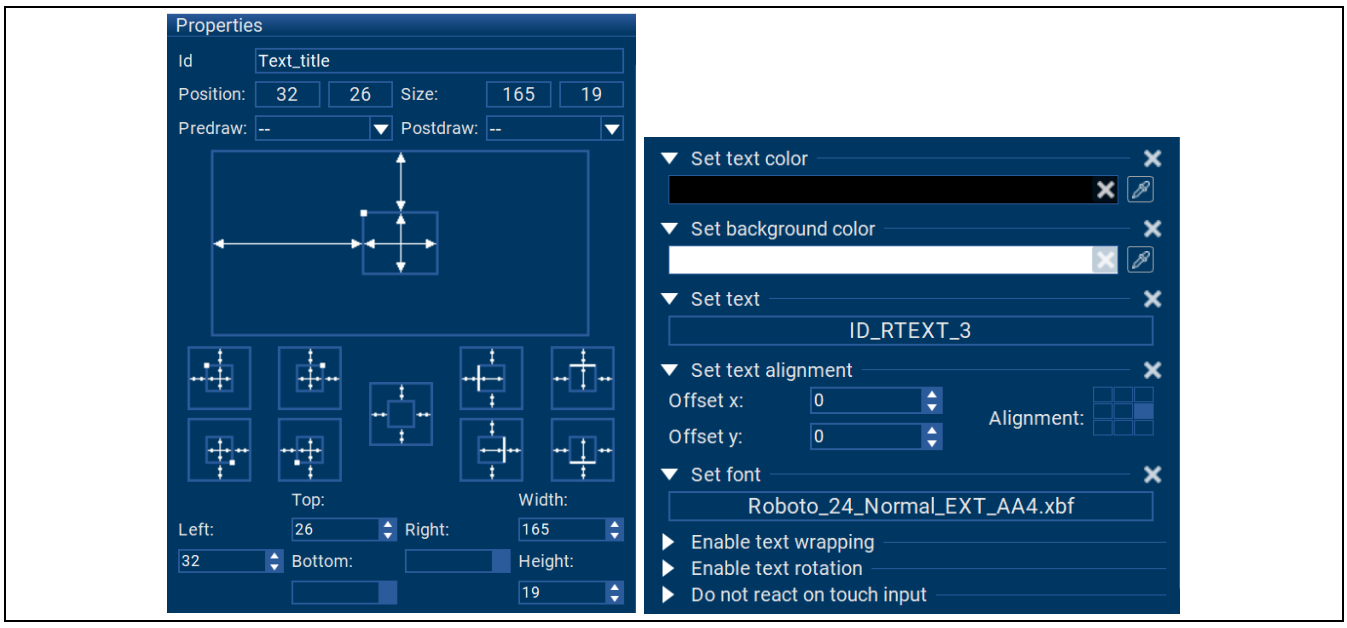


图 2-63 ID_TEXT_title 属性设置

2. 透过 Add objects 新增 Gauge 对象(ID_Gauge_BG)作为变化数值的背景。

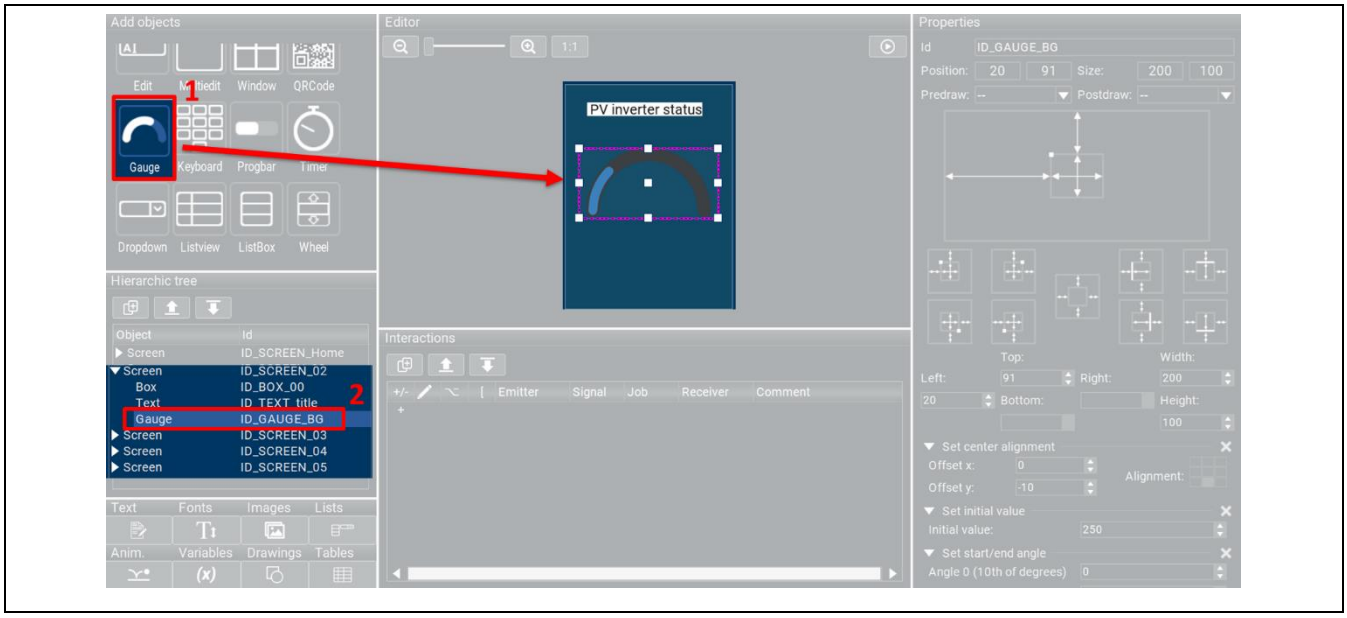


图 2-64 新增物件 ID_Gauge_BG

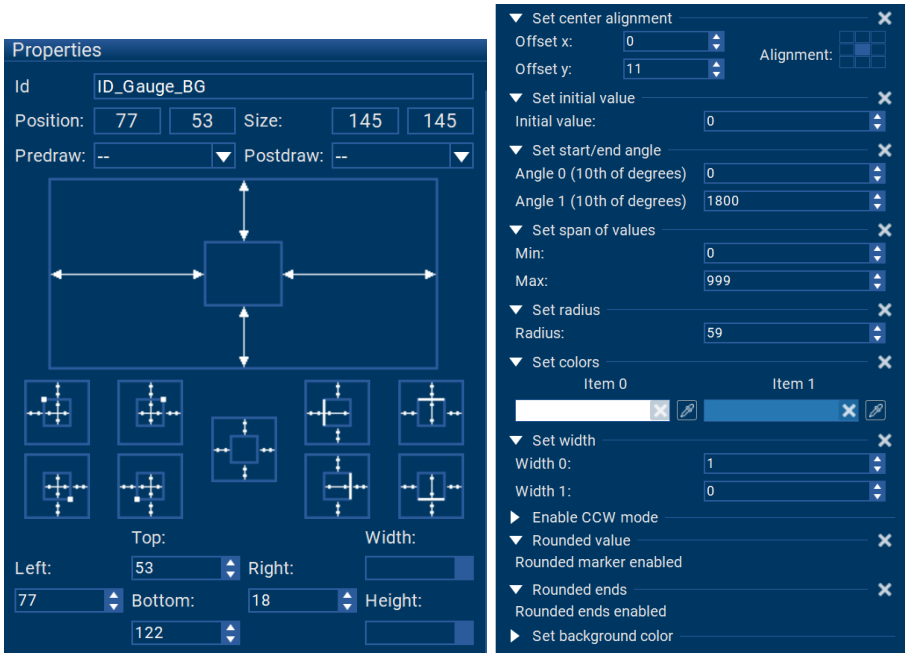


图 2-65 ID_Gauge_BG 属性设置

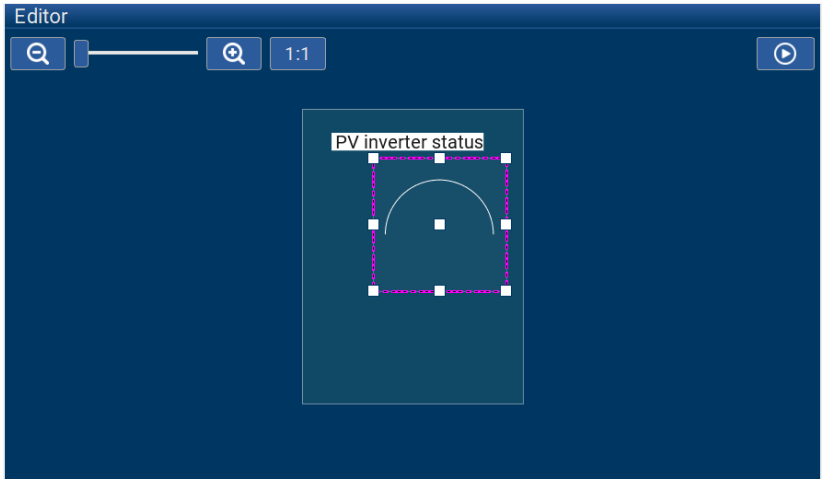


图 2-66 ID_Gauge_BG 设置画面

3. 新增第二个 Gauge(ID_Gauge_BAR)对象作为数值显示。

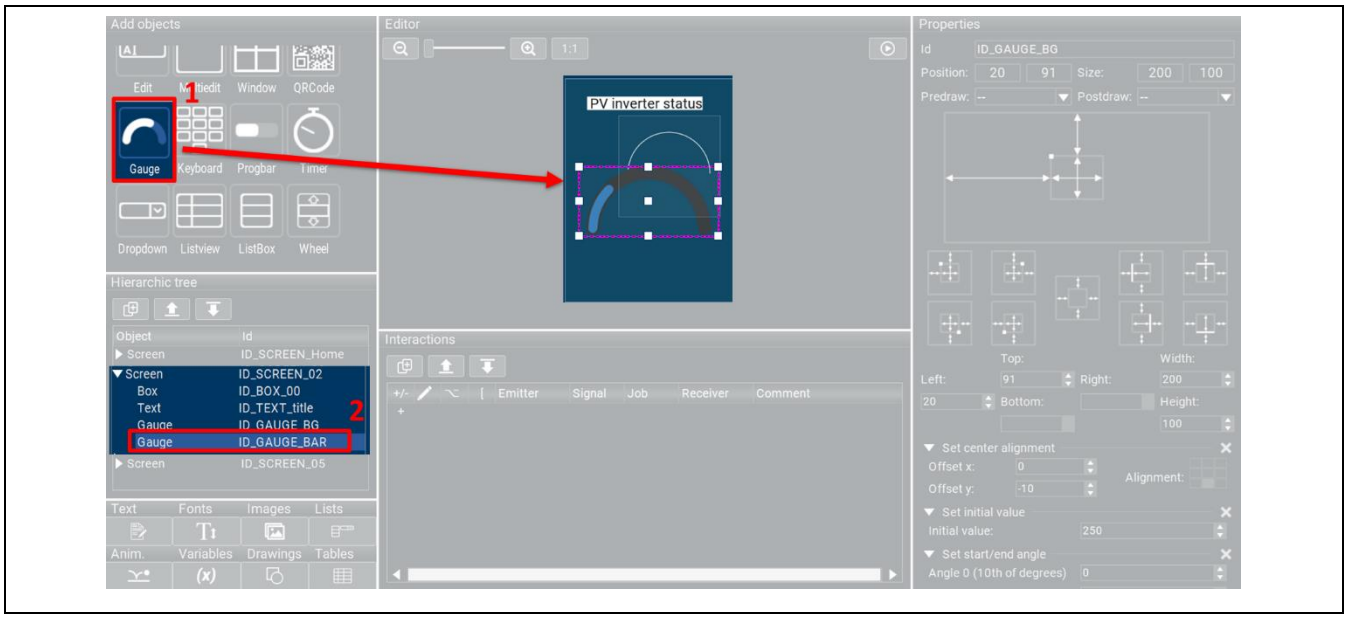


图 2-67 新增物件 ID_Gauge_BAR

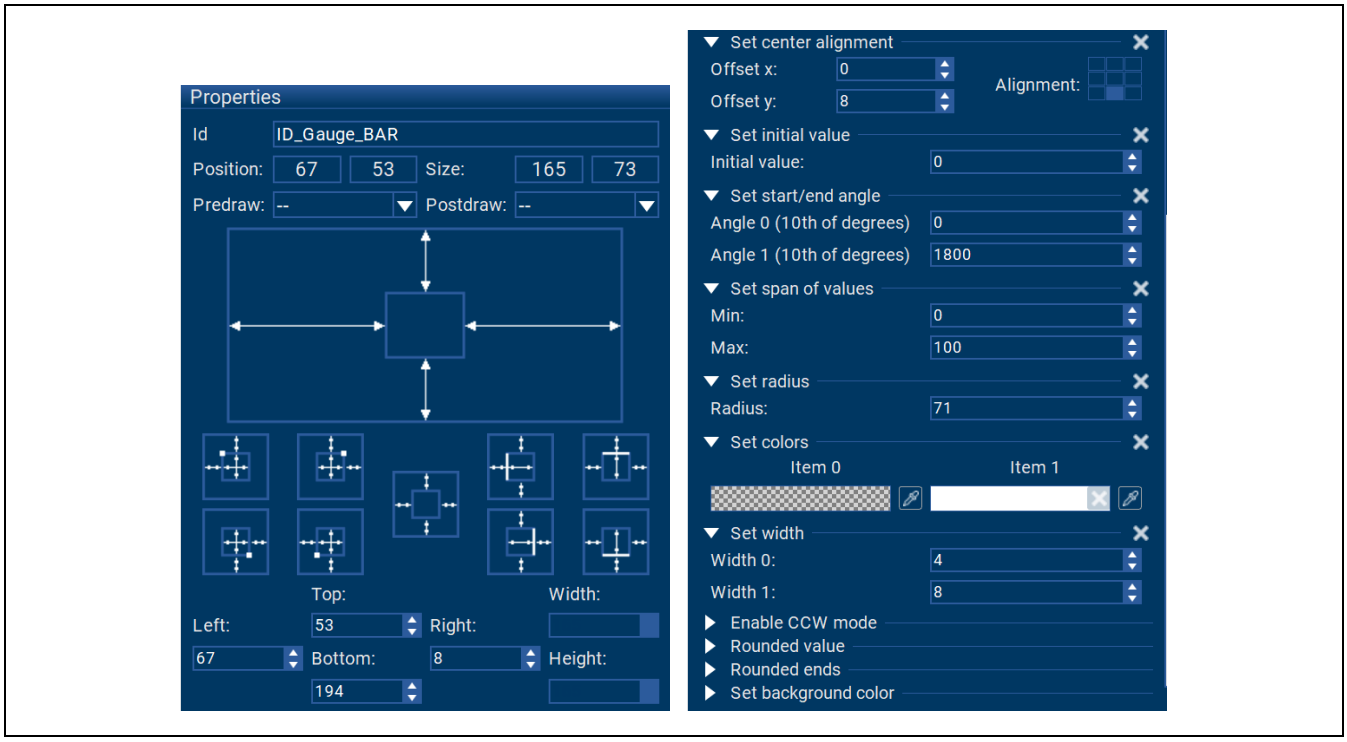


图 2-68 ID_Gauge_BAR 属性设置

4. 新增 Rotary 对象(ID_Rotary)做数值转动，并选择图片 Rotary_Background.png、Rotary_Marker.png 设置为背景及指标。

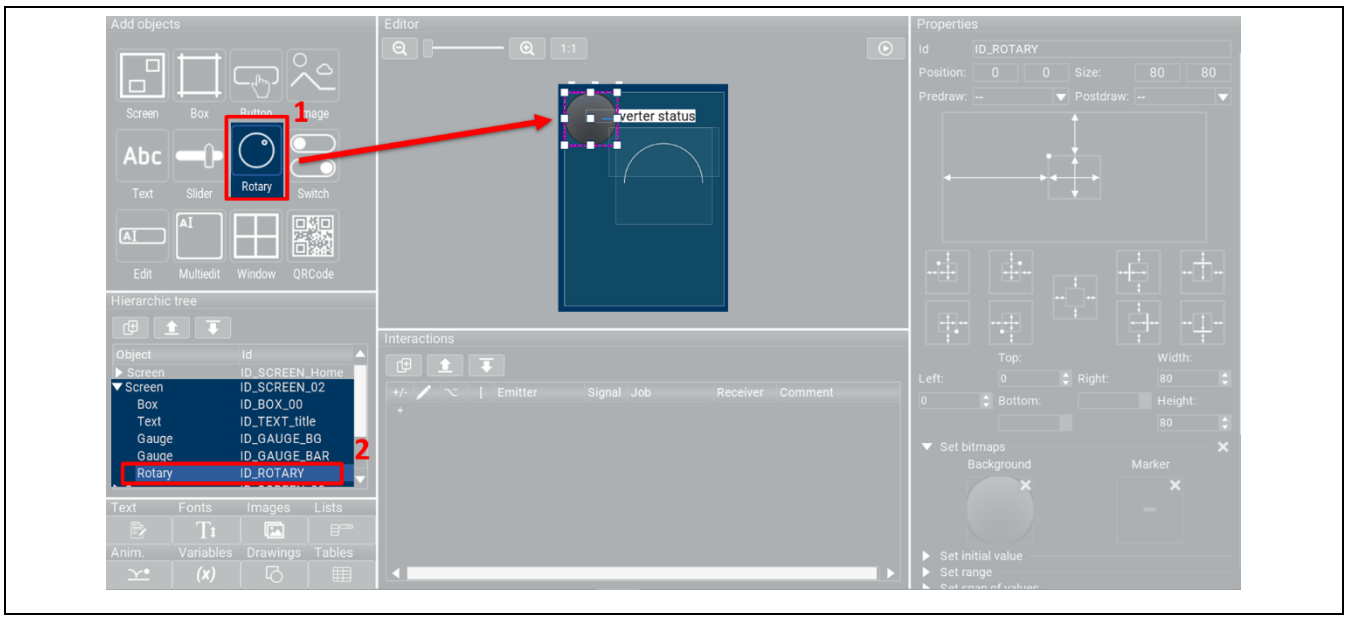


图 2-69 新增 Rotary 物件

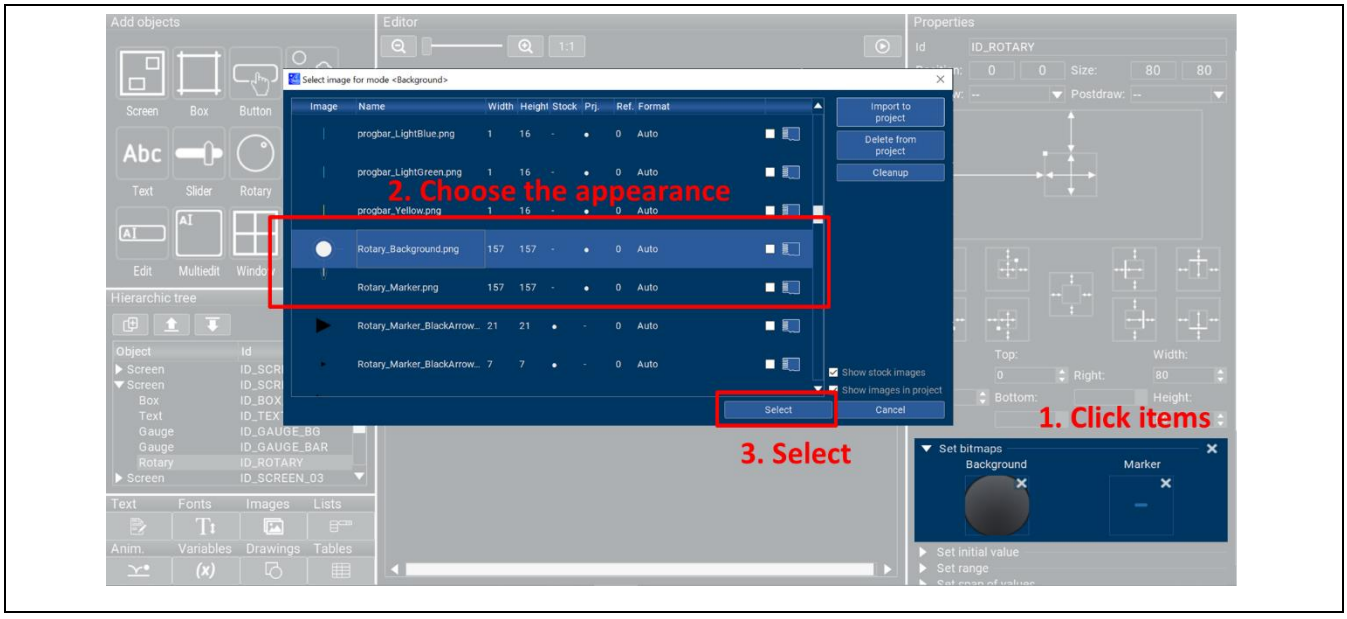


图 2-70 ID_Rotary 外观设置

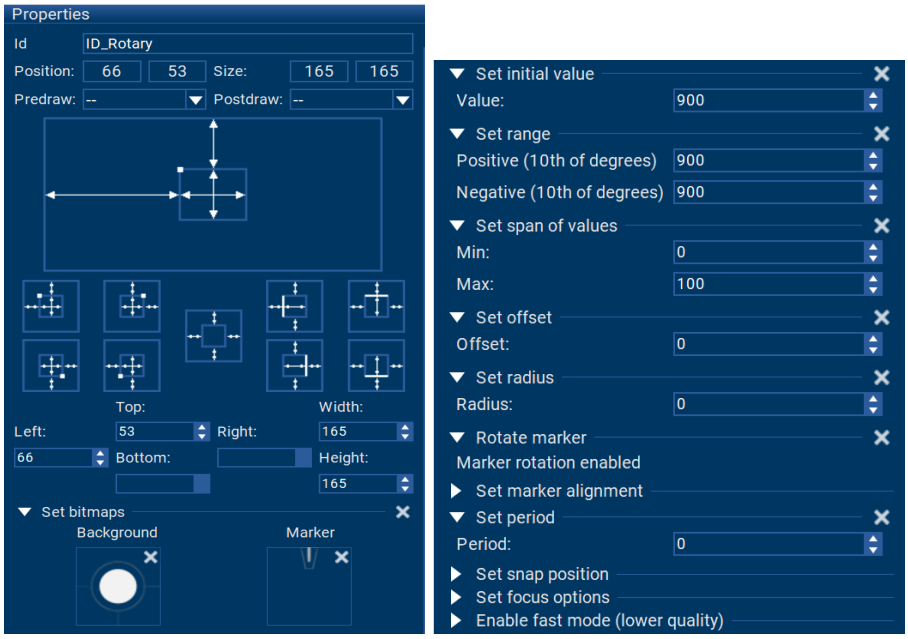


图 2-71 ID_Rotary 属性设置与外观

5. 新增一个 TEXT 对象(TEXT_value)显示 Rotary(ID_Rotary)数值，并依图 2-72 设置属性。

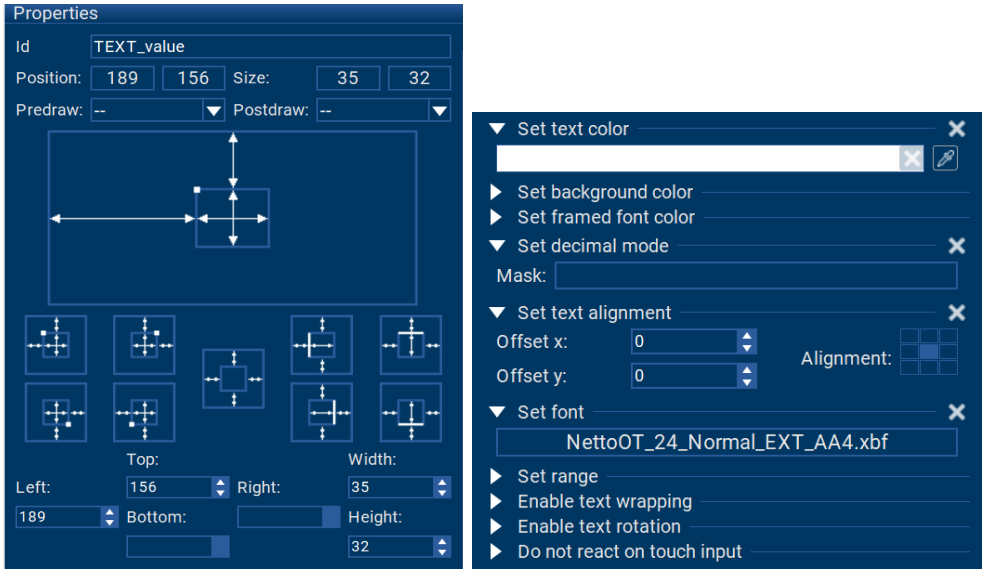


图 2-72 TEXT_value 属性设置

6. 新增两个 Switch 对象(SWITCH_01、SWITCH_02)作为功能开关，并将图片 Swtich_Pressed_83x22.png、Swtich_Unpressed_83x22.png 设置为 BG-Left 与 BG-Right 显示画面，并依图 2-74~图 2-77 设置属性。

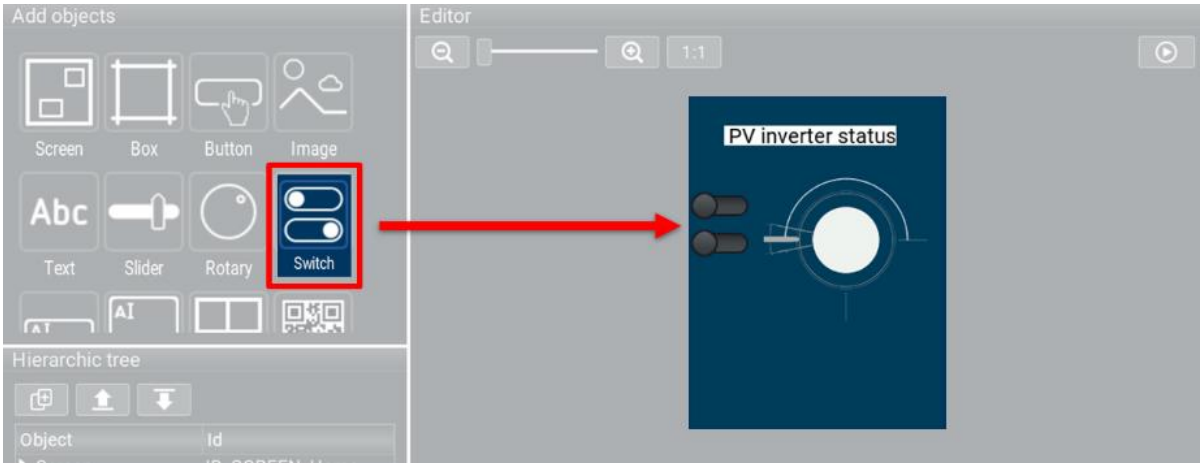


图 2-73 新增 Switch 物件

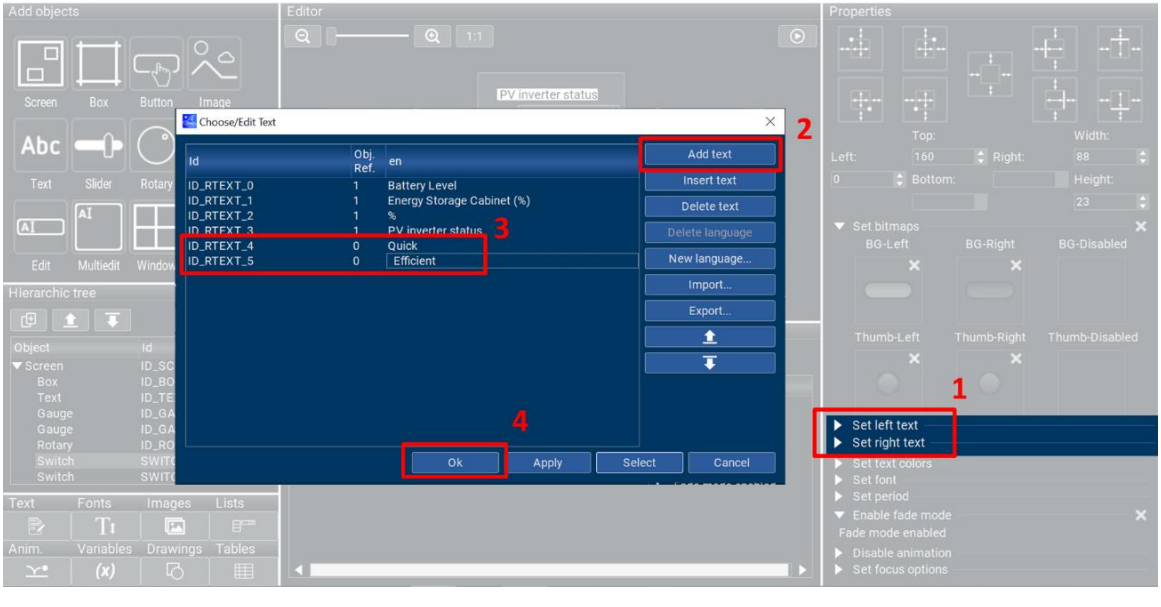


图 2-74 新增 Switch 显示文字

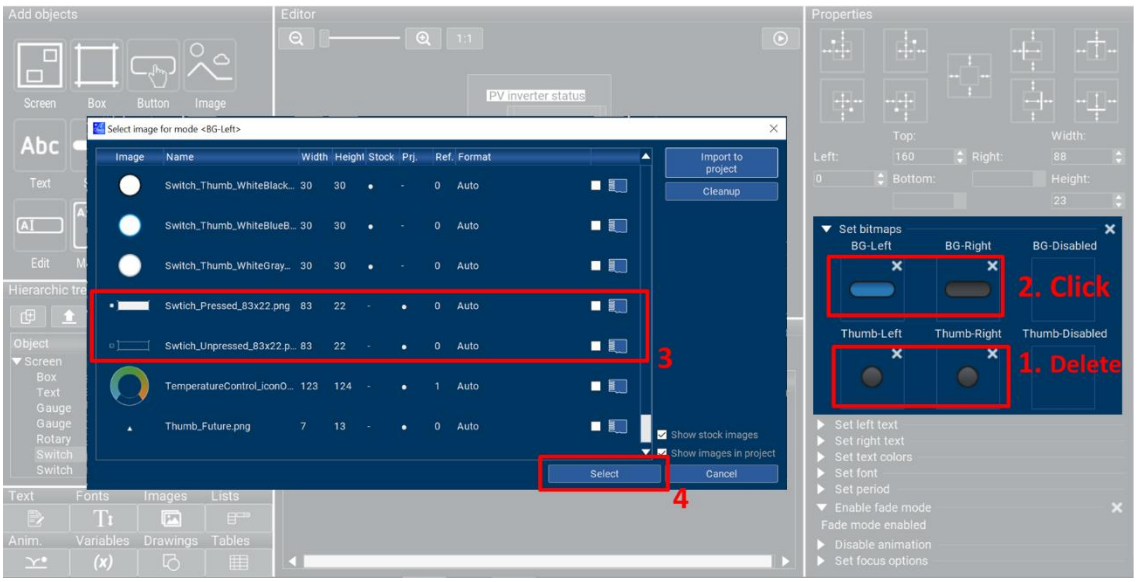


图 2-75 设定 Switch 外观

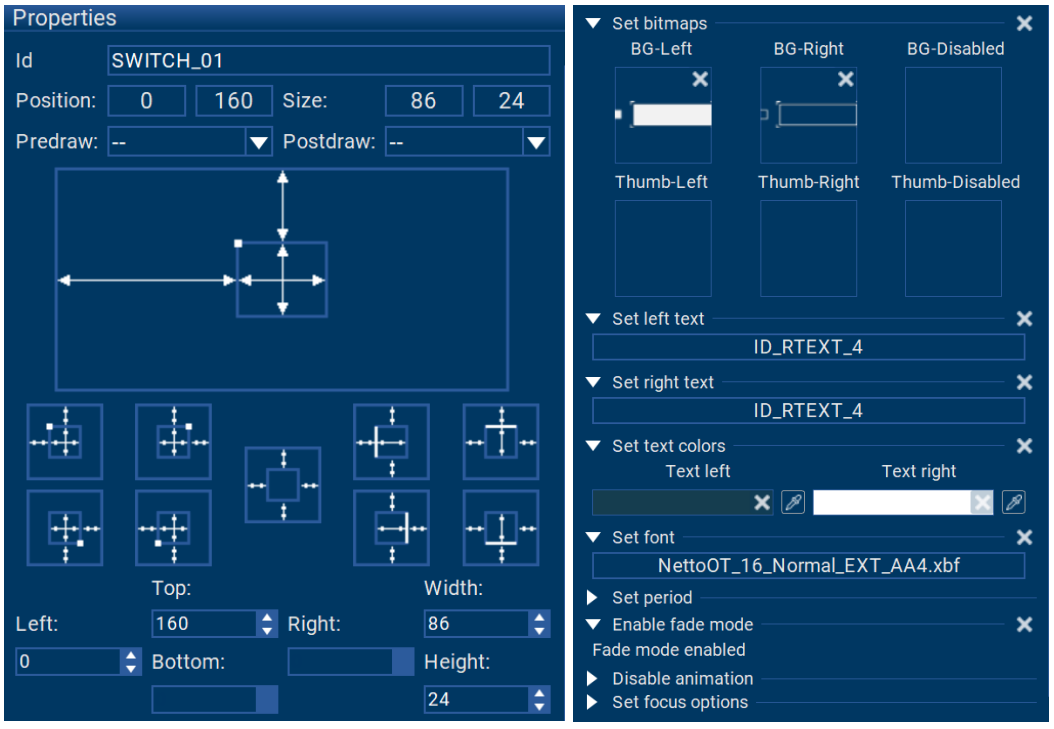


图 2-76 SWITCH_01 属性设置

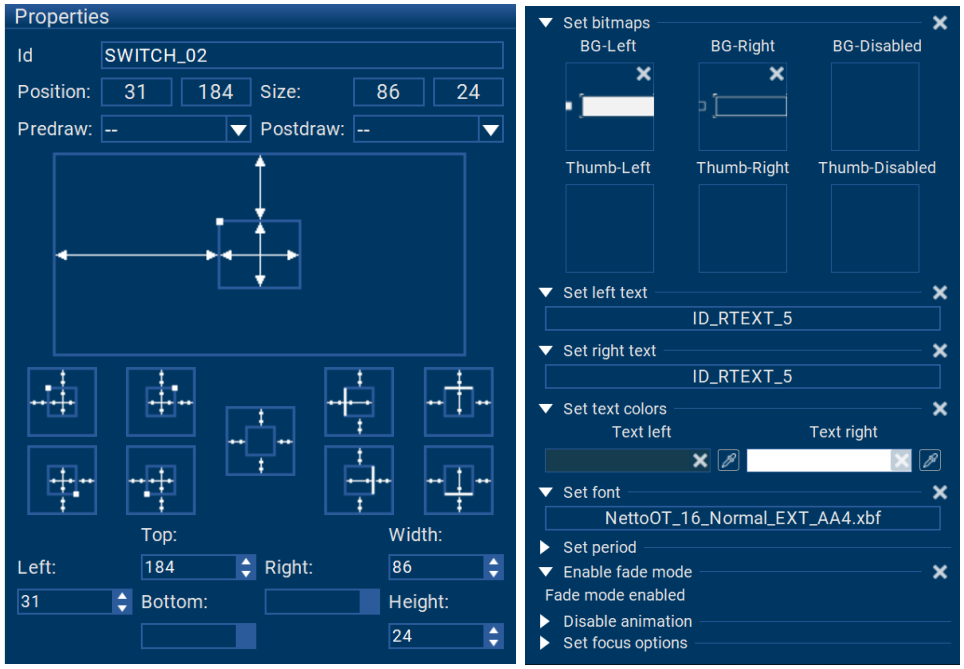


图 2-77 SWITCH_02 属性设置

7. 依图 2-78~图 2-89 步骤新增 Thrust bearing 与 Guide bearing 的控制开关。

1) 新增两组 Switch 及 Progbar 对象(SWITCH_TB、PROGBAR_TB、SWITCH_GB、PROGBAR_GB)，选择图片 Future_Switch_Button_Pressed_20x84.png、Future_Switch_Button_Unpressed_84x23.png 作为 Switch 图示。

