

DIView

Common Scripts

Ruby

2020/06/01



- The concepts of scripts
- The scripts of Basic Graphics
- The scripts of Window Controls
- The scripts of Extend Controls
- The Action scripts
- The Window scripts
- The Color scripts

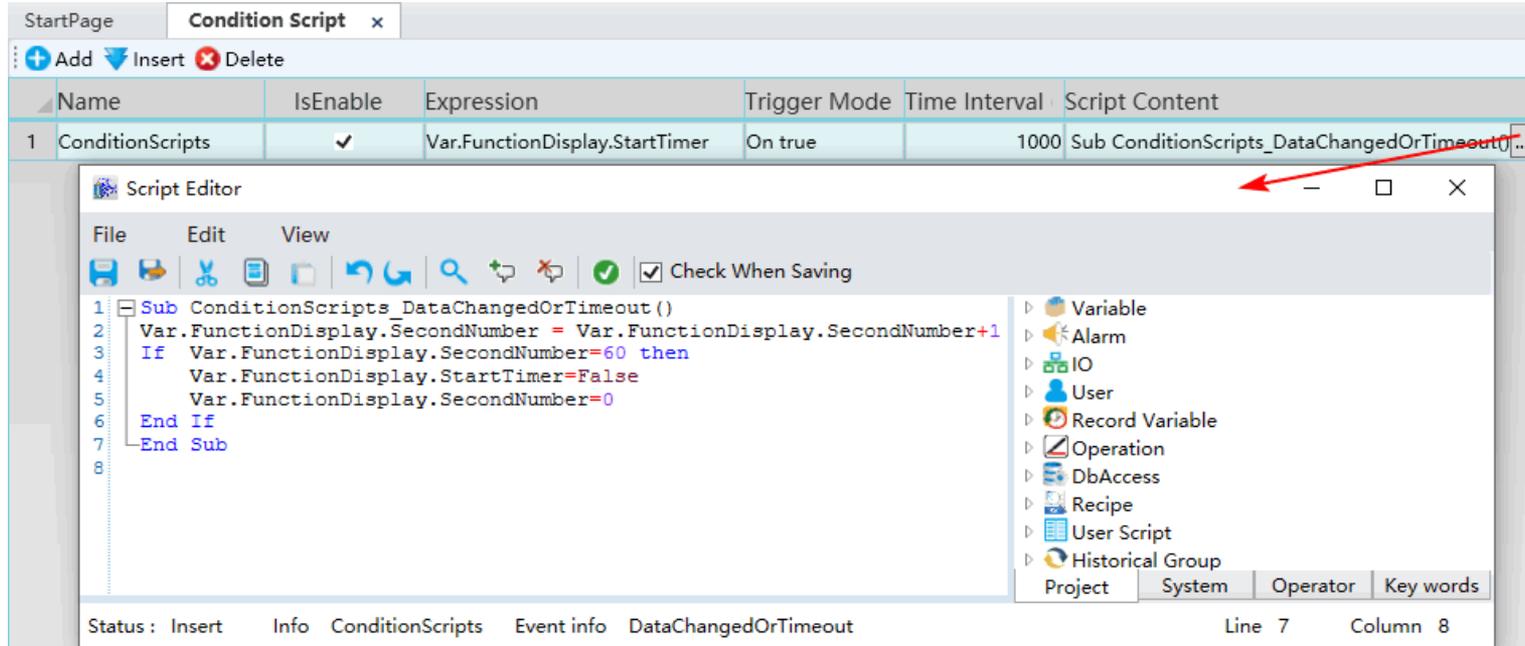
In this chapter, you will learn

- ... more about common scripts
- ... more about six types of scripts in the DIAView

- The concepts of scripts
- The scripts of Basic Graphics
- The scripts of Window Controls
- The scripts of Extend Controls
- The Action scripts
- The Window scripts
- The Color scripts

Although the functions of DIAView are comprehensive and powerful, the functions required by customers are different. Some functions need to be customized according to customer needs. DIAView can write related programs through a script editor to complete some special tasks and functions.

The event configuration, window program and user program in DIAView all need to use the script editor to write scripts. DIAView adopts VB Script language, users can write logic control programs according to VB Script language grammar specification, so as to complete specific functions and enhance the usability of the system.



Name	IsEnable	Expression	Trigger Mode	Time Interval	Script Content
1 ConditionScripts	✓	Var.FunctionDisplay.StartTimer	On true	1000	Sub ConditionScripts_DataChangedOrTimeout()...

```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 Sub ConditionScripts_DataChangedOrTimeout ()
2 Var.FunctionDisplay.SecondNumber = Var.FunctionDisplay.SecondNumber+1
3 If Var.FunctionDisplay.SecondNumber=60 then
4   Var.FunctionDisplay.StartTimer=False
5   Var.FunctionDisplay.SecondNumber=0
6 End If
7 End Sub
8
```

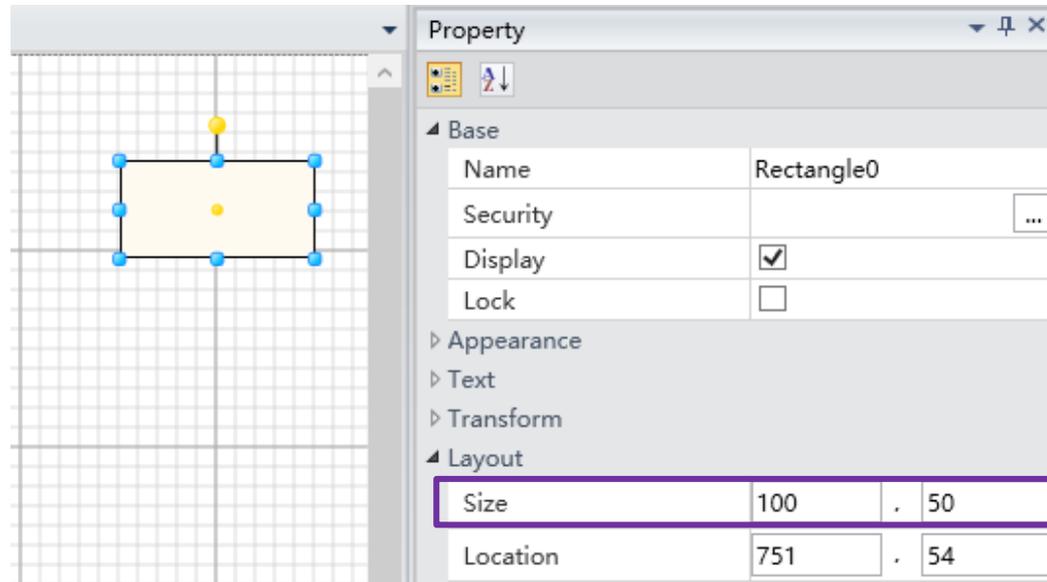
Status: Insert Info ConditionScripts Event info DataChangedOrTimeout Line 7 Column 8

- The concepts of scripts
- **The scripts of Basic Graphics**
- The scripts of Window Controls
- The scripts of Extend Controls
- The Action scripts
- The Window scripts
- The Color scripts

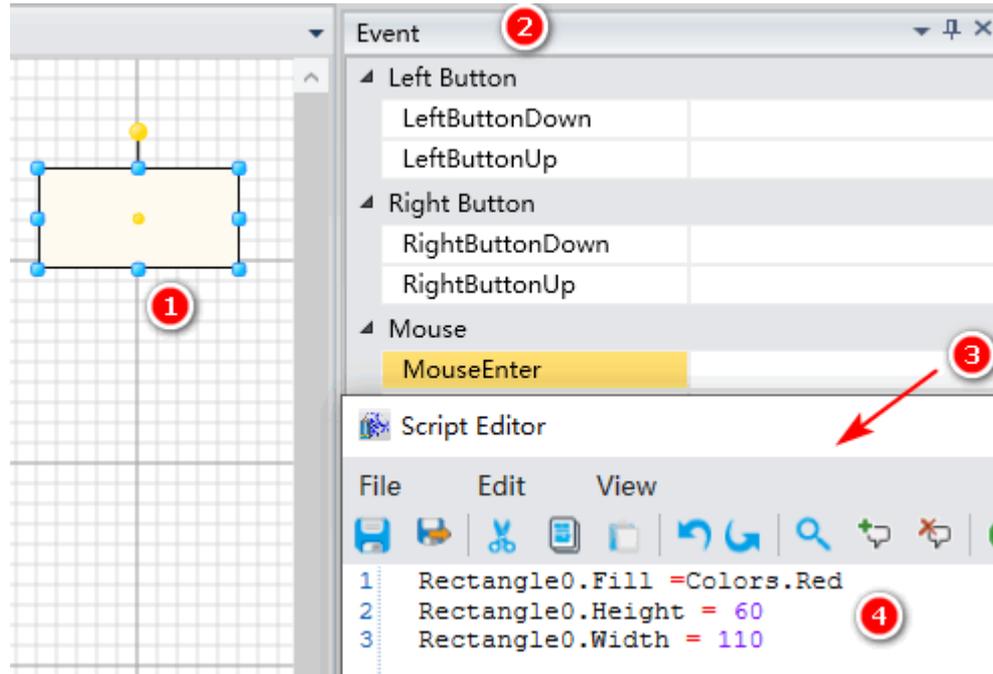
➤ Fill ,Height ,Width example:

Create a rectangle and set its properties with scripts

(1) Create a Rectangle0 (Size: 100*50) in the Window0



(2) Configure the MouseEnter event of the Rectangle0

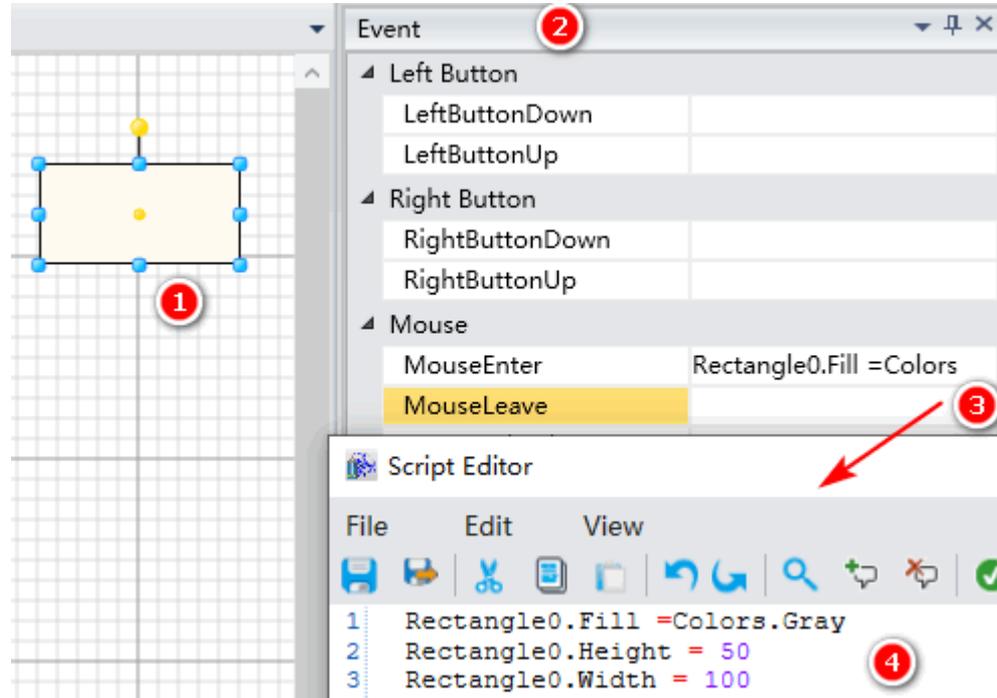


The screenshot shows a software interface with a grid on the left and two panels on the right. A yellow rectangle is positioned on the grid, with a red circle labeled '1' at its bottom-right corner. The 'Event' panel on the right has a red circle labeled '2' at its top. It lists three event categories: 'Left Button', 'Right Button', and 'Mouse'. The 'MouseEnter' event under the 'Mouse' category is highlighted in yellow and has a red circle labeled '3' at its bottom-right corner. A red arrow points from this event to the 'Script Editor' panel below. The 'Script Editor' panel has a menu bar with 'File', 'Edit', and 'View' options, and a toolbar with icons for file operations and editing. The script content is as follows:

```
1 Rectangle0.Fill = Colors.Red
2 Rectangle0.Height = 60
3 Rectangle0.Width = 110
```

A red circle labeled '4' is positioned at the end of the third line of code.

(3) Configure the MouseLeave event of the Rectangle0



The screenshot displays a software interface with a grid on the left and a configuration panel on the right. A yellow rectangle is positioned on the grid, with a red circle labeled '1' at its bottom-right corner. The configuration panel, titled 'Event', has a red circle labeled '2' at its top. It lists several events: 'Left Button', 'Right Button', and 'Mouse'. Under the 'Mouse' section, the 'MouseLeave' event is selected and highlighted in yellow, with a red circle labeled '3' at its bottom-right corner. Below the event configuration is a 'Script Editor' window with a menu bar (File, Edit, View) and a toolbar. The script editor contains three lines of code:

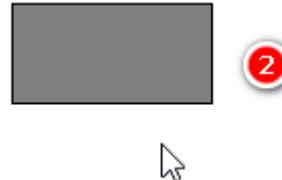
```
1 Rectangle0.Fill =Colors.Gray
2 Rectangle0.Height = 50
3 Rectangle0.Width = 100
```

 A red arrow points from the 'MouseLeave' event in the configuration panel to the script editor. A red circle labeled '4' is located at the bottom right of the script editor.

(4) Run the current project.



Red,110*60

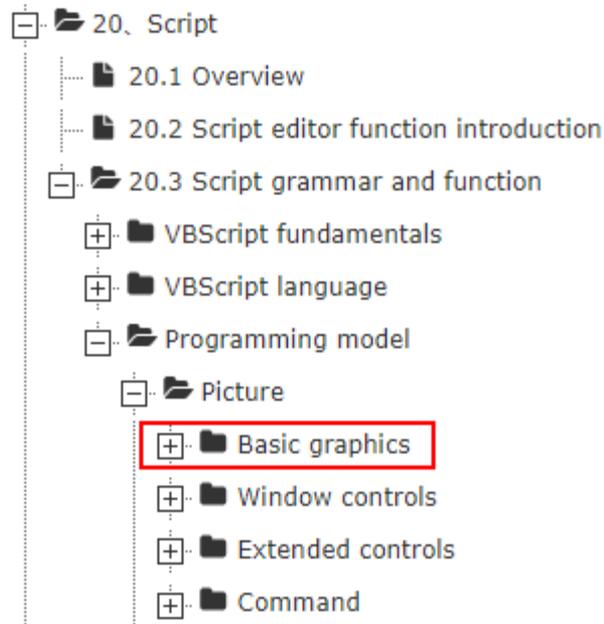


Gray,100*50

① When the mouse enters the Rectangle0, the Rectangle0 becomes red and the size of it becomes 110*60

② When the mouse leaves the Rectangle0, the Rectangle0 becomes gray and the size of it becomes 100*50

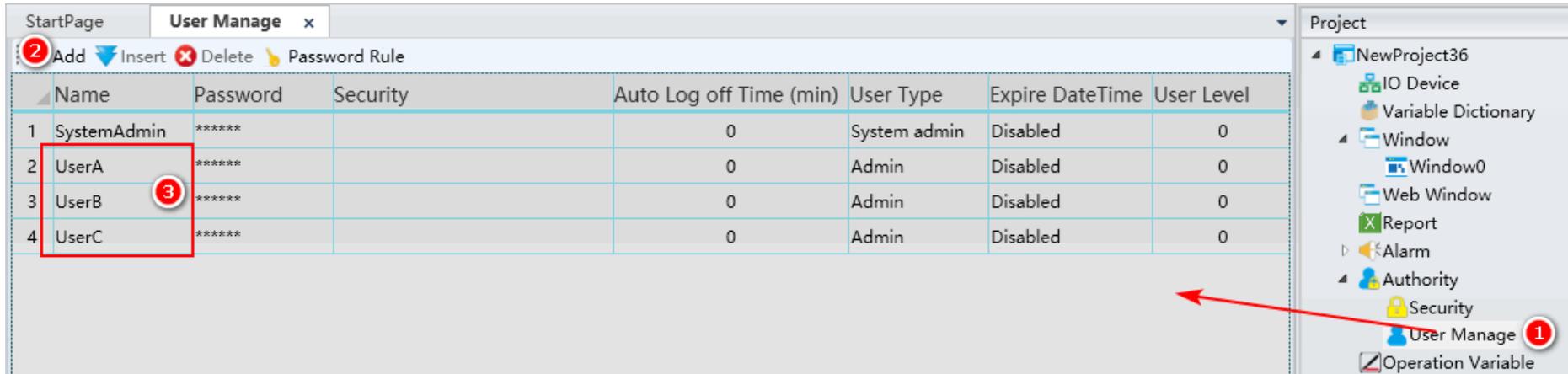
For more details about the scripts usage of basic graphics, please refer to the section “20.3 Script grammar and function” in the user manual. As shown in the figure below:



- The concepts of scripts
- The scripts of Basic Graphics
- **The scripts of Window Controls**
- The scripts of Extend Controls
- The Action scripts
- The Window scripts
- The Color scripts

➤ AddItems example1:

Get users name in the current project with AddItems script
(1) Create 3 users: UserA , UserB , UserC



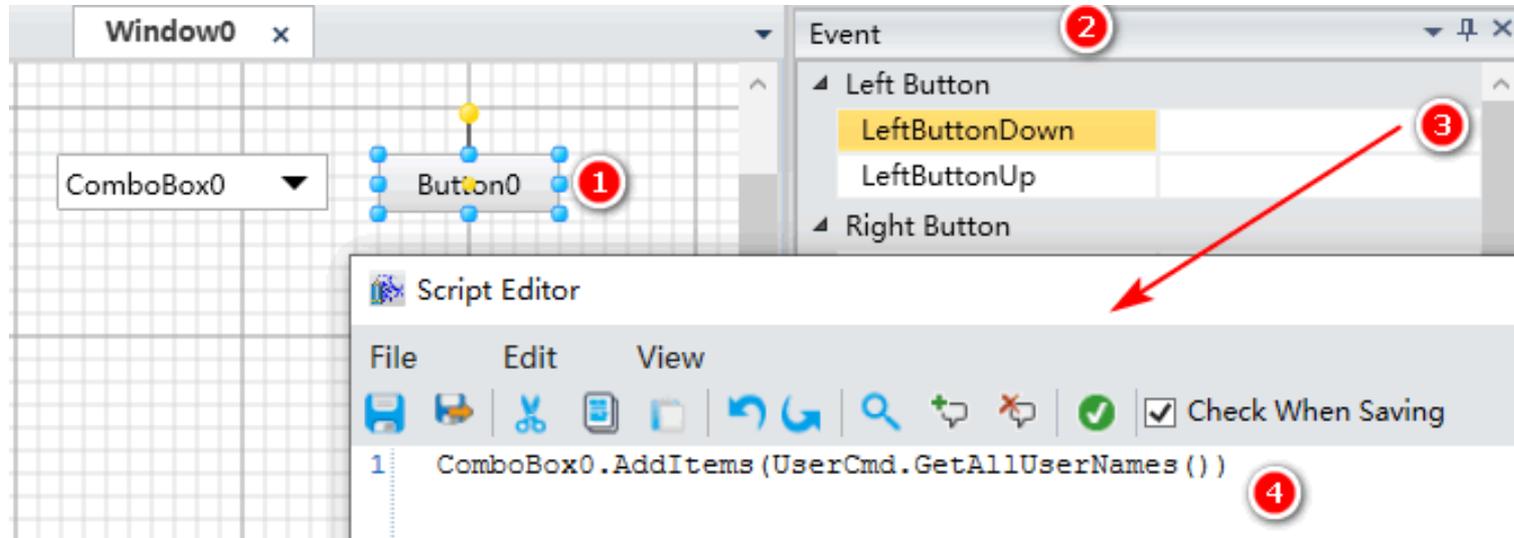
The screenshot shows a software interface with a 'User Manage' window. The window contains a table with the following data:

	Name	Password	Security	Auto Log off Time (min)	User Type	Expire DateTime	User Level
1	SystemAdmin	*****		0	System admin	Disabled	0
2	UserA	*****		0	Admin	Disabled	0
3	UserB	*****		0	Admin	Disabled	0
4	UserC	*****		0	Admin	Disabled	0

The 'User Manage' window is part of a larger application. The 'Project' tree on the right shows a hierarchy starting with 'NewProject36', which includes 'IO Device', 'Variable Dictionary', 'Window', 'Window0', 'Web Window', 'Report', 'Alarm', 'Authority', 'Security', 'User Manage', and 'Operation Variable'. A red circle with the number '1' is around the 'User Manage' item in the tree, and a red arrow points from it to the table. Another red circle with the number '2' is around the 'Add' button in the table's toolbar, and a red circle with the number '3' is around the 'UserB' row in the table.

※Refer to the section "12.3 User" in user manual.

(2) Create a ComboBox0 and a Button0 in the Window0, configure the LeftButtonDown event of the Button0.



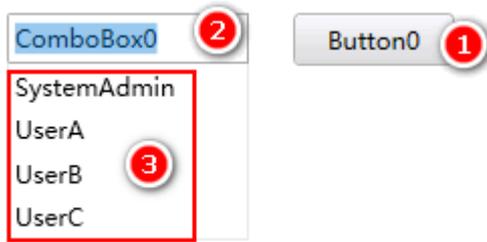
The screenshot displays a software development environment with the following components:

- Window0:** A grid-based workspace containing a **ComboBox0** and a **Button0**. A red circle with the number 1 is placed over the Button0.
- Event:** A panel on the right showing a tree view of events. The **Left Button** folder is expanded, and **LeftButtonDown** is selected and highlighted in yellow. A red circle with the number 2 is over the Event panel header, and a red circle with the number 3 is over the LeftButtonDown event.
- Script Editor:** A window with a menu bar (File, Edit, View) and a toolbar. The script content is:

```
1: ComboBox0.AddItems (UserCmd.GetAllUserNames ())
```

A red circle with the number 4 is placed over the end of the script line. A red arrow points from the LeftButtonDown event in the Event panel to the script editor.

(3) Run the current project.

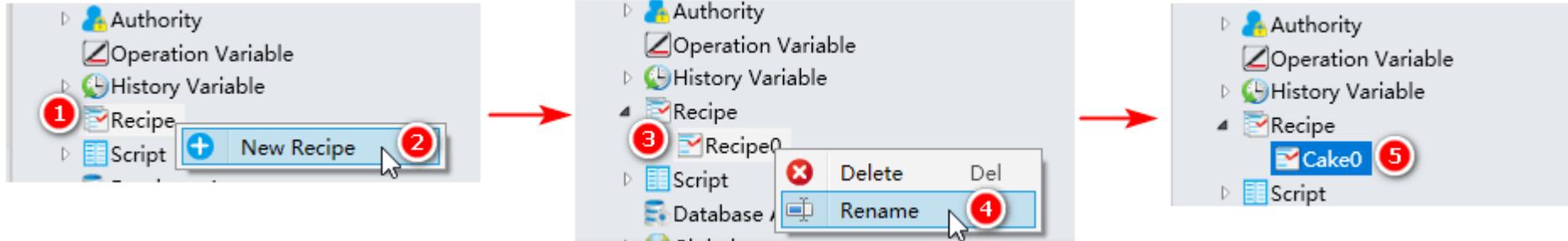


① Click the Button0

②③ The ComboBox0 displays the all users' name in the current project—
SystemAdmin , UserA , UserB , UserC

➤ AddItems example2:

Get recipe information with AddItems script
(1) Create 1 recipe: Cake0

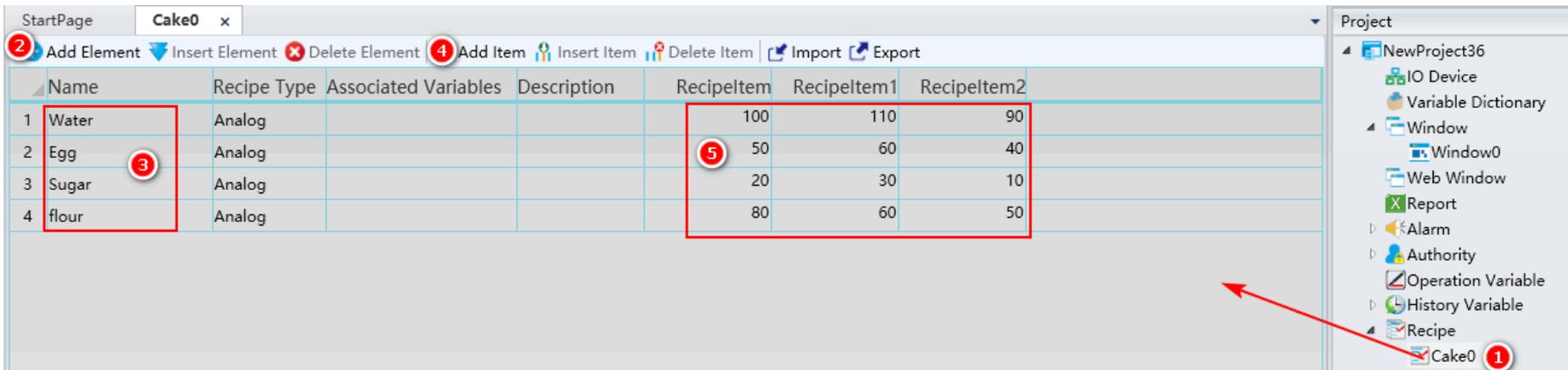


① ③ Right click

② ④ Click

⑤ Rename as “Cake0”

(2) Configure parameters for cake0



The screenshot displays the 'Cake0' configuration window. The table below shows the recipe items:

	Name	Recipe Type	Associated Variables	Description	Recipeltem	Recipeltem1	Recipeltem2
1	Water	Analog			100	110	90
2	Egg	Analog			50	60	40
3	Sugar	Analog			20	30	10
4	flour	Analog			80	60	50

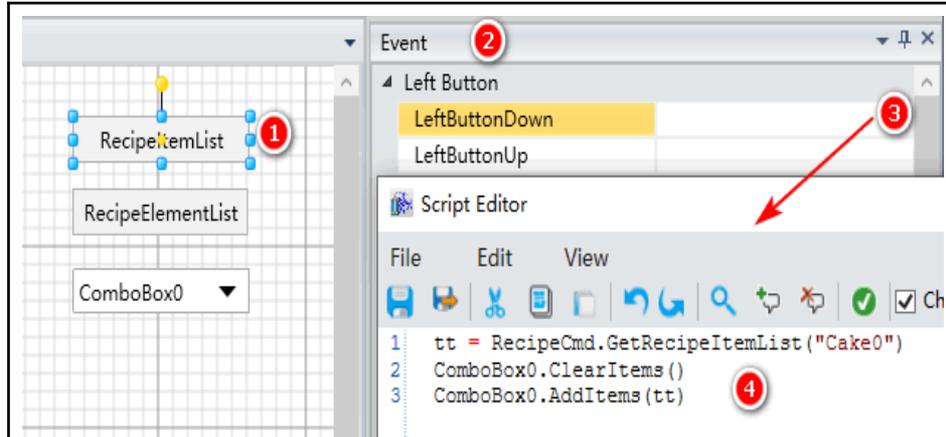
The project tree on the right shows the following structure:

- NewProject36
 - IO Device
 - Variable Dictionary
 - Window
 - Window0
 - Web Window
 - Report
 - Alarm
 - Authority
 - Operation Variable
 - History Variable
 - Recipe
 - Cake0

※Refer to the section "15.2 Recipe configuration" in user manual.

AddItems Script of ComboBox

(3) Create a ComboBox0 and two buttons(RecipeItemList, RecipeElementList) in the Window0, configure the LeftButtonDown event of the two buttons.



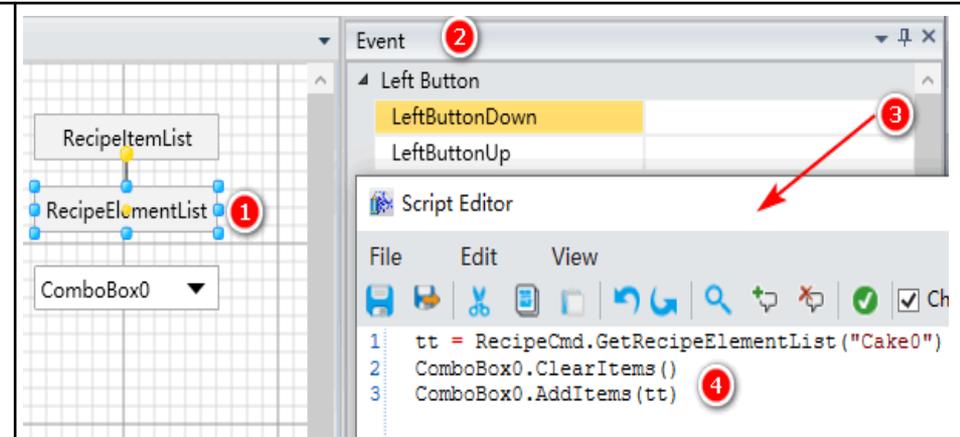
Event 2

- Left Button
 - LeftButtonDown
 - LeftButtonUp

Script Editor

```
File Edit View  
1 tt = RecipeCmd.GetRecipeItemList("Cake0")  
2 ComboBox0.ClearItems()  
3 ComboBox0.AddItems(tt)
```

RecipeItemList



Event 2

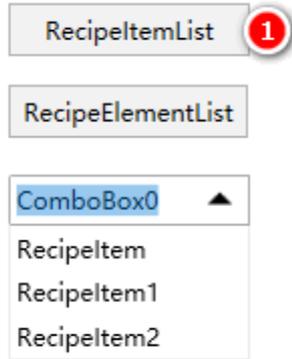
- Left Button
 - LeftButtonDown
 - LeftButtonUp

Script Editor

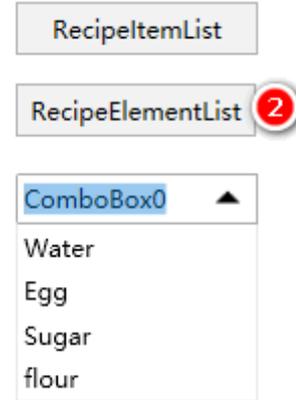
```
File Edit View  
1 tt = RecipeCmd.GetRecipeElementList("Cake0")  
2 ComboBox0.ClearItems()  
3 ComboBox0.AddItems(tt)
```

RecipeElementList

(4)Run the current project.



