Charge Pile GUI Designed by AppWizard

Example Code Introduction for 32-bit NuMicro® Family

Document Information

Application	This example code utilizes the SEGGER emWin AppWizard tool in conjunction with the M467 series development platform to achieve a charge pile graphical user interface (GUI) design.
BSP Version	M460_Series_BSP_CMSIS_V3.00.002 emWin Software Package for M460 V3.0
Hardware	NuMaker-M467HJ 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. Overview

This example code demonstrates how to use the SEGGER emWin AppWizard tool in conjunction with the NuMaker-M467HJ V1.0 development platform to design the user interface and implement interactivity for a charge pile project.

1.1 Principle

1.1.1 Block Diagram

This example code directly utilizes the NuMaker-M467HJ V1.0 evaluation board from Nuvoton's GUI development platform. The required components for development include a 4.3-inch TFT-LCD capacitive touch panel with resolution in 480x272, the NuMicro M467HJHAN microcontroller on the main board, and a 4Mbyte SPI Flash for external storage. The system block diagram is depicted in Figure 1-2.

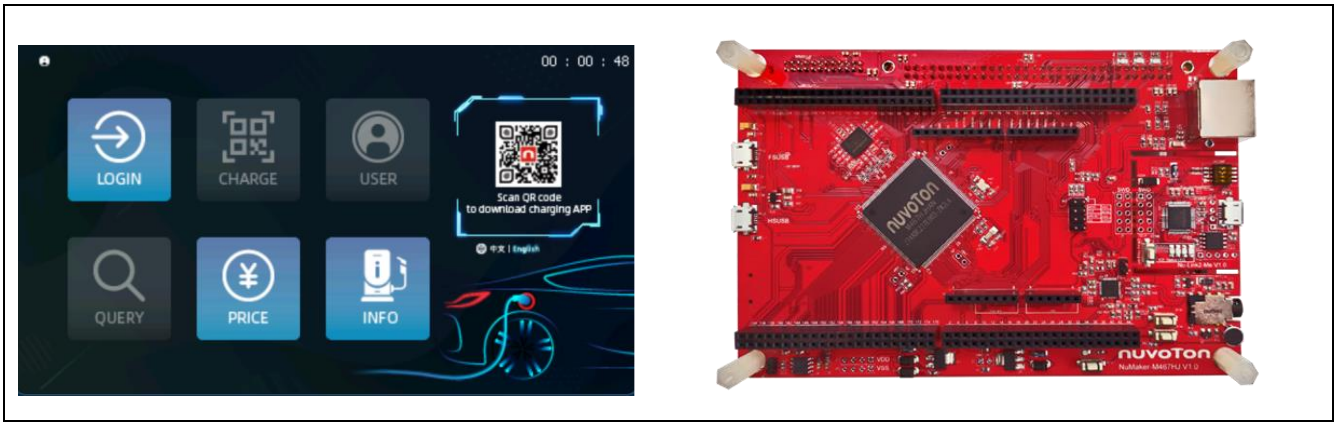


Figure 1-1 NuMaker-M467HJ V1.0

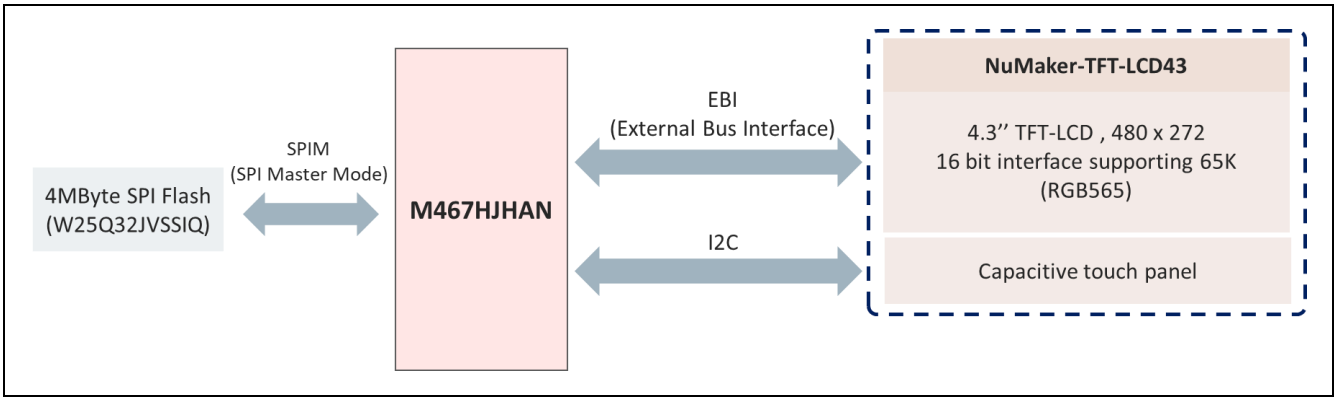
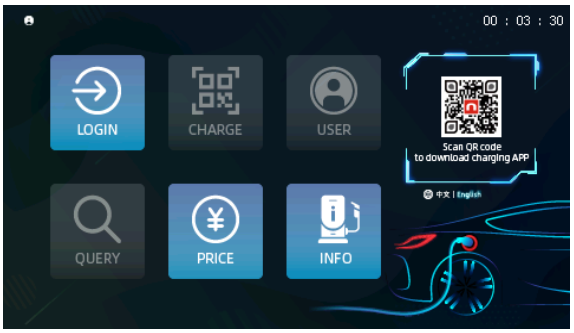
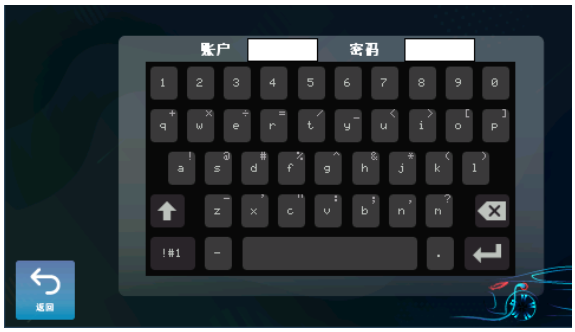
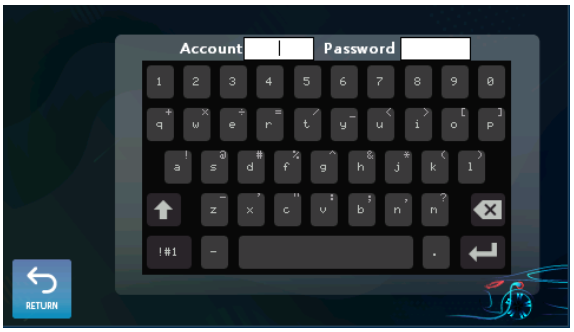
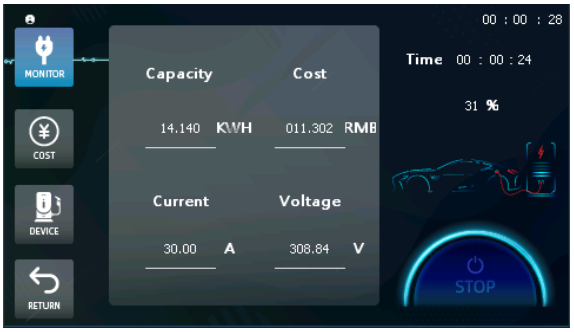
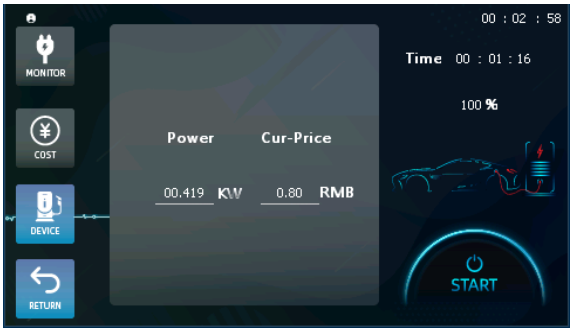
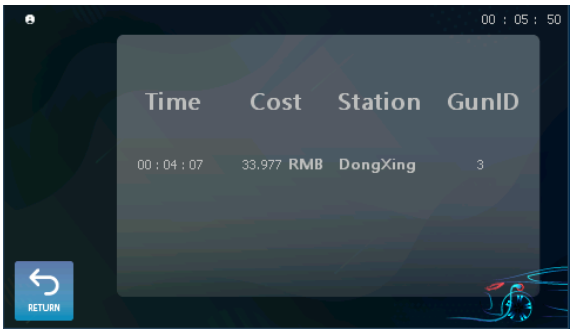
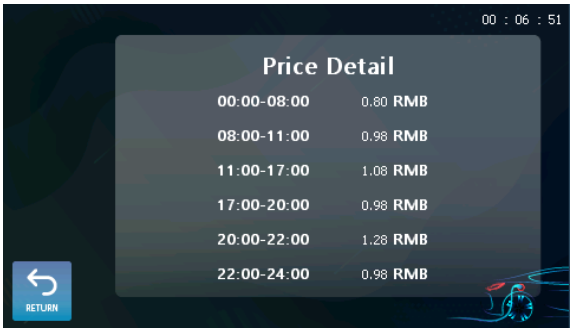


Figure 1-2 Block Diagram

1.1.2 Charge Pile Screen

The design for the screen in this example is as follows.

	Chinese interface	English interface
Screen 1		
Screen 2		
Screen 3		
Screen 3		

Screen 3		
Screen 4		
Screen 5		
Screen 6		

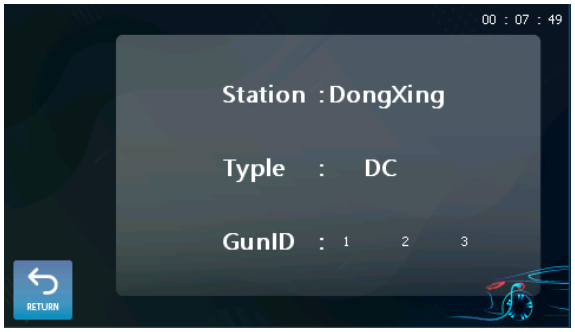
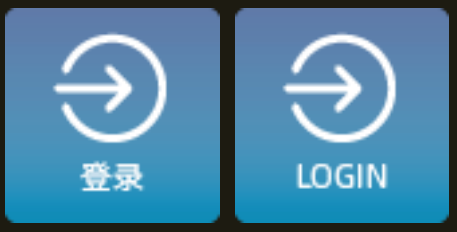
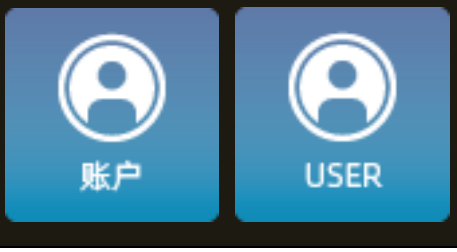
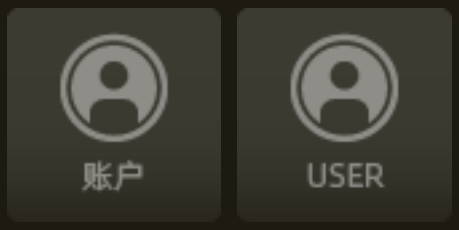
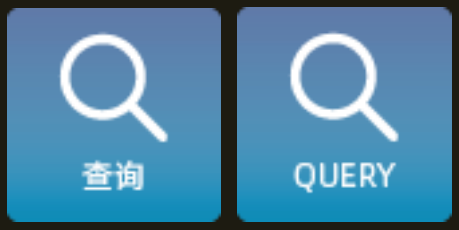
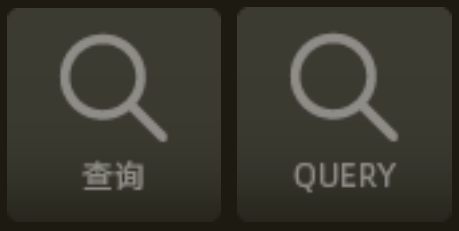
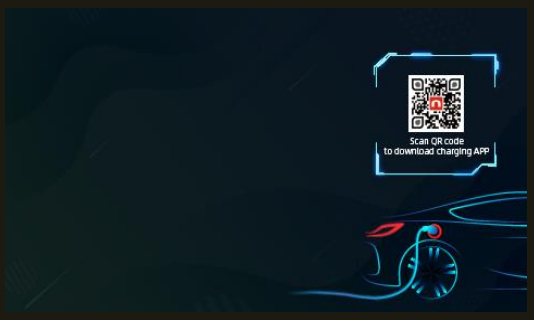
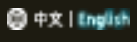
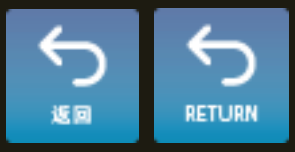
Screen 7		

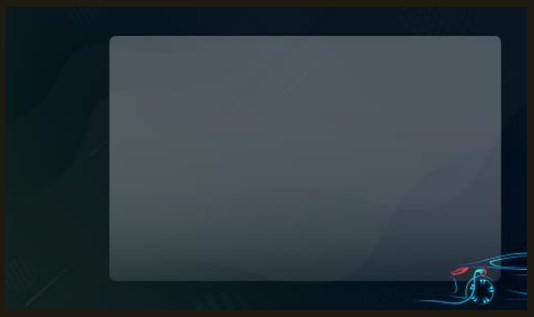
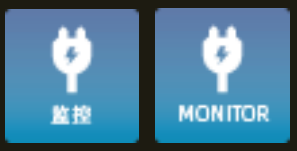
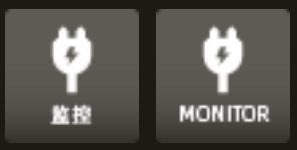
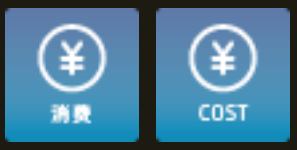
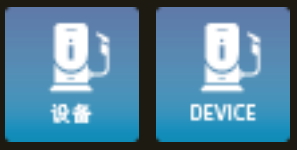
Table 1-1 Charge Pile Screen

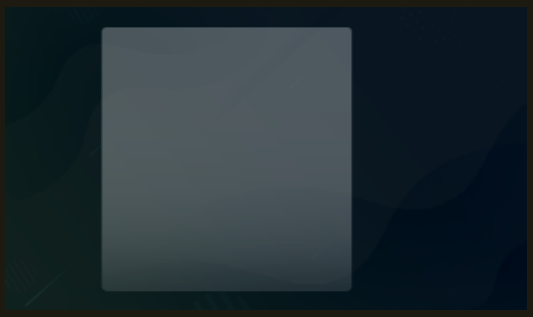
1.1.3 Overview of Elements Used in the Charge Pile Screen

ChargePile App Des	Icon Name	ChargePile App GUI	Object
Login button valid	Login_Pressed_Ini、Login_Pressed_Ini_En		Image
Charge button valid	Charge_Pressed_Ini、Charge_Pressed_Ini_En		
Charge button invalid	Charge_Lock_Ini、Charge_Lock_Ini_En		
Account button valid	Account_Pressed_Ini、Account_Pressed_Ini_En		

Account button invalid	Account_Lock_Ini、Account_Lock_Ini_En	
Query button valid	Query_Pressed_Ini、Query_Pressed_Ini_En	
Query button invalid	Query_Lock_Ini、Query_Lock_Ini_En	
Price button valid	Price_Pressed_Ini、Price_Pressed_Ini_En	
Price button invalid	Price_Ini、Price_Ini_En	
Infomation button valid	Info_Pressed_Ini、Info_Pressed_Ini_En	

Infomation button invalid	Info_Ini、Info_Ini_En	
Initial screen background	Bg2、Bg2_En	 
Chinese	Lan_Cn	
English	Lan_En	
user	Head_Ini	
Return button valid	Return1、Return1_En	
Return button invalid	Return0、Return0_En	

Other screen background	Bg3	
Monitor button valid	Mon、Mon_En	
Monitor button invalid	Mon_0、Mon_0_En	
Cost button valid	Cost、Cost_En	
Cost button invalid	Cost_0、Cost_0_En	
Device button valid	Device、Device_En	
Device button invalid	Device_0、Device_0_En	
Start charging	Start	

Stop chargeing	Stop_Pressed	
0% electric capacity	che0	
0%~19% electric capacity	che1	
20%~39% electric capacity	che2	
40%~59% electric capacity	che3	
60%~79% electric capacity	che4	
80%~100% electric capacity	che5	
Charging screen background	Bg_Cha	
Connector	Connect	

Short data underline	ExpenseLine	———	
Long data underline	MonLine	———	

Table 1-2 Elements Used

1.1.4 Supporting Resources

- SEGGER emWin Forum：Segger emWin Forum
- Nuvoton Forum：Nuvoton emWin Forum
- User manual of SEGGER emWin AppWizard：AppWizard User Manual

1.2 Execution Steps

The process of completing a GUI project using the AppWizard can be divided into five stages. In the first stage, hardware and software preparation are required. The second stage involves designing the screens, setting up objects within the AppWizard interface. The third stage involves defining interactions, and configuring animations within the AppWizard interface. In the fourth stage, the generated code from AppWizard is exported, and custom code can be added to enhance additional functionalities. The fifth stage is programming the board.

Each of the stages are described in section 1.2.1 to 1.2.5.

1.2.1 Preparation of Hardware and Software

1. The hardware used is NuMaker-M467HJ V1.0. Please refer to Schematic and User Manual for more related information.
2. The software resource requires downloading the AppWizard software tool.
    1) Please click the following link, "emWin Software Package for M460" to access the download page. Click the "emWin Software Package for M460" item to initiate the download. Prior to downloading, registration and login to the Nuvoton official website are required.
    2) After the download is complete, extract the files and open the "M460_emWin_NonOS" folder.
    3) According to the path in Table 1-3, click "AppWizard_V134a_630a_Install.exe" to install the AppWizard tool. Please note that if you have an older version of AppWizard, you need to completely uninstall the old version before installing the new version.

M460_emWin_NonOS
emWin
Tool
AppWizard_V134a_630a_Install.exe

Table 1-3 AppWizard – Installation File Path

1.2.2 GUI Designed in AppWizard

1. Create a new project and add the screens required for the project.
    1) Open the AppWizard tool using the desktop shortcut.

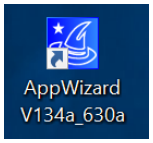


Figure 1-3 Open AppWizard Tool

2) Click "Create new project" to create new project.

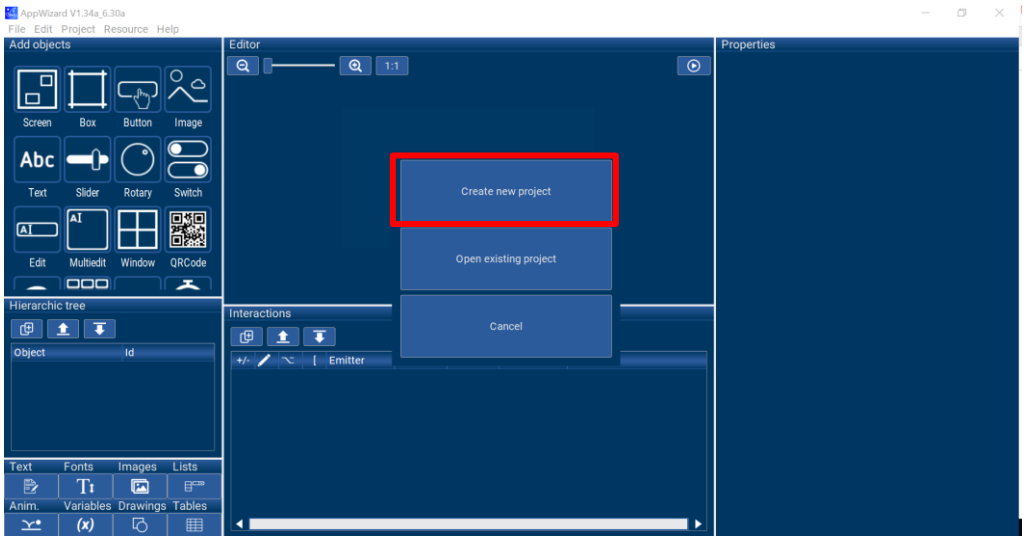


Figure 1-4 Create New Project

3) Edit project path and project name. Ensure that the color format should match the screen's specification. According to Figure 1-5, choose 16 Bit, GUICC_M565, and check "Enable Multibuffering".

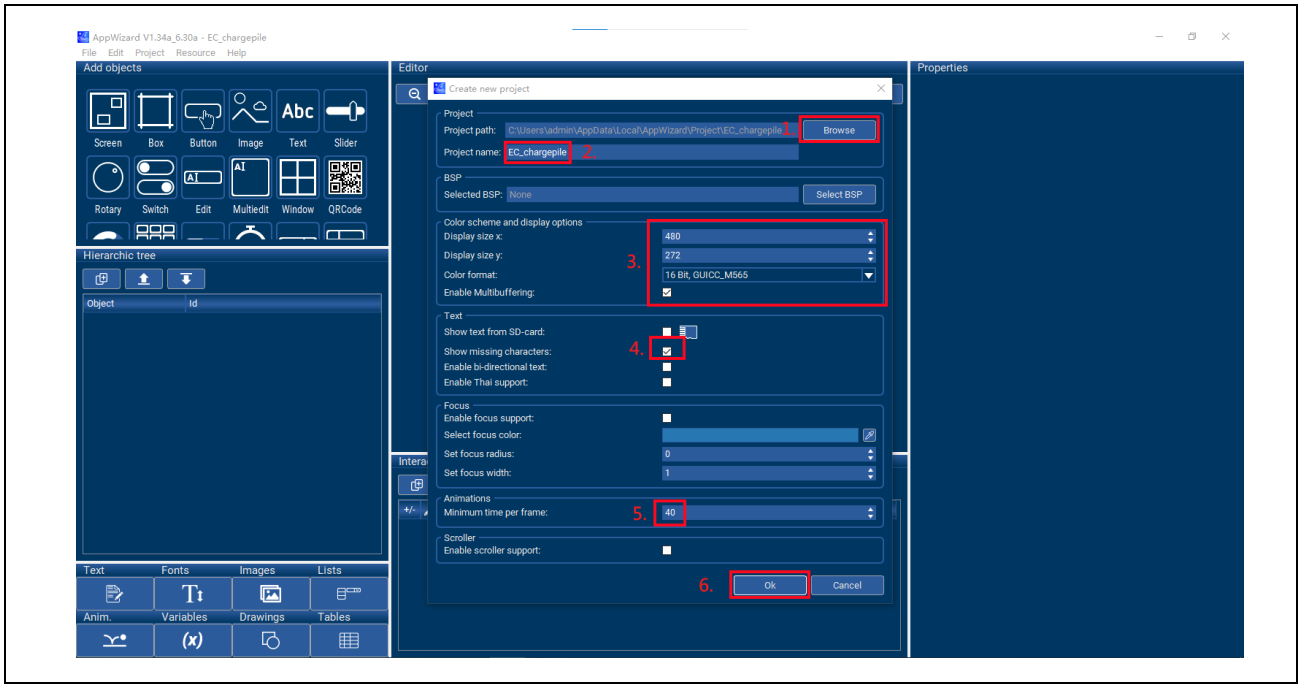


Figure 1-5 New Project Setting

4) Click "Screen" seven times under "Add objects" to create seven screens. In "Hierarchic tree", name the screens as "ID_SCREEN_Init", "ID_SCREEN_ Login", "ID_SCREEN_ Charge", "ID_SCREEN_ Account", "ID_SCREEN_ Query", "ID_SCREEN_ Price" and "ID_SCREEN_ Info" respectively for easier identification.

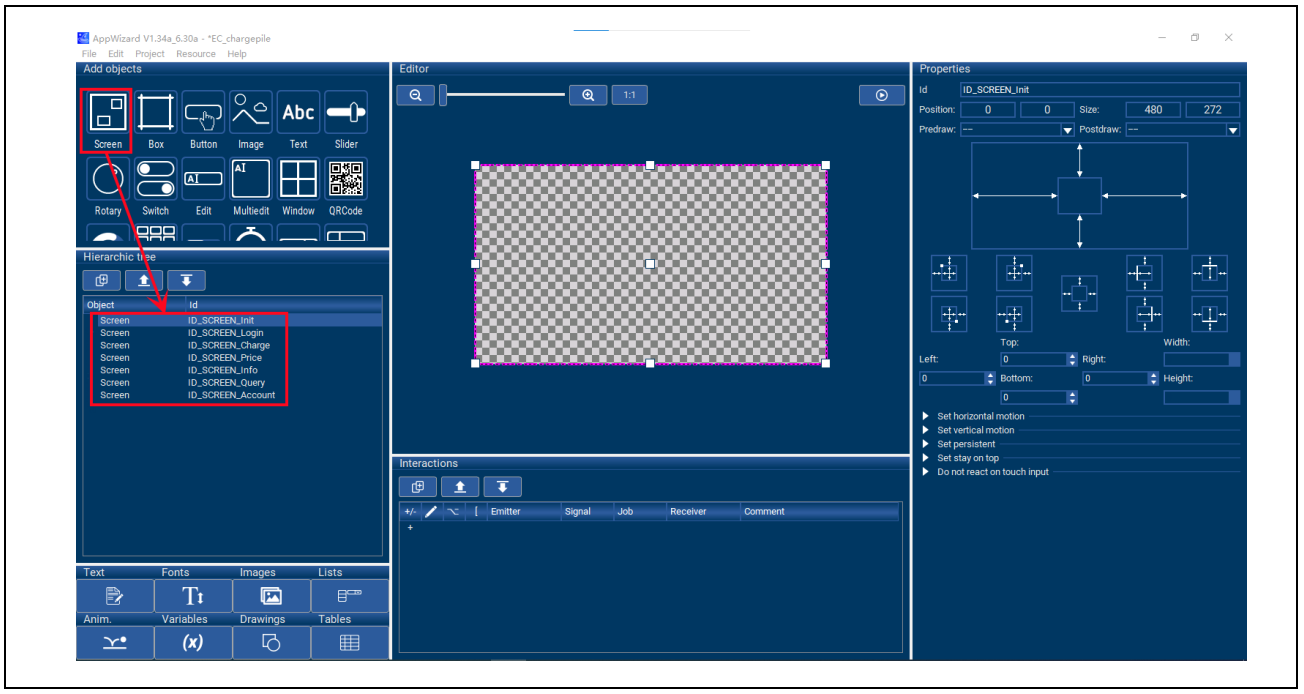


Figure 1-6 Add Screens

2. Init screen design. Including background icon and user icon and username and real-time time display and six buttons for screen switching and one button for language switching.

    1) Click "ID_SCREEN_Init" to edit the "Init" screen. Under "Add objects", select "Image" to create a background. Modify the name to "ID_IMAGE_Bg_Ini", then set the location and the size in Properties.

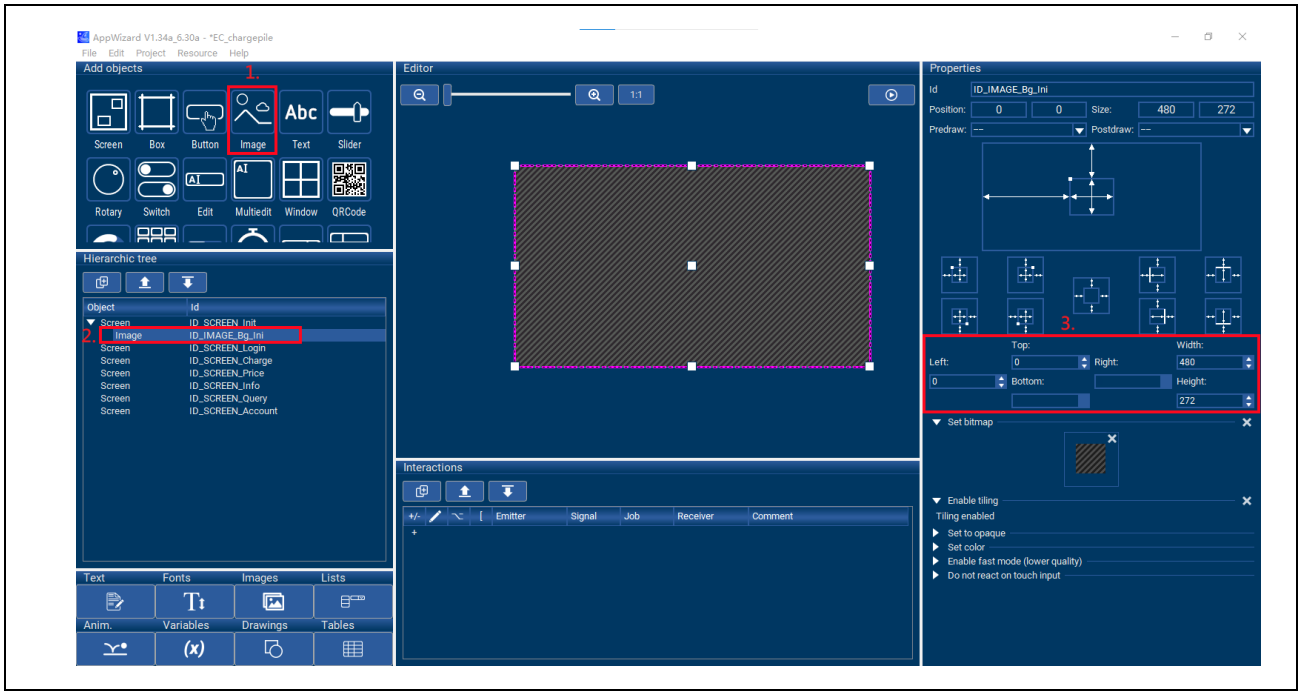
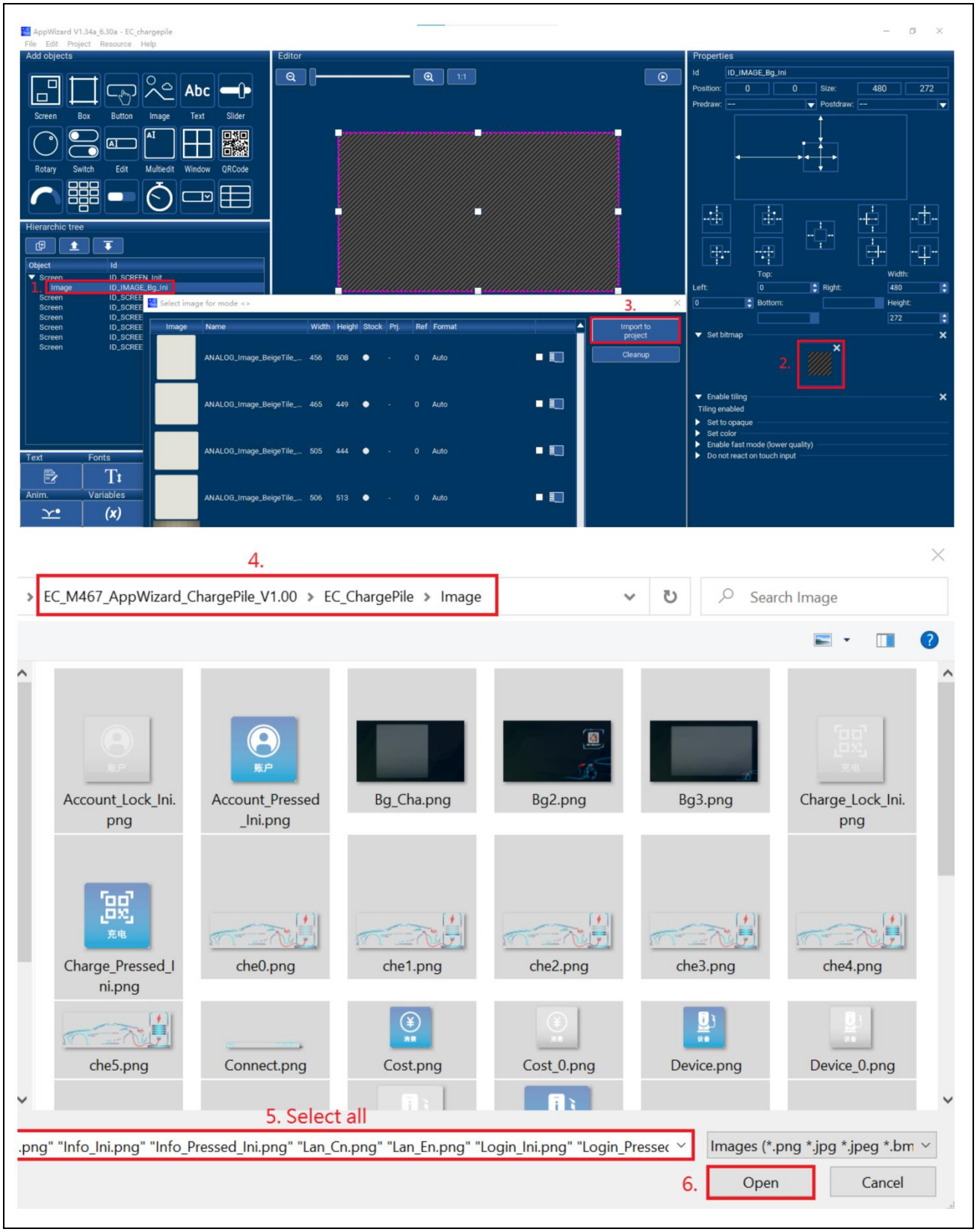


Figure 1-7 Create Background with Image and Set the Position and the Size

2) In Properties, set the bitmaps property for the "Bg".