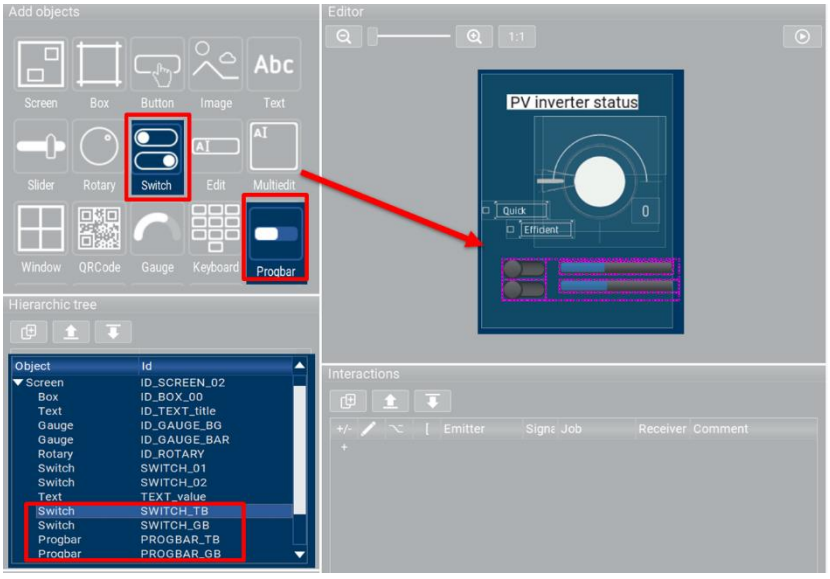


图 2-78 新增两组 Switch 与 Progbar 物件

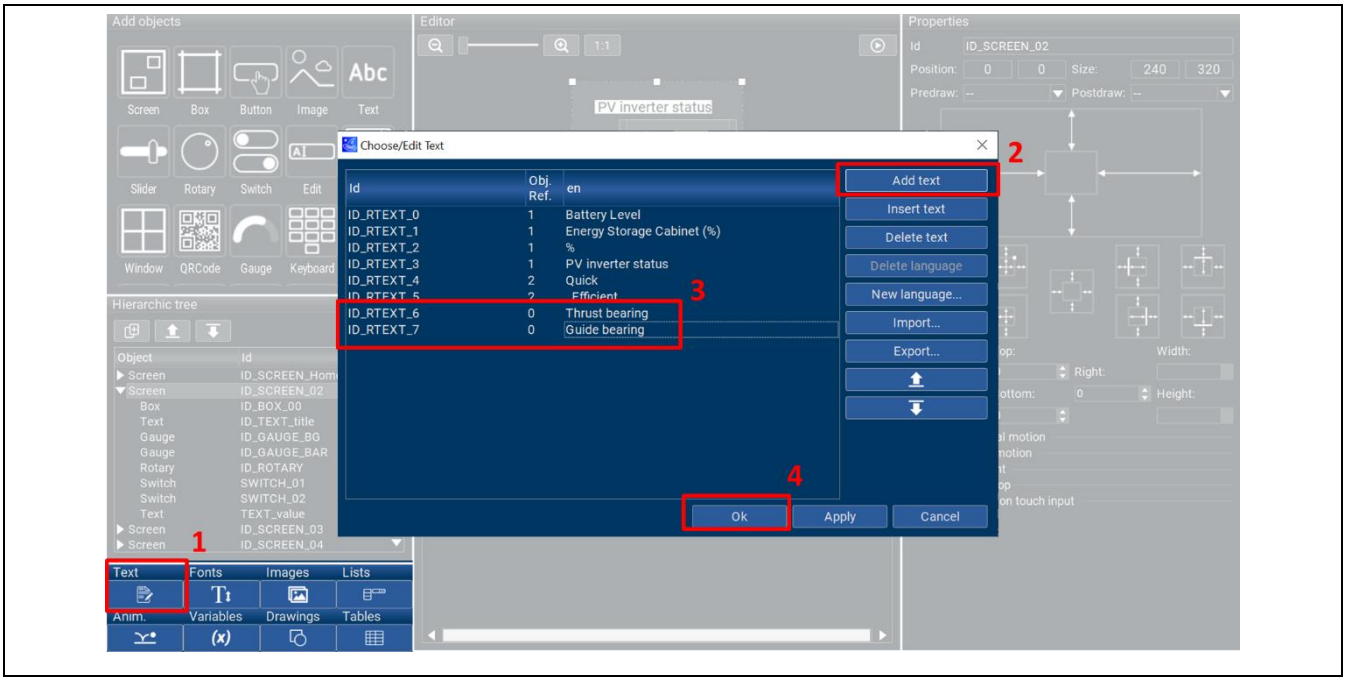


图 2-79 新增 switch 文字内容

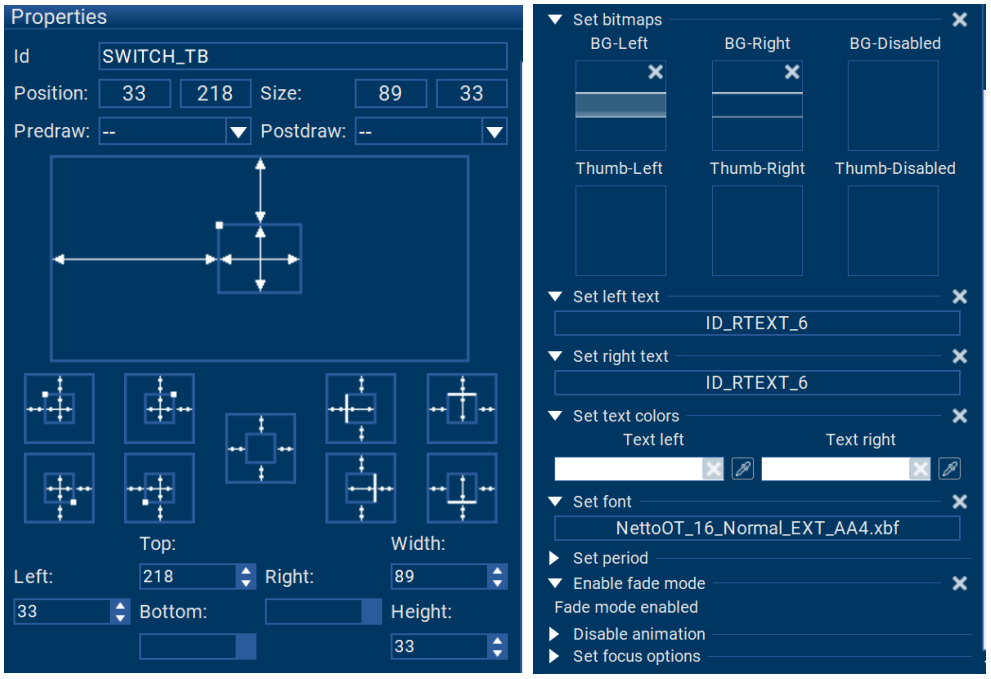


图 2-80 SWITCH_TB 属性设置

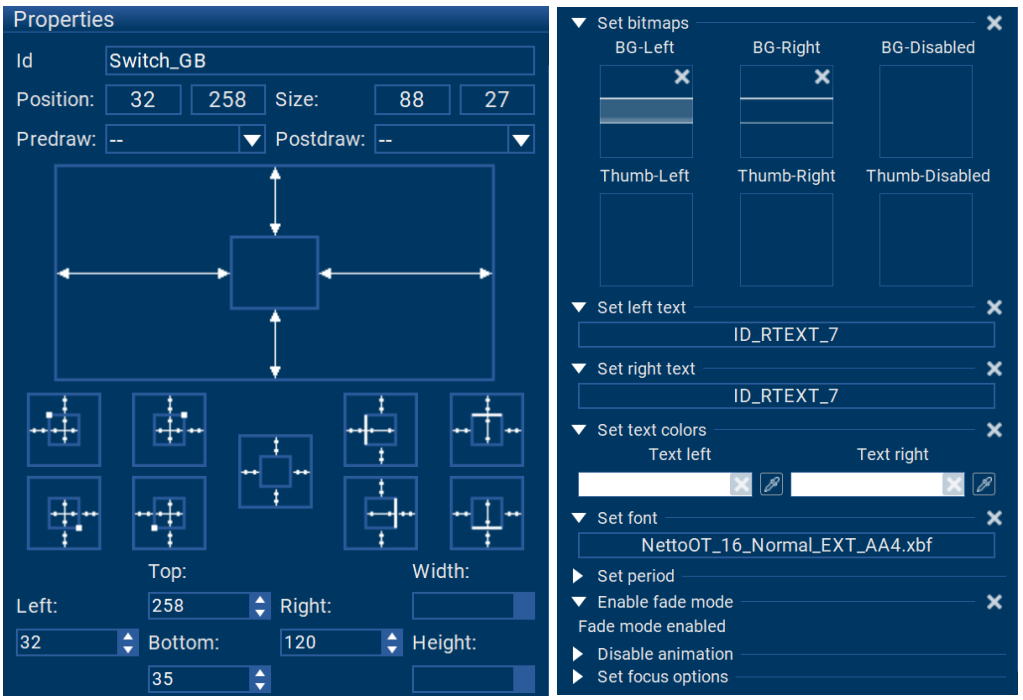


图 2-81 SWITCH_GB 属性设置

2) 选择图片 Progbar_Future_START_80x23.png 作为 Progbar(PROGBAR_TB、PROGBAR_GB)的 bitmap。

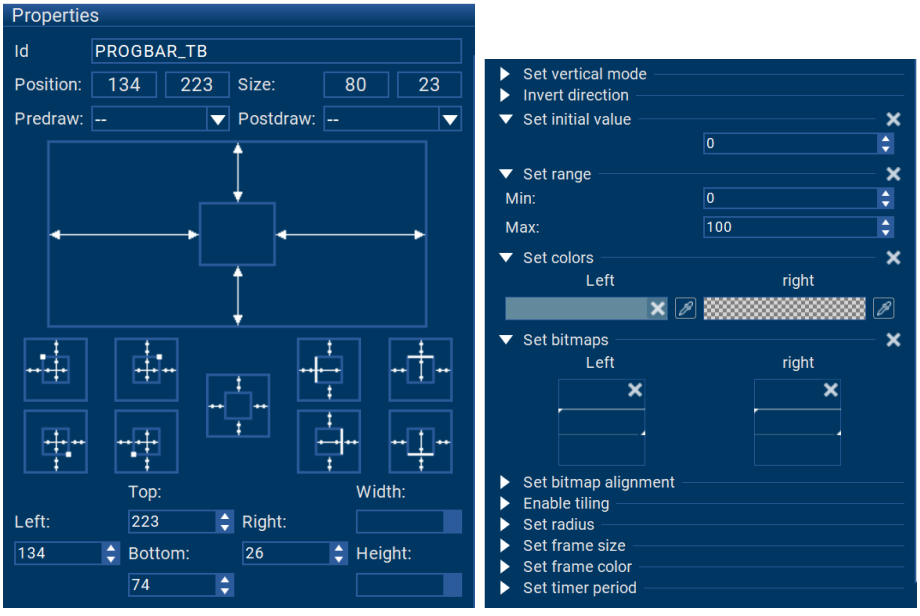


图 2-82 PROGBAR_TB 属性设置

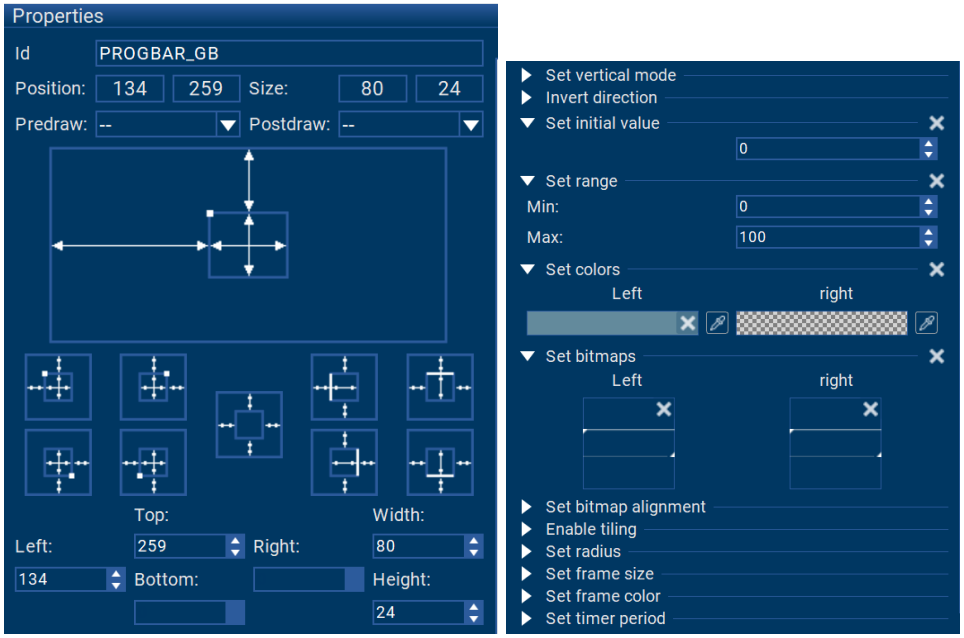


图 2-83 PROGBAR_GB 属性设置

3) 新增四个 TEXT 对象来显示数值与单位。

4) 设置第一组 Progbar(PROGBAR_TB)的 TEXT 对象，分别以 TEXT_num1、TEXT_unit1 表示数值与单位，并迭放在 Progbar 物件上。

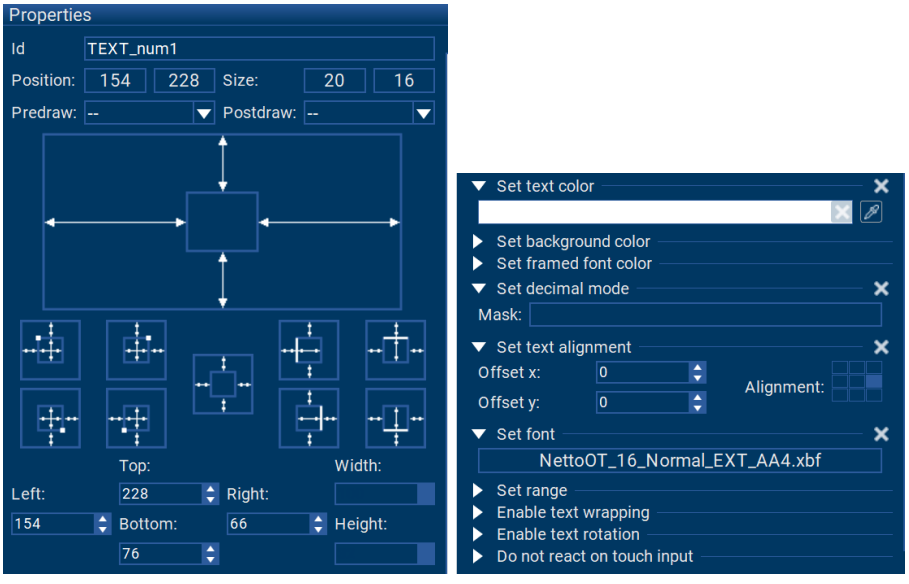


图 2-84 TEXT_num1 属性设置

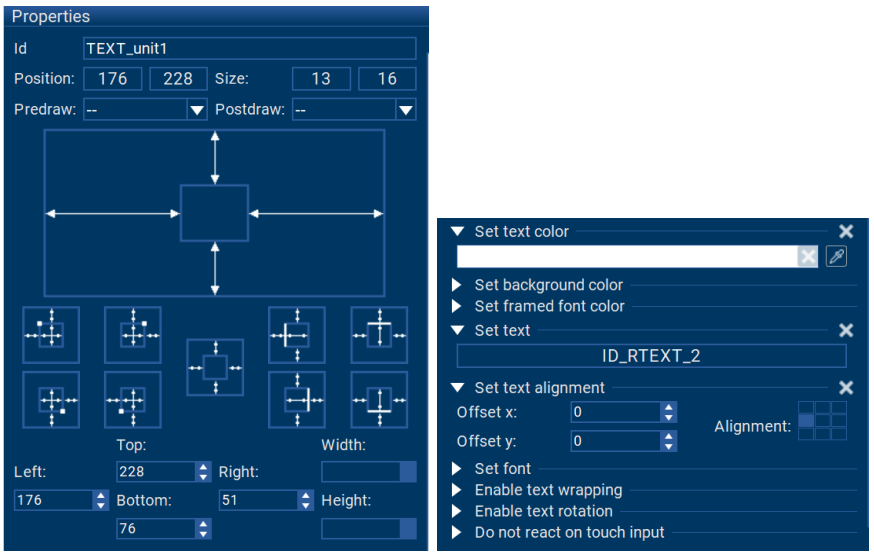


图 2-85 TEXT_unit1 属性设置

5) 设置第二组 Progbar(PROGBAR_GB)的 TEXT 对象，分别以 TEXT_num2、TEXT_unit2 表示数值与单位，并迭放在 Progbar 物件上。

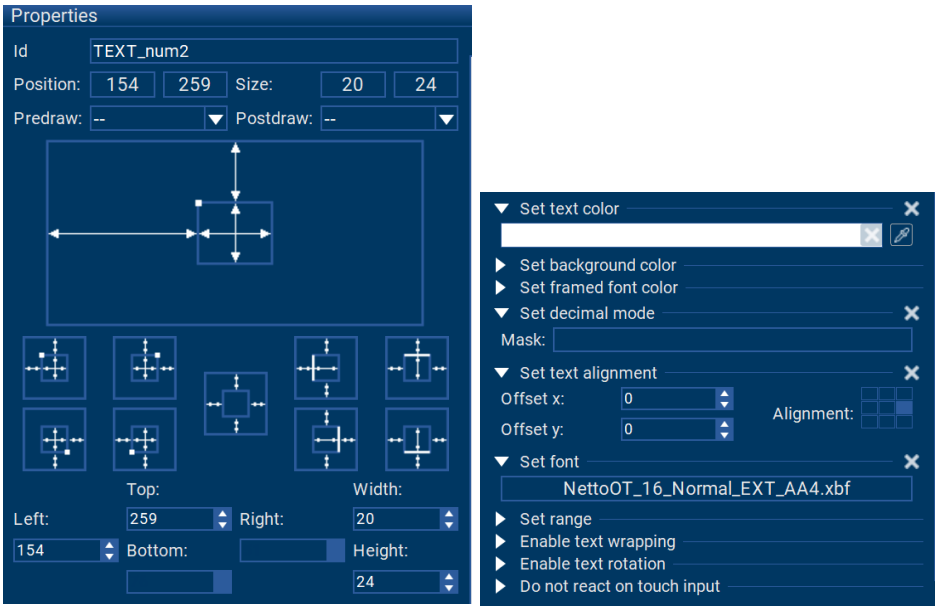


图 2-86 TEXT_num2 属性设置

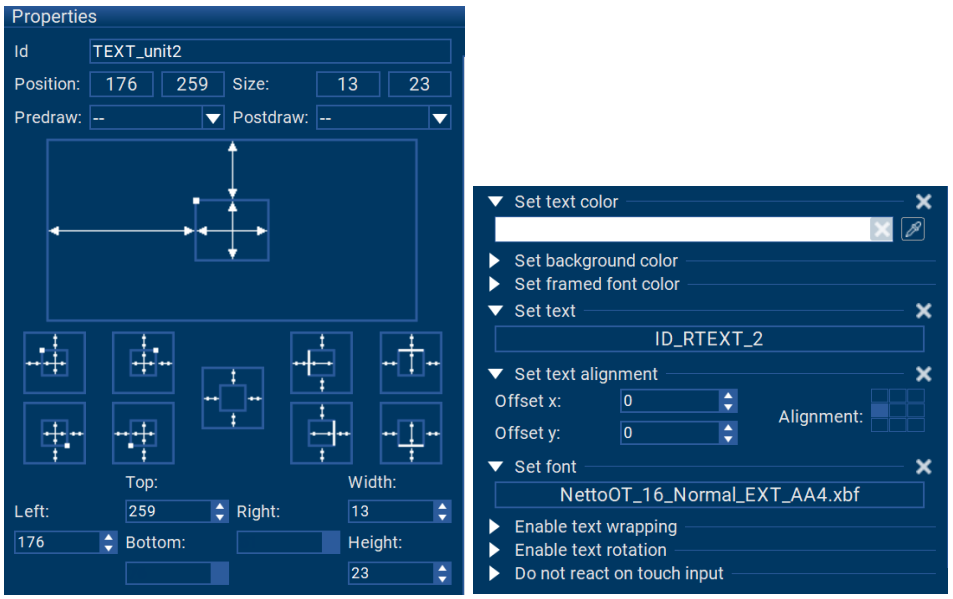


图 2-87 TEXT_unit2 属性设置

6) 新增 Slider 对象(ID_SLIDER)作为数值调整开关，并将图片 Slider_Start_Future.png、Slider_End_Future.png、Thumb_Future.png 设置为 Slider 的 Shaft left、Shaft right 与 Thumb up。

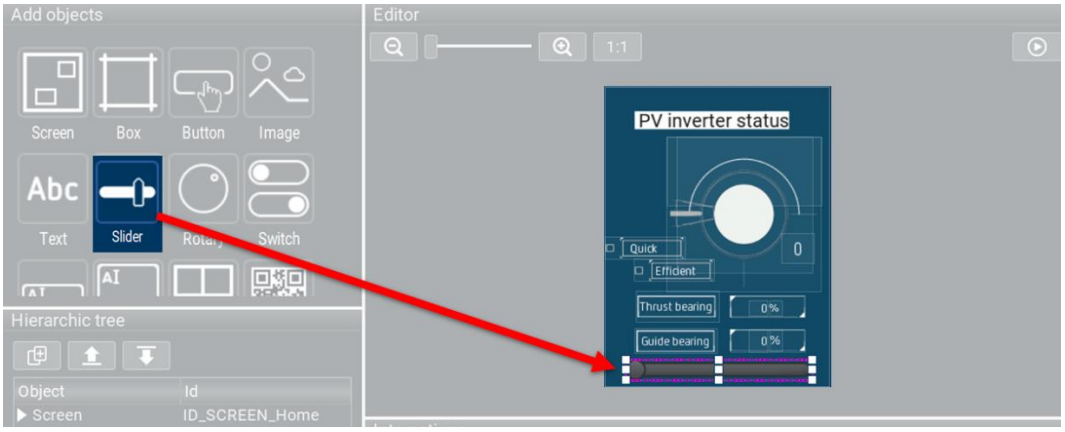


图 2-88 新增 Slider 物件

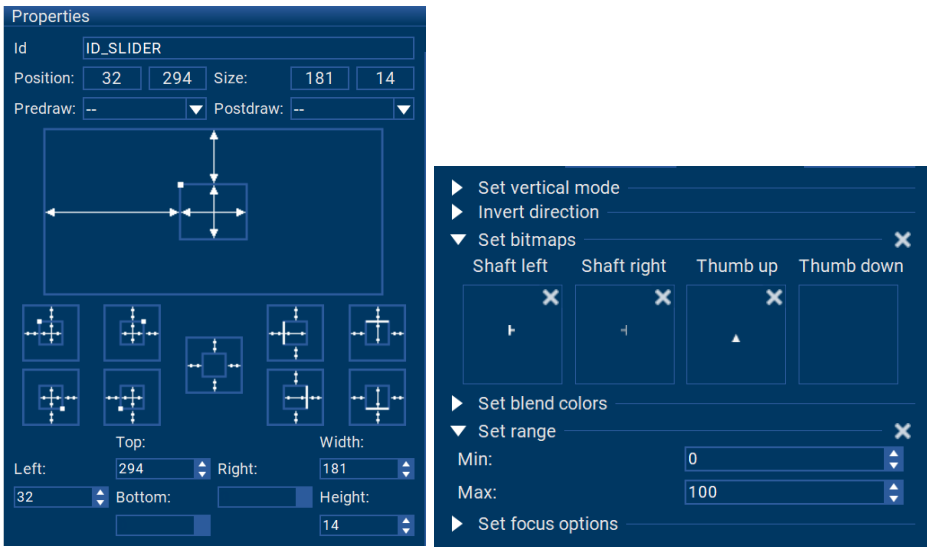


图 2-89 ID_SLIDER 属性设置

8. 新增两个 Button 对象作为前页(btn_p2_back)与后页(btn_p2_next)的换页按钮，并分别设图片 arrow2_20x20_new.png、arrow1_20x20_new.png 作为按钮图标。

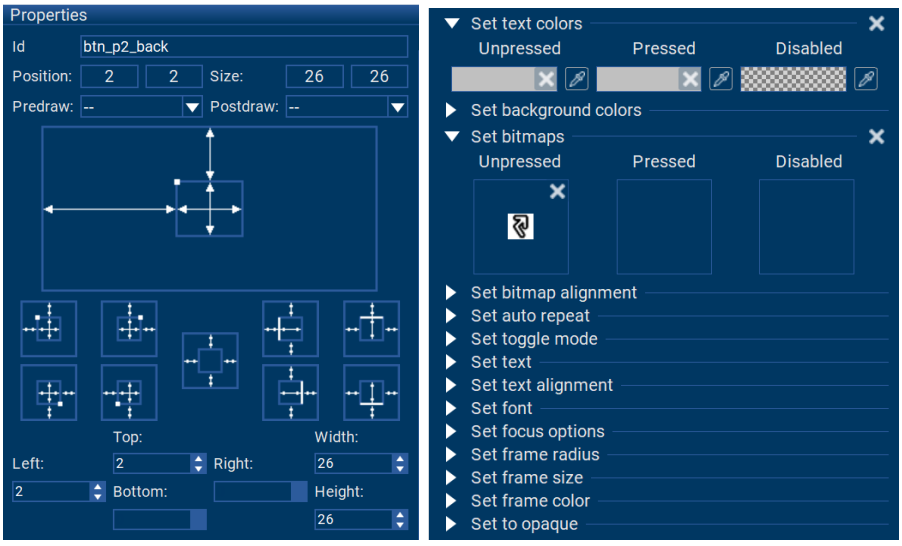


图 2-90 btn_p2_back 属性设置

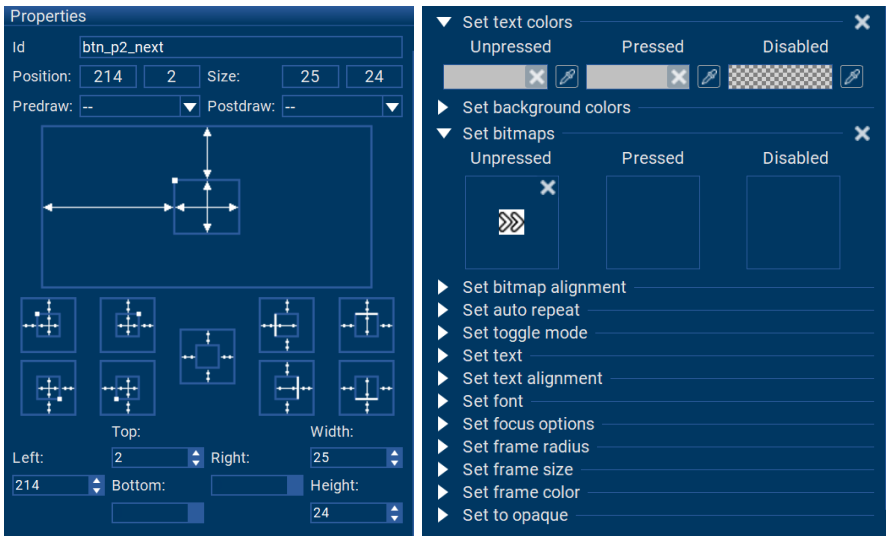


图 2-91 btn_p2_next 属性设置

2.3.2.2 对象互动设置

对象设置完成画面如图 2-92 所示，接着要建立对象间的交互作用(interactions)，使操作对象时有相应行为显示。

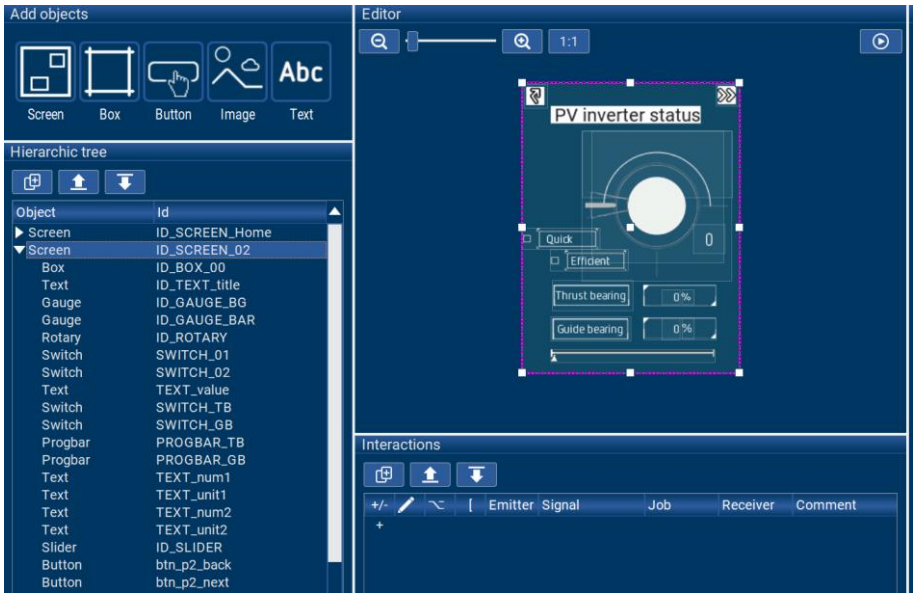


图 2-92 页面二设计完成图

1. 新增 Rotary(ID_ROTARY)、Gauge(ID_GAUGE_BAR)与 TEXT(TEXT_value)对象间的交互关系，使得 Gauge 数值会随 Rotary 转动而变化并显示数值。

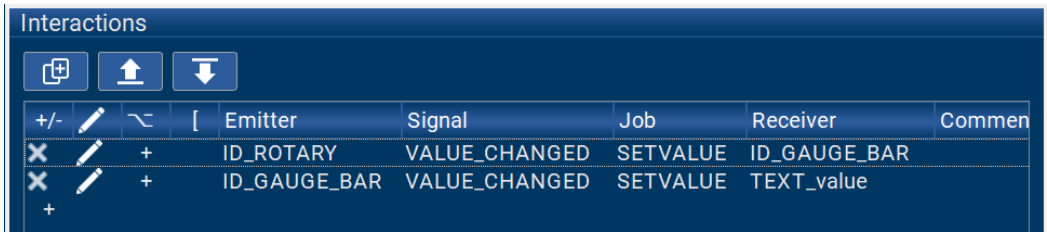


图 2-93 Rotary、Gauge 与 TEXT 交互作用设置

2. 新增两组 Progbar(PROGBAR_TB、PROGBAR_GB)与 TEXT(TEXT_num1、TEXT_num2)对象的交互关系，使得 TEXT 显示数值会随着 Progbar 变化。

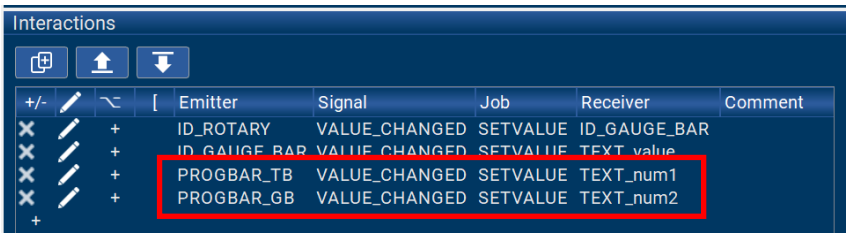


图 2-94 Progbar 与 TEXT 交互作用设置

3. 设置 Slider(ID_SLIDER)、Progbar(PROGBAR_TB、PROGBAR_GB)与 Switch(SWITCH_TB、SWITCH_GB)对象的交互关系，让使用者可透过 Slider 调整 Progbar 数值，并以 Switch 切换数值要映射数值的 Progbar。