RecipeltemList



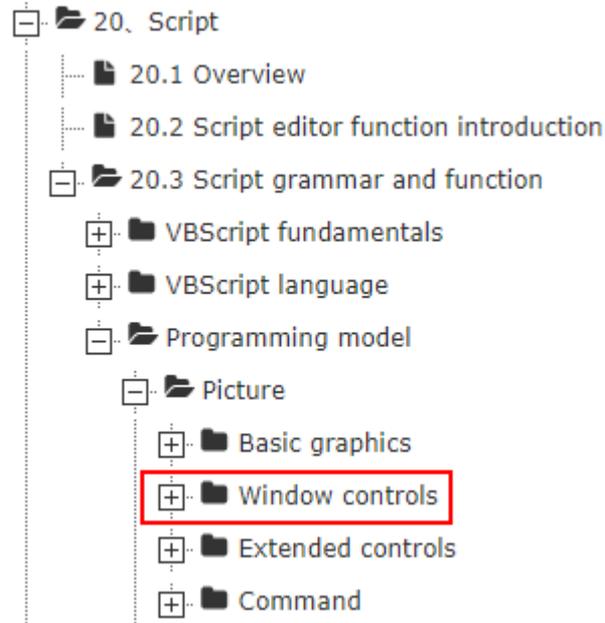
RecipeElementList

①Click the “RecipeltemList” button, the ComboBox0 displays the recipe items of Cake0—Recipeltem , Recipeltem1 , Recipeltem2

②Click the “RecipeElementList” button, the ComboBox0 displays the recipe elements of Cake0—Water , Egg , Sugar, flour

The Scripts of Window Controls

For more details about the scripts usage of window controls, please refer to the section “20.3 Script grammar and function” in the user manual. As shown in the figure below:

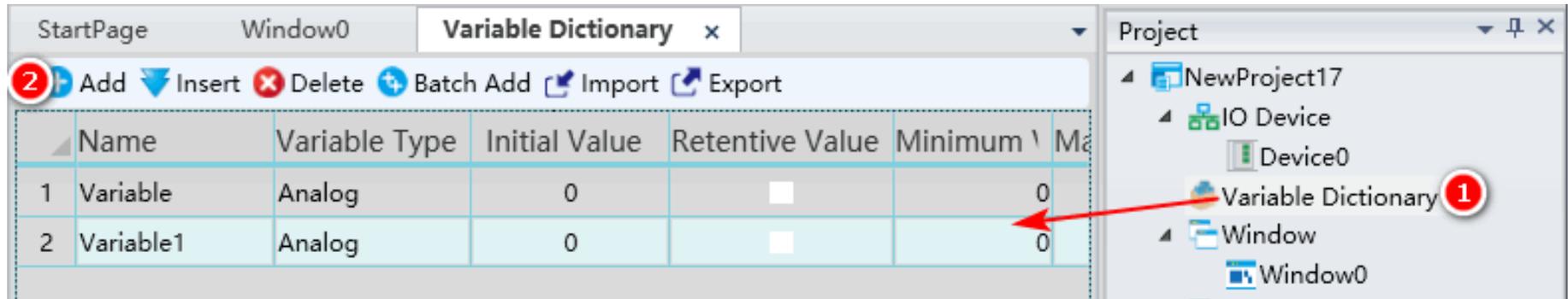


- The concepts of scripts
- The scripts of Basic Graphics
- The scripts of Window Controls
- **The scripts of Extend Controls**
- The Action scripts
- The Window scripts
- The Color scripts

➤ QueryHistoryDate example :

Query the history data from the moment the project starts to the current time

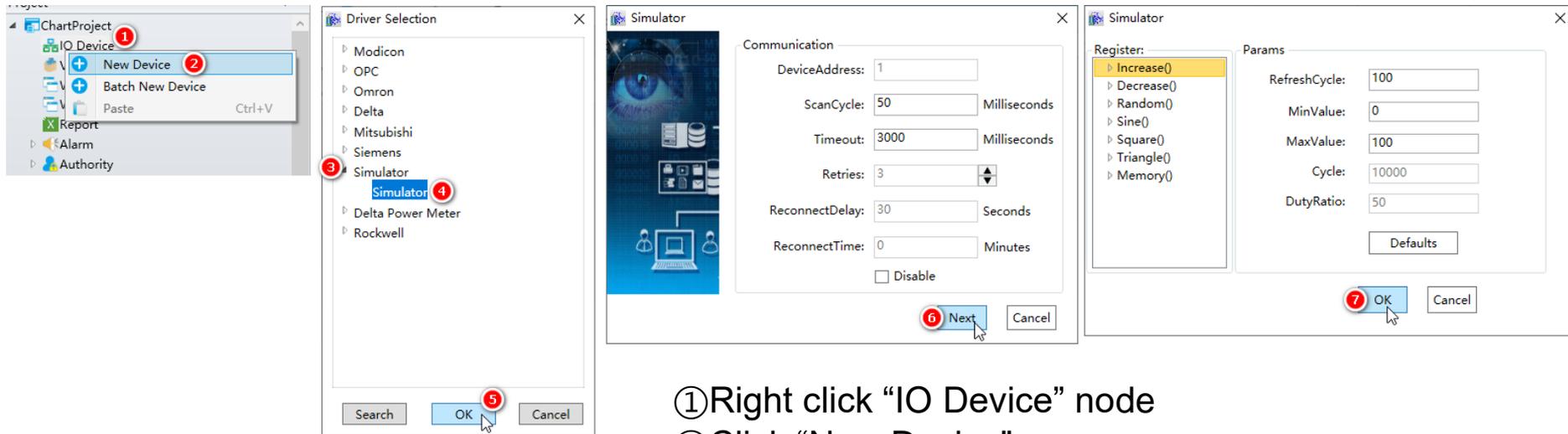
(1) Create 2 variables : Variable , Variable1



Name	Variable Type	Initial Value	Retentive Value	Minimum \	Ma
1 Variable	Analog	0	<input type="checkbox"/>	0	
2 Variable1	Analog	0	<input type="checkbox"/>	0	

✘ Refer to the section "6.3 Variables" in user manual.

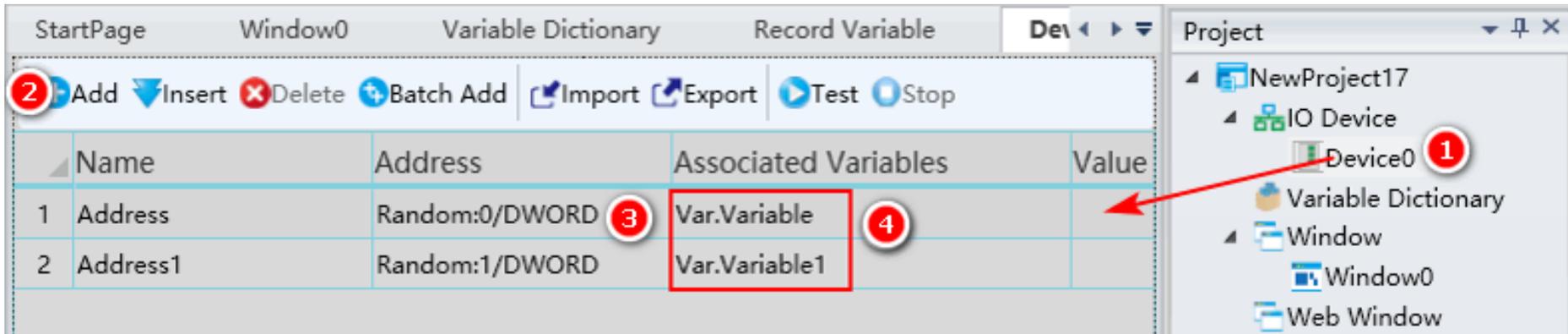
(2) Create a simulated device : Device0



- ① Right click “IO Device” node
- ② Click “New Device”
- ③④ Double click “Simulator”

※ Refer to the section “5.10.1 Simulator” in user manual.

(3) Create two simulation address in the Device0 that associated with Variable, Variable1 respectively



StartPage Window0 Variable Dictionary Record Variable Dev

2 Add Insert Delete Batch Add Import Export Test Stop

	Name	Address	Associated Variables	Value
1	Address	Random:0/DWORD	Var.Variable	
2	Address1	Random:1/DWORD	Var.Variable1	

Project

- NewProject17
 - IO Device
 - Device0
 - Variable Dictionary
 - Window
 - Window0
 - Web Window

(4) Create two historical variables in the Record Variable that associated with Variable, Variable1 respectively

The screenshot displays the software interface for configuring variables and timers. The main window shows a table with columns: Name, Associated Variables, Mode, Timer, and Deadband. Two rows are visible: 'RecordVariable' and 'RecordVariable1'. The 'Associated Variables' column for both rows contains 'Var.Variable' and 'Var.Variable1' respectively. A red box highlights the 'Associated Variables' column for the first row, and a red circle '3' is placed over the text 'Var.Variable'. A red circle '4' is placed over the 'Timer' column for the second row, which contains 's1'. A red arrow points from this 's1' to the 'Timer Browser' window below. The 'Timer Browser' window shows a table with columns: IsEnable, Name, Timer Unit, Timer Coefficient, Relative Time, and Description. The first row is selected, showing 's1' with 'IsEnable' checked. A red circle '5' is placed over the 'IsEnable' checkbox. On the right side, a tree view shows the project structure, with 'Record Variable' highlighted and a red circle '1' next to it.

Name	Associated Variables	Mode	Timer	Deadband
1 RecordVariable	Var.Variable	Timing	s1	N/A
2 RecordVariable1	Var.Variable1	Timing	s1	N/A

IsEnable	Name	Timer Unit	Timer Coefficient	Relative Time	Description
<input checked="" type="checkbox"/>	s1	Second	1	0001-01-01 00:00:00	Triggers every second
<input type="checkbox"/>	s10	Second	10	0001-01-01 00:00:00	Triggers every 10 seconds
<input type="checkbox"/>	s30	Second	30	0001-01-01 00:00:00	Triggers every 30 seconds
<input type="checkbox"/>	m1	Minute	1	0001-01-01 00:00:00	Triggers every minute
<input type="checkbox"/>	m30	Minute	30	0001-01-01 00:00:00	Triggers every 30 minutes

✘Refer to the section "14.2 Setting history record variable" in user manual.



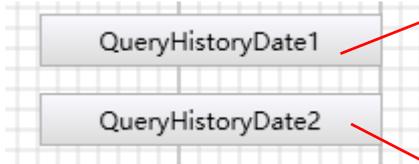
The Scripts of HistoryChart

(5) Create a HistoryChart0 in the Window0, and add 2 curves in the HistoryChart0, Series0 associated RecordVariable, Series1 associated RecordVariable1

The screenshot shows the configuration of a History Chart. The **History Series Editor** window is open, showing two series: **Series0** and **Series1**, both set to **Line** style. The **Display Record Variables** dialog is also open, showing the **Record Variable** list with **RecordVariable1** selected. Red callouts and arrows indicate the following steps:

- 1**: Click the ellipsis button next to **SeriesCollection** in the **Series** list.
- 2**: Click the **Add** button in the **History Series Editor**.
- 3**: Select **Series1** in the **Member(2/2)** list.
- 4**: Click the ellipsis button next to **VariablePath** in the **Display Record Variables** dialog.
- 5**: Select **RecordVariable1** in the **Variables Recorded** list.

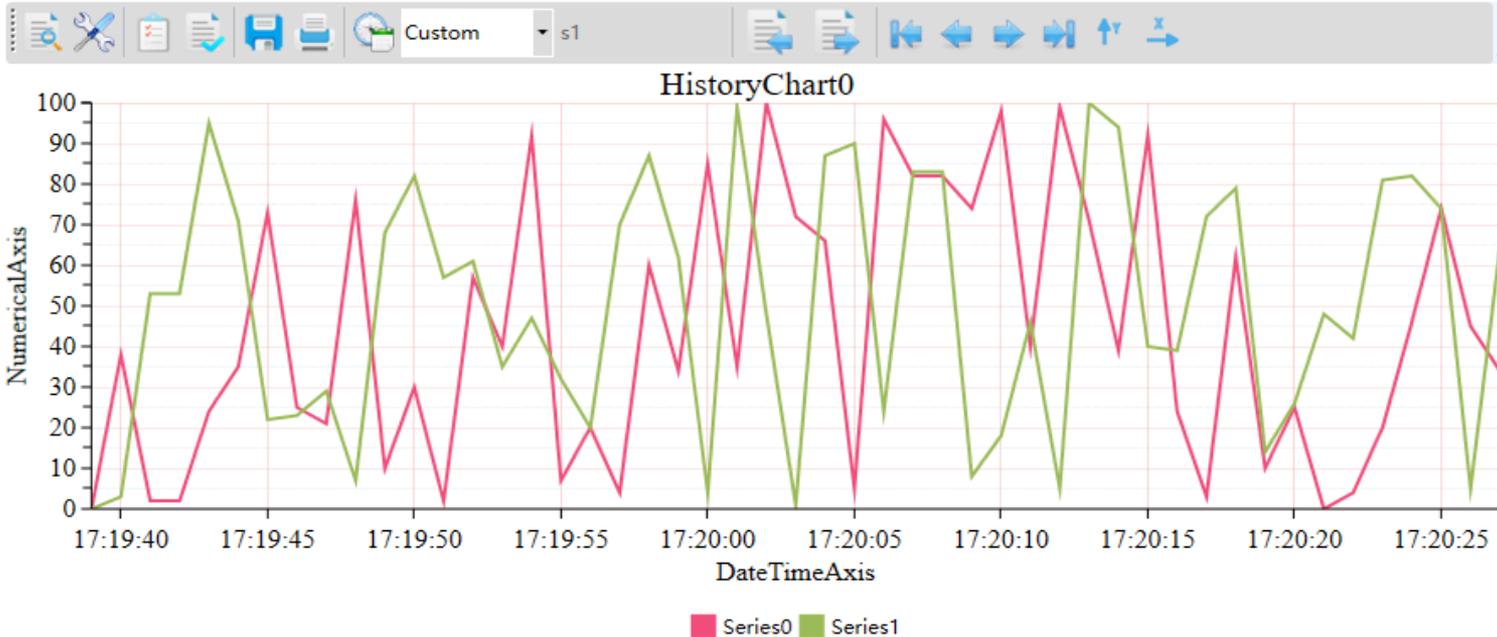
(6) Create two buttons in the Window0, configure the LeftButtonDown event of the two buttons



```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 Call HistoryChart0.QueryStartTime (Sys.StartTime)
2 'Set the query start time to the moment when the project starts
3 Call HistoryChart0.QueryEndTime (Sys.Now)
4 'Set the query end time to the system current time
5 Call HistoryChart0.SetVarRecordRulerName ("s1")
6 Call HistoryChart0.QueryHistoryDate ()
```

```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 Call HistoryChart0.SetVarRecordRulerName ("s1")
2 Call HistoryChart0.QueryHistoryDate (Sys.StartTime, Sys.Now, 1, 0)
```

(7) Run the Window0

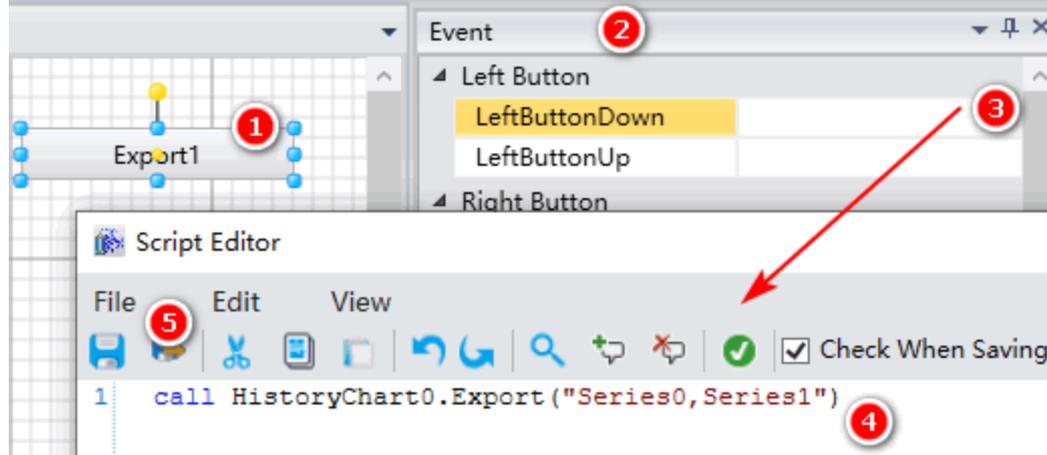


Click the "QueryHistoryDate1" or "QueryHistoryDate2" button · the HistoryChart0 displays the history data from the moment the project starts to the current time

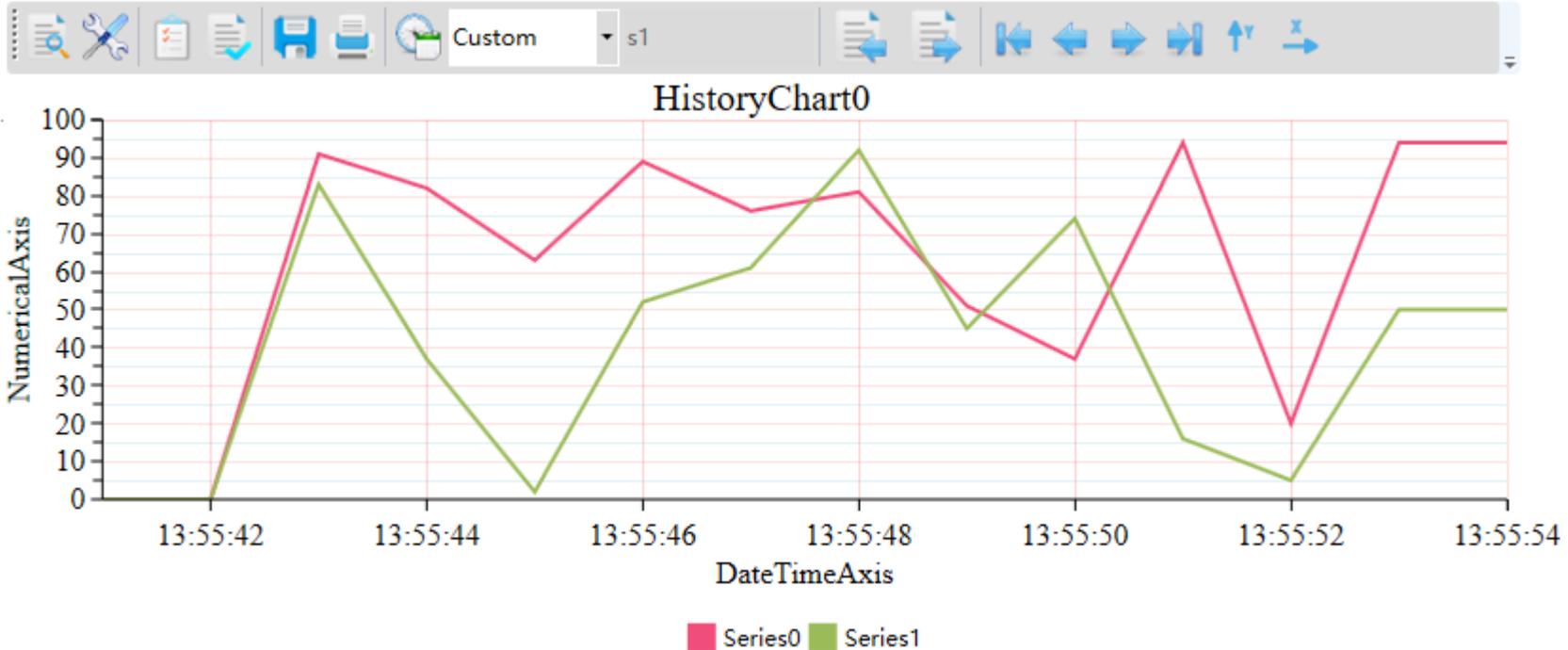
➤ **Export** example1 :

Export history data over a period of time

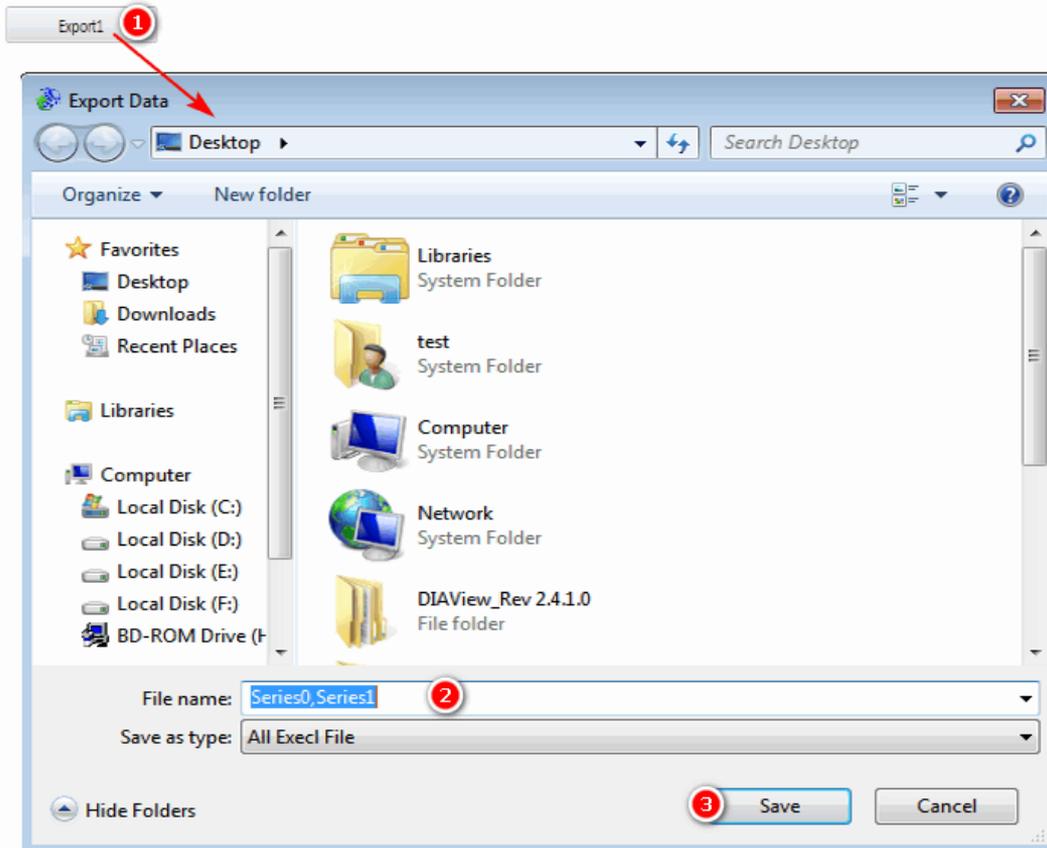
The first 5 steps are the same as(1)(2)(3)(4)(5) steps of QueryHistoryDate example
(6)Create one button(Export1) in the Window0, configure the LeftButtonDown event of the button



(7) Run the Window0, query history data



(8) Export the history data queried in the previous step

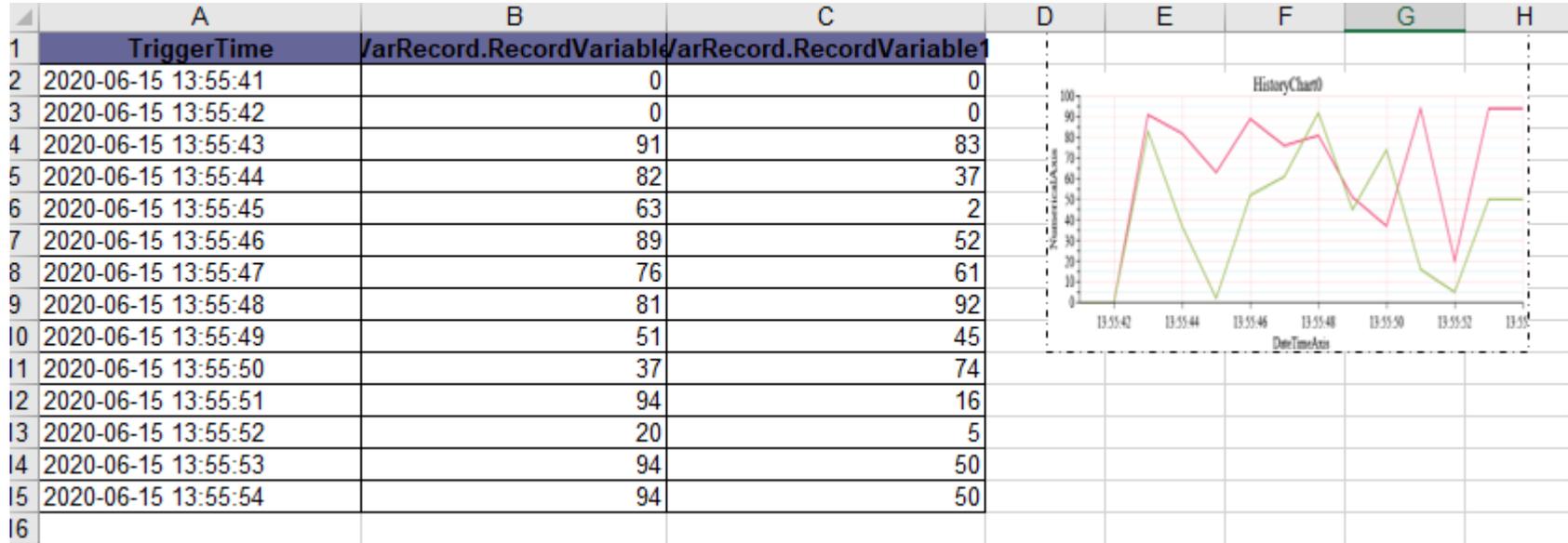


Click the “Export1” button, then Series0 and Series1 of HistoryChart0 are exported to the desktop successfully, and the name of Excel file is “Series0, Series1”



The Scripts of HistoryChart

(9) The exported Excel file is shown below



The Scripts of HistoryChart

➤ Export example2 :

Export history data over a period of time by report template

The first 5 steps are the same as(1)(2)(3)(4)(5) steps of QueryHistoryDate example

(6)Create a report template(Report0) and configure history data for it

The screenshot displays the software interface for configuring a report template. The ribbon at the top includes toolbars for Font, Cell, Realtime Variable, History Data, Data Source, Report Guide, Print Setting, and PDF Setting. The main workspace shows a grid with columns A-F and rows 1-24. A 'History Data' dialog box is open, with the following fields and values:

- DataPoint: (empty)
- BaseLineTime: (empty)
- Year: Year
- Month: Month
- Day: Day
- Hour: Hour
- Minute: Minute
- Second: Second
- Category: TriggerTime

The 'History Variables' dialog box is also open, showing the following configuration:

- HistoryVariables: (empty)
- Historical Group: RecordVariable
- RecordVariable: RecordVariable1

The 'Project' pane on the right shows a tree view of the project structure, with 'Report0' highlighted under the 'Report' folder.

Red circles with numbers 1 through 9 highlight the following steps in the configuration process:

- 1: Selecting 'Report0' in the Project pane.
- 2: Clicking on cell A1 in the grid.
- 3: Clicking on the 'History Data' dialog box.
- 4: Selecting 'TriggerTime' in the Category dropdown.
- 5: Clicking on the 'DataPoint' field.
- 6: Clicking on the 'History Variables' dialog box.
- 7: Clicking on the 'RecordVariable' field.
- 8: Clicking on the 'OK' button in the 'History Variables' dialog box.
- 9: Clicking on the 'OK' button in the 'History Data' dialog box.

(7) The configuration result of Report0 is as follows

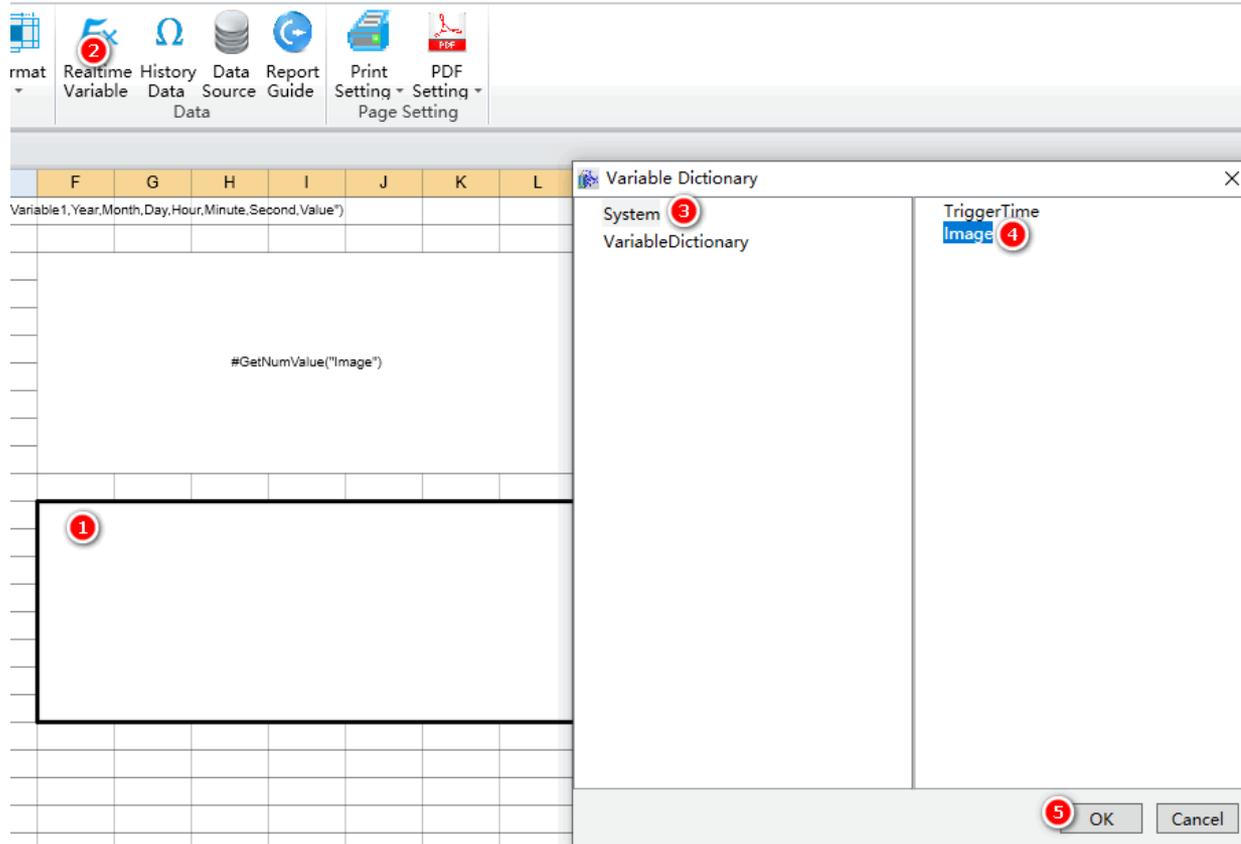
StartPage	Window0	Report0 x											
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	#GetHisData	#GetHisDa	#GetHisData("VarRecord.RecordVariable1,Year,Month,Day,Hour,Minute,Second,Value")										
2													
3													
4													
5													
6													

#GetHisData("VarRecord.RecordVariable1,Year,Month,Day,Hour,Minute,Second,Value")

#GetHisData("VarRecord.RecordVariable,Year,Month,Day,Hour,Minute,Second,Value")

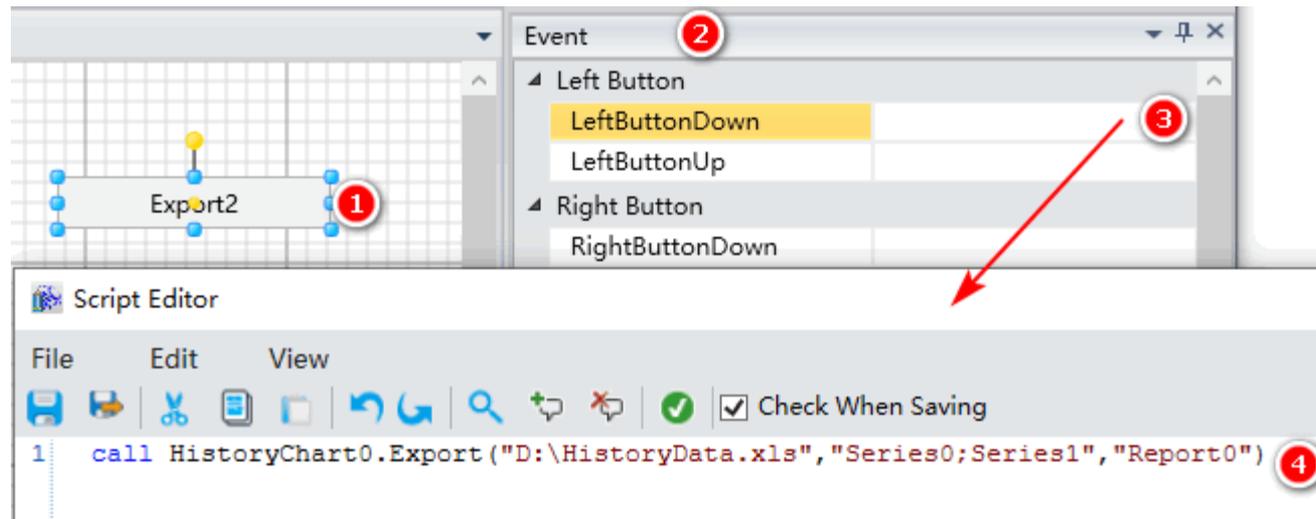
#GetHisData("VarRecord.RecordVariable,Year,Month,Day,Hour,Minute,Second,TriggerTime")

(8) Configure two areas used to display images for Report0



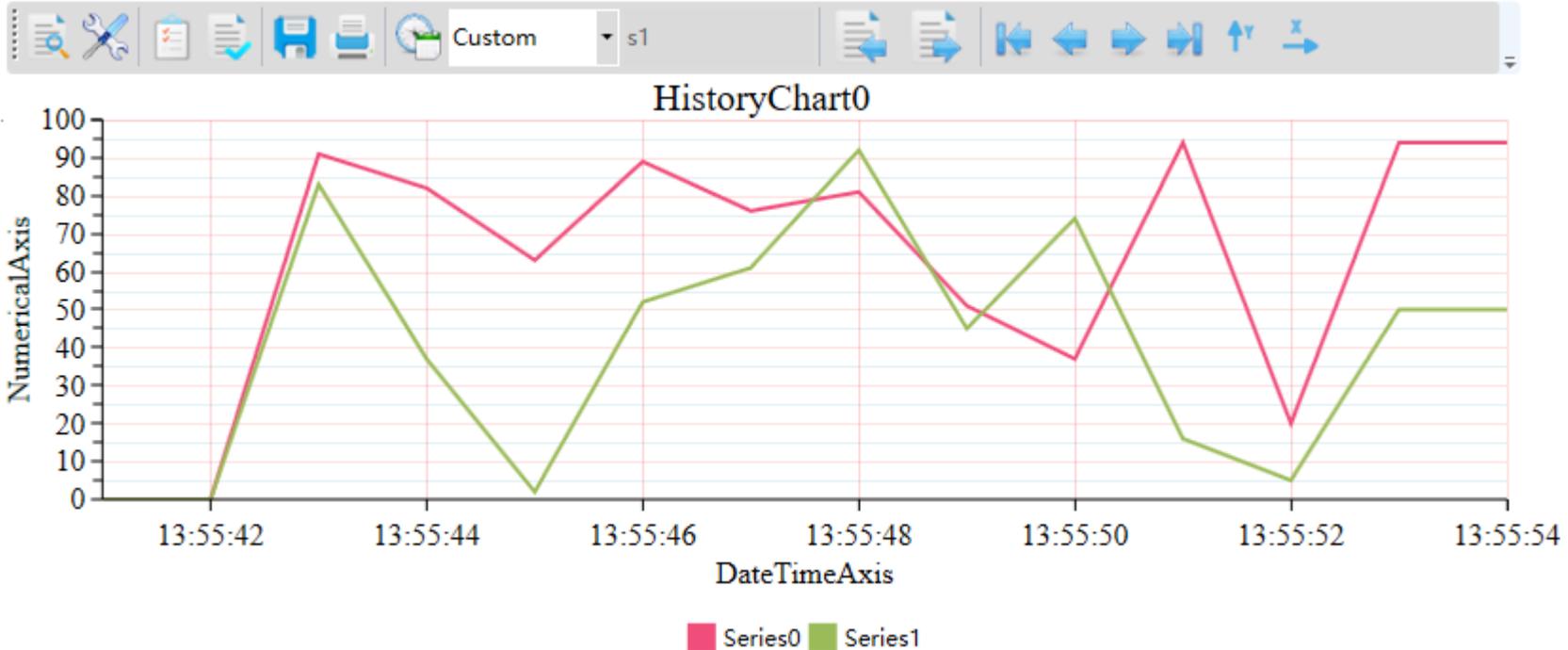
The screenshot displays the DELTA software interface. At the top, a menu bar includes options like Realtime Variable, History Data, Data Source, Report Guide, Print Setting, and PDF Setting. A red circle with the number '2' highlights the 'Realtime Variable' menu item. Below the menu bar, a grid area contains a formula: `#GetNumValue("Image")`. A red circle with the number '1' highlights a rectangular selection box in the grid. A 'Variable Dictionary' dialog box is open, showing a tree view with 'System' and 'VariableDictionary'. A red circle with the number '3' highlights the 'System' node. Under 'VariableDictionary', the 'TriggerTime' node is expanded, and a red circle with the number '4' highlights the 'image' sub-node. At the bottom of the dialog box, a red circle with the number '5' highlights the 'OK' button.

