如何在SymmetrE中添加动画

Version 1.0

此 白皮书 由

Honeywell

创 建

http://customer.honeywell.cn

Email to ECC AP TAC team: ecc.ap.tac@honeywell.com

Version 1.0 2009-05

目　录

一、 前言

1. *引言*

众所周知，优秀的用户界面能够：

- 生动活泼的展示系统全貌，给用户身临其境，一览众山“晓”的体验；
- 帮助操作员更好的全面认识我们的系统，从而减少误操作，保证系统正常运行；
- 抓住业主的眼球，让我们在激烈的竞争中脱颖而出。

然而，目前许多 SymmetrE 项目的用户界面，都不免流于单调呆板，乏善可陈，客户体验一般。虽然能基本满足控制要求，但距离生动的客户体验相去甚远。

SymmetrE 提供了灵活的方式来丰富用户界面，增加人机界面的友好度，这里我们介绍其中的一种——在 SymmetrE 中添加动画。希望对大家做图形界面有所帮助，

2. *添加动画的方法*

在 SymmetrE 中添加动画的方式有三种

- 插入 GIF 图片；
- 插入 Sha 文件；
- 运用脚本，制造动画效果。

下面我们逐一讲解每种方式

二、 插入 GIF 图片

1. *什么是 GIF 图片*

GIF 是用于压缩具有单调颜色和清晰细节的图像（如线状图、徽标或带文字的插图）的标准格式。

GIF(Graphics Interchange Format)的原义是“图像互换格式”，是 CompuServe 公司在 1987 年开发的图像文件格式。GIF 文件的数据，是一种基于 LZW 算法的连续色调的无损压缩格式。其压缩率一般在 50％左右，它不属于任何应用程序。目前几乎所有相关软件都支持它，公共领域有大量的软件在使用 GIF 图像文件。GIF 图像文件的数据是经过压缩的，而且是采用了可变长度等压缩算法。所以 GIF 的图像深度从 lbit 到 8bit，也即 GIF 最多支持 256 种色彩的图像。GIF 格式的另一个特点是其在一个 GIF 文件中可以存多幅彩色图像，如果把存于一个文件中的多幅图像数据逐幅读出并显示到屏幕上，就可构成一种最简单的动画。

GIF 分为静态 GIF 和动画 GIF 两种，支持透明背景图像，适用于多种操作系统，“体型”很小，网上很多小动画都是 GIF 格式。其实 GIF 是将多幅图像保存为一个图像文件，从而形成动画,所以归根到底 GIF 仍然是图片文件格式。但 GIF 只能显示 256 色。

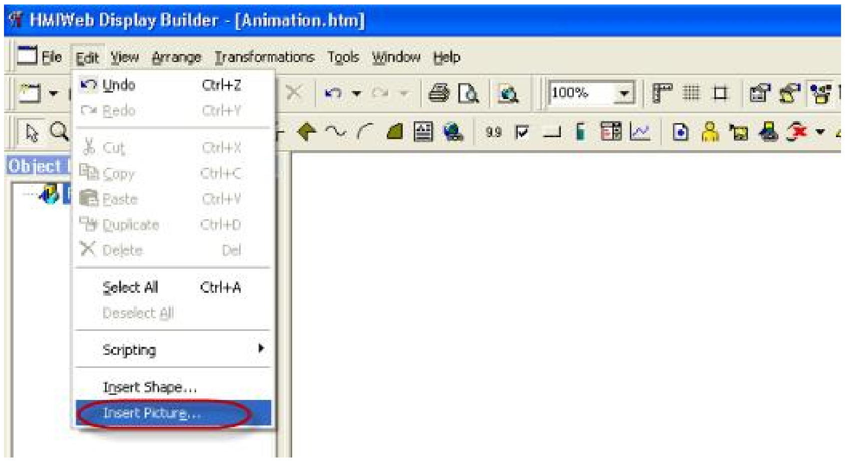
GIF 格式自 1987 年由 CompuServe 公司引入后，因其体积小而成像相对清晰，特别适合于初期慢速的互联网，而从此大受欢迎。它采用无损压缩技术，只要图像不多于 256 色，则可既减少文件的大小，又保持成像的质量。

2. *如何在 SymmetrE 中插入 GIF 图片*

这里以在 SymmetrE 中插入一片转动的风扇为例，简单的介绍一下操作步骤。

首先，将 gif 图片 Fan_On.gif 复制到 C:\honeywell\client\abstract 目录下,同时我们也将 Fan_Off.gif 复制到同一目录下，以供对比。

其次，打开 HMIWeb Display Builder，新建一个 display，在菜单栏中 Edit 下选择 Insert Pictuer…，指出 GIF 图片所在路径，打开。



这样，这些 GIF 图片都被插入到 Display 中了。

最后，在 Station 中打开该页面，就可以看到转动的风扇了。

Station - Default - (Animation.htm)
Station Edit View Action Configure Help

3. *如何制作 GIF 图片*

目前市面上有很多 GIF 制作软件,推荐使用 Microsoft GIF Animator（免费的），它简单实用，容易上手。如果追求更多图片效果可以用其它第三方的收费软件。

GIF 素材，有很多网站都提供专业的 GIF 素材，用户可根据自己的需要去网上下载。具体网址就不推荐了。

三、 插入 Sha 文件

1. *什么是 Sha 文件*

Sha 文件，全称 Shape 文件，是 SymmetrE 中的一种专有格式——sha 格式。它有以下两种：

- Dynamic Shape：可链接动态数据的的 shape 文件。
- Shape Sequence：一组相关的 shape 文件，每个 shape 文件都不能有动态数据连接。Shape Sequence 可以用来表现某个设备的不同状态。(每个 shape 标识一个对应的状态，做动画效果（每个 shape 文件是动画中的一帧）

Sha 文件能嵌在视图中，单个 sha 文件也可以是 Shape Sequence 中一个对象。

2. *Shape Gallery*

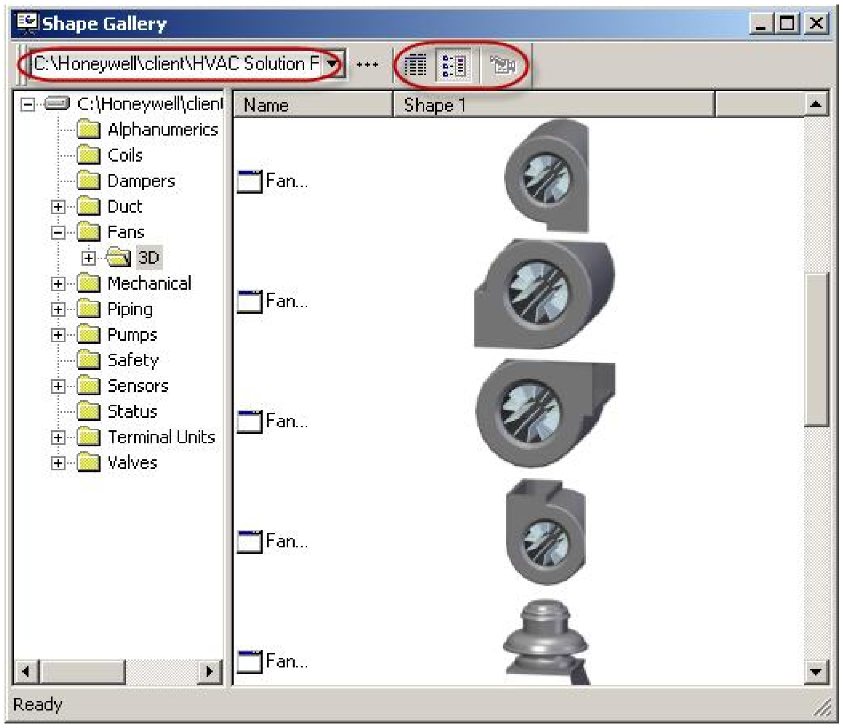
SymmetrE 提供了常用的 sha 文件库，也提供了 Sha 文件的管理工具—Shape Gallery，它能够预览 sha 文件，还能帮助快速将 sha 文件插入到视图中。