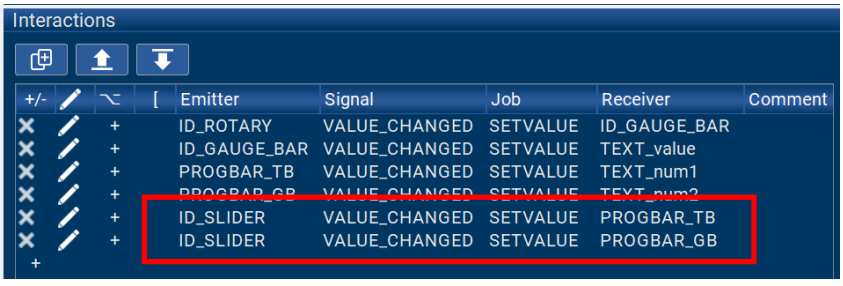


图 2-95 Slider 与 Progbar 交互作用设置

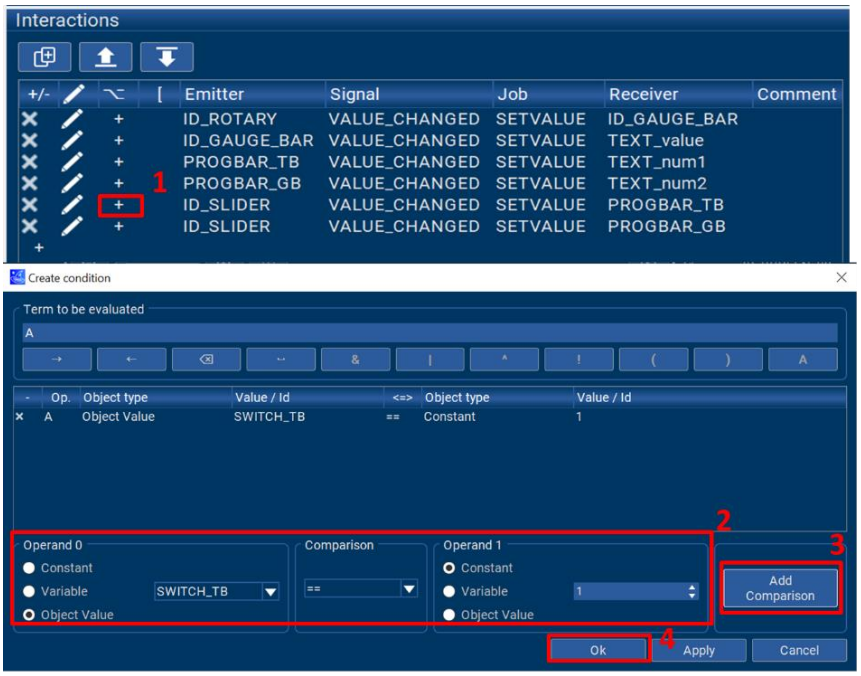


图 2-96 Switch 与 Slider 开关设置 1

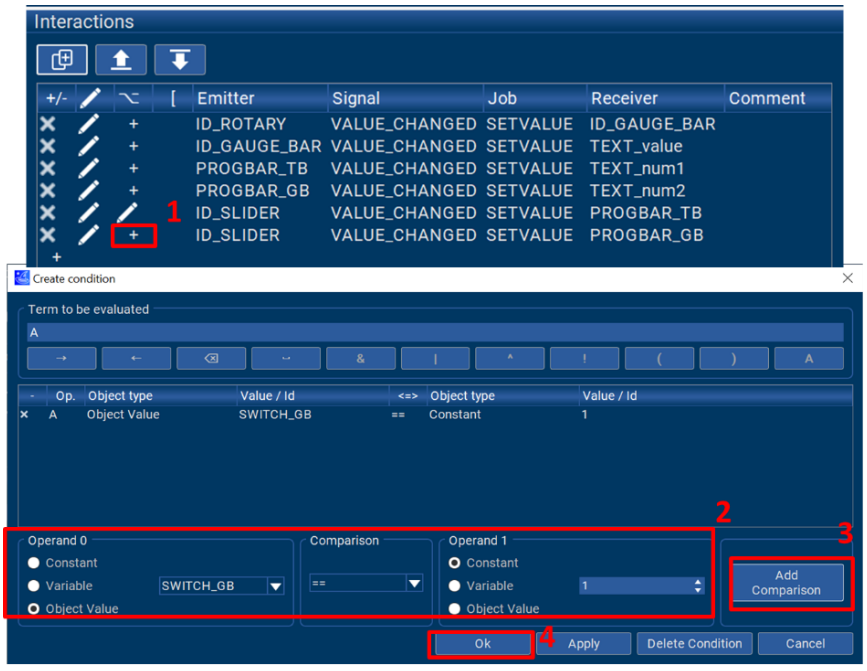


图 2-97 Switch 与 Slider 开关设置 2

4. 设置 Button(btn_p2_back、btn_p2_next)对象的换页功能，使点选按钮可以切回前一页与下一页。

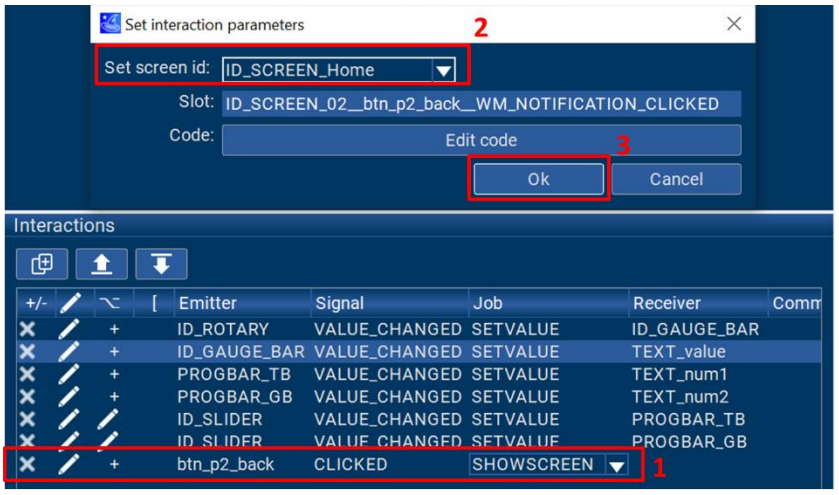


图 2-98 btn_p2_back 换页设置

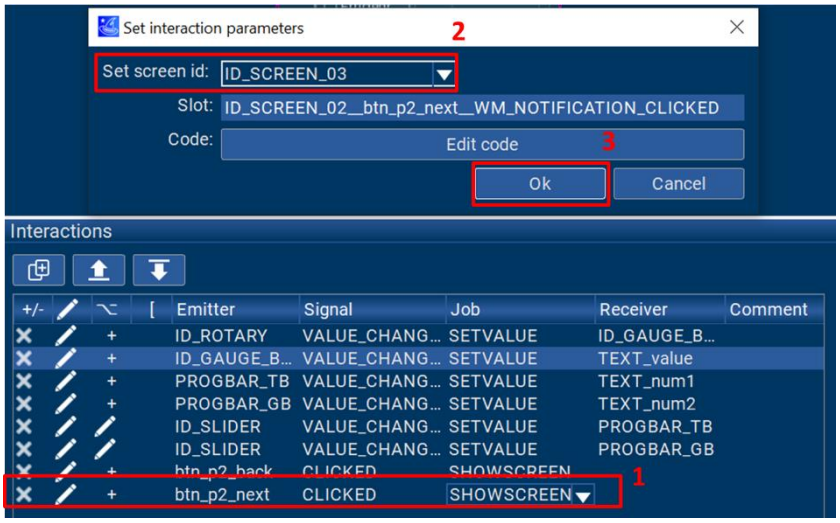


图 2-99 btn_p2_next 换页设置

2.3.3 三相电流显示设计(页面三)

三相电流应用大多需要显示电能波形，由于目前 AppWizard 不支持 Graphic 对象，故波形显示需透过 emWin library 以 GUI_Builder 设计，再将 emWin dialog 整合到 appWizard 项目中，此章节将详述 emWin dialog 设置波形以及将其嵌入 appWizard 的方法，此设计所用对象如表 2-4~表 2-5 所示。

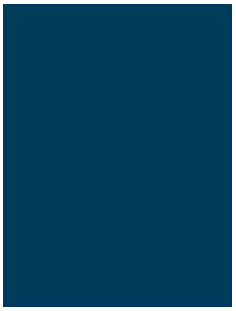
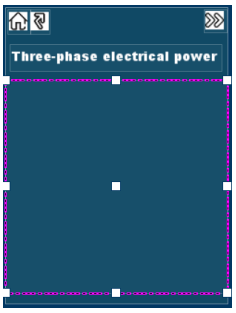
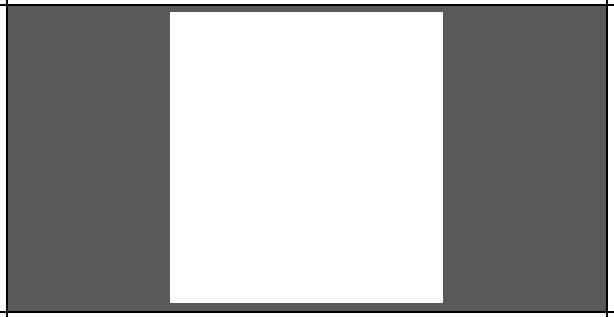
New Energy Gateway Description	Application GUI	Widget Type
背景图 240x320		Box
标题文字		Text
emWin dialog 放置区		window
回主页按钮		Button
上一页按钮		
下一页按钮		

表 2-4 三相电流显示设计对象 (AppWizard)

New Energy Gateway Description	Application GUI	Object Type
240x230 物件框		window
显示波形选择		Check Box

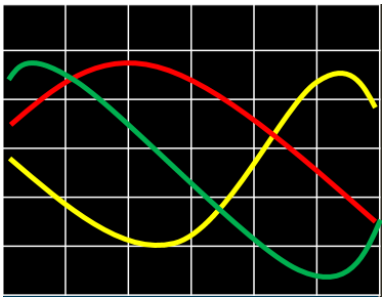
波形显示		Graphic

表 2-5 三相电流显示设计对象 (emWin)

2.3.3.1 EmWin Dialog 设置

1. 开启 Gui_Builder.exe，新增大小为 240*220 的 Window 对象作为 emWin dialog 基底。(位置: \EC_NUC980_AppWizard_New_Energy_Gateway_V1.00\Software Tools\GUI_builder)

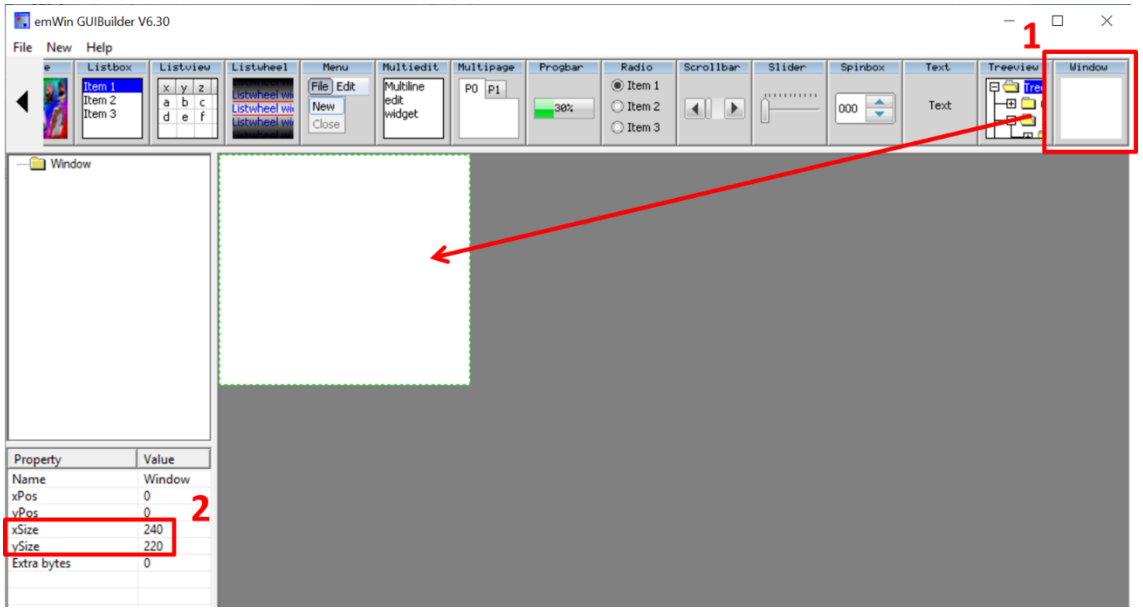


图 2-100 新增 window 物件并重设大小

2. 新增大小为 240x174 的 Graphic 对象作波形绘制，并将其位置贴齐 Window 对象下方。

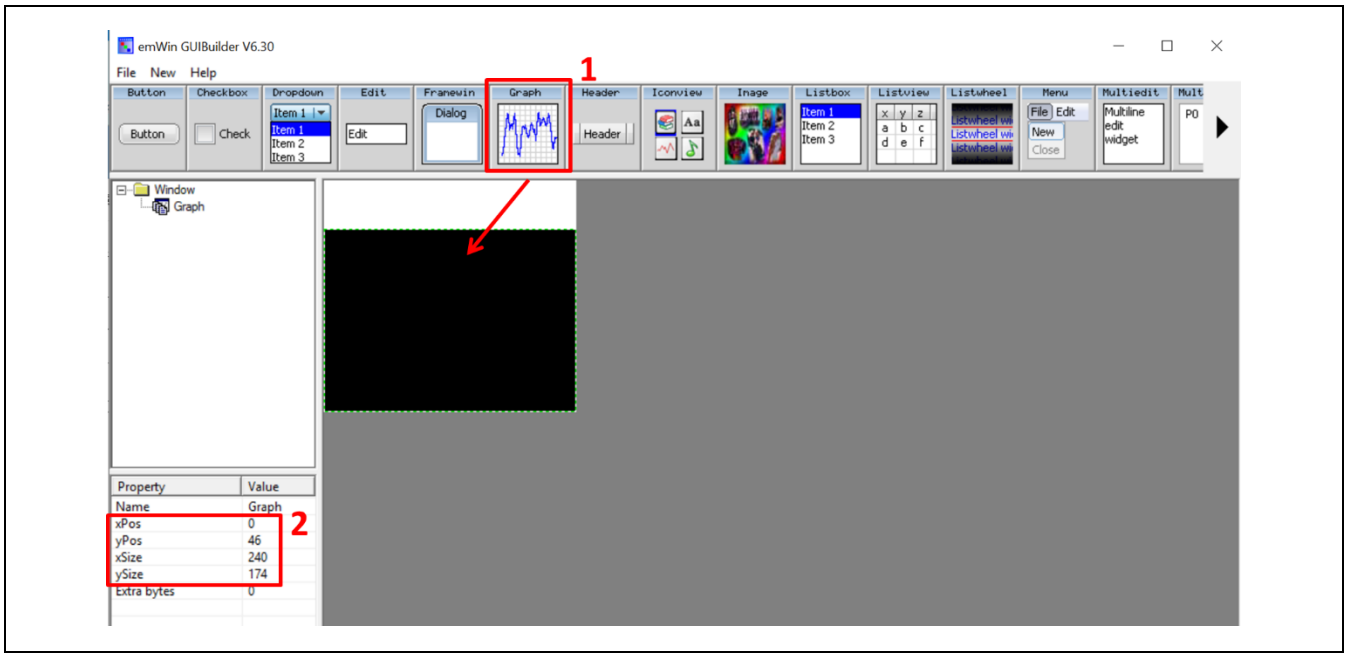


图 2-101 新增 Graphic 对象并调整位置

3. 新增三个 checkbox 对象作为各波形显示的功能开关。

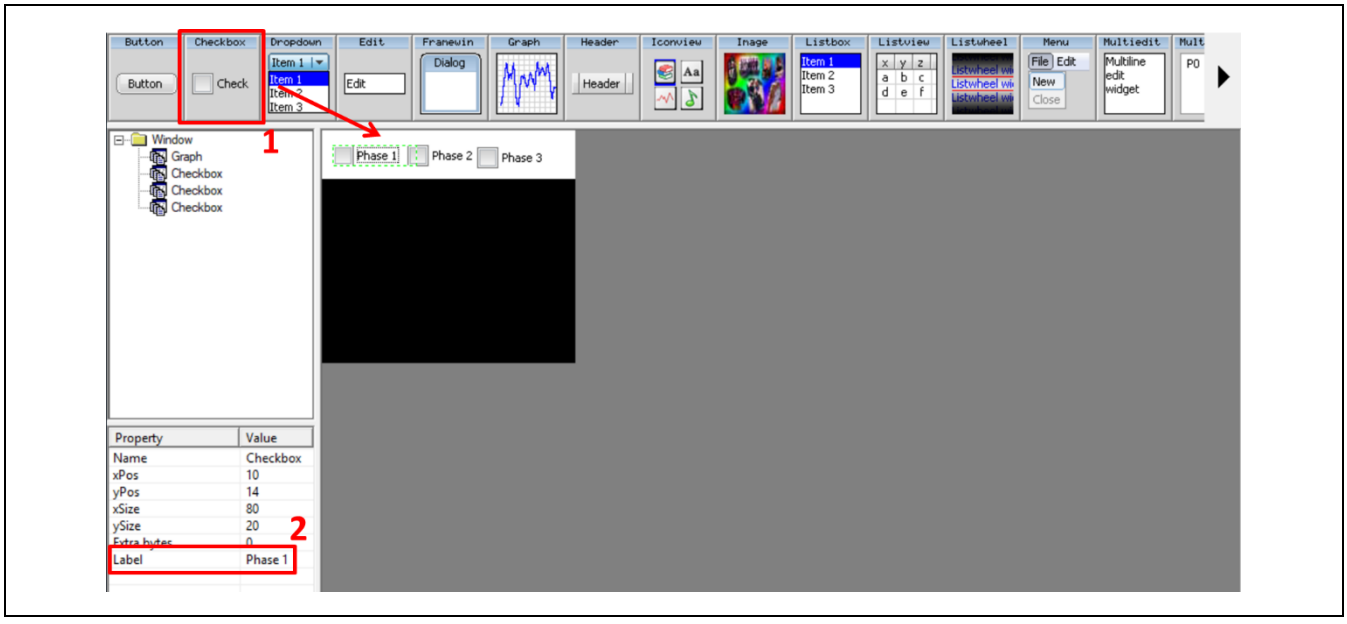


图 2-102 新增 Checkbox 对象并调整显示名称

4. 接口设置完后，点选左上角工具栏进行存盘，档案将自动储存于与 GUI_Builder.exe 相同目录底下。

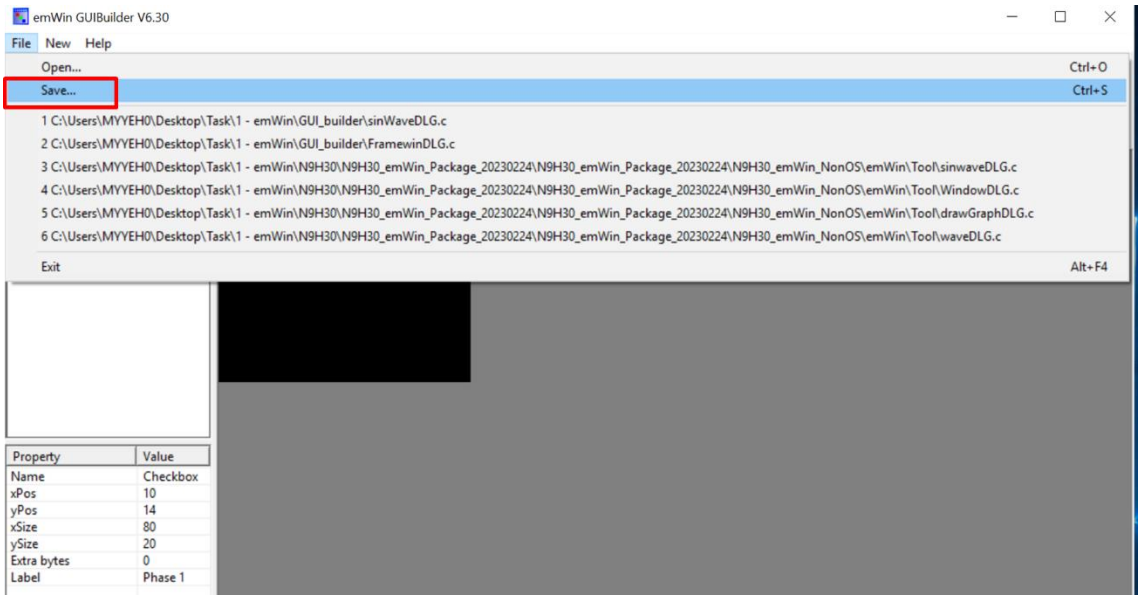


图 2-103 储存 emWin dialog

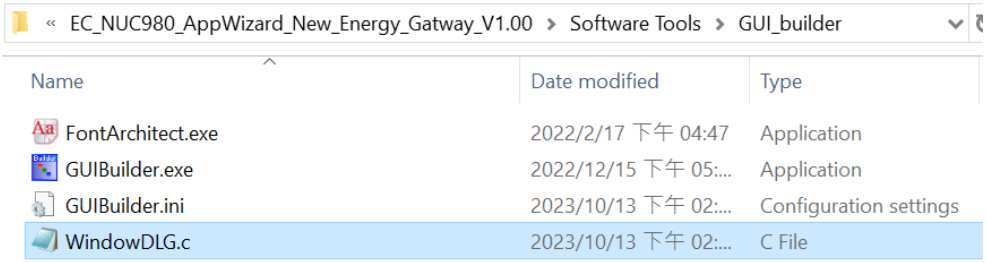


图 2-104 emWin 档案默认储存位置

5. 开启完成的 emWin dialog 档案(DLG.c)，在*_cbDialog* 函数内新增程序代码控制波形的绘制，并设置透过切换 check box 选择是否显示指定线条的功能。

   1) 宣告数组 ahData[3]来控制波形。(若要显示三项波形数值，可另外宣告一组数组储存数值，并于 AppWizard 内新增相关变量接收数值。详细方法请参考 AppWizard 官方范例 *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”内设定变量初始值。

```
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)内新增 case WM_TIMER，使波形数值随时间更新。

```
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) 于 case WM_NOTIFY_PARENT 内新增一组 Switch-Case，使程序依 checkbox 的状态更新 GRAPHIC 对象的数值。

```
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) 将更新后的 emWin dialog 档案置于 AppWizard project 指定路径(/Source/CustomCode) 下，位置如图 2-105 所示。

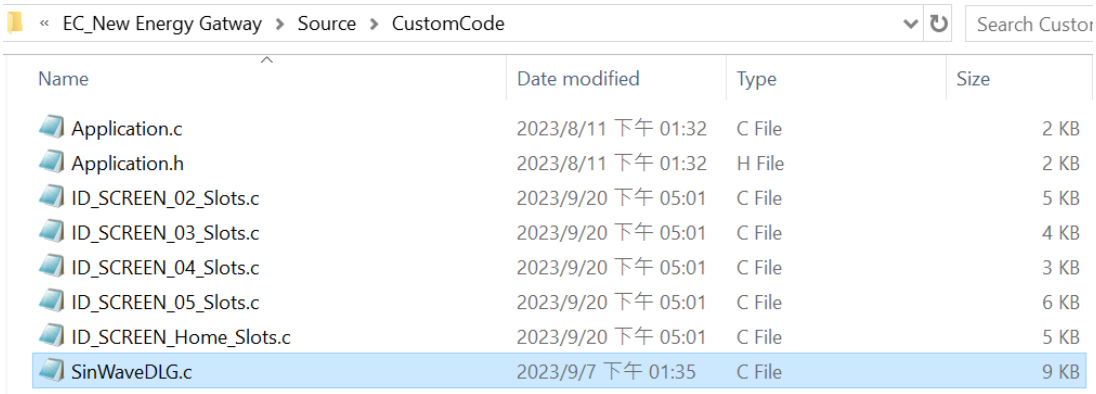


图 2-105 EmWin Dialog 放置位置

2.3.3.2 GUI 对象配置

1. 点选 ID_SCREEN_03 开始编辑三相电流显示页面，透过 Add objects 新增三个 Button 对象，以图片 home_20x20_new.png、arrow2_20x20_new.png、arrow1_20x20_new.png 作为回首页(btn_p3_home)、上一页(btn_p3_back)与下一页(btn_p3_next)的按钮图标，并依图 2-106 ~ 图 2-108 设置属性。

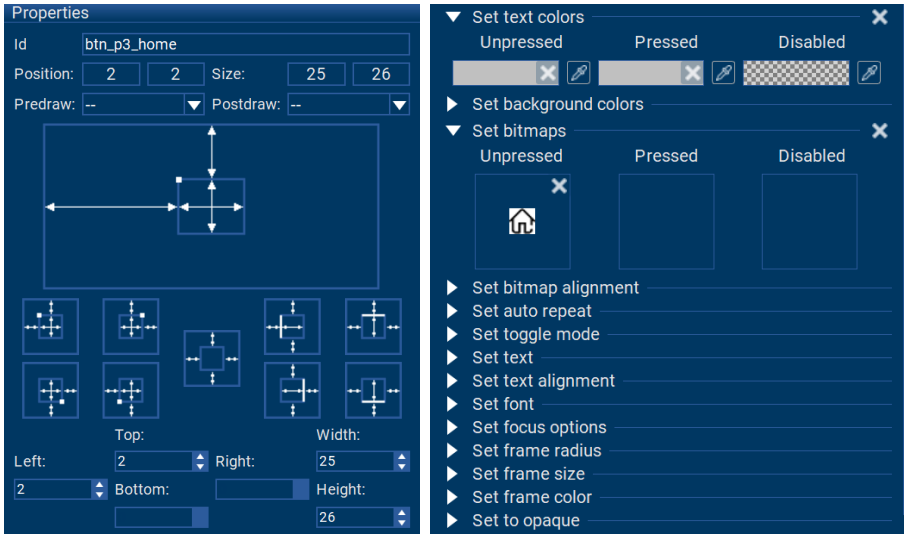


图 2-106 btn_p3_home 属性设置

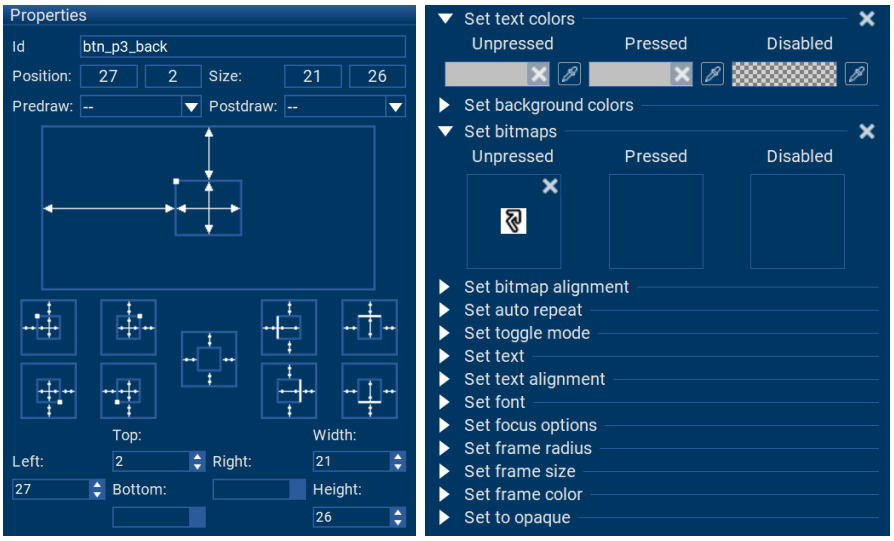


图 2-107 btn_p3_back 属性设置

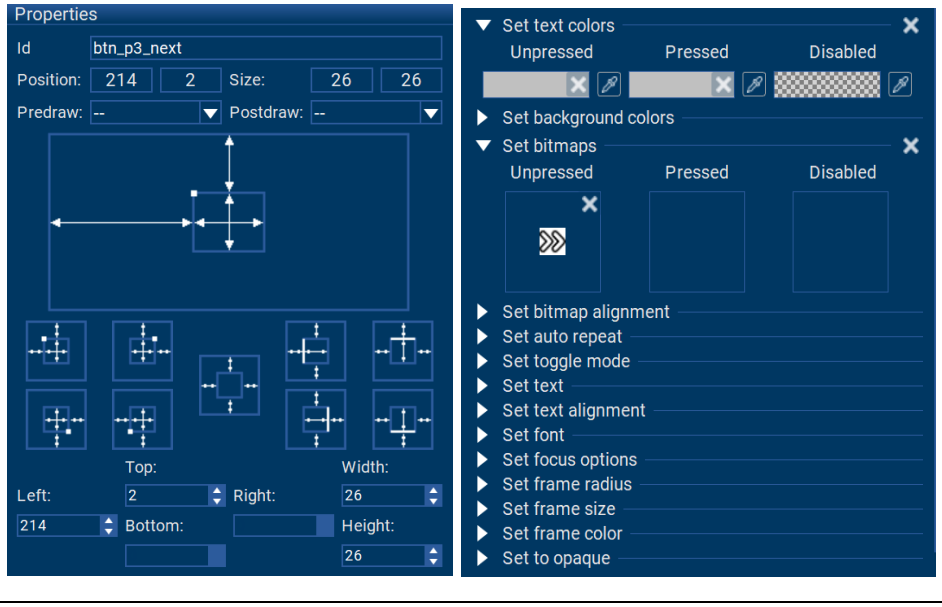


图 2-108 btn_p3_next 属性设置

2. 为了在 appWizard 内嵌入 emwin dialog，因此新增一个大小与 emWin dialog 相同的 window 对象(emwin_window)，并依图 2-110 设置对象属性。

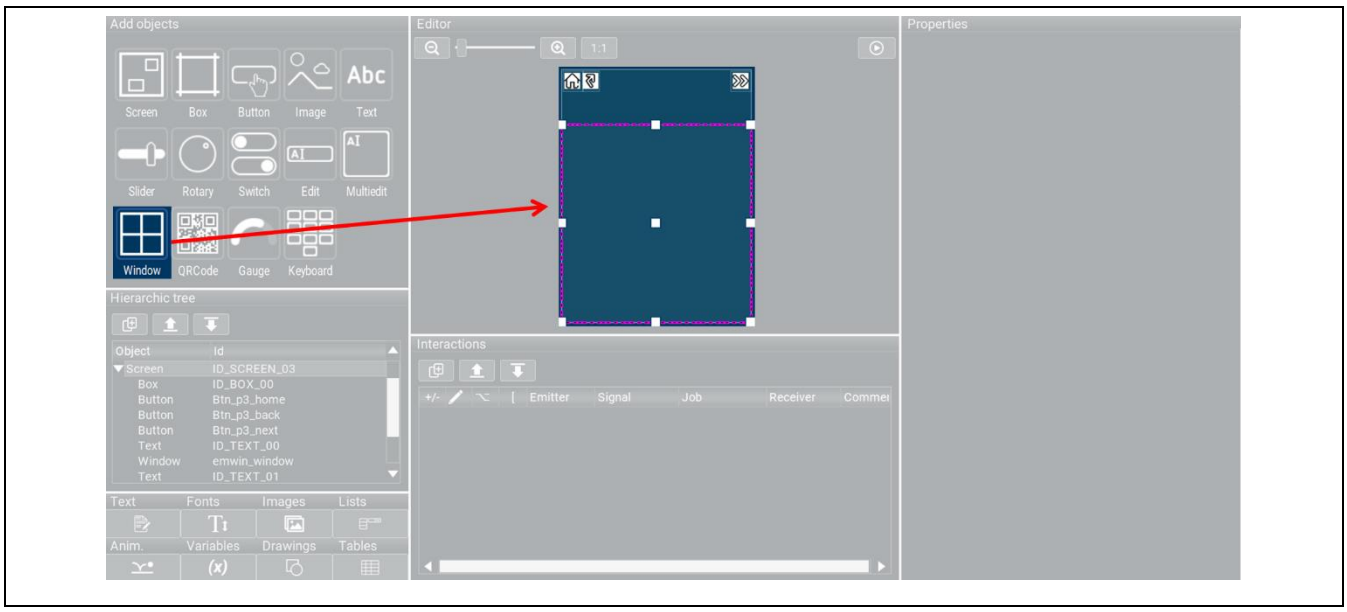


图 2-109 新增 window 物件

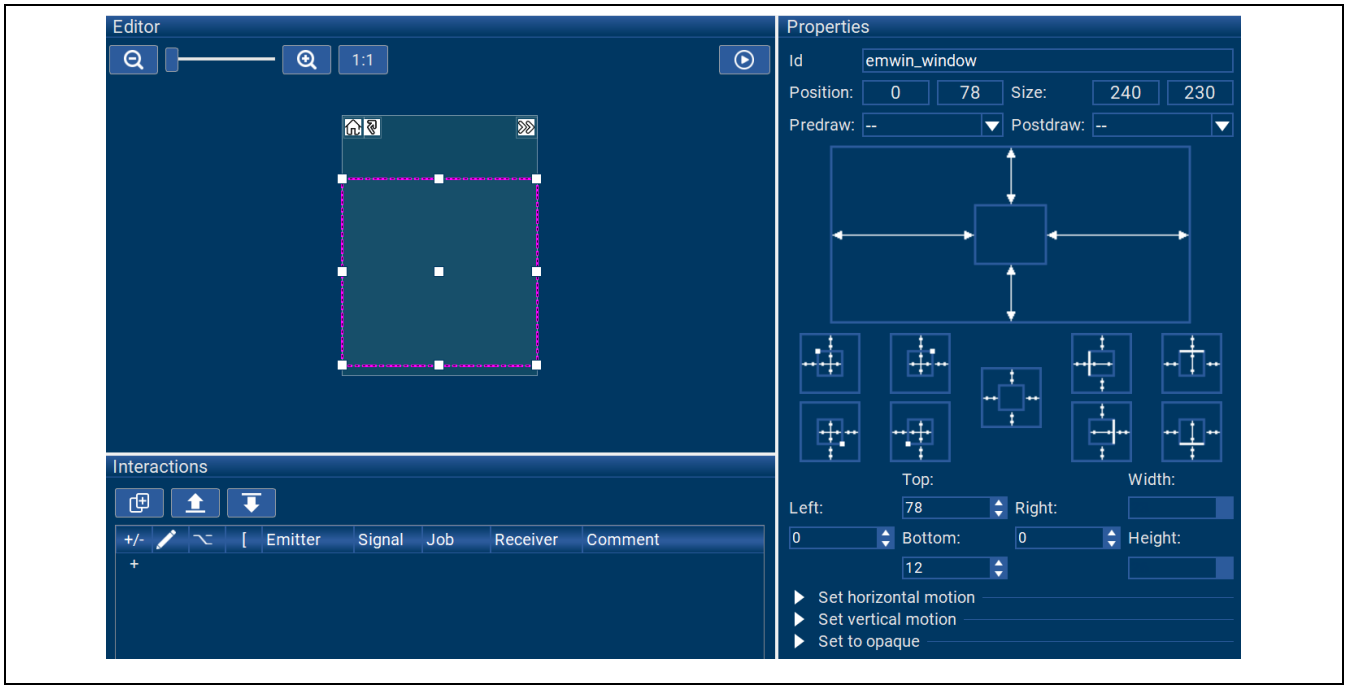


图 2-110 emwin_window 属性设置

3. 新增一个 TEXT 对象(ID_TEXT_00)显示标题文字，依图 2-111~图 2-113 设置显示文字”Three-phase electrical power”与属性。

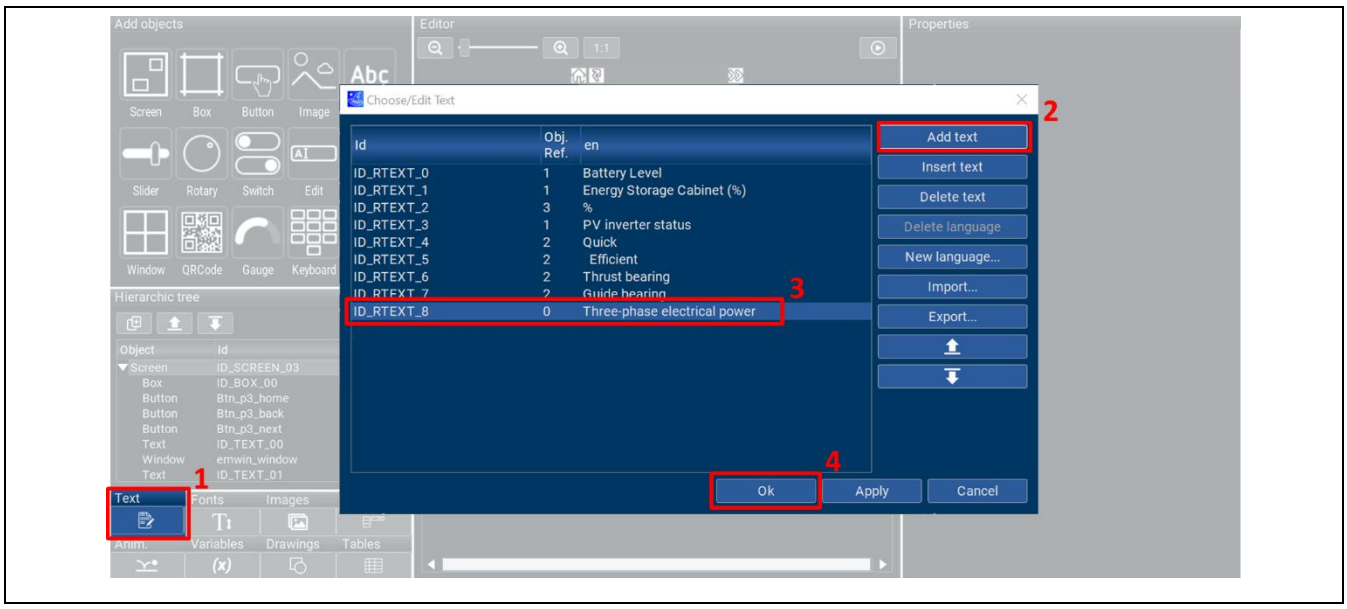


图 2-111 新增显示文字(ID_RTEXT_8)

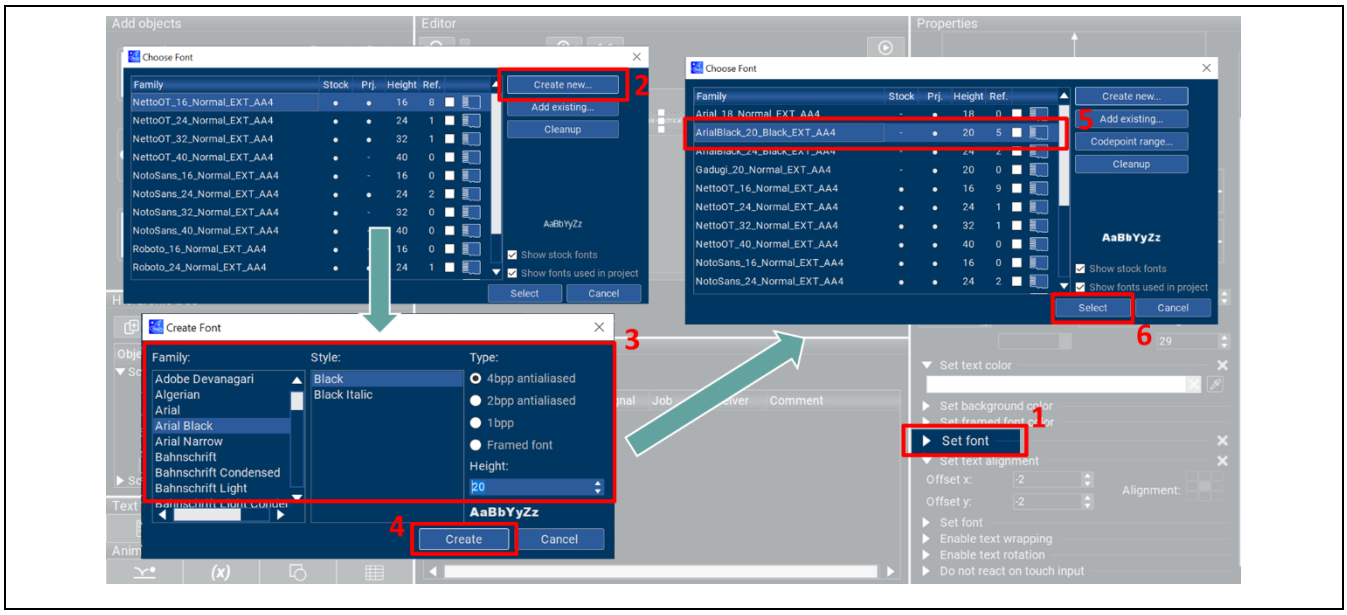


图 2-112 新增自定义文字格式

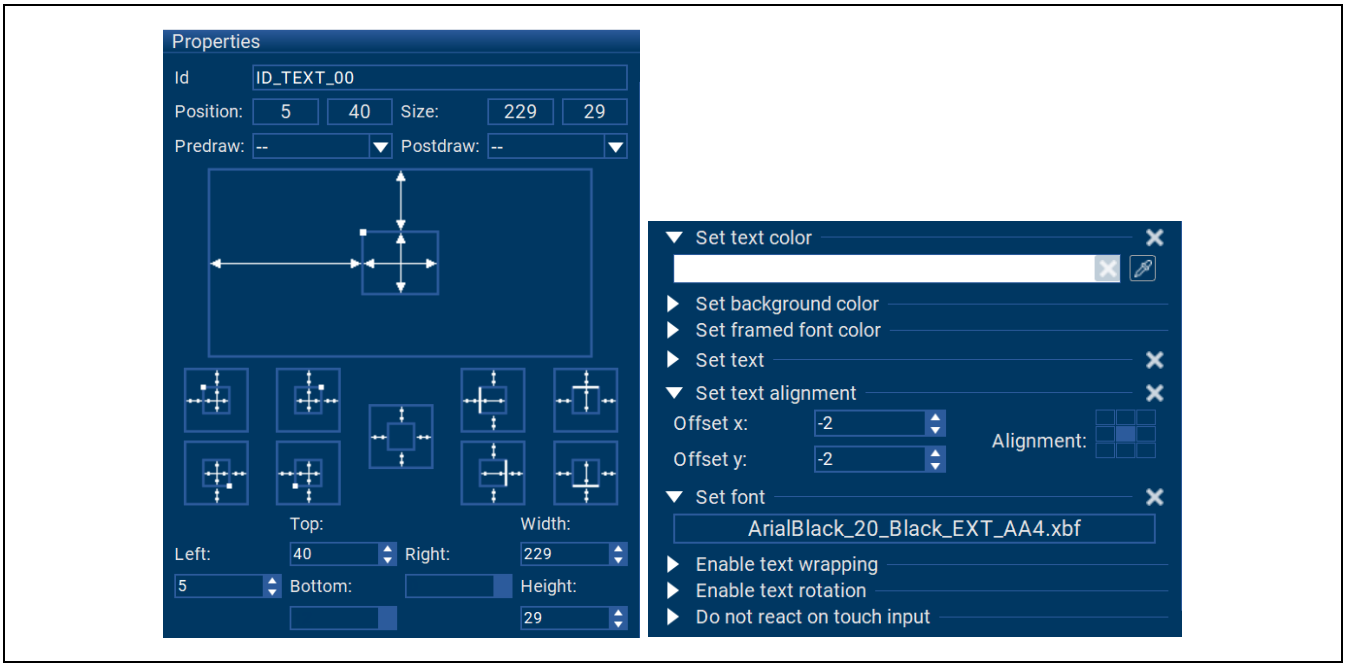


图 2-113 ID_TEXT_00 属性设置

2.3.3.3 将 EmWin Dialog 放入 AppWizard 专案

要将 emWin dialog 嵌入 AppWizard 项目的话，需透过 AppWizard 的 interactions 操作，详细流程如下所述。

1. 在 interactions 内新增一组 **Emitter** 为 *ID_SCREEN_03*(要被嵌入 emwin dialog 的 screen ID)、**Signal** 为 *INITDIALOG* 且 **Job** 为 *NULL* 的交互设置，并设定 Slot 名称为 *ID_SCREEN_03_Init*。(Slot 代表此动作在 *ID_SCREEN_03_slot* 内的函数名称)

图 2-114 EmWin Dialog 交互作用设置

2. 先将目前项目导出，以便更新程序使 emWin dialog 结合到 AppWizard 项目中。

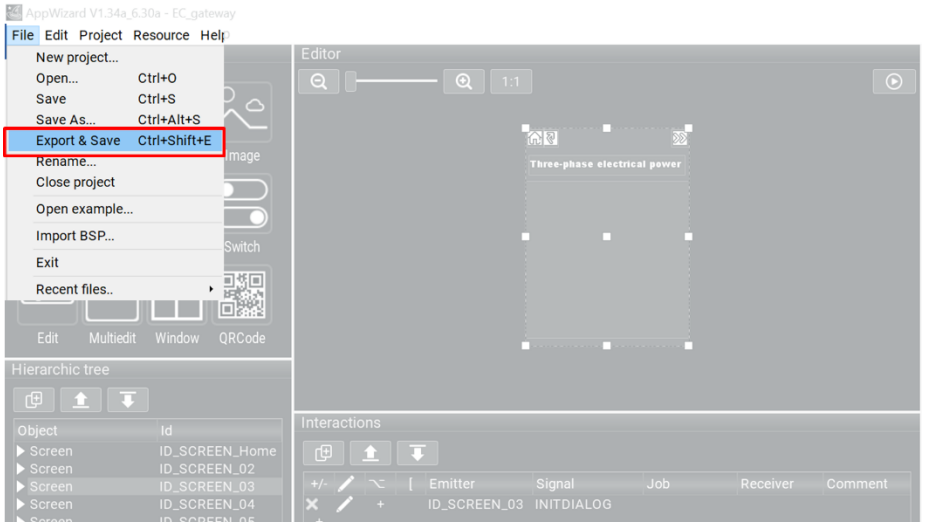


图 2-115 导出目前项目

3. 在项目文件夹位置*Source**CustomCode* 开启要嵌入 emWin dialog 的接口程序 (ID_SCREEN_03_Slots.c)，找到步骤 1 设置的 slot 函数名称(ID_SCREEN_03_Init)后，对内容进行下述修改：