(9) Create one button(Export2) in the Window0, configure the LeftButtonDown event of the button



The screenshot displays a software development environment. On the left, a grid-based workspace contains a button labeled "Export2" with a red circle '1' next to it. On the right, an "Event" panel (marked with a red circle '2') lists events for "Left Button" and "Right Button". The "LeftButtonDown" event is selected and highlighted in yellow, with a red circle '3' next to it. A red arrow points from this event to the "Script Editor" below. The "Script Editor" has a menu bar (File, Edit, View) and a toolbar with various icons. A red circle '4' is next to the script code: `1 call HistoryChart0.Export("D:\HistoryData.xls", "Series0;Series1", "Report0")`.

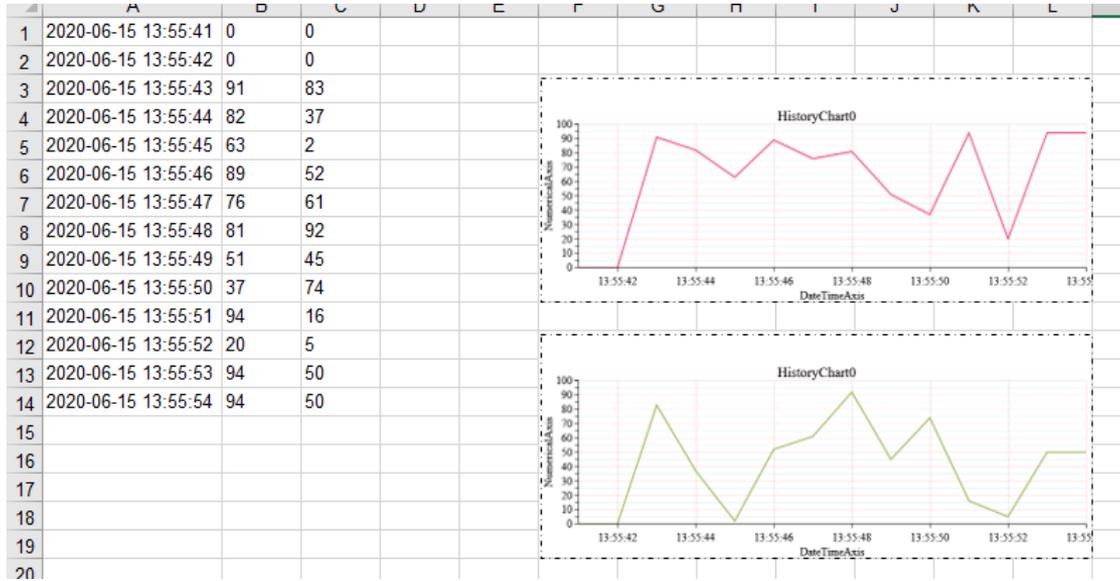
(10) Run the Window0, query history data





The Scripts of HistoryChart

(11) Export the history data by report template, the exported excel file is shown below



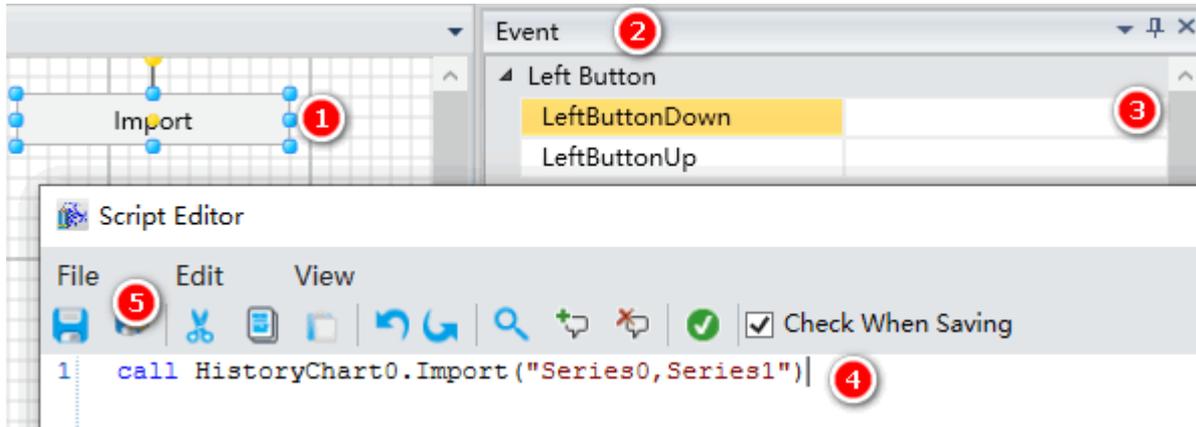
Click the “Export2” button, then Series0 and Series1 of HistoryChart0 are exported to the disk D successfully, and the file name is “HistoryData”

➤ Import example :

Import history data from excel file to history chart

The first 5 steps are the same as(1)(2)(3)(4)(5) steps of QueryHistoryDate example

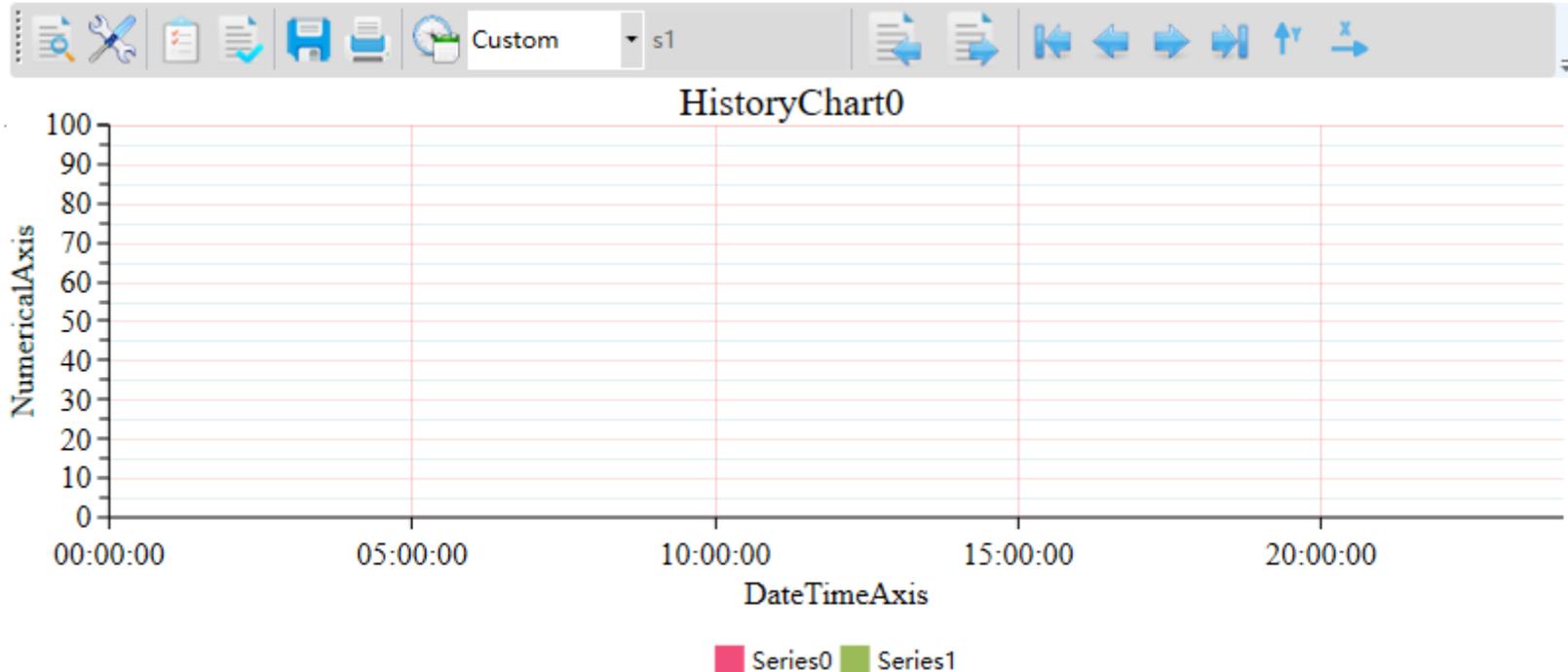
(6)Create one button(Import) in the Window0, configure the LeftButtonDown event of the button



The screenshot displays a software interface with the following components:

- Event Window (2):** Shows the 'Left Button' event configuration. The 'LeftButtonDown' event is selected (3).
- Script Editor (4):** Contains the script: `call HistoryChart0.Import ("Series0, Series1")`.
- File Menu (5):** Shows the 'File' menu with icons for Save, Copy, Paste, Undo, Redo, Find, and Check When Saving.
- Import Button (1):** A button labeled 'Import' is visible in the background.

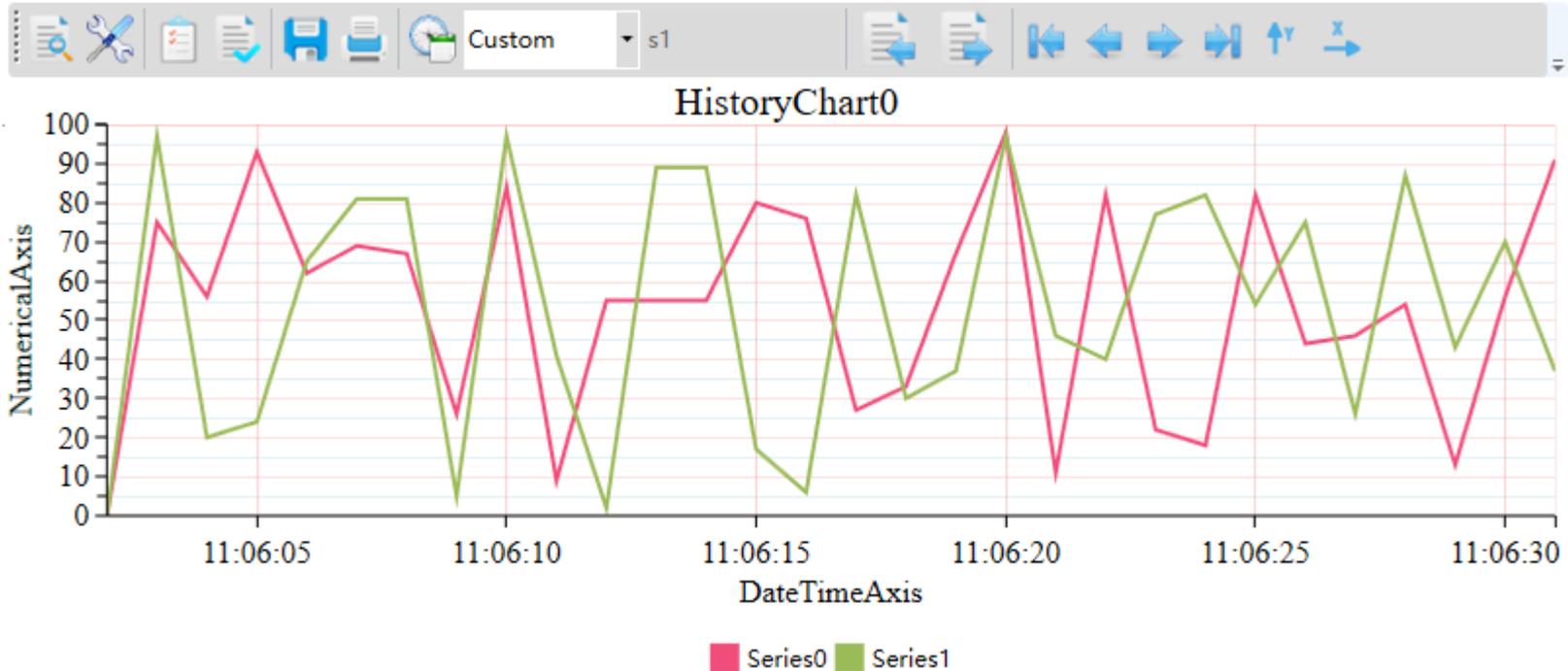
(7) Run the Window0



(8) Click the “Import” button to import history data to HistoryChart0



(9)The history data are imported to HistoryChart0 successfully, as shown below

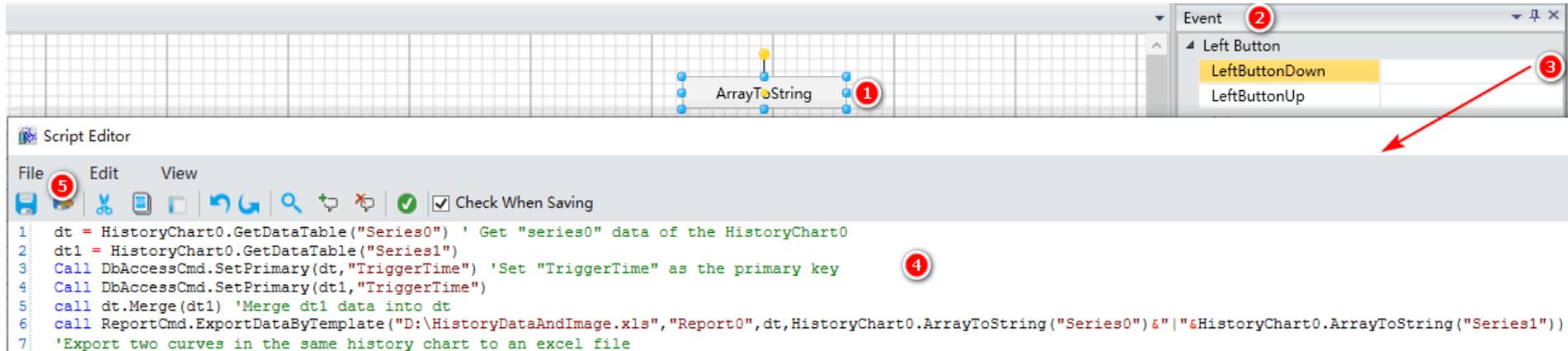


➤ ArrayToString example :

Export two curves in the same history chart to an excel

The first 8 steps are the same as the first 8 steps of Export example2

(9) Create one button(ArrayToString) in the Window0, configure the LeftButtonDown event of the button

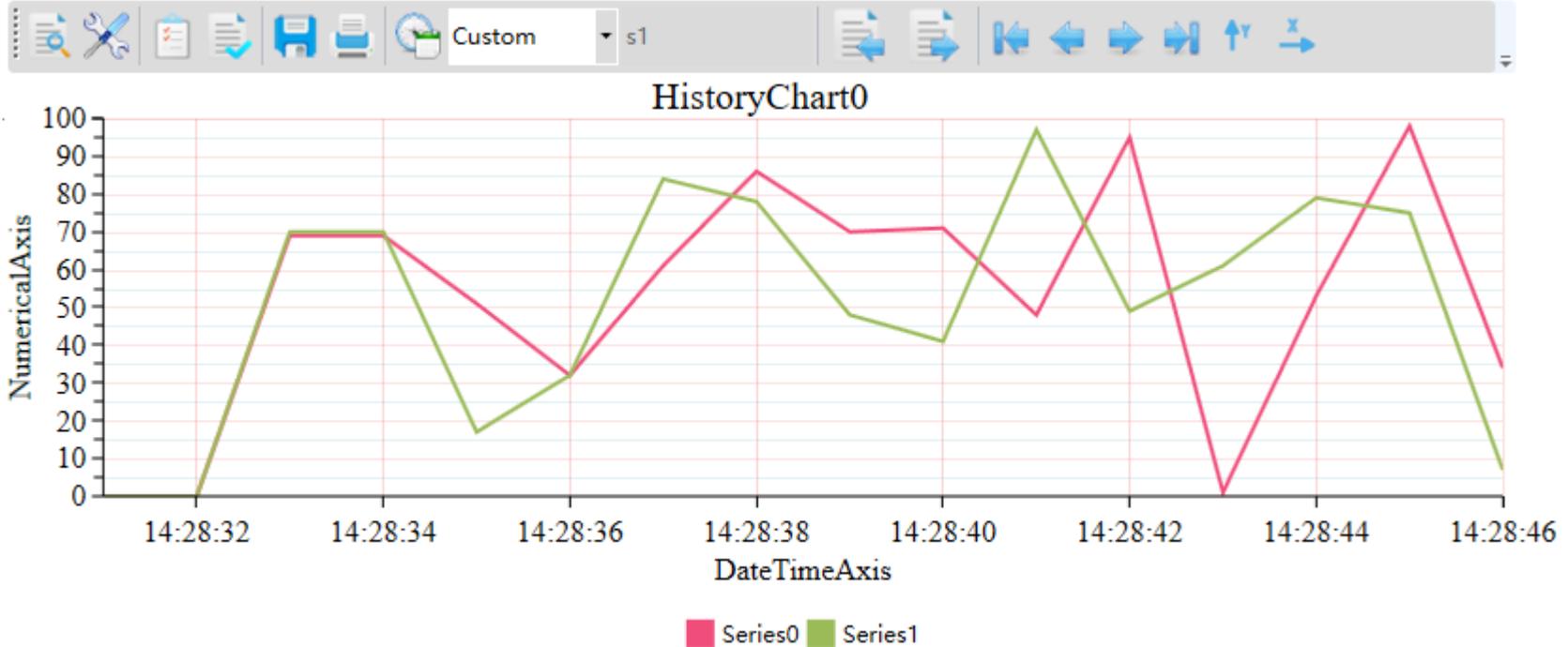


The screenshot displays a software interface with a script editor and an event configuration panel. The script editor contains the following code:

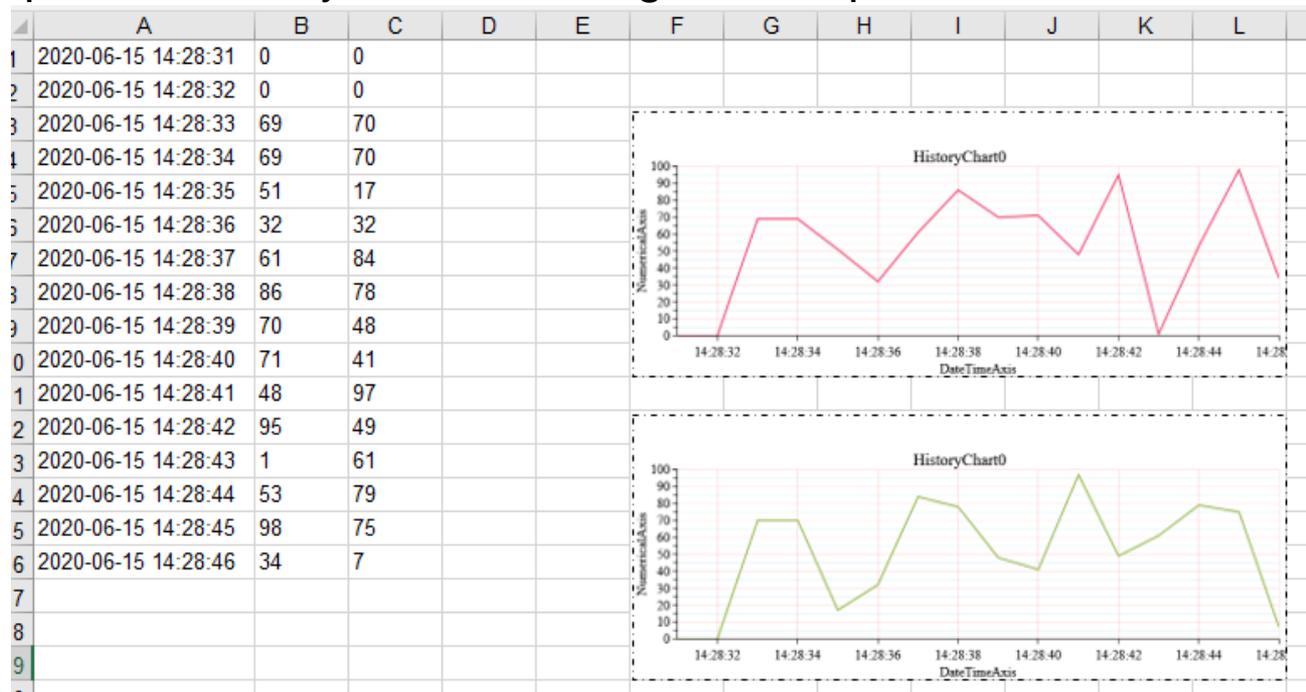
```
1 dt = HistoryChart0.GetDataTable("Series0") ' Get "series0" data of the HistoryChart0
2 dt1 = HistoryChart0.GetDataTable("Series1")
3 Call DbAccessCmd.SetPrimary(dt,"TriggerTime") 'Set "TriggerTime" as the primary key
4 Call DbAccessCmd.SetPrimary(dt1,"TriggerTime")
5 call dt.Merge(dt1) 'Merge dt1 data into dt
6 call ReportCmd.ExportDataByTemplate("D:\HistoryDataAndImage.xls", "Report0", dt, HistoryChart0.ArrayToString("Series0") & "|" & HistoryChart0.ArrayToString("Series1"))
7 'Export two curves in the same history chart to an excel file
```

The event configuration panel shows the 'Left Button' event selected for the 'LeftButtonDown' event. A red arrow points from the 'LeftButtonDown' event in the panel to the script editor.

(10) Run the Window0, query history data



(11)Export the history data and image, the exported excel file is shown below



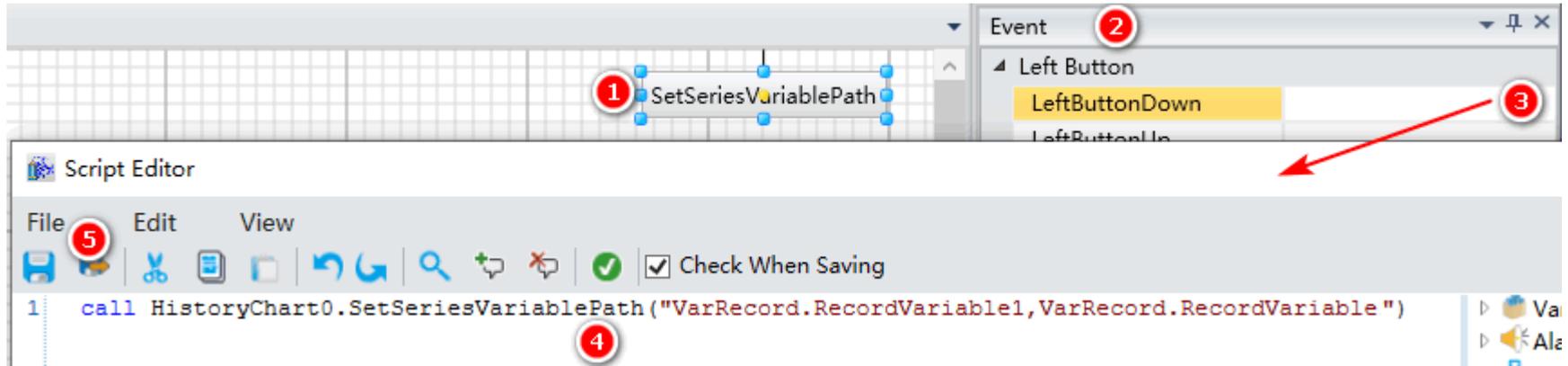
Click the “ArrayToImage” button, then Series0 and Series1 of HistoryChart0 are exported to the disk D successfully, and the file name is “HistoryDataAndImage”

➤ SetSeriesVariablePath example :

Exchange history variables associated with the two curves

The first 5 steps are the same as (1)(2)(3)(4)(5) steps of QueryHistoryDate example

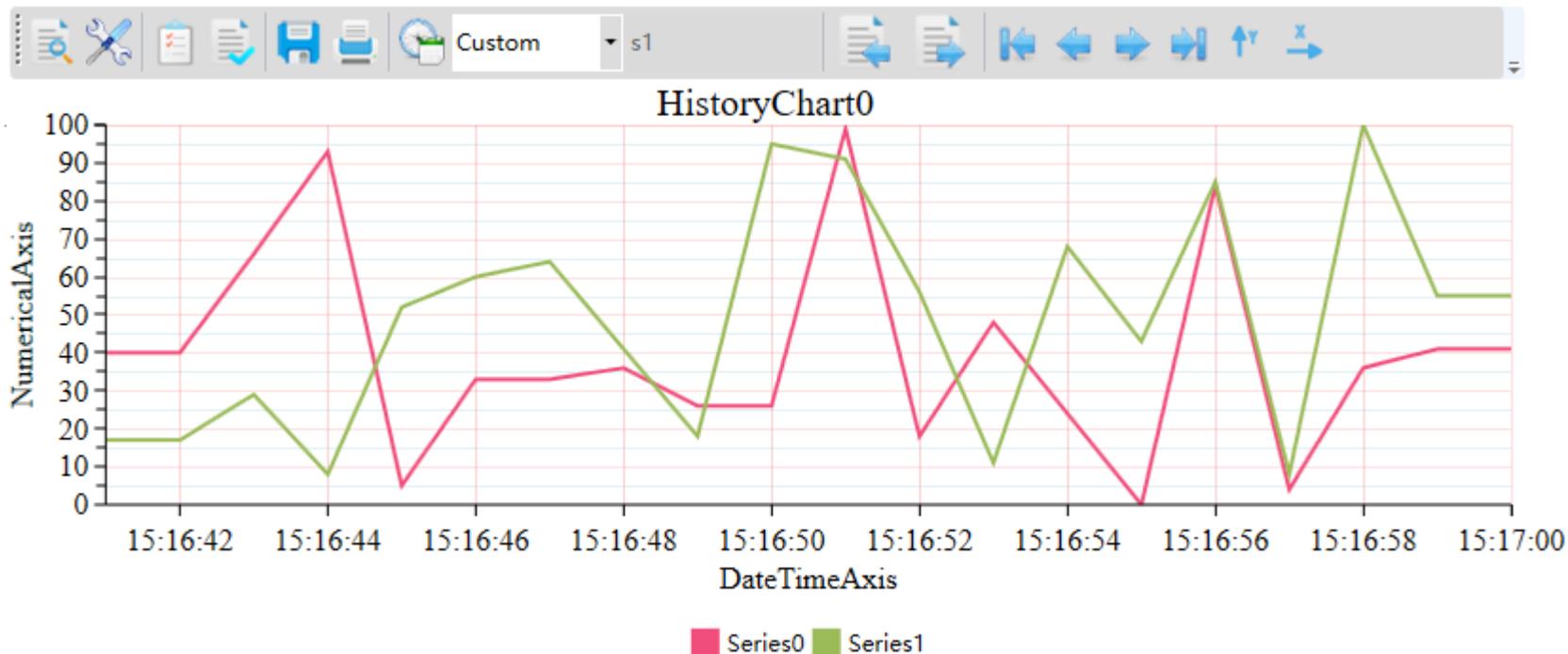
(6) Create one button (SetSeriesVariablePath) in the Window0, configure the LeftButtonDown event of the button



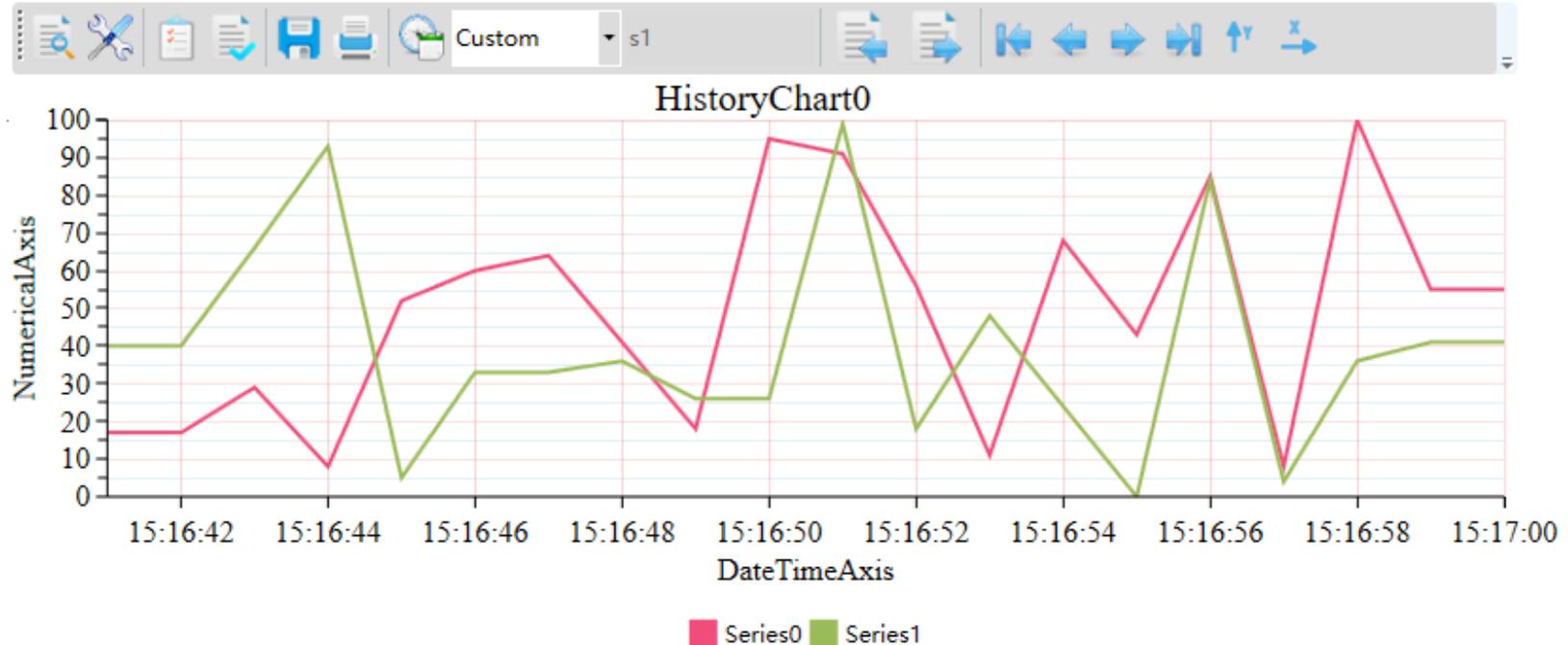
The screenshot displays the configuration of a button in the HistoryChart application. The interface is divided into several sections:

- Grid:** A button labeled "SetSeriesVariablePath" is placed on a grid. A red circle with the number "1" is positioned over the button.
- Event List:** A panel titled "Event" (2) shows a list of events for the "Left Button". The "LeftButtonDown" event is highlighted in yellow, and a red circle with the number "3" is positioned over it. A red arrow points from this event to the script editor.
- Script Editor:** A window titled "Script Editor" (5) contains a menu bar with "File", "Edit", and "View". Below the menu bar is a toolbar with various icons. The script content shows a single line of code: `1 call HistoryChart0.SetSeriesVariablePath ("VarRecord.RecordVariable1, VarRecord.RecordVariable ")`. A red circle with the number "4" is positioned over the code.