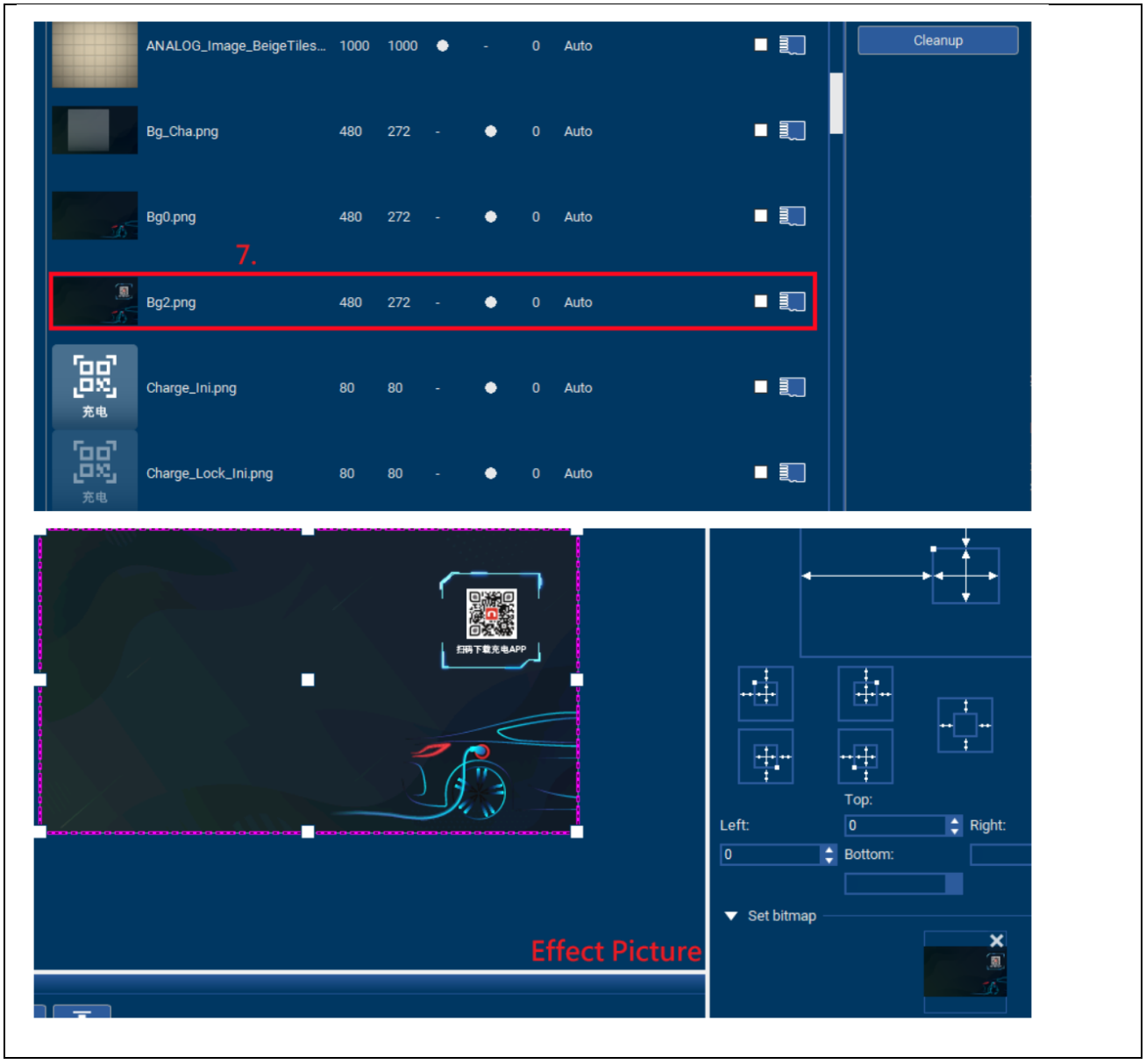


Figure 1-8 Import the Icons Required and Set the Bitmap Property for the "Bg"

3) Click "ID_SCREEN_Init". To create screen switchting buttons, click "Button" seven times under "Add objects". In "Hierarchic tree", name the buttons as "ID_BUTTON_Login_Ini", "ID_BUTTON_Charge_Ini", "ID_BUTTON_Acc_Ini", "ID_BUTTON_Query_Ini", "ID_BUTTON_Price_Ini", "ID_BUTTON_Info_Ini" and "ID_BUTTON_Lan_Ini" respectively for easier identification.Then set the position and the size of the objects.

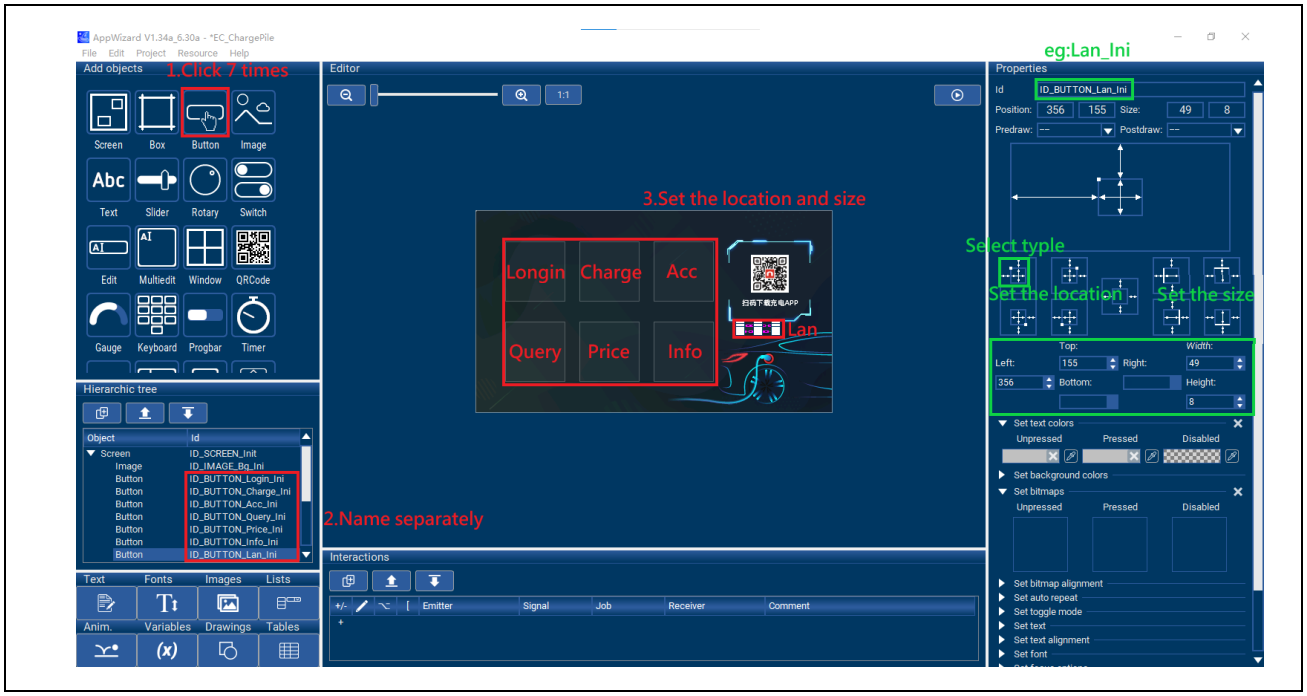


Figure 1-9 Create Screen Switching Buttons

4) In Properties, set the bitmaps property for each button (Refer to Step 2 (2) in Section 1.2.2, ignoring substeps 3 through 6).

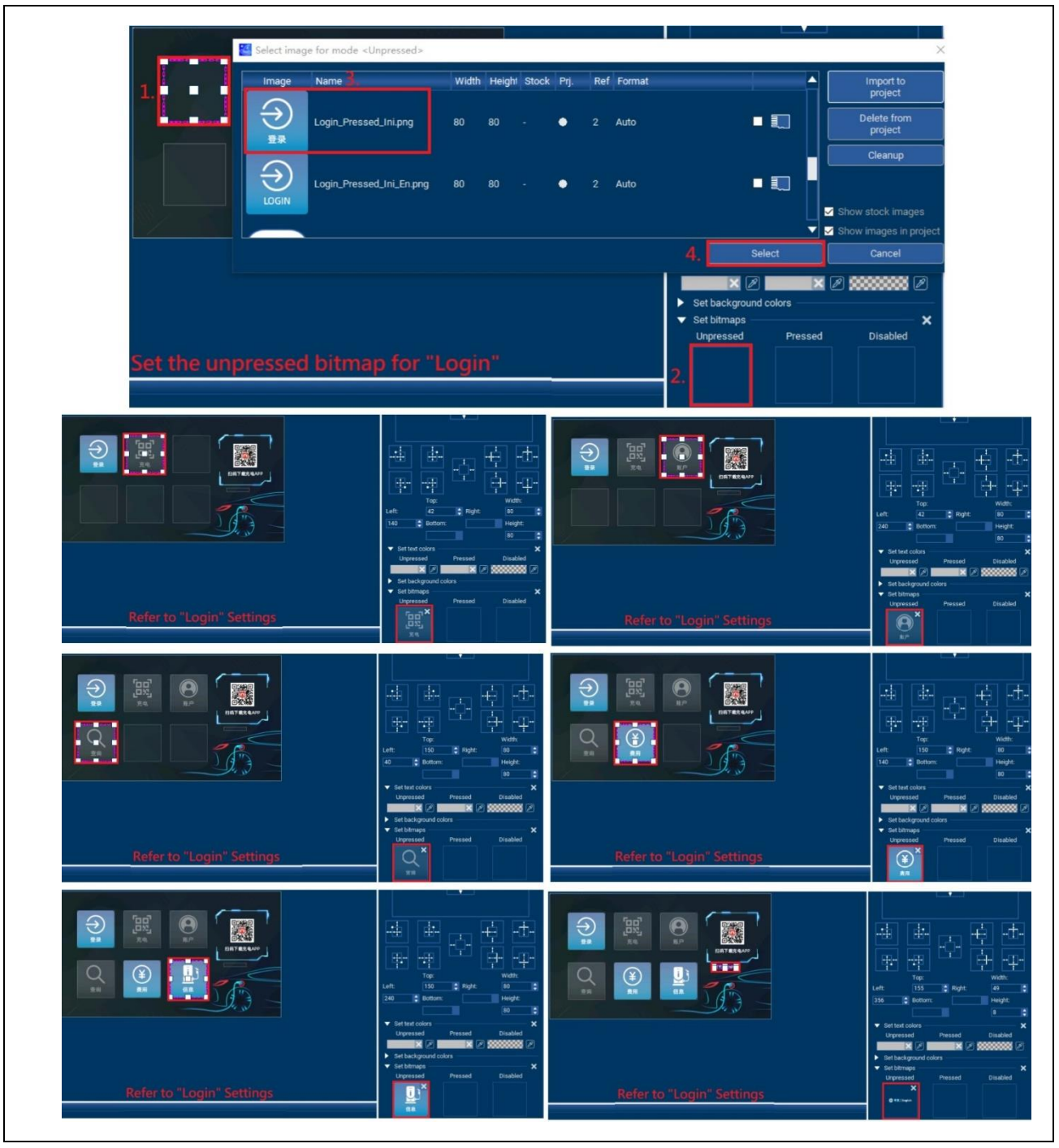
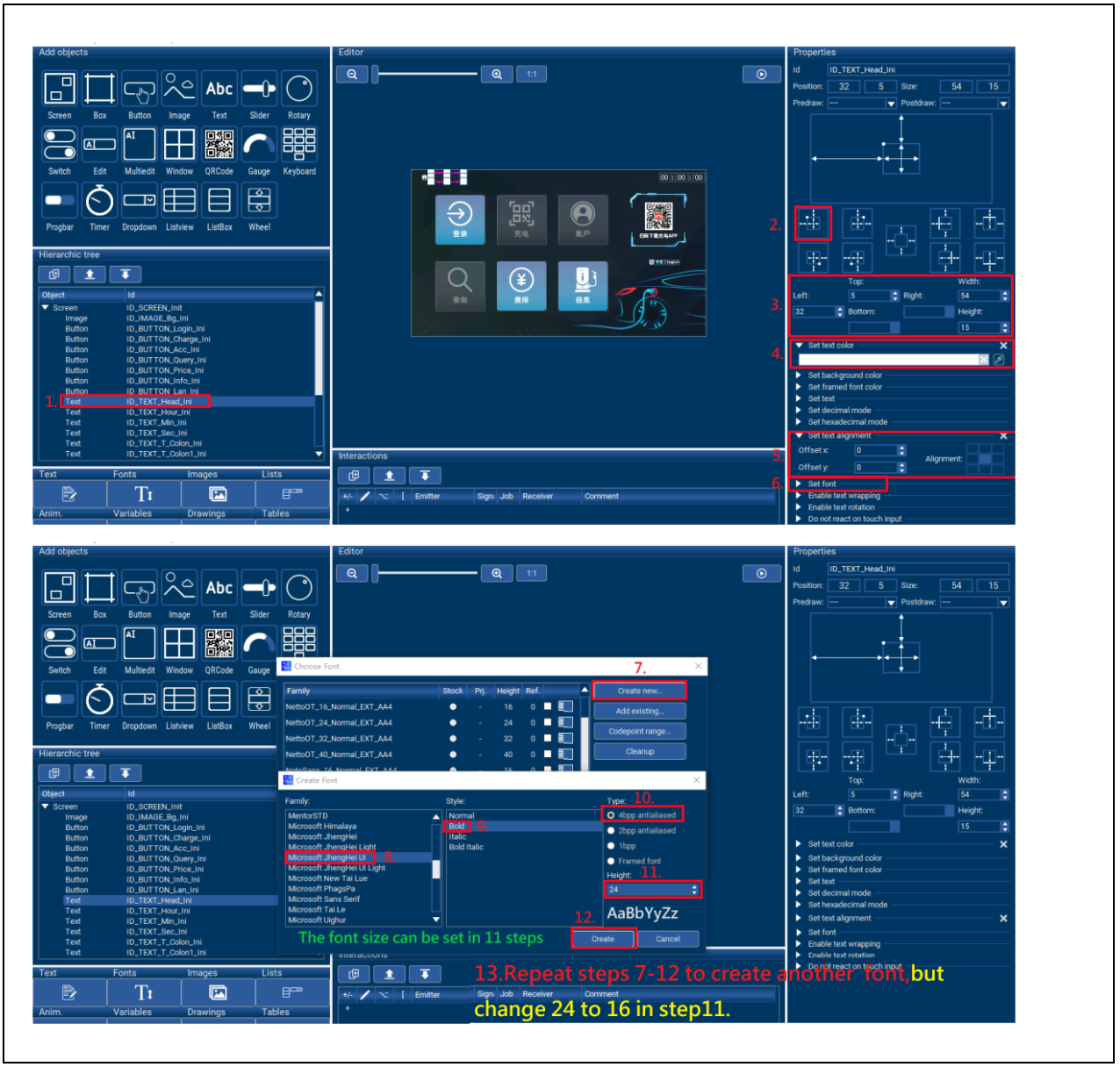


Figure 1-10 Set Each Button Bitmaps Property

5) Click "ID_SCREEN_Init". For display the text, click "Text" six times under "Add objects".Then re-named them as "ID_TEXT_Head_Ini", "ID_TEXT_Hour_Ini", "ID_TEXT_Min_Ini", "ID_TEXT_Sec_Ini", "ID_TEXT_T_Colon_Ini" and "ID_TEXT_T_Colon1_Ini" for easy identification in "Hierarchic tree".

6) Sets the position, the size, the color, the alignment mode and the font of the "ID_TEXT_Head_Ini" for displaying text. Notice substeps 7 through 13 is to create two types of fonts.



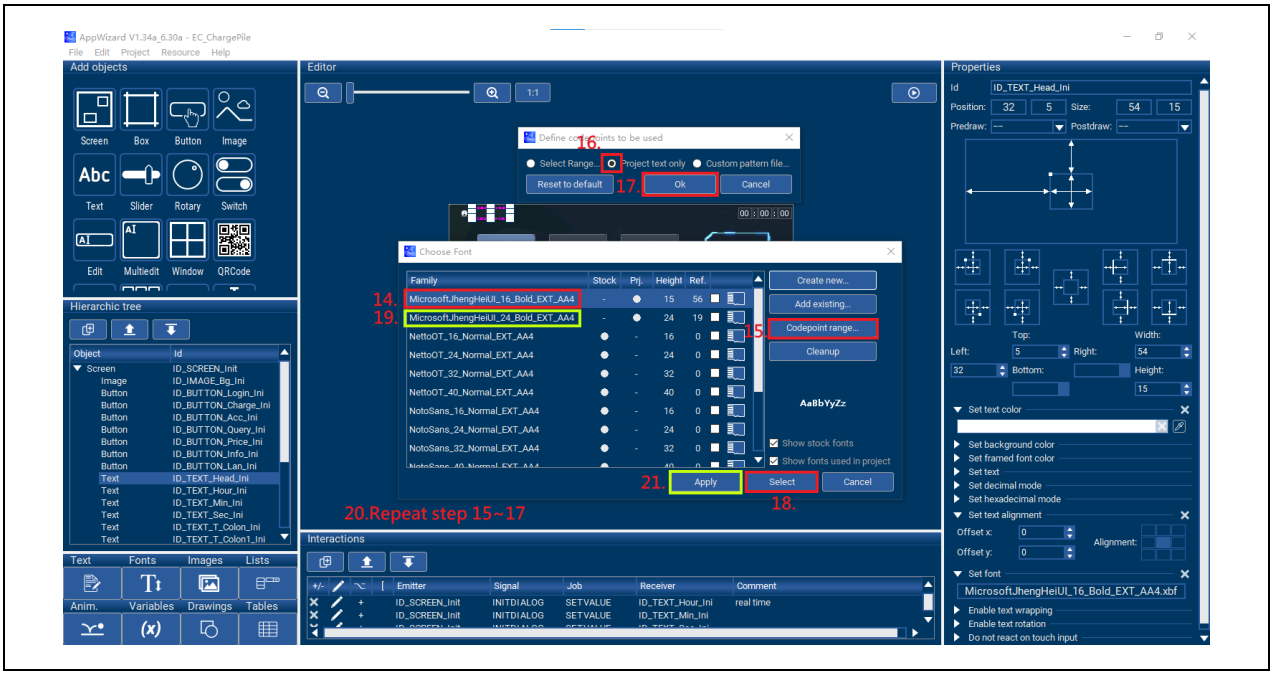


Figure 1-11 Set the Relevant Properties for Display Text

7) Set the properties of the "ID_TEXT_Hour_Ini" for display 2-digit values (Settings of "ID_TEXT_Min_Ini" and "ID_TEXT_Sec_Ini" are similar to setting of "ID_TEXT_Hour_Ini").

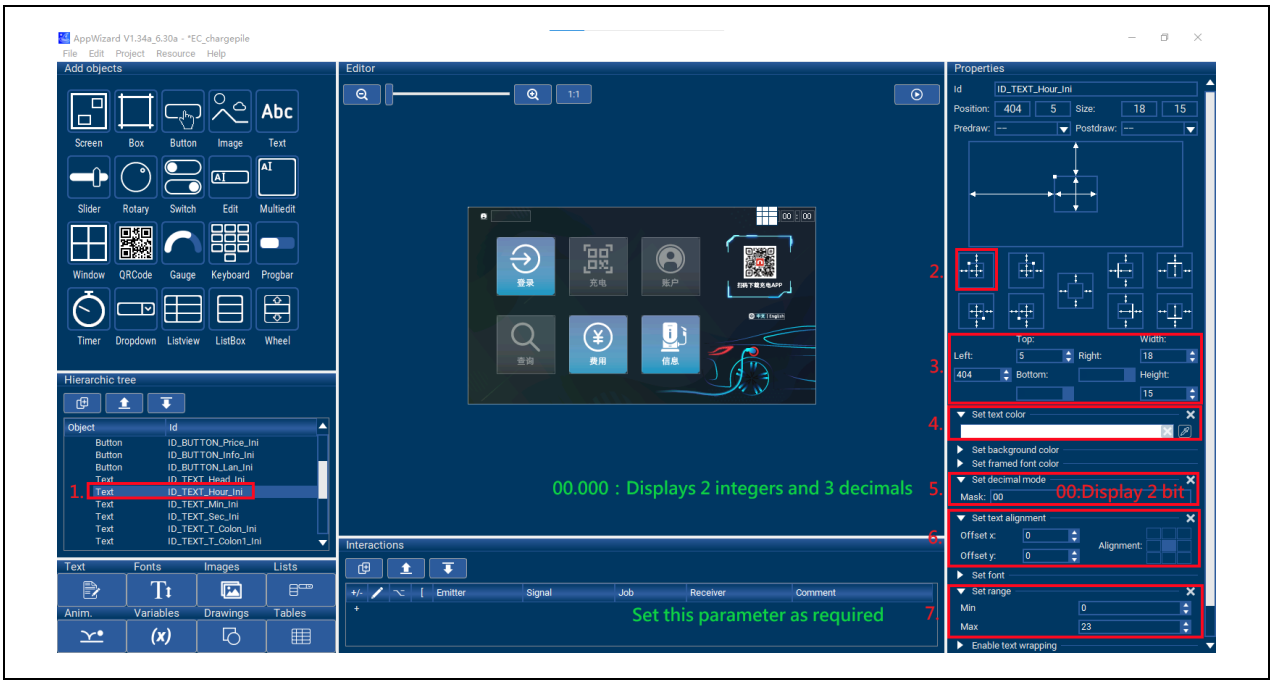


Figure 1-12 Set the Relevant Properties for Display Value

8) Set "ID_TEXT_T_Colon_Ini" for displaying the separators between hour and minute in both English and Chinese (The setting of "ID_TEXT_T_Colon1_Ini" is similar to "ID_TEXT_T_Colon_Ini"). Note the purpose of substeps 7 through 12 is to create Chinese and English.

Figure 1-13 Set the Properties of Text for Displaying Chinese and English

9) Click "ID_SCREEN_Init" to edit the first screen. Click "Image" in "Add objects" to create the user icon and change the name to "ID_IMAGE_User_Ini", then set the location and the size and bitmaps of this object.

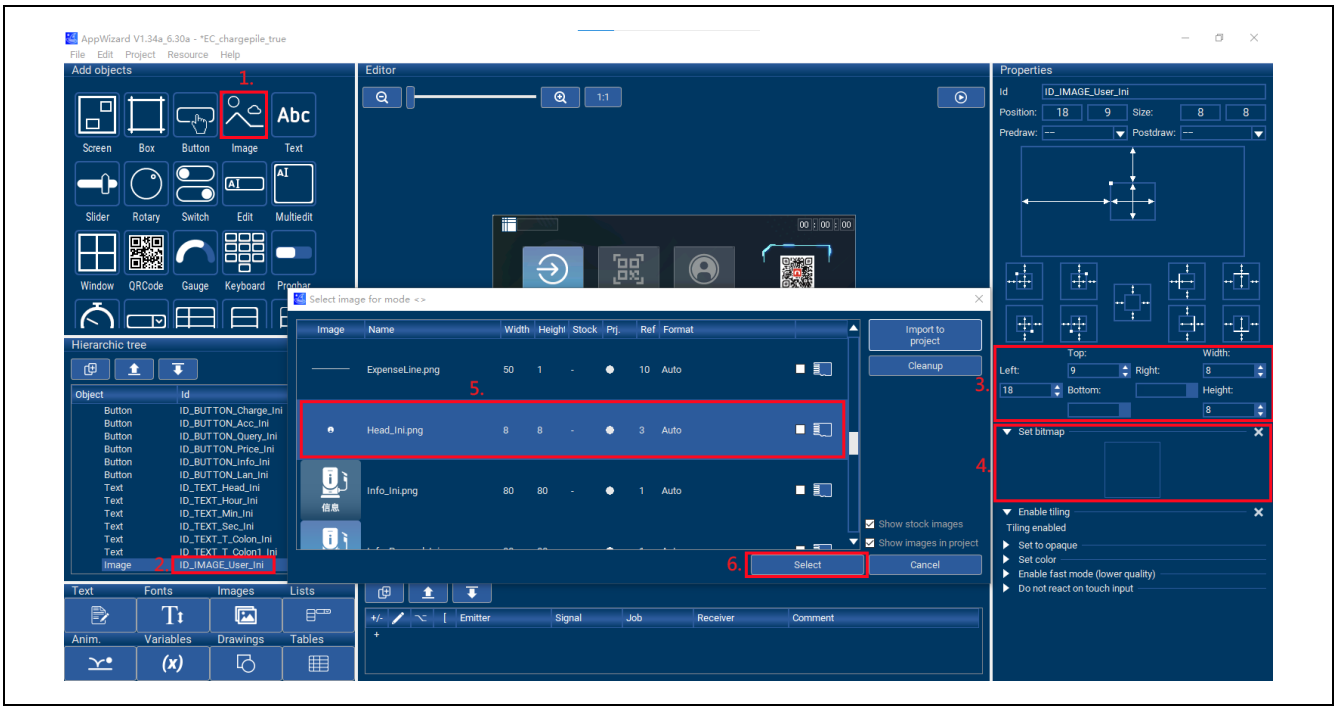


Figure 1-14 Set the Location, Size and Bitmaps of the "User"

10) Click "ID_SCREEN_Init" to edit the first screen. Click "Timer" in "Add objects" to create a timer and change the name to "ID_TIMER_1S_Ini". Set the timing period and enable the automatic restart mode.

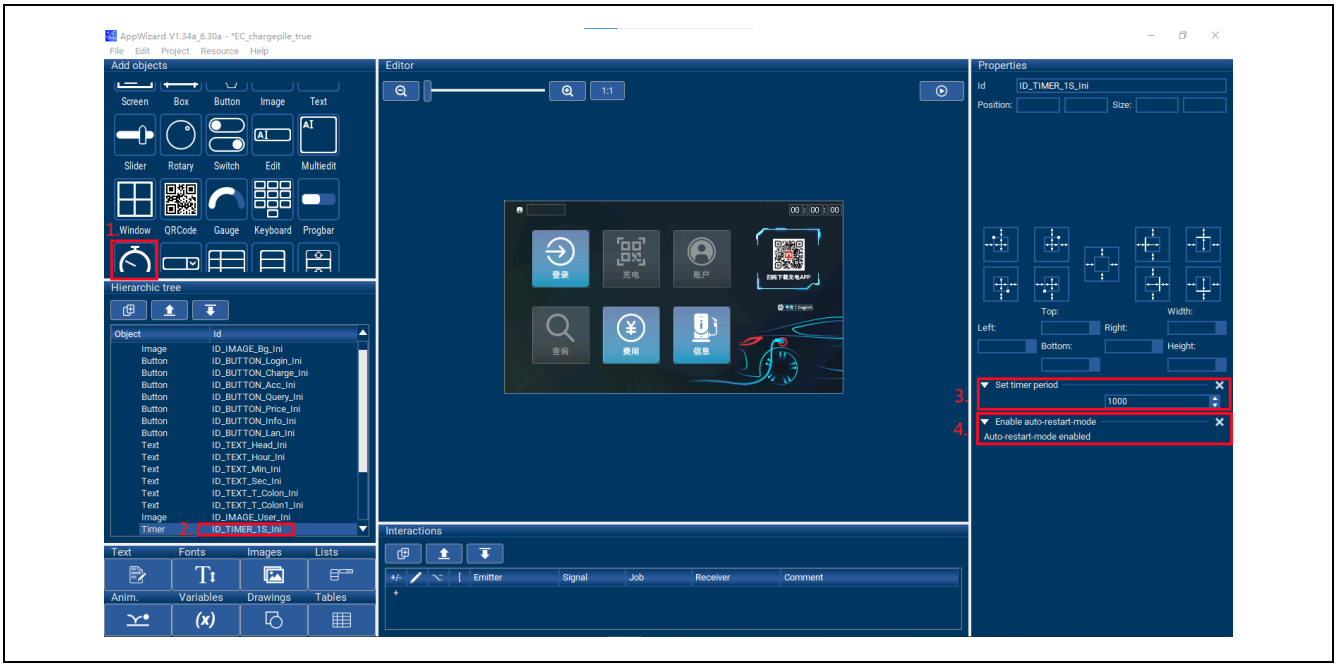


Figure 1-15 Set the Properties of the Timer Object

3. Charge screen design. Including the main part, the monitoring-window, the consumption-window and the device-window.

    1) Charge screen main part. Refer to the "Init" screen design to add objects and set their properties. Including three switch buttons, one return button and one start/stop charging button, displaying the real-time time, charging time and username, loading pattern of the background and the user. The percentage of charge is displayed numerically and graphically.

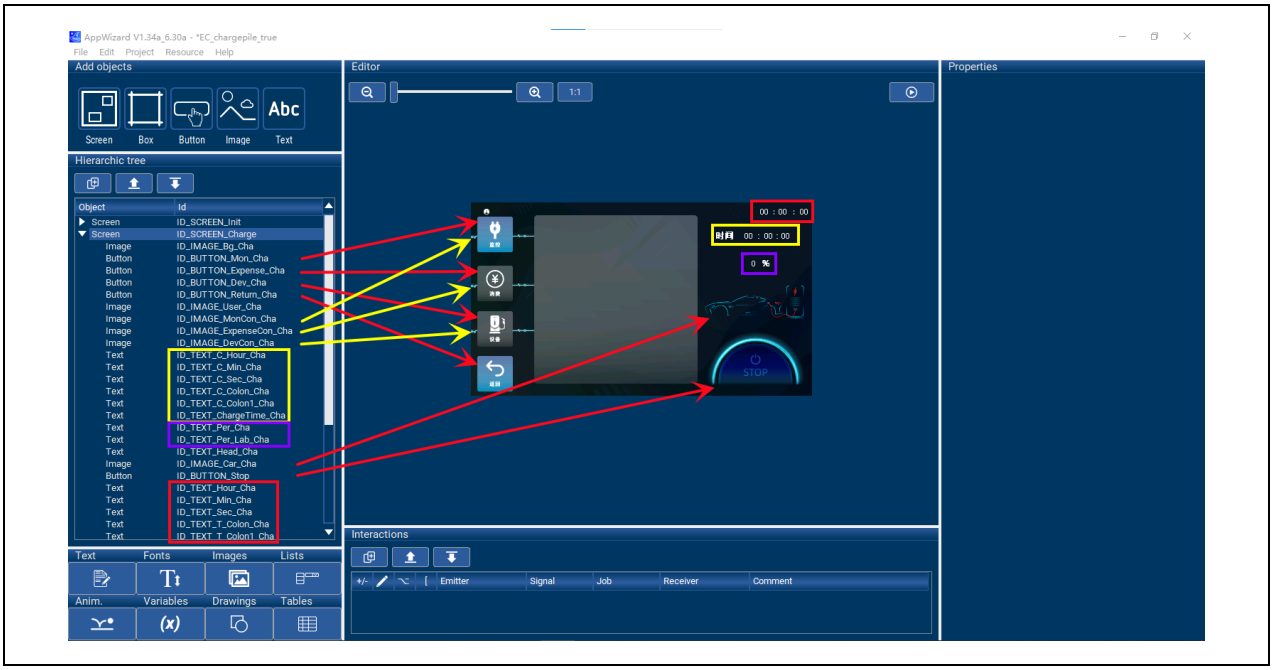


Figure 1-16 Add All Objects and Set Their Associated Properties

Figure 1-17 Set the Properties of the "Bg"

Figure 1-18 Set the Attributes of the "Mon", "Dev", "Expense" and "Return"

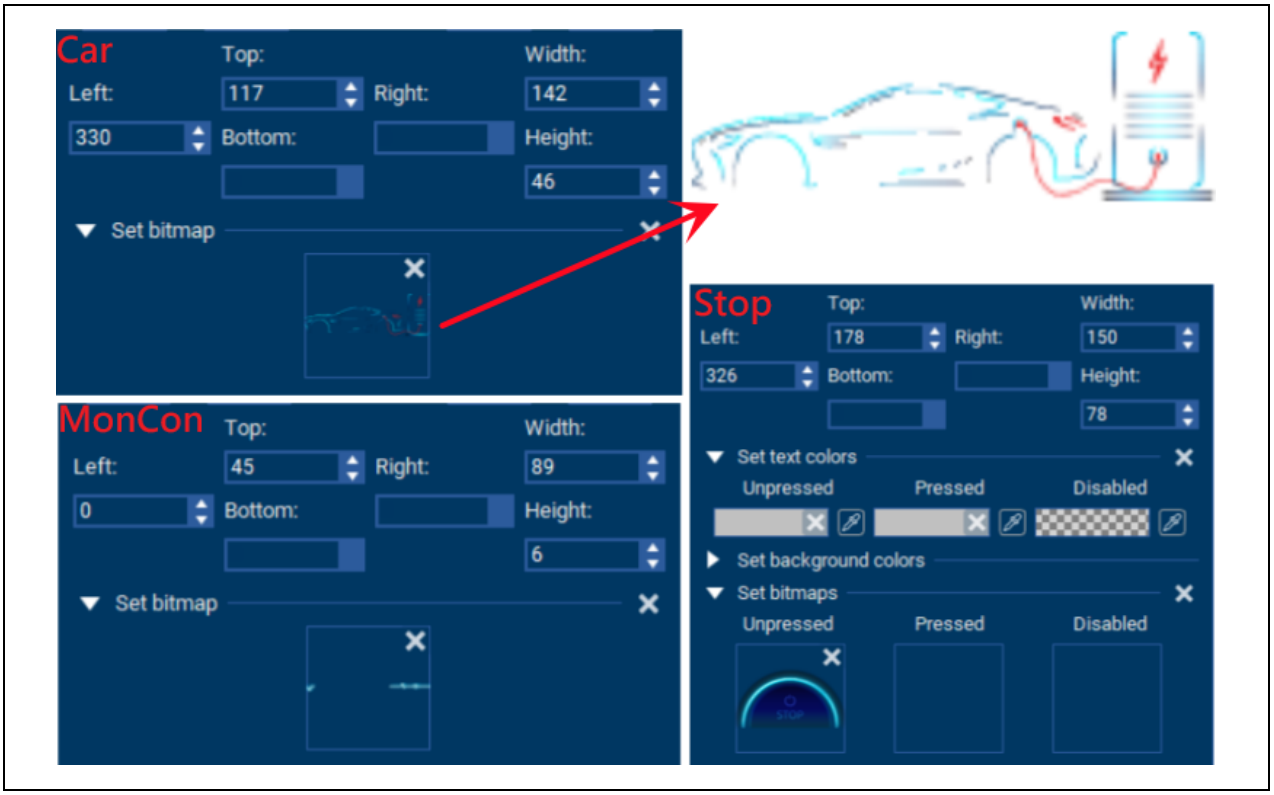


Figure 1-19 Set the Attributes of the "Car", "MonCon" and "Stop"

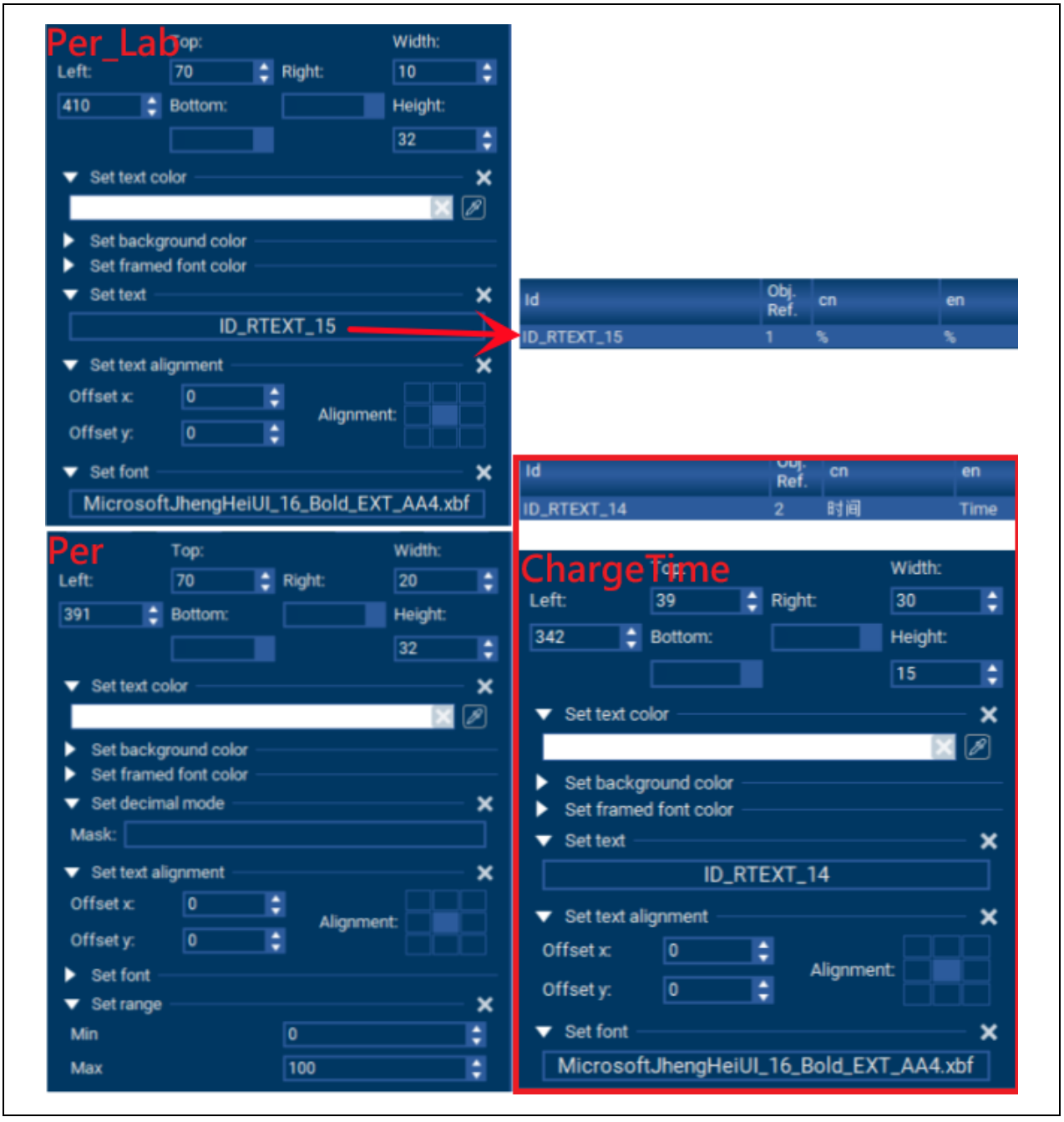


Figure 1-20 Set the Attributes of the "Per_Lab", "Per" and "ChargeTime"

2) Charge screen monitoring-window. Including cumulative electric capacity, cumulative consumption display, charging current and charging voltage display.