```
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. 在相同路径开启 emWin dialog 程序(SinWaveDLG.c)，依图 2-116~图 2-117 修改后存盘。

图 2-116 新增 include 宣告

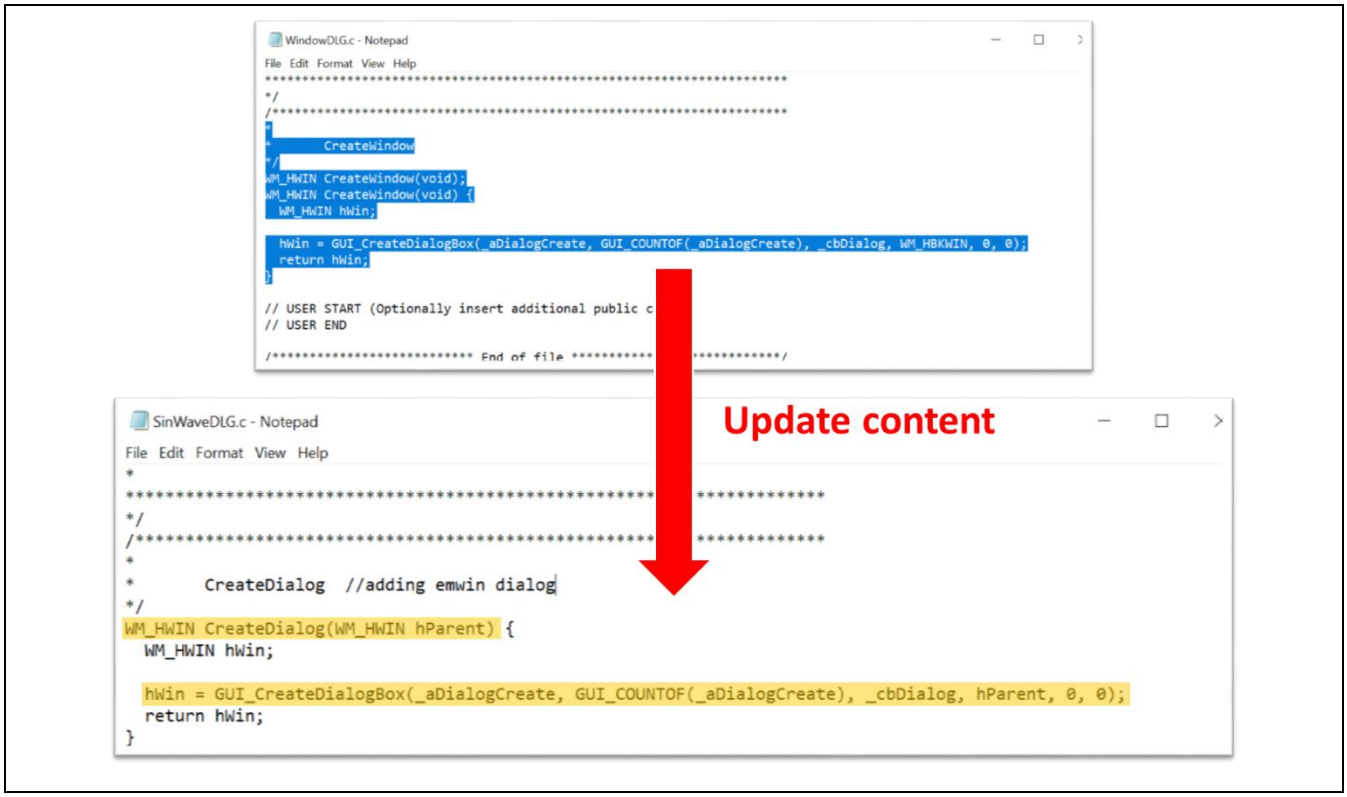


图 2-117 更新 dialog 宣告函数

emWin dialog 内容：

```
…

#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 对象互动设置

对象设置完成画面如图 2-118，接着要建立对象间的交互作用(interactions)，使操作对象时能有相应行为显示。 (Note: emWin dialog 效果无法在 AppWizard 预览，需部署到开发版确认)

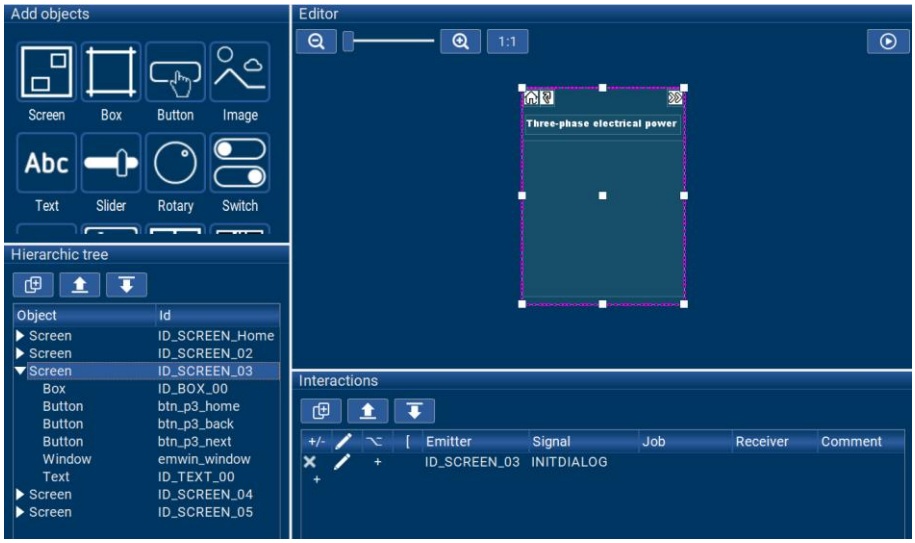


图 2-118 页面三设计完成图

1. 设置 Button(btn_p3_home、btn_p3_back、btn_p3_next)对象的换页功能，使点选按钮可以切回首页、前一页及下一页。

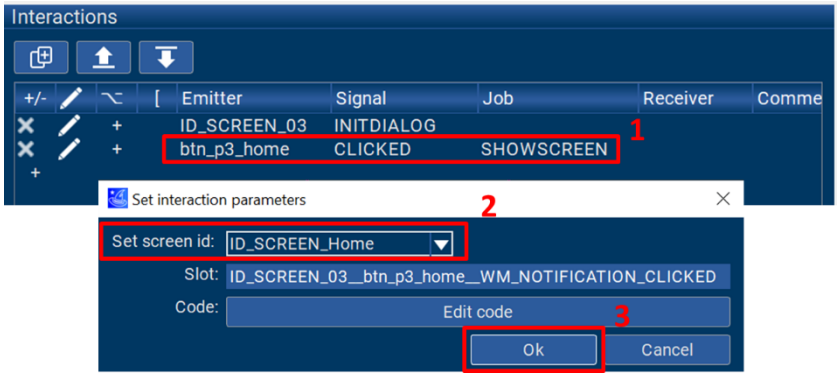


图 2-119 btn_p3_home 换页设置

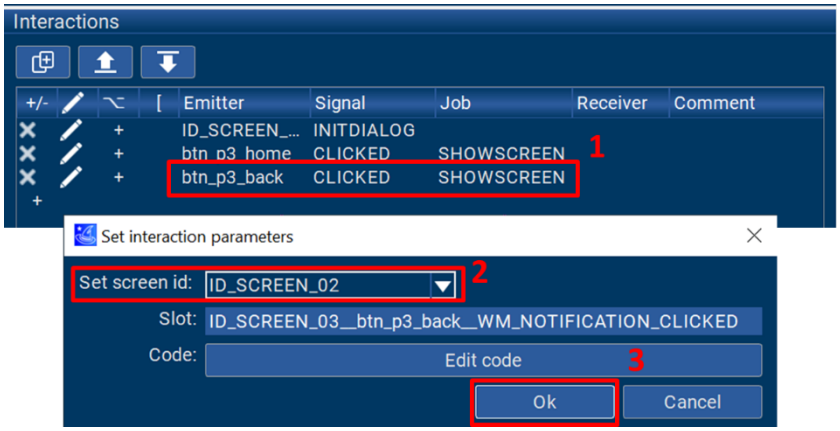


图 2-120 btn_p3_ back 换页设置

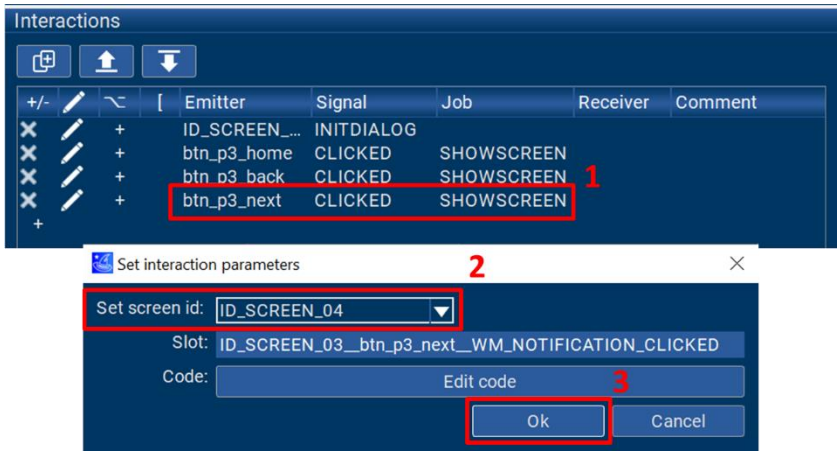


图 2-121btn_p3_ next 换页设置

2.3.4 用电量表设计(页面四)

柱形图是网关常用来做为数值计量或呈现的方式之一，在此页也结合 appWizard 的动画显示功能，使画面内容更加丰富生动，且可利用 QR Code 导向特定网址供用户查阅(可用于详细用电量报表、数据分析等数据页面的显示)，此设计所用对象如表 2-6 所示。

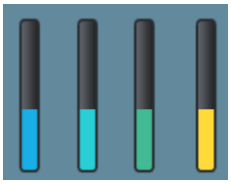
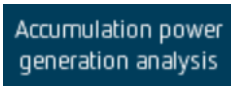
New Energy Gateway Description	Application GUI	Widget Type
背景图 240x320		Box
动画背景图 200x207		Box
累积用电量直方图		Progbar
网页导向 QR code		QR code
标题文字		Text
直方图说明		
QR code 说明		
回主页按钮		Button
上一页按钮		
下一页按钮		

表 2-6 用电量表设计对象

2.3.4.1 GUI 对象配置

1. 点选 ID_SCREEN_04 开始编辑用电量表画面，透过 Add objects 新增三个 Button 对象，并分别以图片 home_20x20_new.png、arrow2_20x20_new.png、arrow1_20x20_new.png 作为回首页(btn_p4_home)、上一页(btn_p4_back)与下一页(btn_p4_next)的按钮图标。

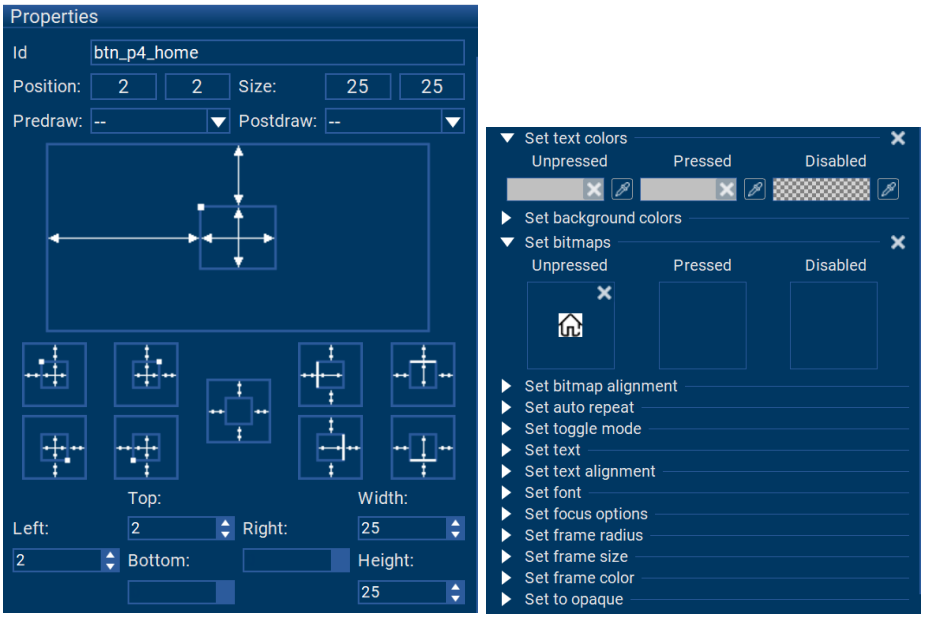


图 2-122 btn_p4_home 属性设置

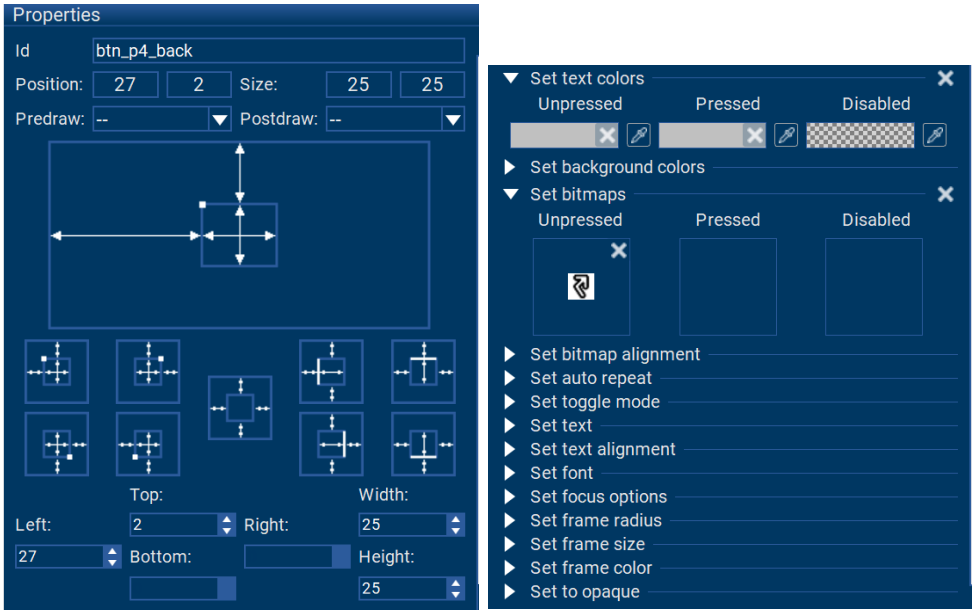


图 2-123 btn_p4_back 属性设置

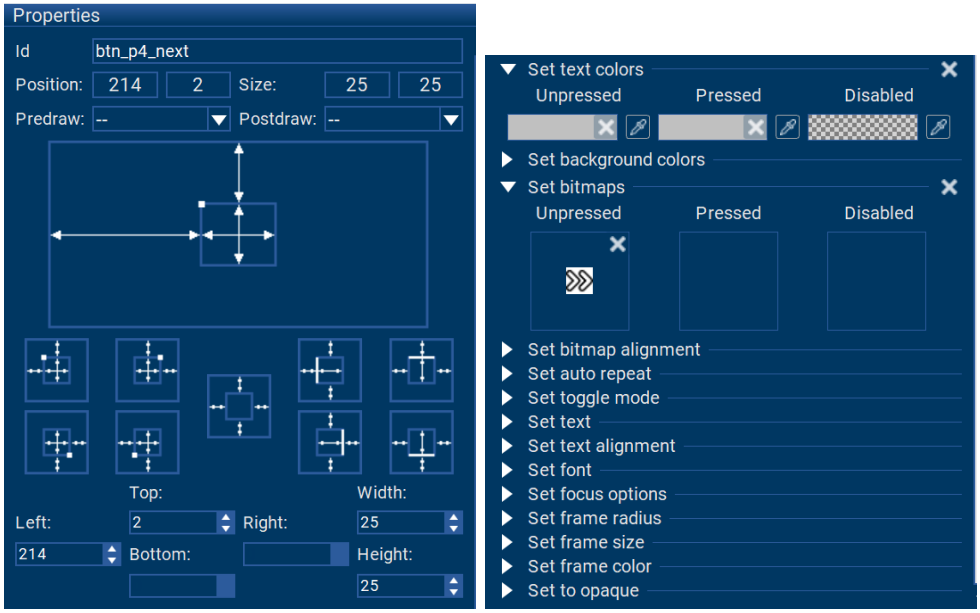


图 2-124 btn_p4_next 属性设置

2. 新增第二个 BOX 对象(ID_BOX_01)作为动画区域的背景色，属性设置如图 2-125。

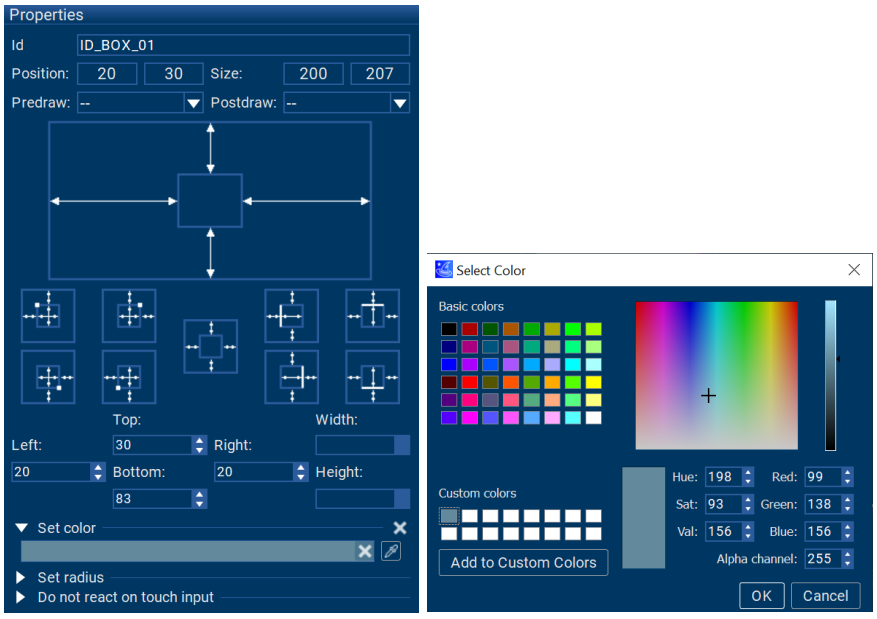


图 2-125 ID_BOX_01 属性设置

3. 新增四个 Progbar 对象(PROGBAR_00~PROGBAR_03)作为不同层级的电量使用显示，依图 2-126~图 2-129 设置属性。首先开启 Progbar 的垂直模式，并分别以图片 progbar_LightBlue.png、progbar_LightGreen.png、progbar_Green.png 与 progbar_Yellow.png 作为 Progbar 的 bitmap bottom，并选用图片 DARK_Progbar_Tile_V_Gray_16x1.png 作为 bitmap top。

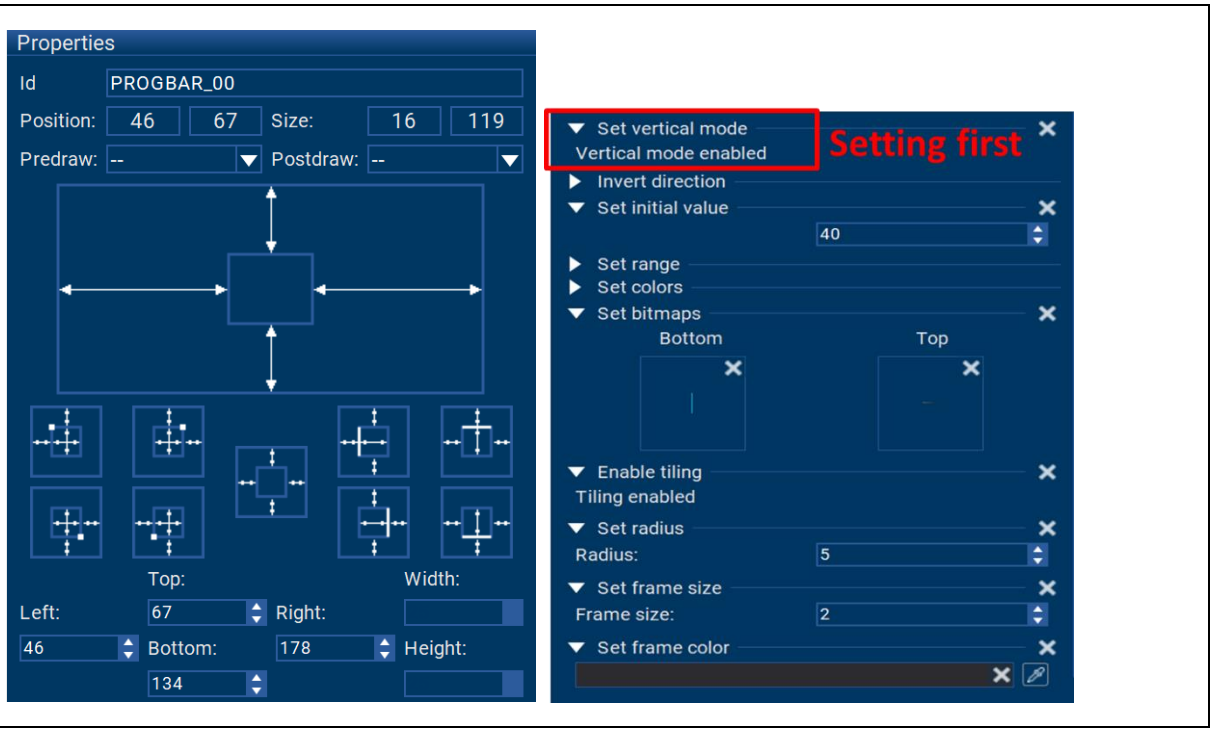


图 2-126 PROGB AR_00 属性设置

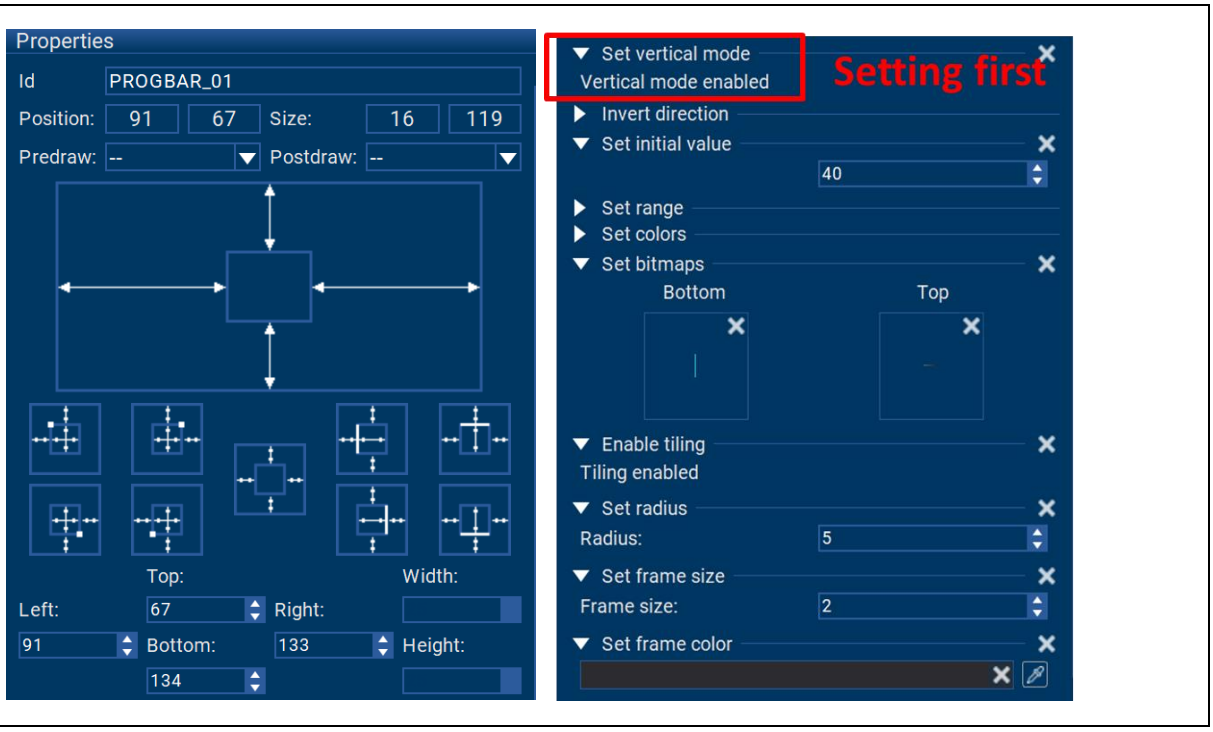


图 2-127 PROGBAR_01 属性设置

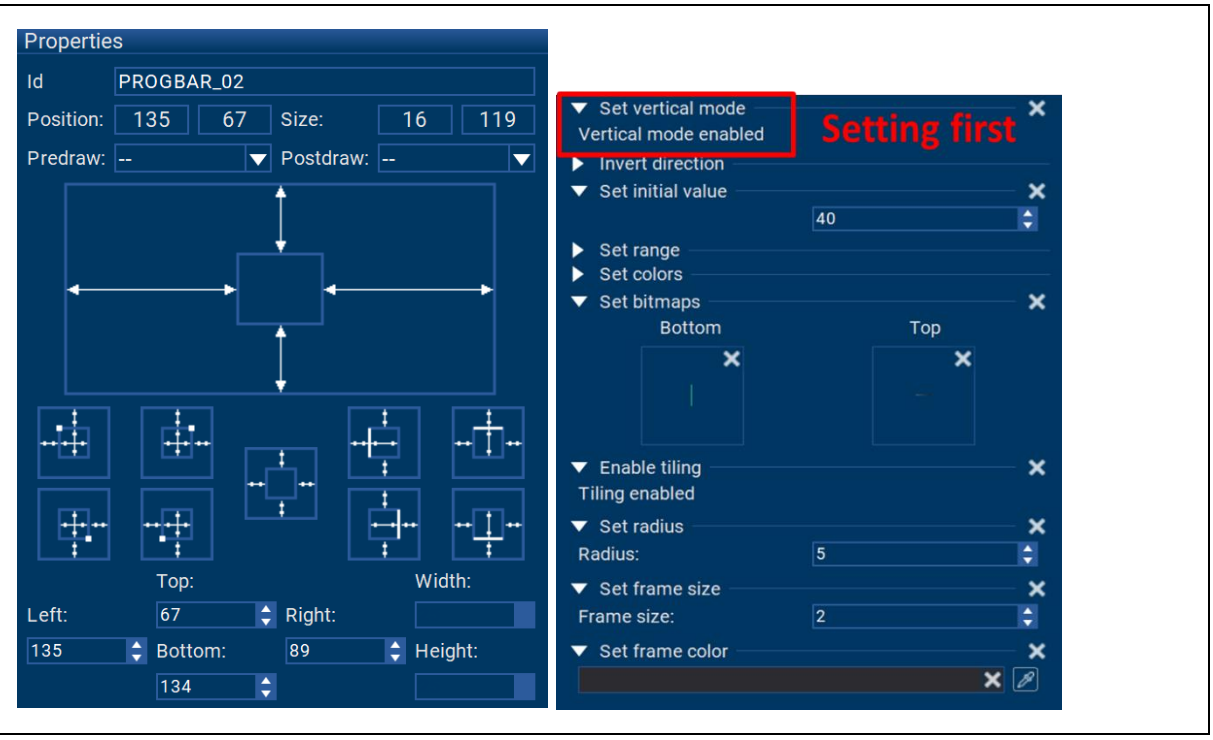


图 2-128 PROGBAR_02 属性设置

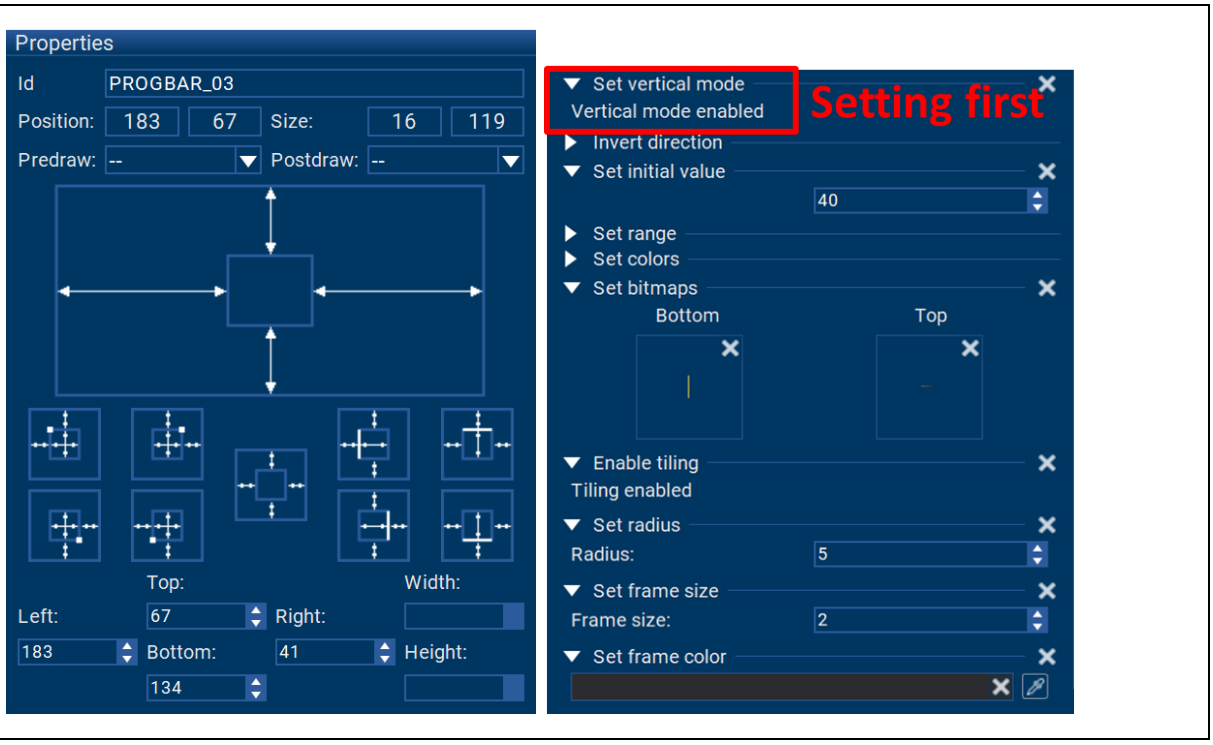


图 2-129 PROGBAR_03 属性设置

4. 新增两个 TEXT 对象作为标题(ID_TEXT_00)与页尾文字(ID_TEXT_01)，将显示文字分别设为”Monitoring Report”与”Accumulation power generation analysis”，并依图 2-132~图 2-133 设置属性。

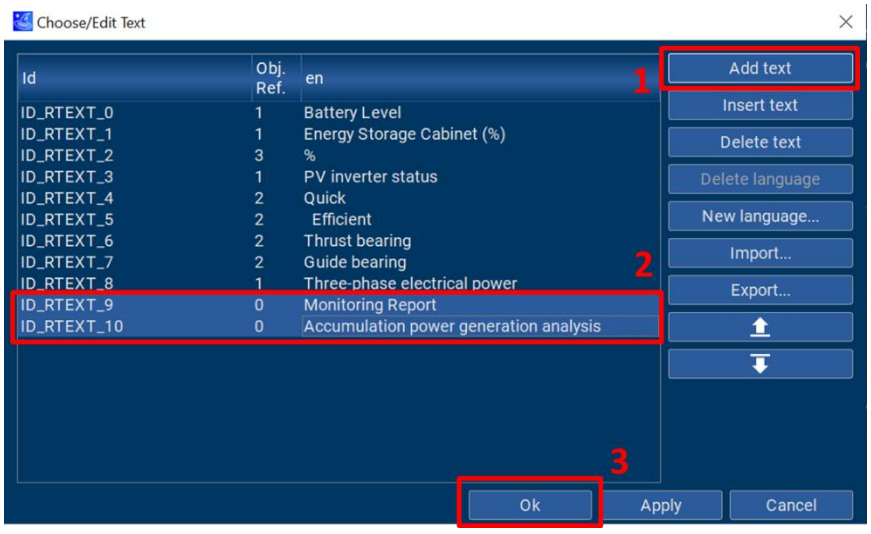


图 2-130 新增显示文字

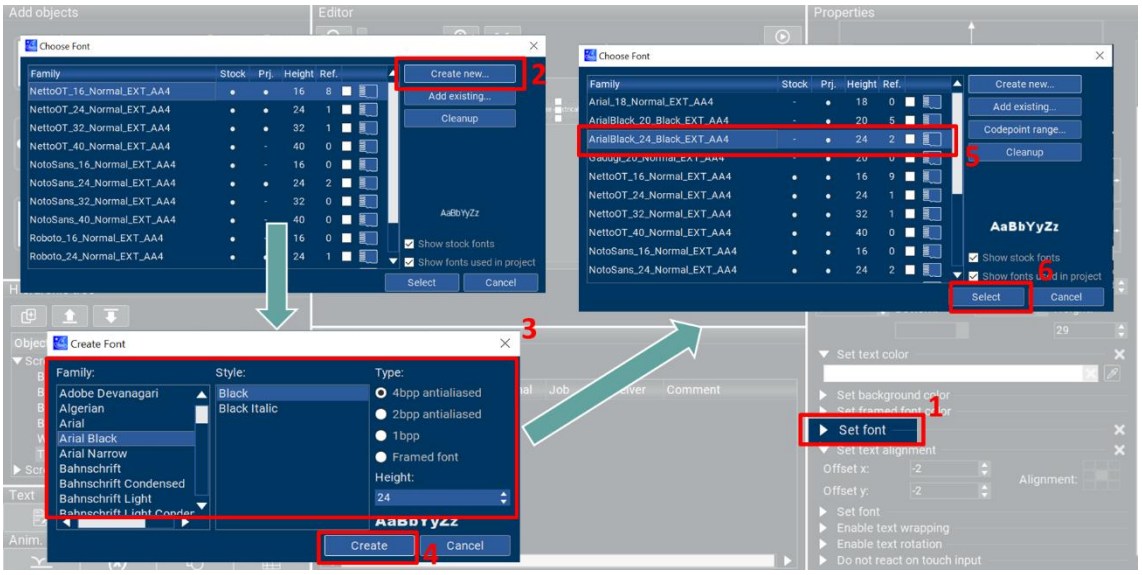


图 2-131 设置自定义字体(ID_TEXT_00)

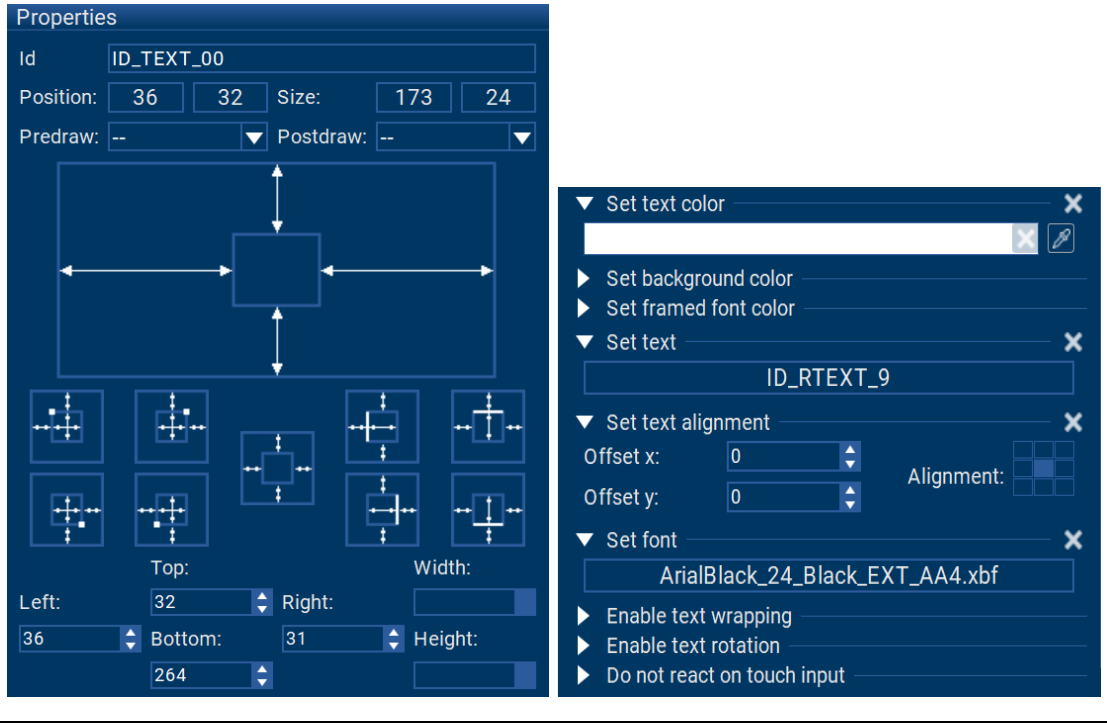


图 2-132 ID_TEXT_00 属性

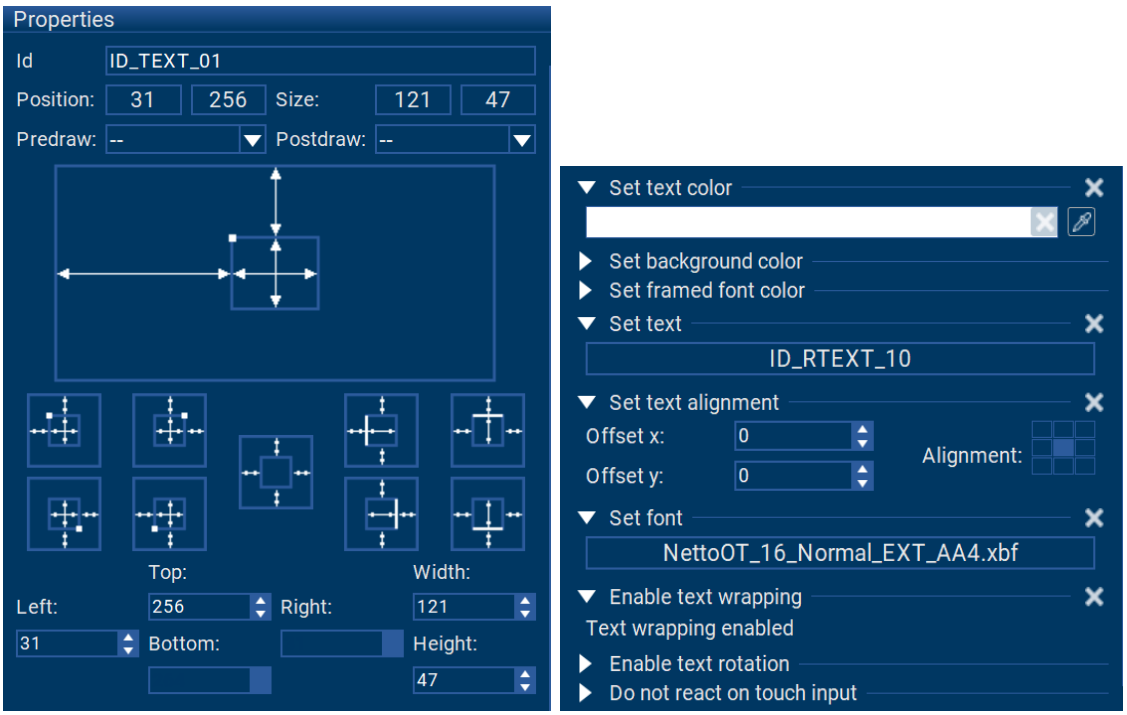


图 2-133 ID_TEXT_01 属性

5. 为了设置各直方图图说，新增四个 TEXT 对象(ID_TEXT_02~ID_TEXT_04)，分别将文字设为”Instant Power (W)”、“Daily (KW)”、“Monthly (MW)”与”Cumulative power(GW)”，依图 2-135~图 2-138 设置属性。

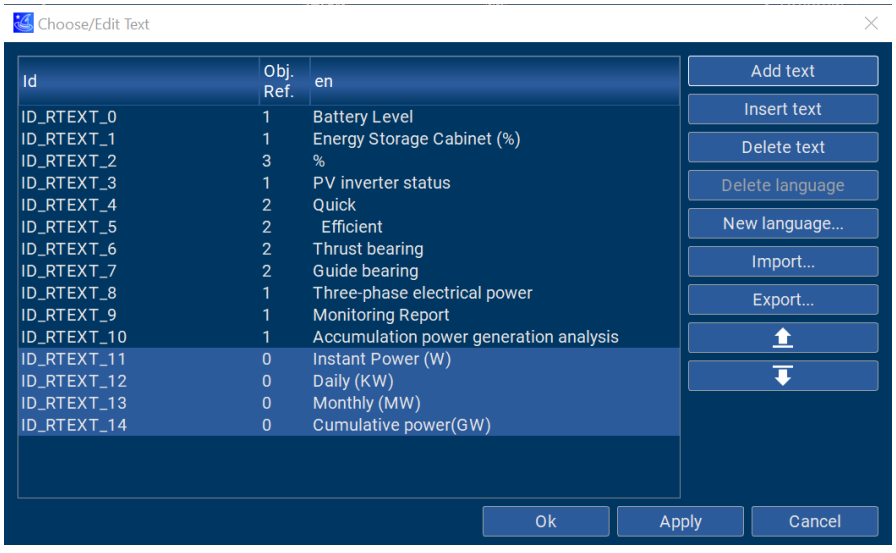


图 2-134 新增显示文字(Progbar 说明)

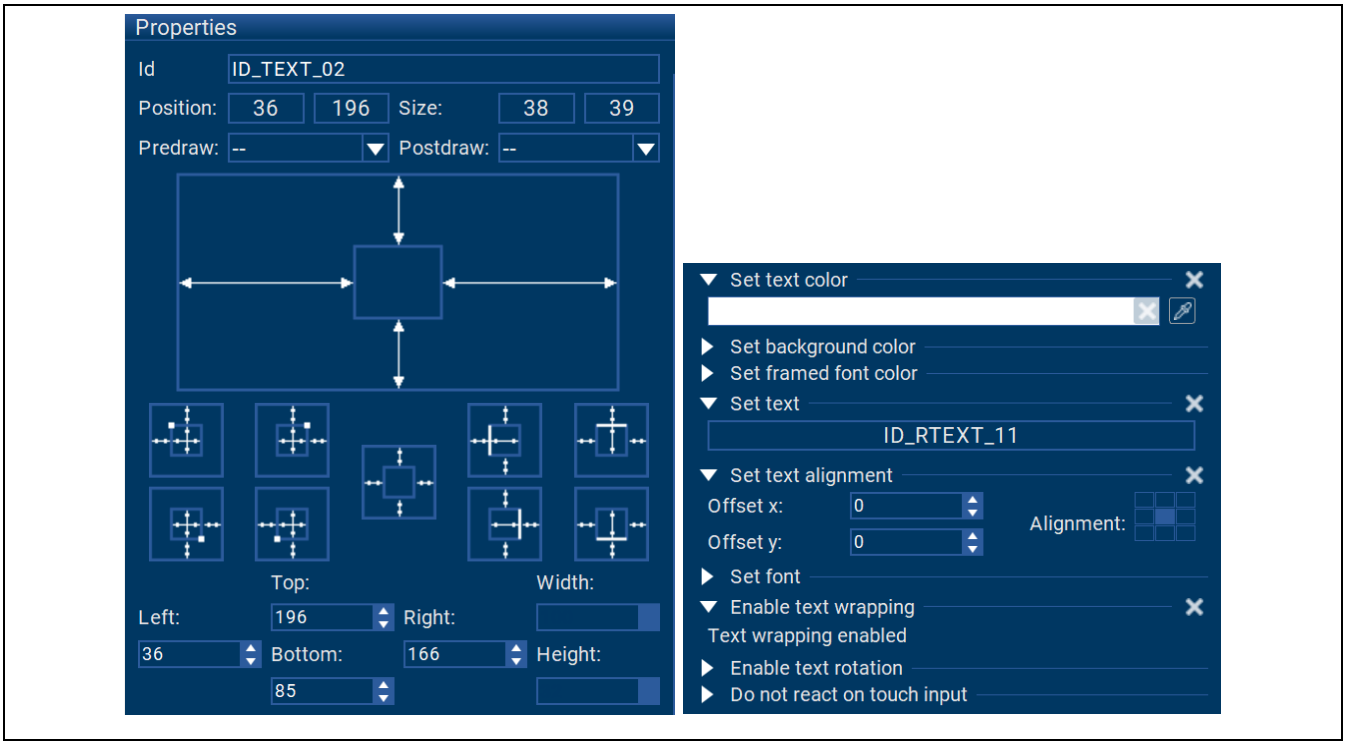


图 2-135 ID_TEXT_02 属性设置

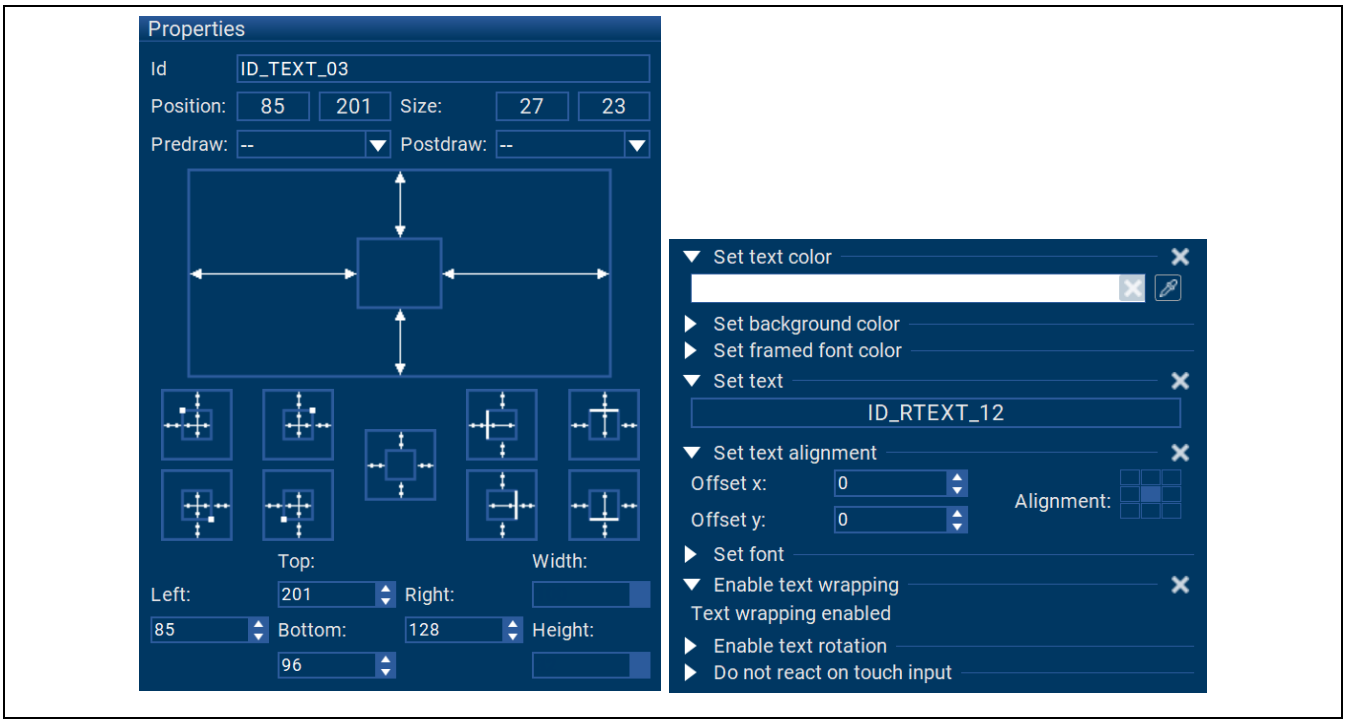


图 2-136 ID_TEXT_03 属性设置

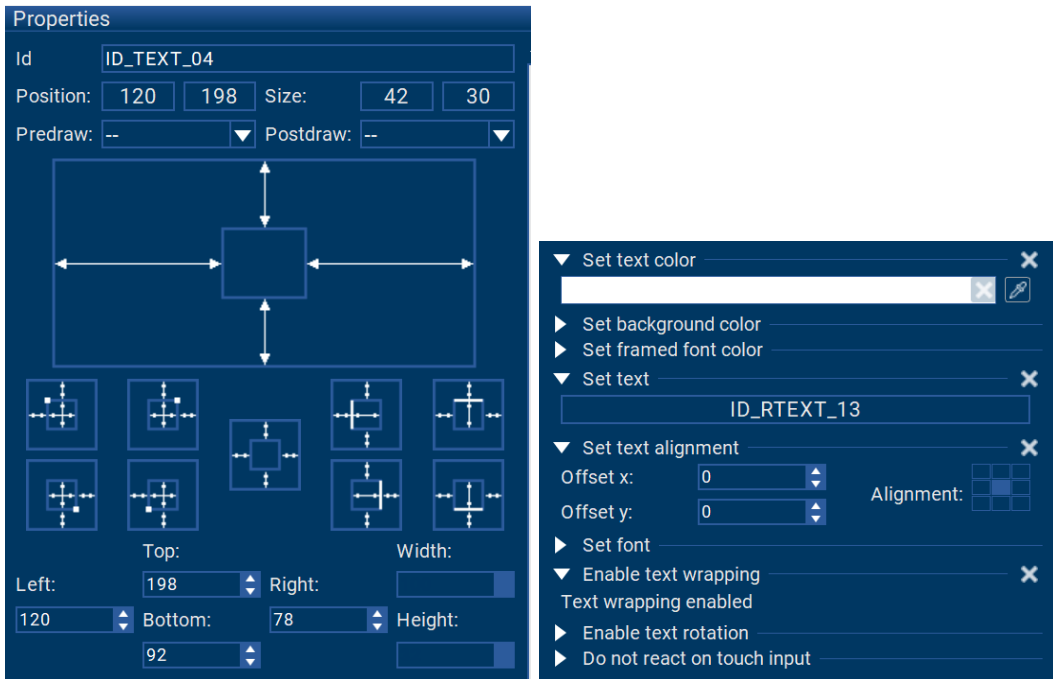