(7) Run the Window0,query history data



(8) Click the "SetSeriesVariablePath" button, then query history data again, as shown below



Comparing the image in the previous step, it can be concluded that the history variables associated with the two curves are exchanged.

➤ AddPoint, UpdatePoint, DeletePoint, DeletevalueAll example :

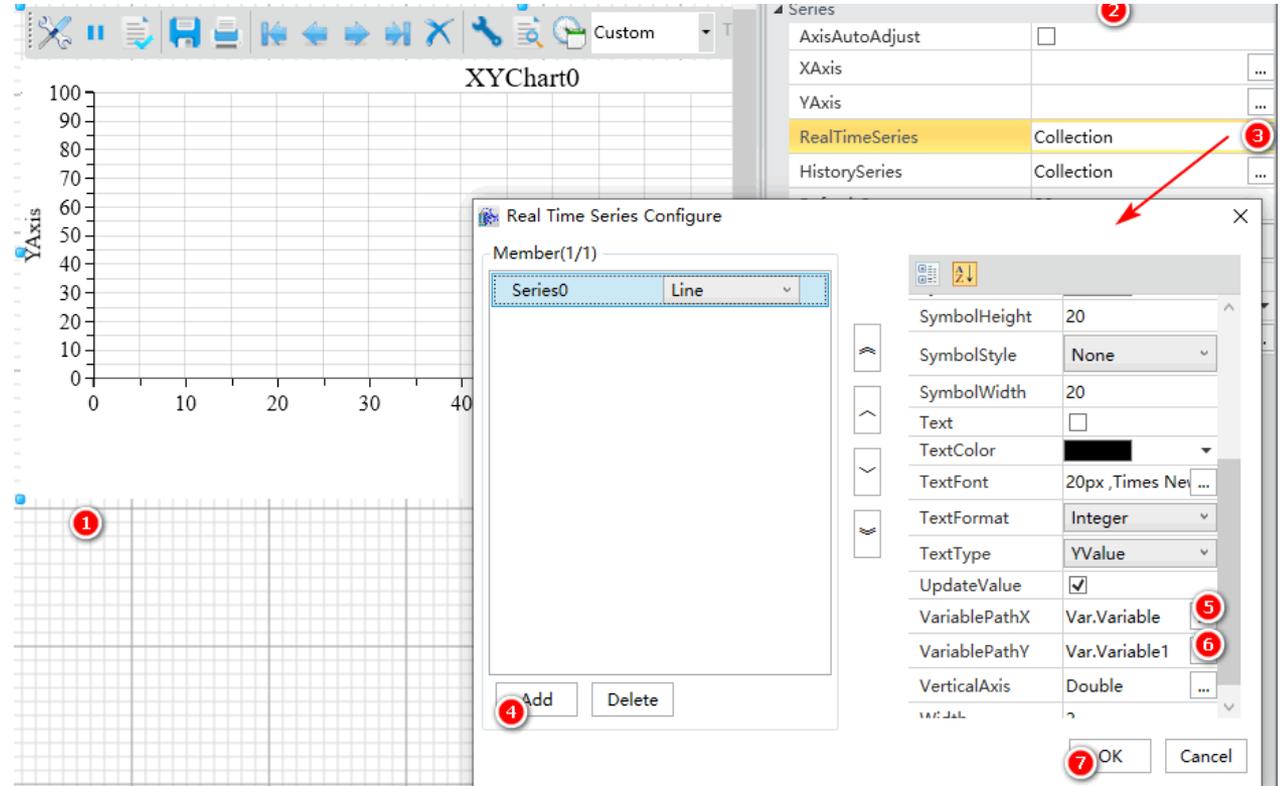
Add a point in the XYChart

(1) Create 2 variables : Variable , Variable1

(2) Create a simulated device : Device0

(3) Create two simulation address in the Device0 that associated with Variable, Variable1 respectively

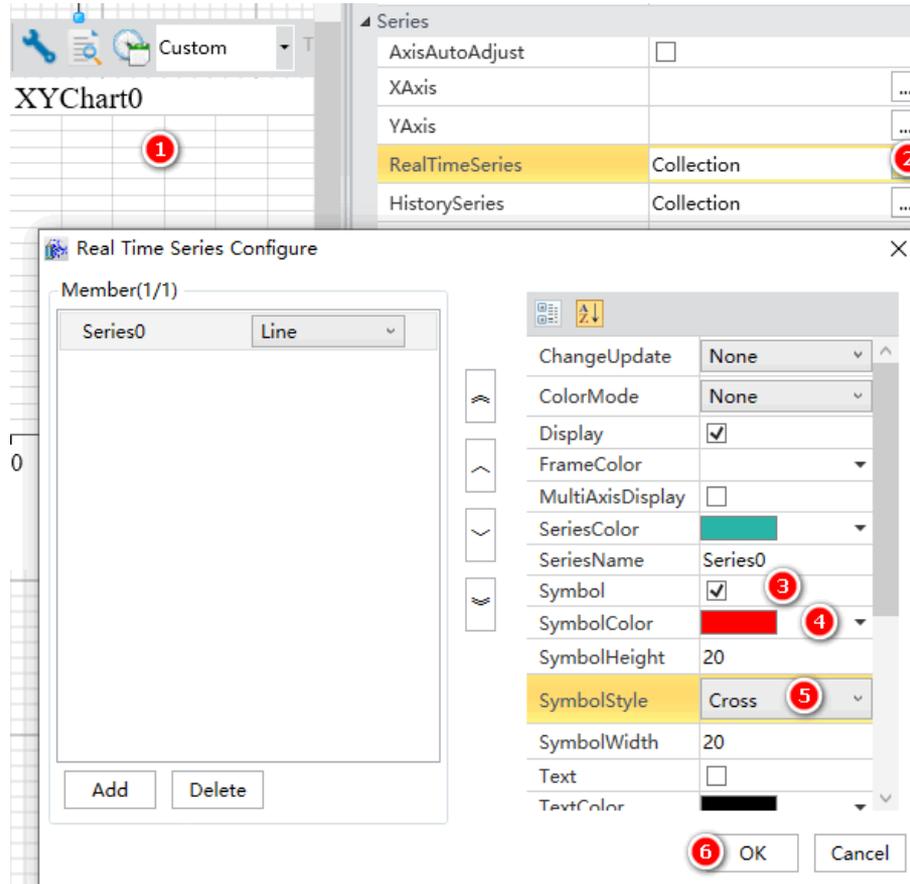
(4) Create a XYChart0 in the Window0, and add one curves in the XYChart0, VariablePathX of Series0 associated with Variable, VariablePathY of Series0 associated with Variable1



The screenshot displays the software interface for configuring an XYChart. The main window shows a grid with the Y-axis labeled 'Y Axis' ranging from 0 to 100 and the X-axis ranging from 0 to 40. A toolbar at the top contains various icons for chart manipulation. A 'Series' list on the right shows 'RealTimeSeries' selected. A 'Real Time Series Configure' dialog box is open, showing 'Member(1/1)' with 'Series0' and 'Line' selected. The dialog has several fields and buttons, with red circles and arrows indicating the following steps:

- 1: Clicking the 'Add' button at the bottom left of the dialog.
- 2: Clicking the 'Series' list on the right.
- 3: Clicking the 'RealTimeSeries' entry in the 'Series' list.
- 4: Clicking the 'Add' button at the bottom left of the dialog.
- 5: Clicking the 'VariablePathX' field.
- 6: Clicking the 'VariablePathY' field.
- 7: Clicking the 'OK' button at the bottom right of the dialog.

(5) Set display the symbols of Series0

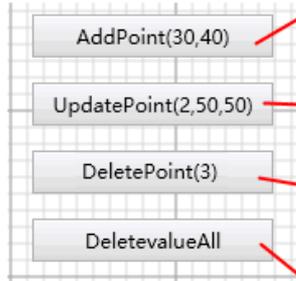


The screenshot shows the configuration interface for an XYChart. The main window displays a grid with the text "XYChart0" and a red circle with the number "1" indicating the chart area. A "Series" panel on the right lists "RealTimeSeries" and "HistorySeries", with "RealTimeSeries" selected and a red circle with the number "2" next to it. A "Real Time Series Configure" dialog box is open, showing the "Member(1/1)" section with "Series0" and "Line" selected. The dialog box has several settings, with a red circle with the number "3" next to the "Symbol" checkbox, a red circle with the number "4" next to the "SymbolColor" dropdown, and a red circle with the number "5" next to the "SymbolStyle" dropdown. The "SymbolStyle" is set to "Cross". At the bottom of the dialog box, there are "Add" and "Delete" buttons, and a "6" next to the "OK" button.

Property	Value
AxisAutoAdjust	<input type="checkbox"/>
XAxis	...
YAxis	...
RealTimeSeries	Collection
HistorySeries	Collection

Property	Value
ChangeUpdate	None
ColorMode	None
Display	<input checked="" type="checkbox"/>
FrameColor	...
MultiAxisDisplay	<input type="checkbox"/>
SeriesColor	...
SeriesName	Series0
Symbol	<input checked="" type="checkbox"/>
SymbolColor	...
SymbolHeight	20
SymbolStyle	Cross
SymbolWidth	20
Text	<input type="checkbox"/>
TextColor	...

(6) Create 4 buttons in the Window, configure the LeftButtonDown event of the 4 buttons



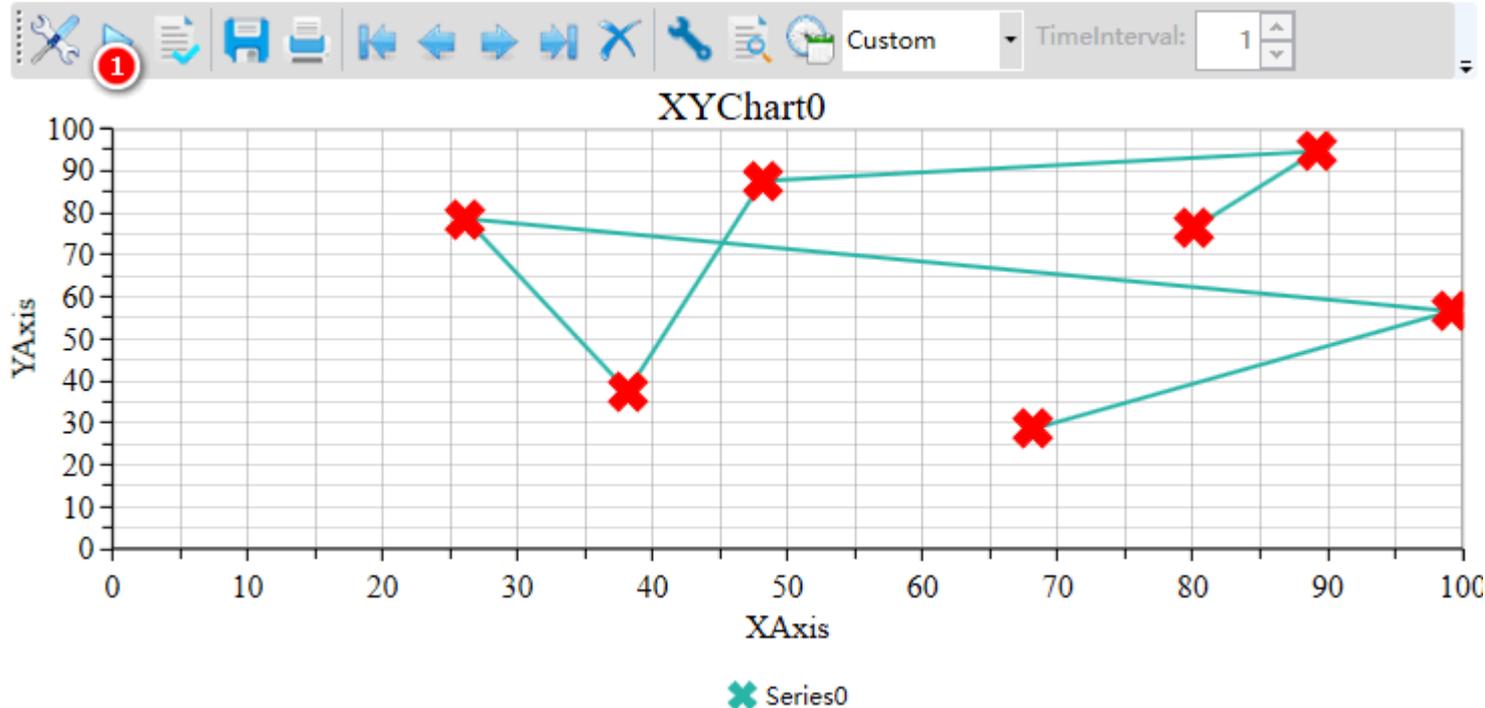
```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 |call XYChart0.AddPoint("Series0",30,40)
```

```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 | Call XYChart0.UpdatePoint("Series0",2,50,50)
```

```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 | call XYChart0.DeletePoint("Series0",3)
```

```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 |call XYChart0.DeletevalueAll()
```

(7) Run Window0 for a while, then press the pause button in XYChart0 to stop data refresh



(8) Execute scripts

- ① Click the "AddPoint(30,40)" button, Series0 will add a data point (30,40) at the end of itself
- ② Click the "UpdatePoint(2,50,50)" button, the data point with index 2 on Series0 will be updated to (50,50)
- ③ Click the "DeletePoint(3)" button, the data point with index 3 on Series0 will be deleted
- ④ Click the "DeletevalueAll" button, all the data points on the XYChart0 will be deleted

➤ SetDateConditionListStartTime example :

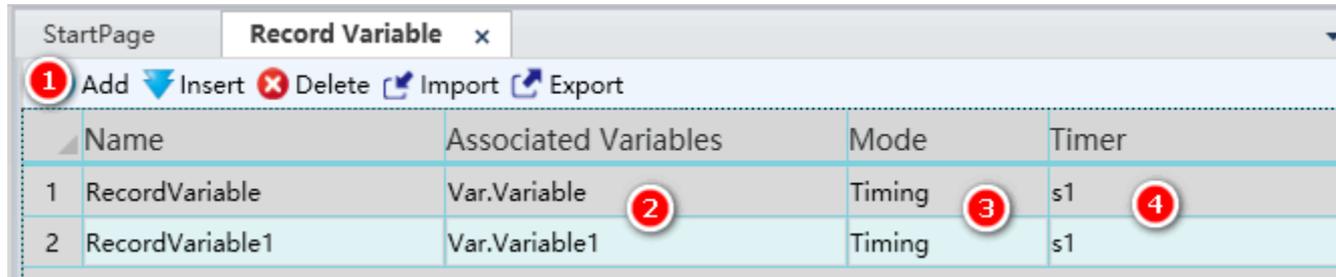
Query the minimum, average, and maximum data of two elements on the HistoryColumnChart over a period of time

(1) Create 2 variables : Variable , Variable1

(2) Create a simulated device : Device0

(3) Create two simulation address in the Device0 that associated with Variable, Variable1 respectively

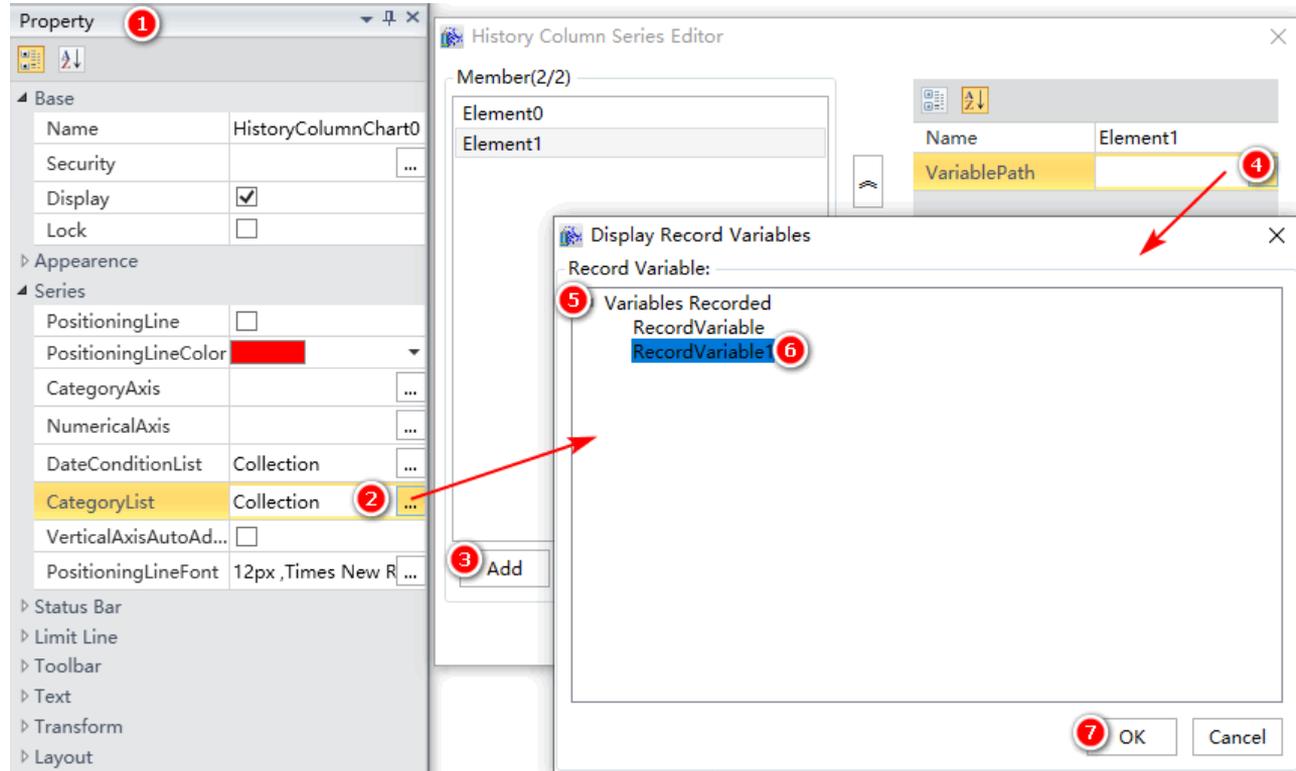
(4) Create two historical variables(RecordVariable, RecordVariable1) in the Record Variable node that associated with Variable, Variable1 respectively



	Name	Associated Variables	Mode	Timer
1	RecordVariable	Var.Variable 2	Timing 3	s1 4
2	RecordVariable1	Var.Variable1	Timing	s1

The Scripts of HistoryColumnChart

(5) Create a HistoryColumnChart0 in the Window0, and add 2 elements in the HistoryColumnChart0, Element0 associated RecordVariable, Element1 associated RecordVariable1



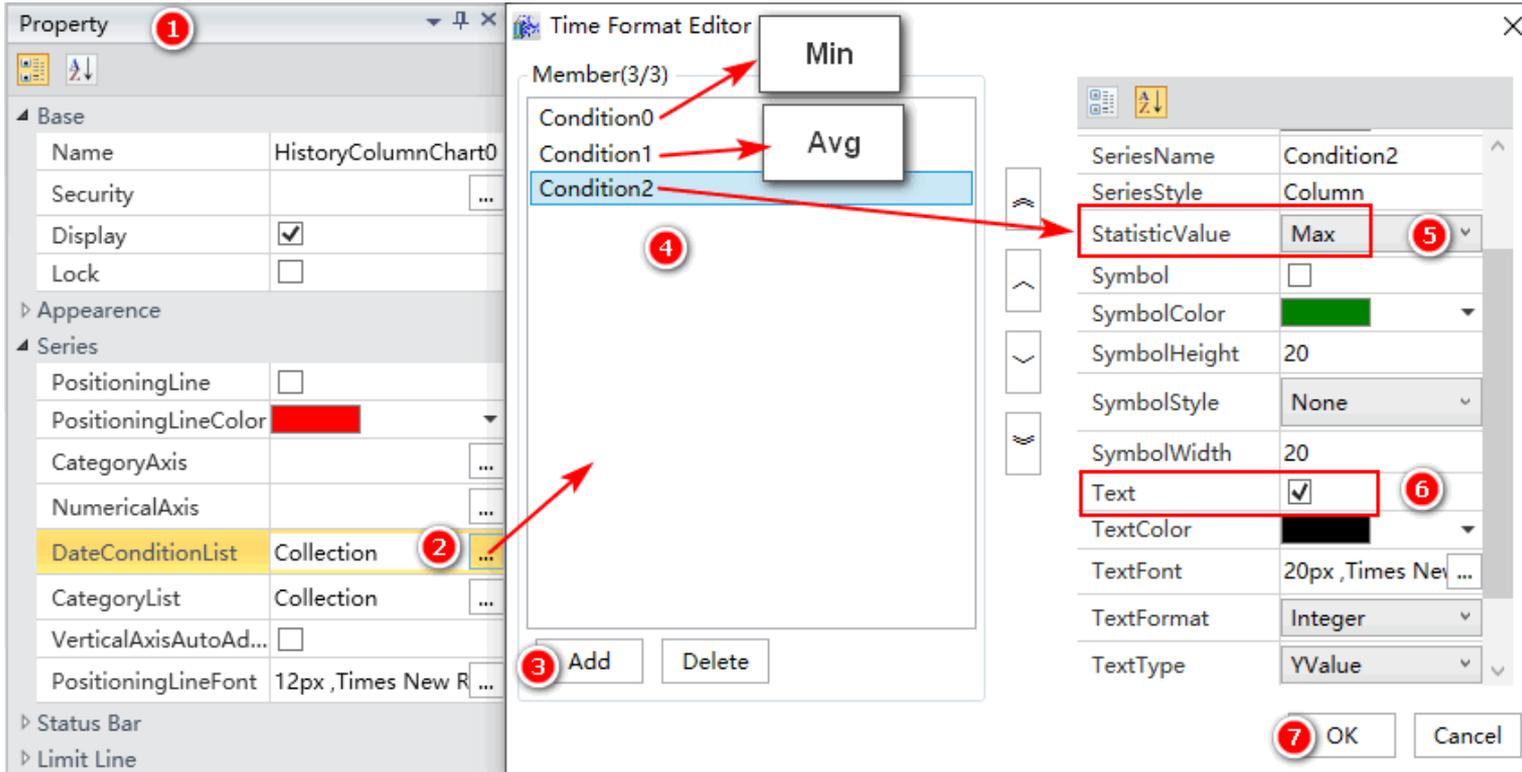
The screenshot illustrates the configuration steps for a HistoryColumnChart. It consists of three main windows:

- Property Window:** Shows the configuration for 'HistoryColumnChart0'. The 'CategoryList' property is highlighted in yellow and has a red circle '2' next to it. A red circle '1' is next to the window title.
- History Column Series Editor:** Shows the configuration for 'Element1'. The 'VariablePath' property is highlighted in yellow and has a red circle '4' next to it. A red circle '3' is next to the 'Add' button.
- Display Record Variables Dialog:** Shows the configuration for 'RecordVariable1'. The 'RecordVariable1' entry is highlighted in blue and has a red circle '6' next to it. A red circle '5' is next to the 'Variables Recorded' label. A red circle '7' is next to the 'OK' button.

Red arrows indicate the flow of configuration: from the 'CategoryList' property in the Property window to the 'Add' button in the History Column Series Editor, and from the 'Add' button to the 'Display Record Variables' dialog.

(6) Set conditions properties

The StatisticValue property of Condition0, Condition1, Condition2 are the Min, Avg, Max respectively



The screenshot displays two windows from a software application. The left window, titled 'Property', shows the configuration for 'HistoryColumnChart0'. The right window, titled 'Time Format Editor', shows the configuration for 'Condition2'.

Property Window (Left):

- 1. The 'Property' window title bar is circled in red.
- 2. The 'DateConditionList' property is highlighted in yellow, and its value 'Collection' is circled in red.

Time Format Editor Window (Right):

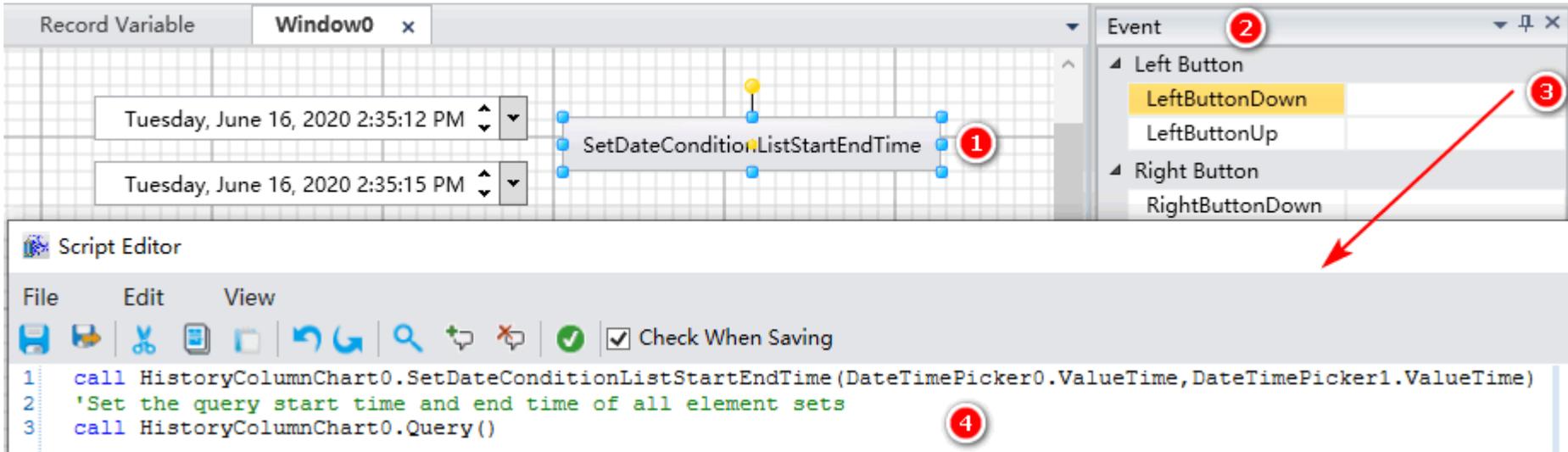
- 3. The 'Add' button is circled in red.
- 4. The 'Condition2' row in the list is highlighted in blue, and the number '4' is circled in red.
- 5. The 'StatisticValue' property is set to 'Max', and this value is circled in red.
- 6. The 'Text' property is checked, and this checkbox is circled in red.
- 7. The 'OK' button is circled in red.

Annotations:

- Red arrows point from 'Condition0' to 'Min', 'Condition1' to 'Avg', and 'Condition2' to 'Max' in the 'Time Format Editor' window.
- A red arrow points from the 'DateConditionList' property in the 'Property' window to the 'Condition2' row in the 'Time Format Editor' window.

The Scripts of HistoryColumnChart

(7) Create two DateTimePicker(DateTimePicker0, DateTimePicker1) and a button in the Window0, configure the LeftButtonDown event of the button



Record Variable Window0 x

Tuesday, June 16, 2020 2:35:12 PM

Tuesday, June 16, 2020 2:35:15 PM

SetDateConditionListStartEndTime

Event

- Left Button
 - LeftButtonDown
 - LeftButtonUp
- Right Button
 - RightButtonDown

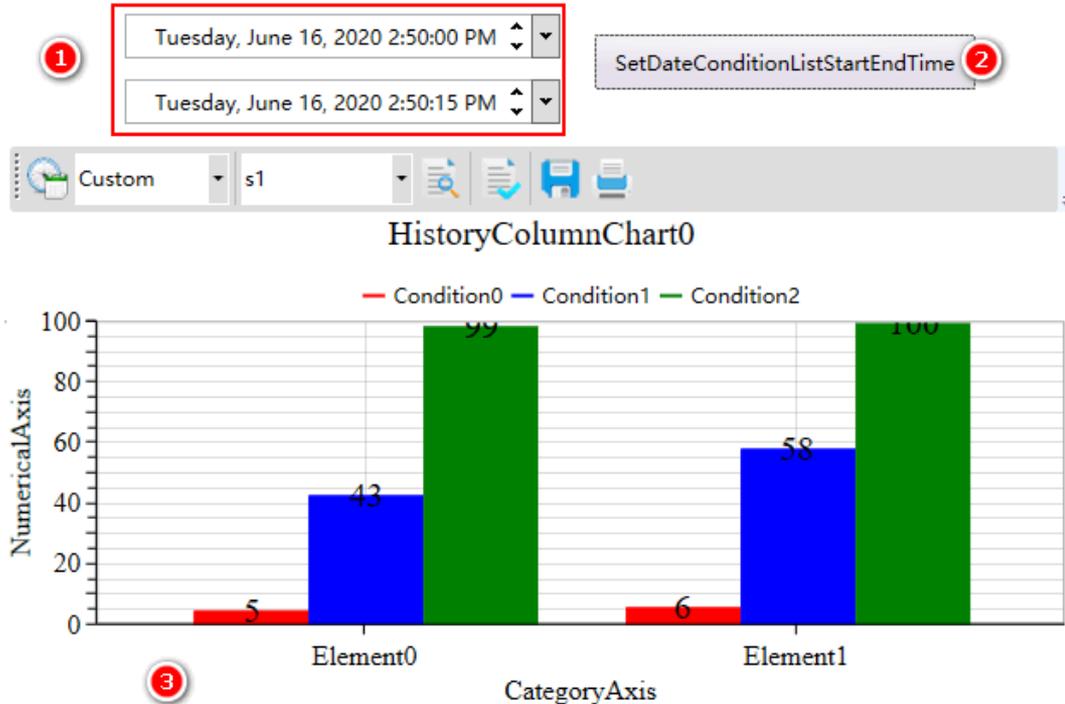
Script Editor

File Edit View

Check When Saving

```
1 call HistoryColumnChart0.SetDateConditionListStartEndTime(DateTimePicker0.ValueTime, DateTimePicker1.ValueTime)
2 'Set the query start time and end time of all element sets
3 call HistoryColumnChart0.Query()
```

(8) Run the Window0



① Set the query start time and end time

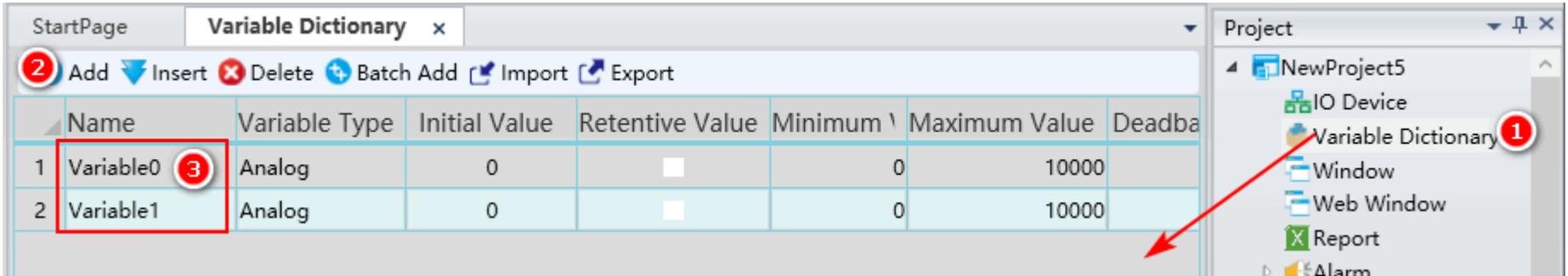
② Apply the set query period to the HistoryColumnChart0 , then query the data

③ Query result: The minimum, average and maximum values of Element0 are 5,43,99; the minimum, average and maximum values of Element1 are 6,58,100

➤ **GetCurrentAlarmColumnInfo** example :

Get the content of the row 2 and column 7 of the real-time alarm window

(1) Create 2 variables : Variable0 , Variable1



	Name	Variable Type	Initial Value	Retentive Value	Minimum \	Maximum Value	Deadba
1	Variable0	Analog	0	<input type="checkbox"/>	0	10000	
2	Variable1	Analog	0	<input type="checkbox"/>	0	10000	

※Refer to the section "6.3 Variables" in user manual.