- Shape Gallery 的使用方法：

选择菜单栏->View->Shape Gallery 或者在工具栏中点击 ，打开 Shape Gallery

a. 选择文件对应的路径和浏览方式

b. 选择参看方式： 细节， 预览， 播放。



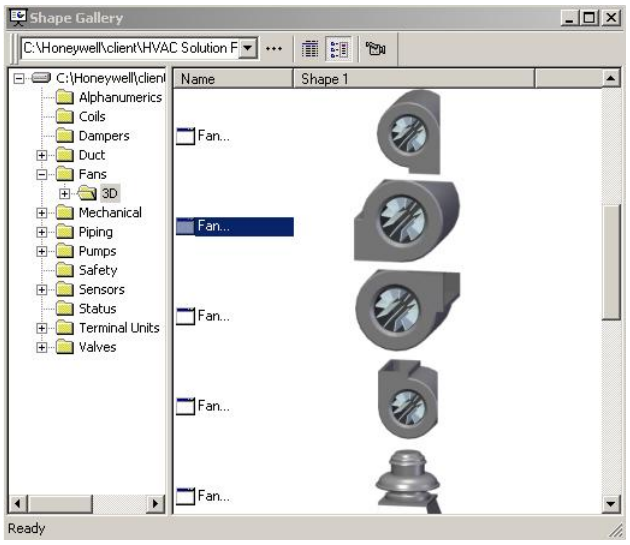
3. *插入 Sha 文件*

这里我们以插入一个转动的风机为例，来说明具体的操作步骤：

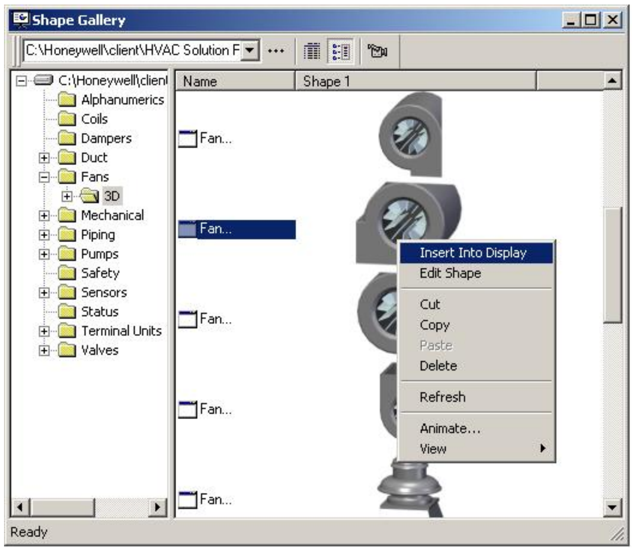
- 利用 Shape Gallery

a. 打开 HMI Display Builder，新建一个视图，

b. 打开 Shape Gallery,选中要插入的文件



c. 右键单击选中的对象，选择 Insert to Display



这样我们就把这个风机插入视图中了。

- 直接插入

方法类似直接插入图片，在菜单栏中 Edit 下选择 *Insert Shape…*，指出 Shape 文件所在路径，打开。这个方法的缺陷是无法预览 shape 图形。

4. *如何制作 Sha 文件*

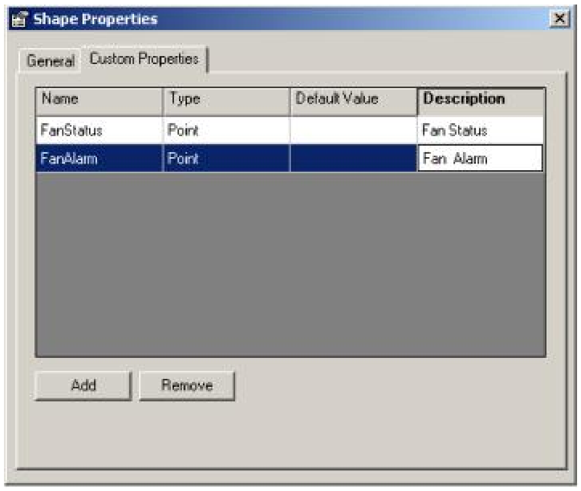
4.1. 如何制作 Dynamic shape 文件

下面我们以制作一个风机 shape 文件，为例，风机关联两个数据库的点 FanStatus 和 FanAlarm 。

a. 新建一个 Dynamic Shape 文件

b. 点击 Properties Window，切换至 Customer Properties 栏，进行自定义属性是设置。

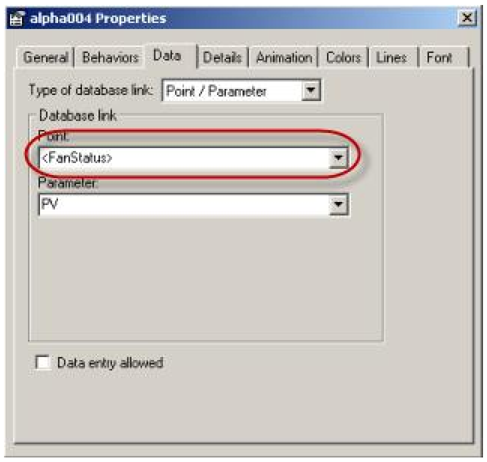
c. 点击 Add 添加，输入自定义属性如下：


Shape Properties

General | Custom Properties

Name	Type	Default Value	Description
FanStatus	Point		Fan Status
FanAlarm	Point		Fan Alarm