图 2-137 ID_TEXT_04 属性设置

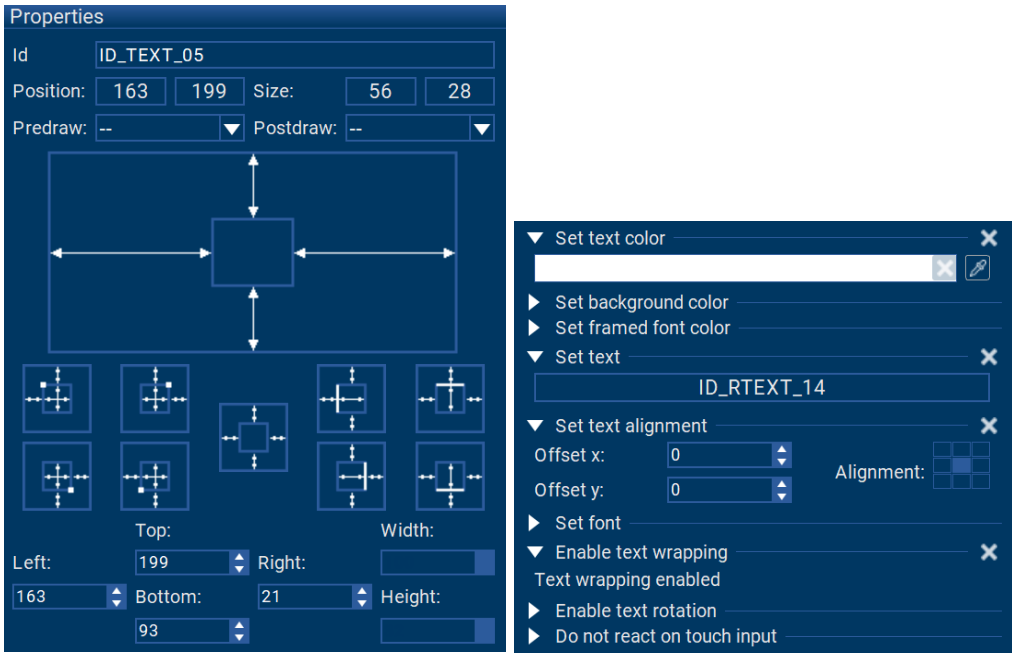


图 2-138 ID_TEXT_05 属性设置

6. 新增一个 QR code 对象(ID_QRCODE)，用户可透过扫描 QR code 导向特定网页。由 Edit Text 内新增网址(*https://www.nuvoton.com/products/microprocessors/arm9-mpus/*)后，依图 2-141 设置属性。

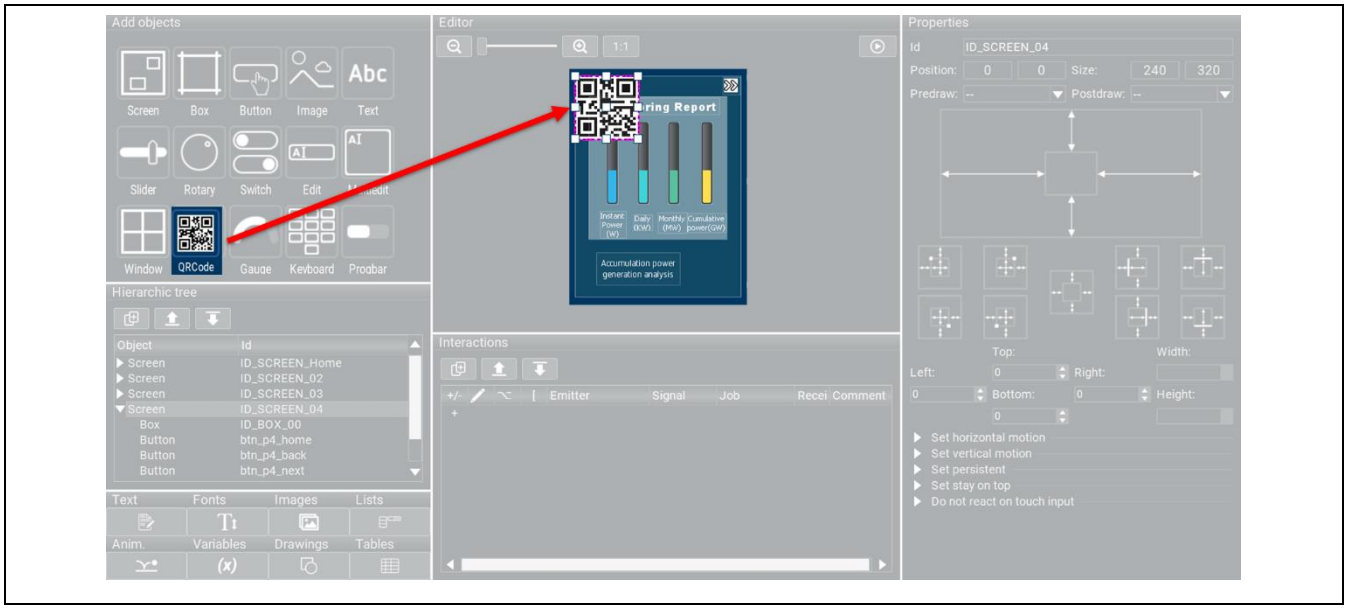


图 2-139 新增 QR Code 物件

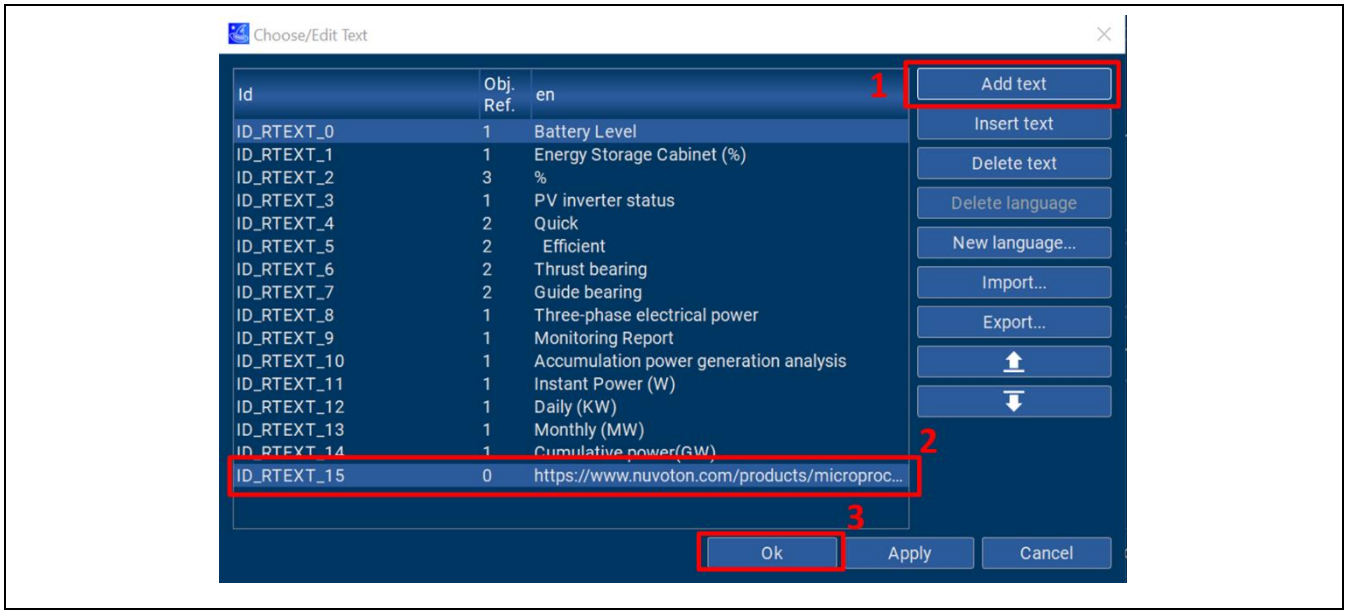


图 2-140 在 Edit Text 新增网址做为内容

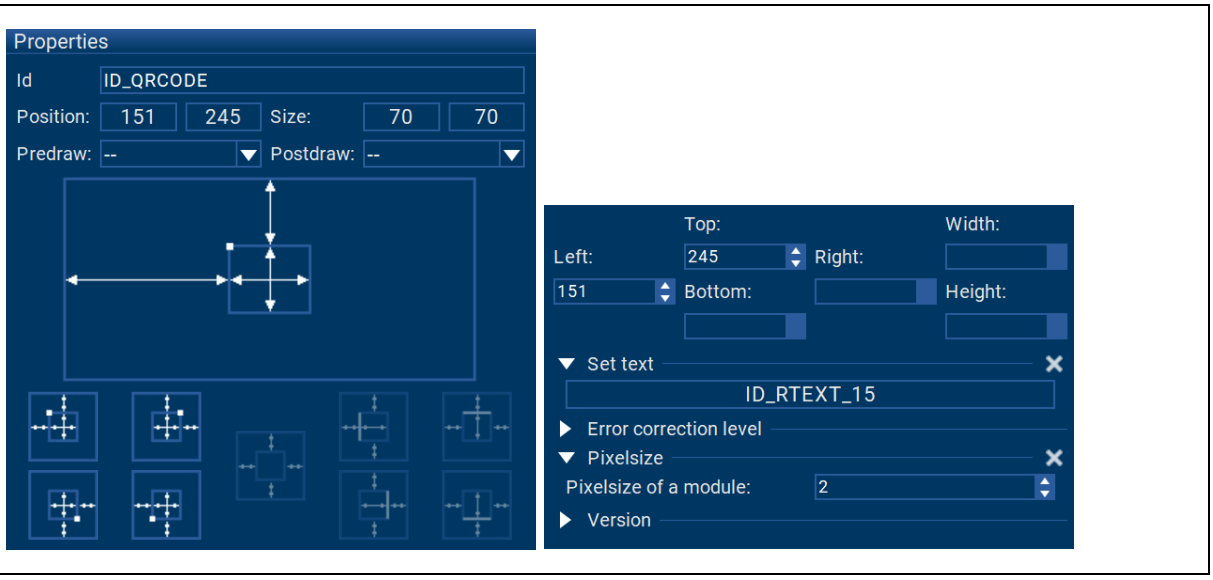


图 2-141 ID_QRCODE 属性设置

2.3.4.2 对象互动设置

对象设置完成画面如图 2-142，接着要建立对象间的交互作用(interactions)，使操作对象时有相应行为显示。

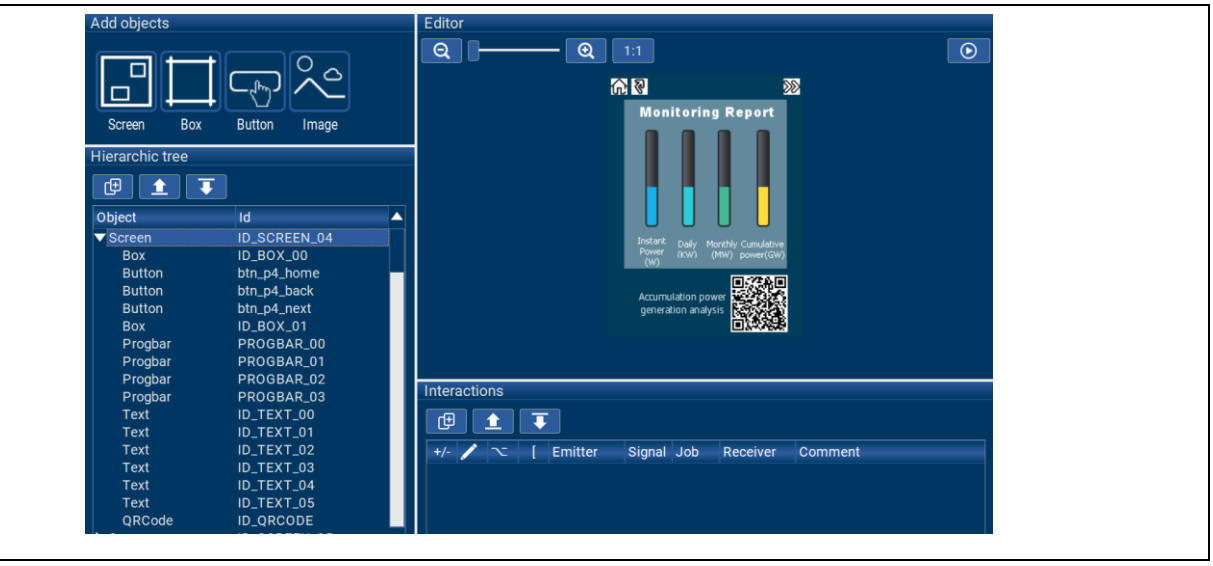


图 2-142 页面四设计完成图

1. 为了使柱形图的数值可以动态呈现，点选左下角 **Anim** 按钮新增动画(ANIM_PW_LINE)，接着依图 2-143~图 2-151 步骤设置动画细节。

   1) 透过 Add item 新增动画后，将其命名为 ANIM_PW_LINE，并设置动画长度(Period)为 2000(ms)。

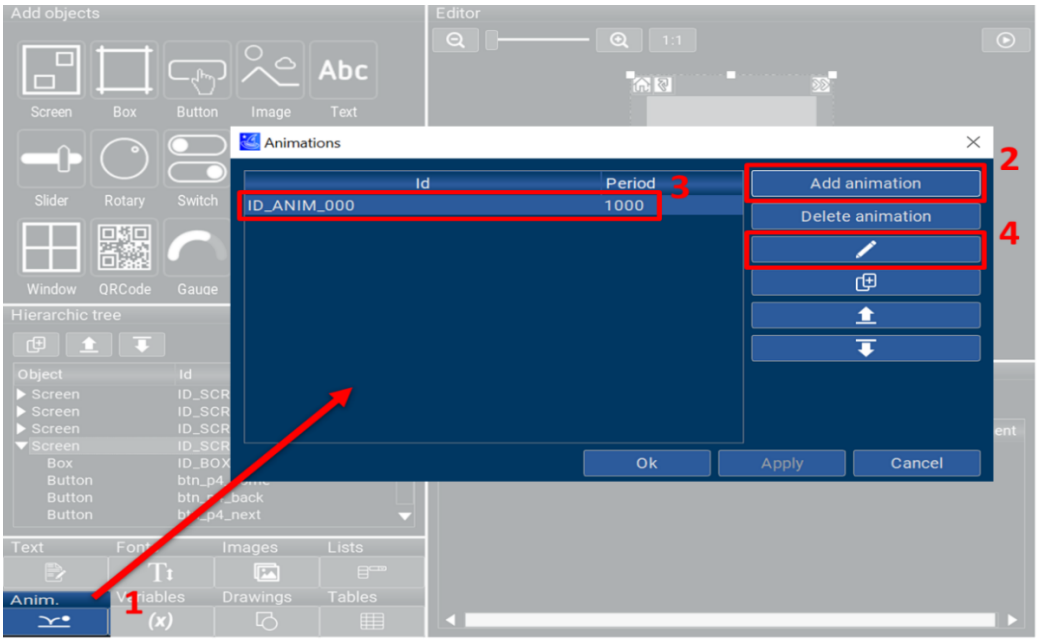


图 2-143 新增动画

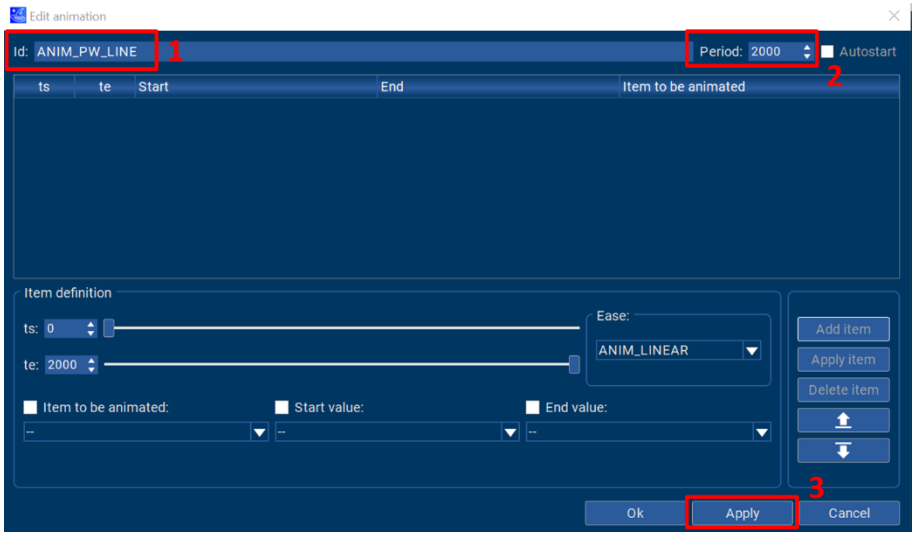


图 2-144 设置动画名称与时间长度

2) 透过 ts 与 te 设置动画的开始与结束时间。

3) 设置响应动画的对象 ID 与变化起始值、结束数值与 Ease(ANIM_ACCELDECEL)。

4) 点选 Add item 即可将此动作新增到动画 ANIM_PW_LINE 中。

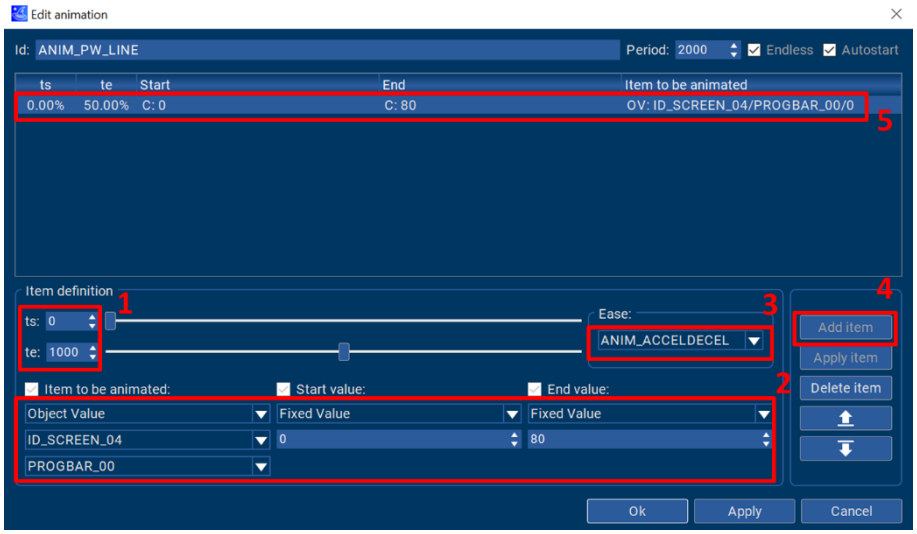


图 2-145 新增 PROGBAR_00 动画

5) 同步骤 1)~4)，设置其它 Progbar 对象(PROGBAR_01~ PROGBAR_03)的动画。

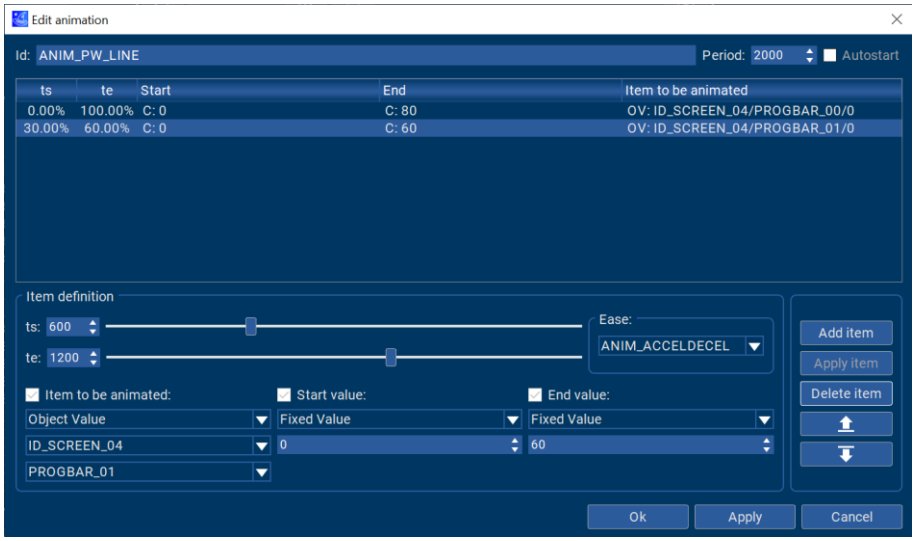


图 2-146 新增 PROGBAR_01 动画

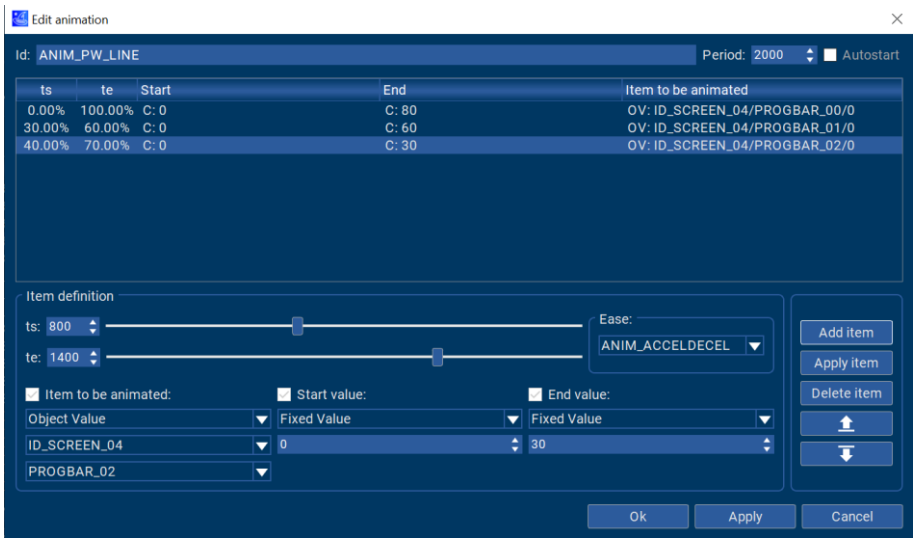


图 2-147 新增 PROGBAR_02 动画

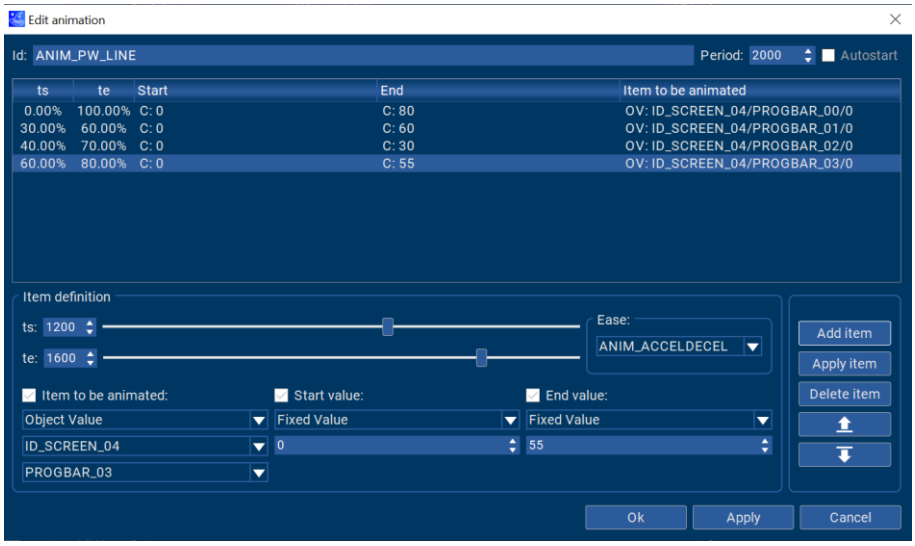


图 2-148 新增 PROGBAR_03 动画

6) 所有动作设置完成后，勾选「Autostart」与「Endless」使动画自动开始并无限循环。

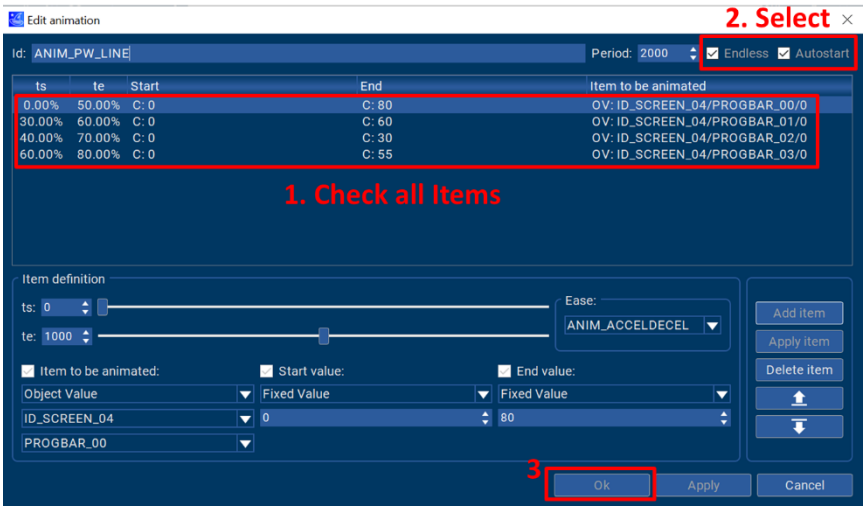


图 2-149 设置动画自动开始

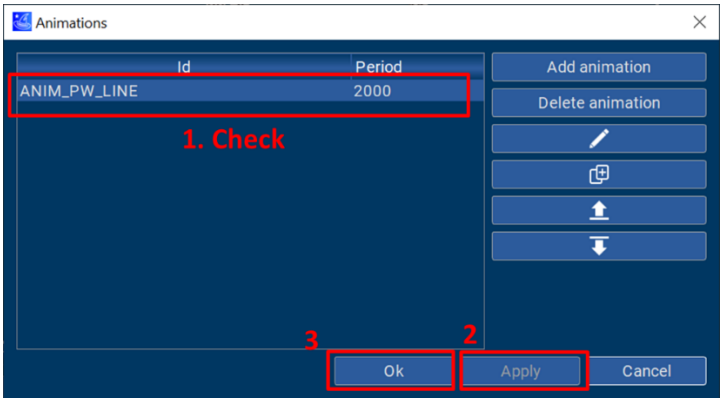


图 2-150 储存 ANIM_PW_LINE 动画设置

7) 在 interactions 字段新增:Emitter 为 ID_SCREEN_04、Signal 为 INITDIALOG、Job 为 ANIMCREATE 的项目，将 Animation id 设为 ANIM_PW_LINE。

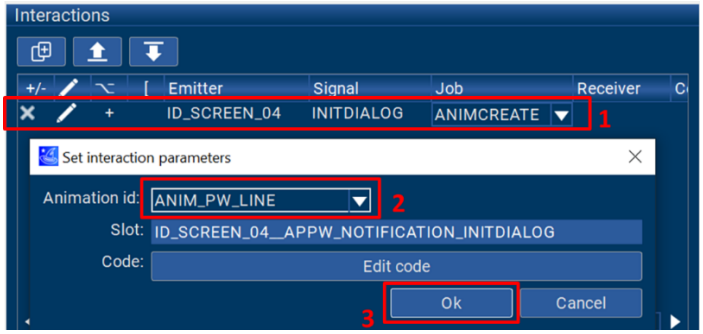


图 2-151 设置动画显示

2. 设置 Button(btn_p4_home、btn_p4_back、btn_p4_next)对象的换页功能，使点选按钮可以切回首页、前一页及下一页。

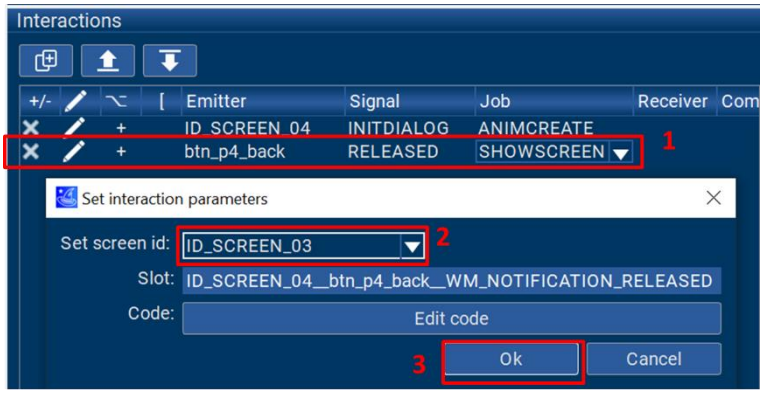


图 2-152 btn_p4_back 换页设置

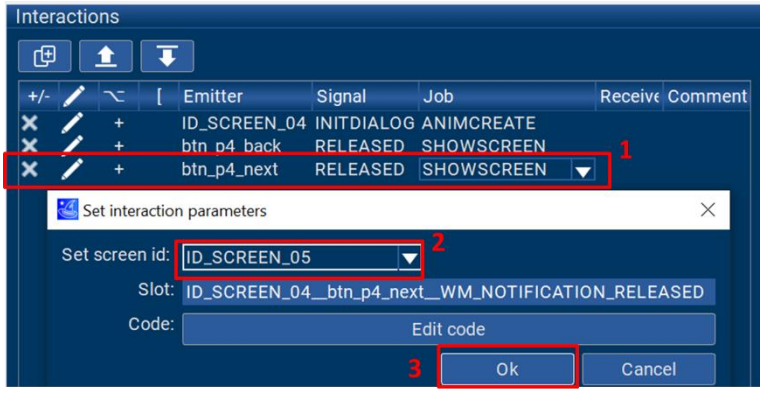


图 2-153 btn_p4_next 换页设置

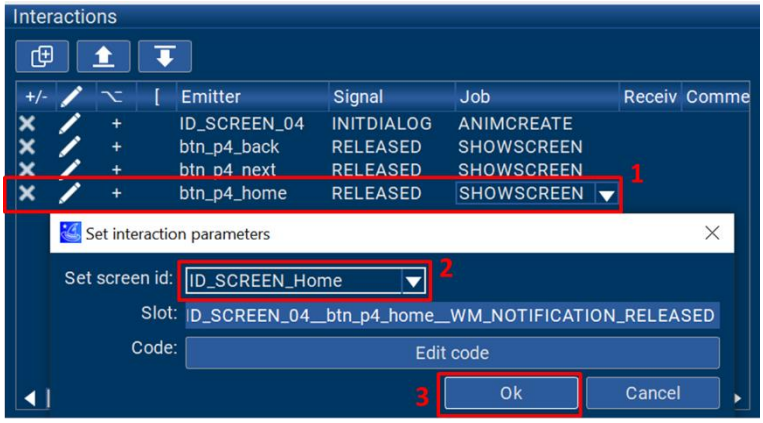
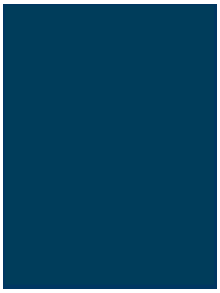
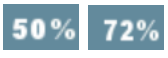


图 2-154 btn_p4_home 换页设置

2.3.5 转换率调整画面设计(页面五)

网关产品中不乏能源转换、逆变器等应用，除了可透过简单的 Slider 设定转换率调整外，本设计同时设置 Switch 作为安全开关(锁定调整率)，以降低人员操作不慎、误触时导致的损坏，此设计所用对象如表 2-7 所示。

New Energy Gateway Description	Application GUI	Widget Type
背景图 240x320		Box
拉霸背景图 150x254		Box
标题文字		Text
百分比显示		

转换幅度调整拉霸		Slider
锁定开关		Switch
档案上传按钮		Button
回主页按钮		
上一页按钮		

表 2-7 转换率调整设计对象

2.3.5.1 GUI 对象配置

1. 点选 ID_SCREEN_05 进行页面五的设计，透过 Add objects 新增第二个 Box 对象 (ID_BOX_01)作为拉霸背景，并调整齐属性。

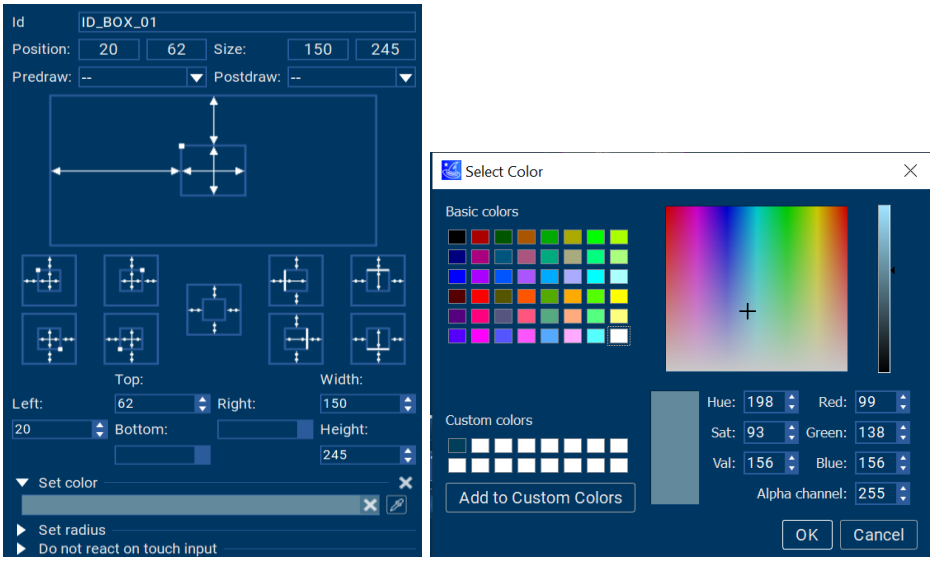


图 2-155 ID_BOX_01 属性设置

2. 新增五个 Text 对象，分别作为标题(ID_TEXT_00，显示文字为 Amplitude adjustment)、两个拉霸数值(ID_TEXT_01、ID_TEXT_03)与两个百分比符号(ID_TEXT_02、ID_TEXT_04)，依图 2-157~图 2-161 设置属性。

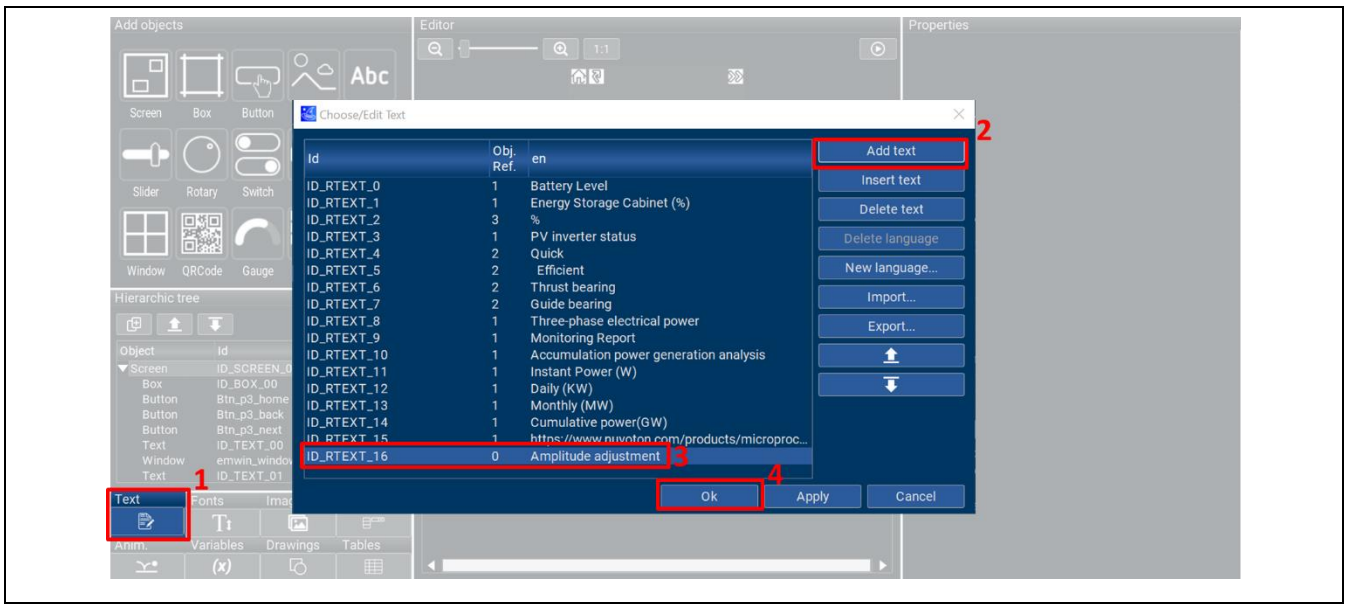


图 2-156 新增 ID_TEXT_00 显示文字

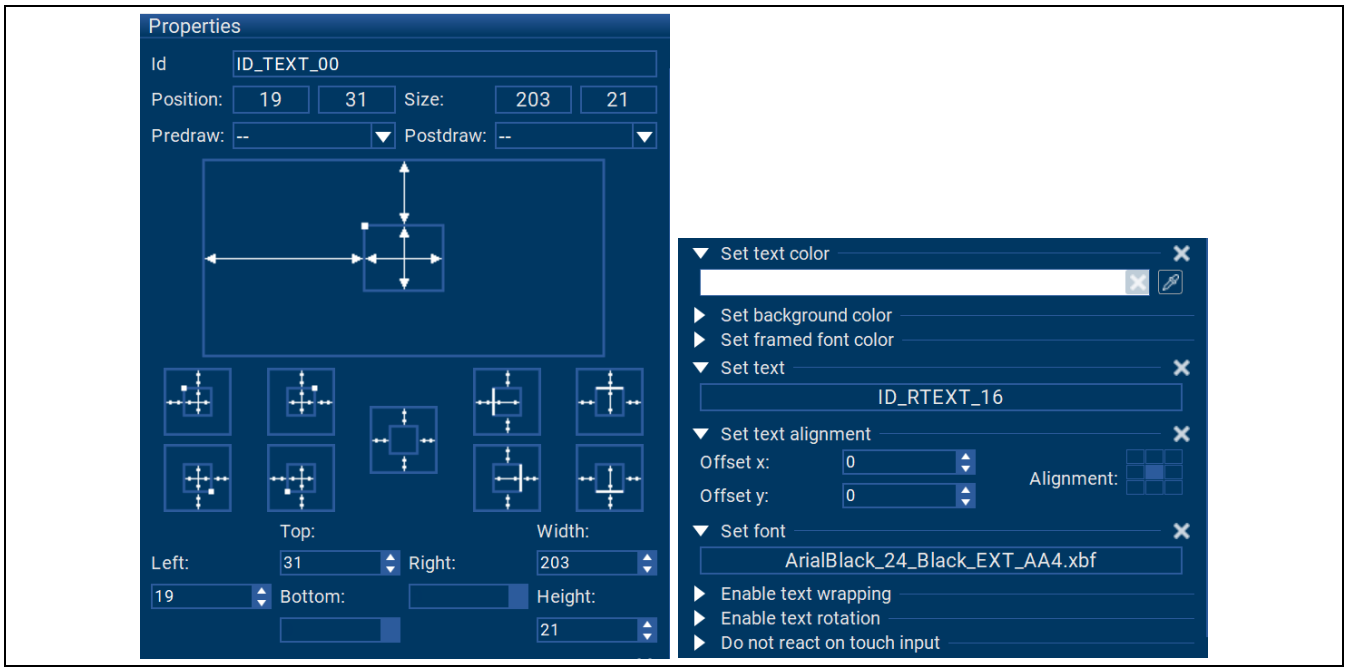


图 2-157 ID_TEXT_00 属性设置

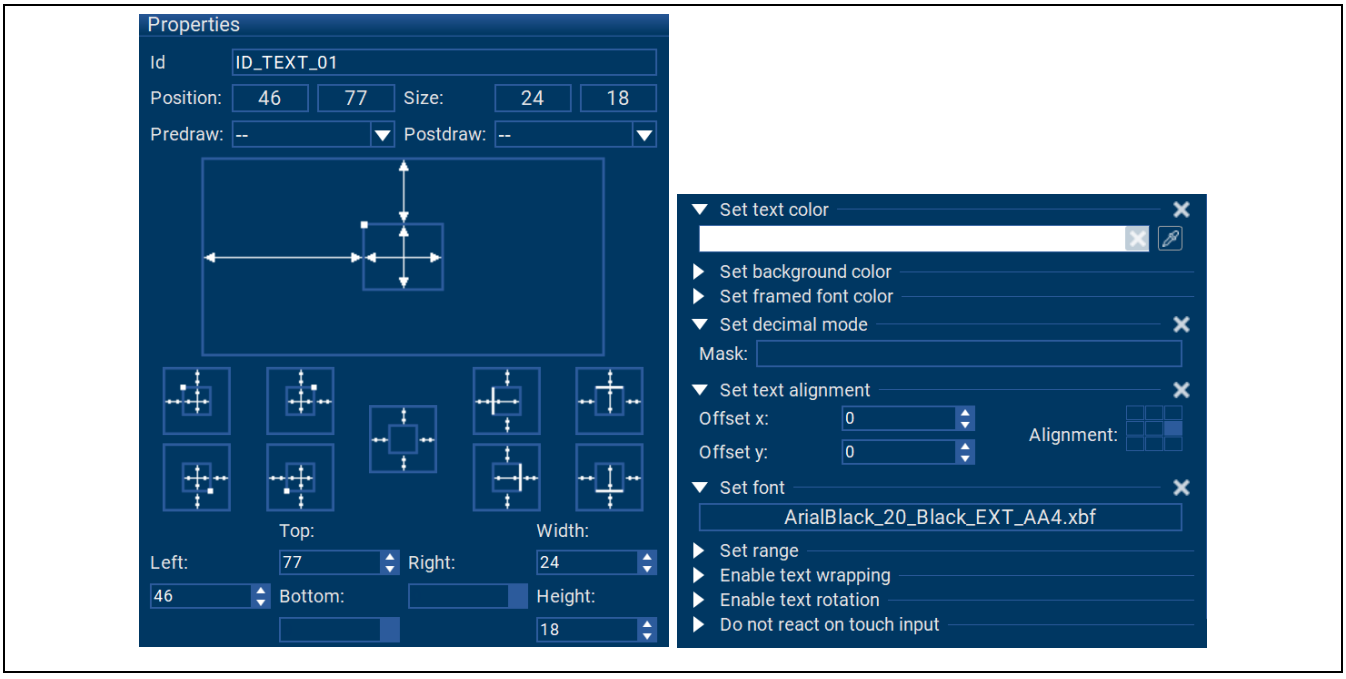


图 2-158 ID_TEXT_01 属性设置

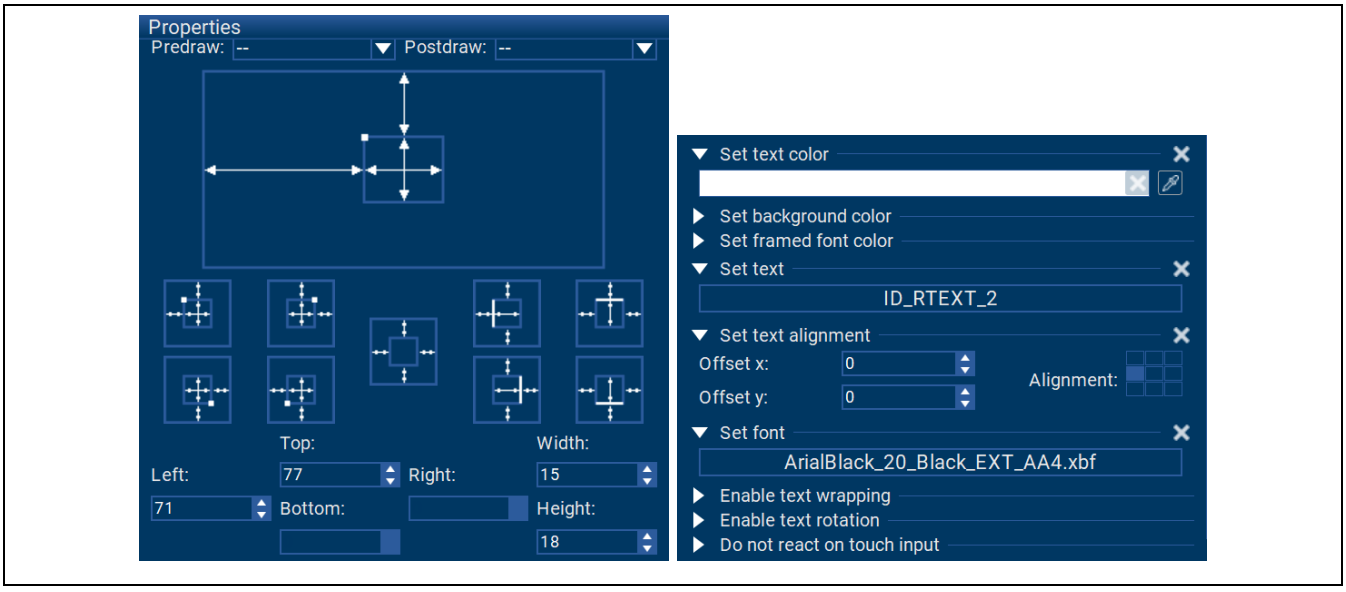


图 2-159 ID_TEXT_02 属性设置

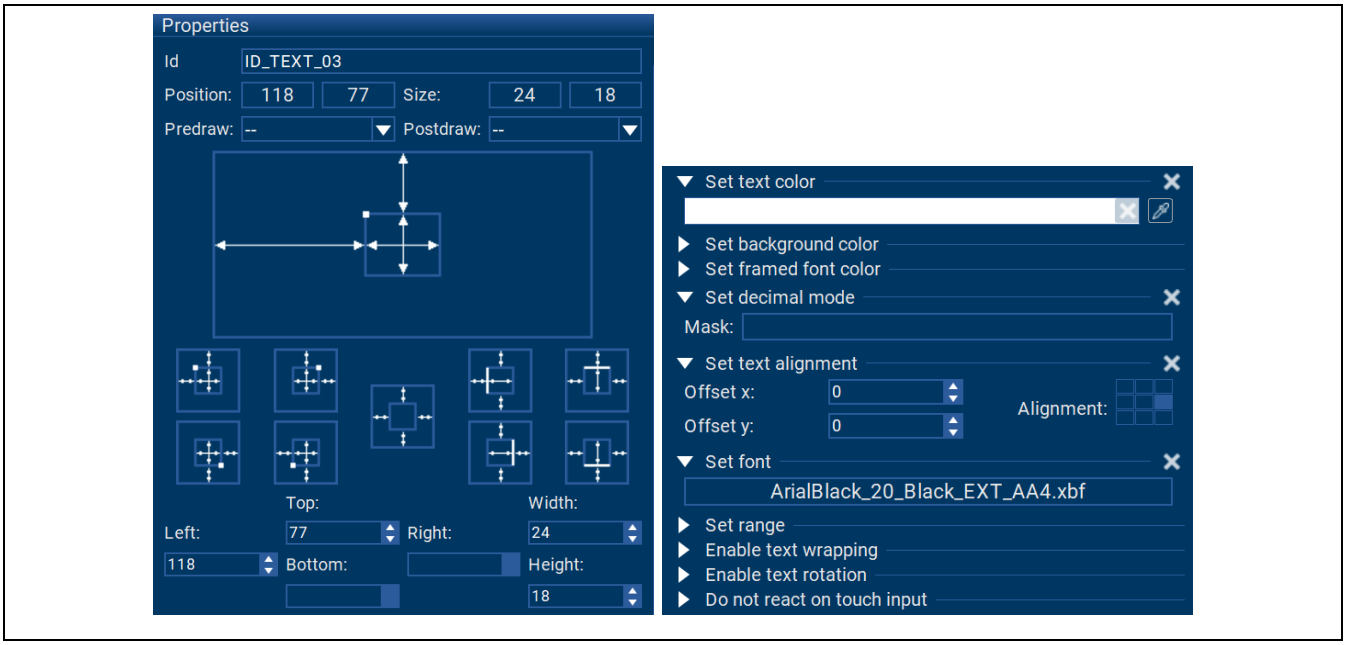


图 2-160 ID_TEXT_03 属性设置