(2) Create 2 alarm variables (AlarmVariable0, AlarmVariable1) that associated with the Variable0, Variable1 respectively

Name	Associated Variables	Alarm Level	Alarm Configuration	Description
1 AlarmVariable0	Var.Variable0	Slight	LowLow10	
2 AlarmVariable1	Var.Variable1	Serious		

Alarm Configuration

Common Alarm

Limit Value Alarm

Alarm Value	Alarm Text
<input type="checkbox"/> LowLow	10 LowLow
<input type="checkbox"/> Low	30 Low
<input type="checkbox"/> High	70 High
<input checked="" type="checkbox"/> HighHigh	90 HighHigh90
<input type="checkbox"/> Deadband	0

Deviation Alarm

Alarm Value	Alarm Text
<input type="checkbox"/> Major	80 Major
<input type="checkbox"/> Minor	20 Minor

TargetValue: 0

Deadband: 0

Rate of Change Alarm

Alarm Value	Alarm Text
<input type="checkbox"/> Rate	0 Rate

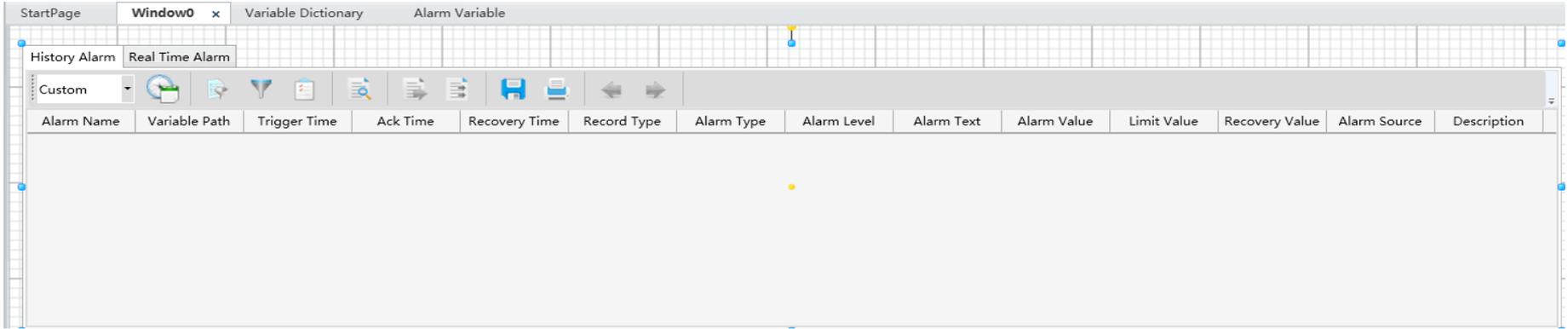
Type: Second

AlarmDelayTime: 0 Second

OK Cancel

※Refer to the section "11.3 Alarm variable" in user manual.

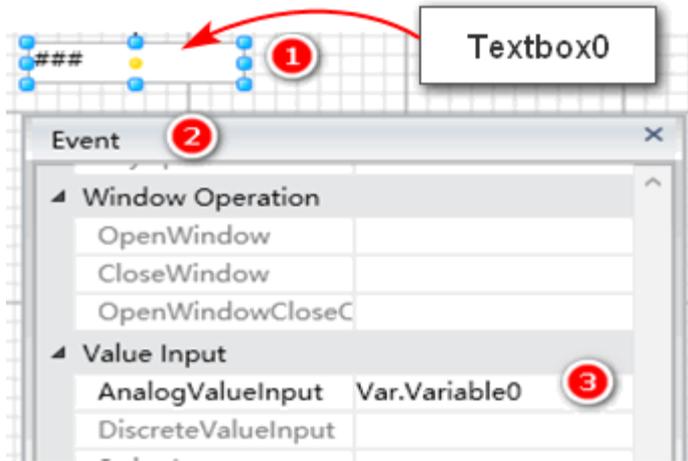
(3) Create a AlarmWindow0 in Window0



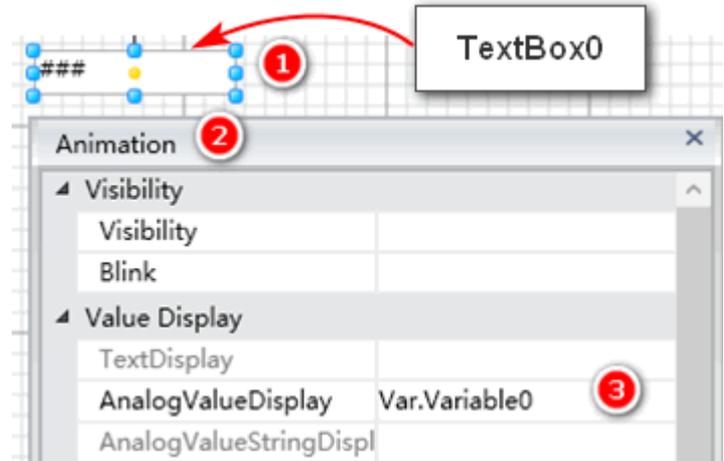
The screenshot shows the AlarmWindow configuration interface. The top bar includes tabs for 'StartPage', 'Window0 x', 'Variable Dictionary', and 'Alarm Variable'. Below the tabs, there are two sub-tabs: 'History Alarm' and 'Real Time Alarm'. A toolbar with various icons is visible, including a 'Custom' dropdown menu. The main area is a table with the following columns: Alarm Name, Variable Path, Trigger Time, Ack Time, Recovery Time, Record Type, Alarm Type, Alarm Level, Alarm Text, Alarm Value, Limit Value, Recovery Value, Alarm Source, and Description. The table is currently empty.

Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Value	Limit Value	Recovery Value	Alarm Source	Description
------------	---------------	--------------	----------	---------------	-------------	------------	-------------	------------	-------------	-------------	----------------	--------------	-------------

(4) Create 2 text box (Textbox0 , TextBox1) in Window0, the analog value input event and analog value display animation of Textbox0 , TextBox1 are associated with Variable0, Variable1 respectively

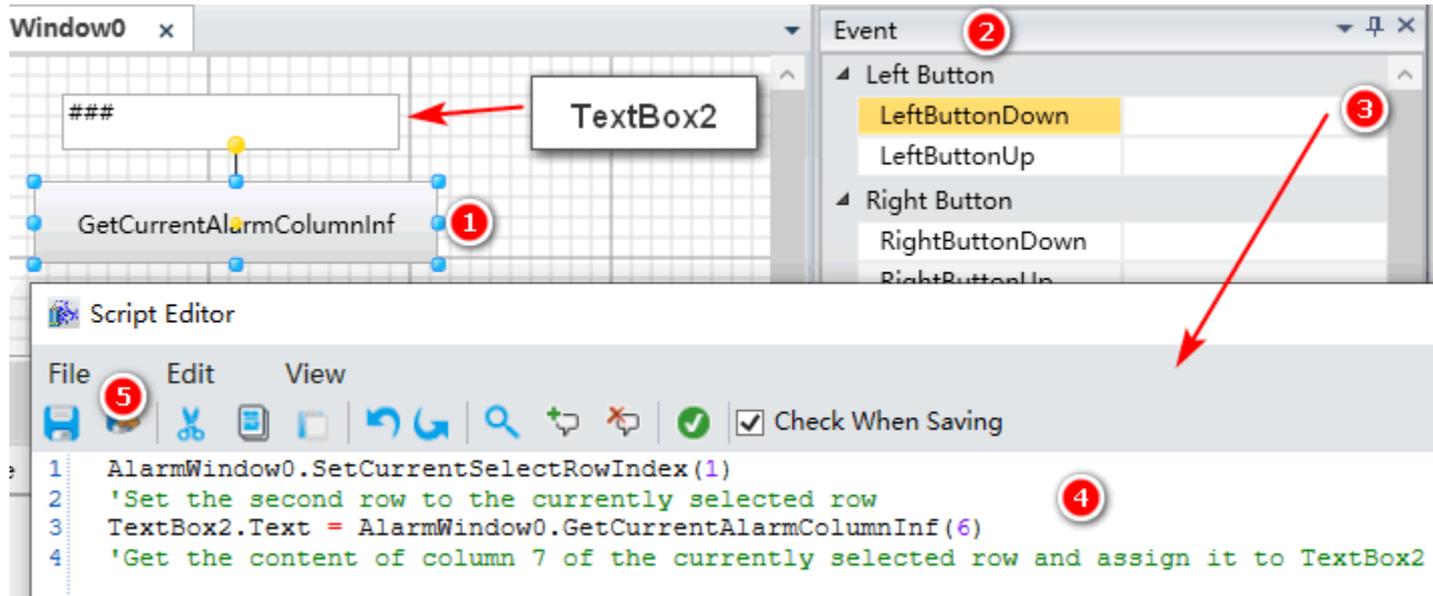


Analog Value Input



Analog Value Display

(5) Create a TextBox2 and a button(GetCurrentAlarmColumnInf) in the Window0, configure the LeftButtonDown event of the button

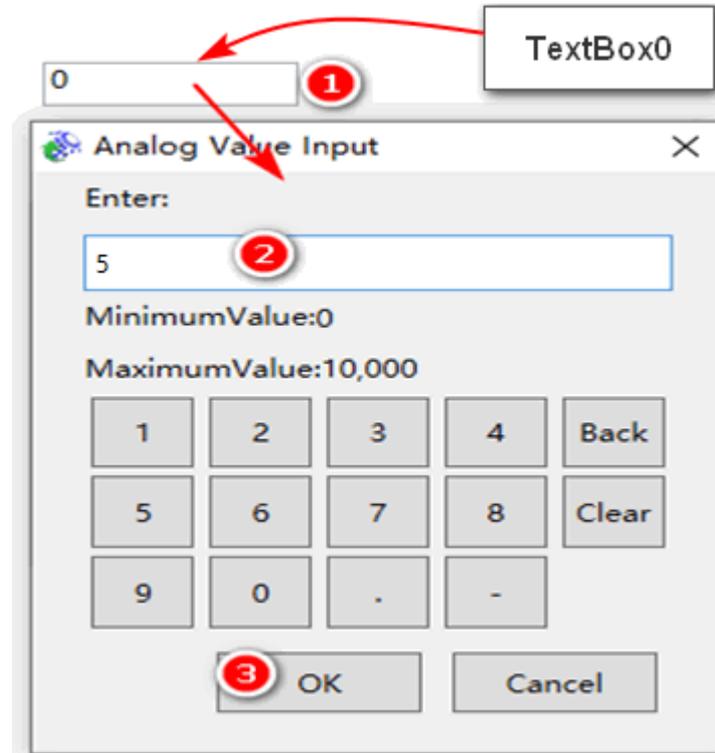


The screenshot displays the development environment for 'Window0'. In the design view, a 'GetCurrentAlarmColumnInf' button is connected to a 'TextBox2' control. The 'Event' viewer on the right shows the 'Left Button' section with 'LeftButtonDown' selected. The 'Script Editor' at the bottom contains the following code:

```
1 AlarmWindow0.SetCurrentSelectRowIndex(1)
2 'Set the second row to the currently selected row
3 TextBox2.Text = AlarmWindow0.GetCurrentAlarmColumnInf(6)
4 'Get the content of column 7 of the currently selected row and assign it to TextBox2
```

Red arrows and numbered callouts (1-5) highlight key elements: (1) the button, (2) the event viewer, (3) the selected event, (4) the script editor, and (5) the 'File' menu.

(6) Run the Window0. In Textbox0, input 3,13,5 in sequence;
In Textbox1, input 96,86,97 in sequence



(7)The Real Time Alarm window displays as follows

1

2

3

History Alarm

Real Time Alarm

4

Ack Selected ▾













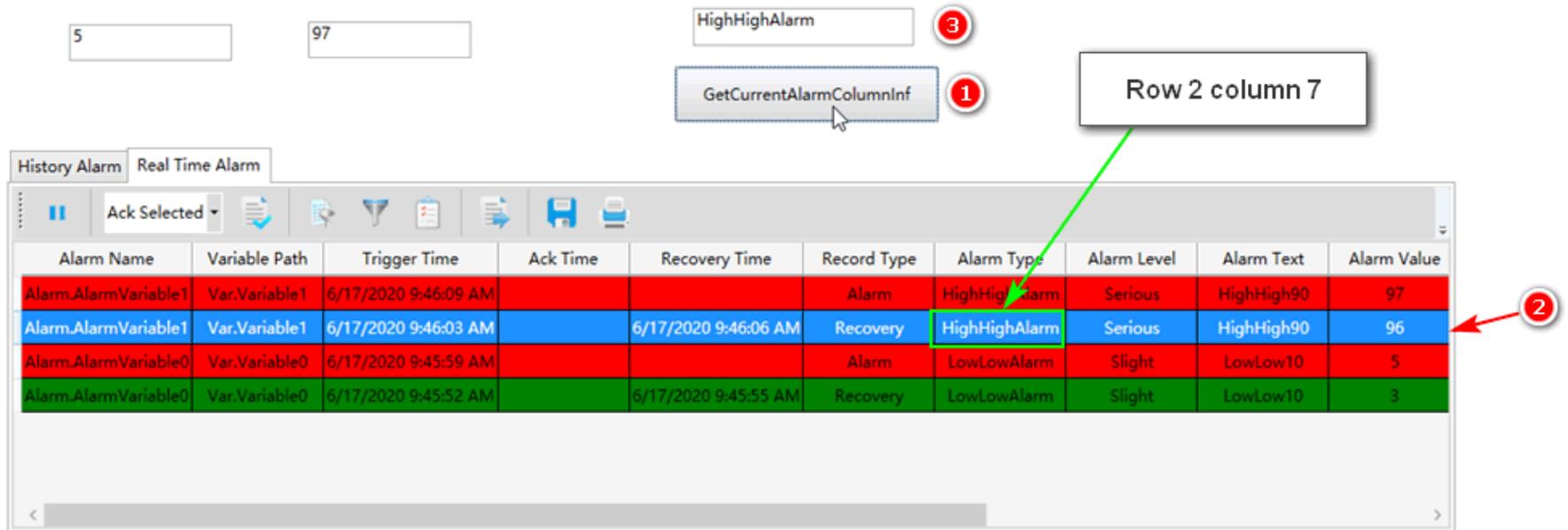


Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Valu
Alarm.AlarmVariable1	Var.Variable1	6/17/2020 10:24:31 AM			Alarm	HighHighAlarm	Serious	HighHigh90	97
Alarm.AlarmVariable1	Var.Variable1	6/17/2020 10:24:25 AM		6/17/2020 10:24:28 AM	Recovery	HighHighAlarm	Serious	HighHigh90	96
Alarm.AlarmVariable0	Var.Variable0	6/17/2020 10:24:22 AM			Alarm	LowLowAlarm	Slight	LowLow10	5
Alarm.AlarmVariable0	Var.Variable0	6/17/2020 10:24:16 AM		6/17/2020 10:24:19 AM	Recovery	LowLowAlarm	Slight	LowLow10	3

(8)Execute the scripts

①Click the “GetCurrentAlarmColumnInf” button

③The TextBox2 displays the content of column 7 of the currently selected row --HighHighAlarm



The screenshot shows the AlarmWindow interface. At the top, there are two text boxes containing the values '5' and '97'. Below them is a button labeled 'HighHighAlarm' with a red circle containing the number '3'. Below that is a button labeled 'GetCurrentAlarmColumnInf' with a red circle containing the number '1'. A green arrow points from a box labeled 'Row 2 column 7' to the 'HighHighAlarm' cell in the second row of the table. A red arrow points from a red circle containing the number '2' to the '96' value in the 'Alarm Value' column of the second row.

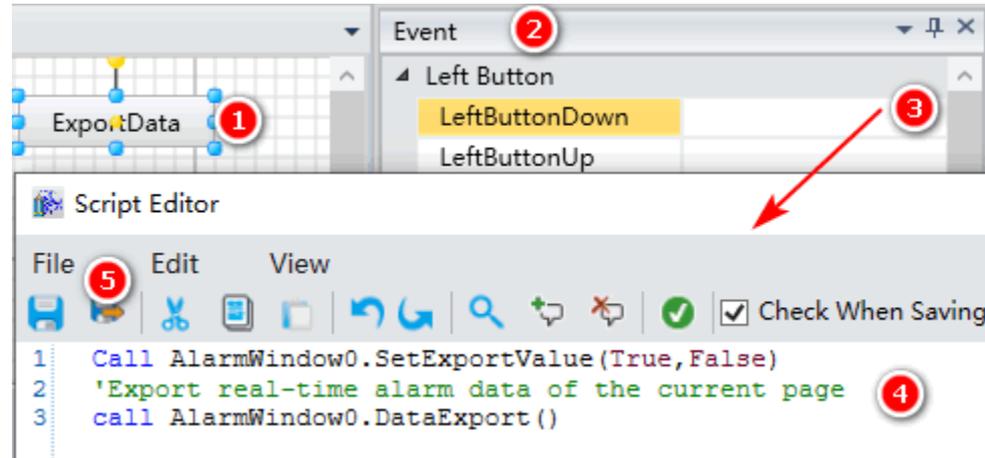
Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Value
Alarm.AlarmVariable1	Var.Variable1	6/17/2020 9:46:09 AM			Alarm	HighHighAlarm	Serious	HighHigh90	97
Alarm.AlarmVariable1	Var.Variable1	6/17/2020 9:46:03 AM		6/17/2020 9:46:06 AM	Recovery	HighHighAlarm	Serious	HighHigh90	96
Alarm.AlarmVariable0	Var.Variable0	6/17/2020 9:45:59 AM			Alarm	LowLowAlarm	Slight	LowLow10	5
Alarm.AlarmVariable0	Var.Variable0	6/17/2020 9:45:52 AM		6/17/2020 9:45:55 AM	Recovery	LowLowAlarm	Slight	LowLow10	3

➤ ExportData example1 :

Export real-time alarm data of the current page

The first 4 steps are the same as(1)(2)(3)(4) steps of GetCurrentAlarmColumnInf example

(5)Create a button(ExportData) in the Window0, configure the LeftButtonDown event of the button

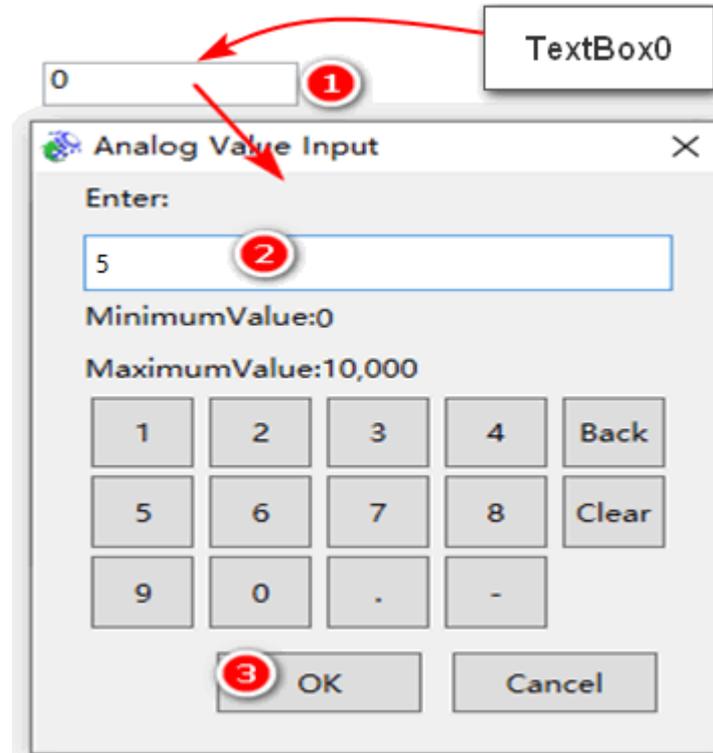


The screenshot displays the configuration interface for the AlarmWindow. The top panel shows the Event Editor for the 'Left Button' control. The 'LeftButtonDown' event is selected and highlighted in yellow. The bottom panel shows the Script Editor with the following code:

```
1 Call AlarmWindow0.SetExportValue(True, False)
2 'Export real-time alarm data of the current page
3 call AlarmWindow0.DataExport ()
```

Red circles with numbers 1 through 5 are overlaid on the image to indicate the steps: 1 points to the 'ExportData' button in the control tree; 2 points to the 'Event' tab; 3 points to the 'LeftButtonDown' event; 4 points to the script code; and 5 points to the 'File' menu in the Script Editor.

(6) Run the Window0. In Textbox0, input 3,13,5 in sequence;
In Textbox1, input 96,86,97 in sequence

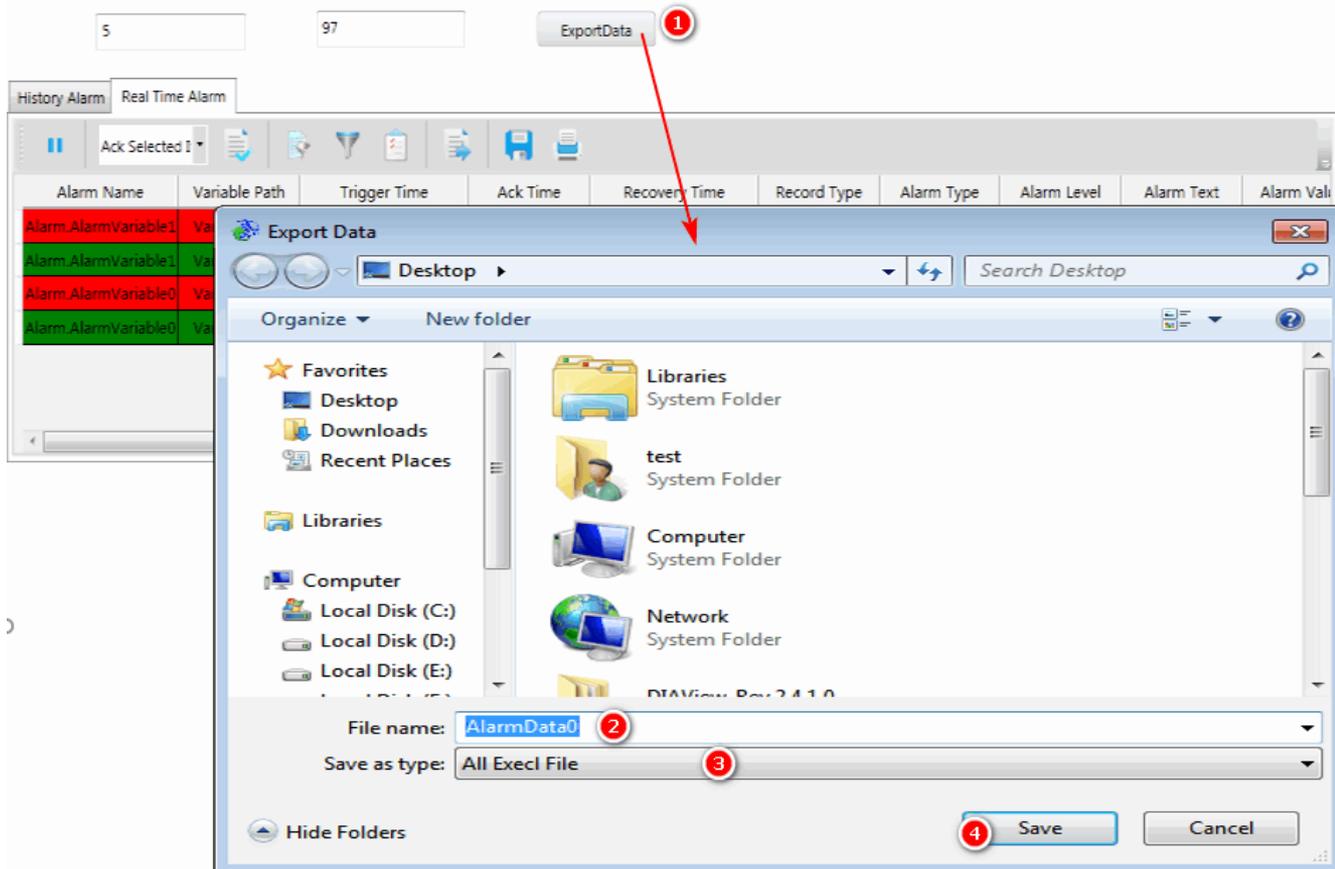


(7)The Real Time Alarm window displays as follows

History Alarm | Real Time Alarm

Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Value
Alarm.AlarmVariable1	Var.Variable1	6/17/2020 11:58:34 AM			Alarm	HighHighAlarm	Serious	HighHigh90	97
Alarm.AlarmVariable1	Var.Variable1	6/17/2020 11:58:28 AM		6/17/2020 11:58:31 AM	Recovery	HighHighAlarm	Serious	HighHigh90	96
Alarm.AlarmVariable0	Var.Variable0	6/17/2020 11:58:23 AM			Alarm	LowLowAlarm	Slight	LowLow10	5
Alarm.AlarmVariable0	Var.Variable0	6/17/2020 11:58:18 AM		6/17/2020 11:58:21 AM	Recovery	LowLowAlarm	Slight	LowLow10	3

(8) Execute the scripts. Click the “ExportData” button to export alarm data to excel file





The Scripts of AlarmWindow

(9)The excel file exported displays as follows

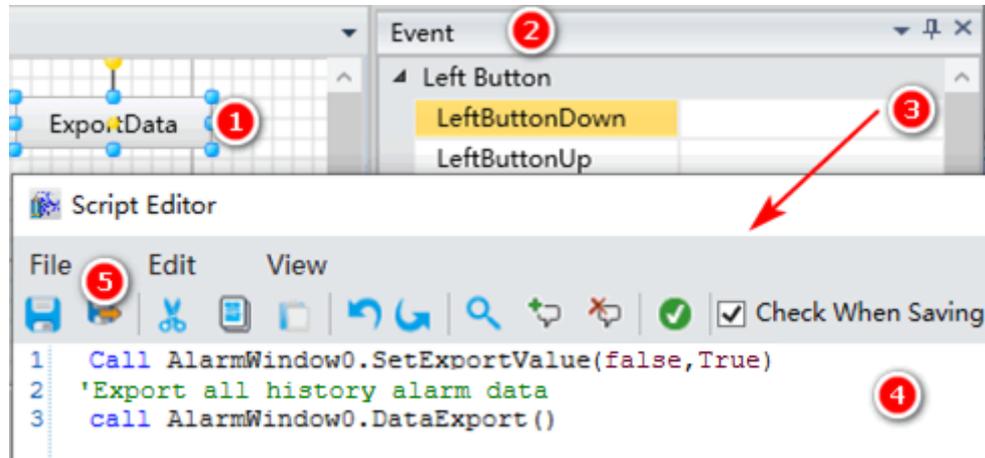
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Value	Limit Value	Current Value	Recovery Value	Alarm Source	Description
2	Alarm.AlarmVariable1	Var.Variable1	2020-06-17 11:55:35			Alarm	HighHighAlarm	Serious	HighHigh90	97	90	97		test-PC	
3	Alarm.AlarmVariable1	Var.Variable1	2020-06-17 11:55:26		2020-06-17 11:55:30	Recovery	HighHighAlarm	Serious	HighHigh90	96	90	97	86	test-PC	
4	Alarm.AlarmVariable0	Var.Variable0	2020-06-17 11:55:19			Alarm	LowLowAlarm	Slight	LowLow10	5	10	5		test-PC	
5	Alarm.AlarmVariable0	Var.Variable0	2020-06-17 11:55:13		2020-06-17 11:55:17	Recovery	LowLowAlarm	Slight	LowLow10	3	10	5	13	test-PC	
6															

➤ **ExportData** example2 :

Export all history alarm data

The first 4 steps are the same as(1)(2)(3)(4) steps of **GetCurrentAlarmColumnInf** example

(5)Create a button(ExportData) in the Window0, configure the LeftButtonDown event of the button



- (6) Run the Window0. In Textbox0, input 3,13,5 in sequence;
 In Textbox1, input 96,86,97 in sequence
- (7) The History Alarm window displays as follows

3 history Alarm Real Time Alarm

NearestOne! 4 5 No.1Page

Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Value
1 AlarmVariable	Var.Variable0	6/18/2020 9:52:50 AM		6/18/2020 9:52:52 AM	Recovery	LowLowAlarm	Slight	LowLow10	3
2 AlarmVariable	Var.Variable0	6/18/2020 9:52:54 AM			Alarm	LowLowAlarm	Slight	LowLow10	5
3 AlarmVariable1	Var.Variable1	6/18/2020 9:53:03 AM		6/18/2020 9:53:07 AM	Recovery	HighHighAlarm	Serious	HighHigh90	96
4 AlarmVariable1	Var.Variable1	6/18/2020 9:53:12 AM			Alarm	HighHighAlarm	Serious	HighHigh90	97



The Scripts of AlarmWindow

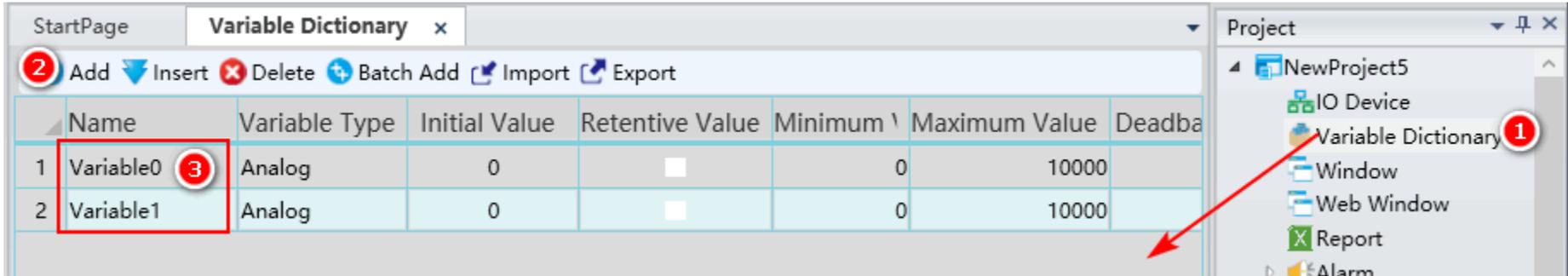
- (8) Execute the scripts. Click the “ExportData” button to export alarm data to excel file
- (9) The excel file exported displays as follows

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Alarm Name	Variable Path	Trigger Time	Ack Time	Recovery Time	Record Type	Alarm Type	Alarm Level	Alarm Text	Alarm Value	Limit Value	Recovery Value	Alarm Source	Description
2	Alarm.Alarm	Var.Variable0	2020-06-18 09:52:50		2020-06-18 09:52:52	Recovery	LowLowAlarm	Slight	LowLow10	3	10	13	CNWXJ6IAF	
3	Alarm.Alarm	Var.Variable0	2020-06-18 09:52:54			Alarm	LowLowAlarm	Slight	LowLow10	5	10		CNWXJ6IAF	
4	Alarm.Alarm	Var.Variable1	2020-06-18 09:53:03		2020-06-18 09:53:07	Recovery	HighHighAlarm	Serious	HighHigh90	96	90	86	CNWXJ6IAF	
5	Alarm.Alarm	Var.Variable1	2020-06-18 09:53:12			Alarm	HighHighAlarm	Serious	HighHigh90	97	90		CNWXJ6IAF	

➤ SetRealTimeVariableChange example :

Reports display real-time data

(1) Create 2 variables : Variable0 , Variable1



Name	Variable Type	Initial Value	Retentive Value	Minimum Value	Maximum Value	Deadband
1 Variable0	Analog	0	<input type="checkbox"/>	0	10000	
2 Variable1	Analog	0	<input type="checkbox"/>	0	10000	

※Refer to the section "6.3 Variables" in user manual.