d. 插入一个风机图片，加入两个点的描述：用文本框编辑“运行状态”，添加一个 Alphanumeric,在 Data 按如下设置

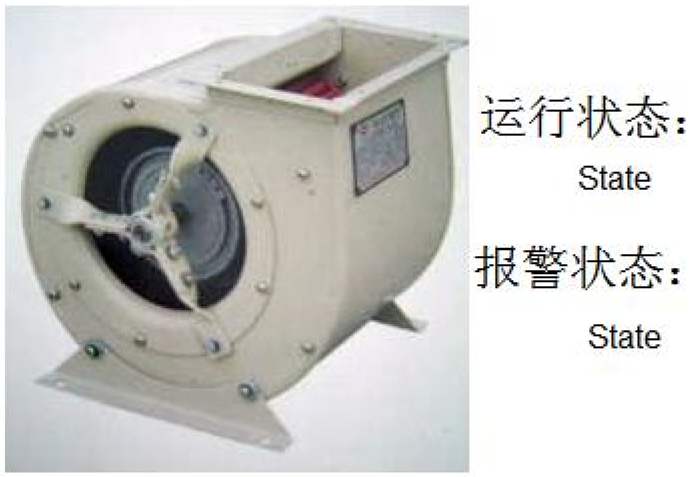

alpha004 Properties

General | Behaviors | Data | Details | Animation | Colors | Lines | Font

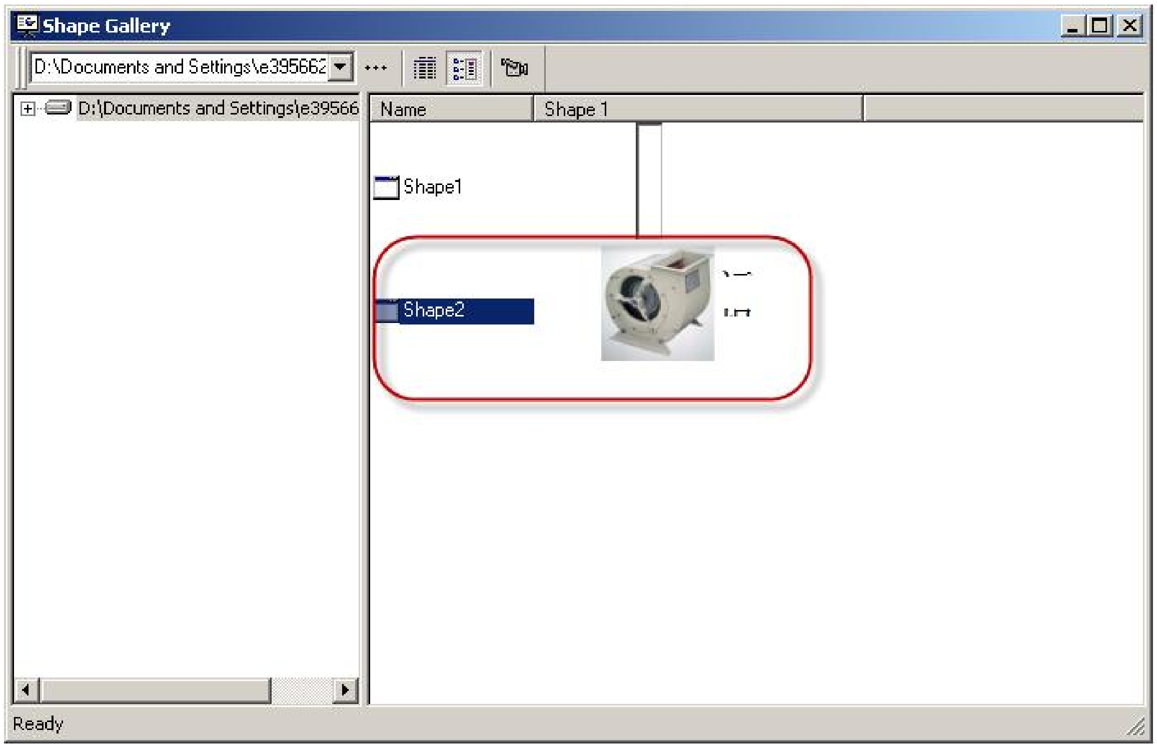
Type of database link: Point / Parameter
Database link
Point: <FanStatus>
Parameter: PV
Data entry allowed

e. 同样的方式编辑 “报警状态”。

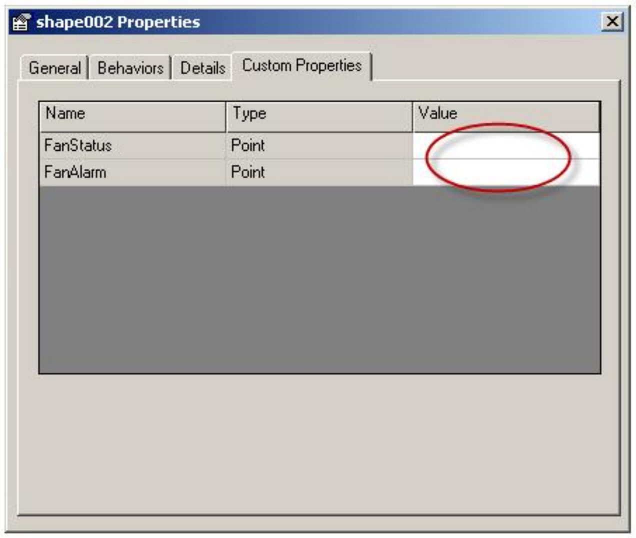
f. 选中所有对象，把它们组合成一个对象。保存(Save)



这样一个动态 Shape 文件就做好了，在 Shape Gallery 中查看：


Shape Gallery
D:\Documents and Settings\e395662
D:\Documents and Settings\e39566
Name
Shape 1
Shape1
Shape2
Ready