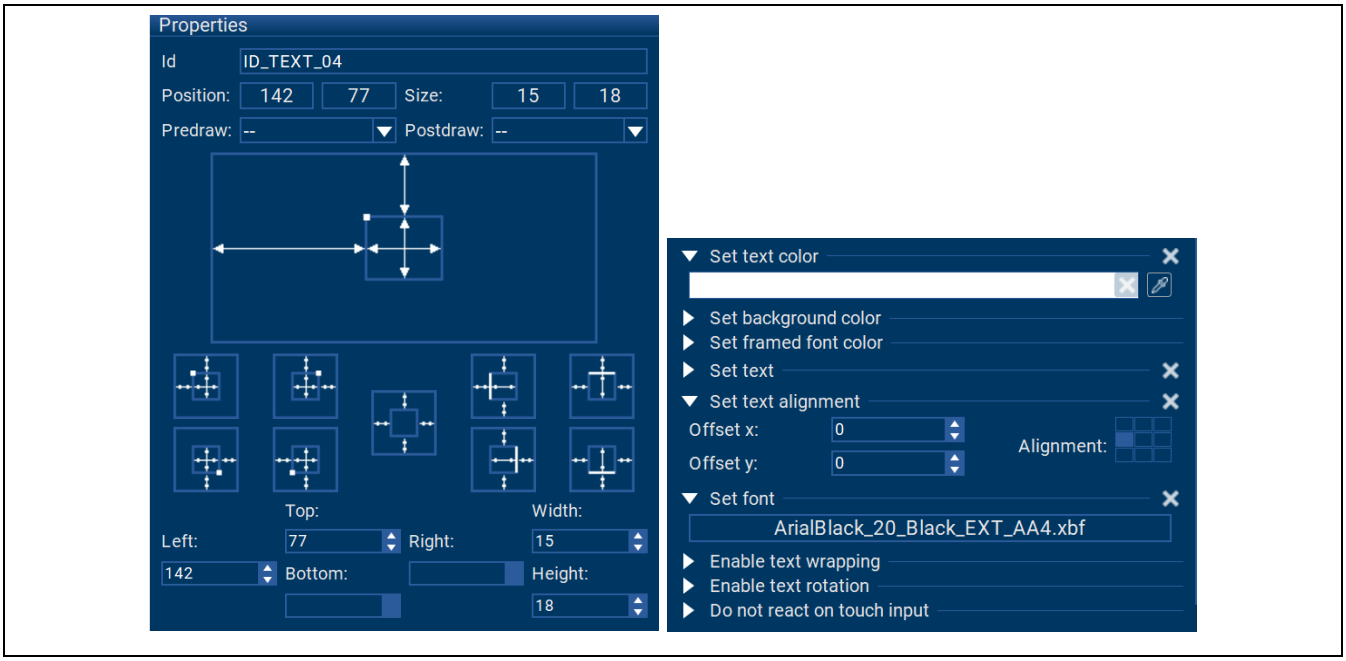


图 2-161 ID_TEXT_04 属性设置

3. 新增两个 slider 对象(SLIDER_T1、SLIDER_T2)做数值调整，在属性栏开启 Vertical mode 与 Invert direction 功能(点击箭头符号开启)，使 slider 外观垂直、数值由下往上递增，并以图片 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 分别做为 bitmap 中的 Shaft bottom、Shaft top、Thumb unpressed 与 Thumb pressed。

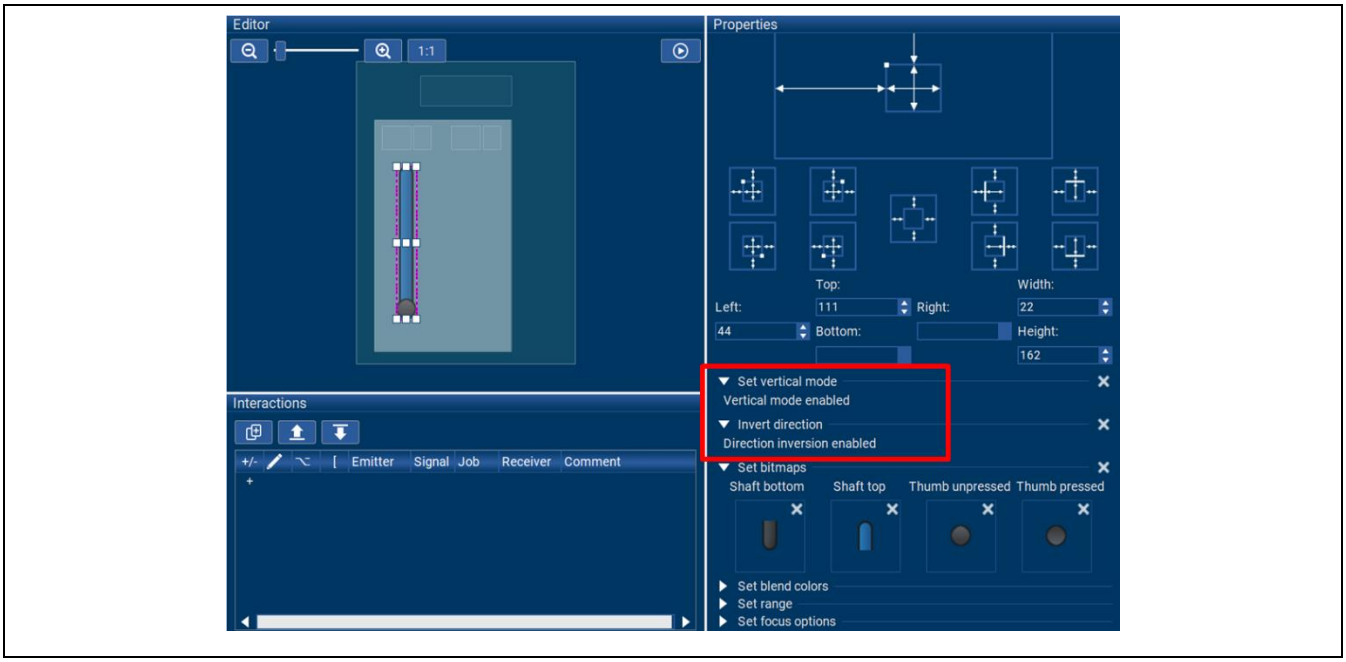


图 2-162 设置 Vertical mode 与 Invert direction

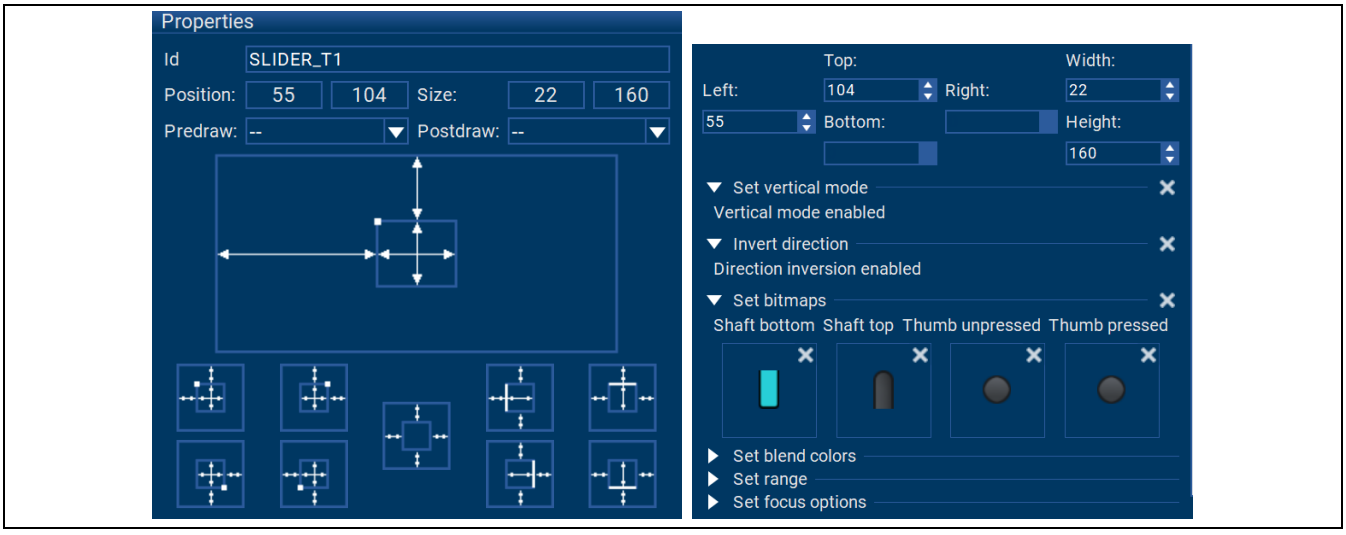


图 2-163 SLIDER_T1 属性设置

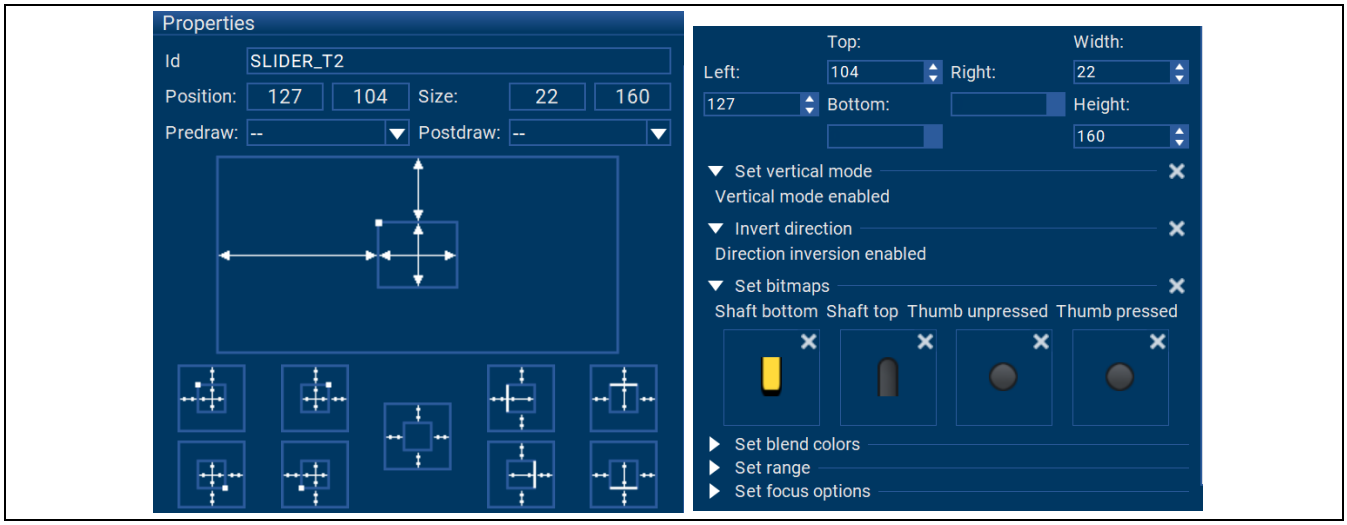


图 2-164 SLIDER_T2 属性设置

4. 为了设置 slider 调整数值的开关，新增一个 Switch 对象(SWITCH_locker)，以图片 Switch_Button_Pressed_68x22.png 与 Switch_Button_Unpressed_68x22.png 作为外观，并依图 2-165 ~ 图 2-166 设置属性。

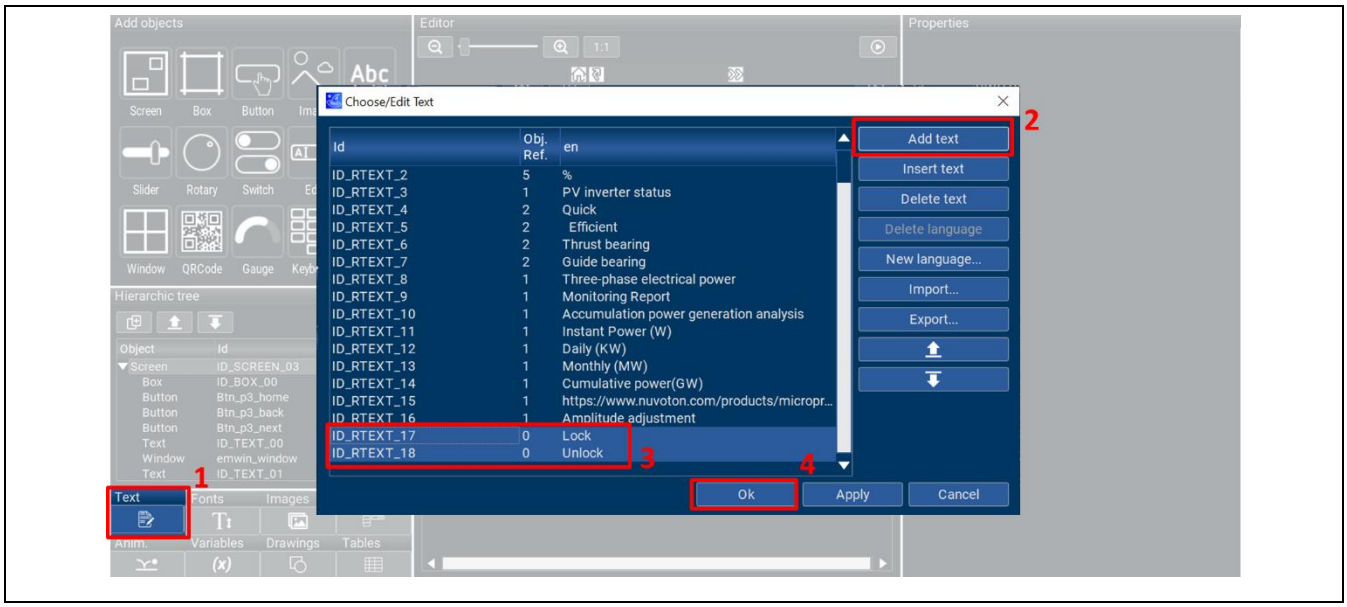


图 2-165 新增 SWITCH_locker 显示文字(Lock、Unlock)

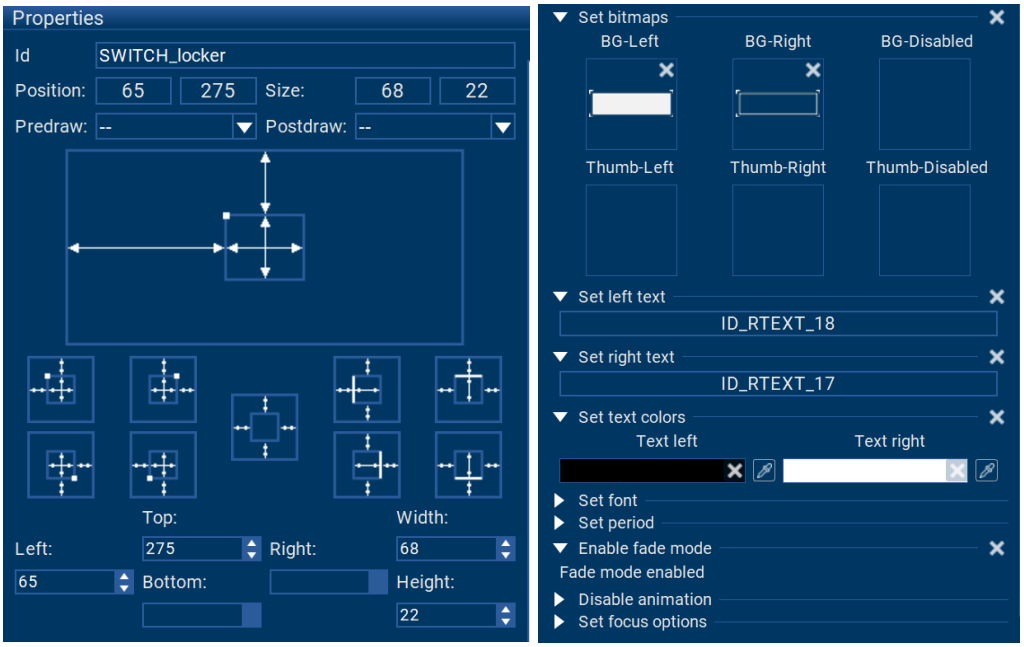


图 2-166 SWITCH_locker 属性设置

5. 新增三个 button 对象做换页(btn_p5_home、btn_p5_back)与数据上传(btn_upload)功能，属性如图 2-167~图 2-170，并以图片 home_20x20_new.png、arrow2_20x20_new.png、Squere_Button_Unpressed_40x40.png 与 Squere_Button_Pressed_40x40.png 设置外观。

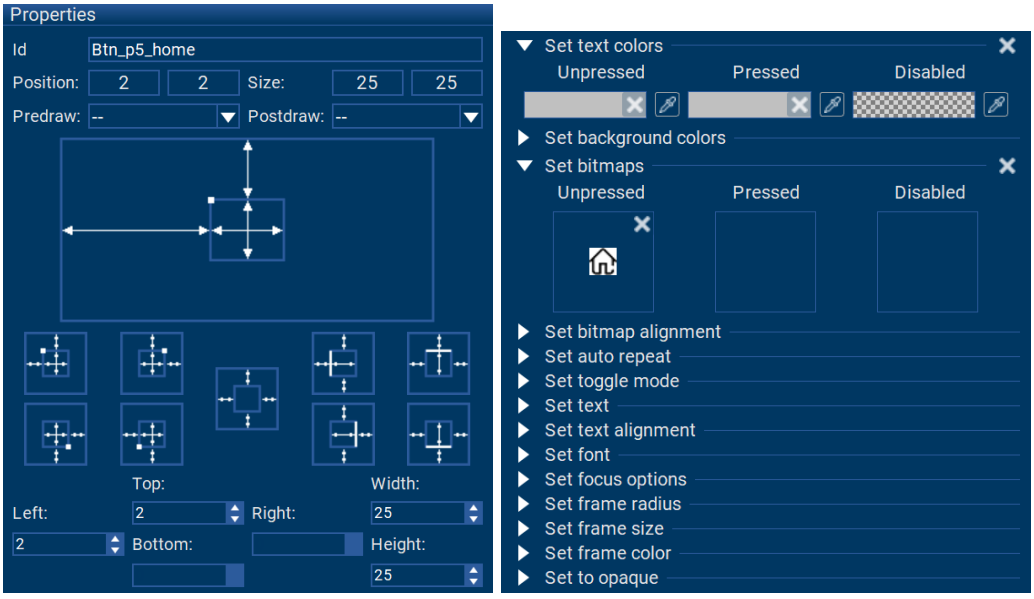


图 2-167 btn_p5_home 属性设置

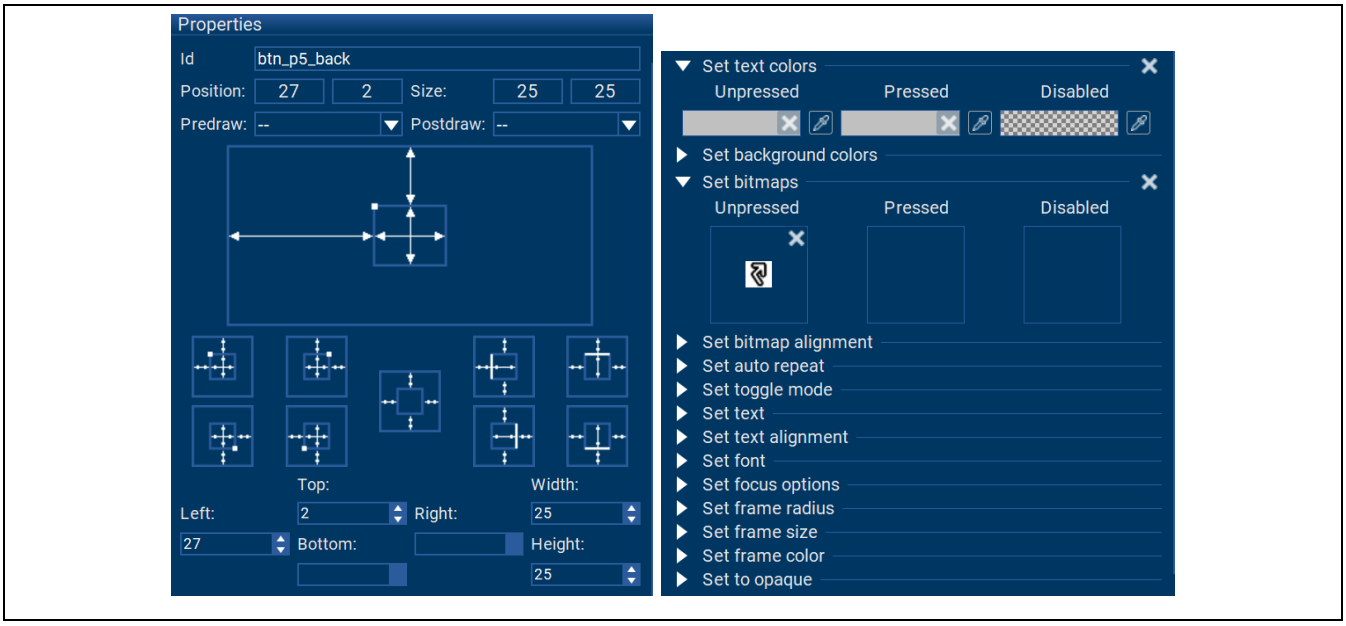


图 2-168 btn_p5_back 属性设置

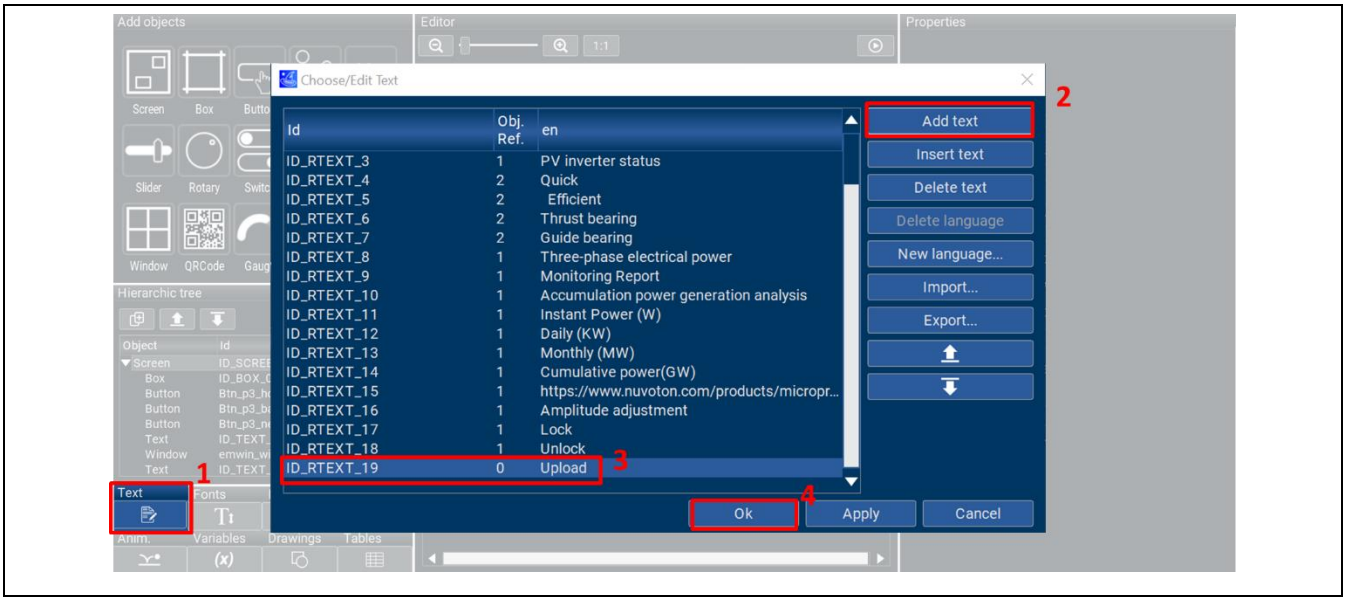


图 2-169 新增 btn_upload 显示文字(Upload)

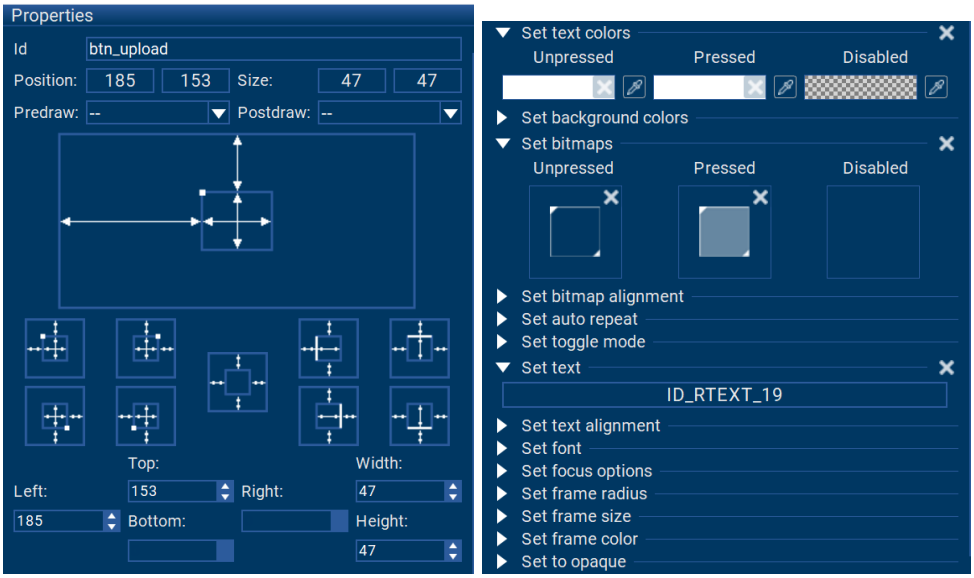


图 2-170 btn_upload 属性设置

2.3.5.2 对象互动设置

对象设置完成画面如图 2-171 所示，接着要建立对象间的交互作用(interactions)，使操作对象时有相应行为显示。

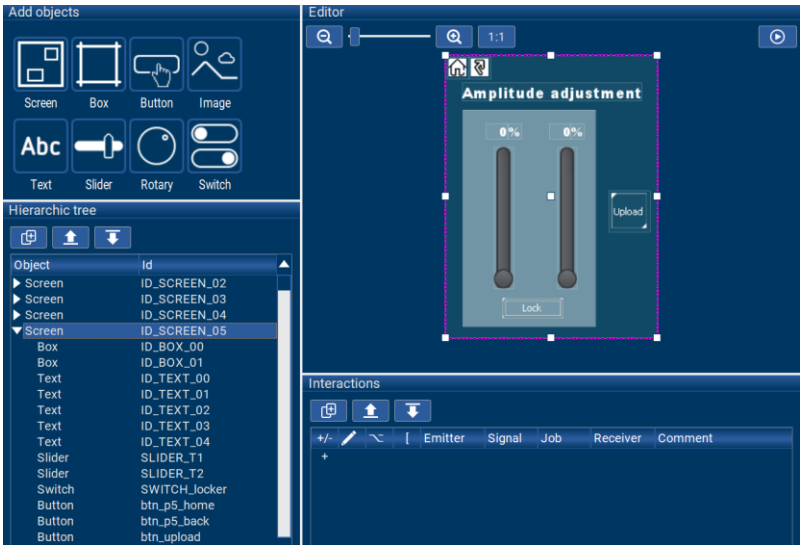


图 2-171 页面五设计完成图

1. 设置 Slider 与 Text 之间的连动，使 TEXT 对象显示 Slider 调整后的数值，功能设置图 2-172~图 2-173。图 2-174、图 2-176 设置开启 switch 后关闭 slider 对象的功能；图 2-175、图 2-177 则设置关闭 switch 后重新使能 slider 对象的功能。

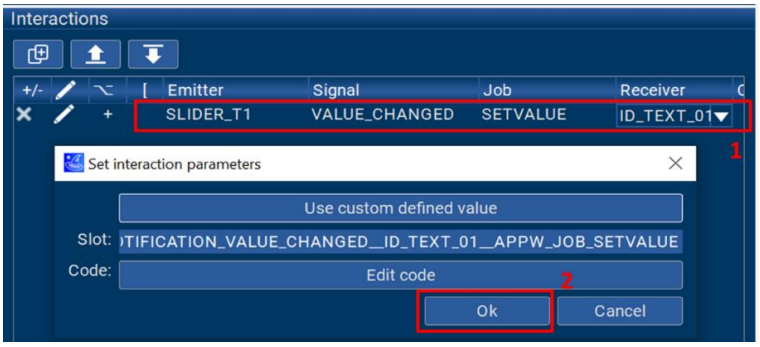


图 2-172 设置 SLIDER_T1 与 ID_TEXT_01 互动

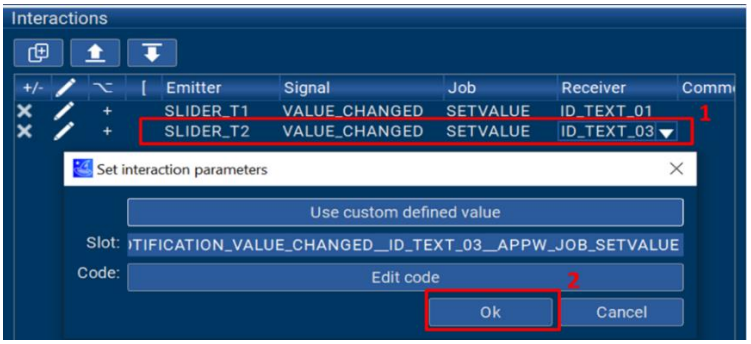


图 2-173 设置 SLIDER_T2 与 ID_TEXT_03 互动

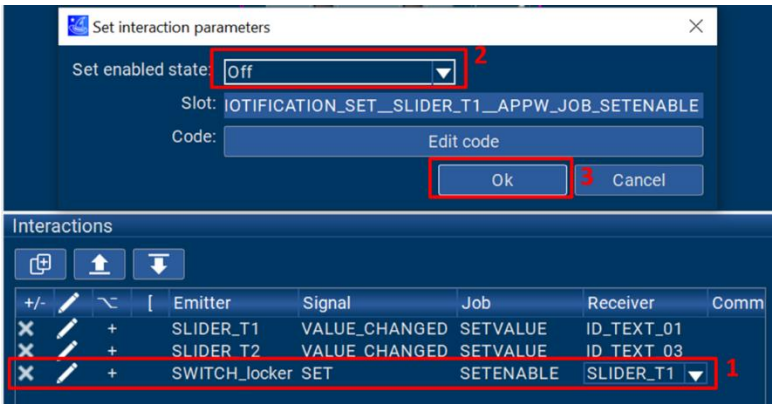


图 2-174 设置 SLIDER_T1 与 Switch_locker 互动 1

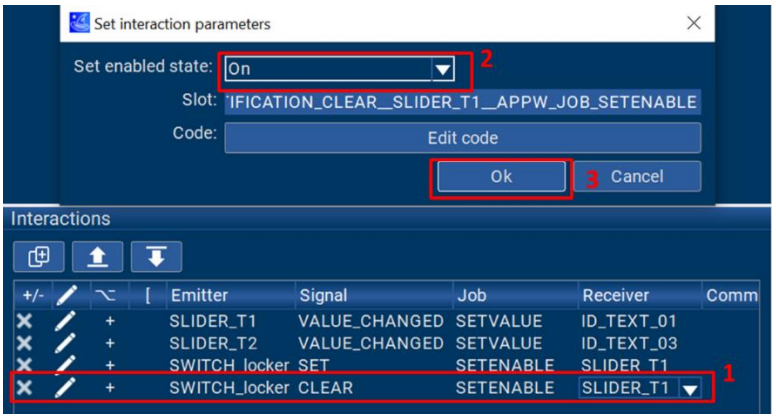


图 2-175 设置 SLIDER_T1 与 Switch_locker 互动 2

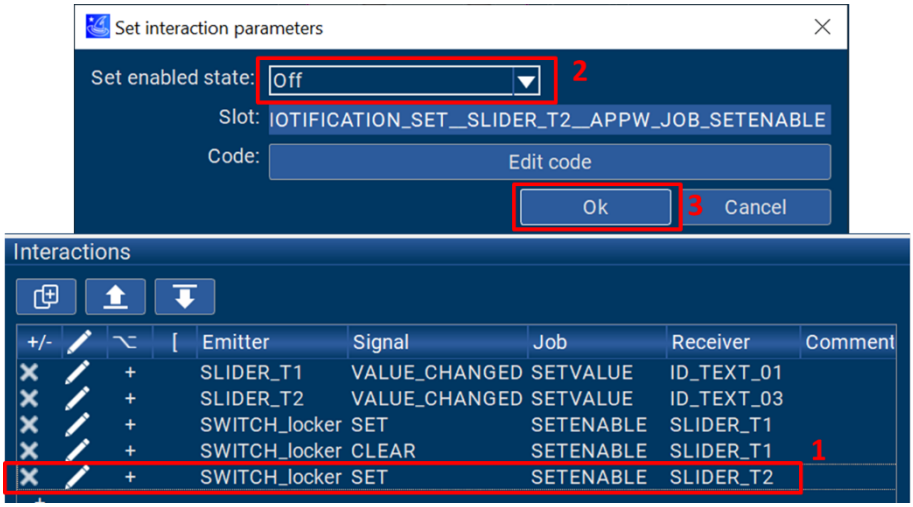


图 2-176 设置 SLIDER_T2 与 Switch_locker 互动 1

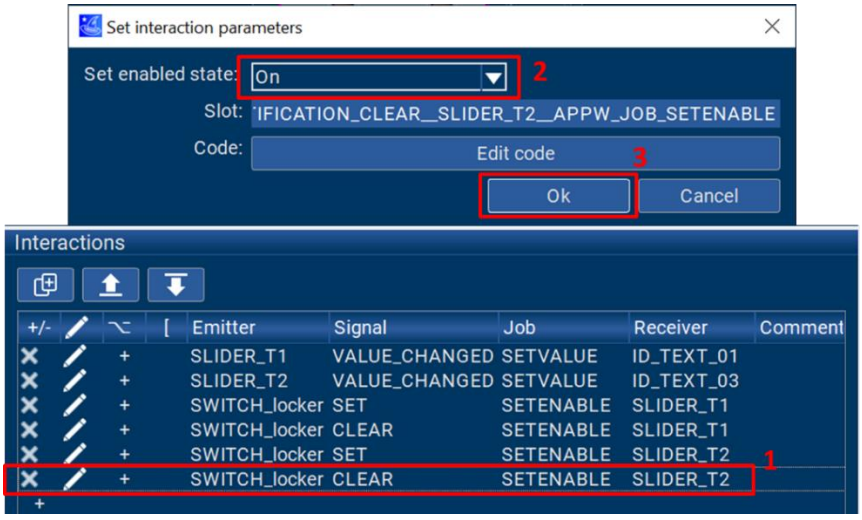


图 2-177 设置 SLIDER_T2 与 Switch_locker 互动 2

2. 设置 Button 对象(btn_p5_home、btn_p5_back)换页功能，透过按钮切换显示页面。

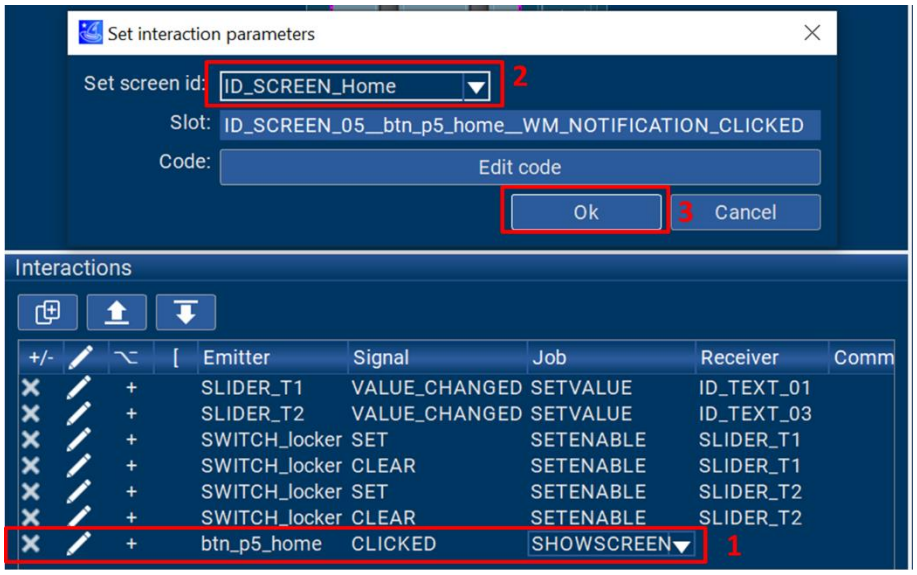


图 2-178 btn_p5_home 换页设置

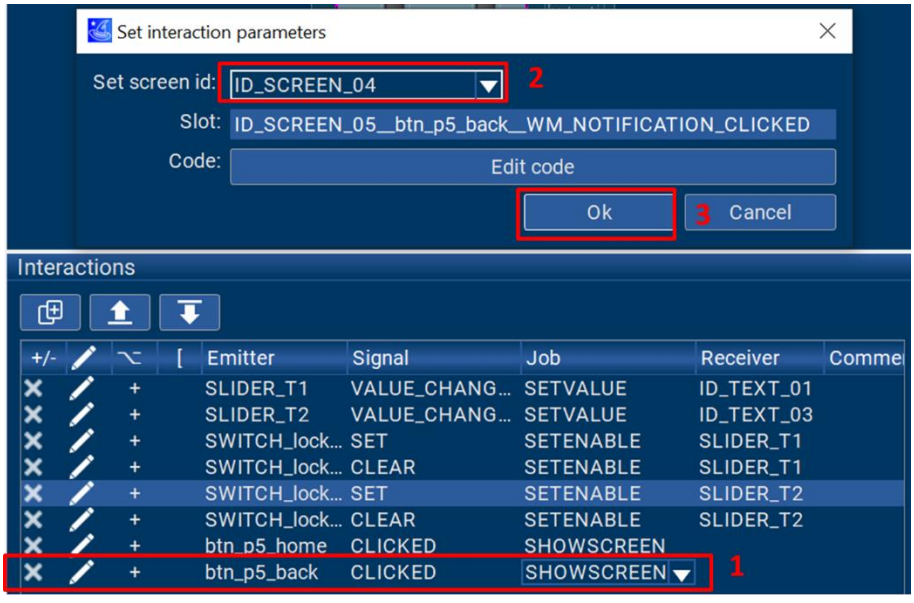


图 2-179 btn_p5_back 换页设置

2.3.6 导出 AppWizard 专案

透过 AppWizard 编辑器上方工具栏将 GUI 设计储存并导出项目(方法同前所述，如图 2-115)，生成文件夹如图 2-180 所示。

图 2-180 专案生成文件夹

2.4 将应用程序部署至开发板

2.4.1 透过 EmWin Package 生成烧录檔

EmWin package 含所有 emWin AppWizard 需要的 library 与资源，将设计完成的 GUI 项目文件夹放入 package 内并执行 Makefile 档案，即可生成程序 bin 文件，详细操作流程请参考 package 内的 README.txt，而将导出的 AppWizard 项目置入 emWin Package 的方法如下所述。

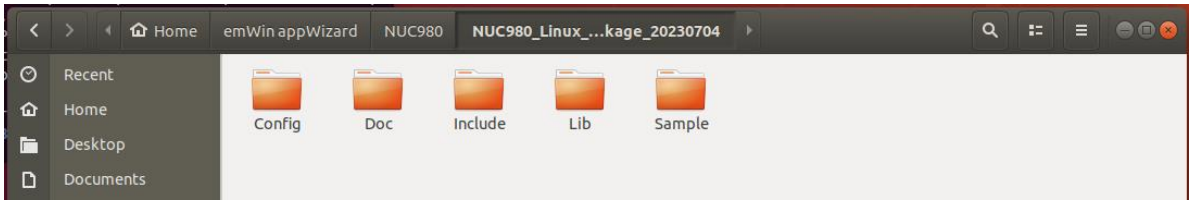


图 2-181 emWin Package 内容(解压缩后)

1. 将 appWizard 项目生成的文件夹”**Resource**”与”**Source**”复制到 Linux 环境中的 emWin package 路径下: */Sample/SimpleDemoAppwizard/Application*

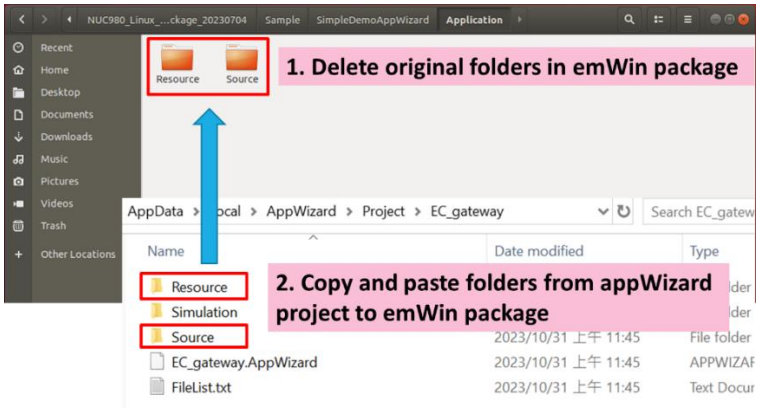


图 2-182 将 AppWizard 生成档案放入 Package

2. 回到上一层路径(/Sample/SimpleDemoAppwizard)，透过鼠标右键点选”Open in Terminal”开启 console 窗口输入指令”make”编译档案，如图 2-183~图 2-185 所示。