(2) Create a simulated device : Device0

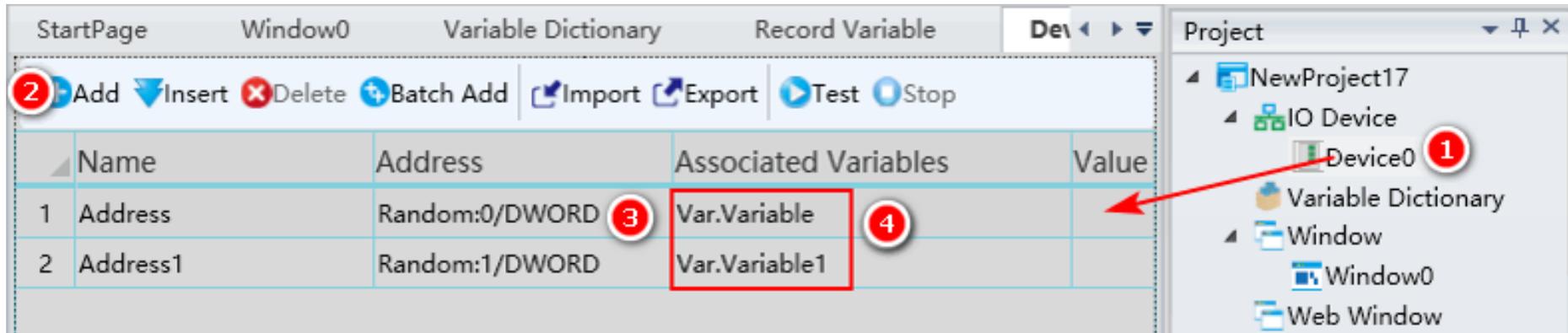
The image shows three sequential screenshots of the software interface, illustrating the steps to create a simulated device:

- Step 1:** A screenshot of the project tree with the 'IO Device' node right-clicked, opening a context menu. A red circle '1' is next to the right-click action.
- Step 2:** The context menu is open, and 'New Device' is highlighted. A red circle '2' is next to the 'New Device' option.
- Step 3:** The 'Driver Selection' dialog is shown with 'Simulator' selected in the list. A red circle '3' is next to the 'Simulator' entry.
- Step 4:** The 'Simulator' dialog is shown with 'Simulator' selected in the 'Register' list. A red circle '4' is next to the 'Simulator' entry.
- Step 5:** The 'Driver Selection' dialog is shown with the 'OK' button highlighted. A red circle '5' is next to the 'OK' button.
- Step 6:** The 'Simulator' dialog is shown with the 'Next' button highlighted. A red circle '6' is next to the 'Next' button.
- Step 7:** The 'Simulator' dialog is shown with the 'OK' button highlighted. A red circle '7' is next to the 'OK' button.

- ① Right click “IO Device” node
- ② Click “New Device”
- ③④ Double click “Simulator”

※ Refer to the section “5.10.1 Simulator” in user manual.

(3) Create two simulation address in the Device0 that associated with Variable, Variable1 respectively



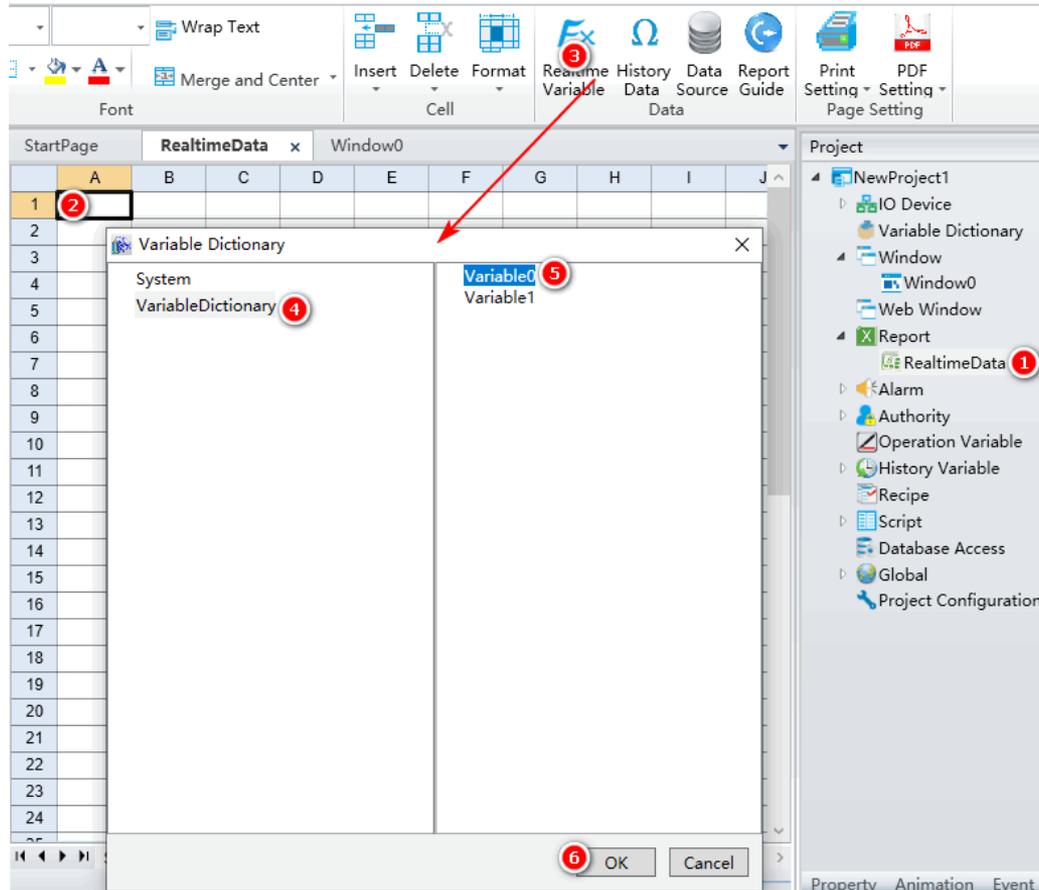
The screenshot displays the software interface with the 'Variable Dictionary' window active. The table below shows the configuration of simulation addresses and their associated variables. The 'Project' tree on the right shows the hierarchy of the project, with 'Device0' highlighted by a red circle and an arrow pointing to the 'Var.Variable' entry in the table.

Name	Address	Associated Variables	Value
1 Address	Random:0/DWORD	Var.Variable	
2 Address1	Random:1/DWORD	Var.Variable1	

The 'Project' tree on the right shows the following structure:

- NewProject17
 - IO Device
 - Device0 (1)
 - Variable Dictionary
 - Window
 - Window0
 - Web Window

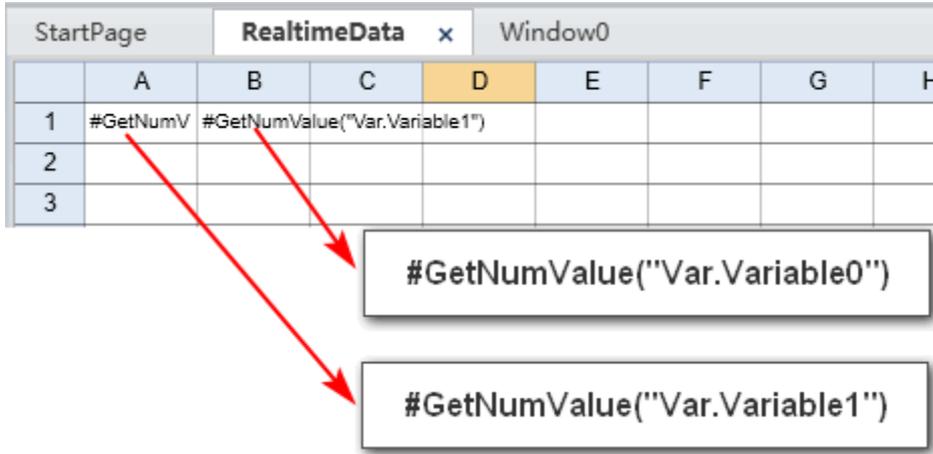
(4) Create a report template(RealtimeData) and configure variables for it



The screenshot displays the software interface for configuring a report template. The main window shows a spreadsheet with columns A through J and rows 1 through 24. The 'RealtimeData' report template is selected in the 'Project' tree on the right. The 'Variable Dictionary' dialog is open, showing the 'System' dictionary with 'VariableDictionary' and 'Variable0' listed. The 'Realtime Variable' button in the ribbon is highlighted with a red circle and arrow. The 'OK' button in the dialog is also highlighted with a red circle and arrow.

(5) The configuration result of RealtimeData is as follows

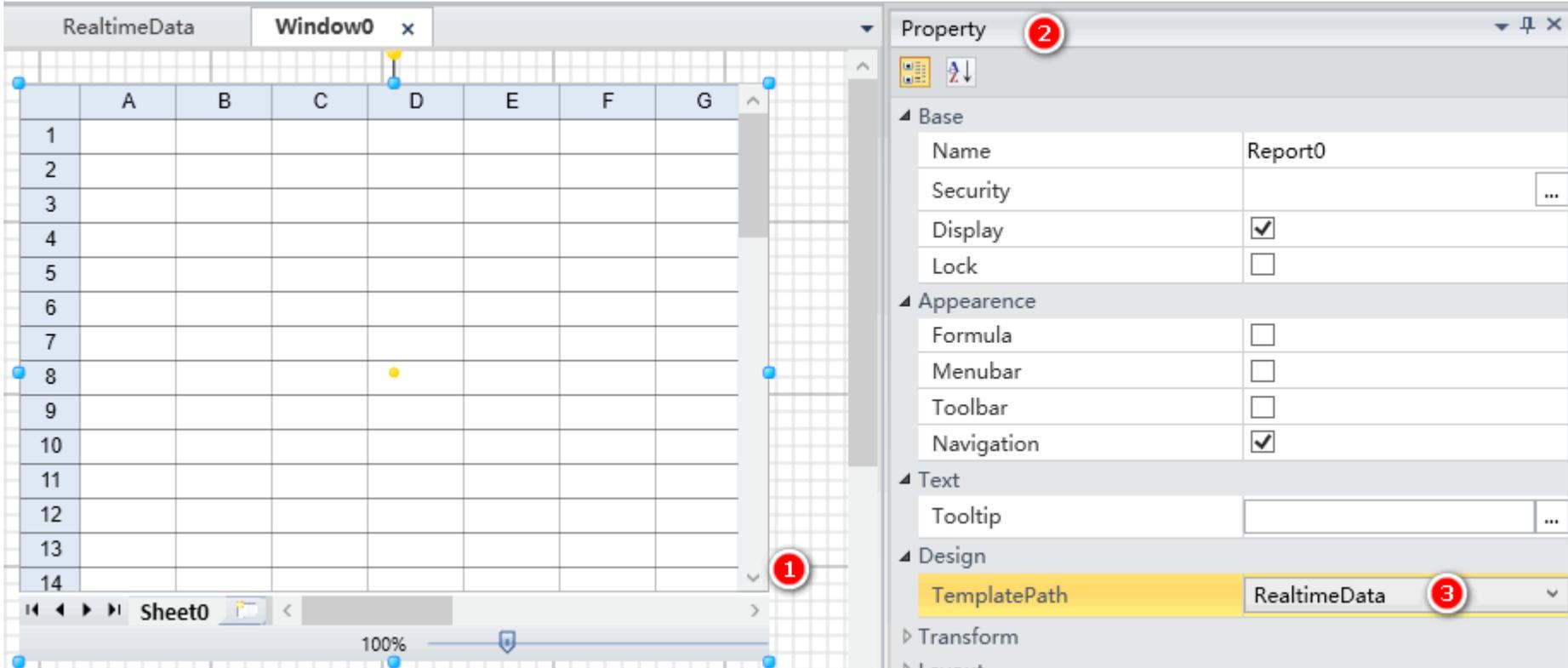
StartPage	RealtimeData	Window0						
	A	B	C	D	E	F	G	H
1	#GetNumV	#GetNumValue("Var.Variable1")						
2								
3								



#GetNumValue("Var.Variable0")

#GetNumValue("Var.Variable1")

(6) Create a Report0 in the Window0, and bound the RealtimeData report template

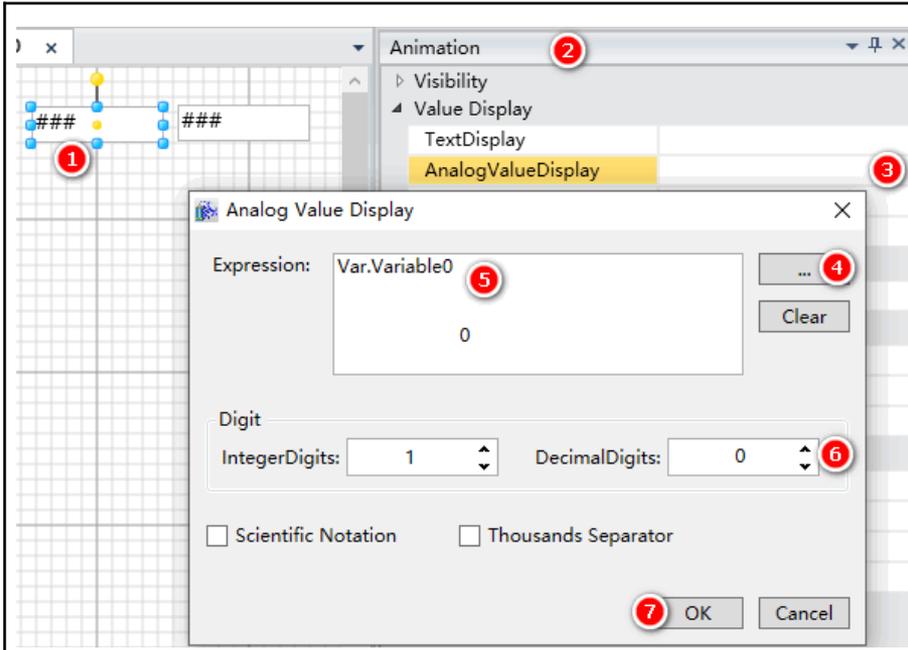


The screenshot displays a software interface with a spreadsheet and a property panel. The spreadsheet, titled 'Window0', shows columns A through G and rows 1 through 14. A yellow dot is located in cell D8. The property panel on the right is titled 'Property' and contains the following sections:

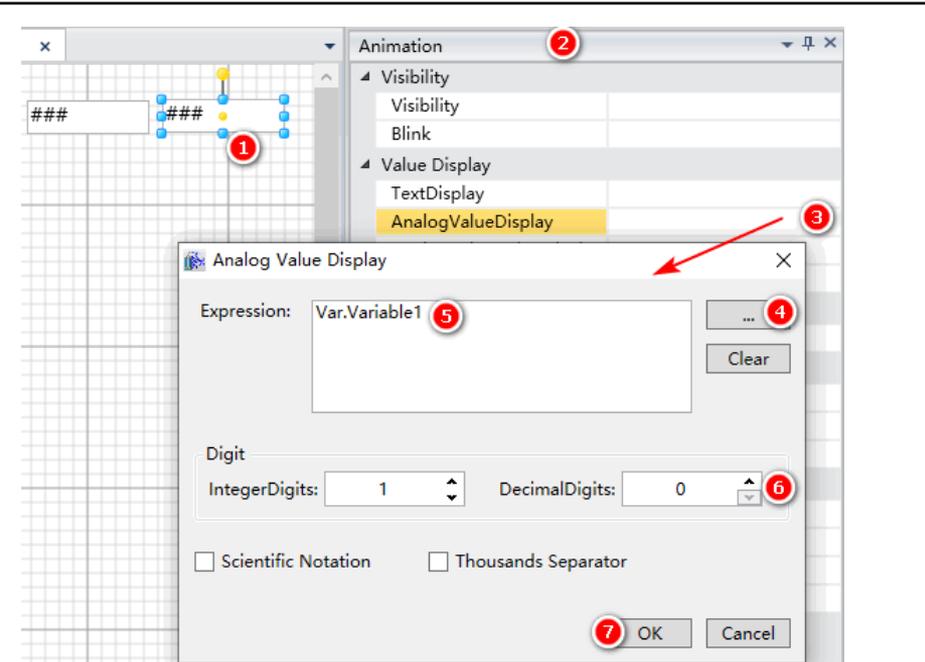
- Base**
 - Name: Report0
 - Security: [...]
 - Display:
 - Lock:
- Appearance**
 - Formula:
 - Menubar:
 - Toolbar:
 - Navigation:
- Text**
 - Tooltip: [...]
- Design**
 - TemplatePath: RealtimeData
- Transform**
- Layout**

Red circles with numbers 1, 2, and 3 highlight specific elements: 1 points to the spreadsheet grid, 2 points to the Property panel title, and 3 points to the TemplatePath dropdown.

(7) Create 2 text box (Textbox0 , TextBox1) in Window0, the analog value display animation of Textbox0 , TextBox1 are associated with Variable0 , Variable1 respectively

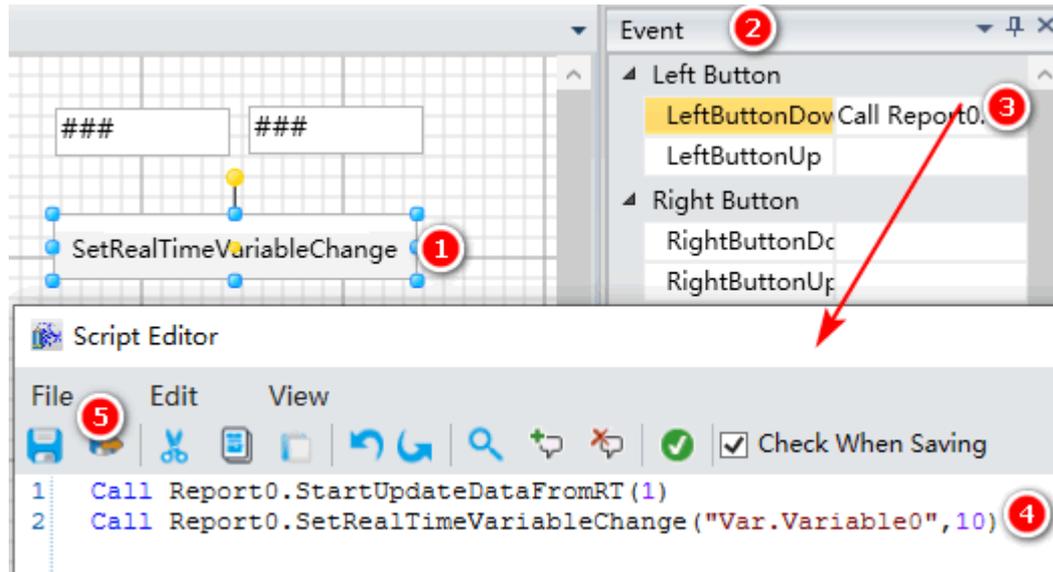


Analog Value Display



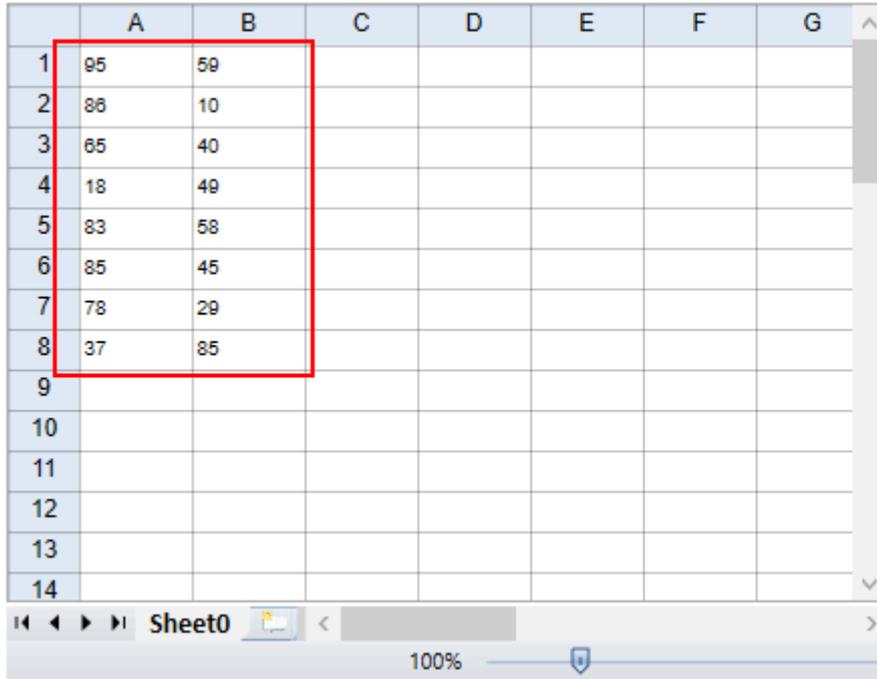
Analog Value Display

(8) Create a button (SetRealTimeVariableChange) in the Window0, configure the LeftButtonDown event of the button



The screenshot displays a software interface for configuring a button. The top section shows a grid with two '###' labels and a button labeled 'SetRealTimeVariableChange' (marked with a red circle 1). To the right, an 'Event' panel (marked with a red circle 2) lists events for 'Left Button' and 'Right Button'. The 'LeftButtonDown' event is selected and linked to 'Call Report0.' (marked with a red circle 3). A red arrow points from this event to the 'Script Editor' below. The 'Script Editor' has a menu bar with 'File' (marked with a red circle 5), 'Edit', and 'View'. The script contains two lines of code: `1 Call Report0.StartUpdateDataFromRT (1)` and `2 Call Report0.SetRealTimeVariableChange ("Var.Variable0", 10)` (marked with a red circle 4). A 'Check When Saving' checkbox is also visible.

(9) Run the Window0, click the “SetRealTimeVariableChange” button , then the first 10 rows of column A and column B of Report0 display the values of Variable0 and Variable 1 in real time



	A	B	C	D	E	F	G
1	95	59					
2	86	10					
3	65	40					
4	18	49					
5	83	58					
6	85	45					
7	78	29					
8	37	85					
9							
10							
11							
12							
13							
14							

37 85

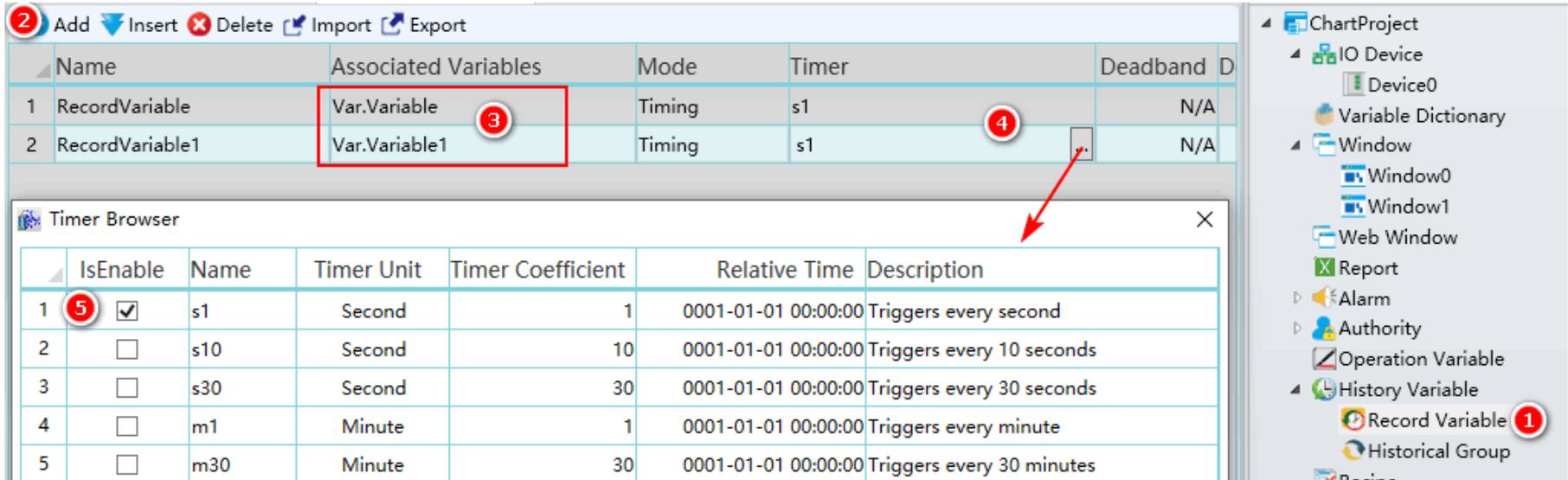
SetRealTimeVariableChange

➤ QueryHistoryData, QueryHistoryDataByCommon example :

Report query and display history data

The first 3 steps are the same as (1)(2)(3) steps of SetRealTimeVariableChange example

(4) Create two historical variables in the Record Variable that associated with Variable, Variable1 respectively



The screenshot shows the software interface with the following components:

- Top Bar:** Add, Insert, Delete, Import, Export
- Main Table:**

Name	Associated Variables	Mode	Timer	Deadband	D
1 RecordVariable	Var.Variable	Timing	s1	N/A	
2 RecordVariable1	Var.Variable1	Timing	s1	N/A	
- Timer Browser:**

IsEnable	Name	Timer Unit	Timer Coefficient	Relative Time	Description
<input checked="" type="checkbox"/>	s1	Second	1	0001-01-01 00:00:00	Triggers every second
<input type="checkbox"/>	s10	Second	10	0001-01-01 00:00:00	Triggers every 10 seconds
<input type="checkbox"/>	s30	Second	30	0001-01-01 00:00:00	Triggers every 30 seconds
<input type="checkbox"/>	m1	Minute	1	0001-01-01 00:00:00	Triggers every minute
<input type="checkbox"/>	m30	Minute	30	0001-01-01 00:00:00	Triggers every 30 minutes
- Right Panel:**
 - ChartProject
 - IO Device
 - Device0
 - Variable Dictionary
 - Window
 - Window0
 - Window1
 - Web Window
 - Report
 - Alarm
 - Authority
 - Operation Variable
 - History Variable
 - Record Variable
 - Historical Group

(5) Create a report template(HistoryData) and configure history variables for it

The screenshot displays the software interface with several key components and numbered callouts:

- 3**: Callout pointing to the **History Variable** icon in the top toolbar.
- 1**: Callout pointing to the **HistoryData** report template in the project tree on the right.
- 2**: Callout pointing to the selected cell (A1) in the spreadsheet.
- 4**: Callout pointing to the **TriggerTime** category in the **History Data** configuration panel.
- 5**: Callout pointing to the **DataPoint** input field in the **History Data** configuration panel.
- 6**: Callout pointing to the **HistoryVariables** section in the **History Variables** dialog box.
- 7**: Callout pointing to the **RecordVariable** section in the **History Variables** dialog box.
- 8**: Callout pointing to the **OK** button at the bottom right of the **History Variables** dialog box.
- 9**: Callout pointing to the **OK** button at the bottom left of the **History Data** configuration panel.

The **History Variables** dialog box shows the following configuration:

- HistoryVariables**: Historical Group
- RecordVariable**: RecordVariable1

The **History Data** configuration panel shows the following settings:

- DataPoint**: [Empty field]
- BaseLineTime**: [Empty field]
- Year**: Year
- Month**: Month
- Day**: Day
- Hour**: Hour
- Minute**: Minute
- Second**: Second
- Category**: TriggerTime

(6) The configuration result of HistoryData is as follows

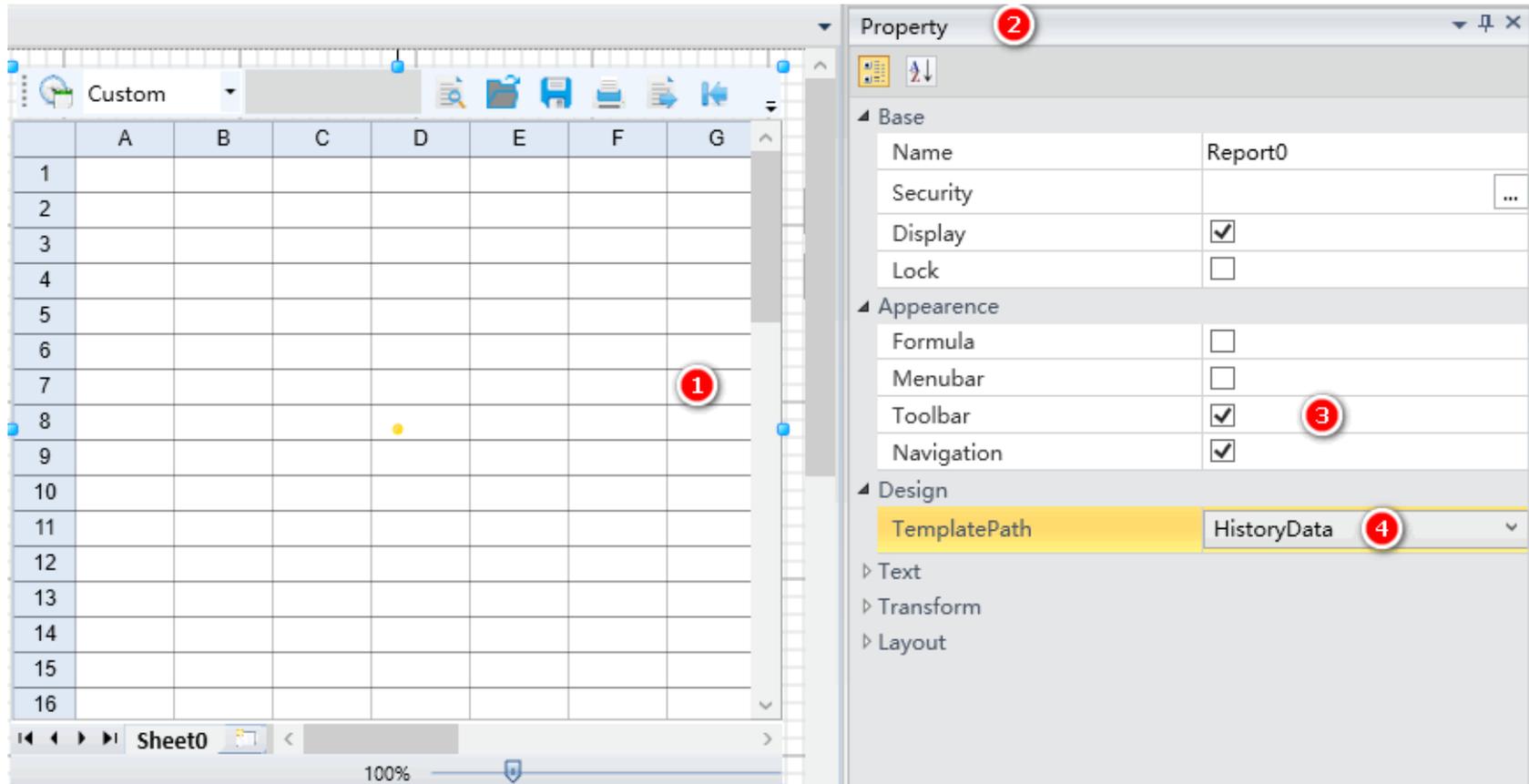
StartPage	Window0	HistoryData x											
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	#GetHisData	#GetHisDa	#GetHisData("VarRecord.RecordVariable1,Year,Month,Day,Hour,Minute,Second,Value")										
2													
3													
4													
5													
6													

#GetHisData("VarRecord.RecordVariable1,Year,Month,Day,Hour,Minute,Second,Value")

#GetHisData("VarRecord.RecordVariable,Year,Month,Day,Hour,Minute,Second,Value")

#GetHisData("VarRecord.RecordVariable,Year,Month,Day,Hour,Minute,Second,TriggerTime")

(7) Create a Report0 in the Window0, and bound the HistoryData report template

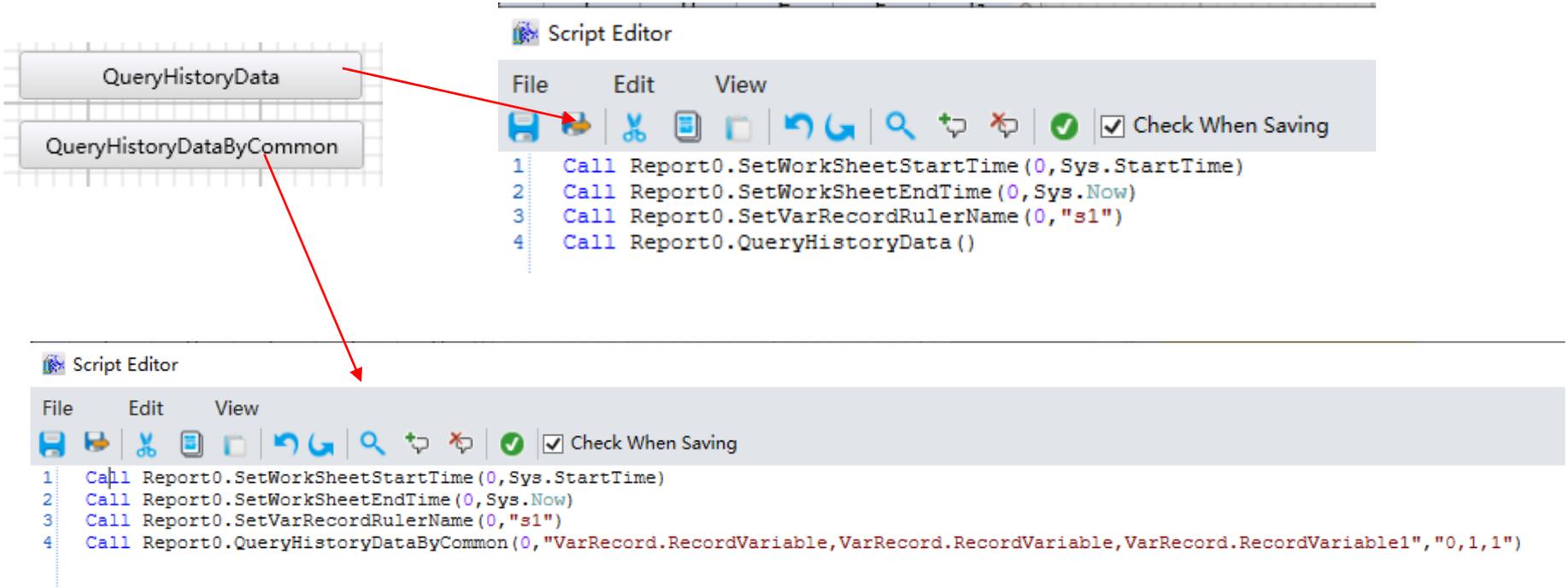


The screenshot displays a software interface with a spreadsheet on the left and a 'Property' window on the right. The spreadsheet shows a grid with columns A through G and rows 1 through 16. A yellow dot is located in cell D8, marked with a red circle containing the number '1'. The 'Property' window is titled 'Property' and contains several sections:

- Base**:
 - Name: Report0
 - Security: ...
 - Display:
 - Lock:
- Appearance**:
 - Formula:
 - Menubar:
 - Toolbar: (marked with a red circle '3')
 - Navigation:
- Design**:
 - TemplatePath: HistoryData (marked with a red circle '4')
 - Text:
 - Transform:
 - Layout:

Other red circles are present: a '2' in the top right of the property window, and a '4' in the bottom right of the property window.