运用测试，新建一个 display 文件，插入刚刚建好的动态 shape 文件。效果如下：



Name	Type	Value
FanStatus	Point	
FanAlarm	Point	

在标记处绑定现场的监控点。

4.2.　如何制作 Shape Sequence

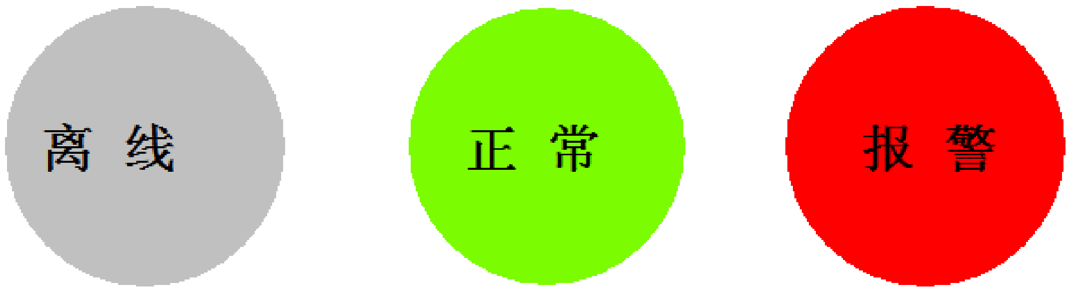
我们以制作一个简单的报警指示为例：

a. 新建一个 Shape Sequence 文件

b. 绘制两个图形，编辑如下：

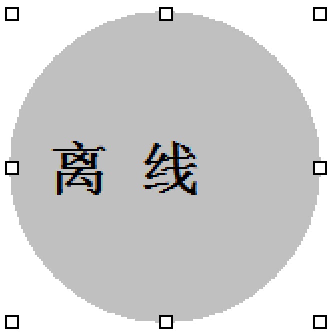
注意：每个图形必须组合（group）成一个对象

图形的摆放顺序（从左至右，从上至下）是按对应的点的状态，第一个为断线的状态。所以下图对应的现场状态分别是（断线，0，1）

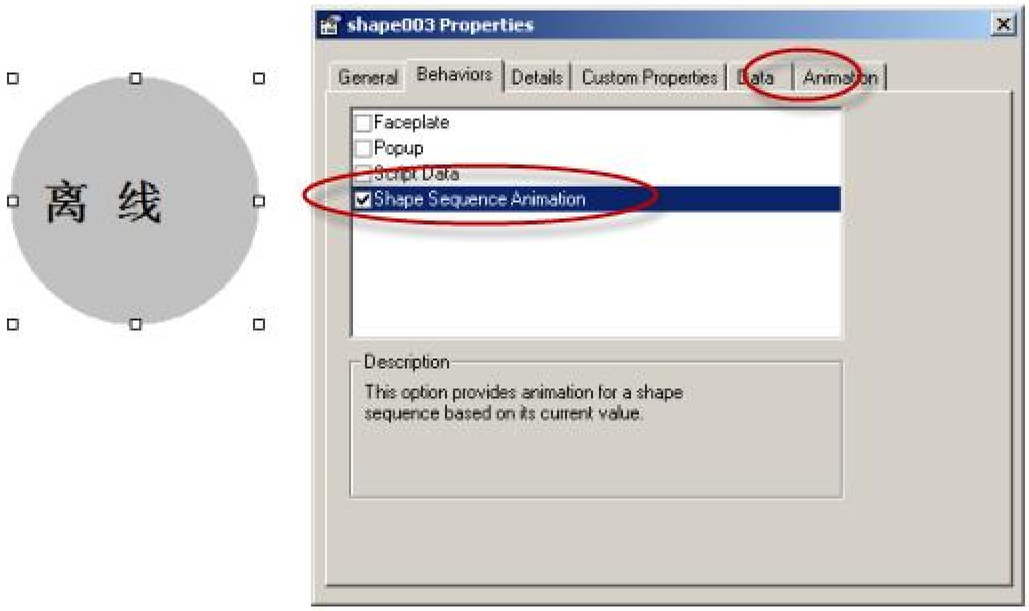


这样一个 Shape Sequence 文件就做好了。

运用测试，将我们做好的 Shape Sequence 文件插入到 display 中。



双击对象，编辑其属性，在 Behaviors 选中 Shape Sequence Animation，选中后会新增两栏 Data 和 Animation。

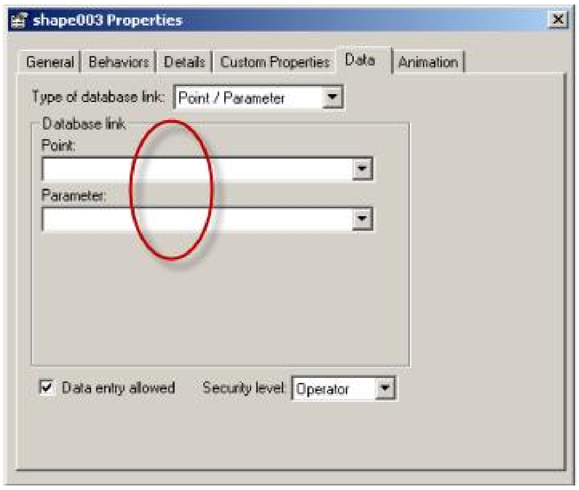


- [ ] Faceplate
- [ ] Popup
- [ ] Script Data
- [x] Shape Sequence Animation

Description

This option provides animation for a shape sequence based on its current value.

在 Data 栏进行点的绑定



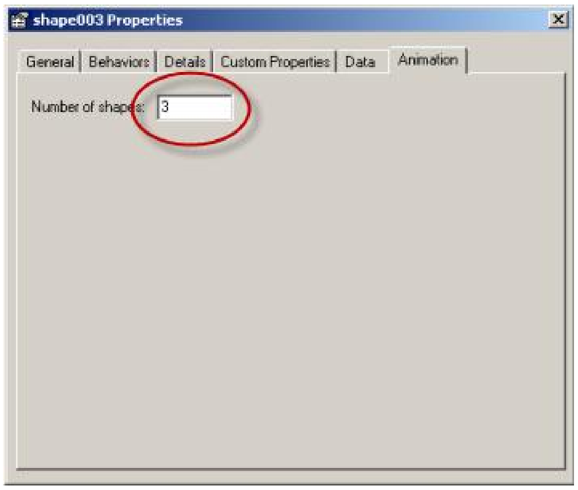
Type of database link: Point / Parameter

Point:

Parameter:

[✓] Data entry allowed Security level: Operator

在 Animation 栏 选择使用的 shape 文件的数量（图形库中，某些 Shape Sequence 文件的 shape 状态多于现场控制要求，那么我们只选择其中几个就好了，如我们输入 3 即仅使用前 3 个）



Number of shapes: 3

5. *Sha 文件关联脚本*

HMIWeb display Builder 还支持 Shape 文件关联脚本，这些脚本存储在 disply 中，可以结合脚本开发出更友好的界面。

四、 脚本

1. *脚本基础*

页面的脚本拓展了 HMIWeb Display 的功能，能创造出更多更友好的页面。脚本的本质是一个个小的程序，他们能实现某些特定功能。

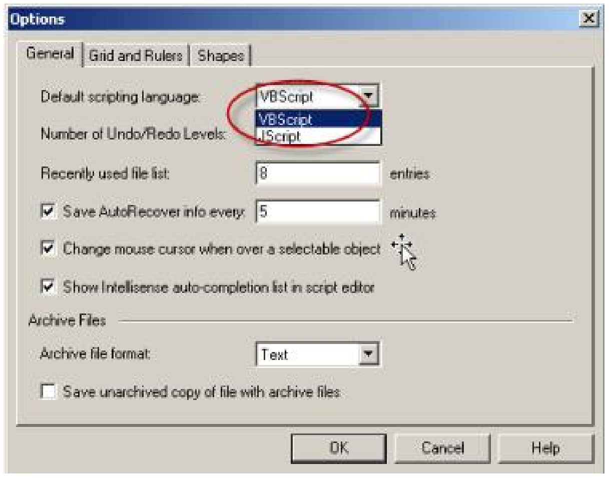
1.1.脚本语言

HMIWeb Display 支持 VBScript 和 JScript

- 脚本语言的切换：

菜单栏 Tools->Options 打开对话框；

在 General 栏，Default Scripting Language 中选择，初始设置是 VBScript。



Default scripting language: VBScript (VBScript / JScript)
Number of Undo/Redo Levels:
Recently used file list: 8 entries
☑ Save AutoRecover info every: 5 minutes
☑ Change mouse cursor when over a selectable object
☑ Show Intellisense auto-completion list in script editor
Archive Files
Archive file format: Text
☐ Save unarchived copy of file with archive files

1.2.脚本常用的术语

Event：脚本是基于 Event 的，脚本只有在相关的 Event 发生时，才会被触发。

Timer：一个非常实用的函数，能让脚本按设定的时间运行。

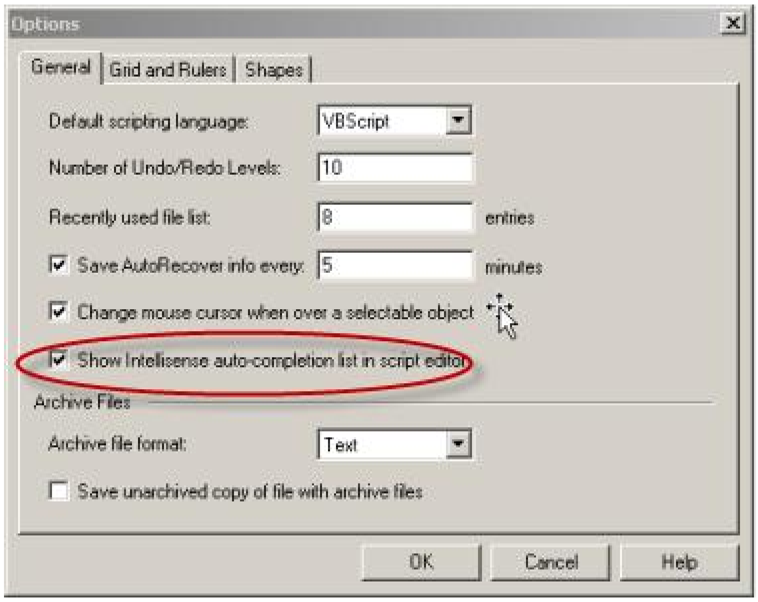
脚本有基于某一 Object（对象）的，基于页面的，基于 Shape 文件的，基于 Station 的等不同类型。