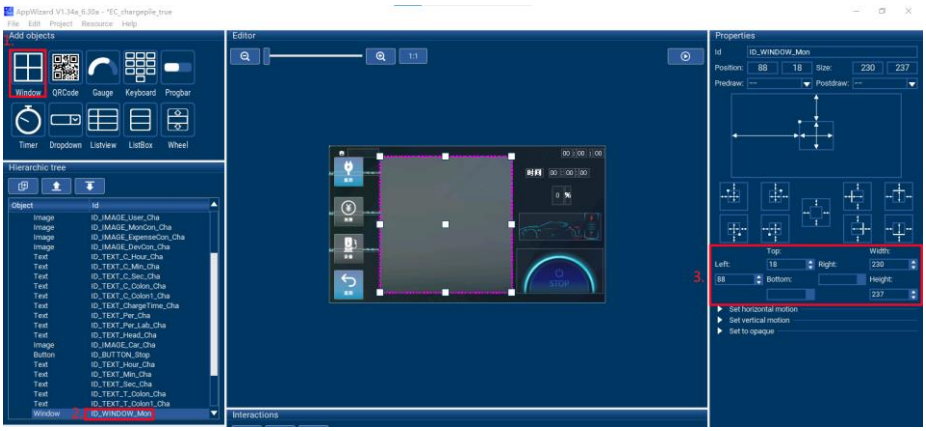


Figure 1-21 New "Mon" Window

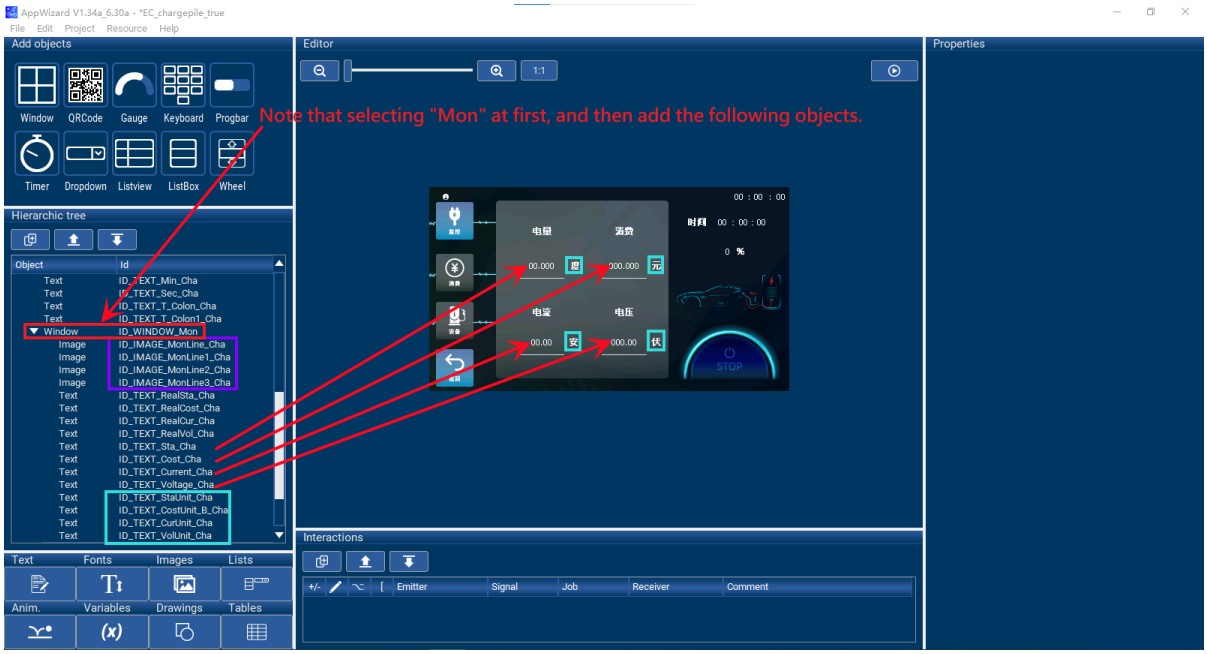


Figure 1-22 Add All Objects of the Window and Set Their Properties

Figure 1-23 Set the Properties of the "RealSta", "Realcost", "RealCur" and "RealVol"

Figure 1-24 Set the Properties of the "StaUnit", "CostUnit", "CurUnit" and "VolUnit"

Figure 1-25 Set the Properties of the "Sta", "Cost", "Current" and "Voltage"

3) Charge screen consumption-window, Refer to Step 3 (2) in Section 1.2.2. Including the display of consumption of electric capacities and costs in each time of the sharp, peak, flat and valley.

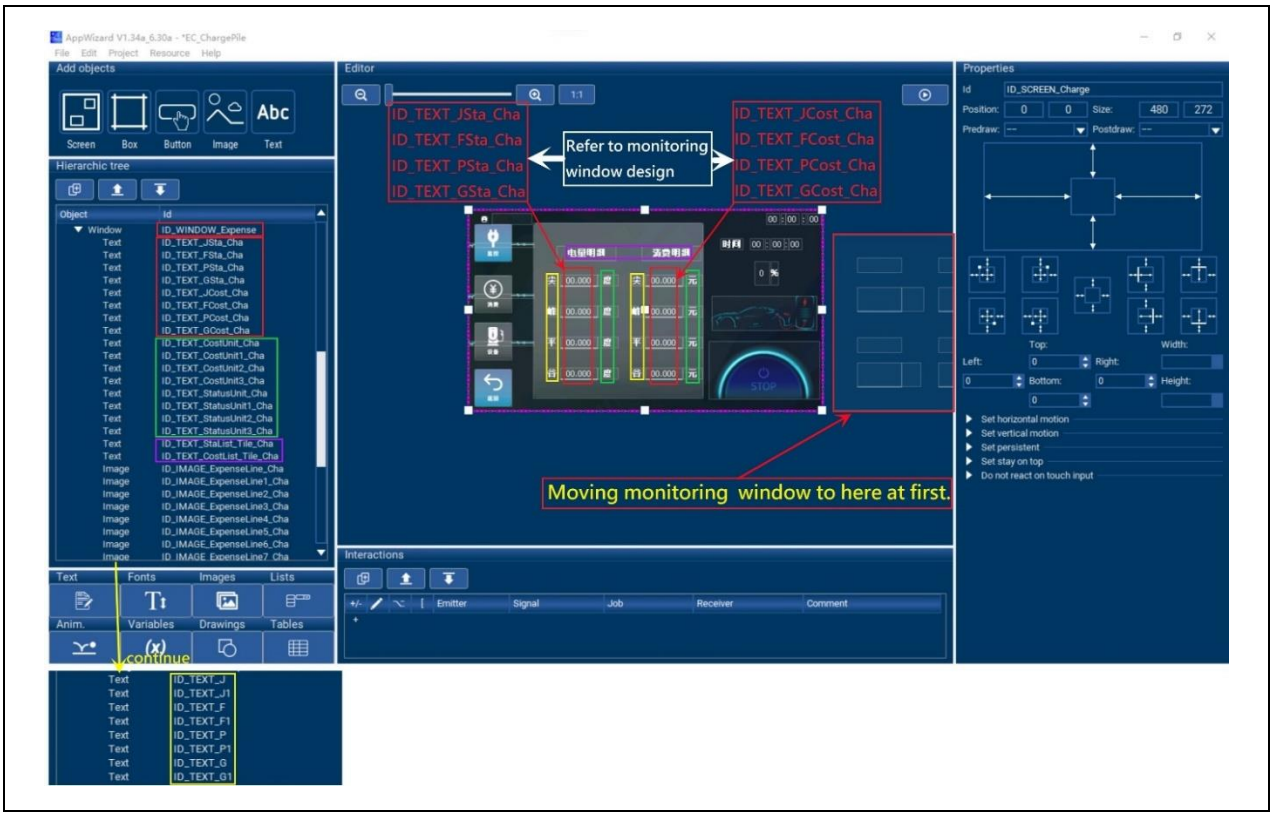


Figure 1-26 Add All Objects of the Window and Set Their Properties

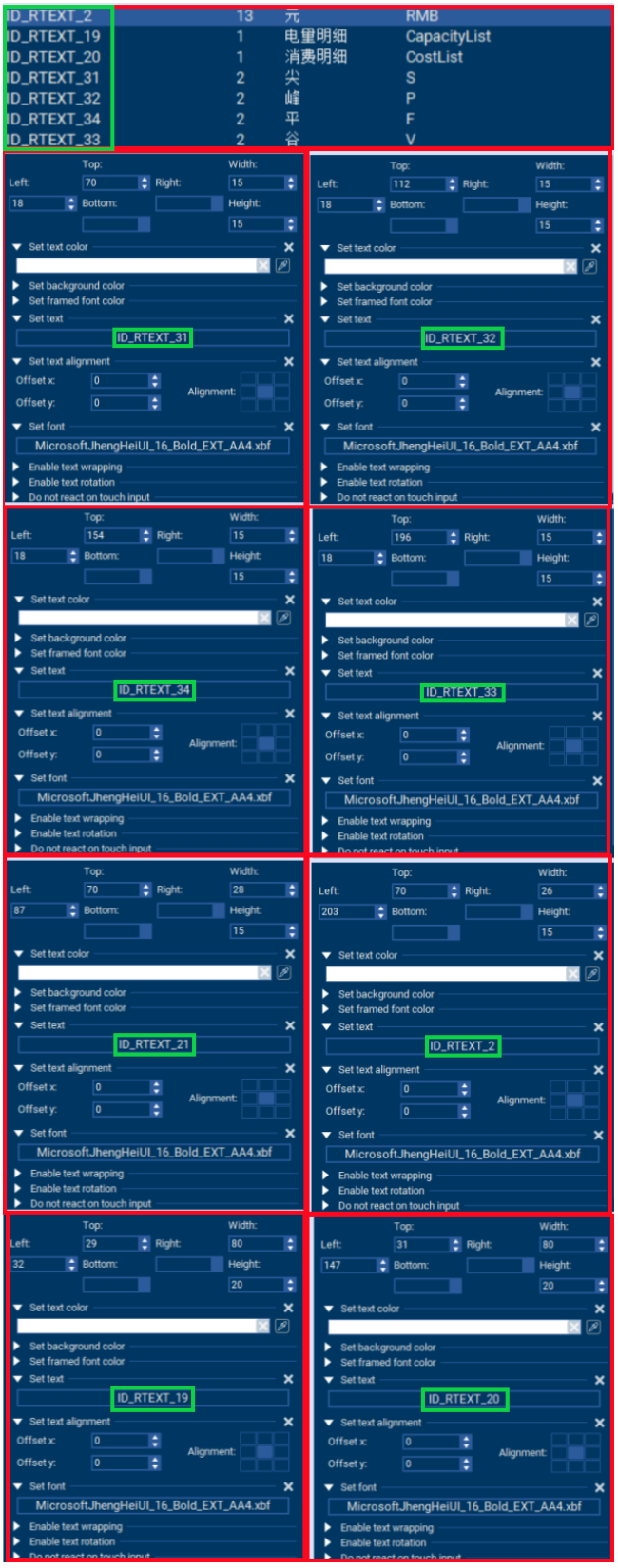


Figure 1-27 Set All Text Object Properties

4) Charge screen device-window, refer to Step 3 (2) in Section 1.2.2. Including the display of the power and the current-price.

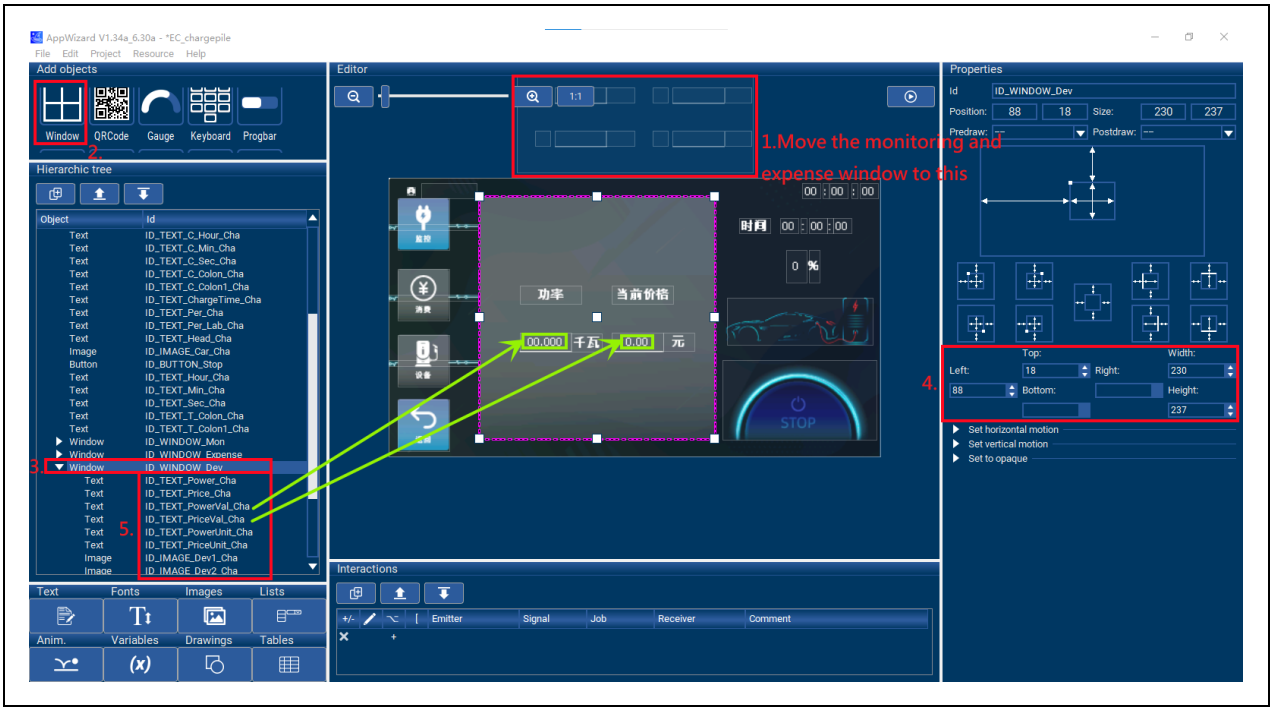


Figure 1-28 Add All Objects of the Window and Set Their Properties

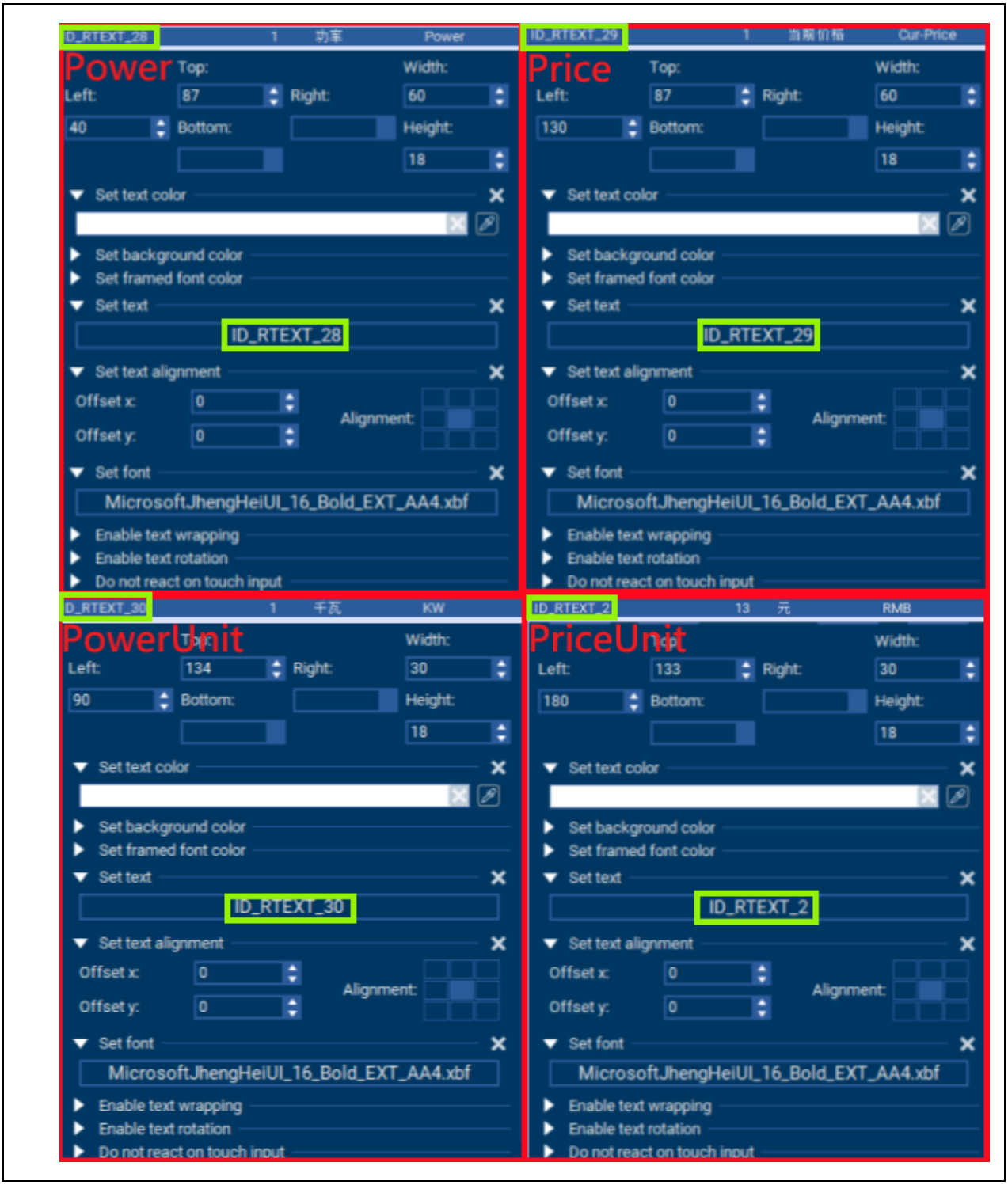


Figure 1-29 Set the Properties of the "Power", "Price", "PowerUnit" and "PriceUnit"

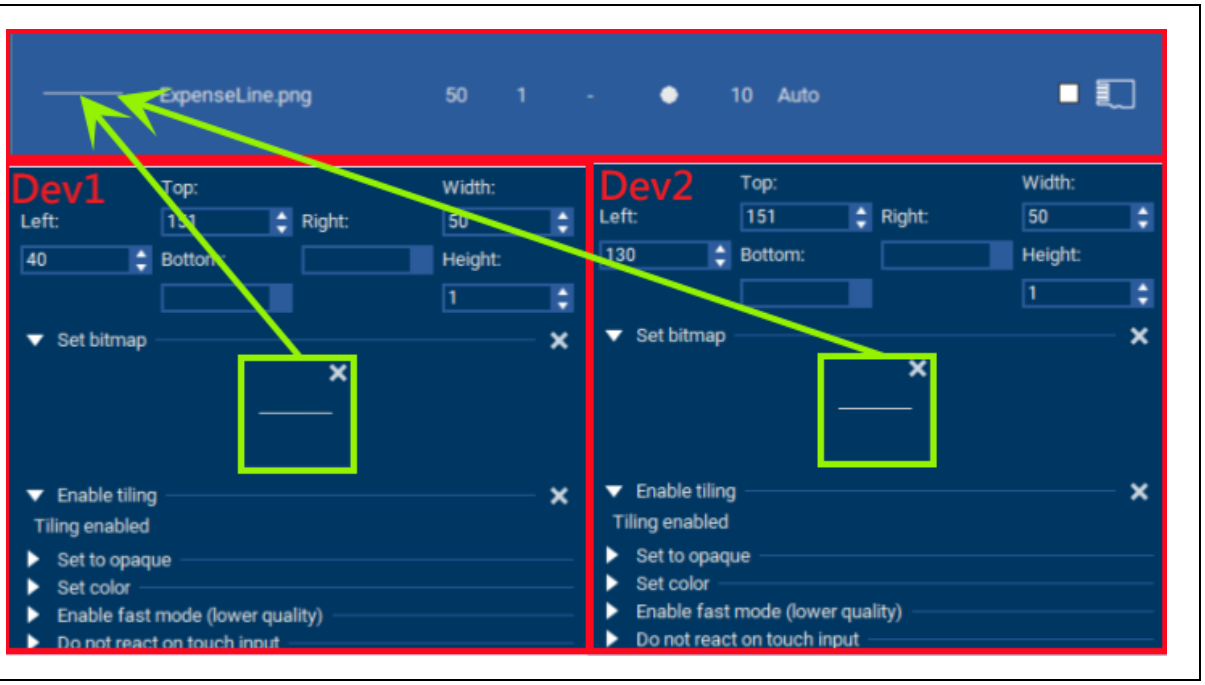


Figure 1-30 Set the Properties of the "Dev1" and the "Dev2"

5) Add two timers. It is used for real-time time display and charging timing display respectively.

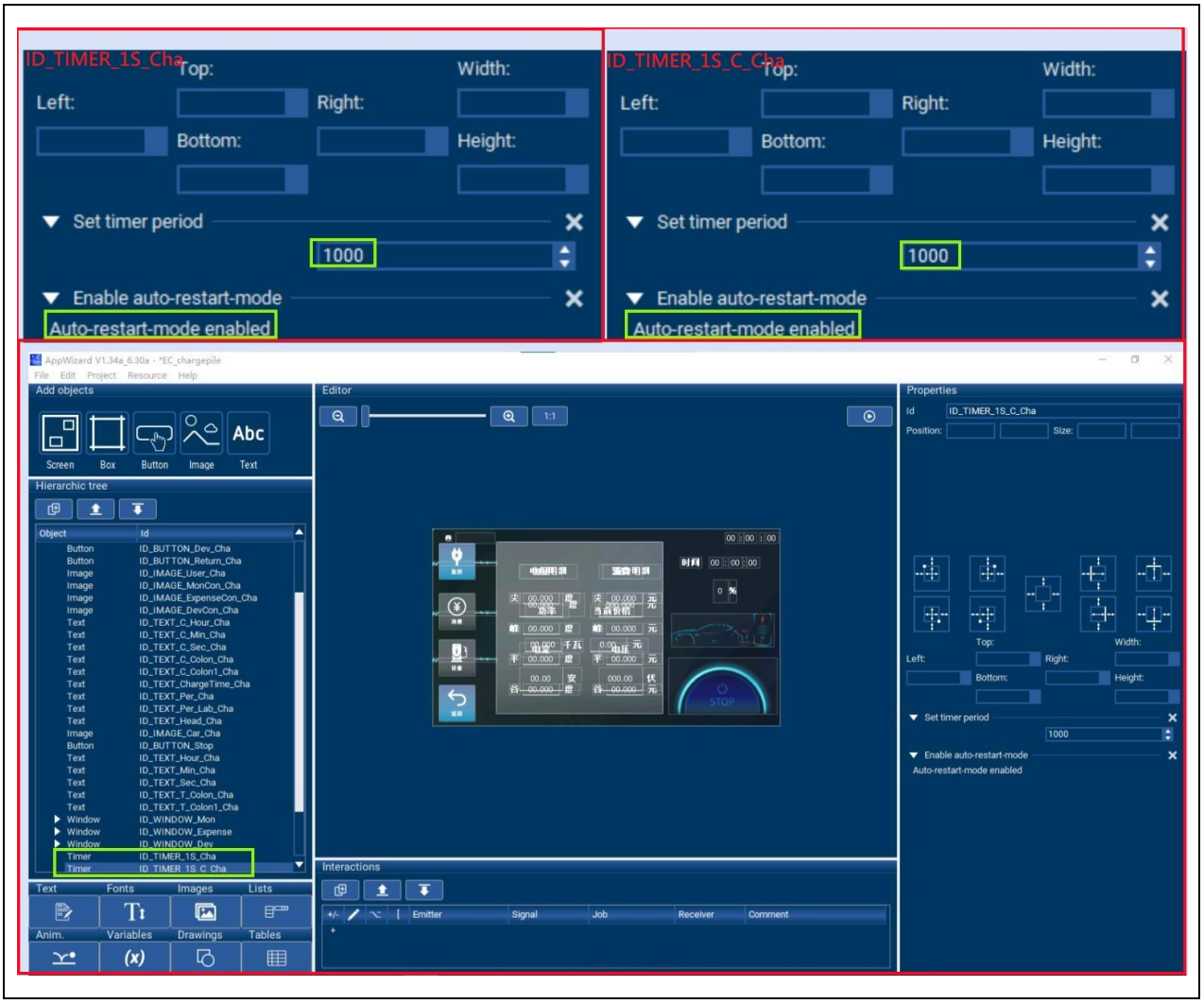


Figure 1-31 Set the Properties of the Timer

4. Price screen design. It mainly includes the price display of six time periods in a day.

    1) Displays prices for each time period. Refer to the "Init" screen design to add all the objects and set their properties.

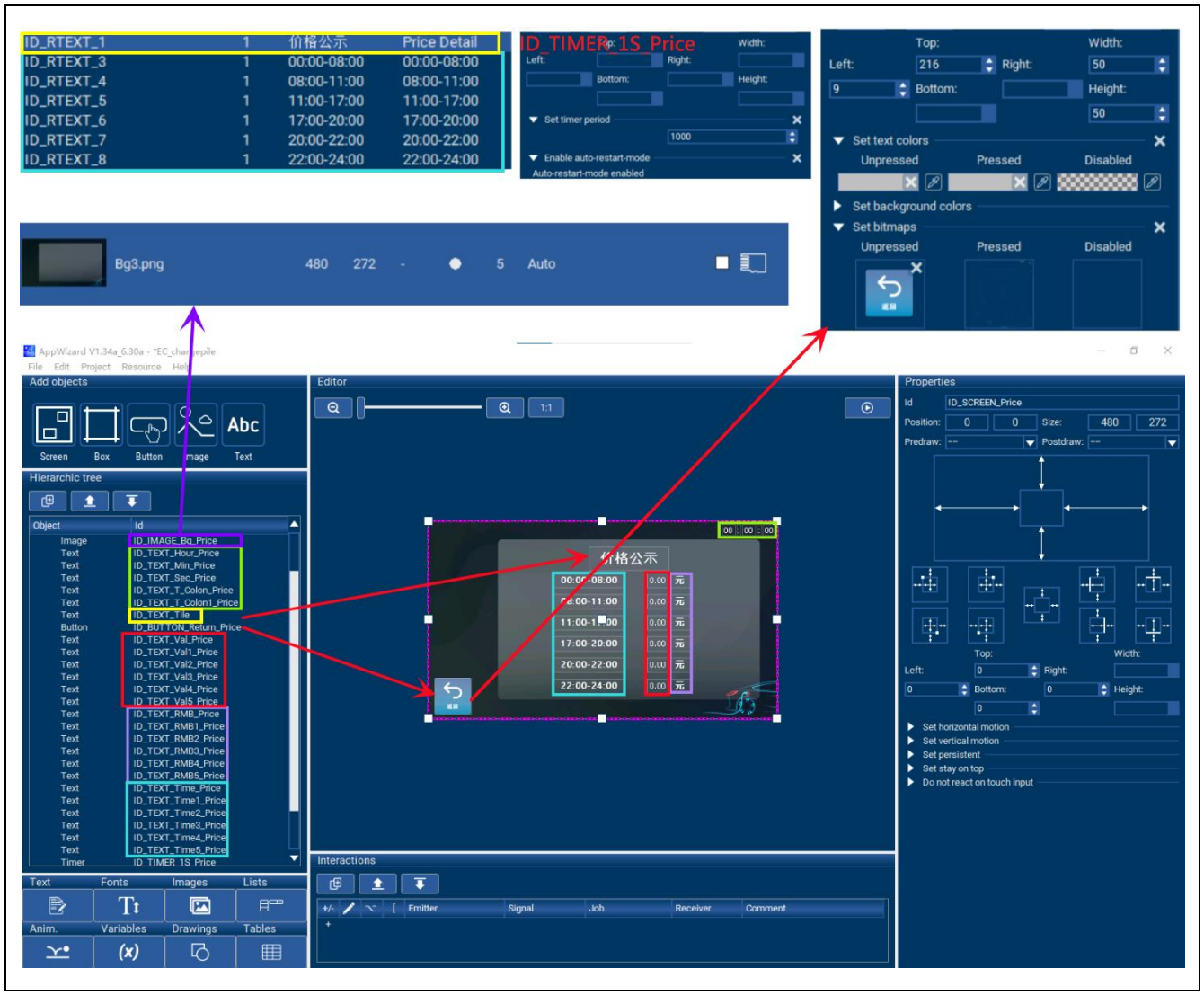


Figure 1-32 Add All Objects and Set Their Associated Properties

5. Info screen design. It mainly includes the name of the current charging station, the charging type and the number of supported charging gun.

    1) Display charging station, the type and number of guns. Refer to the "Init" screen design to add all the objects and set their properties.

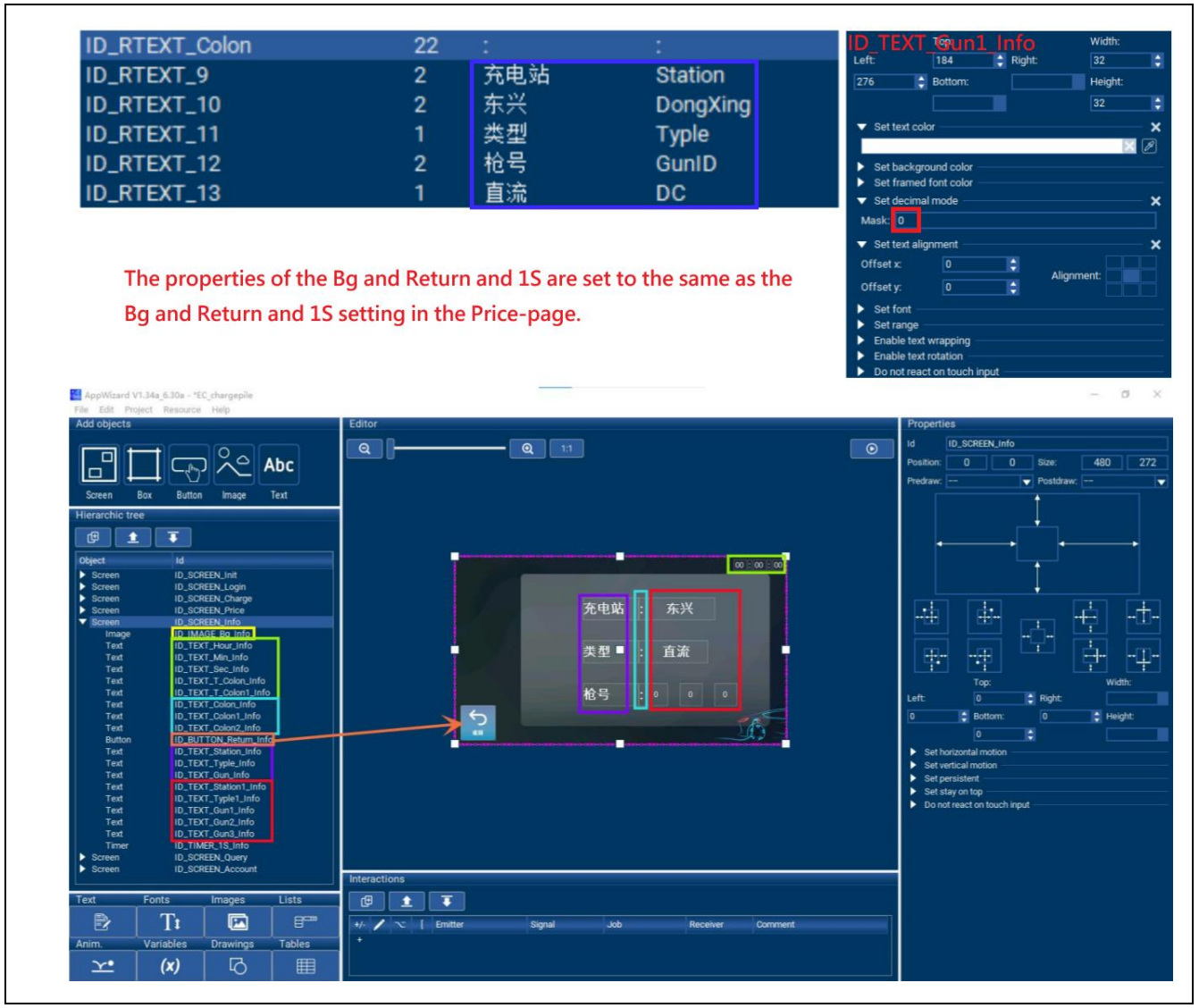


Figure 1-33 Add All Objects and Set Their Associated Properties

6. Query screen design. It mainly includes the last charging time, the cost, the name of the charging station and the number of charging gun.