(8) Create two buttons in the Window0, configure the LeftButtonDown event of the two buttons

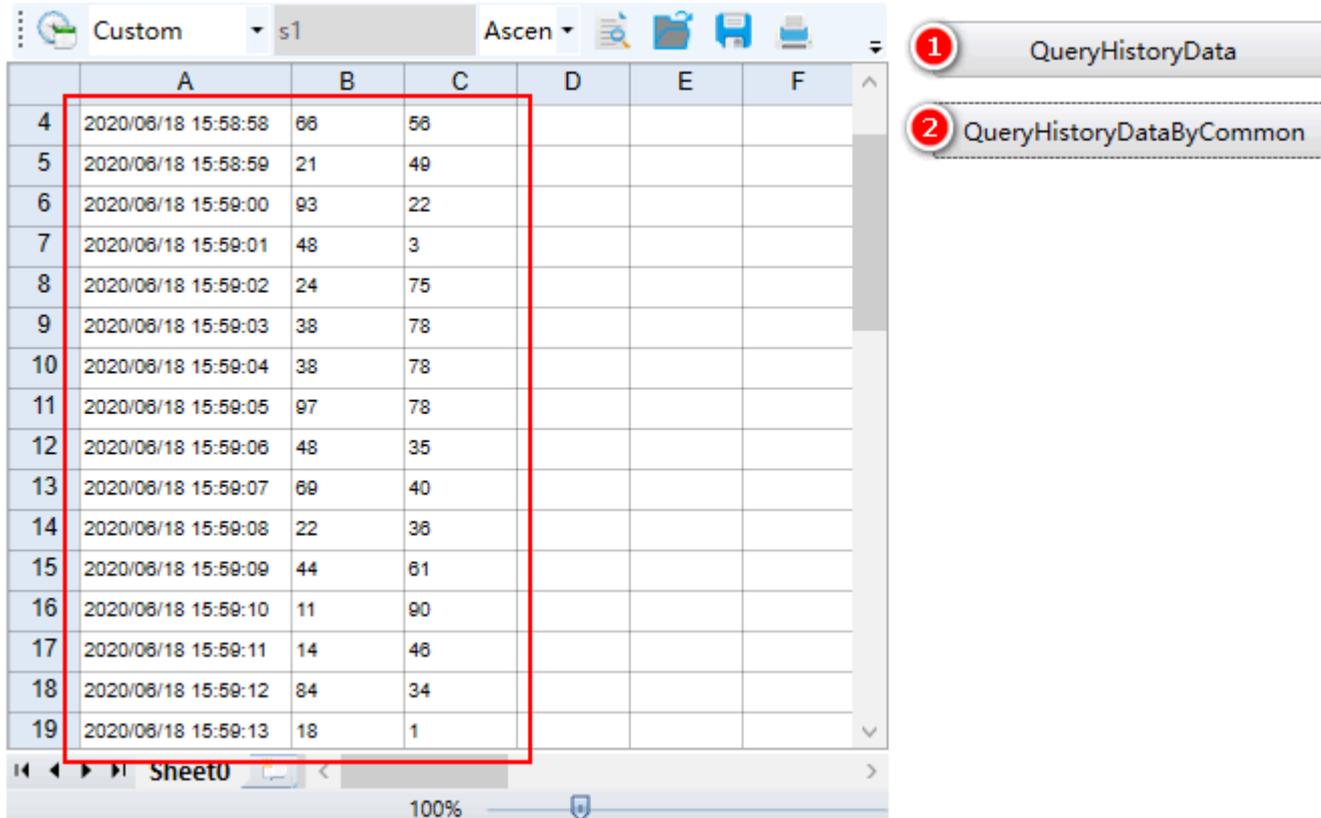


The image shows two screenshots of a report design tool. The top screenshot displays two buttons: "QueryHistoryData" and "QueryHistoryDataByCommon". Red arrows point from these buttons to two separate "Script Editor" windows. The first "Script Editor" window shows the script for the "QueryHistoryData" button, which includes four lines of code: setting start and end times, setting a ruler name, and calling the "QueryHistoryData" method. The second "Script Editor" window shows the script for the "QueryHistoryDataByCommon" button, which includes four lines of code: setting start and end times, setting a ruler name, and calling the "QueryHistoryDataByCommon" method with specific parameters.

```
Script Editor
File Edit View
1 Call Report0.SetWorkSheetStartTime (0, Sys.StartTime)
2 Call Report0.SetWorkSheetEndTime (0, Sys.Now)
3 Call Report0.SetVarRecordRulerName (0, "s1")
4 Call Report0.QueryHistoryData ()

Script Editor
File Edit View
1 Call Report0.SetWorkSheetStartTime (0, Sys.StartTime)
2 Call Report0.SetWorkSheetEndTime (0, Sys.Now)
3 Call Report0.SetVarRecordRulerName (0, "s1")
4 Call Report0.QueryHistoryDataByCommon (0, "VarRecord.RecordVariable,VarRecord.RecordVariable,VarRecord.RecordVariable1", "0,1,1")
```

(9) Run the Window0, click the “QueryHistoryData” or “QueryHistoryDataByCommon” button, then the Report0 displays the history data from the start time to current time



The screenshot shows a report window with a table of query history data. The table has columns A, B, C, D, E, and F. The data is as follows:

	A	B	C	D	E	F
4	2020/08/18 15:58:58	66	56			
5	2020/08/18 15:58:59	21	49			
6	2020/08/18 15:59:00	93	22			
7	2020/08/18 15:59:01	48	3			
8	2020/08/18 15:59:02	24	75			
9	2020/08/18 15:59:03	38	78			
10	2020/08/18 15:59:04	38	78			
11	2020/08/18 15:59:05	97	78			
12	2020/08/18 15:59:06	48	35			
13	2020/08/18 15:59:07	69	40			
14	2020/08/18 15:59:08	22	36			
15	2020/08/18 15:59:09	44	61			
16	2020/08/18 15:59:10	11	90			
17	2020/08/18 15:59:11	14	46			
18	2020/08/18 15:59:12	84	34			
19	2020/08/18 15:59:13	18	1			

Two buttons are visible on the right side of the window:

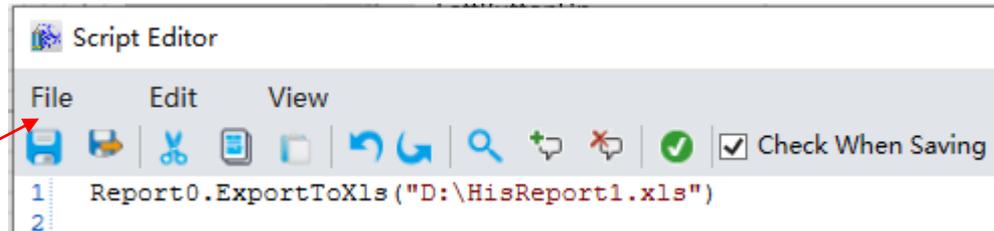
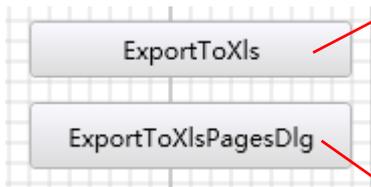
- 1 QueryHistoryData
- 2 QueryHistoryDataByCommon

➤ **ExportToXls, ExportToXlsPagesDlg** example :

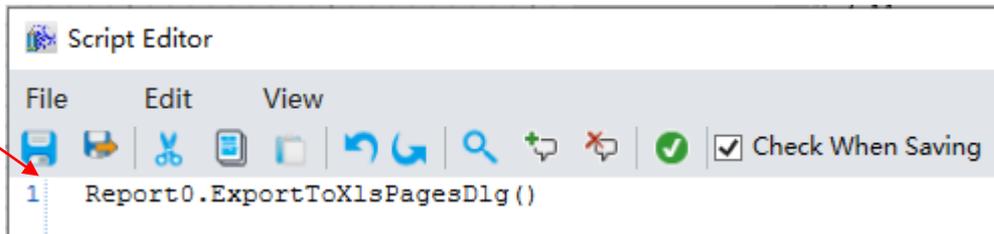
Export the history data displayed by report

The first 7 steps are the same as the first 7 steps of **QueryHistoryData, QueryHistoryDataByCommon** example

(8) Create two buttons in the Window0, configure the LeftButtonDown event of the two buttons

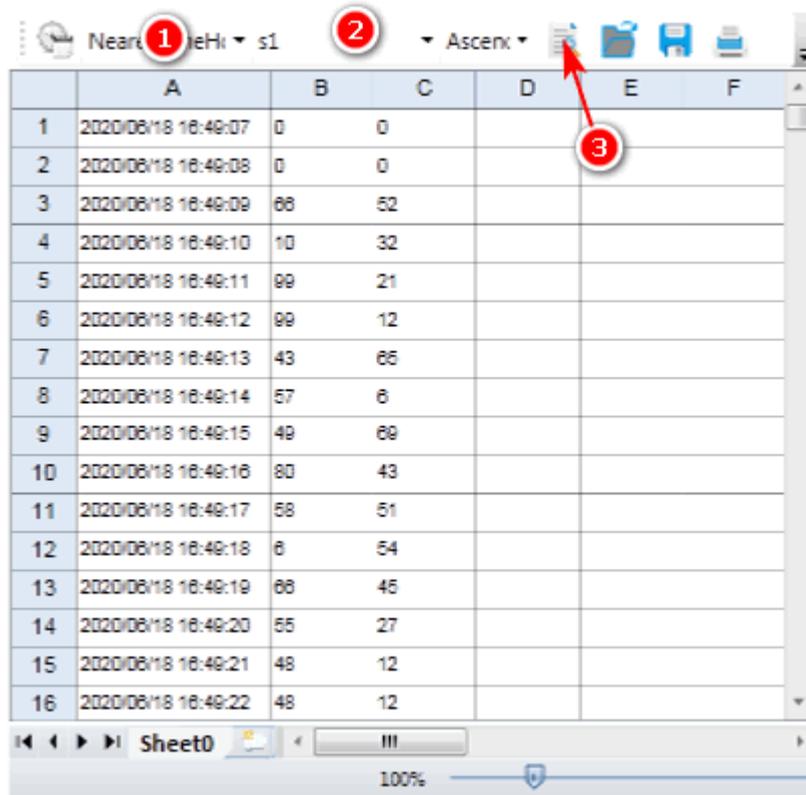


```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 Report0.ExportToXls ("D:\HisReport1.xls")
2
```



```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 Report0.ExportToXlsPagesDlg ()
```

(9) Run the Window0,query history data by toolbar buttons



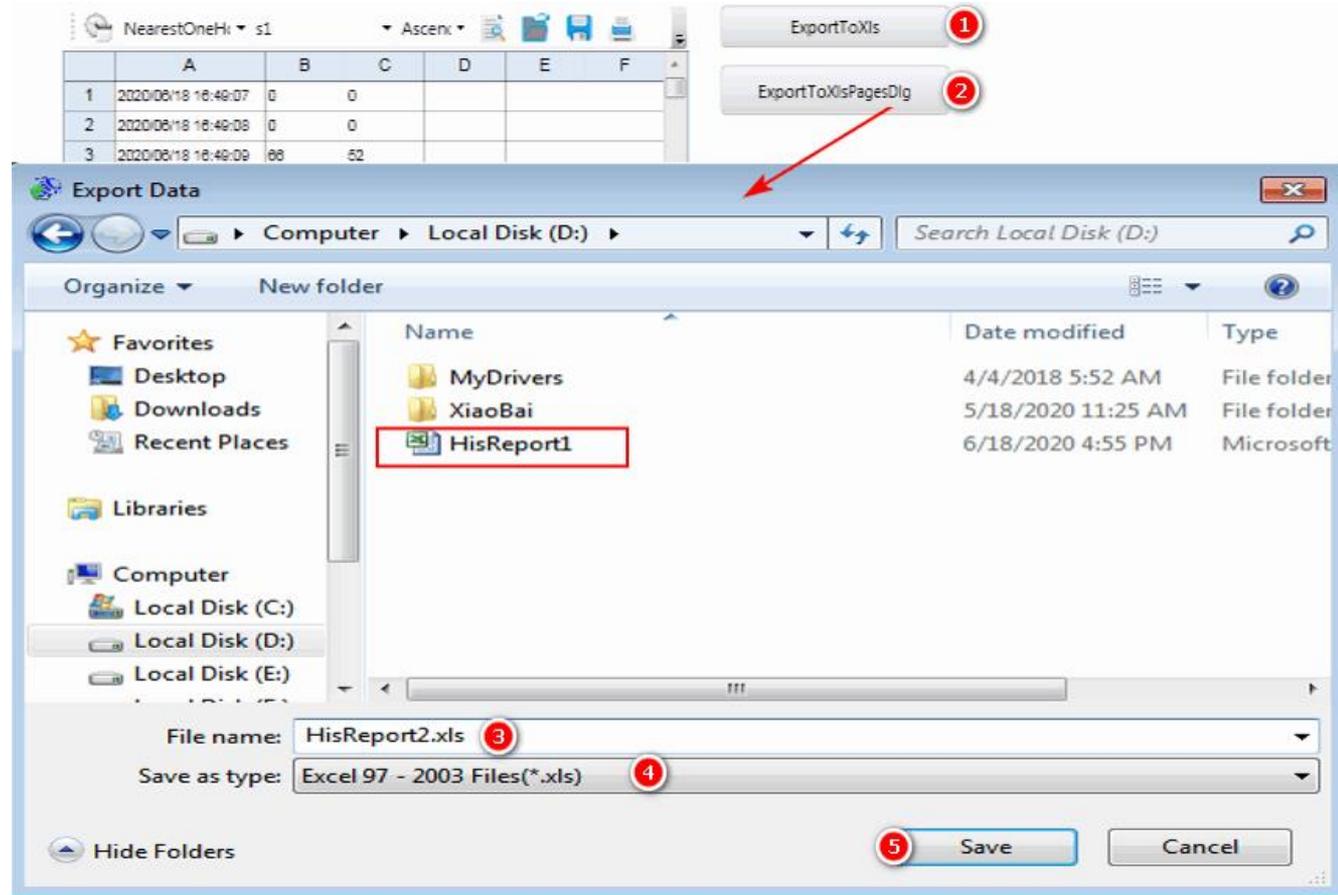
The screenshot shows a data table with 16 rows and 6 columns (A-F). The toolbar above the table includes a search icon (1), a dropdown menu (2) set to 'Ascend', and a toolbar button (3) with a red arrow pointing to it. The data in the table is as follows:

	A	B	C	D	E	F
1	2020/08/18 18:49:07	0	0			
2	2020/08/18 18:49:08	0	0			
3	2020/08/18 18:49:09	68	52			
4	2020/08/18 18:49:10	10	32			
5	2020/08/18 18:49:11	99	21			
6	2020/08/18 18:49:12	99	12			
7	2020/08/18 18:49:13	43	66			
8	2020/08/18 18:49:14	57	6			
9	2020/08/18 18:49:15	49	69			
10	2020/08/18 18:49:16	80	43			
11	2020/08/18 18:49:17	58	51			
12	2020/08/18 18:49:18	6	54			
13	2020/08/18 18:49:19	68	45			
14	2020/08/18 18:49:20	55	27			
15	2020/08/18 18:49:21	48	12			
16	2020/08/18 18:49:22	48	12			

(10) Execute the scripts

① Click the “ExportToXls” button , the data in Report0 is exported to the Excel file under the D disk, the file name is HisReport1

② Click the “ExportDataToXlsByDialog” button , pop up data export dialog , as shown on the right

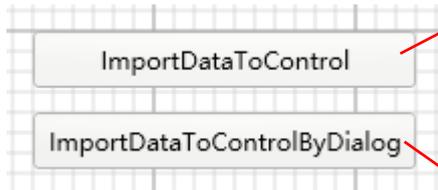


➤ **ImportDataToControl, ImportDataToControlByDialog** example :

Import external Excel file to Report0

The first 7 steps are the same as the first 7 steps of **QueryHistoryData, QueryHistoryDataByCommon** example

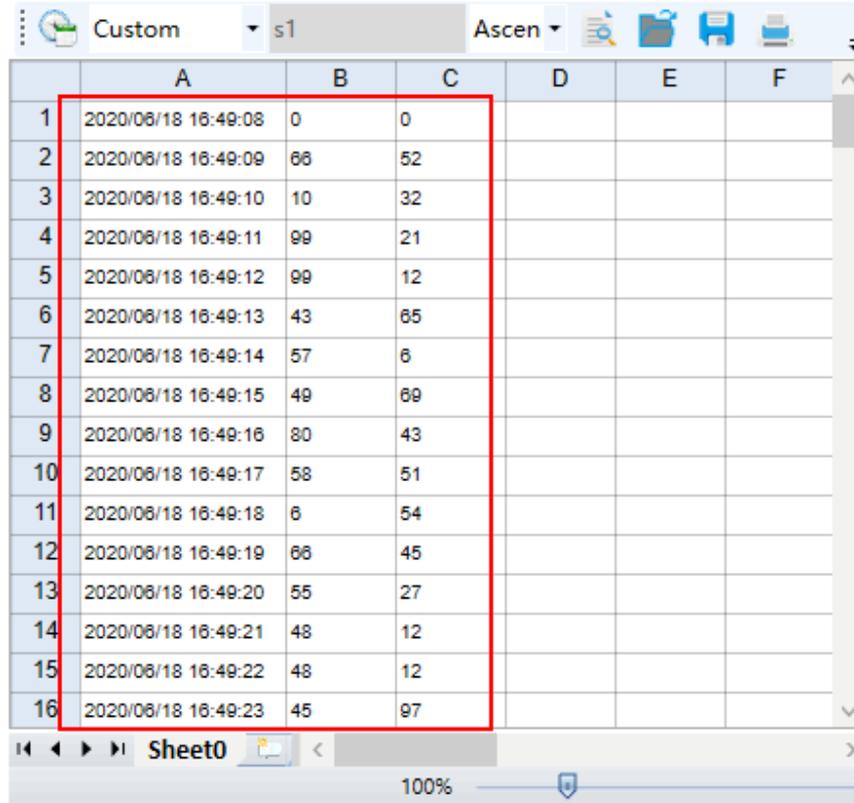
(8) Create two buttons in the Window0, configure the LeftButtonDown event of the two buttons



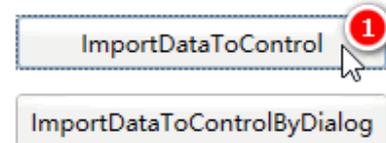
```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 call Report0.ClearCellContent ()
2 'clear the contents of the Report0
3 call Report0.ImportDataToControl ("D:\Hisreport1.xls")
4 'Import external Excel file (D:\Hisreport1.xls) to Report0
```

```
Script Editor
File Edit View
[Icons] [Check When Saving]
1 call Report0.ClearCellContent ()
2 'clear the contents of the Report0
3 call Report0.ImportDataToControlByDialog ()
4 'Select the Excel file to be imported through the pop-up dialog
```

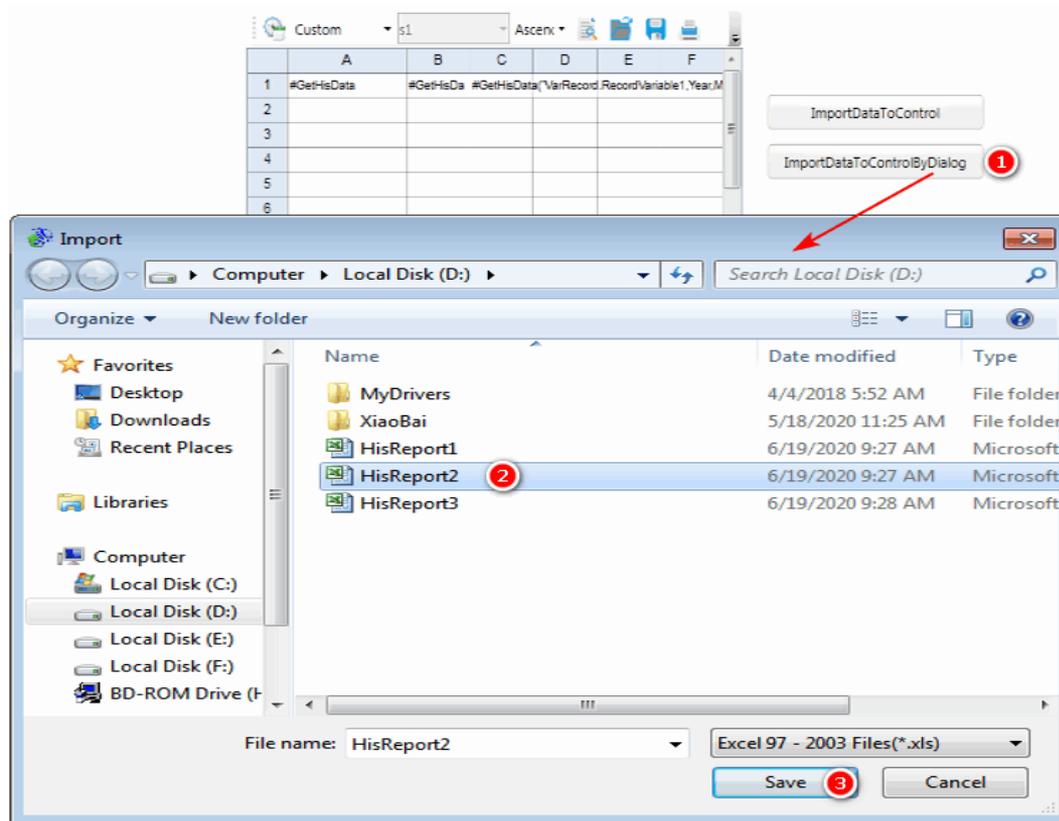
(9) Run the Window0, Click the "ImportDataToControl" to import the external Excel file(D:\Hisreport1.xls) to Report0



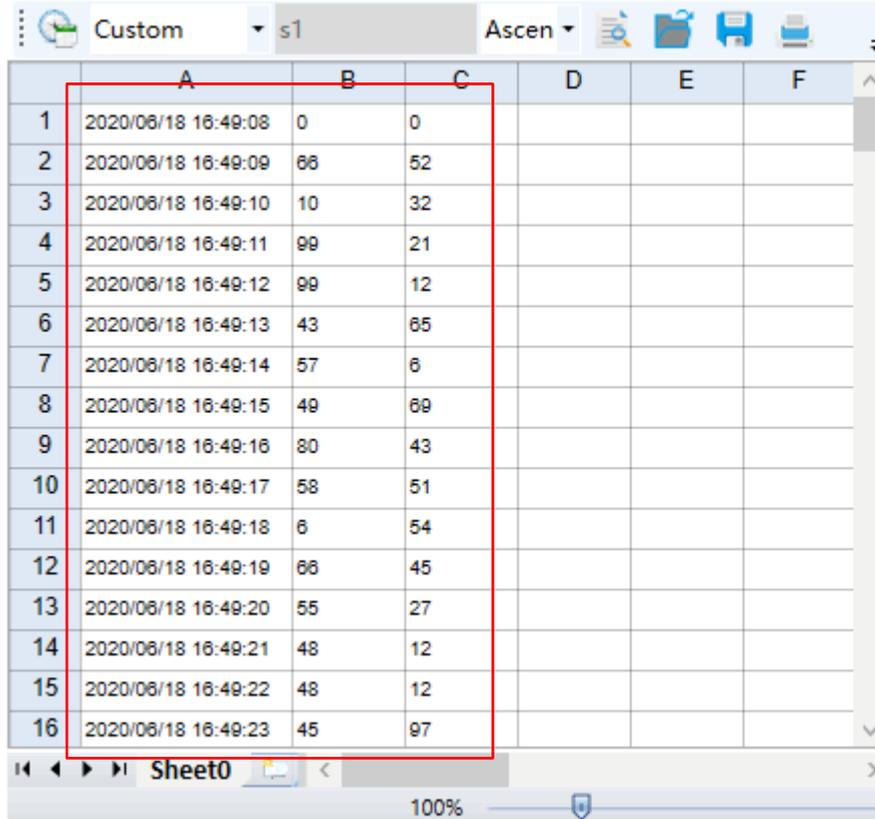
	A	B	C	D	E	F
1	2020/06/18 16:49:08	0	0			
2	2020/06/18 16:49:09	66	52			
3	2020/06/18 16:49:10	10	32			
4	2020/06/18 16:49:11	99	21			
5	2020/06/18 16:49:12	99	12			
6	2020/06/18 16:49:13	43	65			
7	2020/06/18 16:49:14	57	6			
8	2020/06/18 16:49:15	49	69			
9	2020/06/18 16:49:16	80	43			
10	2020/06/18 16:49:17	58	51			
11	2020/06/18 16:49:18	6	54			
12	2020/06/18 16:49:19	66	45			
13	2020/06/18 16:49:20	55	27			
14	2020/06/18 16:49:21	48	12			
15	2020/06/18 16:49:22	48	12			
16	2020/06/18 16:49:23	45	97			



(10) Click the “ImportDataToControlByDialog” button , pop up the import dialog to select the Excel file (D:\Hisreport2.xls) to import to Report0,as shown below



(11) The external Excel file (D:\Hisreport2.xls) was imported to Report0 successfully

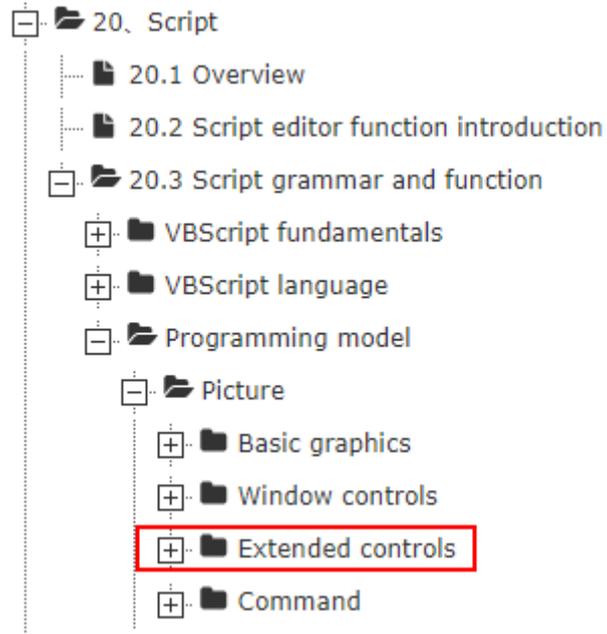


	A	B	C	D	E	F
1	2020/06/18 16:49:08	0	0			
2	2020/06/18 16:49:09	66	52			
3	2020/06/18 16:49:10	10	32			
4	2020/06/18 16:49:11	99	21			
5	2020/06/18 16:49:12	99	12			
6	2020/06/18 16:49:13	43	65			
7	2020/06/18 16:49:14	57	6			
8	2020/06/18 16:49:15	49	69			
9	2020/06/18 16:49:16	80	43			
10	2020/06/18 16:49:17	58	51			
11	2020/06/18 16:49:18	6	54			
12	2020/06/18 16:49:19	68	45			
13	2020/06/18 16:49:20	55	27			
14	2020/06/18 16:49:21	48	12			
15	2020/06/18 16:49:22	48	12			
16	2020/06/18 16:49:23	45	97			

ImportDataToControl

ImportDataToControlByDialog

For more details about the scripts usage of extend controls, please refer to the section “20.3 Script grammar and function” in the user manual. As shown in the figure below:

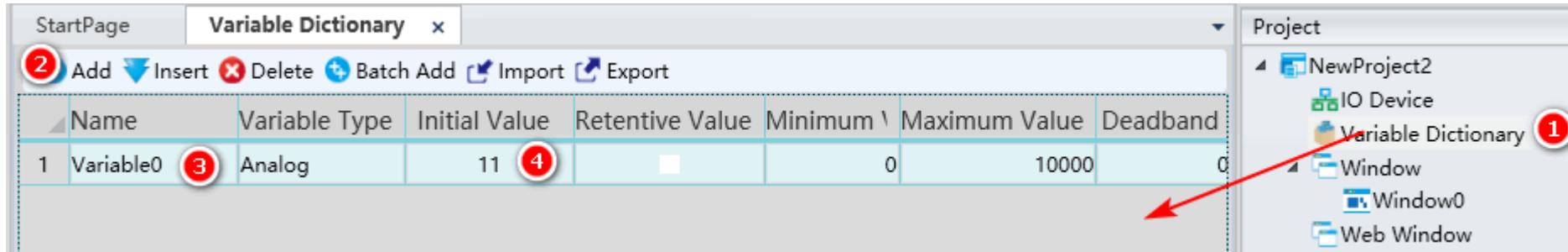


- The concepts of scripts
- The scripts of Basic Graphics
- The scripts of Window Controls
- The scripts of Extend Controls
- **The Action scripts**
- The Window scripts
- The Color scripts

➤ AnalogValueInput example :

Change variable value

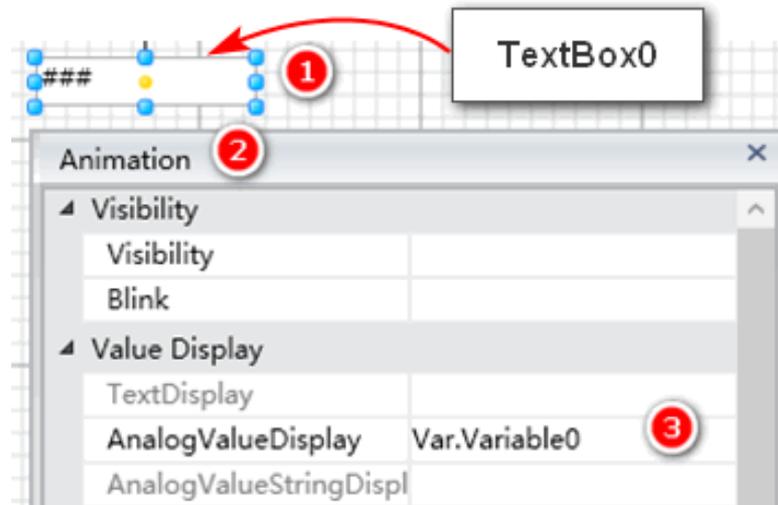
(1) Create a variable : Variable0



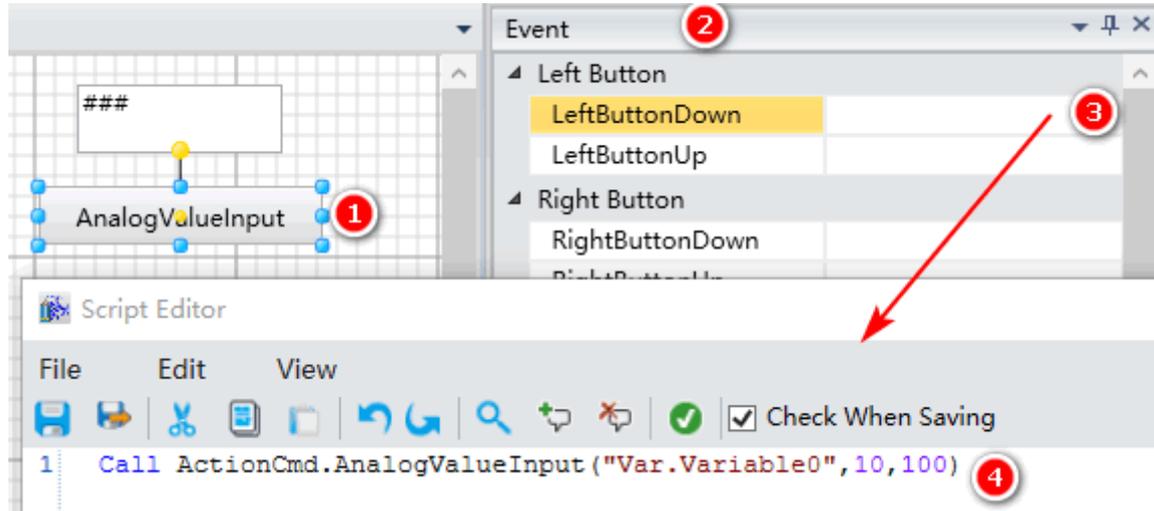
	Name	Variable Type	Initial Value	Retentive Value	Minimum Value	Maximum Value	Deadband
1	Variable0	Analog	11	<input type="checkbox"/>	0	10000	0

※Refer to the section "6.3 Variables" in user manual.