```
$ make
```

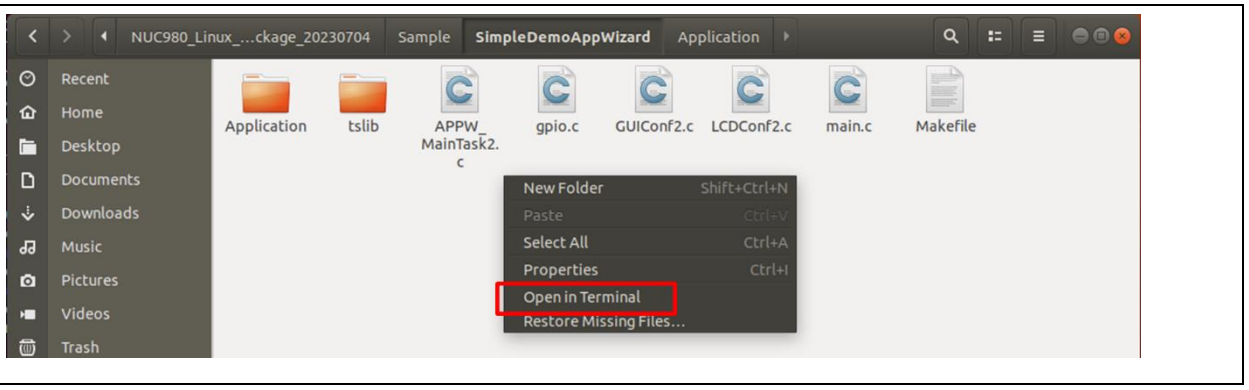


图 2-183 在文件夹内开启 console

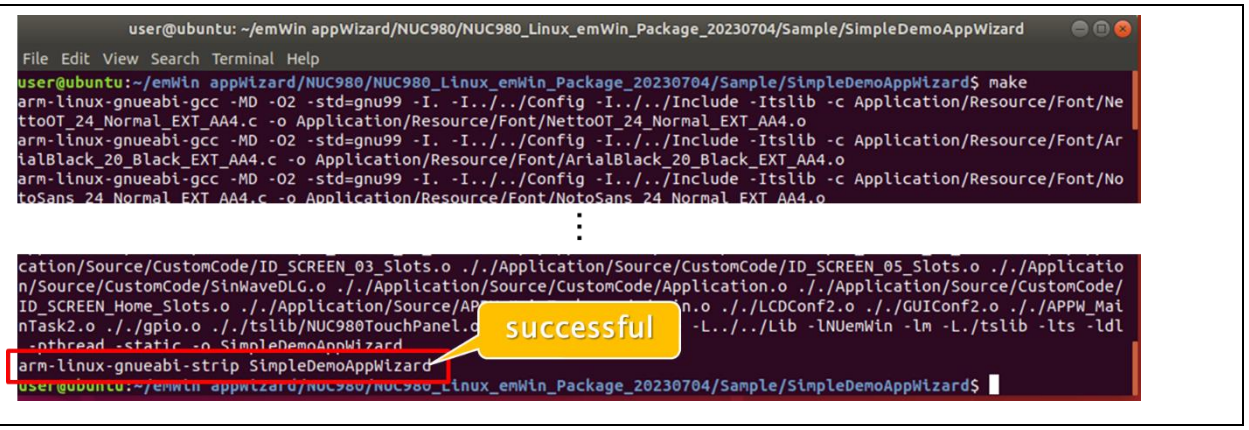


图 2-184 档案编译成功画面

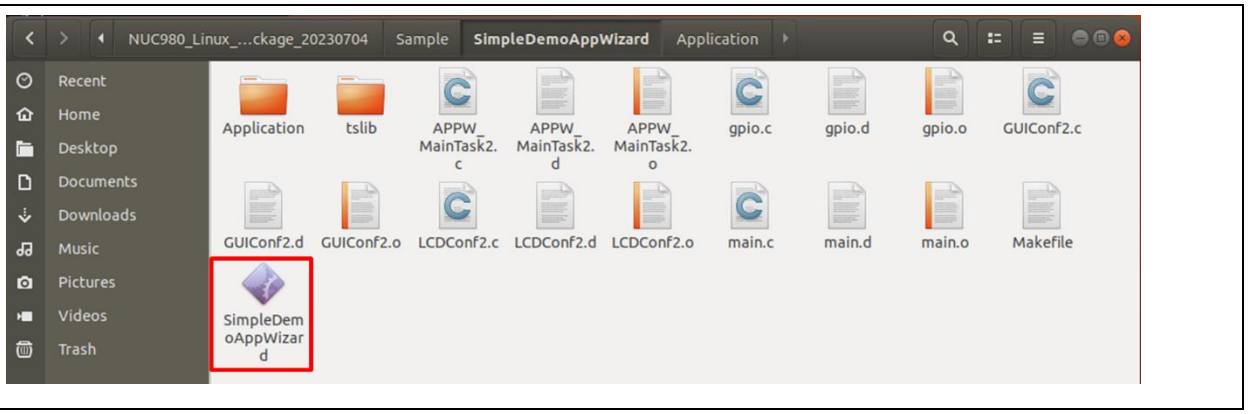


图 2-185 编译生成的 bin 檔

3. 将生成的 bin 档复制到 SD 卡内备用。(备注: SD 卡需为 FAT 格式)

2.4.2 在 NuMaker-IIoT-NUC980G2D 执行 GUI 程序

透过 Type-C USB 连接 NuMaker-IIoT-NUC980G2D 的 VCOM 接口上电，在 Console 中以”root” 登入后，插入 SD 卡，如图 2-186~图 2-188 所示。

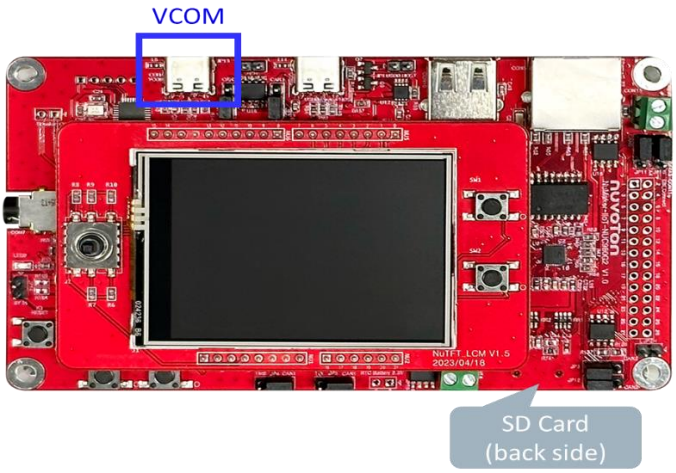


图 2-186 NuMaker-IIoT-NUC980G2D VCOM 与 SD 卡插入位置

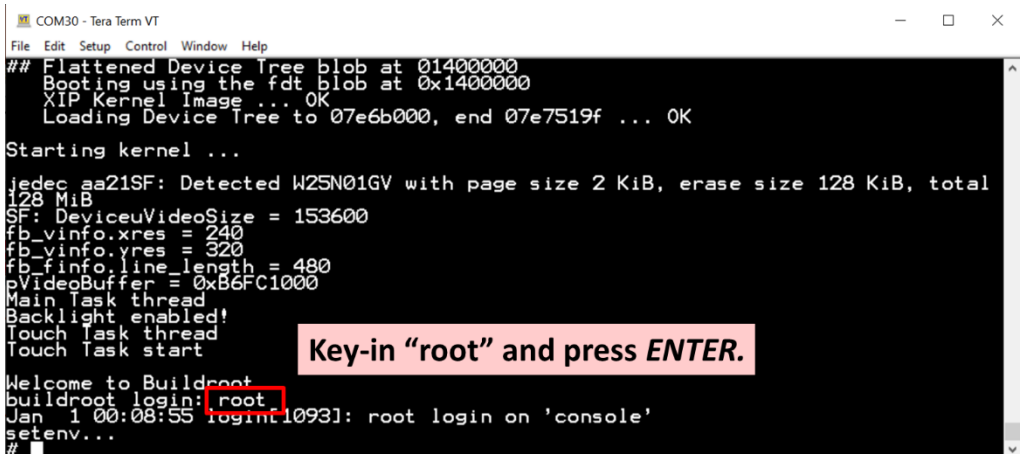


图 2-187 以 root 登入系统

图 2-188 SD 卡读取成功画面

SD 卡插入后，在 Console 内键入指令将 SD 卡内容挂载到 NuMaker-IIoT-NUC980G2D：

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

接着设置 GUI 所需的环境变量：

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

执行档案：

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

2.4.3 執行結果

本范例程序执行结果如图 2-189。

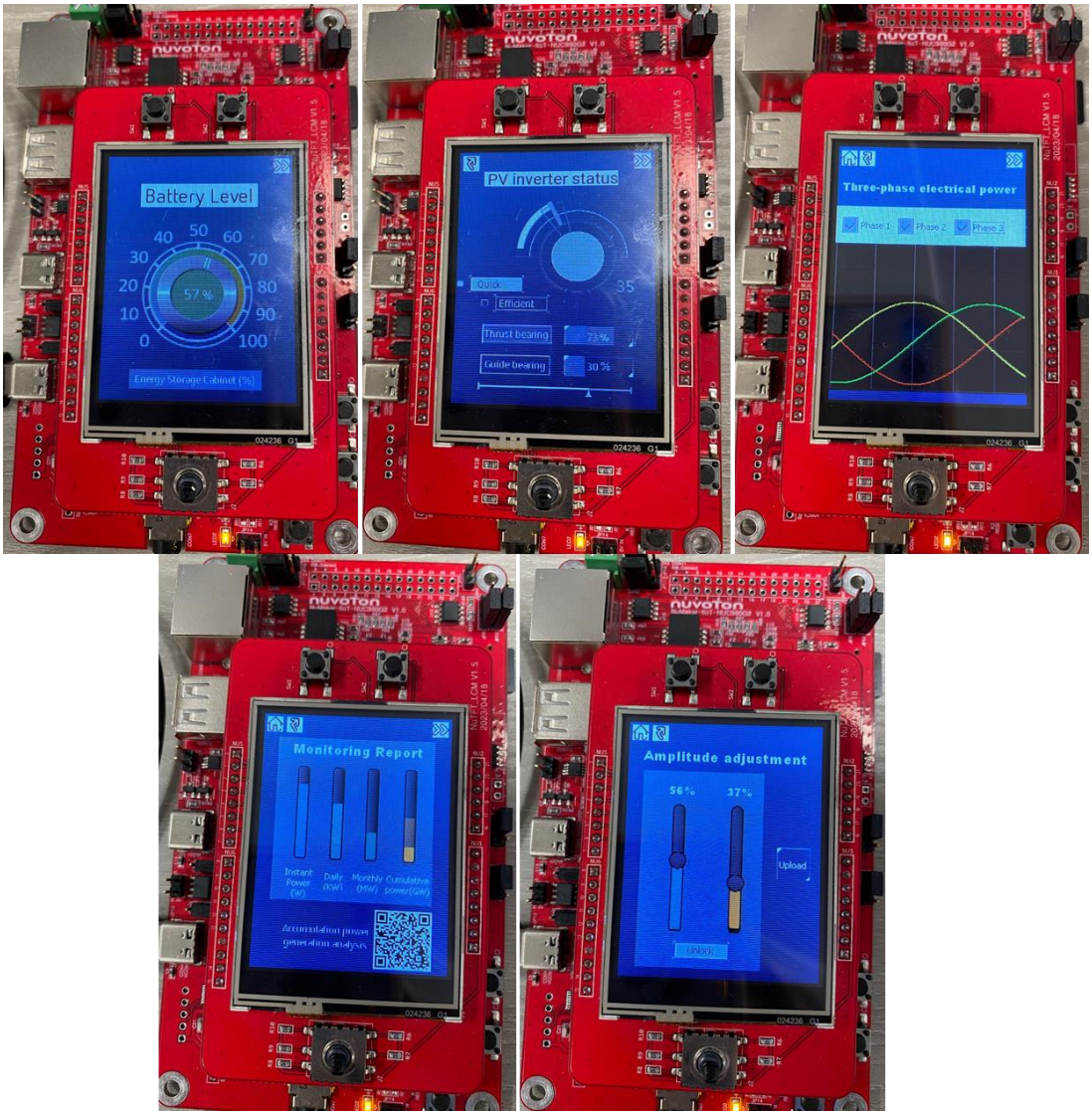


图 2-189 执行结果

3. 程序介绍

在 emWin Linux package 中，*main.c* 会对程序进行初始化，设定触控屏幕、frame buffer 与 Multiple Buffering 等，接着导向应用主程序 *MainTask2()*开始执行项目。下列程序位于 *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 的可用字节数及 Multiple Buffering 功能开关设置，可分别于 *LCDConf2.c* 及 *GUIConf2.c* 中调整相关参数。

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)
```