    1) Display charging information. Refer to the "Init" screen design to add all the objects and set their properties.

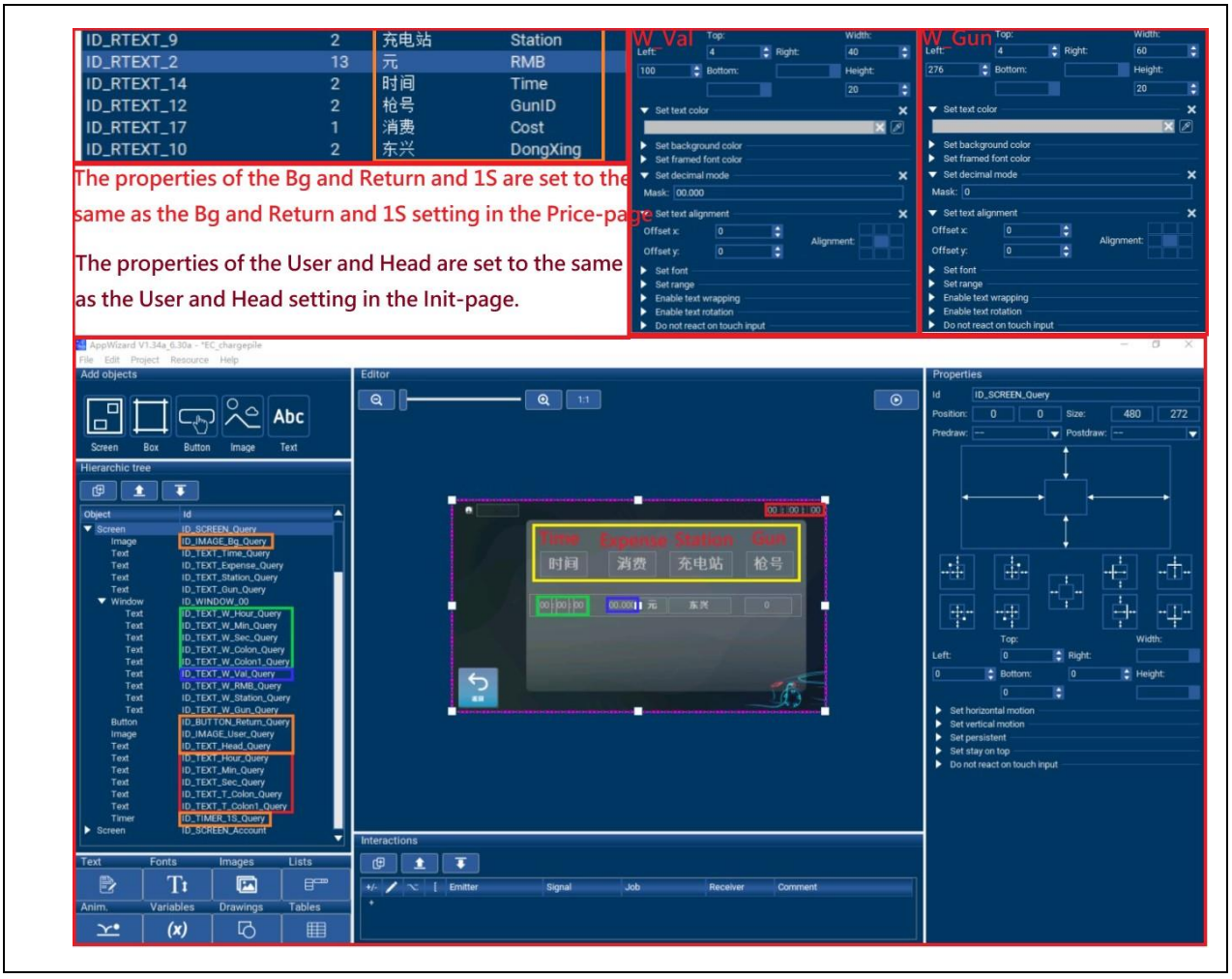


Figure 1-34 Add All Objects and Set Their Associated Properties

7. Account screen design. It mainly includes account name and the balance and the number of mobile Phone.

    1) Show the name and the balance and mobile number of the account. Refer to the "Init" screen design to add all the objects and set their properties.

Figure 1-35 Add All Objects and Set Their Associated Properties

8. Login screen design. It mainly includes receiving the username and password entered by the user and displaying password status.

    1) Enter the username and password then display the password status. Refer to the "Init" screen design to add all the objects and set their properties.

Figure 1-36 Add All Objects and Set Their Associated Properties

1.2.3 Interactions Designed in AppWizard

1. Login screen interaction design. This includes clicking the "Return" button to return to the "Init" screen. Receive the username and password entered by the user. Show in real-time whether the password is entered correctly in combination with the code.

    1) Click the "Return" button to return to the "Init" screen.

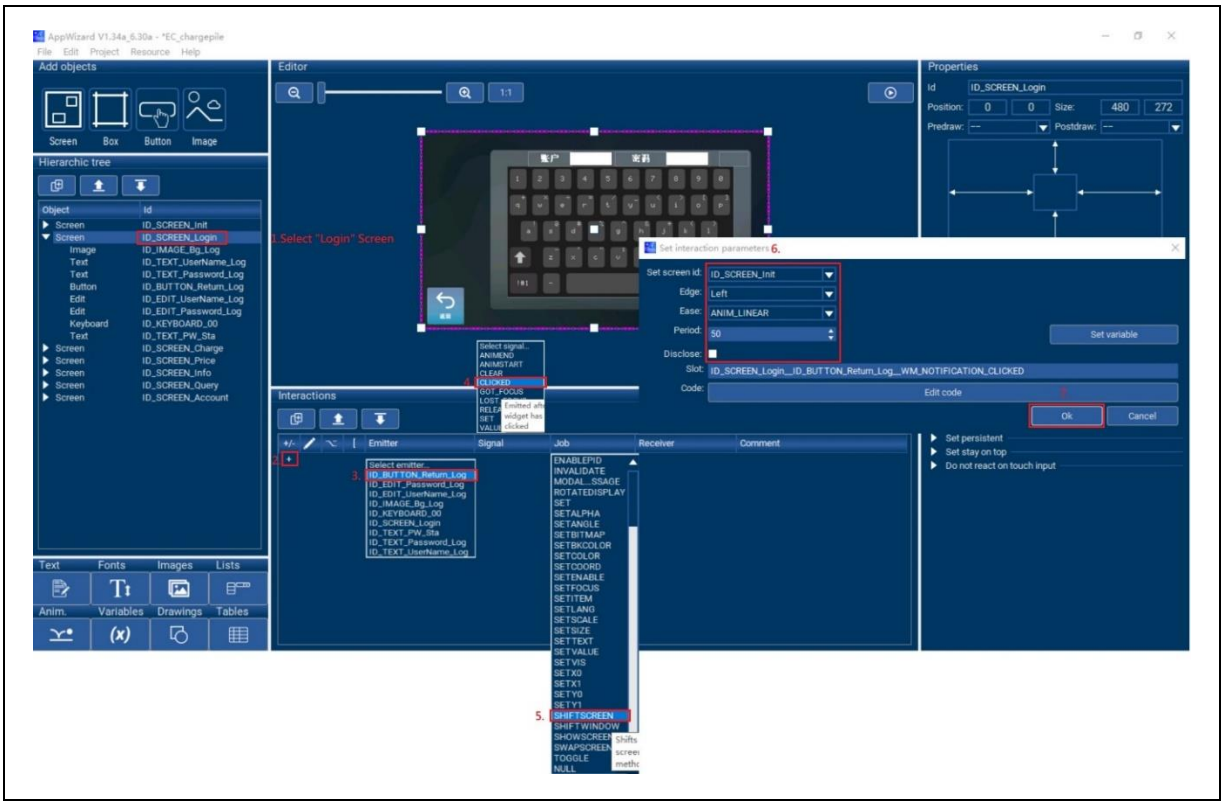


Figure 1-37 Click the "Return" Button to Return to the "Init" Screen

2) Load the "Login" screen and focus on the "UserName" object.

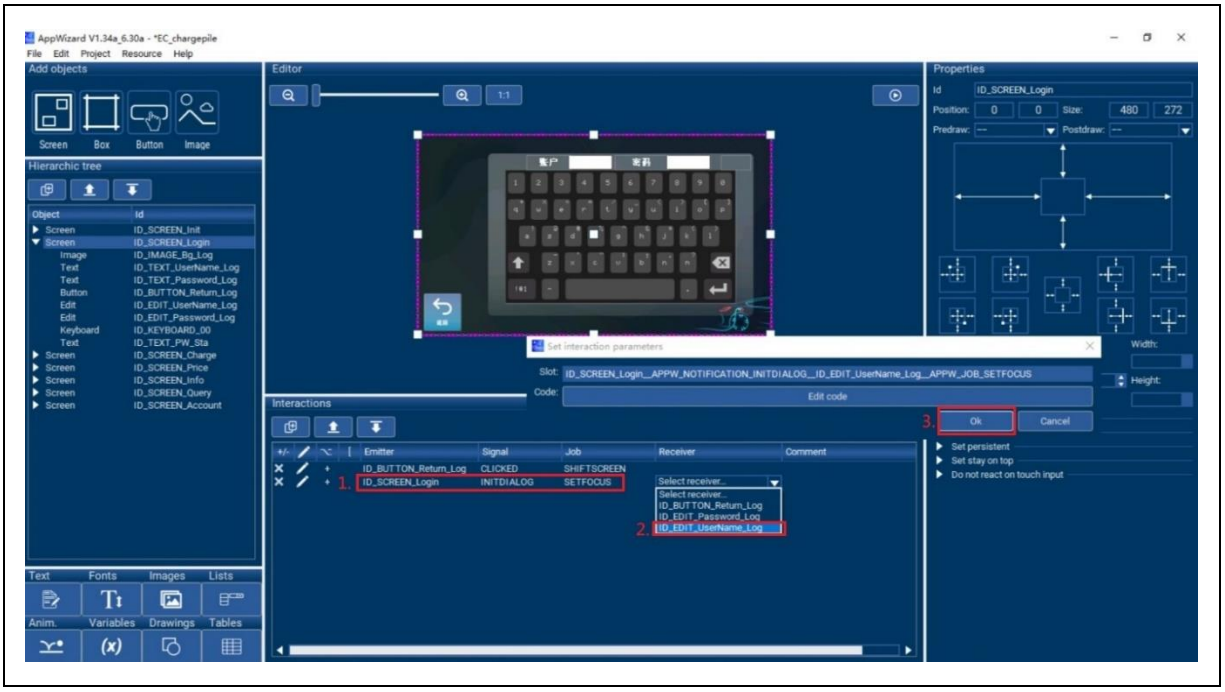


Figure 1-38 Load the "Login" Screen and Focus on the "UserName" Object

3) Load the "Login" screen and set a different pattern for the "Return" depending on the "Lan" value.

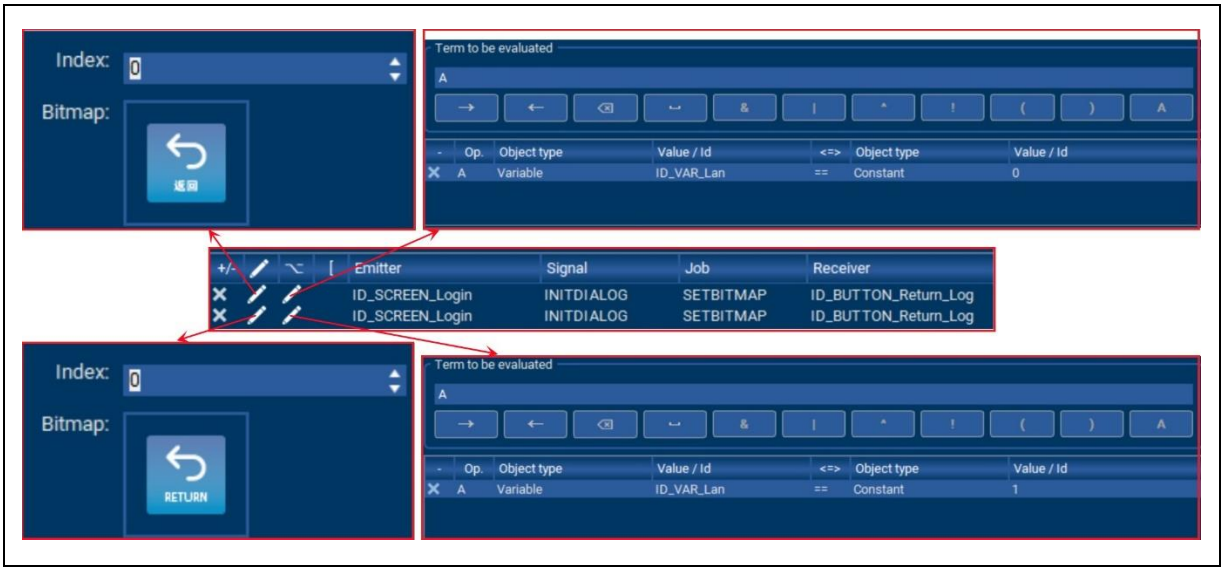


Figure 1-39 Set a Different Pattern for the "Return" when Loading the "Login" Screen

4) Define the password status variable "PW_1". If the password is correct, "PW_1" is set to 1 by the code, and if the password is incorrect, "PW_1" is set to 0 by the code. The default password is "qqqqqqqq".

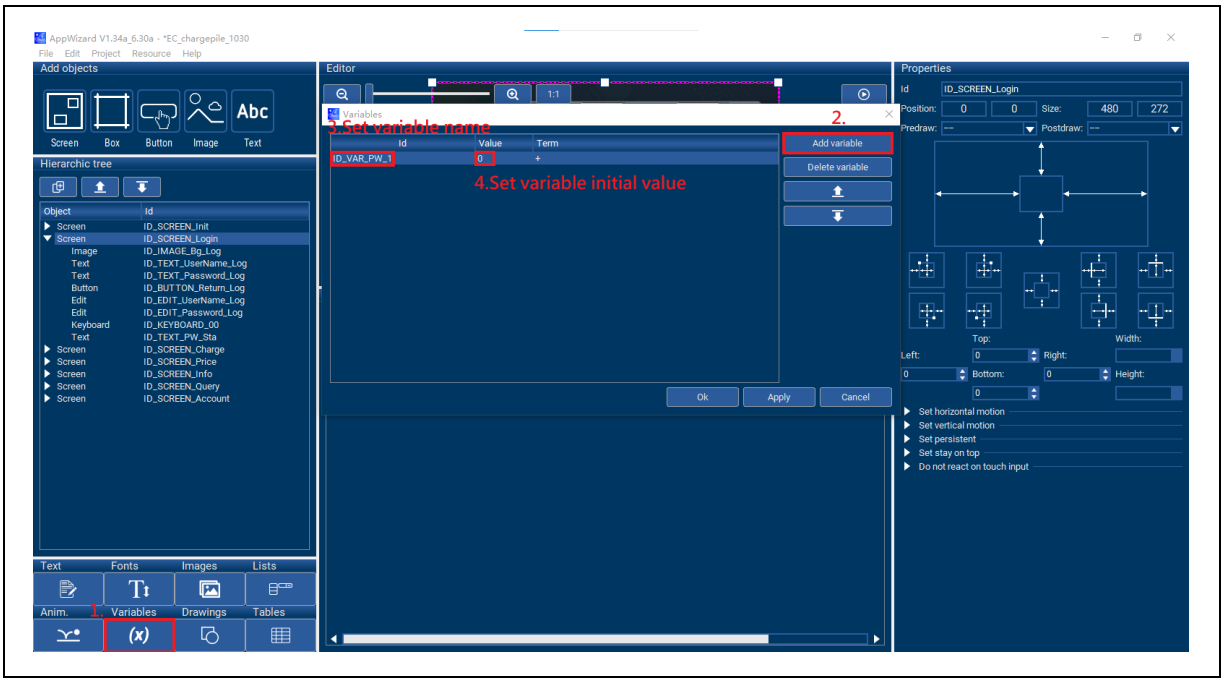


Figure 1-40 "PW_1" Variable Definition

5) User name and password processing logic.

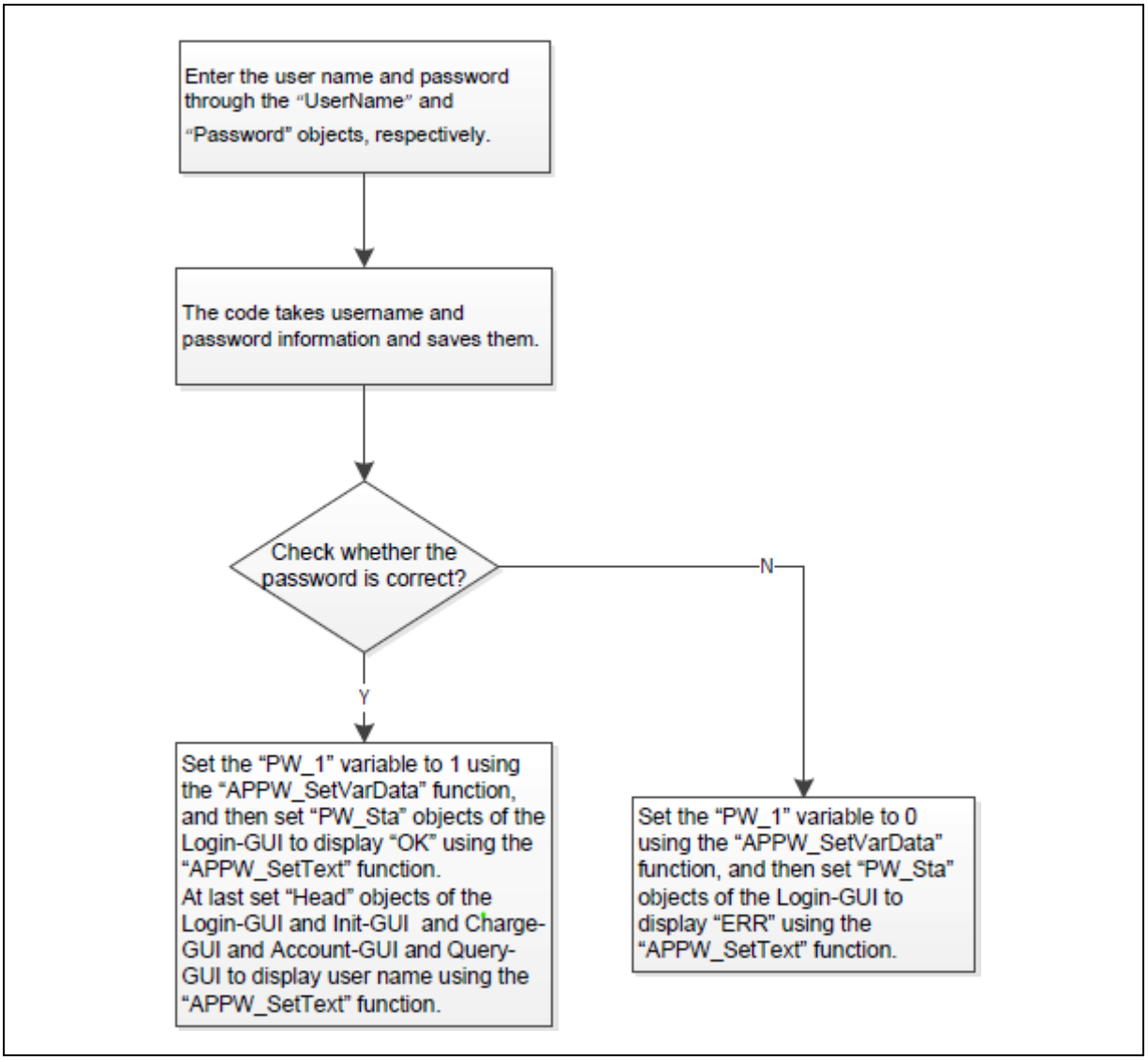


Figure 1-41 User Name and Password Processing Logic

2. Init screen interaction design. Including real-time time display, username display combined with code, clicking different buttons to switch to different screen and clicking the "Lan" button to switch between Chinese and English. The button pattern of the "Charge", the "Account" and the "Query" is displayed differently according to the value of "PW_1" variable.

    1) Define the variables required for real-time time display ("Hour", "Min", "Sec", "SecCnt").

Figure 1-42 Variable Definition ("Hour", "Min", "Sec", "SecCnt")

2) Real-time time display. Once powered on, the time starts at 00:00:00.

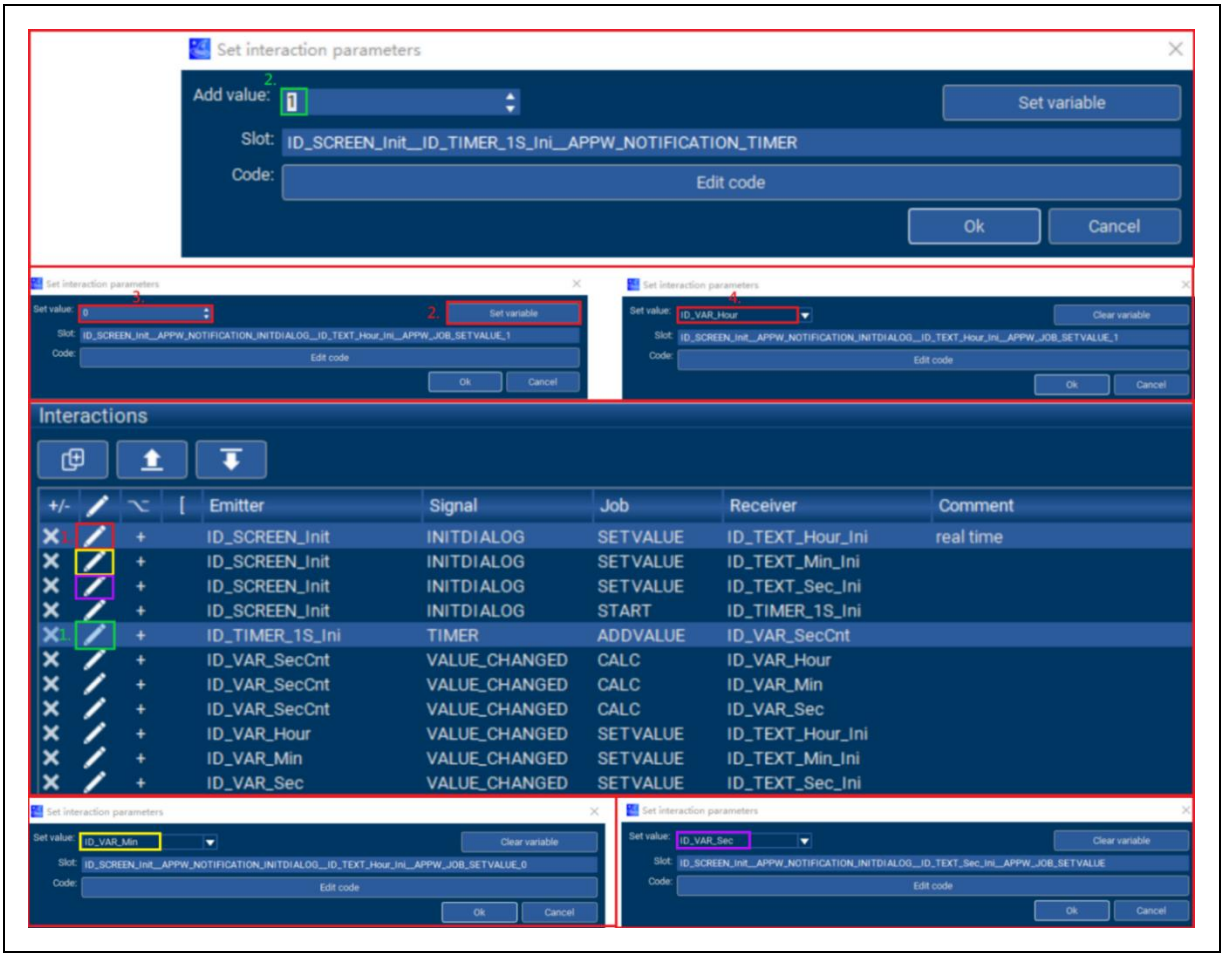


Figure 1-43 Real-time Time Display

3) Define the language variable "Lan". Zero indicates Chinese and one indicates English.

Figure 1-44 Define the Language Variable

4) Load "Init" screen to update the pattern of "Lan" button.

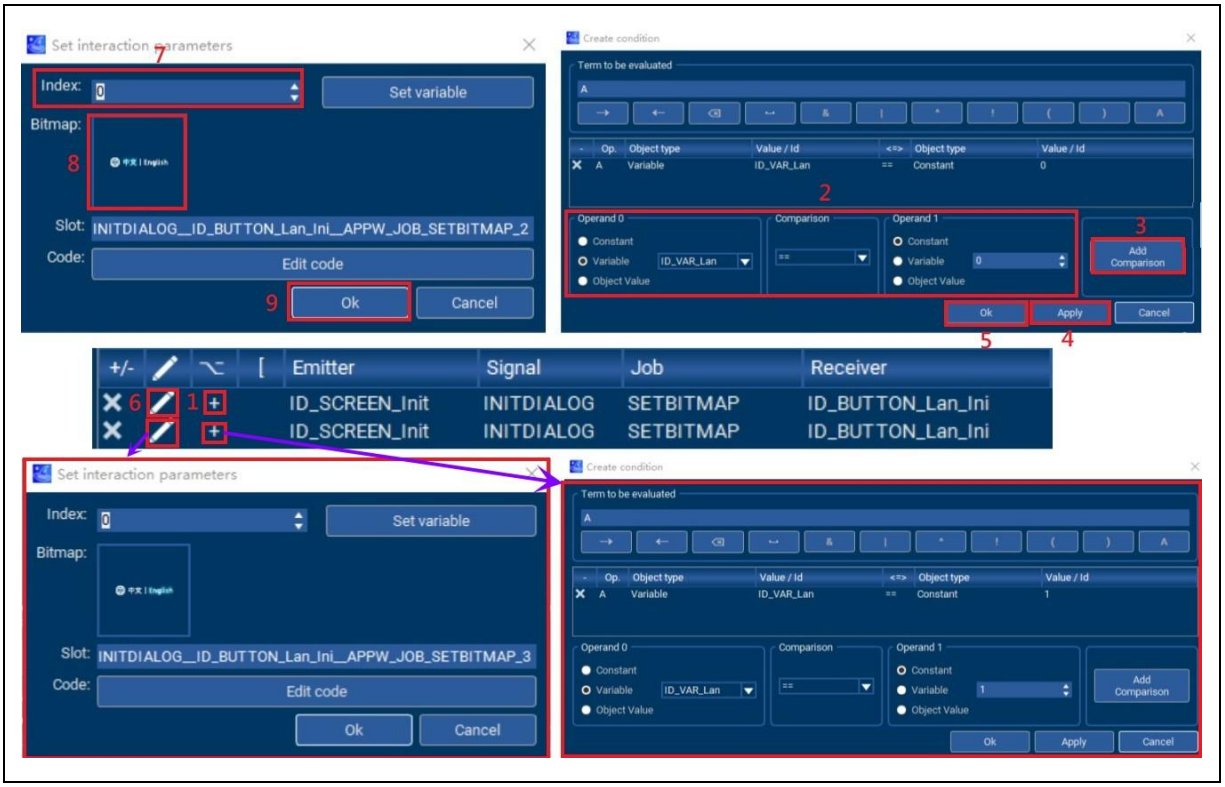


Figure 1-45 Load "Init" Screen to Update the Pattern of "Lan" Button

5) Load "Init" screen and set the "Charge" pattern.

Figure 1-46 Set a Different Pattern for the "Charge" Button when Loading the "Init" Screen

6) Load the "Init" screen and refresh the pattern of the "Bg", "Login", "Price" and "Info", refer to Step 2 (4) in Section 1.2.3. Load "Init" screen and refresh the pattern of the "Acc" and the "Query", refer to Step 2 (5) in Section 1.2.3.

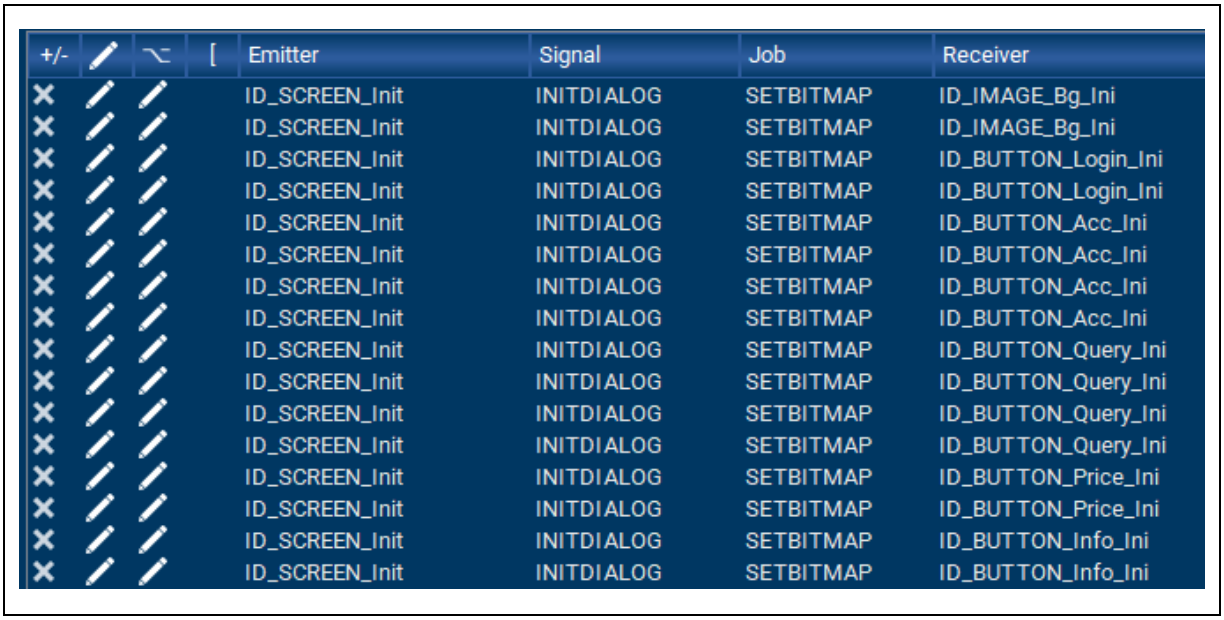


Figure 1-47 Refresh the Pattern of "Bg", "Login", "Acc" and "Query"

7) Click the "Lan" button to change the value of the "Lan" variable, which ranges from 0 to 1.

Figure 1-48 Click the "Lan" Button to Change the Value of the "Lan" Variable

8) Click the "Lan" button to update the "Lan" button pattern.

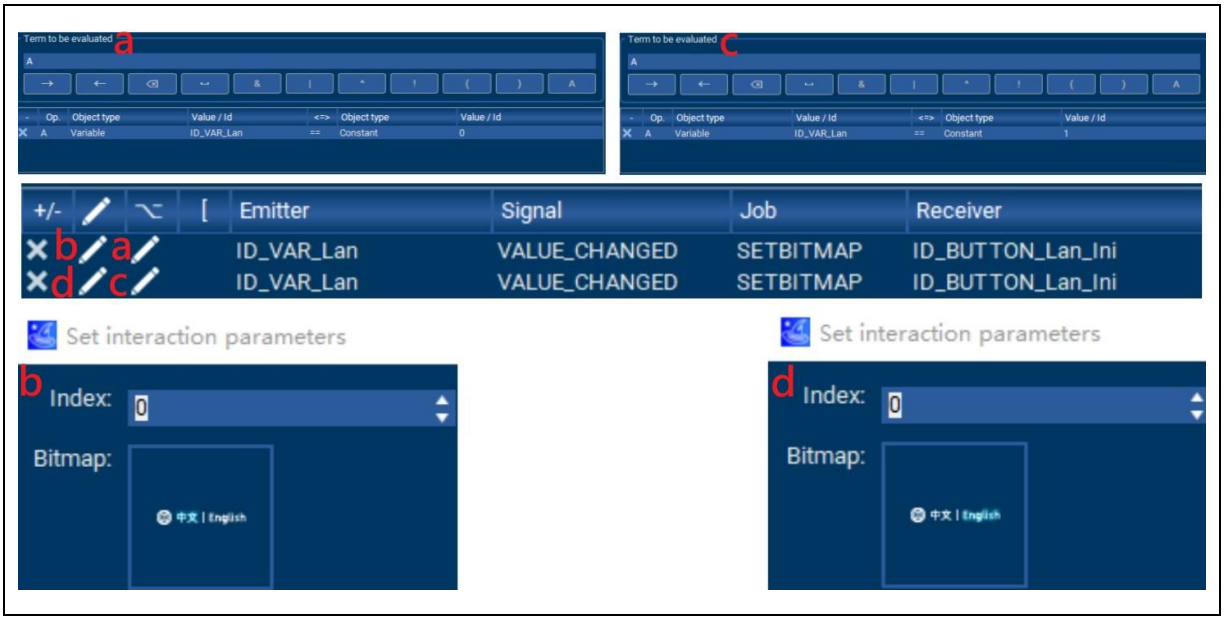


Figure 1-49 Click the "Lan" Button to Update the "Lan" Button Pattern

9) Set "Lan" variable to 0 to correspond to language index 0 and set "Lan" variable to 1 to correspond to language index 1.

Figure 1-50 Set the Mapping between Values of "Lan" Variable and Language Indexes

10) Click the "Login", "Price", "Info", "Charge", "Account" or "Query" to switch the screen.