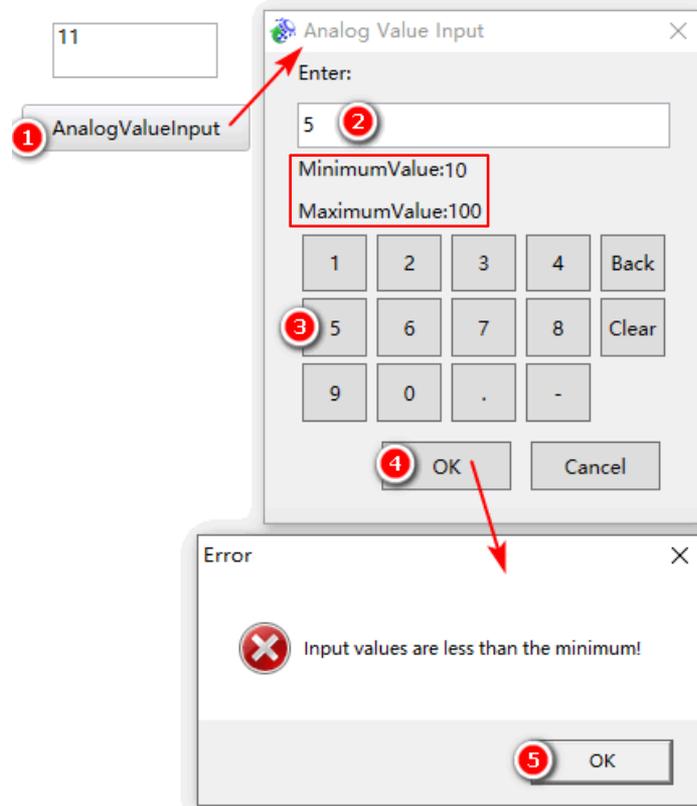
(2) Create a text box (Textbox0) in Window0, the analog value display animation of Textbox0 is associated with Variable0



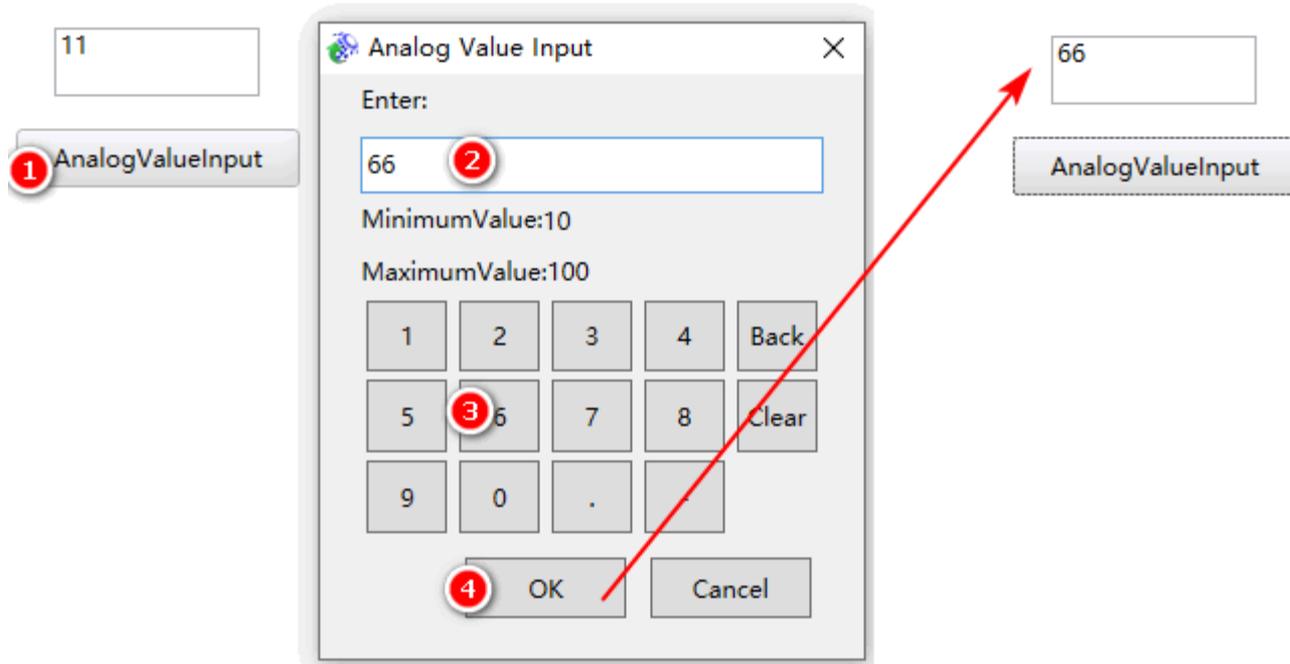
(3) Create a button (AnalogValueInput) in the Window0, configure the LeftButtonDown event of the button



(4) Run the Window0, click the “AnalogValueInput” button, pop up the analog value input dialog, input a number less than 10 or greater than 100 , input failure



(5) Run the Window0, click the “AnalogValueInput” button, pop up the analog value input dialog, input a number between 10 and 100 , input successfully





The Action Scripts

➤ ButtonValueInput example :

The variable increases by 10 every 2 seconds

The first 2 steps are the same as the first 2 steps of AnalogValueInput example

(3) Create a window program in the Window0

1: Click on any blank in the Window0

2: Event window

3: Window Program

4: Add

5: Execute Mode

6: Interval Time(ms)

7: Program

8: Script Editor

9: File

10: OK

Name	Program	Execute Mode	Interval Time(ms)	IsEnabled	Description
Program	(None)	Running	2000	<input checked="" type="checkbox"/>	

```
1 Call ActionCmd.ButtonValueInput ("Var.Variable0", 1, 10)
```

① Click on any blank in the Window0

⑤ Set this program to be executed at runtime

⑥ Set this program to be executed every 2 seconds

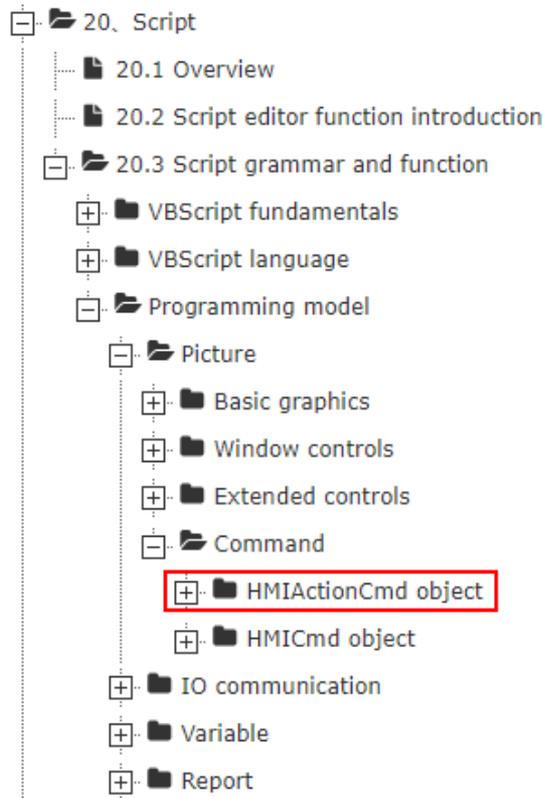


The Action Scripts

(4) Run the Window0 , at the beginning, the Textbox0 displays 11, and then the display value of TextBox0 increases by 10 every 2 seconds

A rectangular text box with a thin black border and a red outer highlight. The number '61' is displayed inside the box.

For more details about the usage of action scripts , please refer to the section “20.3 Script grammar and function” in the user manual. As shown in the figure below:



- The concepts of scripts
- The scripts of Basic Graphics
- The scripts of Window Controls
- The scripts of Extend Controls
- The Action scripts
- The Window scripts
- The Color scripts

➤ OpenDialogWindow, CloseDialogWindow example :

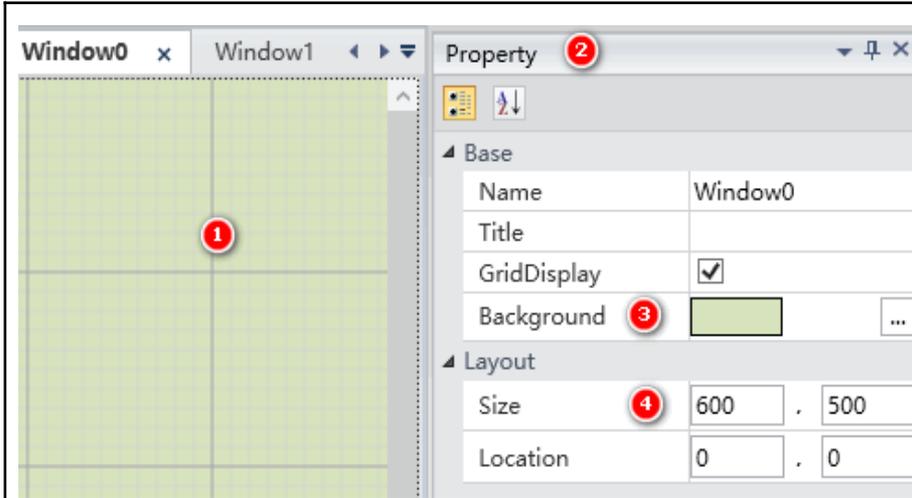
Open a window dialog and then close it

(1) Create 2 windows(Window0, Window1) in the project.



※Refer to the section "7.2.1.1 Add window" in user manual.

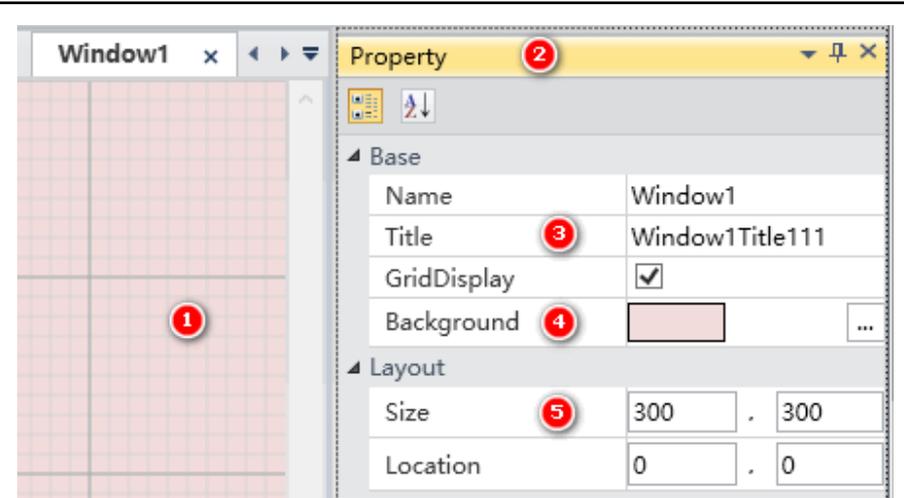
(2) Set the properties of Window0, Window1



Base	
Name	Window0
Title	
GridDisplay	<input checked="" type="checkbox"/>
Background	 ...

Layout	
Size	600 . 500
Location	0 . 0

Window0

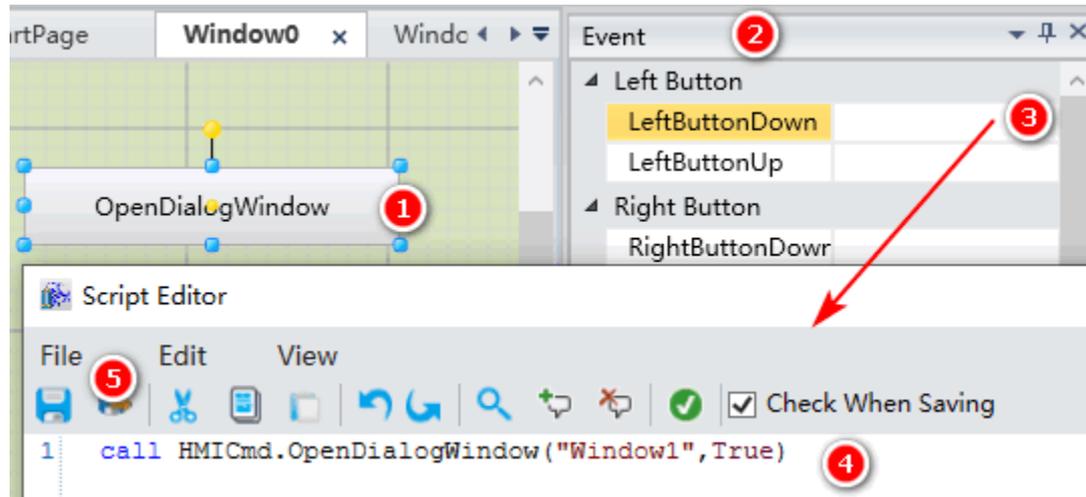


Base	
Name	Window1
Title	Window1Title111
GridDisplay	<input checked="" type="checkbox"/>
Background	 ...

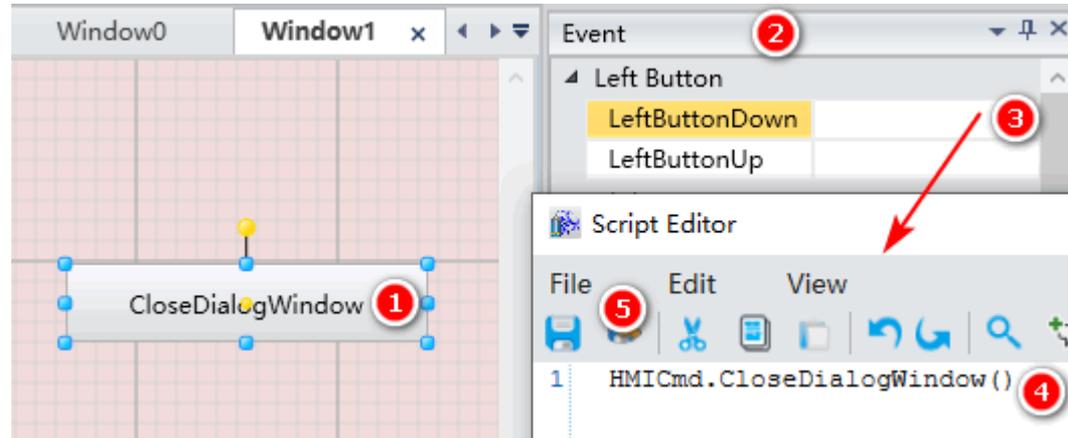
Layout	
Size	300 . 300
Location	0 . 0

Window1

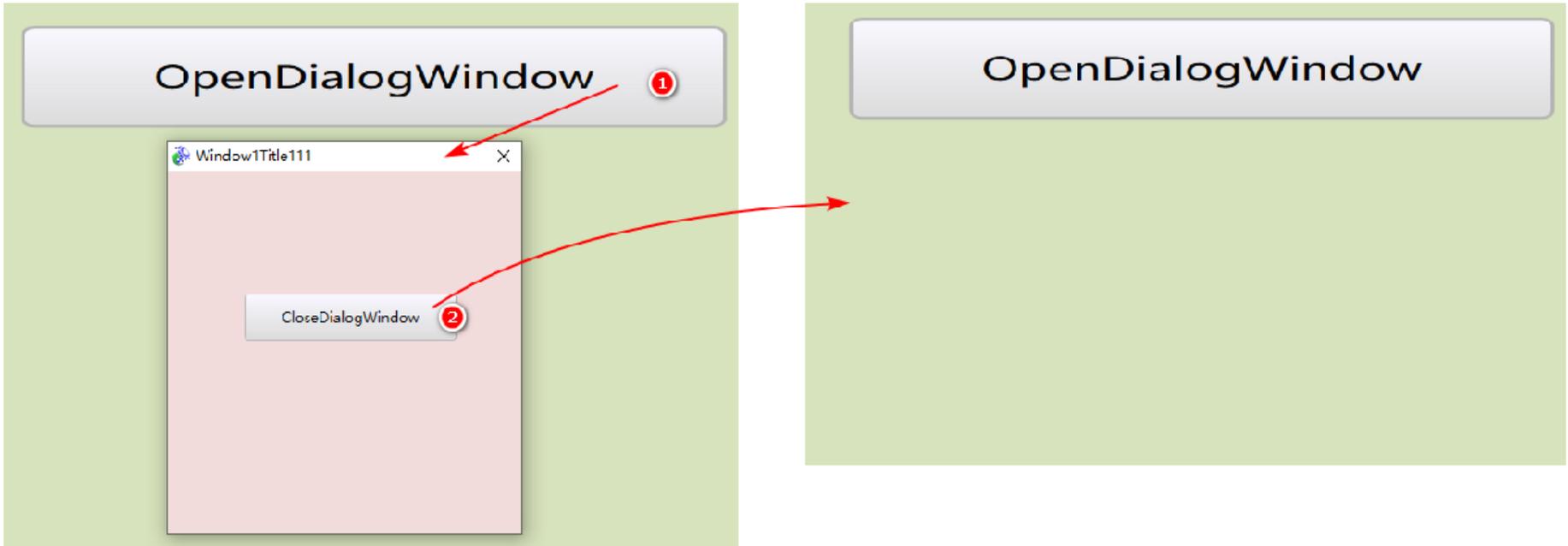
(3) Create a button (OpenDialogWindow) in the Window0, configure the LeftButtonDown event of the button



(4) Create a button (CloseDialogWindow) in the Window1, configure the LeftButtonDown event of the button



(5) Run the Window0



① Click the “OpenDialogWindow” button in the Window0, pop up the Window1 dialog , whose title is “Window1Title111”

② Click the “CloseDialogWindow” button in the Window1 dialog , then the Window1 dialog is closed

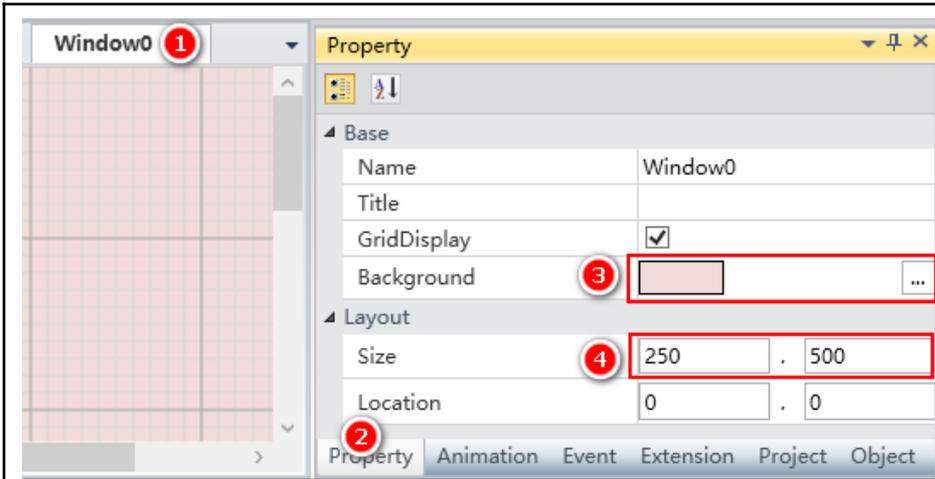
➤ OpenWindowAndCloseOther example :

(1) Create two windows(Window0,Window1) in the project



※Refer to the section "7.2.1.1 Add window" in user manual.

(2) Set the properties of Window0, Window1



Window0 **1**

Property

Base

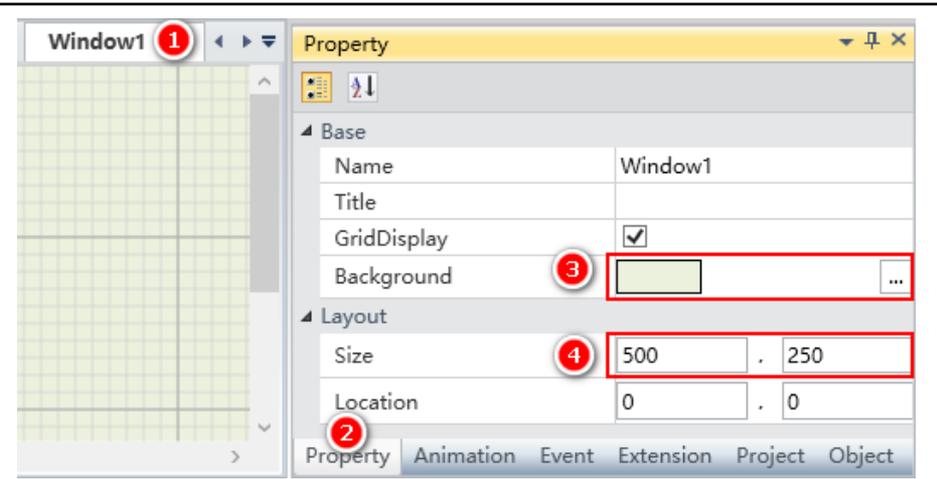
Name	Window0
Title	
GridDisplay	<input checked="" type="checkbox"/>
Background	3 ...

Layout

Size	4 250 . 500
Location	0 . 0

2 Property Animation Event Extension Project Object

Window0



Window1 **1**

Property

Base

Name	Window1
Title	
GridDisplay	<input checked="" type="checkbox"/>
Background	3 ...

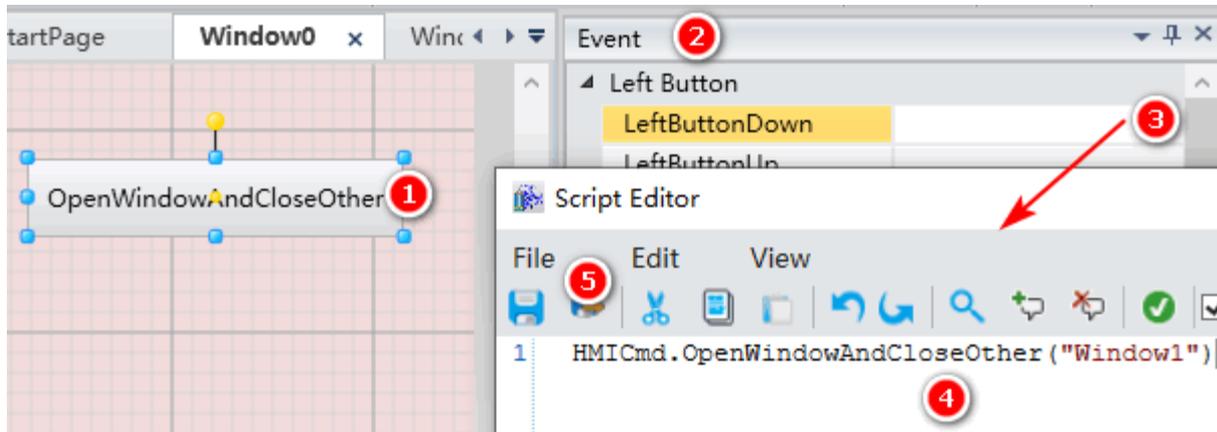
Layout

Size	4 500 . 250
Location	0 . 0

2 Property Animation Event Extension Project Object

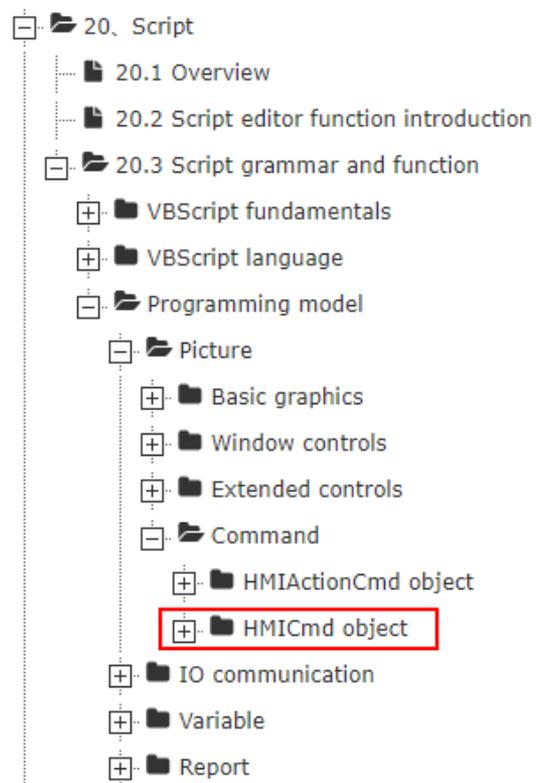
Window1

(3) Create a button (OpenWindowAndCloseOther) in the Window0, configure the LeftButtonDown event of the button



(4) Only run the Window0 default. Then click the "OpenWindowAndCloseOther" button, the Window1 is opened, Window0 is Closed. Only Window1 is running

For more details about the usage of window scripts , please refer to the section “20.3 Script grammar and function” in the user manual. As shown in the figure below:

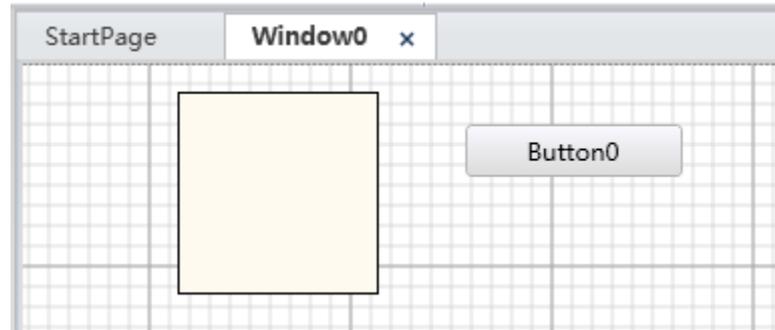


- The concepts of scripts
- The scripts of Basic Graphics
- The scripts of Window Controls
- The scripts of Extend Controls
- The Action scripts
- The Window scripts
- **The Color scripts**

- ColorSelectionBox example(This script calls up the Color Editor) :

The rectangle is filled with a linear gradient color

(1)Create a Rectangle0 and a Button0 in the Window0, configure the left mouse down event of the button0



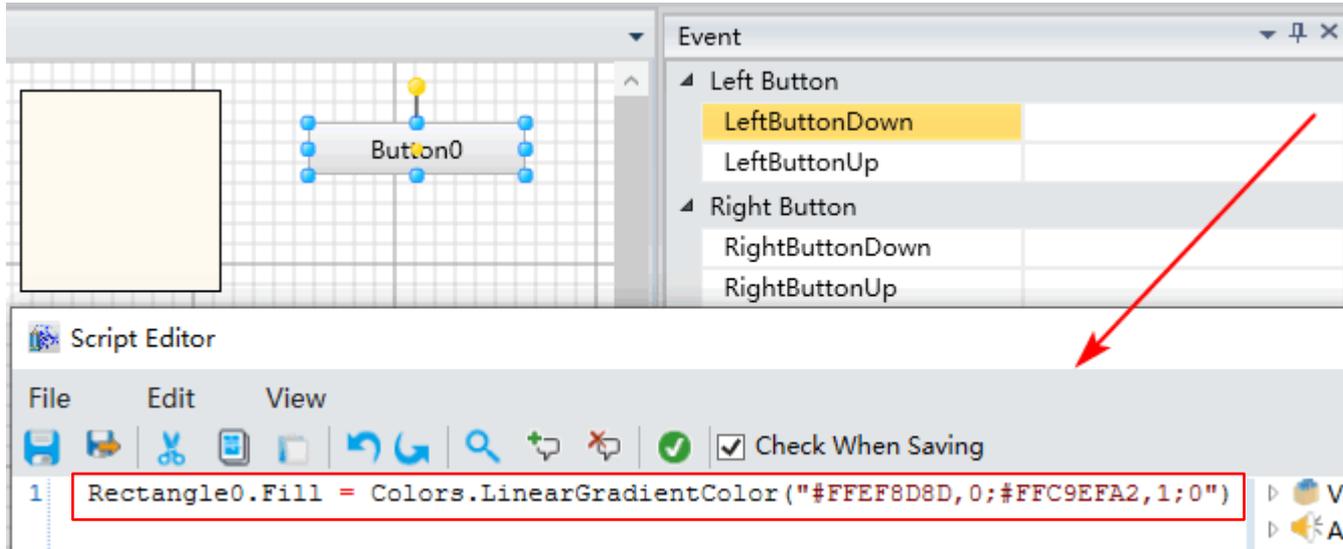
(2) Configure the LeftMouseDown event of the Button0

The Color Scripts

The screenshot illustrates the configuration of the LeftMouseDown event for Button0. The **Event** window shows the **LeftButtonDown** event selected. The **Color Choices** dialog is open, showing the **Gradient** option selected. The **InitialColor** and **FinalColor** are both set to 0% opacity. The **Command** list shows the **ColorSelectionBox()** command selected. The status bar at the bottom indicates the current state: **Status: Insert Info Button0 Event info MouseLeftButtonDown Line 1 Column 19**.

(3) The configuration results are as follows

The Color Scripts

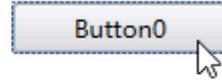
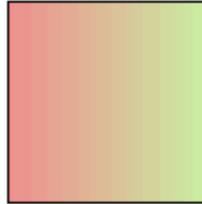


The screenshot displays a software configuration interface. On the left, a grid shows a yellow square labeled 'Rectangle0' and a button labeled 'Button0'. On the right, an 'Event' panel lists events for 'Left Button' and 'Right Button'. The 'LeftButtonDown' event is selected and highlighted in yellow. A red arrow points from this event to the 'Script Editor' below. The 'Script Editor' has a menu bar with 'File', 'Edit', and 'View', and a toolbar with icons for save, copy, paste, undo, redo, search, and a 'Check When Saving' checkbox. The script text is: `1 Rectangle0.Fill = Colors.LinearGradientColor("#FFEF8D8D,0;#FFC9EFA2,1;0")`. The script text is enclosed in a red rectangular box.

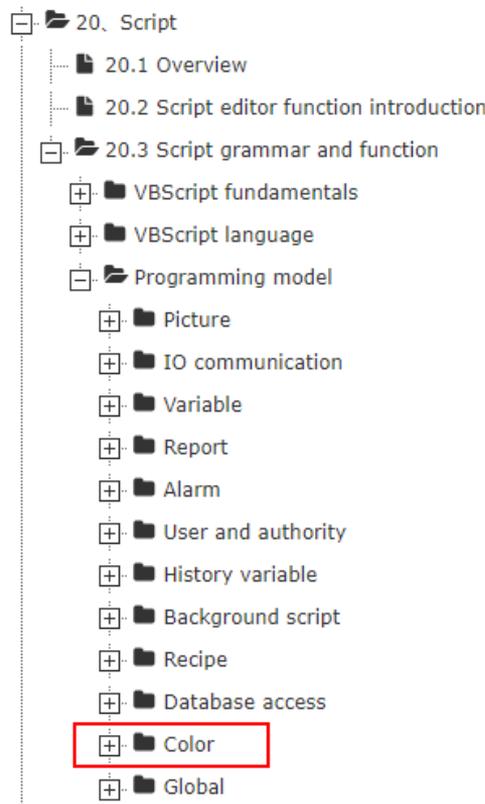


The Color Scripts

(4) Run the Window0, click the Button0, then the Rectangle0 becomes a red-green linear gradient fill



For more details about the usage of color scripts , please refer to the section “20.3 Script grammar and function” in the user manual. As shown in the figure below:



Smarter. Greener. Together.

To learn more about Delta, please visit
www.deltaww.com.