2. *Script Editor*

Script Editor 是 HMIWeb Display Builder 的脚本编辑工具：

- 它能即时保存，当你切换至另外的对象或 Event 时，你编辑的脚本会被自动保存。

- 它能辅助你完成，确认 Show Intellisense auto-completion list in script editor 被选中。


Options

General | Grid and Rulers | Shapes

Default scripting language: VBScript

Number of Undo/Redo Levels: 10

Recently used file list: 8 entries

☑ Save AutoRecover info every: 5 minutes

☑ Change mouse cursor when over a selectable object

☑ Show Intellisense auto-completion list in script editor

Archive Files

Archive file format: Text

☐ Save unarchived copy of file with archive files

OK　Cancel　Help

3. *编写脚本*

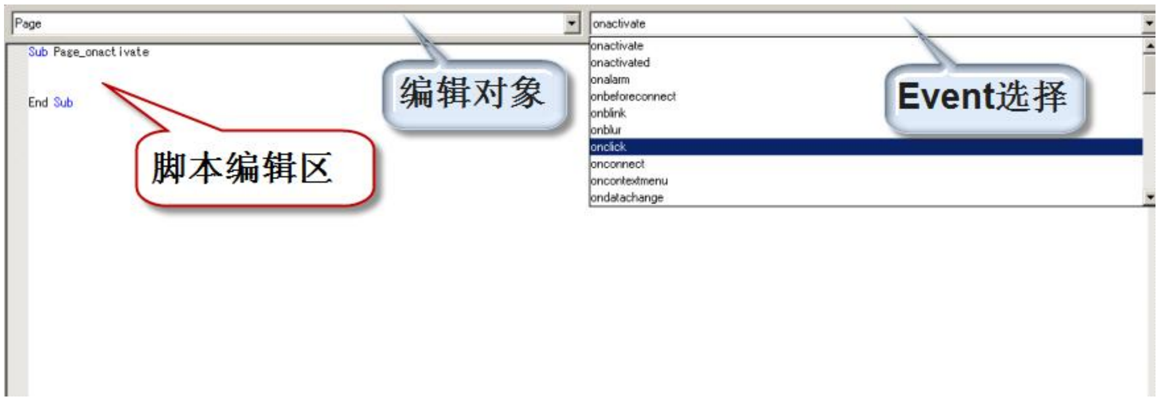
3.1.点击工具栏中的 ，进入 Script Editor；

3.2.在左边选择编辑对象：

下拉菜单中列出了所有对象，包括页面和 General（一些 General-purpose 的脚本和变量声明）；

3.3.选在右边择激活你脚本的 Event；

3.4.在标本编辑区编辑你的脚本，基本框架已经给你列出来了。



```
Sub Page_onactivate

End Sub
```

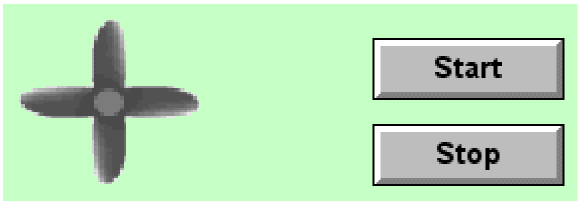
提示：

- Script Editor 支持复制，剪切，粘贴功能
- Script Editor 支持自动保存

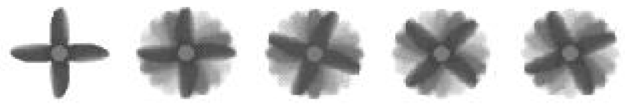
- Script Editor 有语法检查，辅助编辑。

4. *脚本举例*

- 要求：有一个风扇，用“开始”和“停止”按钮来控制，希望风扇是一个动画效果。



- 解决方案：用以下的 Shape Sequence 和一些脚本来完成这个动画



左边第一个表示“ OFF”状态，右边的是个 shape 来模拟“ ON”的动画效果。这些 shape 文件都是位图（bitmap）,它们都是在画图软件中绘制的，然后粘贴到 shape 文件中。用这种方式创建 Shape 文件，能帮助我们用相当简单的方法做出更逼真的效果。

- 脚本参考：

“ Start”按钮基于 Onclick event，我们添加这些脚本：创建一个 200ms timer，来控制风扇转动的速度。

```
Sub cmdStartBtn_onclick()
intTimerID=window.setInterval(“myfunction()”,200)
End Sub
```

在 General section 中我们定义 myfunction 函数

```
Dim intTimerID
Function myfunction()
If fan.value = fan.shapefile.numberOfShapes Then
fan.value = 2
Else
fan.value = fan.value + 1
End If
```

End Function

“Stop”按钮也是基于 onclick event,附上的脚本是：

Sub cmdStopBtn——onclick()

window.clearInterval(intTimerID)

fan.value = 1

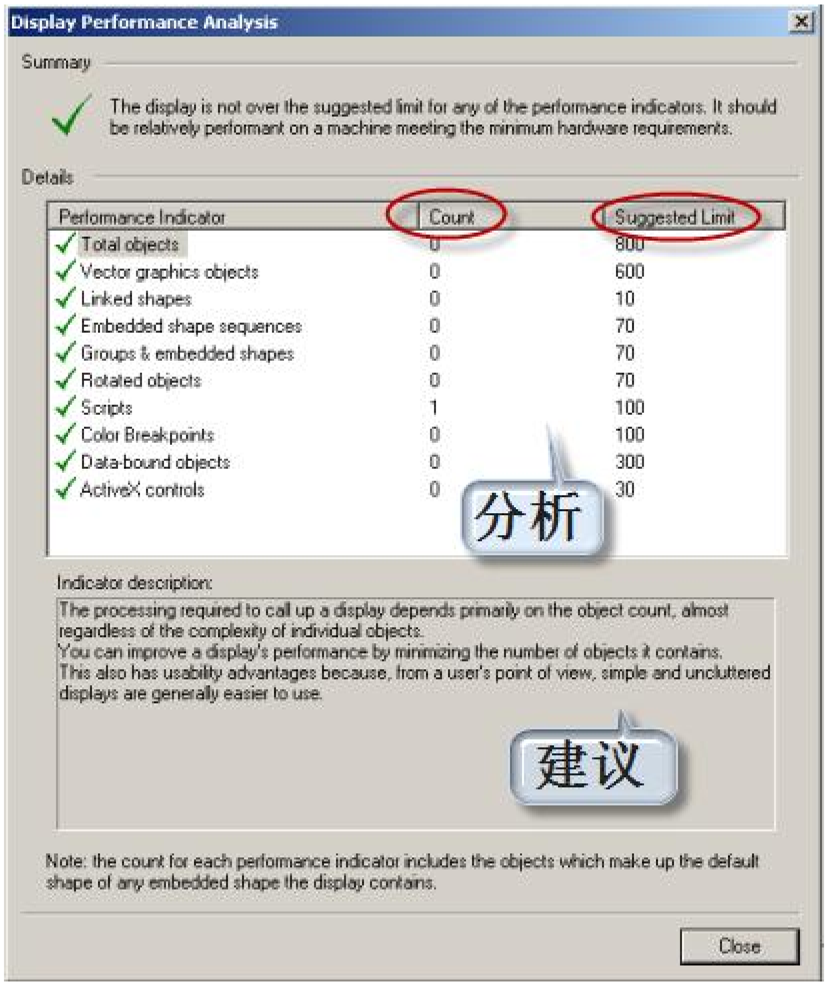
End Sub

更多参见附录 13 Example Scenarios

五、　优化 Display 页面

少量的脚本有助于我们提高页面的表现力，加强客户体验。但是过多的脚本却会给我们带来麻烦，容易导致页面出错。HMIWeb Display Builder 提供了一个页面分析工具——Display Performance Analysis 来辅助我们，避免这样的错误。

点击菜单栏->Tools, Display Performance Analysis:



The display is not over the suggested limit for any of the performance indicators. It should be relatively performant on a machine meeting the minimum hardware requirements.

Performance Indicator	Count	Suggested Limit
✓ Total objects	0	800
✓ Vector graphics objects	0	600
✓ Linked shapes	0	10
✓ Embedded shape sequences	0	70
✓ Groups & embedded shapes	0	70
✓ Rotated objects	0	70
✓ Scripts	1	100
✓ Color Breakpoints	0	100
✓ Data-bound objects	0	300
✓ ActiveX controls	0	30

Indicator description:

The processing required to call up a display depends primarily on the object count, almost regardless of the complexity of individual objects.
You can improve a display's performance by minimizing the number of objects it contains.
This also has usability advantages because, from a user's point of view, simple and uncluttered displays are generally easier to use.