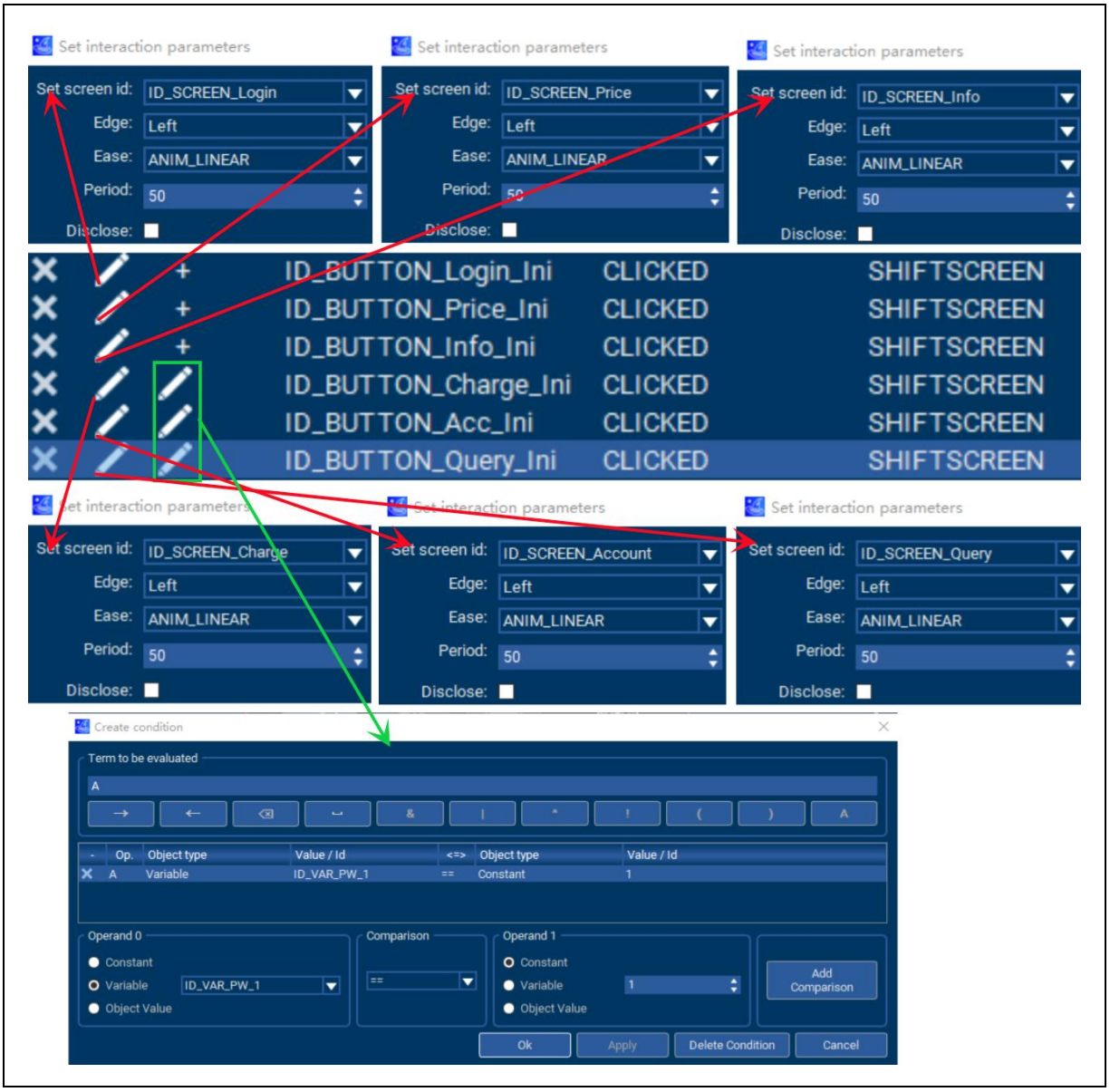


Figure 1-51 Click the Button to Switch Screen

11) The button patterns of the "Charge", the "Account" and the "Query" are displayed differently according to the "PW_1" variable.

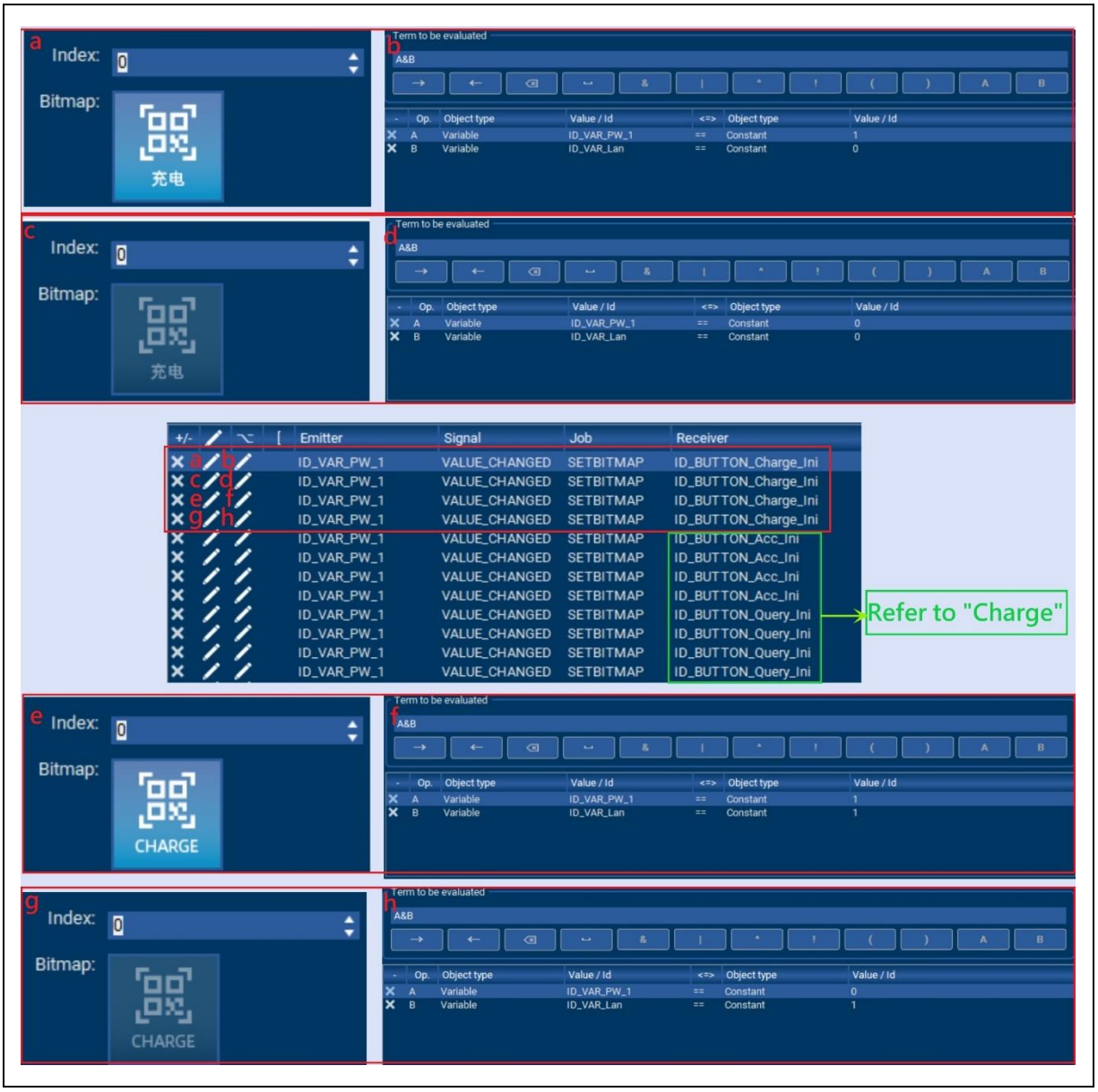


Figure 1-52 The "Charge", "Account" and "Query" Buttons Pattern Display

12) "Lan" variable changing with the value of the "PW_1" variable updates the patterns of "Login", "Charge", "Account", "Query", "Price", "Info" and "Bg".

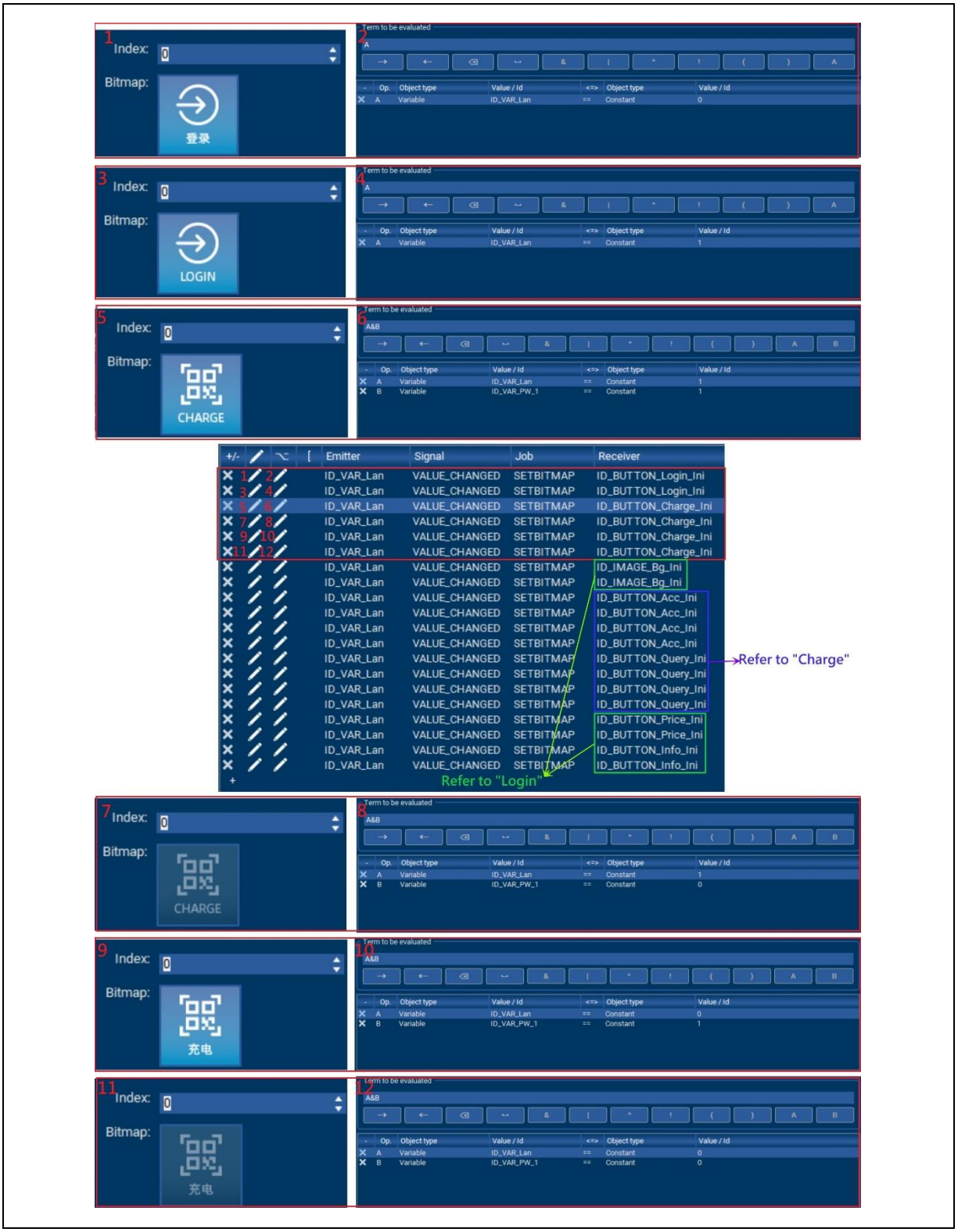


Figure 1-53 All Buttons and Background Pattern Display

3. Charge screen interaction design. This includes clicking the "Return" button to return to the "Init" screen and displaying real-time time. The username is displayed in combination with the code. Display and control charging status, including charging time. The voltage, the current and the power are displayed in real-time. Displaying current electricity price. The total charging capacity and the total charging cost as well as the charging capacity and the charging cost in each period of the sharp, peak, flat and valley are displayed. The charging percentage will be displayed in the form of numbers and graphs. Charging can be started or stopped by clicking a button.

   1) Define the "Charging" variable, where 0 means stop-charging and 1 means charging.

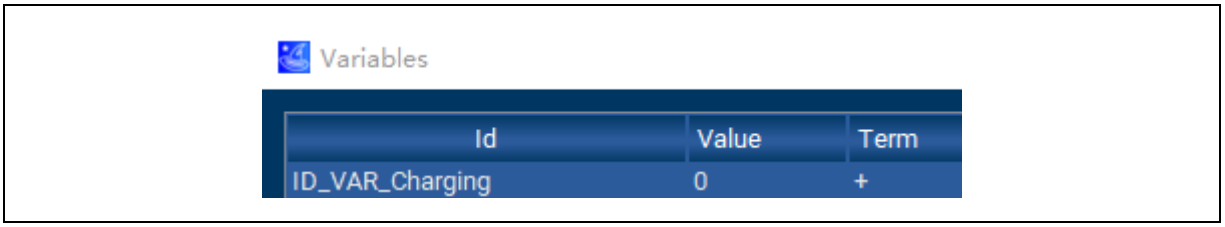


Figure 1-54 Define the "Charging" Variable

   2) Clicking the "Return" button to return to the "Init" screen.

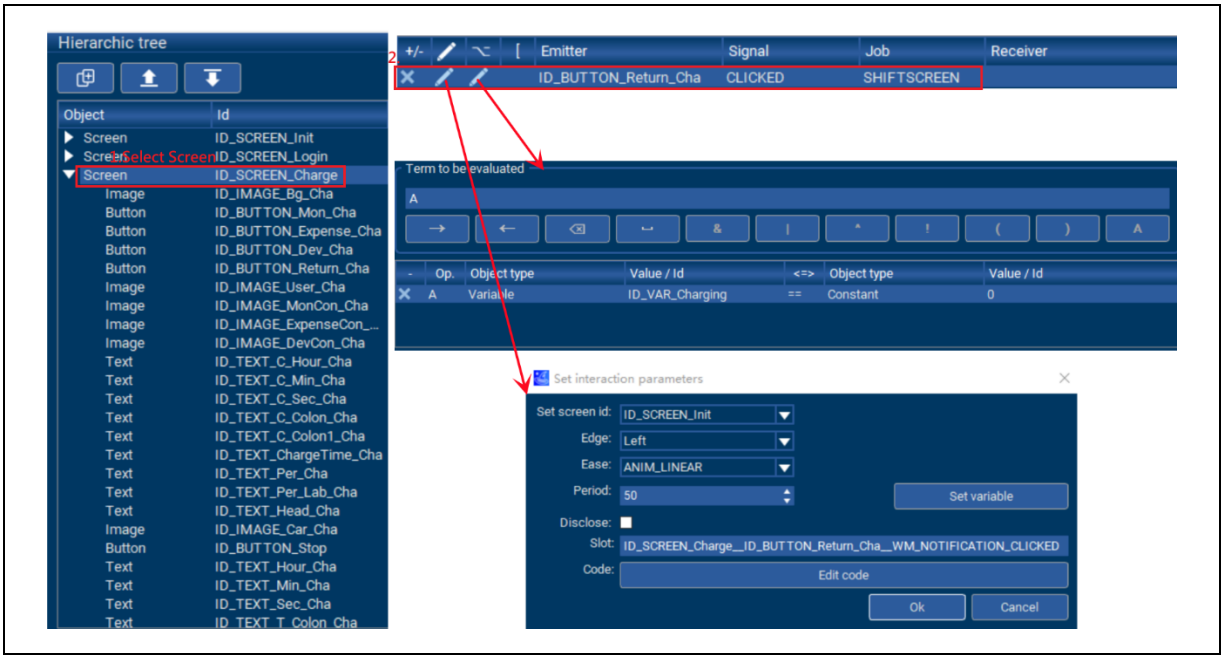
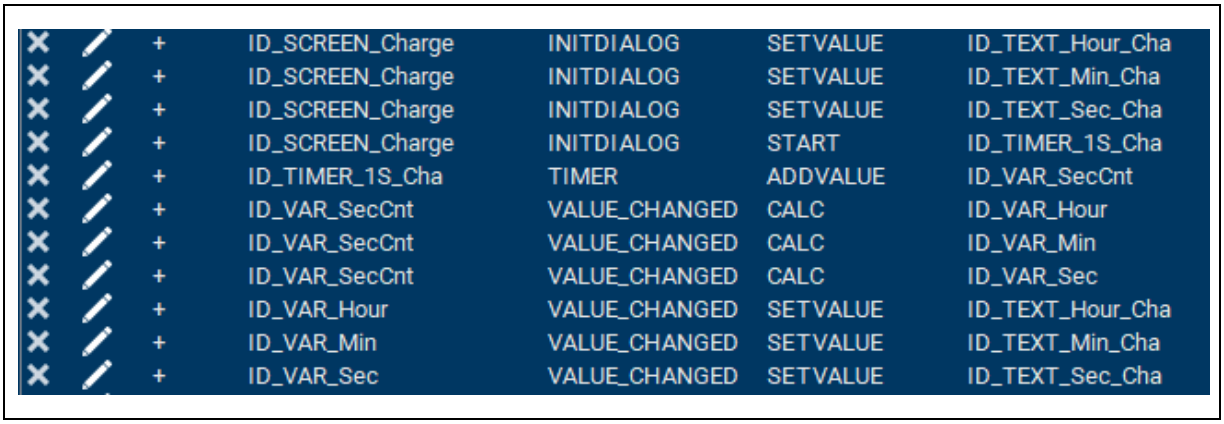


Figure 1-55 Clicking the "Return" Button to Return to the "Init" Screen

3) Real-time time display design. Refer to Step 2 (2) in Section 1.2.3.



+	ID_SCREEN_Charge	INITDIALOG	SETVALUE	ID_TEXT_Hour_Cha
+	ID_SCREEN_Charge	INITDIALOG	SETVALUE	ID_TEXT_Min_Cha
+	ID_SCREEN_Charge	INITDIALOG	SETVALUE	ID_TEXT_Sec_Cha
+	ID_SCREEN_Charge	INITDIALOG	START	ID_TIMER_1S_Cha
+	ID_TIMER_1S_Cha	TIMER	ADDVALUE	ID_VAR_SecCnt
+	ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Hour
+	ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Min
+	ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Sec
+	ID_VAR_Hour	VALUE_CHANGED	SETVALUE	ID_TEXT_Hour_Cha
+	ID_VAR_Min	VALUE_CHANGED	SETVALUE	ID_TEXT_Min_Cha
+	ID_VAR_Sec	VALUE_CHANGED	SETVALUE	ID_TEXT_Sec_Cha

Figure 1-56 Real-time Time Display

4) Define charge-related variables ("ConstantVolMode", "Charge_Hour", "Charge_Min", "Charge_Sec", "Charge_SecCnt"). "ConstantVolMode" equal to 0 is represented as constant current mode, "ConstantVolMode" equal to 1 is represented as constant voltage mode.

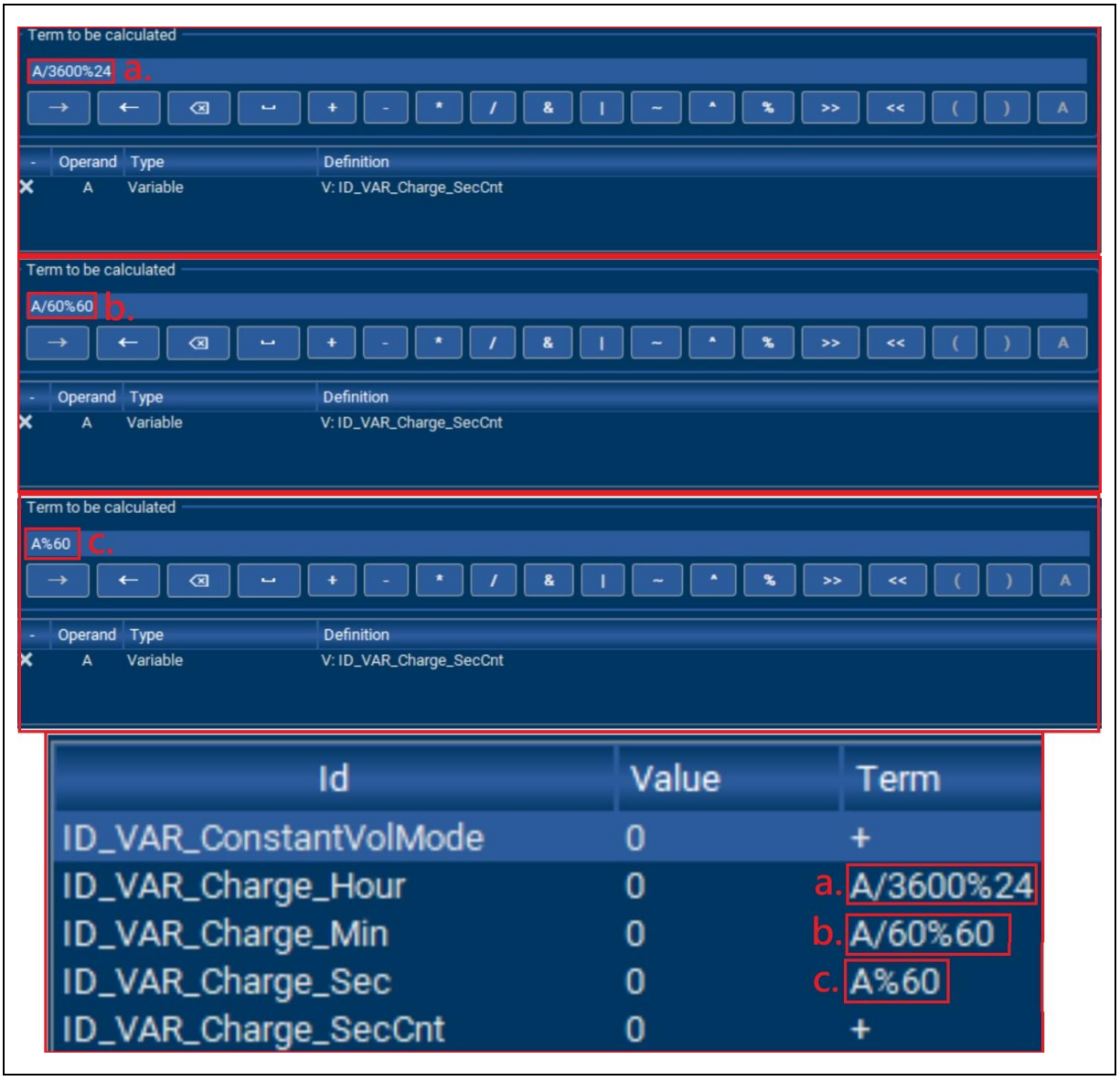


Figure 1-57 Define Charge-related Variables

5) Charging time display. Refer to Step 3 (3) in Section 1.2.3.

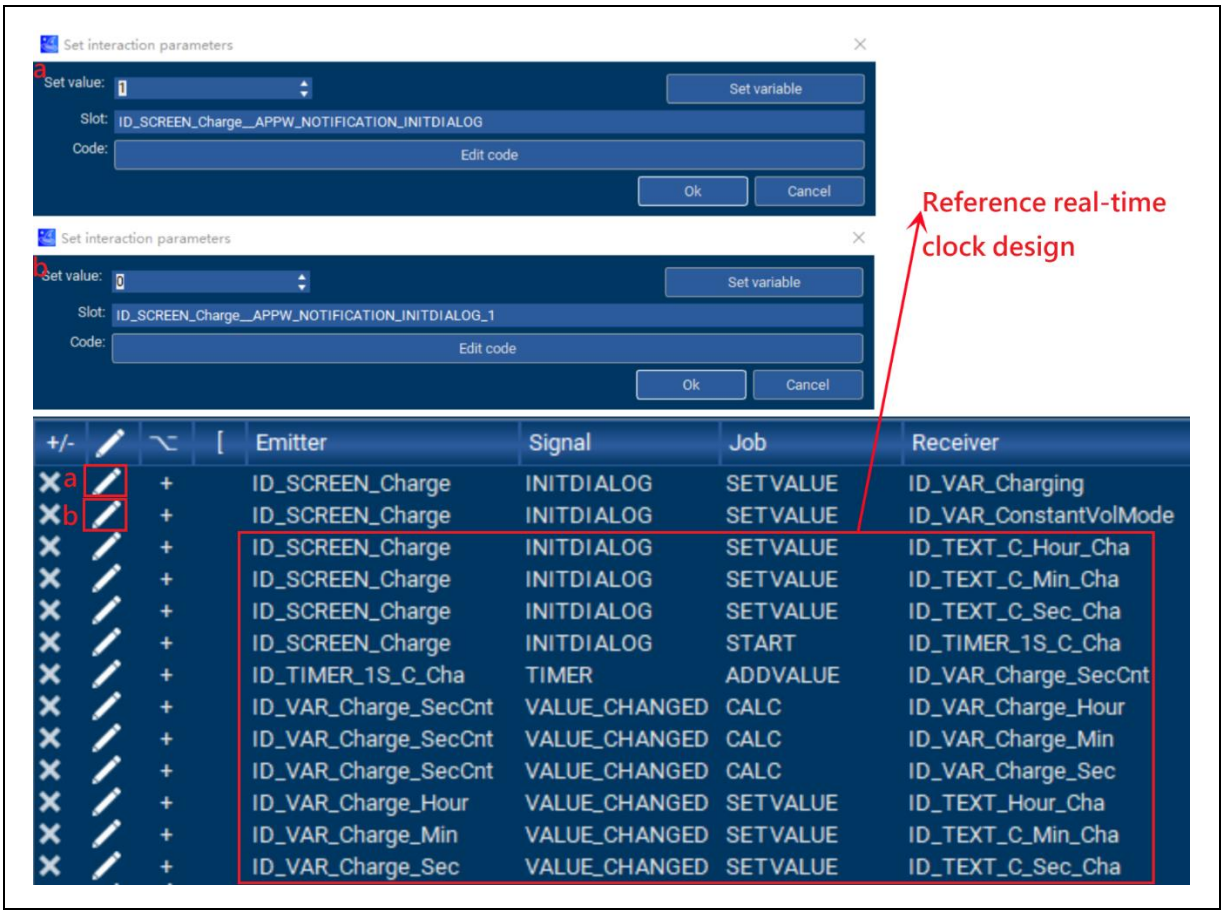


Figure 1-58 Charging Time Display

6) Charging mode updated. "ChargeSecCnt" Less than 60 indicates constant current mode, and greater than or equal to 60 indicates constant voltage mode.

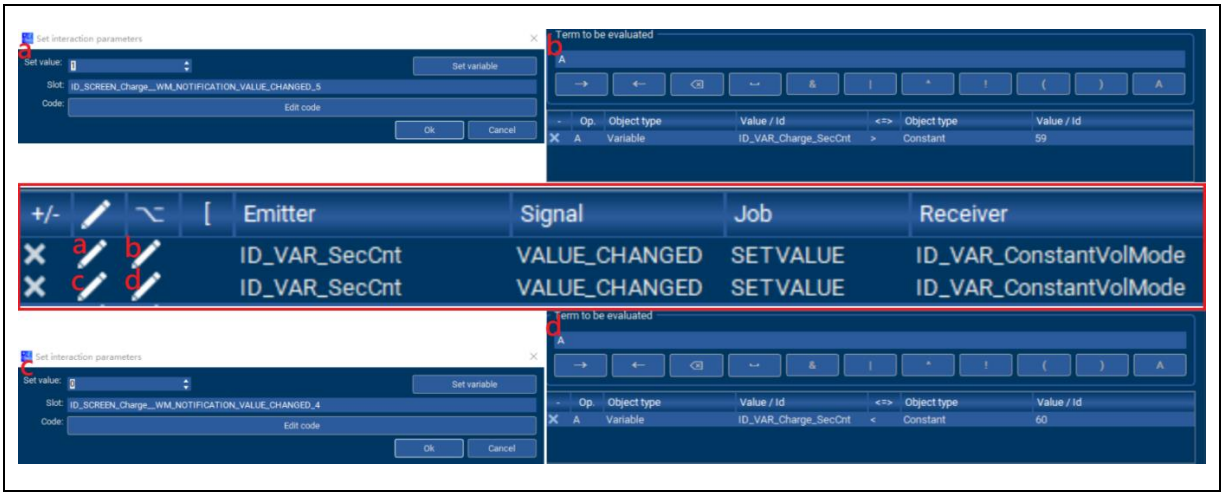


Figure 1-59 Charging Mode Updated

7) Define the current price variable("CurPrice").

Figure 1-60 Define the Current Price Variable

8) Current prices updated. To facilitate the calculation of electric capacities and costs, assume that 1 second is equivalent to 4 minutes.

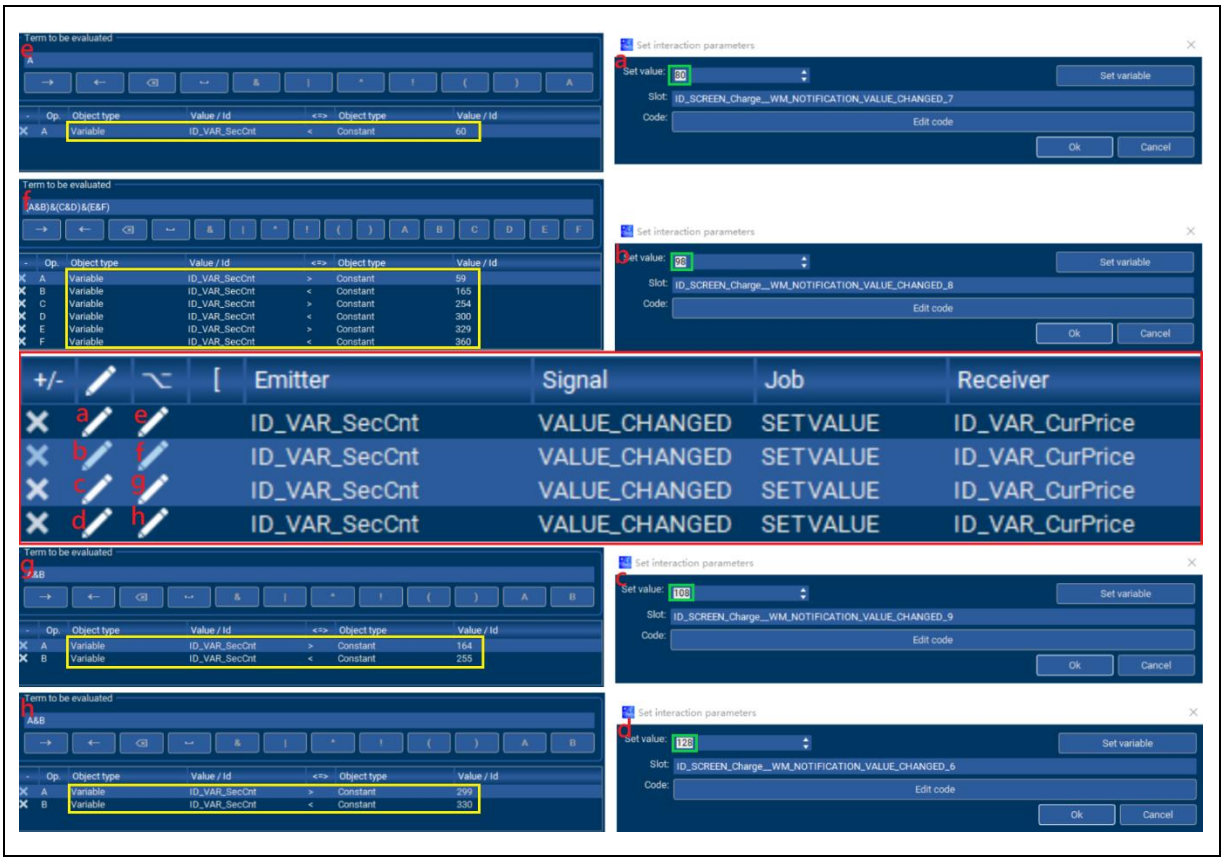


Figure 1-61 Current Prices Updated

9) Define the current and the voltage variables ("Cur", "Voltage"). The initial values are 30A and 281V respectively. In constant current mode, the voltage increases by 1.16V every 4 minutes. In constant voltage mode, the current decreases by 1.8A every 4 minutes.

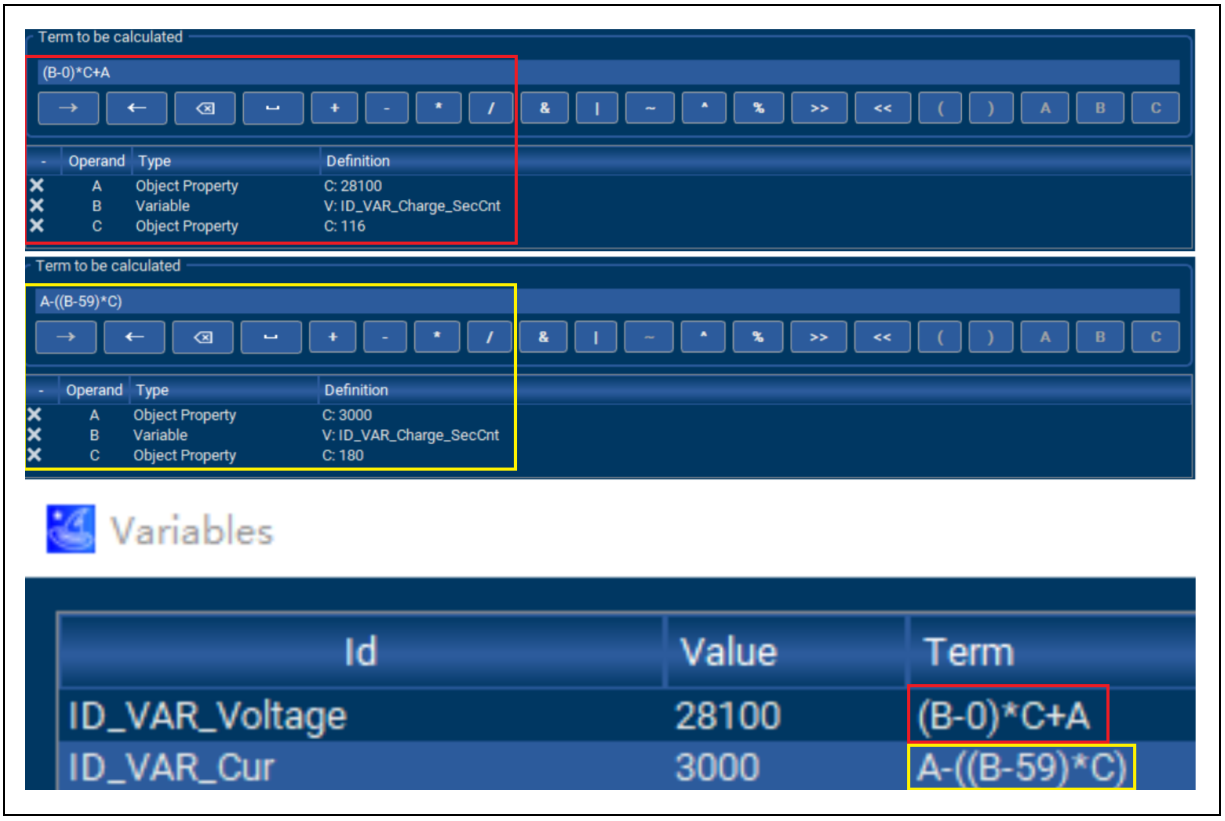


Figure 1-62 Define the Current and the Voltage Variables

10) Display the current and the voltage.