进入主程序 MainTask2()后，将执行 emWin appWizard 初始化、显示 GUI 对象等动作。下列程序位于 *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
  }
```

```
}
```

将 AppWizard 项目导出后，所有 GUI 设计都会被程序化，分别存在对应名称的 c 檔中，例如 Source\Generated 档案包含所有画面中的对象及属性设置，Source\CustomCode 内则包含各画面设置的所有 interactions 定义。若用户想在触发组件时，额外执行某些程序行为，即可在 *ID_SCREEN_Home_Slots.c*、*ID_SCREEN_02_Slots.c ~ ID_SCREEN_05_Slots.c* 的对应程序中新增。

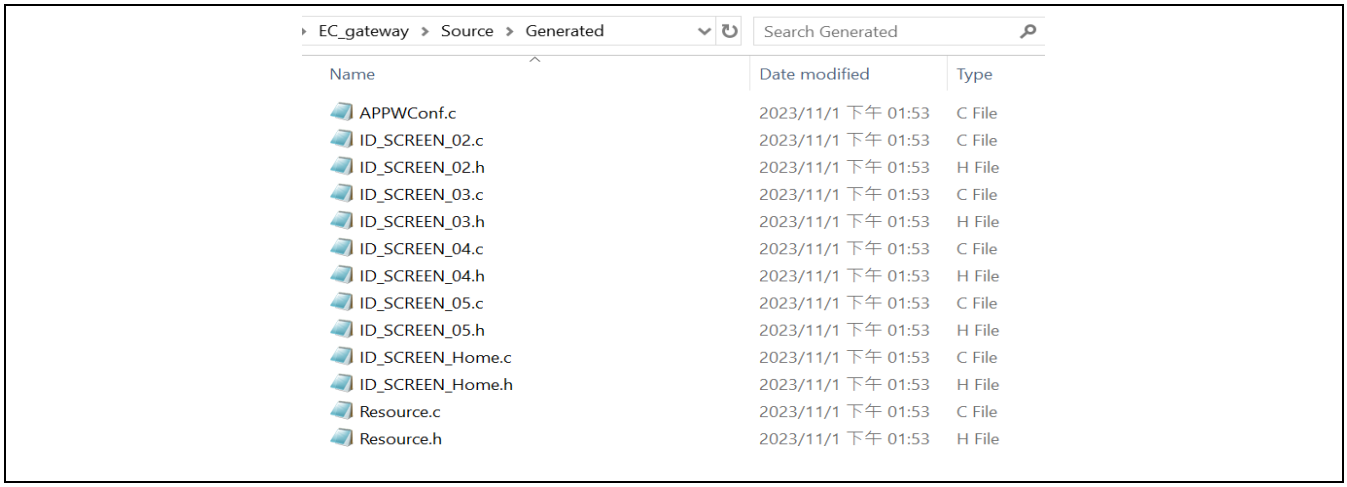


图 3-1 对象属性设置导出程序

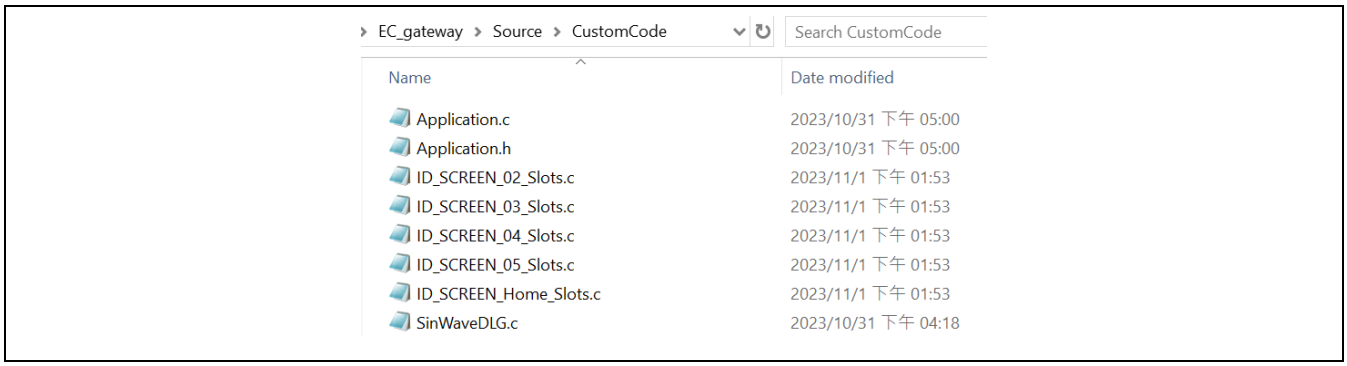


图 3-2 对象互动设置导出程序

以在主画面对象互动时，新增程序为例，客户可透过注记找到对应的 interaction，并在 functions 中新增自定义程序。此段程序代码位于 *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. 软件与硬件需求

4.1 软件需求

- IDE 版本
  - Ubuntu 18.04
- emWin AppWizard 版本
  - AppWizard_V134a_630a

4.2 硬件需求

- 电路组件
  - NuMaker-IIoT-NUC980G2D V1.0
- 线路示意图
  - 如图 4-1，透过 Type-C USB 连接 NuMaker-IIoT-NUC980G2D 的 VCOM 与 PC。
  - 如图 4-2，将 NuMaker-IIoT-NUC980G2D 开发板翻到背面，将 micro SD 插入卡槽。

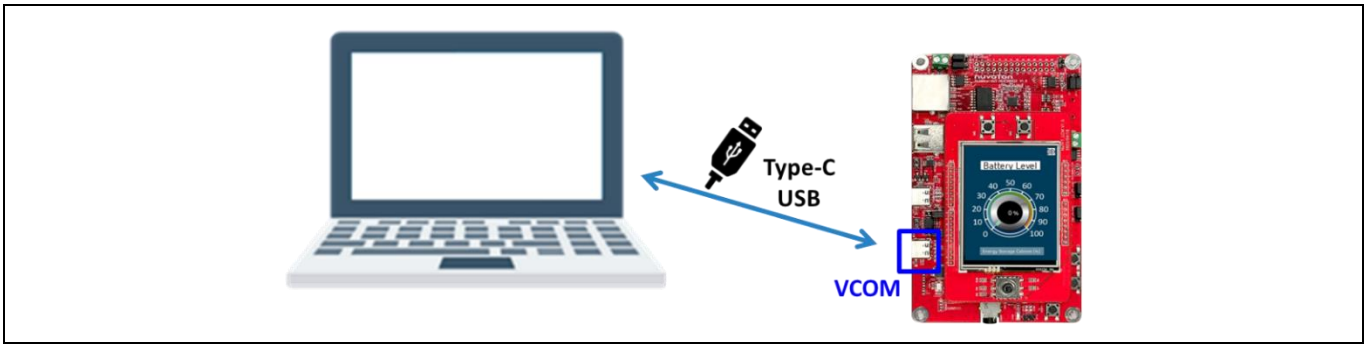


图 4-1 以 Type-C USB 线将 NuMaker-IIoT-NUC980G2D 与 PC 相连

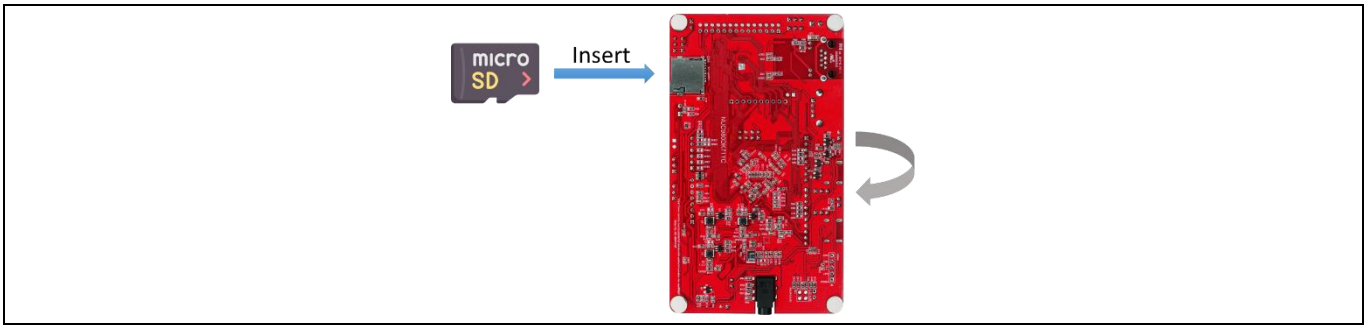


图 4-2 将 micro SD 插入卡槽

5. 目录信息

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.

图 5-1 目录信息

6. 范例程序执行

1. 于新唐官网下载 emWin package，将档案复制到 Linux 环境中解压缩，并依 *README.txt* 步骤操作(详见章节 2.4.1)

2. 将 AppWizard 专案(如同目录内文件夹 EC_gateway)生成文件夹”*Resource*”与”*Source*”复制到 Linux emWin package 路径: */Sample/SimpleDemoAppwizard/Application*(备注:请直接删除原始文件夹，避免透过复制贴上的方式取代)

   - 在 Linux 环境中执行 Makefile 档案，生成程序 bin 文件
   - 将 bin 檔置于 SD 卡，并插入开发板 SD 卡插槽

3. 在 Windows 环境下，透过 console 以 root 权限登入系统

   - 挂载 SD 卡资料
   - 执行 bin 檔

※备注：若发现执行程序发现有部分 Widget 功能失效，可能是 emWin Package 与 AppWizard Project 版本不匹配的缘故。若发生此情形，请依下列步骤处理：

- 安装 emWin package 内所提供的最新版 AppWizard IDE
- 安装完成后，以最新版 AppWizard IDE 开启 AppWizard 项目 (*../EC_NUC980_AppWizard_New_Energy_Gateway_V1.00/EC_gateway*)
- 开启项目后，会跳出需要升级项目的提醒窗口，点选 Yes 即可升级
- 项目升级后，透过工具栏 File>Export & Save 重新导出 AppWizard 档案，接着即可回步骤 2 继续范例程序执行流程

7. 修订纪录

Date	Revision	Description
2024.04.02	1.00	初始发布。

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.