Note: the count for each performance indicator includes the objects which make up the default shape of any embedded shape the display contains.

六、 结语

本文抛砖引玉的简单的介绍了一下 SymmetrE 中动画的应用，正所谓“条条大路通罗马”，我们可以用很多种方法来改进我的页面。希望本文能对您的工作有所帮助和启示，愿我们大家的项目界面越来越友好，给客户最好的体验。

13 Example Scenarios

These examples contain scripts that perform a wide range of realistic tasks.

Notes

- If you are viewing this online, you can copy example code and paste it into your own scripts—see "Copying Example Scripts" on page 353.
- If you have previously written scripts for the DSP version of Display, see "Scripting Differences between HMIWeb and DSP Displays" on page 194.

Miscellaneous Scenarios

Showing and Hiding Objects

Using Event Bubbling to Reduce Script Effort

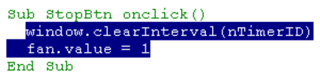
Copying Example Scripts

If you are viewing this online, you can save a lot of time by copying example scripts, pasting them into your own scripts, and then making the appropriate modifications.

The following example describes how to copy a block of code and paste it into one of your scripts.

To copy and paste a block of code:

1 In the help, drag diagonally across the code that you want to copy.



```
Sub StopBtn onclick()
  window.clearInterval(nTimerID)
  fan.value = 1
End Sub
```

2 Copy the code by pressing <Ctrl>+<C>.

3 Switch to the Script Editor.

4 Click where you want to insert the code (typically a blank line), and then press <Ctrl>+<V>.

Calling up a Faceplate on an Alarm Condition

Scenario

You want to call up a faceplate when a “PV HiHigh” alarm condition occurs on a particular point.

Solution

You add an alphanumeric (or indicator) to the display configure it as follows:

- On the Data tab, specify the point ID and set **Parameter** to `AlarmValue`
- On the Details tab, set **Display as** to `State Descriptor`
- On the Behaviors tab, select **Faceplate**

You write a script that simulates a click (in normal operation, clicking the alphanumeric would call up the faceplate) when the alarm condition occurs. You attach the script to the alphanumeric’s OnUpdate event.

Scripts

This script simulates an OnClick event when the specified alarm event occurs.

```
Sub alpha002_OnUpdate
  if alpha002.value = "PV HiHigh" Then
    alpha001.click
  End if
End Sub
```

Adding a ToolTip to an Object

Scenario

You want to make the displays easier to use by including standard DHTML ToolTips for important objects. (A ToolTip appears when a user moves the mouse pointer over an object.)

Solution

You write a script to set the object's title property whenever the value you are interested in changes.

Scripts

This example is for an object called "imgIndicator3".

```
Sub alpha1_onchange
  If alpha1.value = "open" Then
     shape1.title = "Pump is open"
  Else
     shape1.title = "Pump is closed"
  End If
End sub
```

Changing the Color of an Object in a Dynamic Shape

Scenario

You want to create a dynamic shape to represent a pump. You also want the color of a rectangle to change when the PV of the associated status point changes state.

Solution

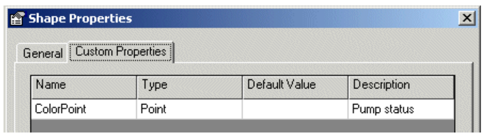
When you create the shape, in addition to defining a custom property (see “About Custom Properties” on page 33), you also:

- Add the custom property to the rectangle’s Script Data tab
- Write a script for the rectangle that controls its color based on the value of the custom property

Each time you insert the shape into a display, you assign the custom property to the appropriate point—in effect, the custom property represents the point. Consequently, when the point’s PV changes, so does the rectangle’s color.

To create the shape:

1. Create the new dynamic shape.
2. Open the Properties Window, select the Custom Properties tab and add a custom property, `ColorPoint`, and set its **Type** to `Point`.

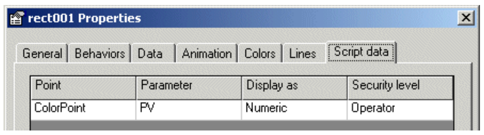


Name	Type	Default Value	Description
ColorPoint	Point		Pump status

3. Add the objects to required to create the pump.

Tip *Don’t forget to group the objects—this is a requirement for a dynamic shape.*

4. Select the rectangle whose color you want to control, open the Properties Window, click the Behaviors tab and select **Script Data**.
5. Click the Script Data tab and add the point you defined in step 2 as a custom property.


rect001 Properties

Point	Parameter	Display as	Security level
ColorPoint	PV	Numeric	Operator

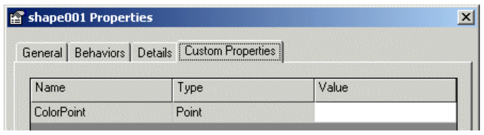
6 With the object still selected, open the Script Editor and, for the OnDataChange event, write the following script:

```
Sub rect001_OnDataChange
  ' Get the appropriate value
   PumpMode = rect001.GetDataValue "ColorPoint.PV"

  ' Now update the pump's appearance
  if PumpMode = 1 then
    Group.rect001.Color = vbRed
  else
    Group.rect001.Color = vbGreen
  end if
End Sub
```

Inserting the shape into displays

Each time you insert this shape into a display, you select the Custom Properties tab and type the ID of the appropriate point in **Value**.


shape001 Properties

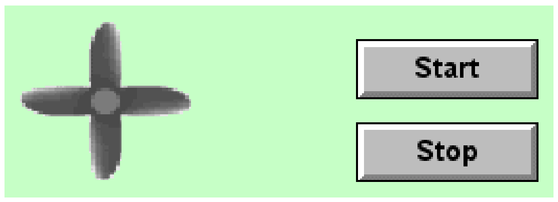
Name	Type	Value
ColorPoint	Point	

Creating an Animation with a Shape Sequence

Scenario

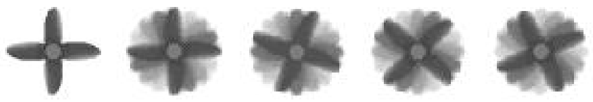
You have a fan that is controlled by **Start** and **Stop** buttons on a display, and want to animate the fan.

The following figure shows the fan and buttons in the display.



Solution

You use the following shape sequence with some scripts to create an animation.



The first (left-hand) shape represents the "off" state, and the other four shapes are used in the animation for the "on" state.

These shapes are bitmap images that were created in a drawing program and then pasted into the shape file. Creating the shapes this way made it easier to produce a more realistic effect.

Scripts

The following script is attached to the **Start** button's onclick event. This creates a 200 ms timer, which controls the speed of the animation.

```
Sub cmdStartBtn_onclick()
  intTimerID = window.setInterval ("myfunction()", 200)
End Sub
```

The function, “myfunction” is defined in the General section. In this example, the script steps through the four “on” shapes (values 2 to 5).

The General section must also contain the definition of the timer ID variable “intTimerID”, which is used when killing the timer.

```
Dim intTimerID
Function myfunction()
  If fan.value = fan.shapefile.numberOfShapes Then
    fan.value = 2
  Else
    fan.value = fan.value +1
  End If
End Function
```

The following script is attached to the **Stop** button’s onclick event. It stops the animation by calling the clearInterval method of the window object, which takes the timer ID as a parameter. The script also selects the “off” shape.

```
Sub cmdStopBtn_onclick()
  window.clearInterval(intTimerID)
  fan.value = 1
End Sub
```

Remarks

- When inserting the shape, you don’t have to specify the number of shapes, because this is controlled by your script.

Creating a Moving Object

Scenario

You have an object that represents a security gate and want it to “slide” open/closed when you issue an open/close command. (The combobox is used to issue the command.)