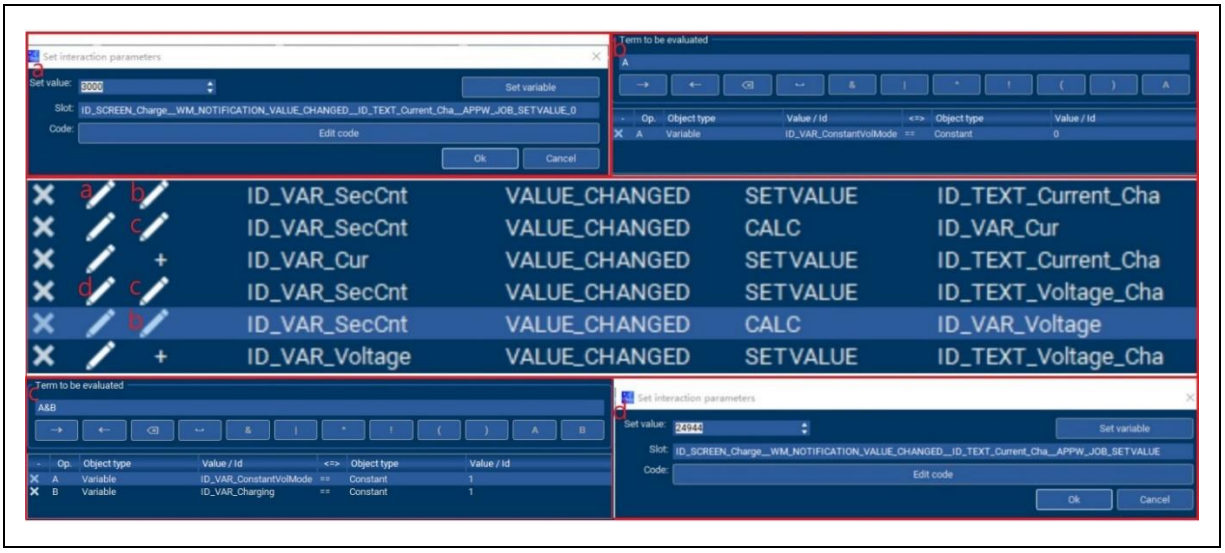


Figure 1-63 Display the Current and the Voltage

11) Define real-time electric capacity marked by the "RealCapacity", real-time consumption marked by the "RealCost", cumulative electric capacity marked by the "SumCapacity" and cumulative consumption marked by the "SumCost".

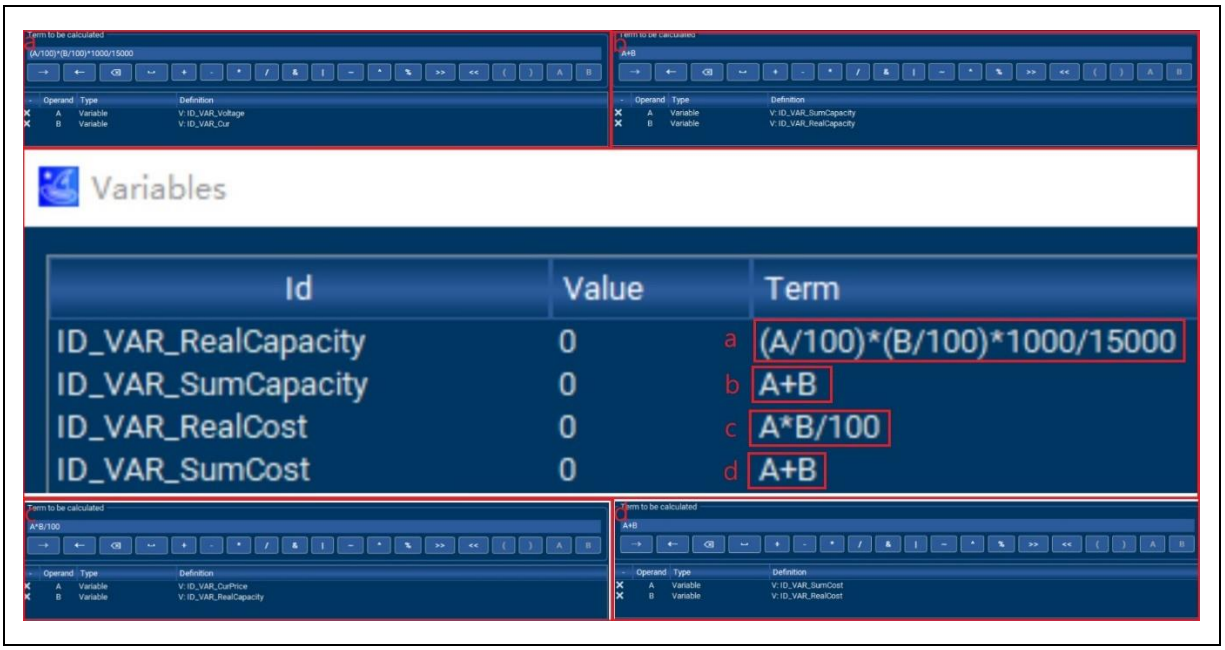


Figure 1-64 Define Variables("RealCapacity", "RealCost", "SumCapacity", "SumCost")

12) Calculate real-time electric capacity, real-time consumption, cumulative electric capacity and cumulative consumption, then display cumulative electric capacity and cumulative consumption.

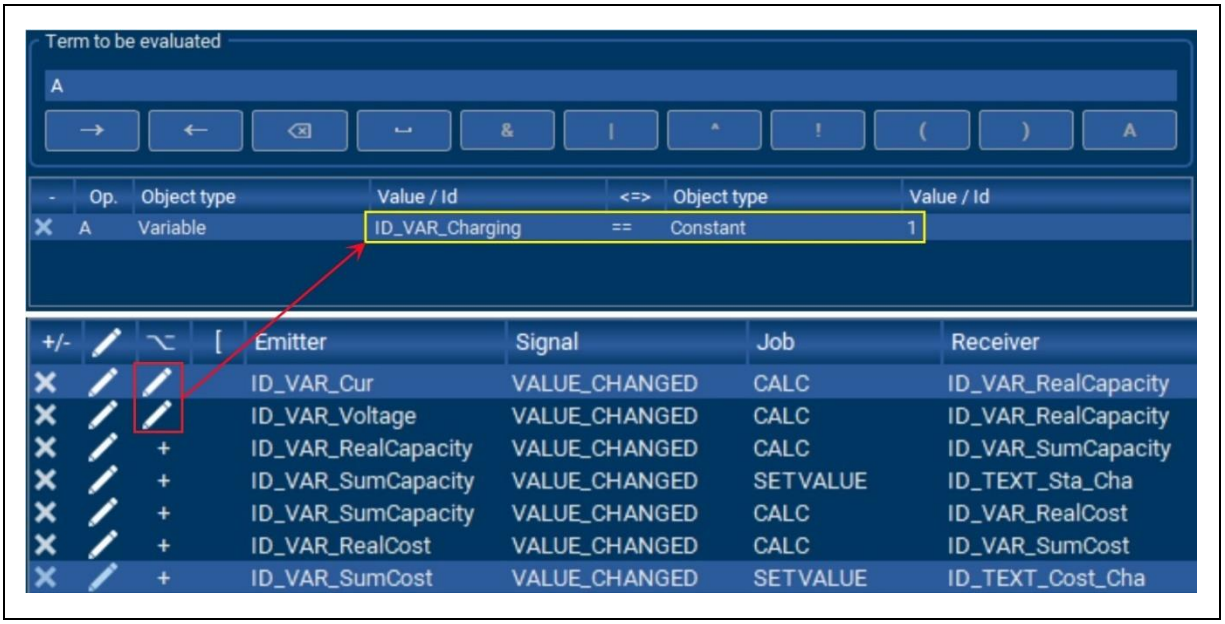


Figure 1-65 Calculate and Display Electric Capacity and Consumption

13) Stop charging, including stopping charging when full and clicking the stop button to stop charging.

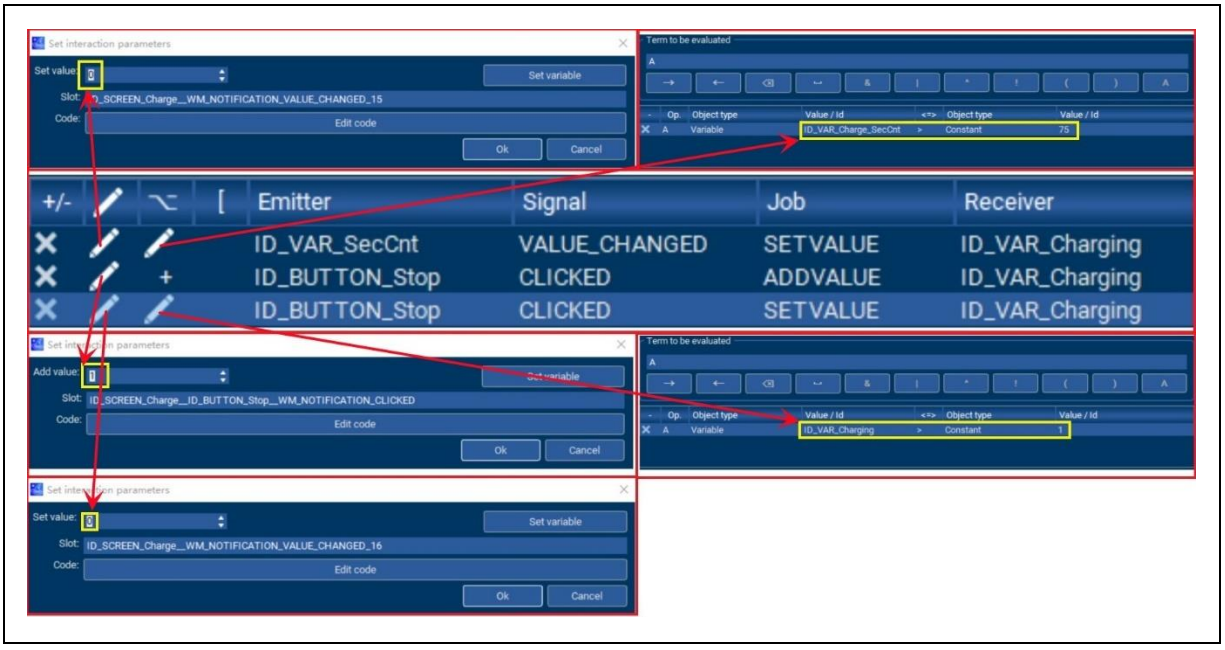


Figure 1-66 Stop Charging

14) Start and stop charging related actions.

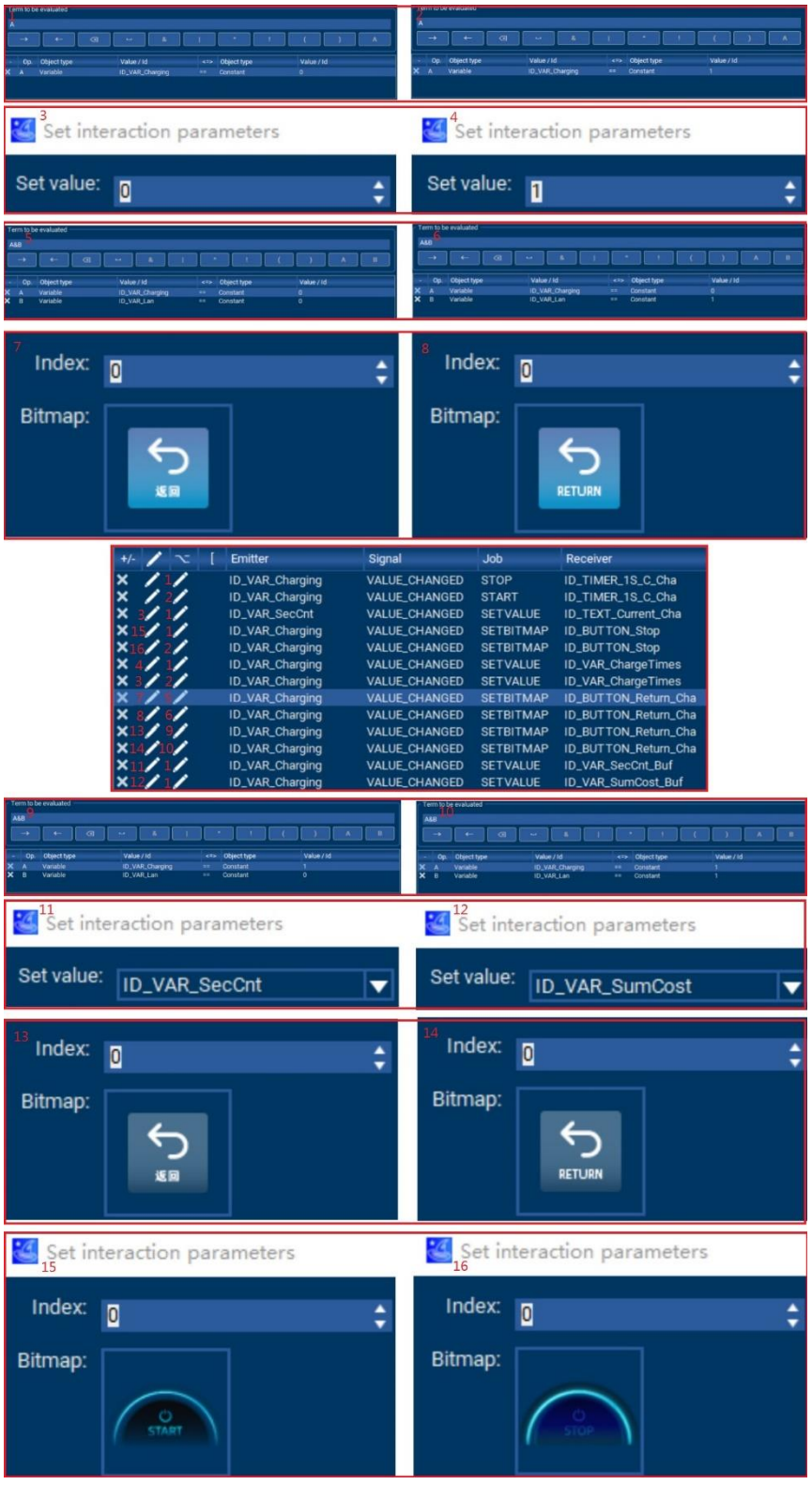


Figure 1-67 Start and Stop Charging Related Actions

15) Monitoring-window is displayed when loading charge screen by default and then enters the charging state.

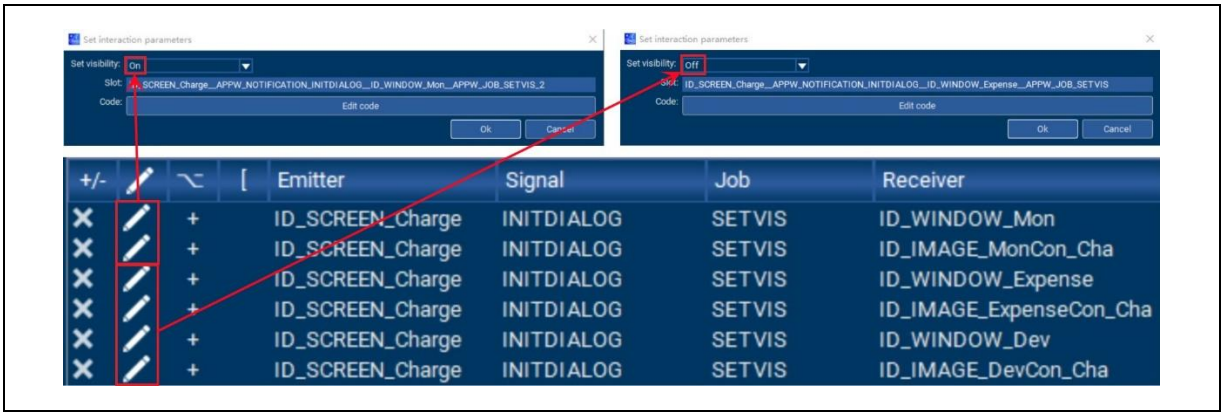


Figure 1-68 Monitoring-window Display when Loading Charge Screen

16) When the "Mon" button is clicked, the monitoring-window information is displayed, but the consumer-window and device-window are not visible.

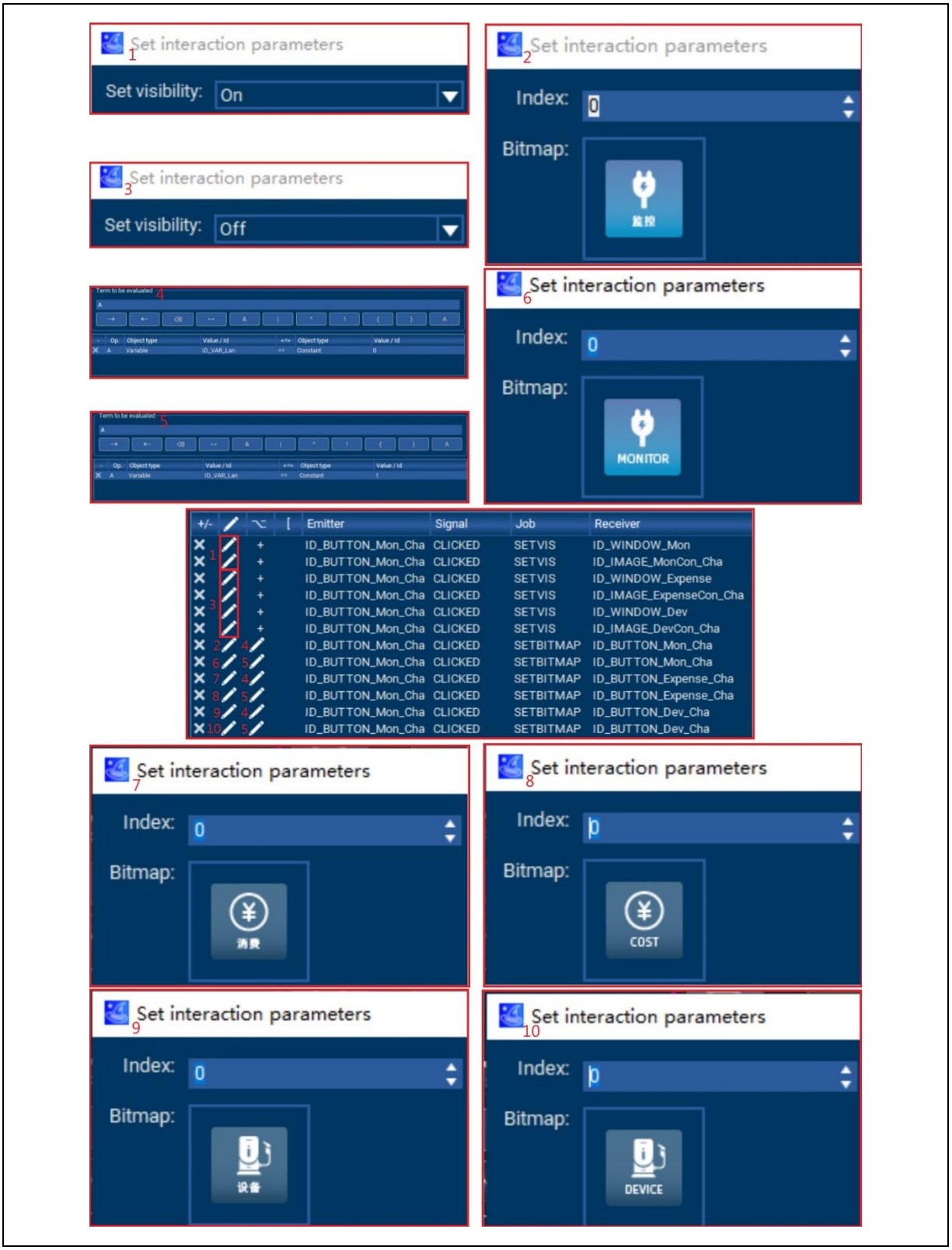
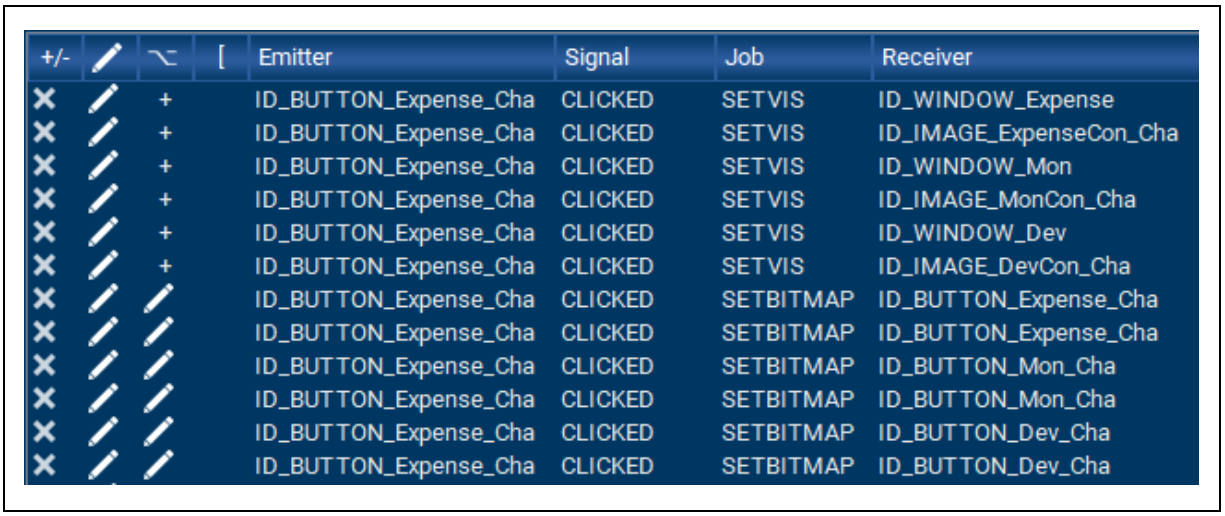


Figure 1-69 The "Mon" Button Clicked

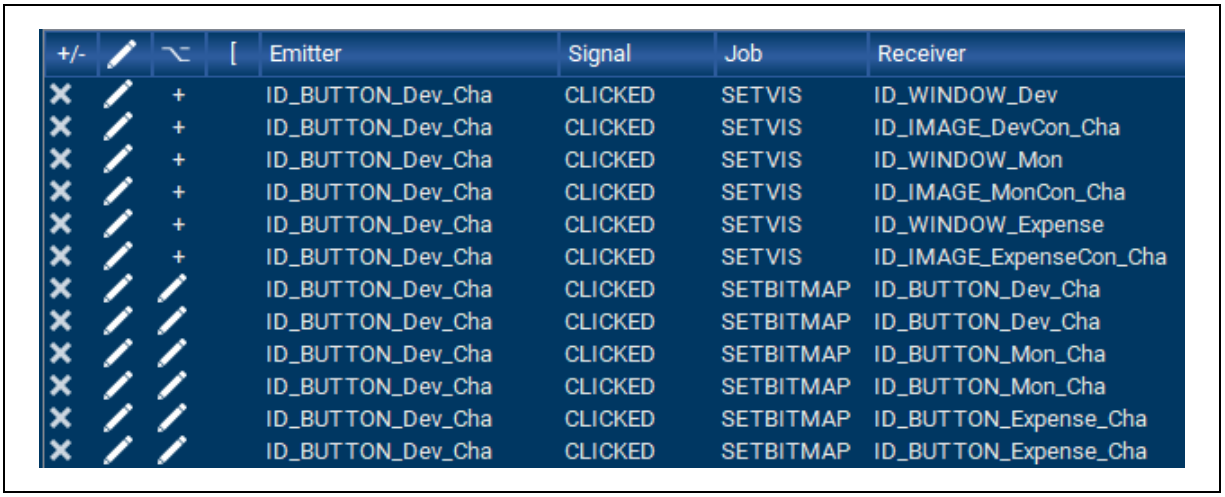
17) The "Expense" button is clicked to display the information of the consumption-window, but the monitoring-window and device-windows are not visible. Refer to Step 3 (16) in Section 1.2.2.



+/-		⌥	[	Emitter	Signal	Job	Receiver
		+		ID_BUTTON_Expense_Cha	CLICKED	SETVIS	ID_WINDOW_Expense
		+		ID_BUTTON_Expense_Cha	CLICKED	SETVIS	ID_IMAGE_ExpenseCon_Cha
		+		ID_BUTTON_Expense_Cha	CLICKED	SETVIS	ID_WINDOW_Mon
		+		ID_BUTTON_Expense_Cha	CLICKED	SETVIS	ID_IMAGE_MonCon_Cha
		+		ID_BUTTON_Expense_Cha	CLICKED	SETVIS	ID_WINDOW_Dev
		+		ID_BUTTON_Expense_Cha	CLICKED	SETVIS	ID_IMAGE_DevCon_Cha
				ID_BUTTON_Expense_Cha	CLICKED	SETBITMAP	ID_BUTTON_Expense_Cha
				ID_BUTTON_Expense_Cha	CLICKED	SETBITMAP	ID_BUTTON_Expense_Cha
				ID_BUTTON_Expense_Cha	CLICKED	SETBITMAP	ID_BUTTON_Mon_Cha
				ID_BUTTON_Expense_Cha	CLICKED	SETBITMAP	ID_BUTTON_Mon_Cha
				ID_BUTTON_Expense_Cha	CLICKED	SETBITMAP	ID_BUTTON_Dev_Cha
				ID_BUTTON_Expense_Cha	CLICKED	SETBITMAP	ID_BUTTON_Dev_Cha

Figure 1-70 The "Expense" Button Clicked

18) The "Dev" button is clicked to display the device-window information, but the consumer-window and the monitoring-window are not visible. Refer to Step 3 (16) in Section 1.2.2.



+/-		⌥	[	Emitter	Signal	Job	Receiver
		+		ID_BUTTON_Dev_Cha	CLICKED	SETVIS	ID_WINDOW_Dev
		+		ID_BUTTON_Dev_Cha	CLICKED	SETVIS	ID_IMAGE_DevCon_Cha
		+		ID_BUTTON_Dev_Cha	CLICKED	SETVIS	ID_WINDOW_Mon
		+		ID_BUTTON_Dev_Cha	CLICKED	SETVIS	ID_IMAGE_MonCon_Cha
		+		ID_BUTTON_Dev_Cha	CLICKED	SETVIS	ID_WINDOW_Expense
		+		ID_BUTTON_Dev_Cha	CLICKED	SETVIS	ID_IMAGE_ExpenseCon_Cha
				ID_BUTTON_Dev_Cha	CLICKED	SETBITMAP	ID_BUTTON_Dev_Cha
				ID_BUTTON_Dev_Cha	CLICKED	SETBITMAP	ID_BUTTON_Dev_Cha
				ID_BUTTON_Dev_Cha	CLICKED	SETBITMAP	ID_BUTTON_Mon_Cha
				ID_BUTTON_Dev_Cha	CLICKED	SETBITMAP	ID_BUTTON_Mon_Cha
				ID_BUTTON_Dev_Cha	CLICKED	SETBITMAP	ID_BUTTON_Expense_Cha
				ID_BUTTON_Dev_Cha	CLICKED	SETBITMAP	ID_BUTTON_Expense_Cha

Figure 1-71 The "Dev" Button Clicked

19) Define the charge percentage variable("ChargePer").

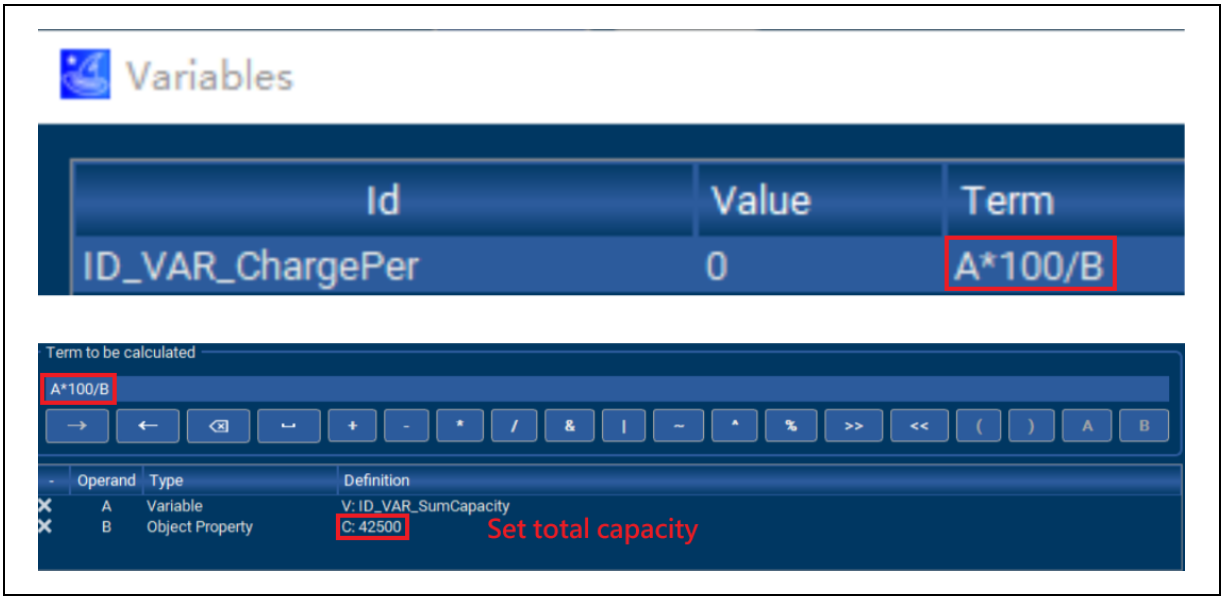


Figure 1-72 Define the Charge Percentage Variable

20) Display charging status. 0%~19% corresponds to 1 grid, 20%~39% corresponds to 2 grids, 40%~59% corresponds to 3 grid, 60%~79% corresponds to 4 grid, 80%~100% corresponds to 5 grid.

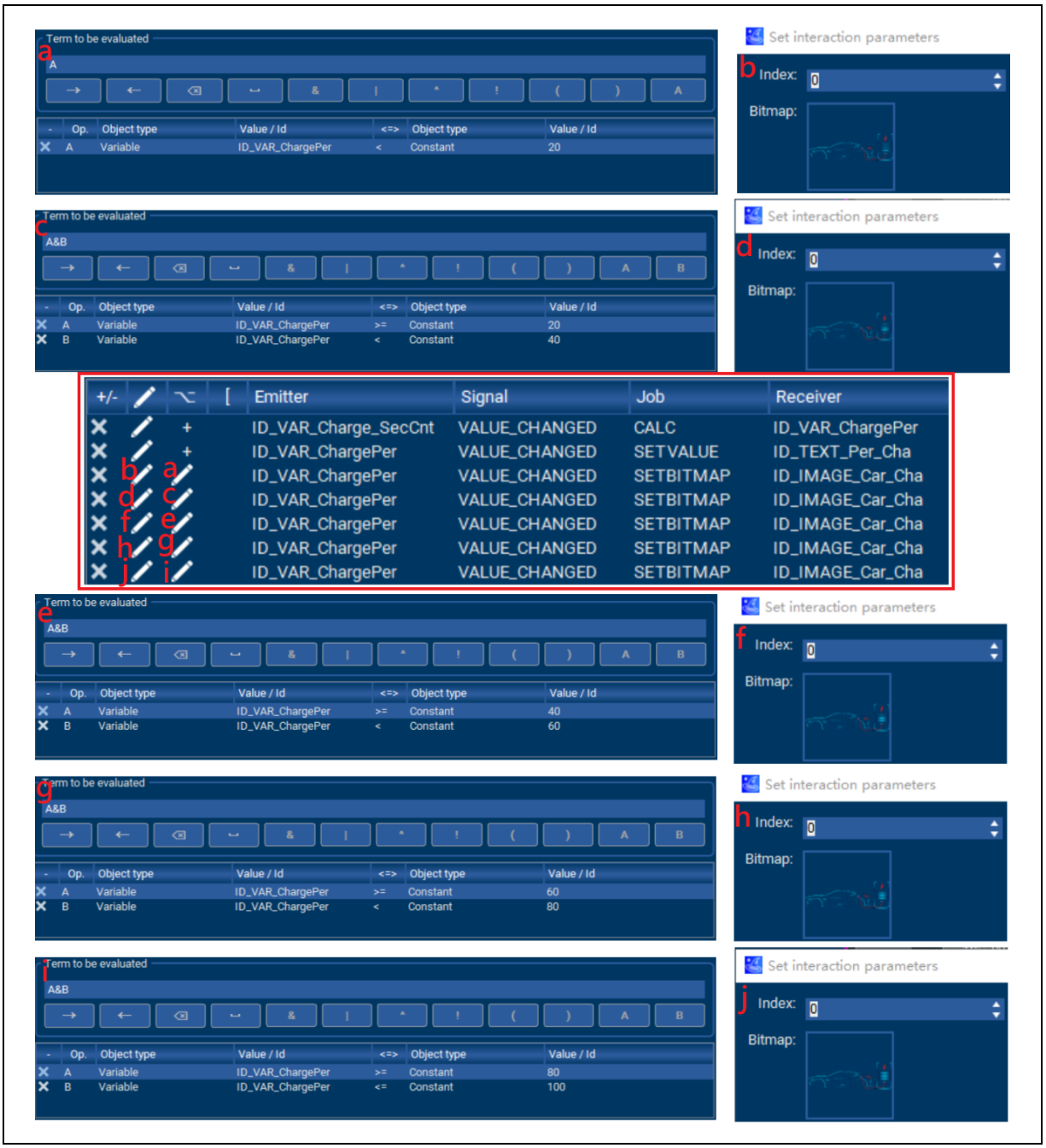


Figure 1-73 Display Charging Status

21) Define the variables of accumulated electric capacity and accumulated cost in each period of the sharp, peak, flat and valley.