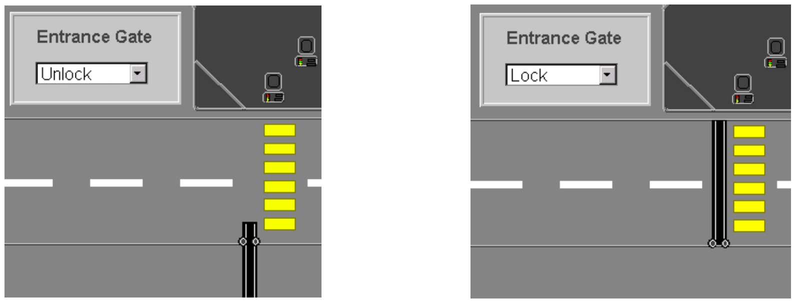


Figure 13.1 The Gate, in its Open and Closed Positions

Solution

You create the animation by moving the gate object in small steps at 100 ms intervals. (The step size and interval determine the “smoothness” of the animation.)

A timer is started in the combobox’s onchange event. Each time a timer event is generated, the `gateTimer` function moves the gate in the appropriate direction until it is either open or closed.

Scripts

The following script is attached to the combobox’s onchange event. This creates a 100 ms timer, which controls the speed of the animation.

Based on the current state of the combobox, either a `1` or a `0` is written into the variable `blnGateState`.

```
Sub cboGateControl_onchange
  intTimerID=window.setInterval("gateTimer()",100)
  If cboGateControl.value = "On" then
    blnGateState = 1
  Else
```

```
    blnGateState = 0
  End If
```

The following script is written into the General section.

The function `gateTimer` is called every 100 ms. It checks the position of the gate by looking at the variable `blnGateState`. If the gate is being opened, it shifts the gate object (`imgGate`) 50 pixels at a time in the open direction until it reaches the fully open position (`600`), and clears the timer. If the gate is being closed, it shifts the gate object 50 pixels at a time in the closed direction until it reaches closed position (`26`), clears the timer and makes sure that the gate is closed in the closed position `26`.

```
Dim blnGateState
Dim intTimerId
Function gateTimer()
  If blnGateState = 1 then
    If imgGate.style.pixelLeft < 600 Then
      imgGate.style.pixelLeft = imgGate.style.pixelLeft + 50
    Else
      window.clearInterval (intTimerId)
    End If
  Else
    If imgGate.style.pixelLeft > 26 then
      imgGate.style.pixelLeft = imgGate.style.pixelLeft - 50
    Else
      window.clearInterval (intTimerId)
      imgGate.style.pixelLeft = 26
    End If
  End If
End Function
```

Changing the Color to Indicate a Parameter's Value (Discrete Changes)

Scenario

You want an object's color to indicate the current value of the PV parameter of a status point.

Solution

You add the point's PV parameter to the object's Script Data tab so that the script can access the parameter. (See "Reading/Writing Point Parameters" on page 184.)

Scripts

You attach the following script to the object's OnDataChange Event event. In this example, the point is called "poista85", and it has two values: `0` and `1`.

```
Sub rect001_onchange
  If rect001.DataValue("poista85.PV") = 0
    rect001.fillColor=vbRed
Else
    rect001.fillColor=vbGreen
  End if
End sub
```

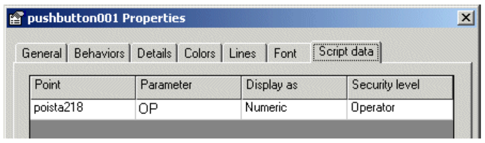
Using a Button to Start a Pump

Scenario

You want to use a button to start a pump. You also want to display a message box that requests the user to confirm the action before starting the pump.

Solution

You add parameter that controls the pump to the button's Script Data tab so that the script can access the parameter—in this example, the OP of a status point called “poista218”. (Remember to select **Script Data** on the Behaviors tab before you do this.)



Point	Parameter	Display as	Security level
poista218	OP	Numeric	Operator

Scripts

You attach this script to the button's onclick event. The message box contains **Yes** and **No** buttons.

```
Sub pushbutton001_onclick
  If pushbutton001.DataValue("poista218.OP") = 0 then
    If MsgBox("Are you sure that you want to start this
pump",vbYesNo) = vbYes Then
      pushbutton001.fillColor = vbGreen
      pushbutton001.DataValue("poista218.OP") = 1
    End If
  End If
  pushbutton002.fillColor = vbRed
End Sub
```

Creating a Jogging Control

Scenario

You want a “jogging” control for a motorized valve. (A jogging control gives the user accurate control over the valve’s position.)

Solution

You add a button and write scripts for its onmousedown and onmouseup events so that the motor runs while the left mouse button is held down.

You add the SP parameter of the valve’s control point (called “ValveControl”) to the button’s Script Data tab, so that you can write to the parameter. (See “Reading/Writing Point Parameters” on page 184.)

Scripts

This script is attached to the button’s onmousedown event.

```
pushbutton001.DataValue("ValveControl.SP") = 1
pushbutton001.fillColor = vbGreen
```

This script is attached to the button’s onmouseup event.

```
pushbutton001.DataValue("ValveControl.SP") = 0
pushbutton001.fillColor = vbRed
```

Performing an Action in Response to a Right-click

Scenario

You want to call up the previous display when the user right-clicks somewhere on the display.

Solution

You use the onmousedown event to detect which button was clicked. When the event is generated, the value of `window.external.button` is checked (a value of `2` indicates that the right button was clicked).

If this is the case, LRN 21 (the Server Display program) is requested with parameter `21` (the “page back” command).

Scripts

The script is attached to the onmousedown event for the Page.

```
If window.event.button = 2 Then
  window.external.requestserverlrn 21,21,0
End If
```

Remarks

- You cannot use the onclick event.

Checking the User's Security Level

Scenario

You want to prevent access to certain information if the person does not know the manager-level password.

Solution

You use a button to control the display of the information. The script first checks the current security level by reading the value of `window.external.SecurityLevel`.

If the current level is not manager, the user is prompted to for the manager-level password. This is done via the `window.external.executeoperatorcommand("psw")` command.

Scripts

This script is added to the button's onclick event.

```
Sub cmdDetails_onclick
  '0 = LVL 1
  '1 = LVL 2
  '2 = Operator
  '3 = Supervisor
  '4 = Engineer
  '5 = Manager
  Dim Level
  'Set Level variable on page loading
  Level = window.external.SecurityLevel
  'Check for presence of manager mode.
  If Level = 5 Then
    'display manager information in Message Zone.
    window.external.MessageZoneText = "Manager Logged On"
    ‘Display manager-only information.
    alpha5.style.visibility = "visible"
    .
    .
  Else
    Msgbox "Insufficient Security Clearance. Please log on as
a manager."
    window.external.executeoperatorcommand ("psw")
  End if
End Sub
```

Remarks

- Alternative ways of displaying the manager-only information include a message box and a popup.
- This scenario assumes Station-based security. If you have operator-based security and the user does not have manager-level access, an appropriate message might be: “You do not have sufficient security clearance”.

Starting an Application by Clicking a Button

Scenario

You want an application to start when a user clicks a button.

Solution

This example starts Microsoft Word using the CreateObject command. (The CreateObject command requires the object's `name` and `type`. In the case of Microsoft Word, these are `Word` and `Application`.)

Having started Word, its object model can be accessed via the variable, `msword`. In this example, its `visible` property is set to true to show the application.

Scripts

This Script is attached to button's onclick event.

```
Sub cmdNotepad_onclick
  Dim msword
  Set msword = CreateObject("Word.Application")
  msword.visible = true
End Sub
```

Controlling Station from Another Application

Scenario

You want to create a training support tool, written in Visual Basic, that presents trainee operators with a series of interactive flow charts they must work through. To improve usability, the support tool instructs Station to display the appropriate display at various stages in a procedure.