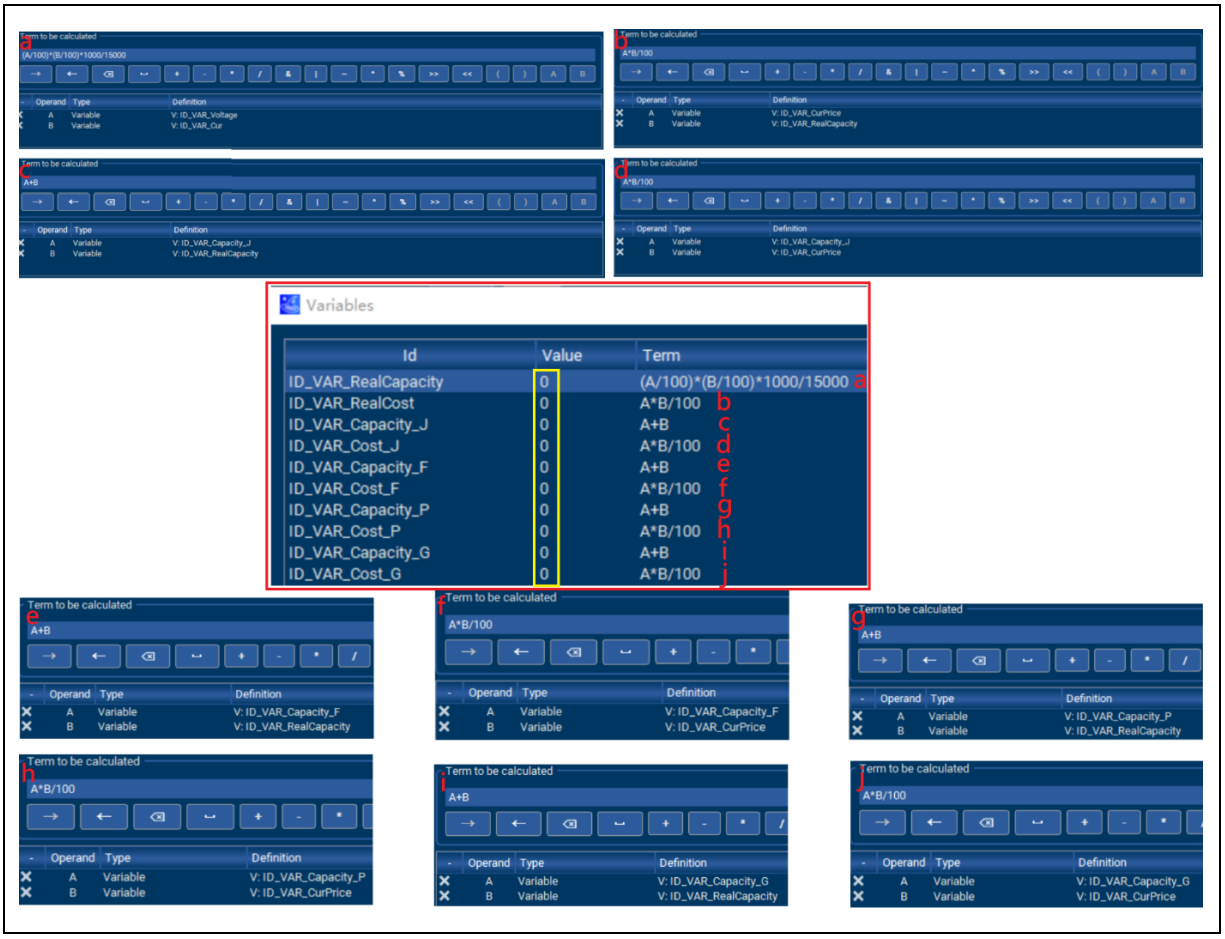


Figure 1-74 Electric Capacity and Cost Variables Definition

22) Display the cumulative electric capacities and cumulative costs in each period of the sharp, peak, flat and valley.

Figure 1-75 Display the Electric Capacity and Cost in Each Period of a Day

23) Defined power variable.

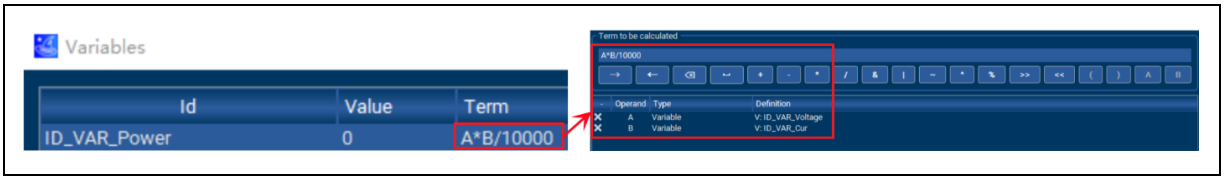


Figure 1-76 Defined Power Variable

24) Power display.

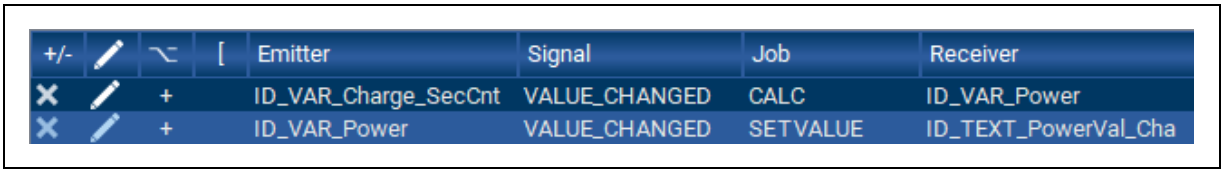


Figure 1-77 Power Display

25) Current price display.

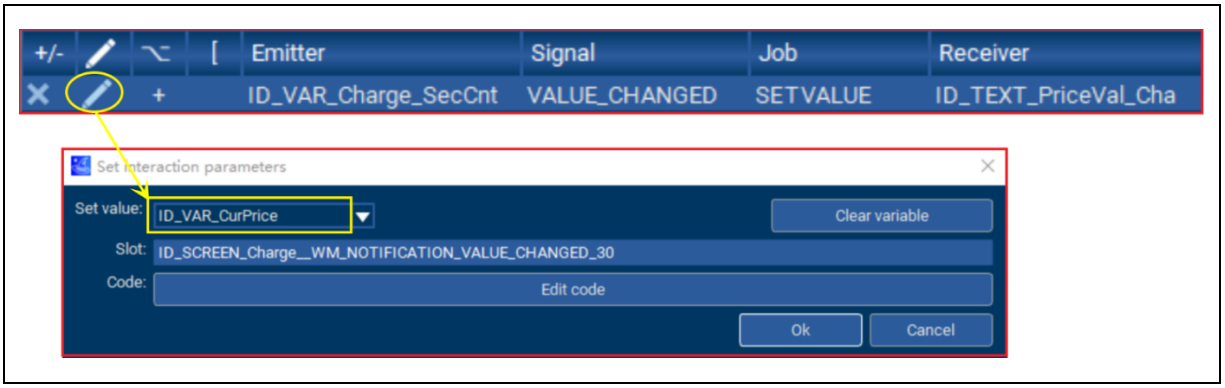


Figure 1-78 Current Price Display

26) Load the "Charge" screen and update the patterns of the "Mon", the "Expense" and the "Dev" according to the value of the "Lan" variable.

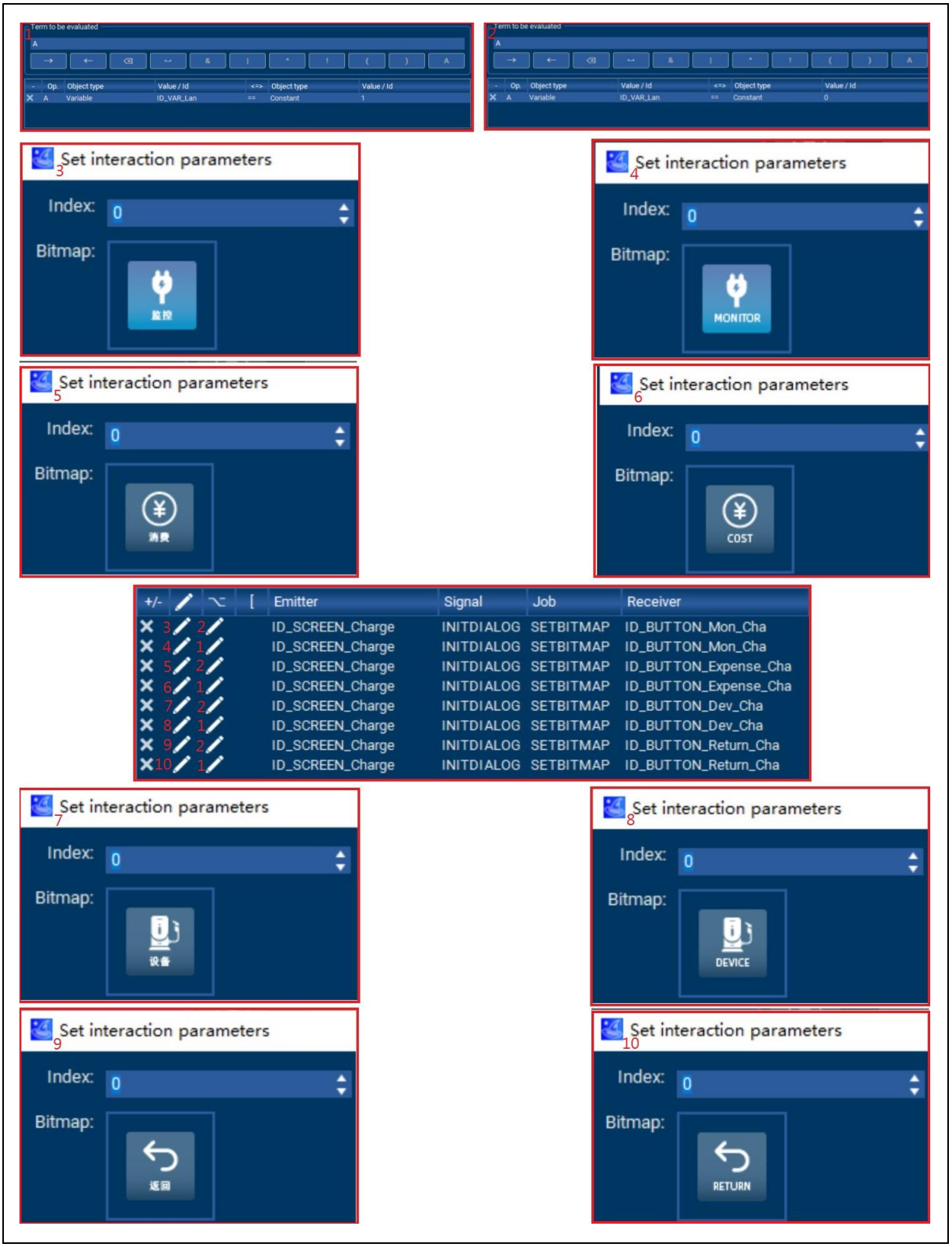


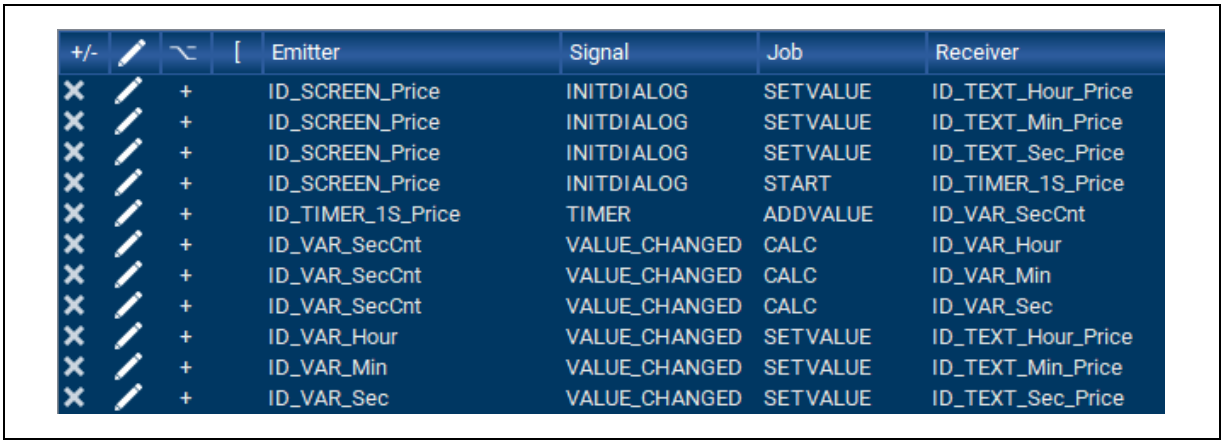
Figure 1-79 Update the Patterns of the "Mon", "Expense" and "Dev"

4. Price screen interaction design. Click the "Return" button to return to the "Init" screen, including the real-time time display and the price of each time period display.

   1) Click the "Return" button to return to the "Init" screen. Refer to Step 1 (1) in Section 1.2.3.

Figure 1-80 Click the "Return" Button

   2) Real-time time display. Refer to Step 2 (2) in Section 1.2.3.



+/-		Emitter	Signal	Job	Receiver
	+	ID_SCREEN_Price	INITDIALOG	SETVALUE	ID_TEXT_Hour_Price
	+	ID_SCREEN_Price	INITDIALOG	SETVALUE	ID_TEXT_Min_Price
	+	ID_SCREEN_Price	INITDIALOG	SETVALUE	ID_TEXT_Sec_Price
	+	ID_SCREEN_Price	INITDIALOG	START	ID_TIMER_1S_Price
	+	ID_TIMER_1S_Price	TIMER	ADDVALUE	ID_VAR_SecCnt
	+	ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Hour
	+	ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Min
	+	ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Sec
	+	ID_VAR_Hour	VALUE_CHANGED	SETVALUE	ID_TEXT_Hour_Price
	+	ID_VAR_Min	VALUE_CHANGED	SETVALUE	ID_TEXT_Min_Price
	+	ID_VAR_Sec	VALUE_CHANGED	SETVALUE	ID_TEXT_Sec_Price

Figure 1-81 Real-time Time Display

   3) Define price variables of each period of the sharp, peak, flat and valley.

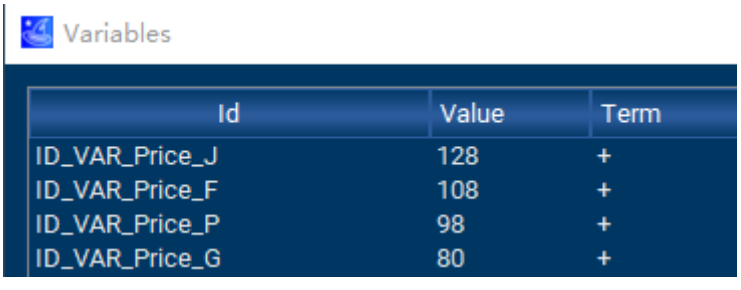


Figure 1-82 Define Price Variable

4) Display prices for each time period.

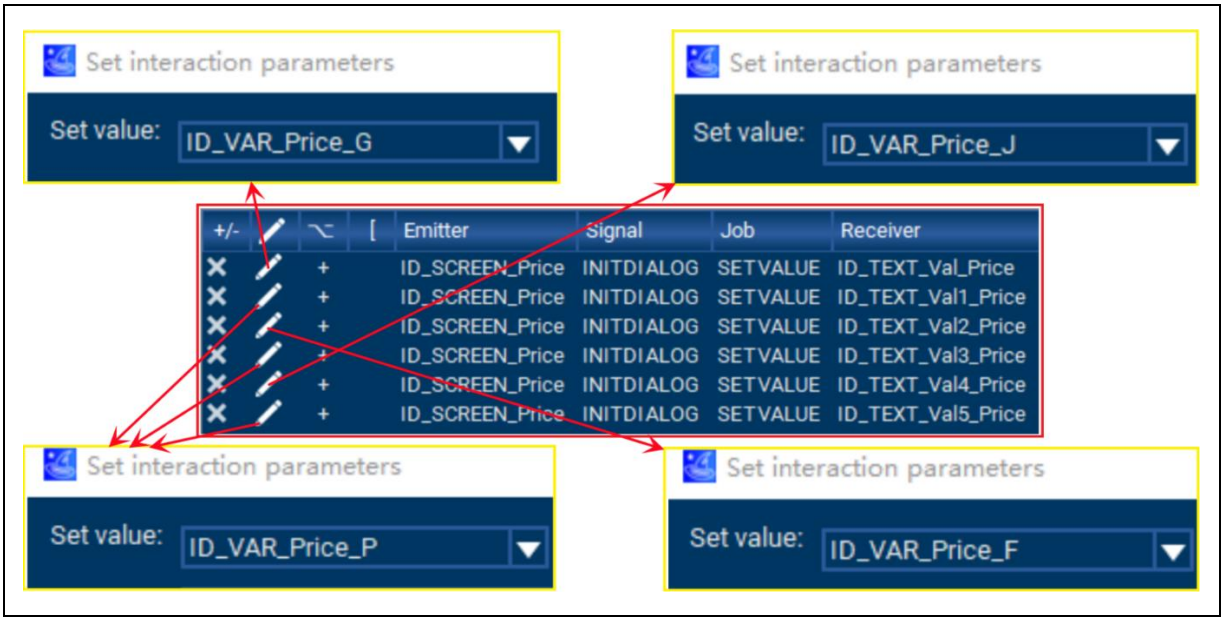


Figure 1-83 Show the Price

5) Load the "Price" screen and update the pattern of the "Return" according to the value of the "Lan" variable. Refer to Step 1 (3) in Section 1.2.3.

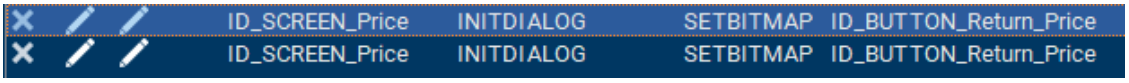


Figure 1-84 Update the Pattern of the "Return"

5. Info screen interaction design. This includes clicking the "Return" button to return to the "Init" screen, displaying real-time time and the name of the charging station and the type of charging and the number of the available charging gun.

1) Click the "Return" button to return to the "Init" screen. Refer to Step 1 (1) in Section 1.2.3.

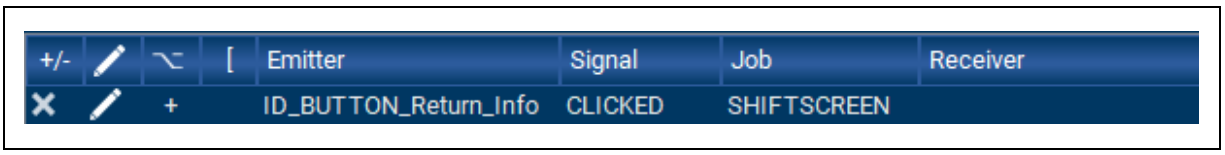


Figure 1-85 Click the "Return" Button

2) Real-time time display. Refer to Step 2 (2) in Section 1.2.3.

+/-		⌥	[	Emitter	Signal	Job	Receiver
✕		+		ID_SCREEN_Info	INITDIALOG	SETVALUE	ID_TEXT_Hour_Info
✕		+		ID_SCREEN_Info	INITDIALOG	SETVALUE	ID_TEXT_Min_Info
✕		+		ID_SCREEN_Info	INITDIALOG	SETVALUE	ID_TEXT_Sec_Info
✕		+		ID_SCREEN_Info	INITDIALOG	START	ID_TIMER_1S_Info
✕		+		ID_TIMER_1S_Info	TIMER	ADDVALUE	ID_VAR_SecCnt
✕		+		ID_VAR_SecCnt	VALUE_CH...	CALC	ID_VAR_Hour
✕		+		ID_VAR_SecCnt	VALUE_CH...	CALC	ID_VAR_Min
✕		+		ID_VAR_SecCnt	VALUE_CH...	CALC	ID_VAR_Sec
✕		+		ID_VAR_Hour	VALUE_CH...	SETVALUE	ID_TEXT_Hour_Info
✕		+		ID_VAR_Min	VALUE_CH...	SETVALUE	ID_TEXT_Min_Info
✕		+		ID_VAR_Sec	VALUE_CH...	SETVALUE	ID_TEXT_Sec_Info

Figure 1-86 Real-time Time Display

3) Gun number initialization.

Figure 1-87 Gun Number Initialization

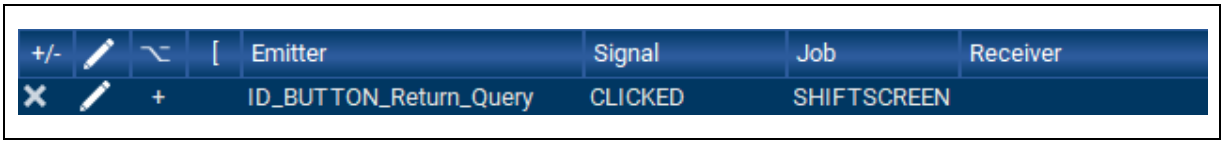
4) Load the "Info" screen and update the pattern of the "Return" according to the value of the "Lan" variable. Refer to Step 1 (3) in Section 1.2.3.

+/-		⌥	[	Emitter	Signal	Job	Receiver
✕				ID_SCREEN_Info	INITDIALOG	SETBITMAP	ID_BUTTON_Return_Info
✕				ID_SCREEN_Info	INITDIALOG	SETBITMAP	ID_BUTTON_Return_Info

Figure 1-88 Update the Pattern of the "Return"

6. Query screen interaction design. This includes clicking the "Return" button to return to the "Init" screen and displaying real-time time and the username in conjunction with the code. The latest charging information is not displayed until the charging is completed.

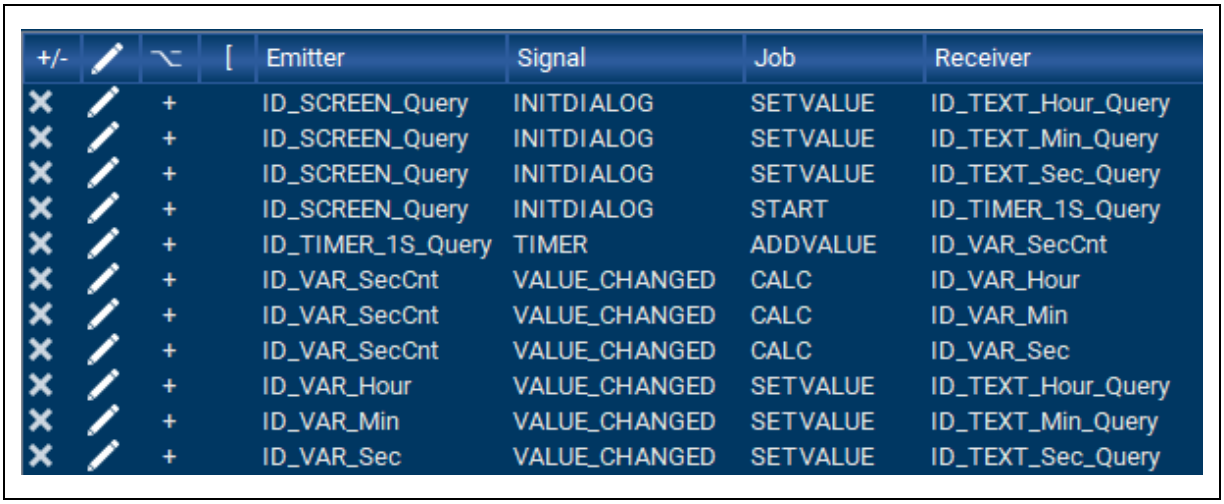
   1) Click the "Return" button to return to the "Init" screen. Refer to Step 1 (1) in Section 1.2.3.



+/-		⌥	[	Emitter	Signal	Job	Receiver
✕	✎	+		ID_BUTTON_Return_Query	CLICKED	SHIFTSCREEN	

Figure 1-89 Click the "Return" Button

   2) Real-time time display. Refer to Step 2 (2) in Section 1.2.3.



+/-		⌥	[	Emitter	Signal	Job	Receiver
✕	✎	+		ID_SCREEN_Query	INITDIALOG	SETVALUE	ID_TEXT_Hour_Query
✕	✎	+		ID_SCREEN_Query	INITDIALOG	SETVALUE	ID_TEXT_Min_Query
✕	✎	+		ID_SCREEN_Query	INITDIALOG	SETVALUE	ID_TEXT_Sec_Query
✕	✎	+		ID_SCREEN_Query	INITDIALOG	START	ID_TIMER_1S_Query
✕	✎	+		ID_TIMER_1S_Query	TIMER	ADDVALUE	ID_VAR_SecCnt
✕	✎	+		ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Hour
✕	✎	+		ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Min
✕	✎	+		ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Sec
✕	✎	+		ID_VAR_Hour	VALUE_CHANGED	SETVALUE	ID_TEXT_Hour_Query
✕	✎	+		ID_VAR_Min	VALUE_CHANGED	SETVALUE	ID_TEXT_Min_Query
✕	✎	+		ID_VAR_Sec	VALUE_CHANGED	SETVALUE	ID_TEXT_Sec_Query

Figure 1-90 Real-time Time Display

   3) Set window display when loading the screen. The window is displayed only after the charging is complete.

Figure 1-91 Set Window Display when Loading the Screen

4) Define the variables of the hour, the minute, the second, the "SecCnt" backup and "SumCost" backup.

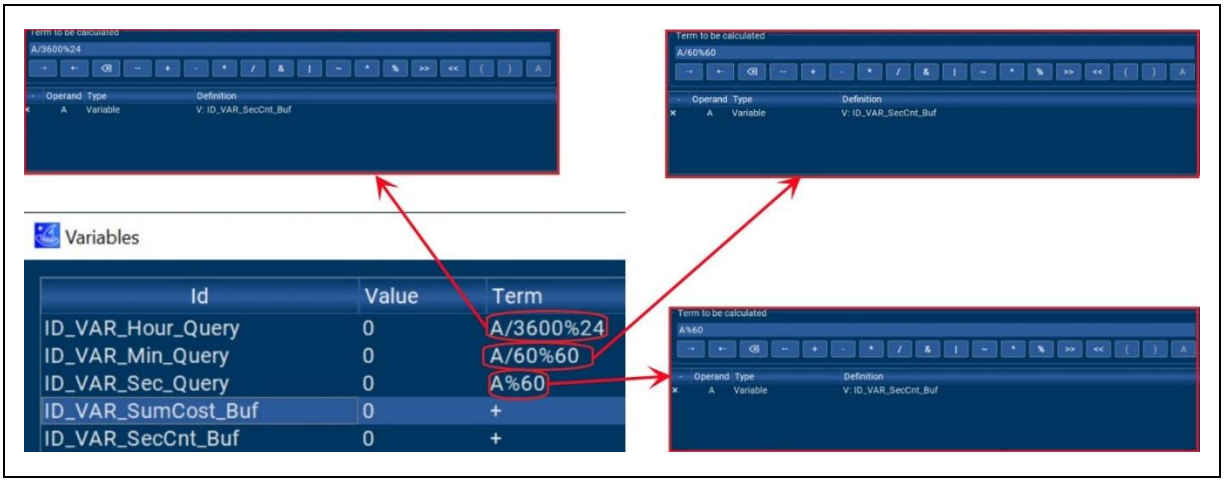


Figure 1-92 Variable Definition

5) Display information about the last charge, including the time, the cost, the charging station and gun number.

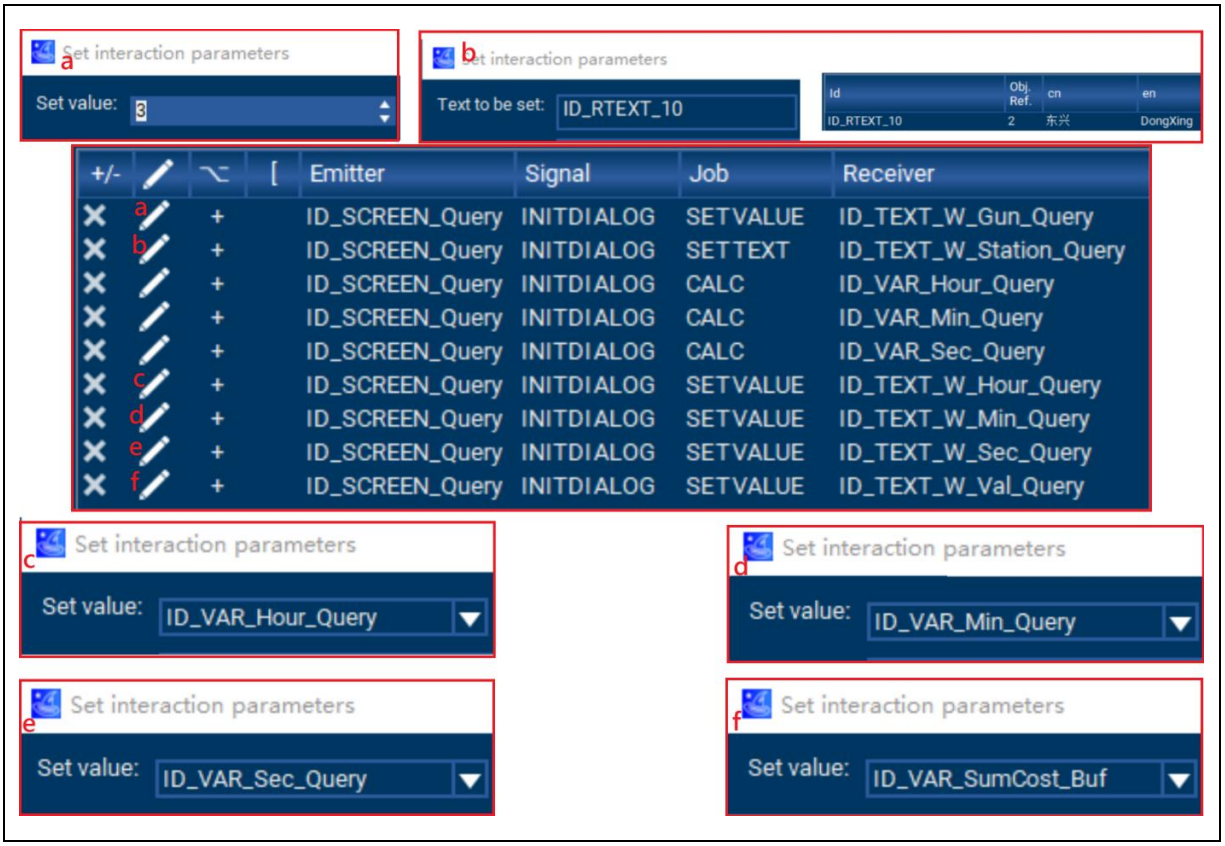
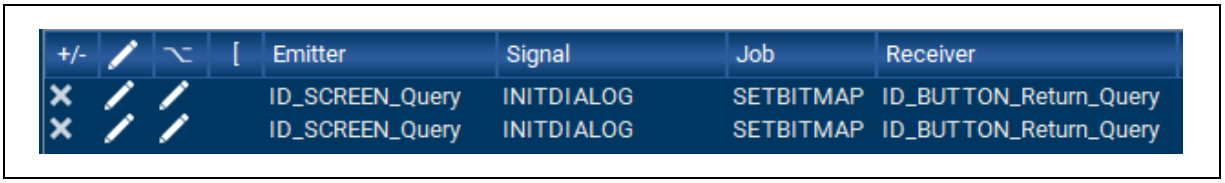


Figure 1-93 Display Information about the Last Charging

6) Load the "Query" screen and update the pattern of the "Return" according to the value of the "Lan" variable. Refer to Step 1 (3) in Section 1.2.3.

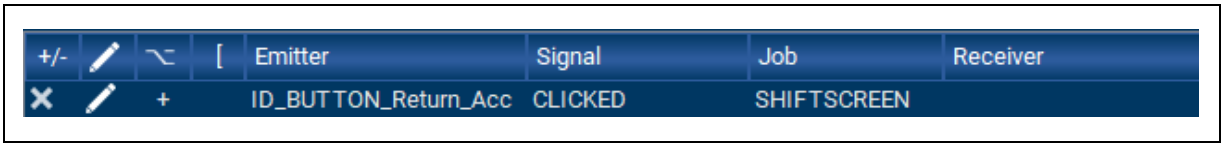


+/-		⌥	[	Emitter	Signal	Job	Receiver
✕				ID_SCREEN_Query	INITDIALOG	SETBITMAP	ID_BUTTON_Return_Query
✕				ID_SCREEN_Query	INITDIALOG	SETBITMAP	ID_BUTTON_Return_Query

Figure 1-94 Update the Pattern of the "Return"

7. Account screen interaction design. This includes clicking the "Return" button to return to the "Init" screen; Displaying the username in conjunction with the code, real-time time, the name, the balance and the phone number of the account.

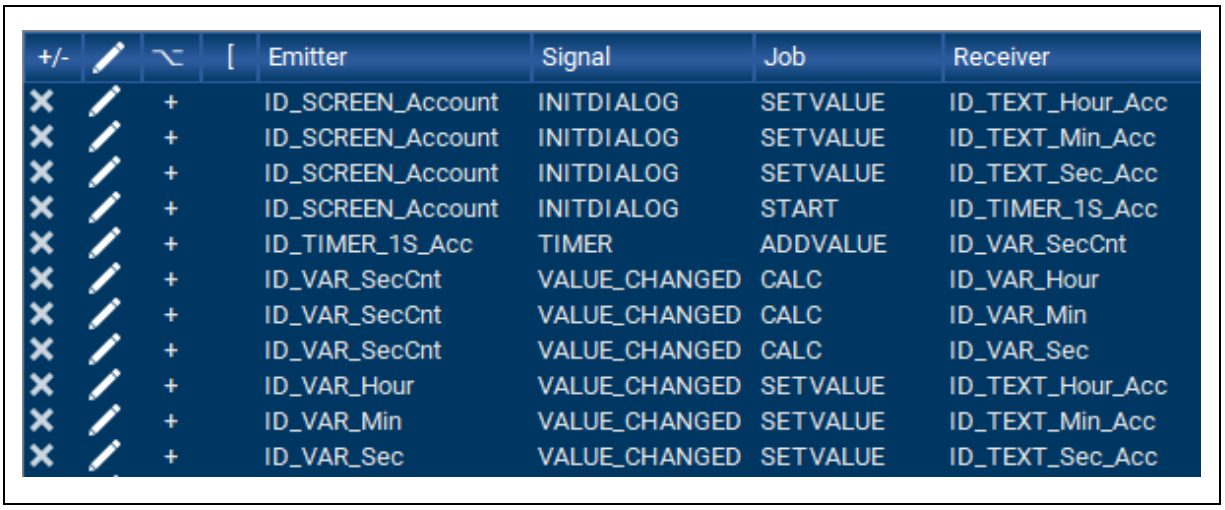
1) Click the "Return" button to return to the "Init" screen. Refer to Step 1 (1) in Section 1.2.3.