Solution

First, start a new Visual Basic project as a standard exe. Then create two command buttons called, **cmdStart** and **cmdPage80**. The **cmdStart** button starts Station and **cmdPage80** button calls up the required display (page 80).

Scripts

The code for the project's General section.

```
Public objStationApp As Object
Option Explicit
```

The the script for the onclick event of **cmdStart**. Station is started using the CreateObject function. Any parameters that need to be passed to Station are done through the object variable `objStationApp`.

```
Private Sub cmdStart_Click()
  Set objStationApp = CreateObject("Station.Application")
End Sub
```

The script for the onclick event of **cmdPage80**. A command is sent to Station to call up page 80.

```
Private Sub cmdPage80_Click()
  objStationApp.CurrentPage = 80
End Sub
```

Responding to Station Events from Another Application

Scenario

You want to create a visible, separate application that responds to events generated by Station. This application is invoked from a display.

Solution

Write a Visual Basic ActiveX EXE that is invoked within Station via page scripts.

To prepare the ActiveX control:

1. Start a new Visual Basic project as an ActiveX EXE.
2. Reference the Station Type Library.

   The ActiveX EXE requires knowledge of the Station Application object, its methods, properties and events. To expose these you need to add the “Station Type Library (Version 2)” to the project references. Station needs to be installed on your computer for its type library to be visible.
3. Add a Station Application variable to your ActiveX EXE.

   This variable stores the Station Application object. Declare the variable “withevents” to allow handling of any application events generated by Station.
4. Add a method to allow Station to connect to your ActiveX EXE.

   This method is called within script to pass in a pointer to the Station Application object. Your ActiveX EXE stores this point in a global variable for future use.
5. Write Event Handlers for the events you are interested in.

   ```
   Private Sub objStation_OnConnect()
     'Insert code here to handle the OnConnect event
   End Sub
   ```
6. Add scripts to your page to invoke your ActiveX EXE

   These scripts create your ActiveX EXE, pass it a reference to the Station Application object, and, if required, instruct the ActiveX EXE to display any forms it contains.

Scripts

The code for the general section of the class module for steps 3, 4 and 6 is shown below for an ActiveX EXE that contains a form called UIForm.

```
Option Explicit
Dim WithEvents m_objStation As Station.Application2
' ***************************************************
' This method MUST exist in this module to use Station
Application
Public Sub SetStationObject(ByRef objStation As
Station.Application2)
```

```
    Set m_objStation = objStation
    Set UIForm.objStation = objStation
End Sub
' ****************************************************
' ****************************************************
' This method MUST exist in this module to load the form via
script
Public Sub Load()
    UIForm.Visible = True
End Sub
' ****************************************************
```

Once your ActiveX EXE is compiled you can write your page script that creates and loads it.

```
Set MyControl=CreateObject("MyControl.clsMyControl")
MyControl.SetStationObject(window.external)
MyControl.Load()
```

Plotting a Parameter Value as it Changes

Scenario

You want to plot changes to a point parameter value as its value changes. (In this example, the PV parameter of a point called “sinewave”.)

Solution

In addition to adding a chart to the display, you also to add a simple object—in this example, a rectangle—and enable the Script Data behavior so that you can add the point parameter to the rectangle’s Script Data tab. (See “Reading/Writing Point Parameters” on page 184.)

You attach a script to the page’s onpagecomplete event, which sets up the chart’s attributes/properties and adds the plot.

In the script for the rectangle’s ondatachange event, you construct one array for the x values and another array for the y values. Because this is done on a per-update basis, only one value is added to each array at each update. The x value is set using Visual Basic’s Now method, which returns the current date/time. The y value is set to the parameter’s current value.

The constructed arrays are then passed as parameters to the AddPlotData method. Note that the plot ID (which is the return value from the AddNewPlot method) is a global variable which is declared in the General section of the page script.

Scripts

The following two variables are defined in the General section:

```
dim g_plotID1
dim g_PlotAdded
```

This script is attached to the page’s onpagecomplete event.

```
Sub Page_onpagecomplete
  g_PlotAdded = false

  'Construct the plot
  dim XData(1), YData(0)
  XData(0) = Now
  XData(1) = DateAdd("s", 1, 0) 'period between updates -> 1
sec

  'Get the first data update value from the rectangle object
  YData(0) = rect001.dataValue("sinewave.PV")
  dim YRangeMin, YRangeMax, XRangeMin, XRangeMax
  YRangeMin = 0
  YRangeMax = 100
  XRangeMin = Now
  XRangeMax = DateAdd("s", 50, XRangeMin)

  'Add the plot to the chart
  g_plotID1 = chart001.AddNewPlot("Pump1 Speed", YRangeMin,
YRangeMax, "RPM", XRangeMin, XRangeMax, "Time")

  g_PlotAdded = true

  'Change the plot color
  chart001.SetPlotColor g_plotID1, vbBlue

  'Now add the first portion of data
  chart001.AddPlotData g_plotID1, YData, XData
End Sub
```

This script is attached to the rectangle's OnDataChange event.

```
Sub rect001_ondatachange
  if g_PlotAdded = true then
    'Create X and Y data arrays
    dim XData(1), YData(0)
    XData(0) = Now
    XData(1) = DateAdd("s", 1, 0) 'period between updates ->
1 sec
    YData(0) = me.dataValue("sinewave.PV")

    chart001.AddPlotData g_plotID1, YData, XData
  End If
End Sub
```

Remarks

The data array created by the rectangle's OnDataChange event is destroyed if the user calls up another display, so a new array is created each time a user calls up the display.

Showing and Hiding Objects

Scenario

You have a display that is difficult to use because it shows too much information. You want to simplify the display by hiding non-critical information unless specifically requested by the user.

Solution

You use a button to toggle the visibility of some objects within a group. (For example, alphanumerics containing non-critical information appear when the button is clicked once and disappear when it is clicked a second time.)

Scripts

The script for the button's onclick event. It controls the visibility of the relevant child objects. (One object, `alpha5`, is used to check whether the objects are visible.)

```
Sub cmdPushbutton_onclick
  If imgGroup1.all("alpha5").style.visibility = "visible"
Then
    imgGroup1.all("alpha5").style.visibility = "hidden"
    imgGroup1.all("alpha6").style.visibility = "hidden"
    imgGroup1.all("alpha7").style.visibility = "hidden"
    .
    .
  Else
    imgGroup1.all("alpha5").style.visibility = "visible"
    imgGroup1.all("alpha6").style.visibility = "visible"
    imgGroup1.all("alpha7").style.visibility = "visible"
    .
    .
  End If
End Sub
```

Using Event Bubbling to Reduce Script Effort

Scenario

You want the mouse pointer to change to a hand symbol over all pushbutton objects.The page that contains a large number of objects, many of which are scripted almost identically and you want to avoid having to write the same script for each pushbutton object.

Solution

Instead of scripting the onmouseover and onmouseout events for all pushbuttons on the page, script these events on the page object. In the event handler, check the ID of the object generating the event and, if it indicates the object is a pushbutton, change the mouse pointer accordingly.

Scripts

The script for the page's onmouseover event.

```
Sub Page_onmouseover
  If (Left(window.event.srcElement.id, 10) = "pushbutton")
Then
   window.event.srcElement.style.cursor = "hand"
  End If
End Sub
```

The script for the page's onmouseout event.

```
Sub Page_onmouseout
  If (Left(window.event.srcElement.id, 10) = "pushbutton")
Then
    window.event.srcElement.style.cursor = "auto"
  End If
End Sub
```

Remarks

- For information on style.cursor see the books/Web sites listed in "Useful Web Sites and Reference Books" on page 190.