+/-		⌥	[	Emitter	Signal	Job	Receiver
✕		+		ID_BUTTON_Return_Acc	CLICKED	SHIFTSCREEN	

Figure 1-95 Click the "Return" Button

2) Real-time time display. Refer to Step 2 (2) in Section 1.2.3.



+/-		⌥	[	Emitter	Signal	Job	Receiver
✕		+		ID_SCREEN_Account	INITDIALOG	SETVALUE	ID_TEXT_Hour_Acc
✕		+		ID_SCREEN_Account	INITDIALOG	SETVALUE	ID_TEXT_Min_Acc
✕		+		ID_SCREEN_Account	INITDIALOG	SETVALUE	ID_TEXT_Sec_Acc
✕		+		ID_SCREEN_Account	INITDIALOG	START	ID_TIMER_1S_Acc
✕		+		ID_TIMER_1S_Acc	TIMER	ADDVALUE	ID_VAR_SecCnt
✕		+		ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Hour
✕		+		ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Min
✕		+		ID_VAR_SecCnt	VALUE_CHANGED	CALC	ID_VAR_Sec
✕		+		ID_VAR_Hour	VALUE_CHANGED	SETVALUE	ID_TEXT_Hour_Acc
✕		+		ID_VAR_Min	VALUE_CHANGED	SETVALUE	ID_TEXT_Min_Acc
✕		+		ID_VAR_Sec	VALUE_CHANGED	SETVALUE	ID_TEXT_Sec_Acc

Figure 1-96 Real-time Time Display

3) Define the balance variable. The initial value is 1000.

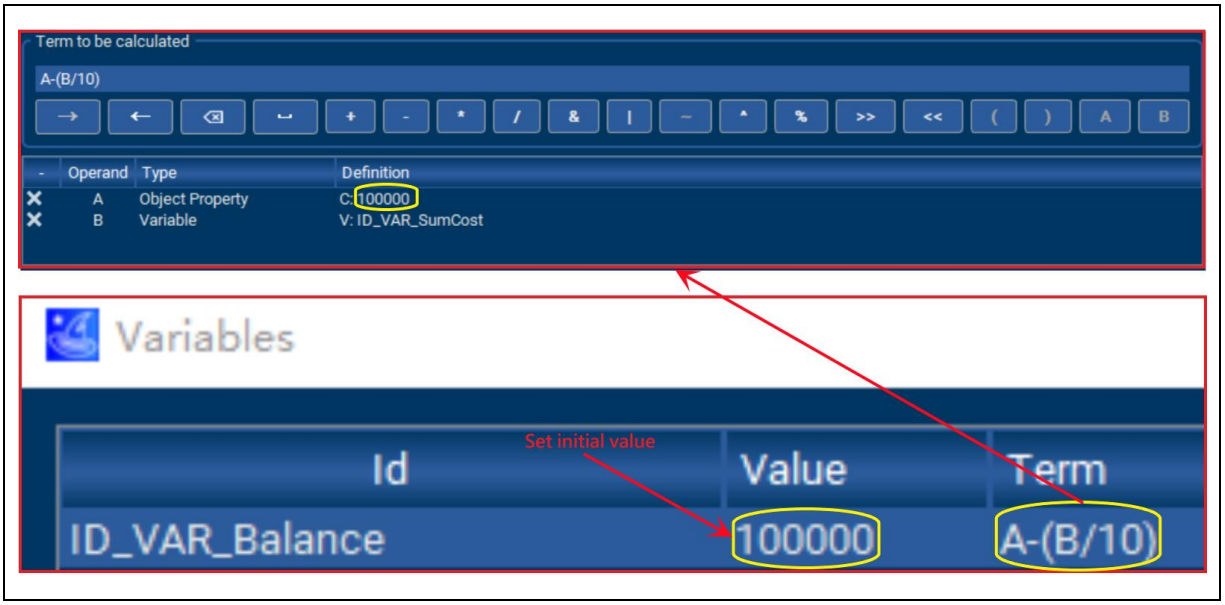


Figure 1-97 Define the Balance Variable

4) Balance display.

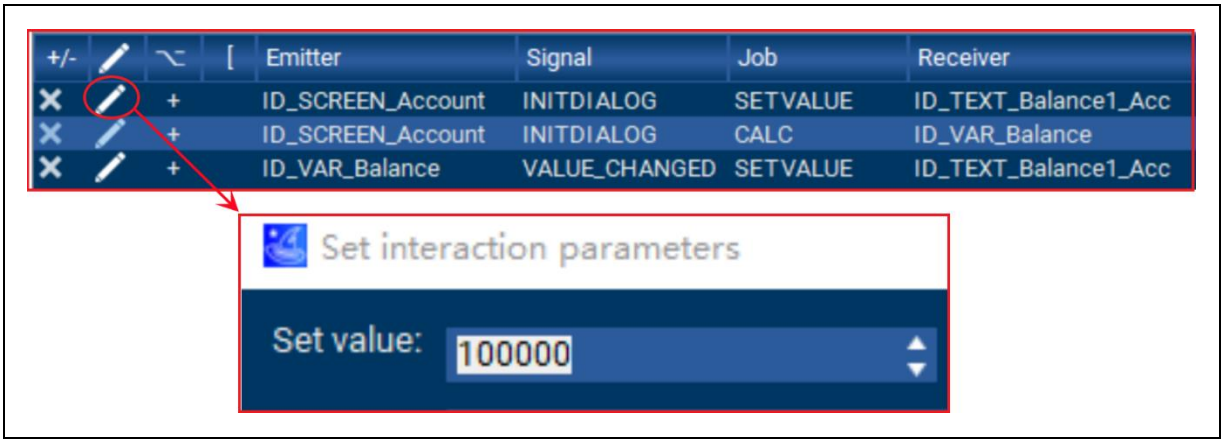


Figure 1-98 Balance Display

5) Load the "Account" screen and update the pattern of the "Return" according to the value of the "Lan" variable. Refer to Step 1 (3) in Section 1.2.3.

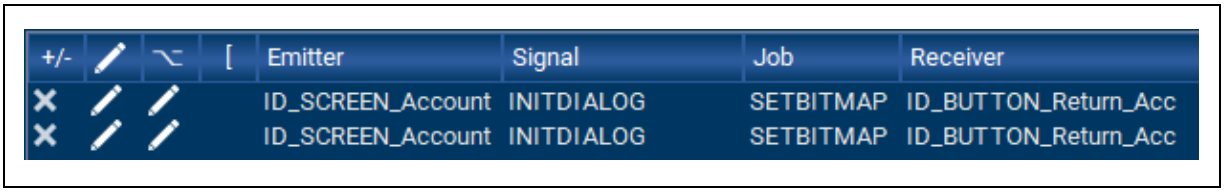


Figure 1-99 Update the Pattern of the "Return"

1.2.4 Coding

1. Once the design is completed, you can export the code by going to "File -> Export & Save", as shown in Figure 1-100. The project file used in this example is named EC_ChargePile.zip. You can refer to Figure 4-1 for the file path. After extracting the file, you can directly open the project file using AppWizard.

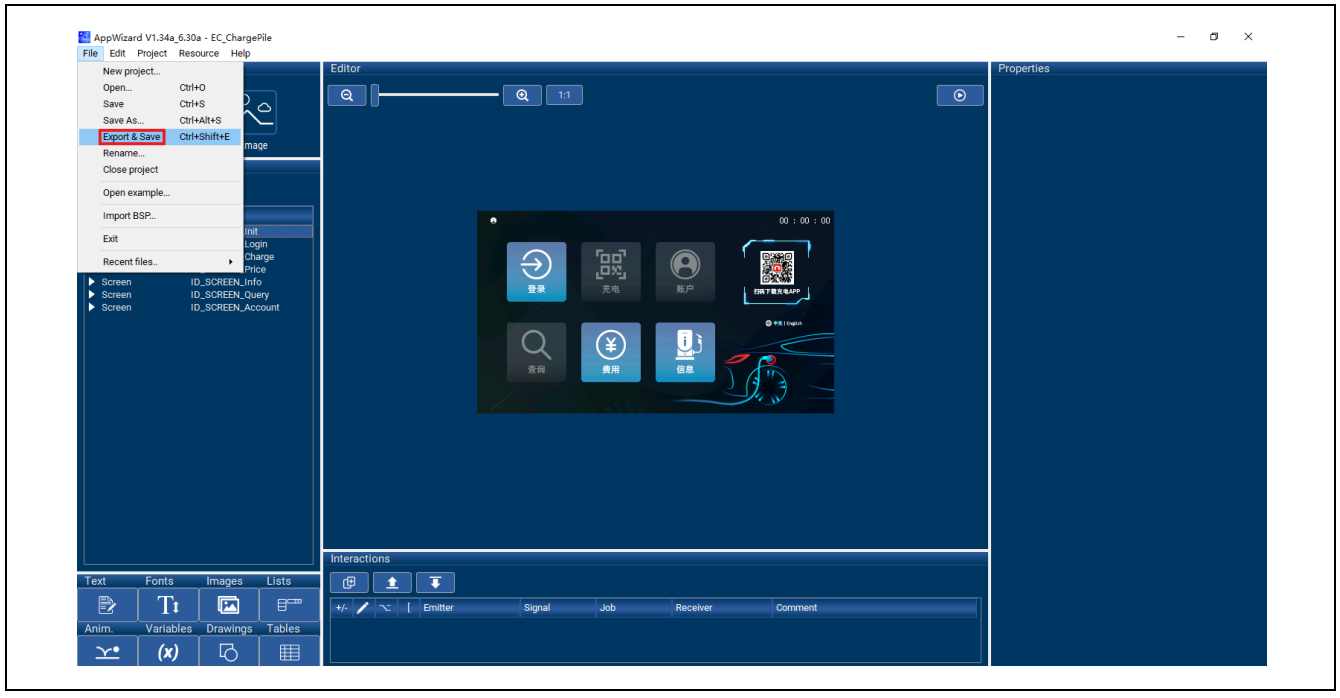


Figure 1-100 Export Code

2. Open the AppWizard project folder by going to "Project -> Open containing folder". Copy the "Resource" and "Source" folders. Navigate to the path "EC_M467_AppWizard_ChargePile _V1.00\SampleCode\ExampleCode\M467_AppWizard_ ChargePile \Application" and replace the existing files with the copied folders. The steps shown in Figure 1-101~Figure 1-102.

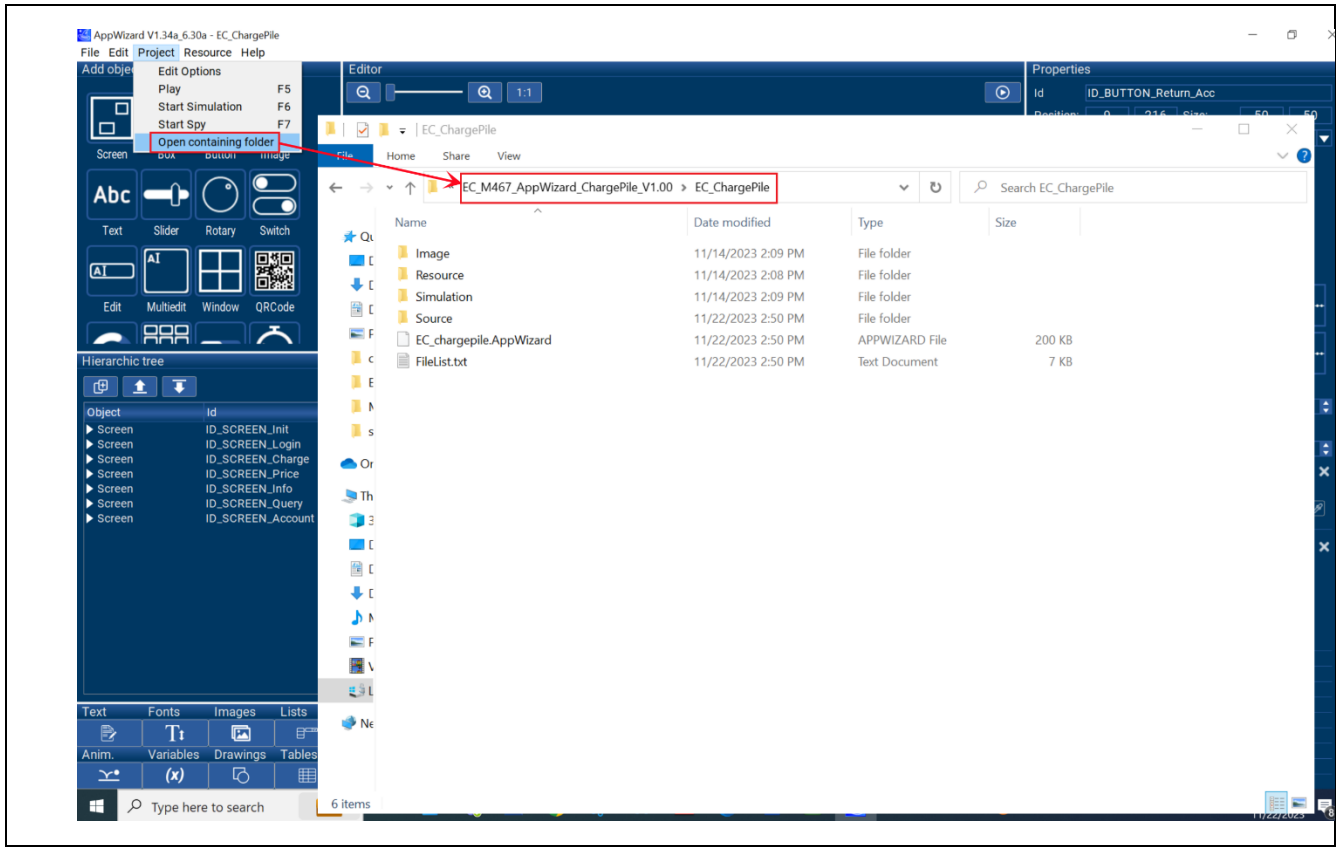


Figure 1-101 Open Project Folder

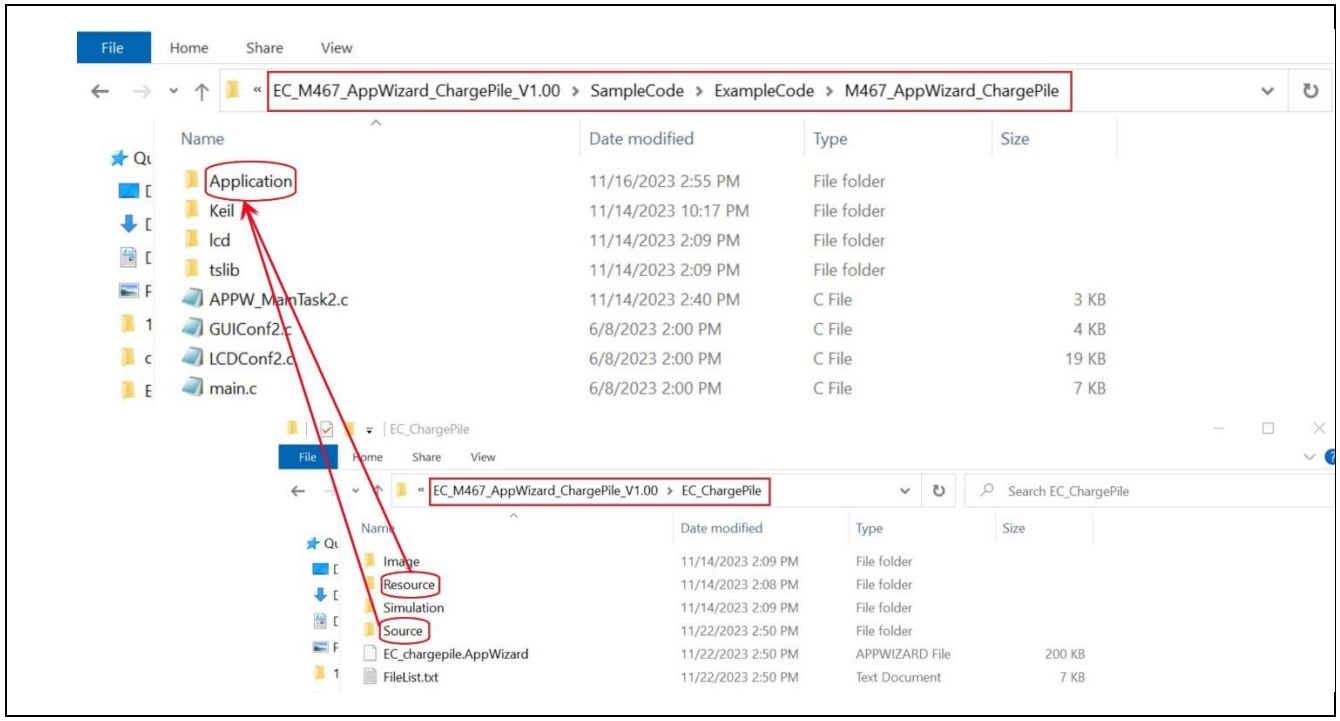


Figure 1-102 Porting to the Code Project

3. For the code steps, please refer to the code explanation in Chapter 2 code description.

1.2.5 Programming the Board

After running on the device, you can still make adjustments to optimize the overall performance. The following are the optimization settings and techniques you can apply.

1. In AppWizard, the image format used should be set to match the screen format. In this example, the setting is "High color with alpha (565)" or "Auto".

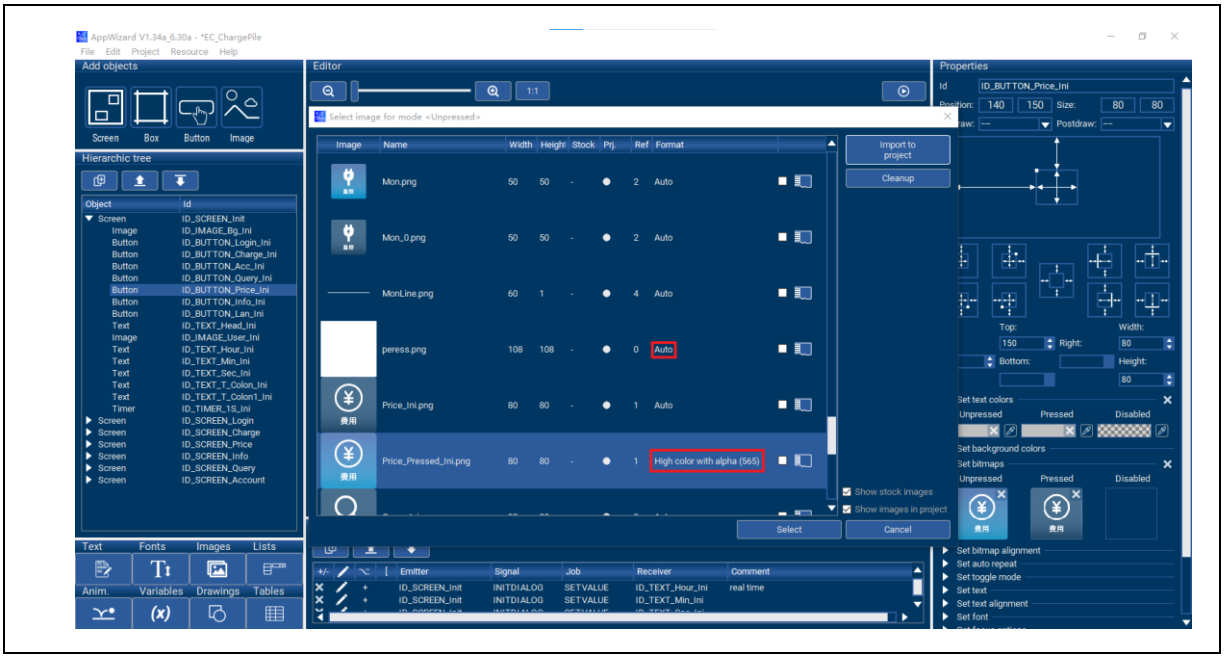


Figure 1-103 Image Setting

1.3 Demo Result

The results of compiling and flashing the code in this example are shown in Figure 1-104.

Figure 1-104 Demo Result

2. Code Description

The main program is entered through *main.c* and consists of three parts as follows:

- Clock initialization and pin definition: sys_init
- Peripheral initialization and open: UART, SPIM, Timer, Touch, etc.
- Main application function: MainTask2()

```
int main(void)
{
    // Initialize system clock
    sys_init();
    // Initialize debug port
    debug_port_init();
    SPIM_init();
    g_pu8FrameBufPtr  = (U8 *)((U32)g_u8FrameBuf);
    g_enable_Touch = 0;
    // Initialize timer
    tmr_init();
    // Startup CTP
    st1633i_low_level_init();
    st1633i_startup_sequence();

    printf("\r\n");
    printf("CPU @ %d Hz\r\n", SystemCoreClock);
    printf("+------------------------------------------------------------+\r\n");
    printf("|               emWin Simple Demo                            |\r\n");
    printf("+------------------------------------------------------------+\r\n");
    printf("\r\n");

    g_enable_Touch = 1;
    // Enable memory device before GUI_Init() to avoid flickering
    printf("Main Task2 -> \n");
    // Main Loop
    // MainTask2 for anti-flickering
    MainTask2();

    return 0;
}
```

The SPIM interface is used in order to store larger-sized image and text file resources in an external SPI Flash memory. The SPI Synchronous Serial Interface Controller (SPI Master Mode) is utilized in Direct Memory Mapping (DMM) mode, where the SPI Flash can be treated as a ROM module. This allows direct reading of the SPI Flash from specified addresses. With the assistance of the built-in cache and by quad read mode(4-wire), the overall read efficiency is improved.

The initialization code for SPIM is as follows. It includes setting the SPIM speed, SPI Flash detection, enabling SPIM cache mode, enabling SPIM DMM mode, and enabling fast quad read mode. Please refer to the documentation of W25Q32JVTBIQ for the specific parameter settings.

```
void SPIM_init(void){
    uint8_t     idBuf[3];
    SYS_UnlockReg();              /* Unlock protected registers */
    SPIM_SET_CLOCK_DIVIDER(2);    /* Set SPIM clock as HCLK divided by 2, 200MHz/2 = 100*/

    SPIM_SET_RXCLKDLY_RDDLYSEL(0);    /* Insert 0 delay cycle. Adjust the sampling clock
of received data to latch the correct data. */
    SPIM_SET_RXCLKDLY_RDEDGE();       /* Use SPI input clock rising edge to sample
received data. */
    SPIM_SET_DCNUM(4);          /* 4 is dummy cycle for command 0xEB(fast quad read).*/

    if (SPIM_InitFlash(1) != 0)        /* Initialized SPI flash */
    {
        printf("SPIM flash initialize failed!\n");
    }

    SPIM_ReadJedecId(idBuf, sizeof (idBuf), 1);
    printf("SPIM get JEDEC ID=0x%02X, 0x%02X, 0x%02X\n", idBuf[0], idBuf[1], idBuf[2]);

    SPIM_DISABLE_CCM();
    SPIM_ENABLE_CACHE();

    if (SPIM_CIPHER_ON)
        SPIM_ENABLE_CIPHER();
    else
        SPIM_DISABLE_CIPHER();

    if (SPIM_Enable_4Bytes_Mode(USE_4_BYTES_MODE, 1) != 0)
    {
        printf("SPIM_Enable_4Bytes_Mode failed!\n");
    }

    SPIM->CTL1 |= SPIM_CTL1_CDINVAL_Msk;        // invalid cache
    SPIM_SetQuadEnable(1, 1);   //set quad mode bit
    SPIM_EnterDirectMapMode(USE_4_BYTES_MODE, CMD_DMA_FAST_QUAD_READ, 0);

}
```

In addition to SPIM initialization, you also need to edit the Scatter file to let the compiler know the location and content of the SPI Flash. In the Keil environment, you can follow the path "Options for Target -> Linker -> Scatter File -> Edit" to open the *.sct* file and edit its contents as follows:

```
;APROM
LOAD_ROM_1  0x0
{
    APROM.bin  0x0
    {
     startup_M460.o (RESET, +FIRST)
     *(+RO)
    }


    SRAM  0x20000000 0x80000
    {
     * (+RW, +ZI)
    }
}

;SPI flash
```

```
LOAD_ROM_2  0x100000
{
    SPIM.bin  0x100000
    {
     MicrosoftJhengHeiUI_16_Bold_EXT_AA4.o (+RO)
     MicrosoftJhengHeiUI_24_Bold_EXT_AA4.o (+RO)
     Account_Lock_Ini.o (+RO)
     Account_Pressed_Ini.o (+RO)
     Account_Lock_Ini_En.o (+RO)
     Account_Pressed_Ini_En.o (+RO)
     Bg_Cha.o (+RO)
     Bg2.o (+RO)
     Bg2_En.o (+RO)
     Bg3.o (+RO)
     Charge_Lock_Ini.o (+RO)
     Charge_Pressed_Ini.o (+RO)
     Charge_Lock_Ini_En.o (+RO)
     Charge_Pressed_Ini_En.o (+RO)
     che0.o (+RO)
     che1.o (+RO)
     che2.o (+RO)
     che3.o (+RO)
     che4.o (+RO)
     che5.o (+RO)
     Connect.o (+RO)
     Cost.o (+RO)
     Cost_0.o (+RO)
     Cost_En.o (+RO)
     Cost_0_En.o (+RO)
     Device.o (+RO)
     Device_0.o (+RO)
     Device_En.o (+RO)
     Device_0_En.o (+RO)
     ExpenseLine.o (+RO)
     Head_Ini.o (+RO)
     Info_Pressed_Ini.o (+RO)
     Info_Pressed_Ini_En.o (+RO)
     Lan_Cn.o (+RO)
     Lan_En.o (+RO)
     Login_Pressed_Ini.o (+RO)
     Login_Pressed_Ini_En.o (+RO)
     Mon.o (+RO)
     Mon_0.o (+RO)
     Mon_En.o (+RO)
     Mon_0_En.o (+RO)
     MonLine.o (+RO)
     Price_Pressed_Ini.o (+RO)
     Price_Pressed_Ini_En.o (+RO)
     Query_Lock_Ini.o (+RO)
     Query_Lock_Ini_En.o (+RO)
     Query_Pressed_Ini.o (+RO)
     Query_Pressed_Ini_En.o (+RO)
     Return0.o (+RO)
     Return1.o (+RO)
     Return0_En.o (+RO)
     Return1_En.o (+RO)
```

```
    Start.o (+RO)
    Stop_Pressed.o (+RO)
   }
}
```

After entering the main application function MainTask2(), the program will execute the relevant functions of AppWizard. Inside the while loop, there will be actions to draw graphics on the screen. In this example code, the APPW_SetVarData() function is used to pass values from the MCU to a variable in the AppWizard; Getting the text value in the AppWizard using the APPW_GetText() function; Using the APPW_SetText() function to pass characters from the MCU to the text variable in the AppWizard. By the method described in Step 1 (5) in Section 1.2.3, you can enter the username and password from the "Login" screen in the AppWizard, and the code checks the input password. If the password is correct, first set the "PW_1" variable to 1, then set the character "OK" to appear on the "PW_Sta" object on the "Login" screen. Finally, the username is displayed on the "Head" object of the "Login", the "Init", "Charge", "Account" and "Query" screens. If the password is wrong, first set the "PW_1" variable to 0 then set the character "ERR" to display on "PW_Sta" object of the "Login" screen.

```
void MainTask2(void)
{
    volatile int i=0;
    volatile int j=0;
    char UserName[20];
    char PW_Val[8] = {'q','q','q','q','q','q','q','q'};
    char PW_OK[3] = {'O','K',' '};
    char PW_ERR[3] = {'E','R','R'};
    char PW_Rev[9];
   // 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)
    {
      // Drawing a frame
      GUI_Exec1();
      APPW_Exec();
      NVT_Draw1();
         i = APPW_GetText(ID_SCREEN_Login,ID_EDIT_UserName_Log,UserName,8);
         i = APPW_GetText(ID_SCREEN_Login,ID_EDIT_Password_Log,PW_Rev,9);

         for(i = 0;i < 8;i++)
         {
             if(PW_Rev[i] != PW_Val[i])
             {
                 APPW_SetVarData(ID_VAR_PW_1, 0);
                 APPW_SetText(ID_SCREEN_Login,ID_TEXT_PW_Sta,PW_ERR);
                 break;
             }
         }
         if(i == 8)
```

```
        {
            APPW_SetVarData(ID_VAR_PW_1, 1);
            APPW_SetText(ID_SCREEN_Login,ID_TEXT_PW_Sta,PW_OK);
            APPW_SetText(ID_SCREEN_Init,ID_TEXT_Head_Ini,UserName);
            APPW_SetText(ID_SCREEN_Charge,ID_TEXT_Head_Cha,UserName);
            APPW_SetText(ID_SCREEN_Account,ID_TEXT_Head_Acc,UserName);
            APPW_SetText(ID_SCREEN_Query,ID_TEXT_Head_Query,UserName);
        }
    }
}
```

The drawing content performed by the AppWizard will be read from "ID_Screen_Init_Slots". You can try to add printf("Button Click\n") to the example code when the "Lan" button is clicked in Step 2 (7) in Section 1.2.3. If you click the "Lan" button, UART will print the "Button Click" content on the PC.

```
/*
File        : ID_SCREEN_Init_Slots.c
Purpose     : AppWizard managed file, function content could be changed-------------------
--------END-OF-HEADER------------------------------
*/

#include "Application.h"
#include "../Generated/Resource.h"
#include "../Generated/ID_SCREEN_Init.h"
/*** Begin of user code area ***/
/*** End of user code area ***/

/*********************************************************************
*
*       Public code
*
**********************************************************************
*/
/*********************************************************************
*
*       cbID_SCREEN_Init
*/
void cbID_SCREEN_Init(WM_MESSAGE * pMsg) {
  GUI_USE_PARA(pMsg);
}
/*********************************************************************
*
*       ID_SCREEN_Init__ID_BUTTON_Lan_Ini__WM_NOTIFICATION_CLICKED_0
*/
void ID_SCREEN_Init__ID_BUTTON_Lan_Ini__WM_NOTIFICATION_CLICKED_0(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);
  printf("Button Click\n");

}
```

3. Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M460_Series_BSP_CMSIS_V3.00.002
- emWin Software Package for M460 version
  - emWin Software Package for M460 V3.0
- IDE version
  - Keil uVersion 5.38.0.0
- AppWizard version
  - AppWizard V1.34a_6.30a

3.2 Hardware Requirements

- Circuit components
  - NuMaker-M467HJ V1.0
- Connect NuMaker-M467HJ V1.0 and PC
  - Connect the Nu-Link2-Me and the daughter board's screen USB connector to the PC using two USB cables, as shown in Figure 3-1.

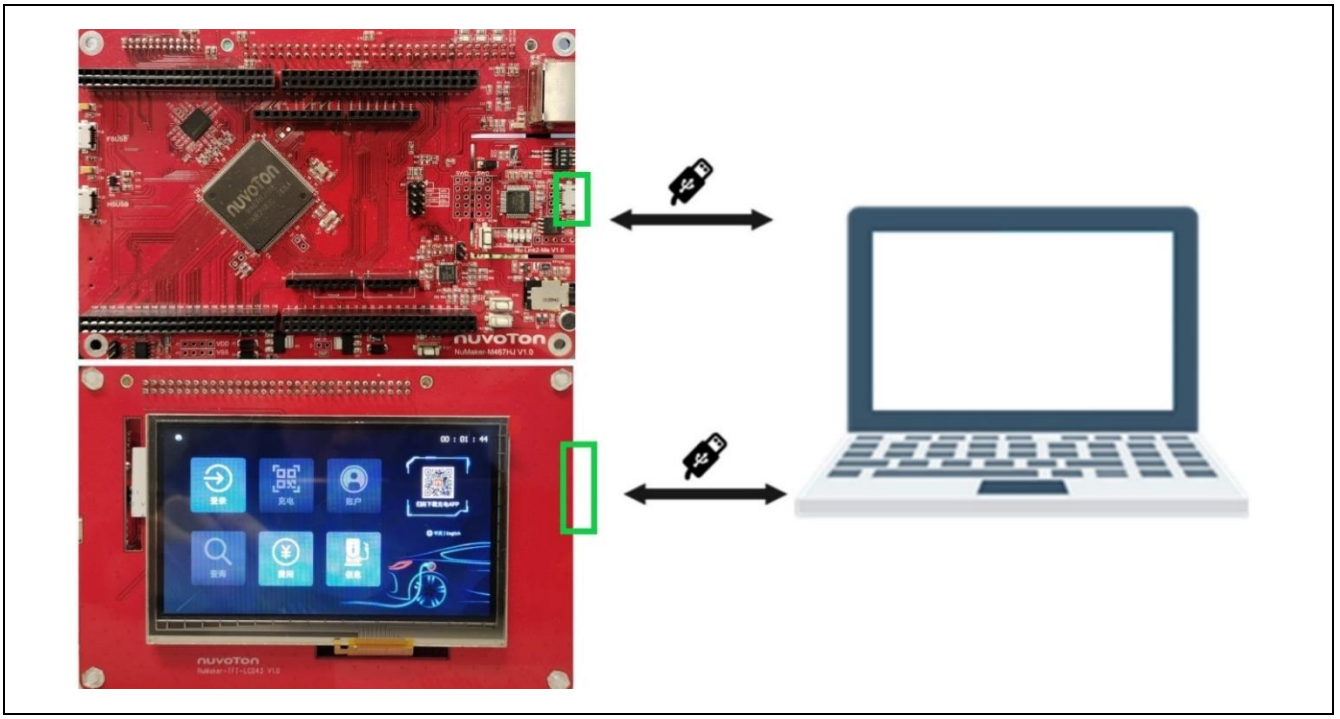


Figure 3-1 Connect NuMaker-M467HJ V1.0 and PC

4. Directory Information

The directory structure is shown below.

EC_M467_AppWizard_ChargePile_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	
Project	Source file of example code
ThirdParty	Header files for emWin project
EC_ChargePile.zip	Charge Pile sample project of AppWizard. Please open by AppWizard.exe.

Figure 4-1 Directory Structure

5. Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *M467_AppWizard_ ChargePile.uvprojx*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run.

6. Revision History

Date	Revision	Description
2023.12.07	1.00	Initial version.

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.